Views on Behaviour Protocols and Their Semantic Foundation

Sebastian Bauer and Rolf Hennicker

Institut für Informatik
Ludwig-Maximilians-Universität München

3rd Conference on Algebra and Coalgebra in Computer Science

September 10, 2009

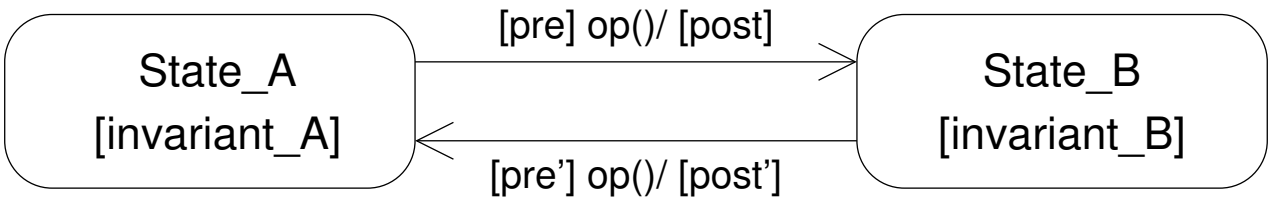
Motivation

Integration of

- control-flow based protocols, like process algebras (e.g. CSP, CCS), or finite state machines
- data-flow based specifications, e.g. invariants, pre- and postconditions (e.g. OCL)

Literature:

- CSP-OZ, Symbolic Transition Systems (STS), LOTOS, etc.
- UML protocol state machines



Goal: Formal semantics, compatibility and compositionality results

Meaning of UML Protocol State Machines ?

UML specification:

- "The protocol state machine defines all allowed transitions for each operation."
- "It specifies ... the allowed call sequences on the classifier's operations."
- The protocol state machine "describes valid interactions."

$\Longrightarrow$ No single interpretation of a protocol!

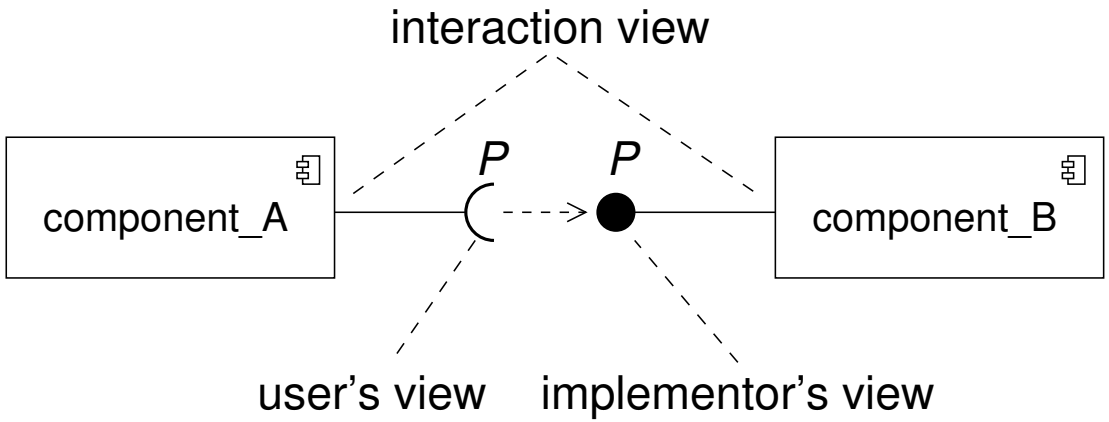
Meaning of UML Protocol State Machines ?

UML specification:

- "The protocol state machine defines all allowed transitions for each operation." $\Longrightarrow$ implementor's view
- "It specifies ... the allowed call sequences on the classifier's operations." $\Longrightarrow$ user's view
- The protocol state machine "describes valid interactions." $\Longrightarrow$ interaction view

$\Longrightarrow$ No single interpretation of a protocol!

Protocol Semantics

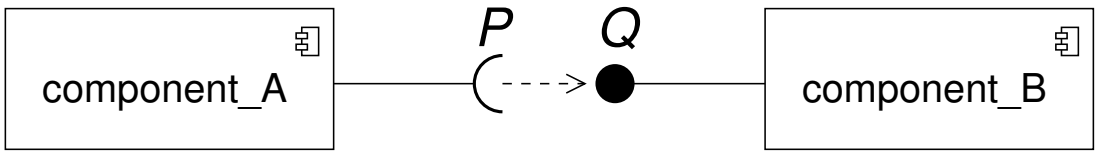


We propose a model-theoretic approach:

