

Composition of interactive Web services based on controller synthesis and modal logic

Philippe Balbiani, Fahima Cheikh, Guillaume Feuillade
Institut de recherche en informatique de Toulouse

Outline

- Service oriented computing
- Composition problem
- Controller synthesis
- Modal logic
- Composition of interactive Web services
- Variants and open problems

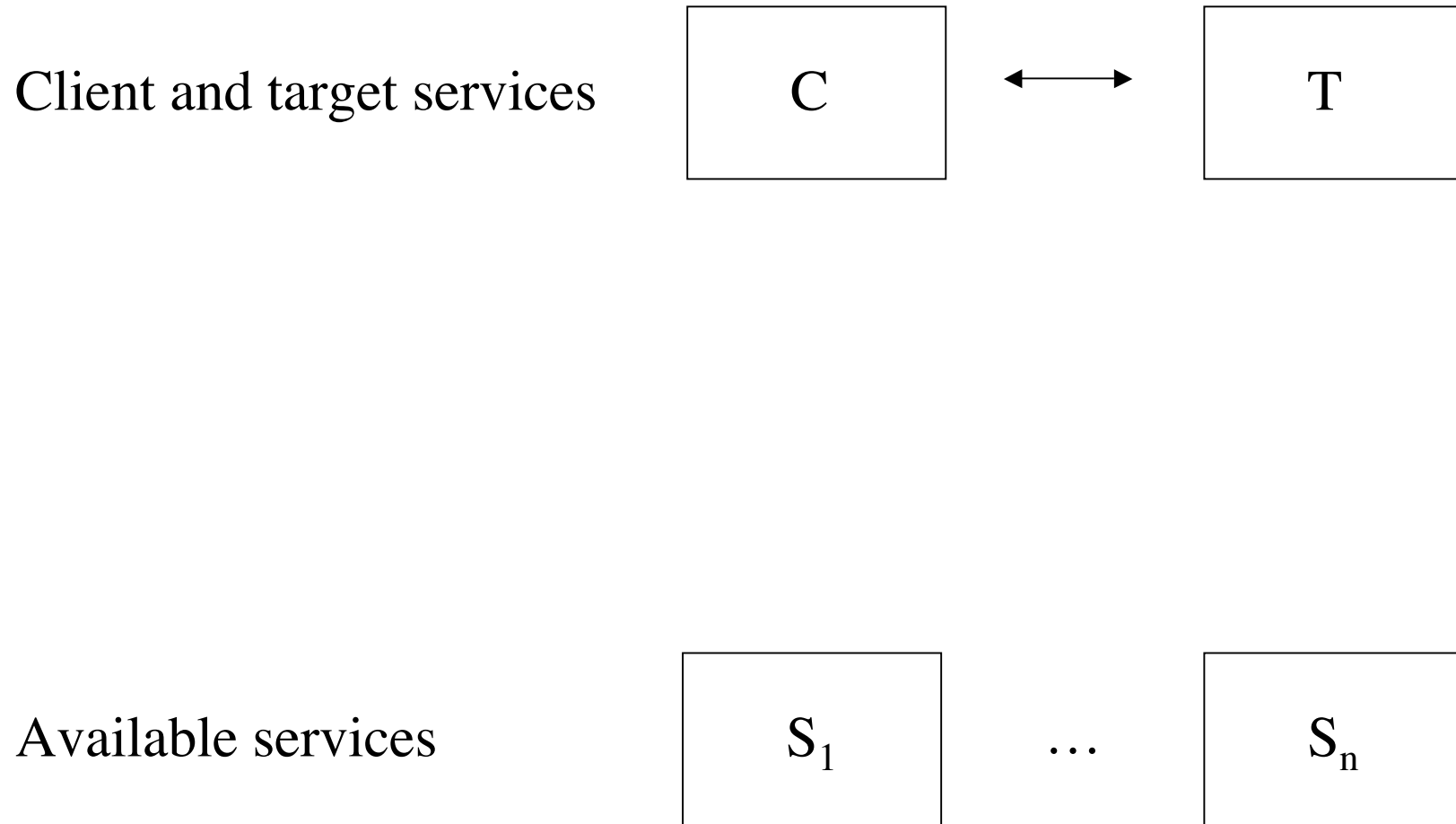
Outline

- **Service oriented computing**
- Composition problem
- Controller synthesis
- Modal logic
- Composition of interactive Web services
- Variants and open problems

Service oriented computing

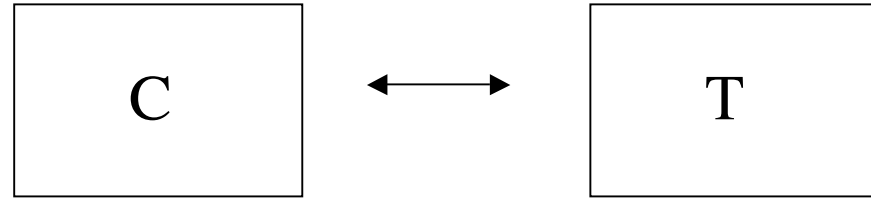
- Services are autonomous platform-independent computational elements that can be described, published, discovered, orchestrated and programmed using XML artifacts for the purpose of developing massively distributed interoperable applications
- Although there can be some value in accessing a single service through a semantically well-founded interface, the greater value is clearly derived through enabling a flexible composition of services
- Service composition concepts involve enough intricacy so as to attract considerable interest and to demand a careful analysis of the underlying principles

Service oriented computing

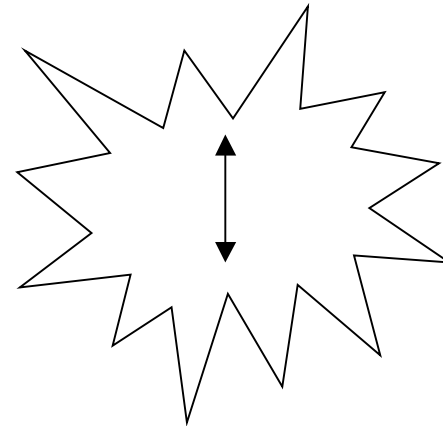


Service oriented computing

Client and target services



Communications : $C \leftrightarrow S_i$

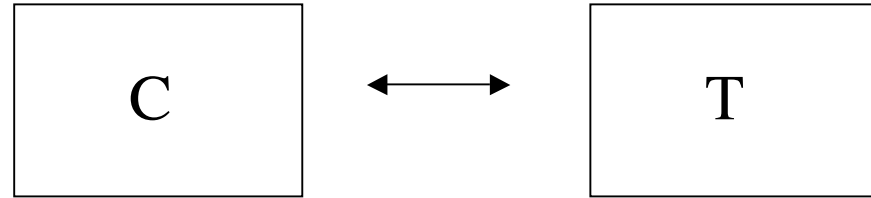


Available services

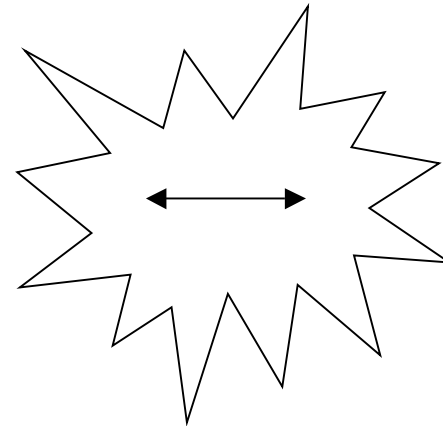


Service oriented computing

Client and target services



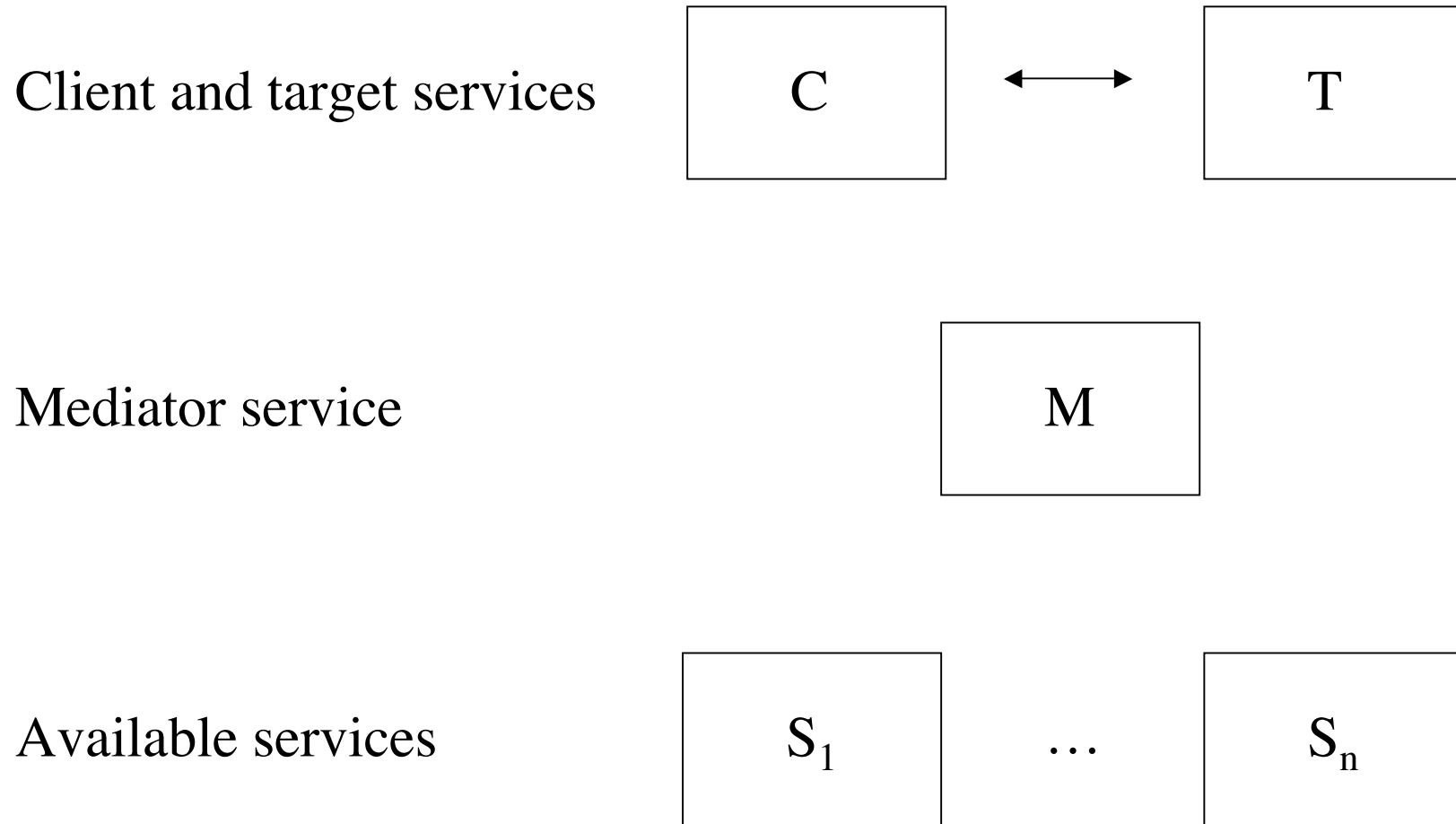
Communications : $S_i \leftrightarrow S_j$



Available services



Service oriented computing

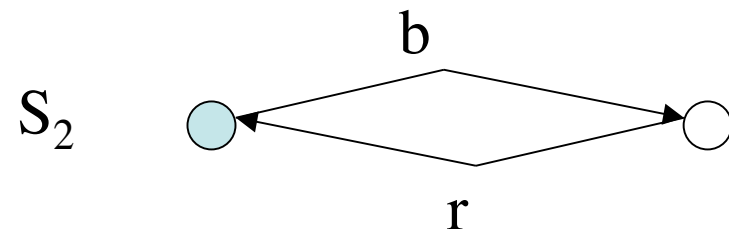
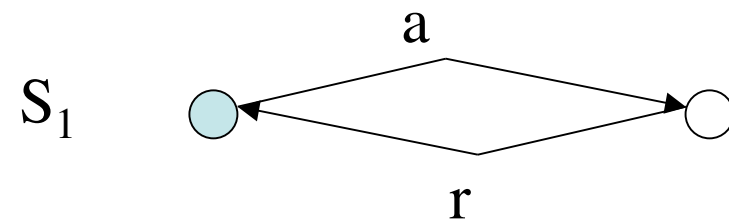
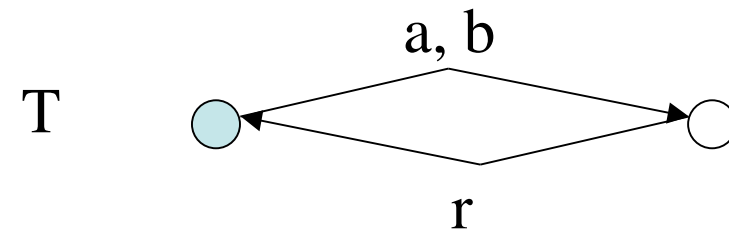