- $[\![P]\!]_{impl}$ = class of all correct implementation models
- $[\![P]\!]_{use}$ = class of all correct user models
- $[\![P]\!]_{iact}$ = class of all correct interaction models

Desired Result

Compatibility of the semantic views with model composition and protocol composition

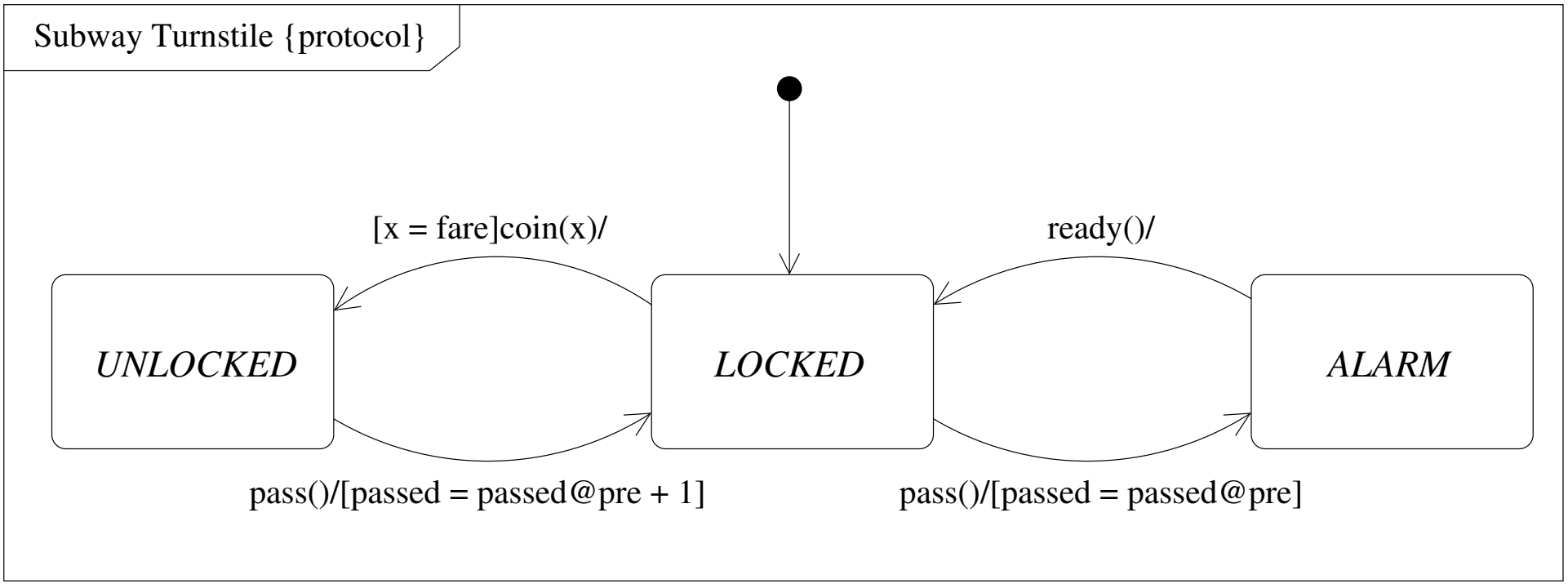


$$M \in [\![P]\!]_{use},\ N \in [\![Q]\!]_{impl},\ \text{and}\ P \sim Q \Longrightarrow M \otimes N \in [\![P \boxtimes Q]\!]_{iact}$$

Outline

Outline

Example: Subway Turnstile



General Assumptions: Observer Signatures, Data States

1. Observer signature Σ_{Obs} (e.g. a set of state variables of visible type)
2. Domain of data states $\mathit{DState}(\Sigma_{\mathrm{Obs}})$ (e.g. class of Σ_{Obs}-algebras)
3. Unary state predicates $\mathcal{S}(\Sigma_{\mathrm{Obs}})$, satisfaction relation

$$\sigma; \rho \models \varphi$$

   where $\sigma \in \mathit{DState}(\Sigma_{\mathrm{Obs}})$, ρ valuation, $\varphi \in \mathcal{S}(\Sigma_{\mathrm{Obs}})$

4. Binary transition predicates $\mathcal{T}(\Sigma_{\mathrm{Obs}})$, satisfaction relation

$$\sigma, \sigma'; \rho \models \pi$$

   where $\sigma, \sigma' \in \mathit{DState}(\Sigma_{\mathrm{Obs}})$, ρ valuation, $\pi \in \mathcal{T}(\Sigma_{\mathrm{Obs}})$