Service oriented computing

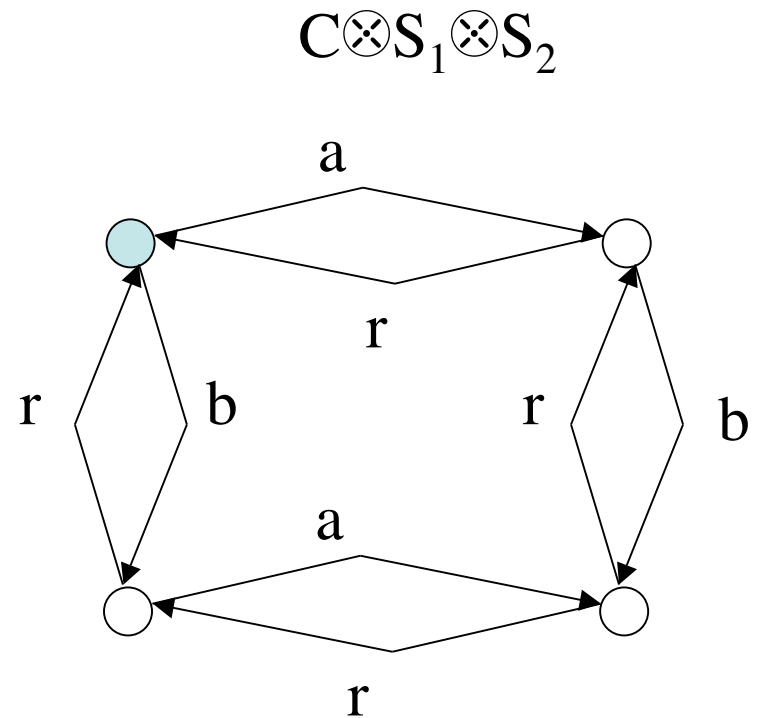
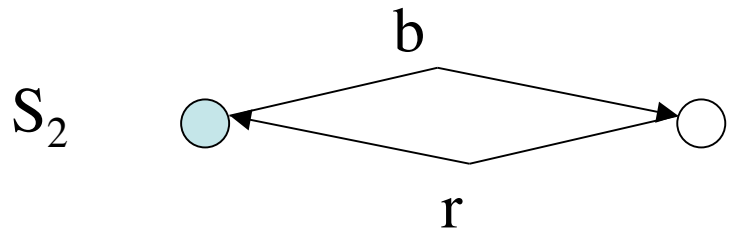
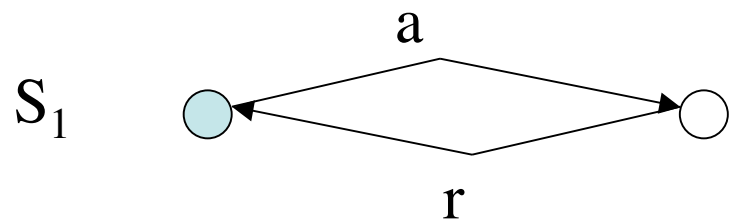
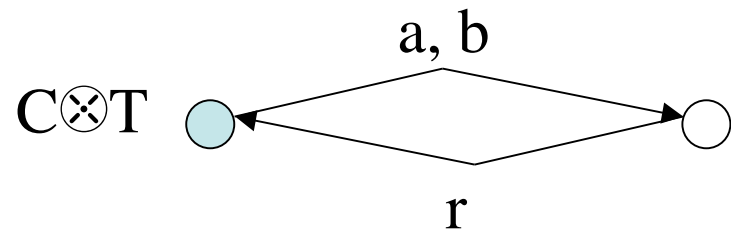
- Community ontology
 - Set of actions
- Client service
 - Service coordinating the target service
- Target service
 - Service using the actions of the community ontology
- Available services
 - Services using the actions of the community ontology
- Mediator service
 - Service coordinating the client service and the available services

Service oriented computing

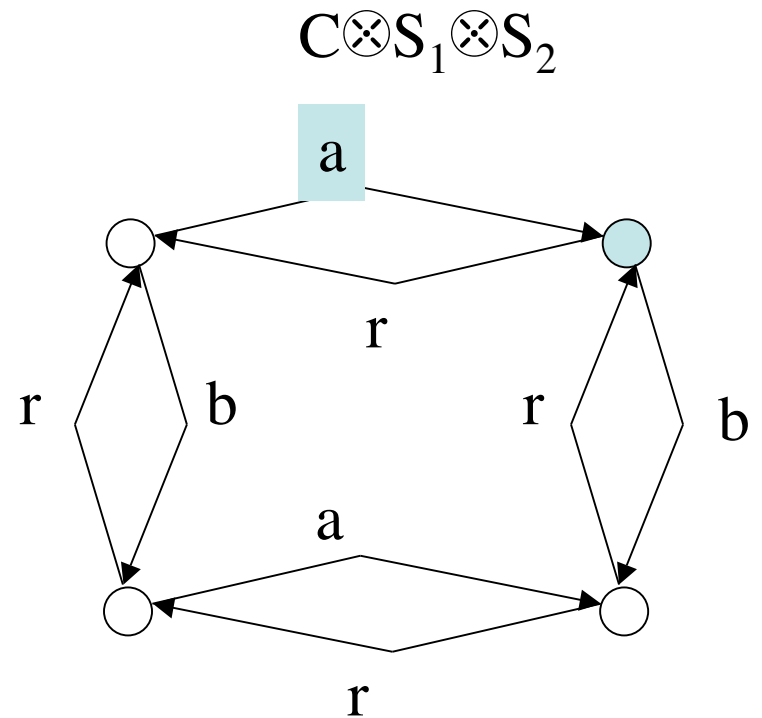
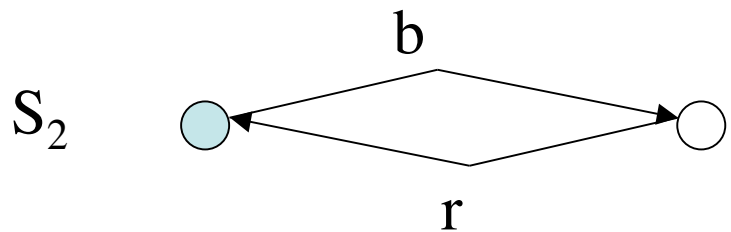
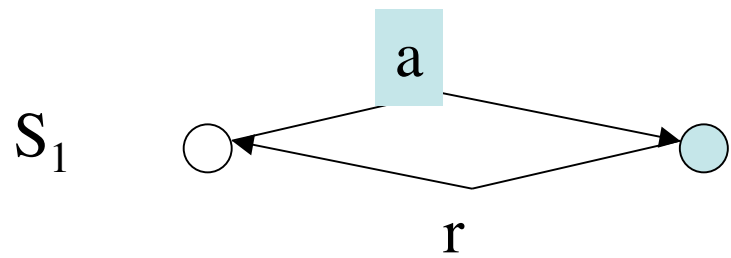
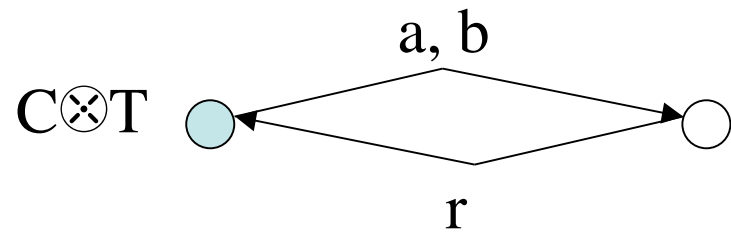
- Community ontology
 - $\Sigma = \{a,b,r\}$
- Client service
 - $C = (Q_C, q_{0C}, \delta_C)$
- Target service
 - $T = (Q_T, q_{0T}, \delta_T)$
- Available services
 - $S_1 = (Q_1, q_{01}, \delta_1)$
 - $S_2 = (Q_2, q_{02}, \delta_2)$



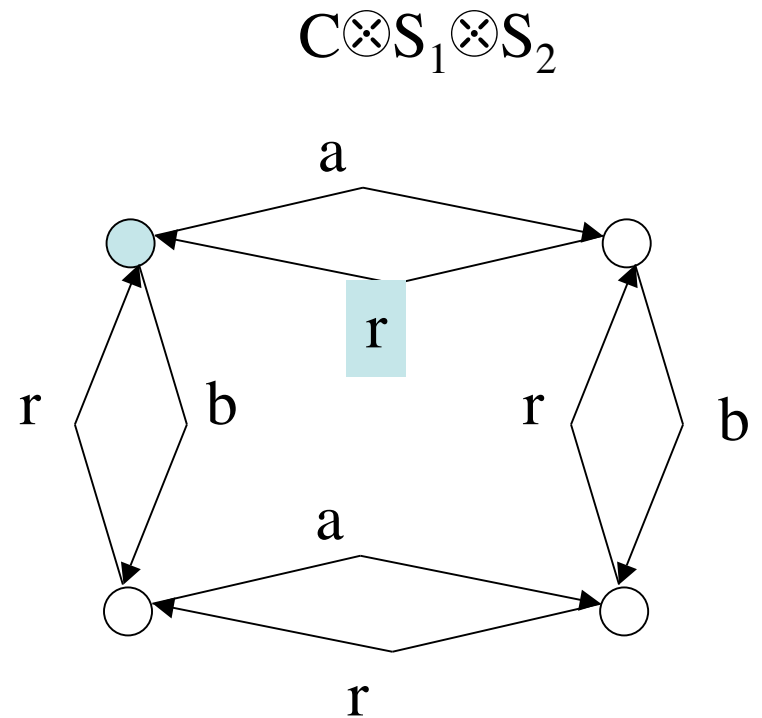
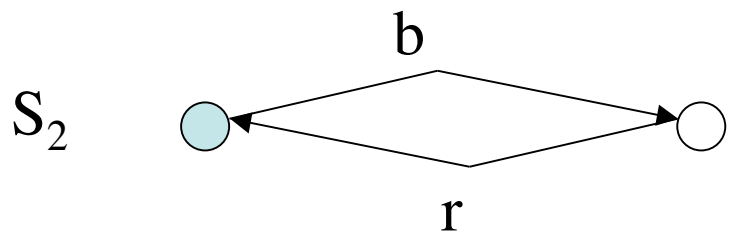
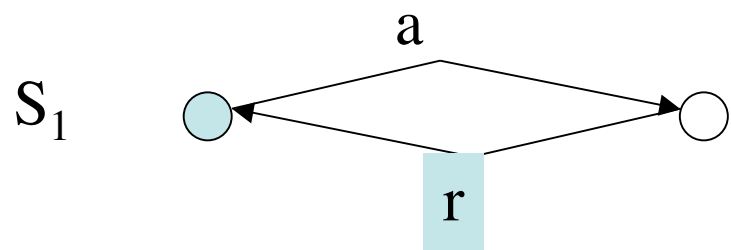
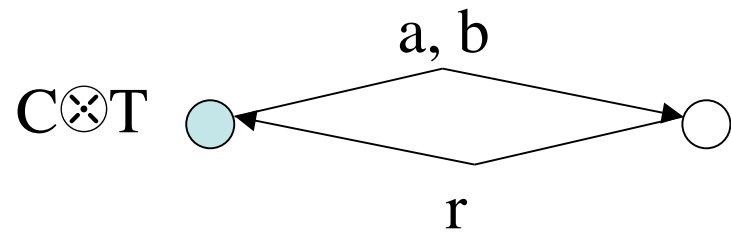
Service oriented computing



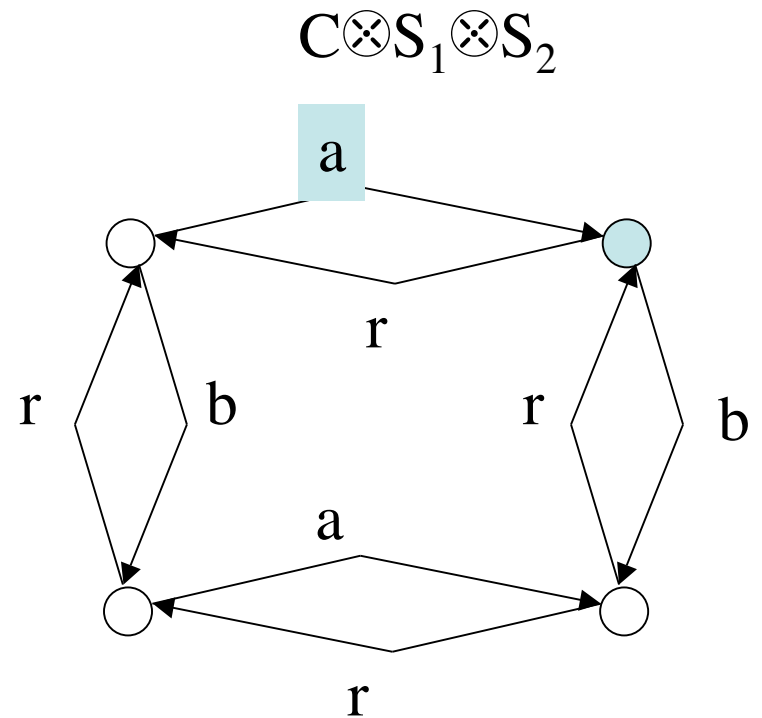
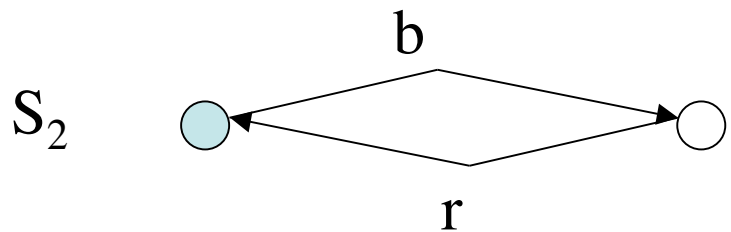
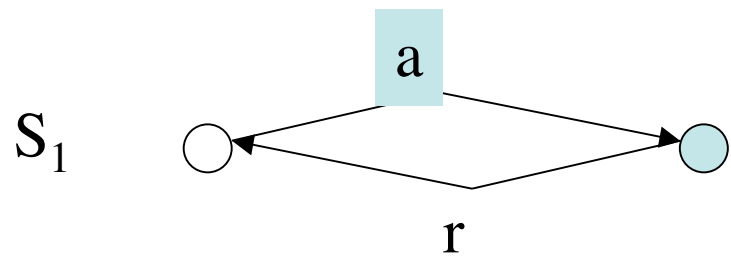
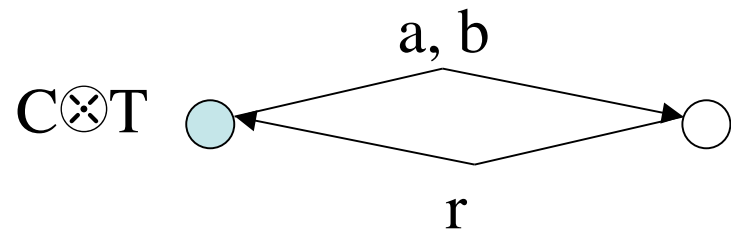
Service oriented computing