Protocol Signatures and Σ-Protocols

Protocol Signature:

$$\Sigma = (\Sigma_{\mathrm{Obs}}, Op)$$

- Σ_{Obs}: observer signature
- Op: set of operations

Σ-Protocol: labelled transition system (LTS)

$$P = (S, s_0, PL, \Delta)$$

- PL: protocol labels $[\varphi]op/[\pi]$ with $op \in Op$, precondition $\varphi \in \mathcal{S}(\Sigma_{\mathrm{Obs}})$, postcondition $\pi \in \mathcal{T}(\Sigma_{\mathrm{Obs}})$

Protocol Composition

Protocol composition of two Σ-protocols P, Q:

$$P \boxtimes Q = (S_P \times S_Q, (s_{P,0}, s_{Q,0}), PL, \Delta_{P \boxtimes Q})$$

- $\Delta_{P \boxtimes Q}$ is the least relation satisfying:
  if $(s_P, s_Q) \in S_P \times S_Q$,

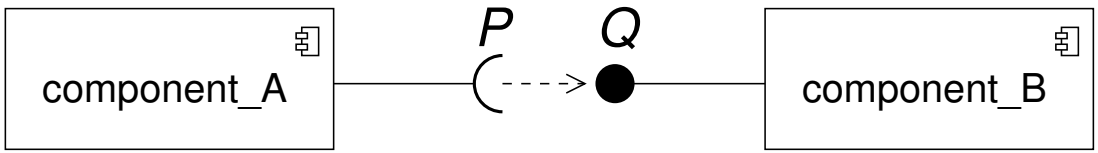
$$(s_P \xrightarrow{[\varphi]op/[\pi]} s'_P) \in \Delta_P \text{ and } (s_Q \xrightarrow{[\varphi']op/[\pi']} s'_Q) \in \Delta_Q$$

  then

$$\left((s_P, s_Q) \xrightarrow{[\varphi \wedge \varphi']op/[\pi \wedge \pi']} (s'_P, s'_Q) \right) \in \Delta_{P \boxtimes Q}$$

Desired Result (Again)

Compatibility of the semantic views with model composition and protocol composition



$$M \in [\![P]\!]_{use},\ N \in [\![Q]\!]_{impl},\ \text{and}\ P \sim Q \Longrightarrow M \otimes N \in [\![P \boxtimes Q]\!]_{iact}$$

Outline

Σ-Implementation Models

- Idea:
  Abstract representation of a program realising the operations of Σ
- Formally: input-enabled, deterministic LTS

$$N = (C \times Q, (c_0, \sigma_0), L, \Delta)$$

state space : $C \times Q$, with C a set of control states,
$Q \subseteq DState(\Sigma_{\mathrm{Obs}})$ a set of data states

transitions :

$$(c, \sigma) \xrightarrow{?op(\rho)} (c', \sigma')$$

$op \in Op$, ρ a valuation of op's input variables

Implementation Semantics $[\![P]\!]_{impl}$

A Σ-implementation model N (with states $C \times Q$) is a correct implementation of a Σ-protocol P (with states S), if the transitions in P are simulated in M, i.e. there exists a relation $\lesssim_{impl} \subseteq S \times (C \times Q)$ such that

(1) $s_0 \lesssim_{impl} (c_0, \sigma_0)$

(2) for all $s \lesssim_{impl} (c, \sigma)$, for all valuations ρ:

if $(s \xrightarrow{[\varphi]op/[\pi]} s') \in \Delta_P$ and $\sigma; \rho \models \varphi$

then there exists $\left((c, \sigma) \xrightarrow{?op(\rho)} (c', \sigma')\right) \in \Delta_N$ such that:

- $s' \lesssim_{impl} (c', \sigma')$,
- $\sigma, \sigma'; \rho \models \pi$.

Definition (Implementation Semantics)

$[\![P]\!]_{impl} = \{\Sigma\text{-impl. } M \mid M \text{ is a correct implementation of } P\}$

Σ-User Models

- Idea:
  Abstract representation of a program where the actions are invocations of required operations
- Formally: LTS

$$M = (C, c_0, UL, \Delta)$$

state space : C a set of control states

transitions :