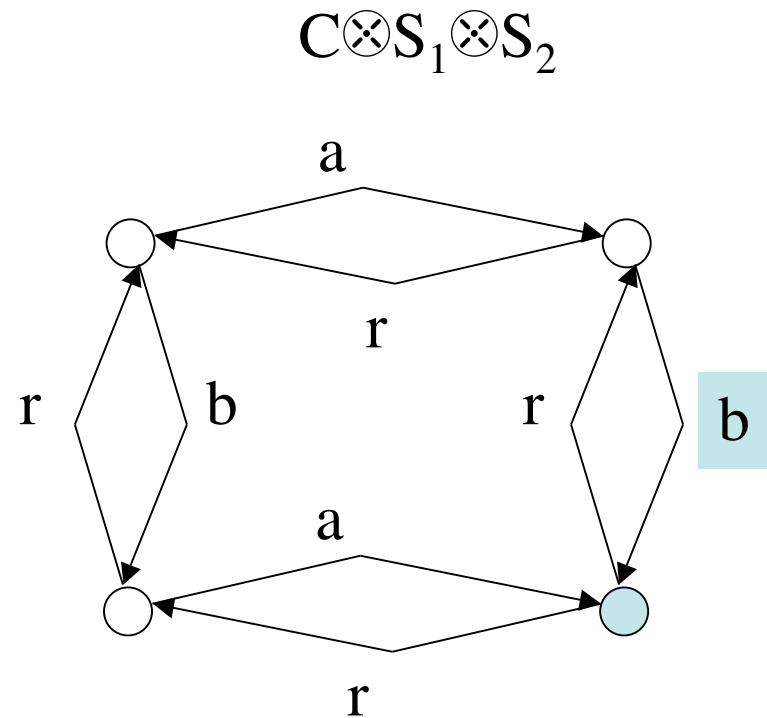
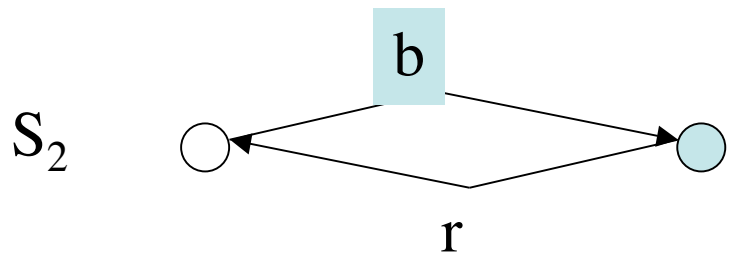
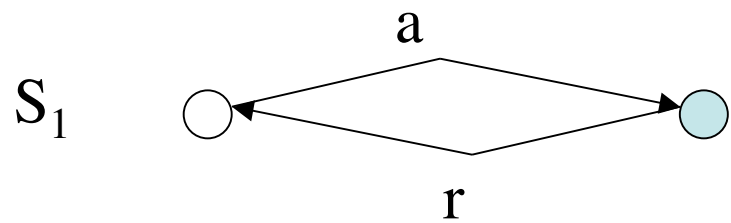
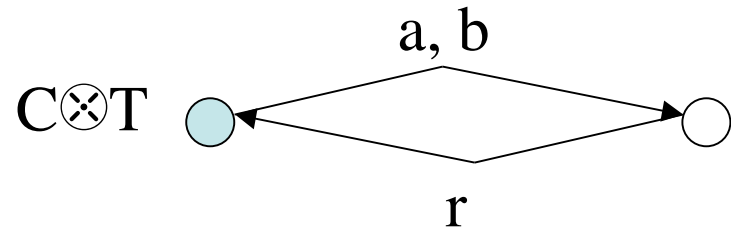
Service oriented computing



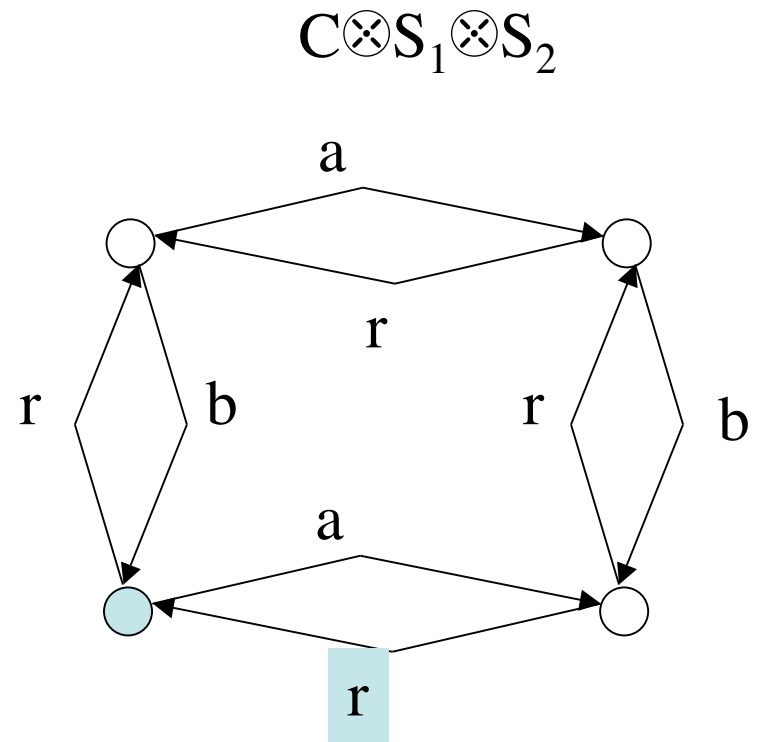
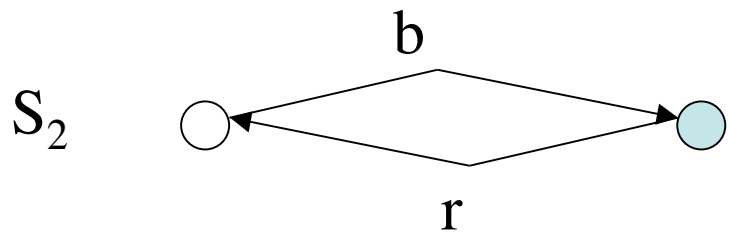
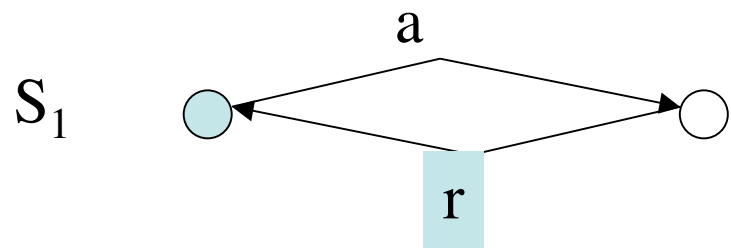
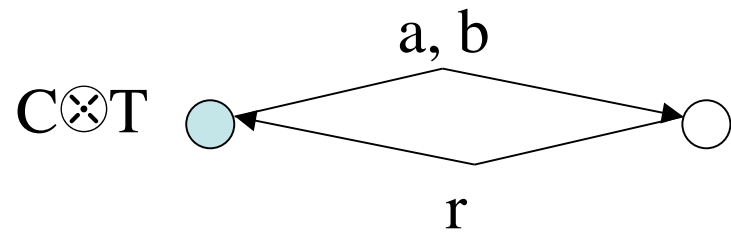
Service oriented computing



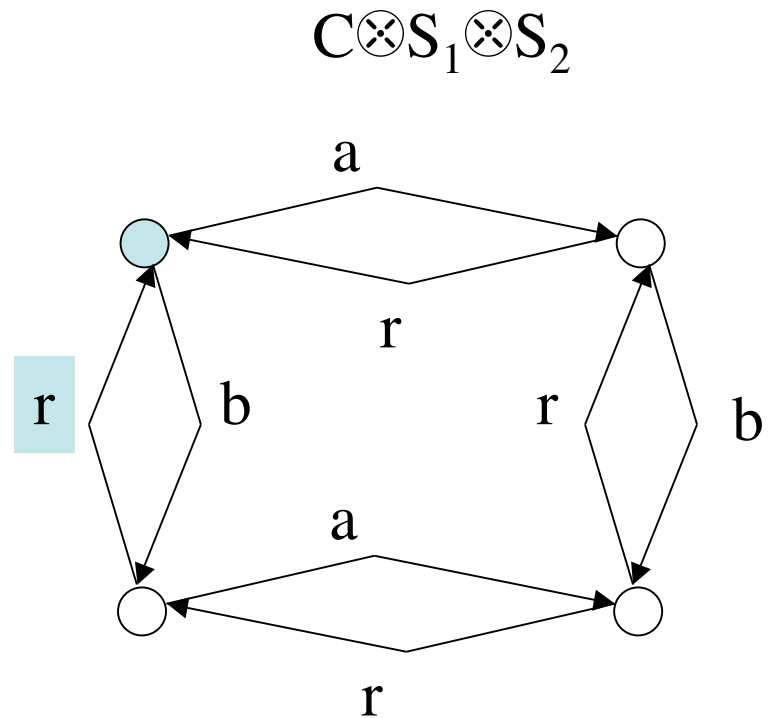
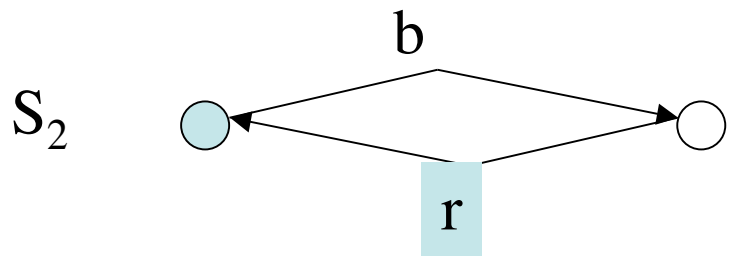
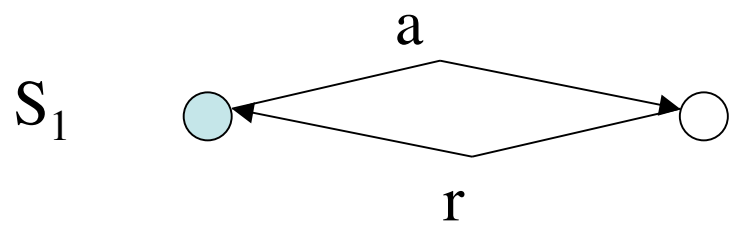
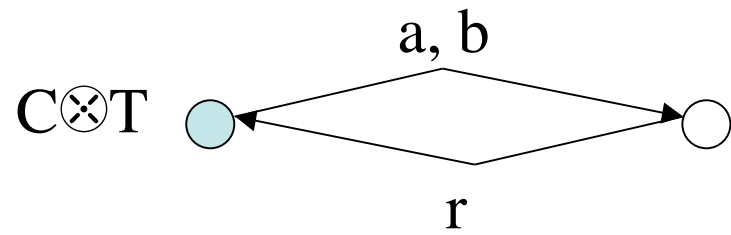
Service oriented computing