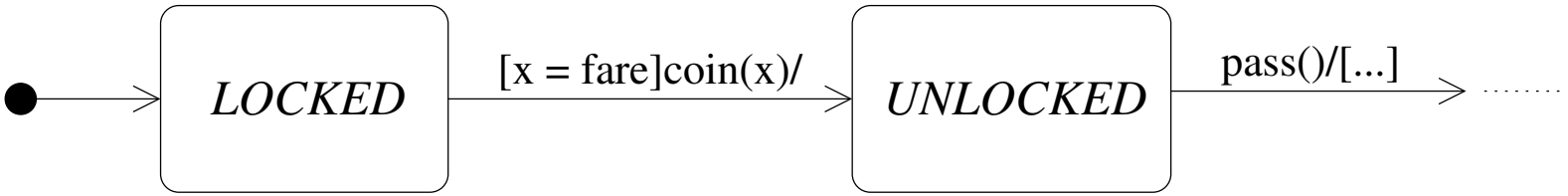
$$c \xrightarrow{[\sigma]!op(\rho)} c'$$

Intuition:

if observed data state equals $\sigma \in DState(\Sigma_{\mathrm{Obs}})$, then send out $op(\rho)$

User Models: Example

Protocol transitions:



(1) Program `int f = getFare(); coin(f); pass();`
corresponds to the set of transitions

$$start \xrightarrow{[\sigma]!coin(\sigma(fare))} paid \xrightarrow{[\sigma']!pass()} passed$$

for all data states $\sigma, \sigma' \in DState(\Sigma_{\mathrm{Obs}})$

(2) `coin(5); pass();` corresponds to

$$start \xrightarrow{[\sigma]!coin(5)} paid \xrightarrow{[\sigma']!pass()} passed$$

for all data states $\sigma, \sigma' \in DState(\Sigma_{\mathrm{Obs}})$

User Semantics $[\![P]\!]_{use}$

A Σ-user model M (with states C) is a correct user of a Σ-protocol P (with states S), if the transitions in M are simulated in P, i.e. there exists a relation $\lesssim_{use} \subseteq \big(C \times \mathcal{P}(DState(\Sigma_{\mathrm{Obs}}))\big) \times S$ such that

(1) $(c_0, DState(\Sigma_{\mathrm{Obs}})) \lesssim_{use} s_0$

(2) for all $(c, T) \lesssim_{use} s$:

if $(c \xrightarrow{[\sigma]!op(\rho)} c') \in \Delta_M$ and $\sigma \in T$

then there exists $(s \xrightarrow{[\varphi]op/[\pi]} s') \in \Delta_P$ such that:

- $\sigma; \rho \vDash \varphi$,
- $(c', T') \lesssim_{use} s'$ where $T' = \{\sigma' \in DState(\Sigma_{\mathrm{Obs}}) \mid \sigma, \sigma'; \rho \vDash \pi\}$.

Definition (User Semantics)

$[\![P]\!]_{use} = \{\Sigma\text{-user } N \mid N \text{ is a correct user of } P\}$

(Synchronous) Model Composition

A Σ-user model M and Σ-implementation model N can be composed to

$$M \otimes N$$

by synchronising corresponding actions:

$$(c_M \xrightarrow{[\sigma]!op(\rho)} c'_M) \in \Delta_M \quad \big((c_N, \sigma) \xrightarrow{?op(\rho)} (c'_N, \sigma')\big) \in \Delta_N$$

$$\Longrightarrow \quad ((c_M, c_N), \sigma) \xrightarrow{op(\rho)} ((c'_M, c'_N), \sigma') \in \Delta_{M \otimes N}$$

Σ-Interaction Models & Semantics

- Idea: describes the (runtime) interactions between user and implementation models when communicating through a synchronous communication channel
- Formalised again as LTS, with state space $C \times Q$, labels $op(\rho)$
- A Σ-interaction model K is a correct interaction of a Σ-protocol P, if the transitions in K are simulated in P, i.e. there exists a relation $\lesssim_{iact} \subseteq (C \times Q) \times S$ such that
  (1) $(c_0, \sigma_0) \lesssim_{iact} s_0$
  (2) for all $(c, \sigma) \lesssim_{iact} s$: if $((c, \sigma) \xrightarrow{op(\rho)} (c', \sigma')) \in \Delta_K$
  then there exists $(s \xrightarrow{[\varphi]op/[\pi]} s') \in \Delta_P$ such that $(c', \sigma') \lesssim_{iact} s'$, $\sigma; \rho \vDash \varphi$, and $\sigma, \sigma'; \rho \vDash \pi$

Definition (Interaction Semantics)

$[\![P]\!]_{iact} = \{\Sigma\text{-interact. } K \mid K \text{ is a correct interaction of } P\}$

Σ-Interaction Models & Semantics

- Idea: describes the (runtime) interactions between user and implementation models when communicating through a synchronous communication channel
- Formalised again as LTS, with state space $C \times Q$, labels $op(\rho)$
- A Σ-interaction model K is a correct interaction of a Σ-protocol P, if the transitions in K are simulated in P, i.e. there exists a relation $\lesssim_{iact} \subseteq (C \times Q) \times S$ such that
  (1) $(c_0, \sigma_0) \lesssim_{iact} s_0$
  (2) for all $(c, \sigma) \lesssim_{iact} s$: if $\left((c, \sigma) \xrightarrow{op(\rho)} (c', \sigma')\right) \in \Delta_K$
  then there exists $(s \xrightarrow{[\varphi]op/[\pi]} s') \in \Delta_P$ such that $(c', \sigma') \lesssim_{iact} s'$, $\sigma; \rho \vDash \varphi$, and $\sigma, \sigma'; \rho \vDash \pi$

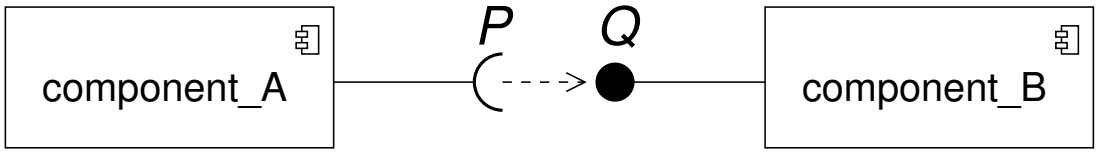
Definition (Interaction Semantics)

$[\![P]\!]_{iact} = \{\Sigma\text{-interact. } K \mid K \text{ is a correct interaction of } P\}$

Outline

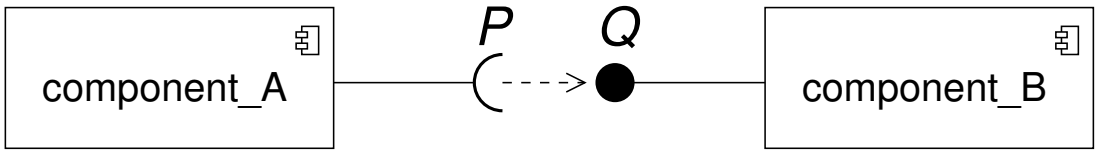
Desired Result (Again)

Compatibility of the semantic views with model composition and protocol composition



$$M \in [\![P]\!]_{use}, N \in [\![Q]\!]_{impl}, \text{ and } \underbrace{P \sim Q}_{!} \Longrightarrow M \otimes N \in [\![P \boxtimes Q]\!]_{iact}$$

Protocol Compatibility



- Semantic compatibility: $P \sim Q$ is defined by

$$[\![P]\!]_{impl} \supseteq [\![Q]\!]_{impl} \quad \text{and} \quad [\![P]\!]_{use} \subseteq [\![Q]\!]_{use}$$

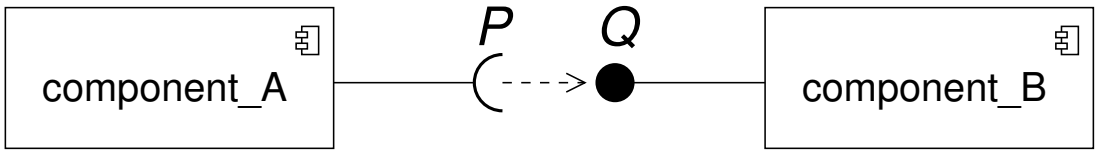
- Syntactic compatibility: $P \sim_S Q$ is defined by a relation between P and Q, where transitions of P are simulated in Q such that
  - preconditions in P may be weakened in Q, and
  - postconditions in P may be strengthened in Q.

Theorem 1

For Σ-protocols P and Q, if $P \sim_S Q$ then $P \sim Q$.

Final Result

Compatibility of the semantic views with model composition and protocol composition



Theorem 2

$M \in [\![P]\!]_{use}, N \in [\![Q]\!]_{impl},$ and $P \sim Q \Longrightarrow M \otimes N \in [\![P \boxtimes Q]\!]_{iact}$

Outline

Conclusion

- Model-theoretic foundation of a (subset of) UML protocol state machines
- Protocol compatibility and compositionality results relating (synchronous) model composition and protocol composition
- Extension to a component model with I/O-protocols for ports with required and provided interfaces *(FACS'09)*
- Extension to asynchronously communicating components *(future work)*

Conclusion

- Model-theoretic foundation of a (subset of) UML protocol state machines
- Protocol compatibility and compositionality results relating (synchronous) model composition and protocol composition
- Extension to a component model with I/O-protocols for ports with required and provided interfaces *(FACS'09)*
- Extension to asynchronously communicating components *(future work)*

Thank you very much for your attention!