Service oriented computing

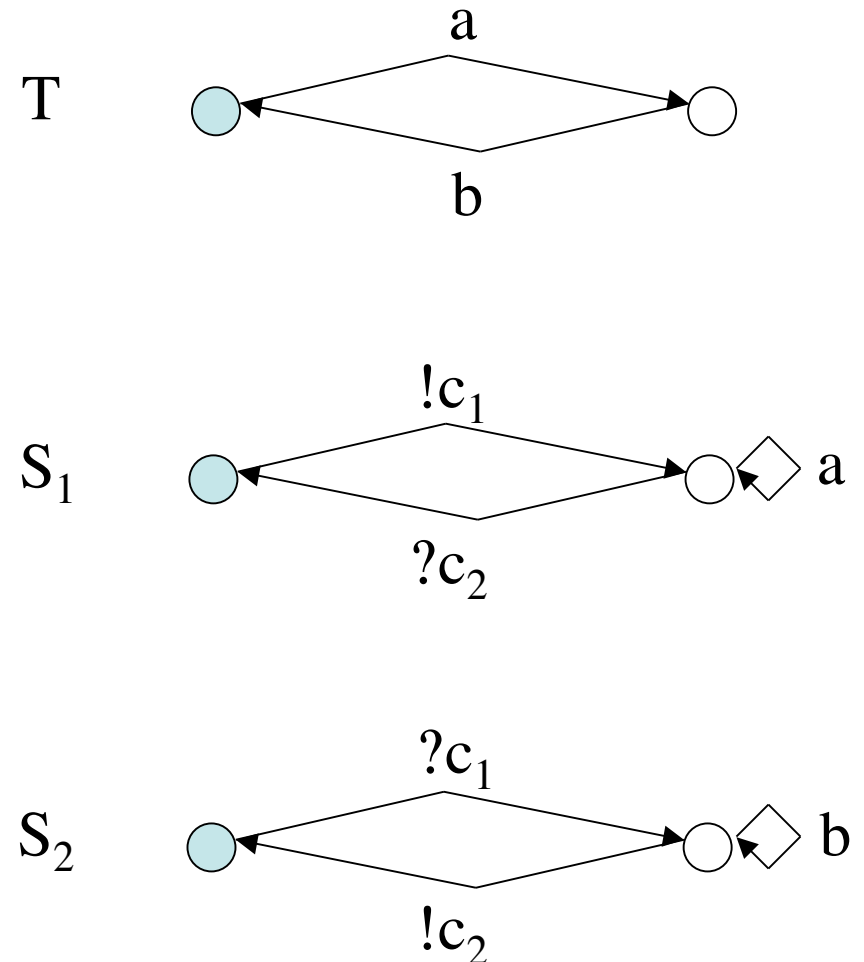


Service oriented computing

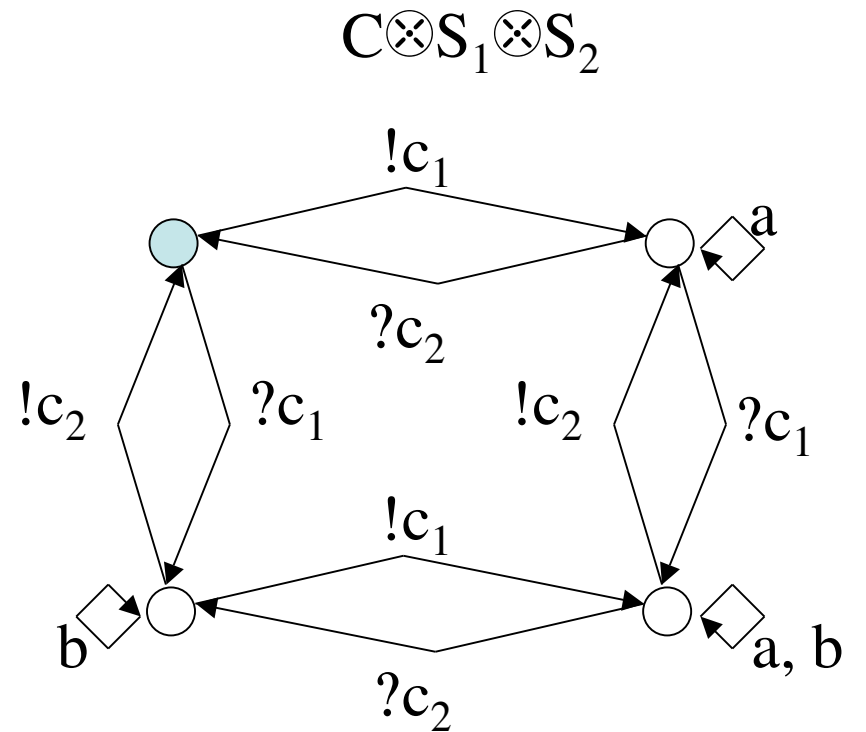
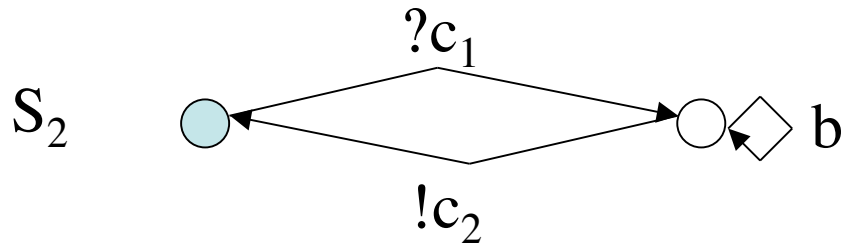
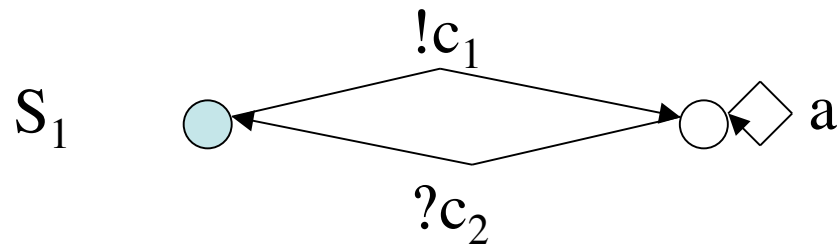
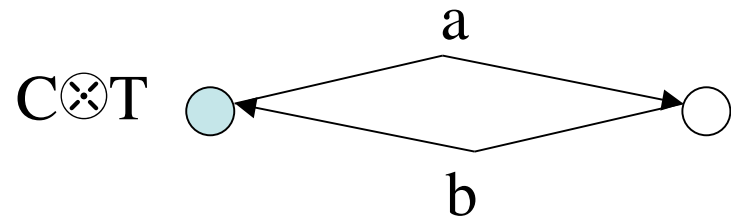


Service oriented computing

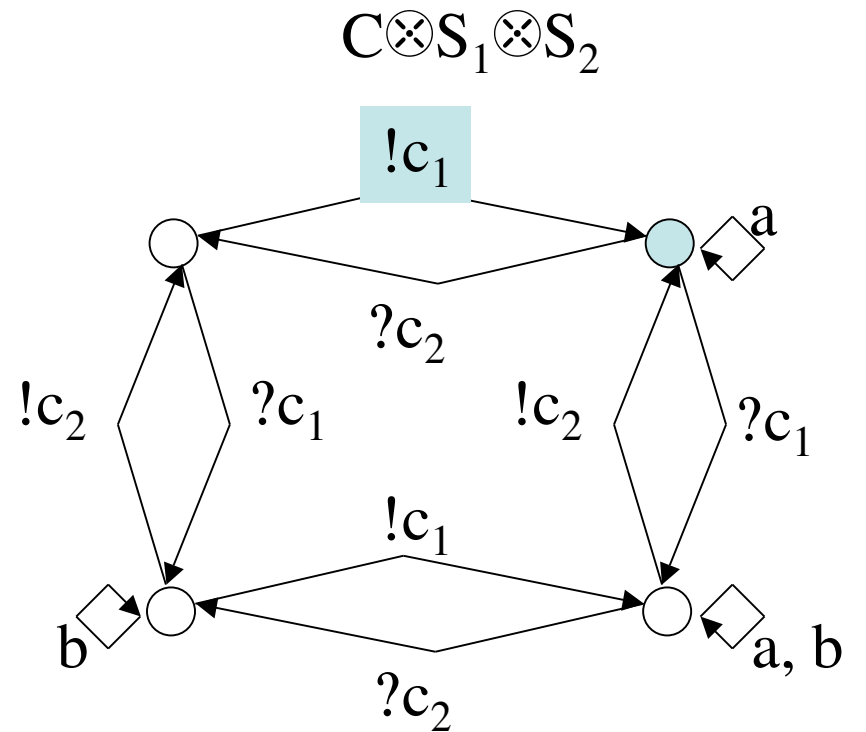
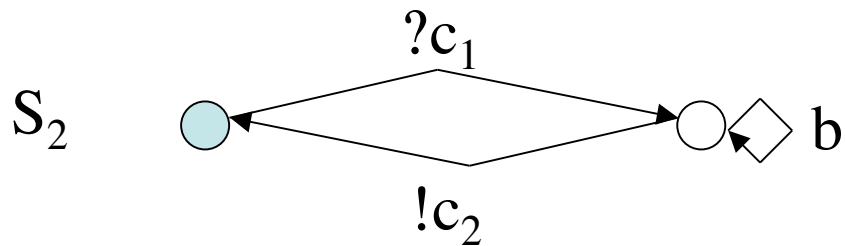
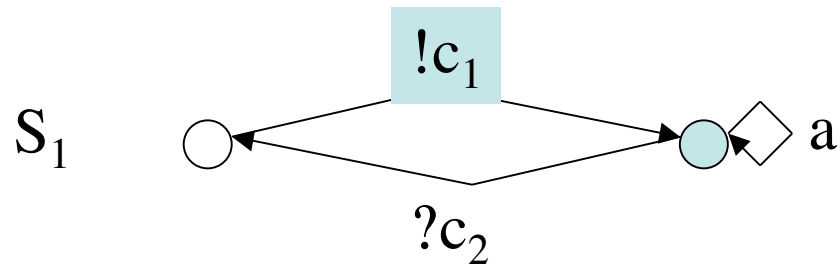
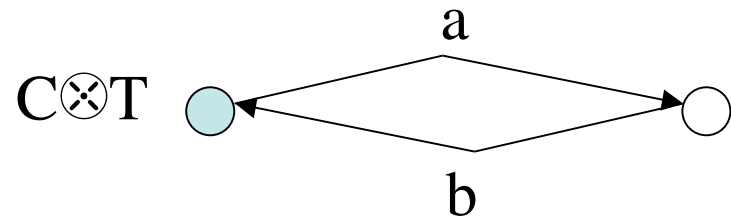
- Community ontology
 - $\Sigma = \{a,b\}$, $Ch = \{c_1,c_2\}$
- Client service
 - $C = (Q_C, q_{0C}, \delta_C)$
- Target service
 - $T = (Q_T, q_{0T}, \delta_T)$
- Available services
 - $S_1 = (Q_1, q_{01}, \delta_1)$
 - $S_2 = (Q_2, q_{02}, \delta_2)$



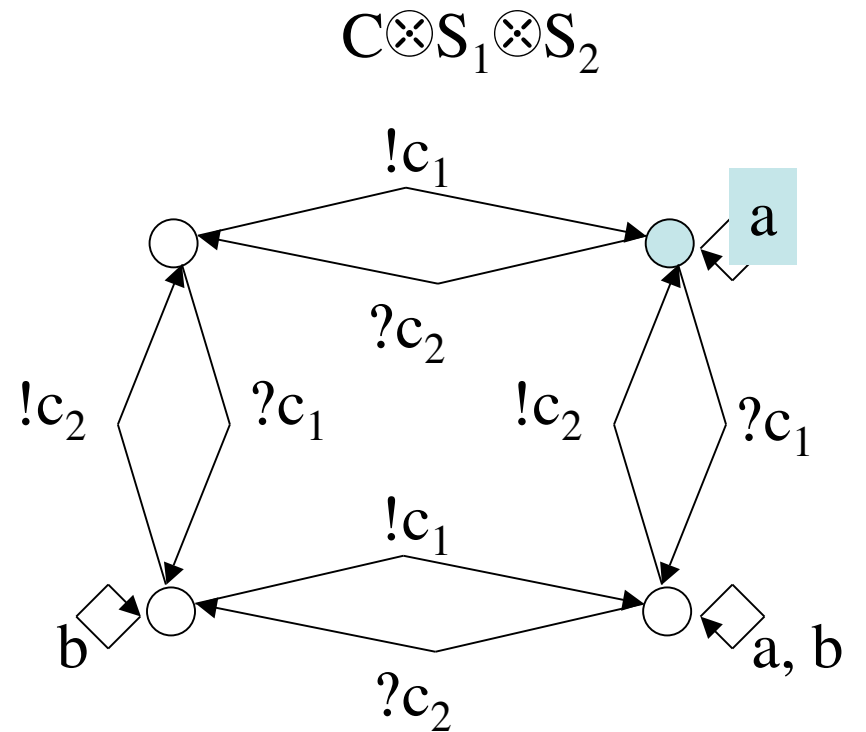
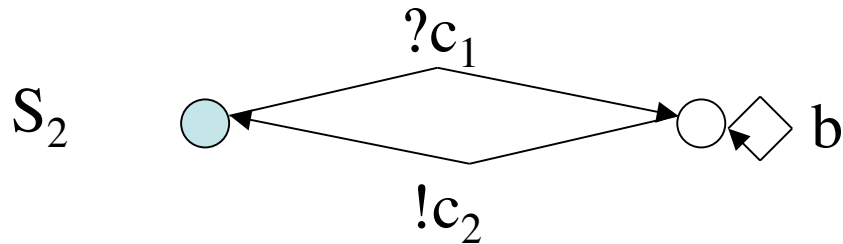
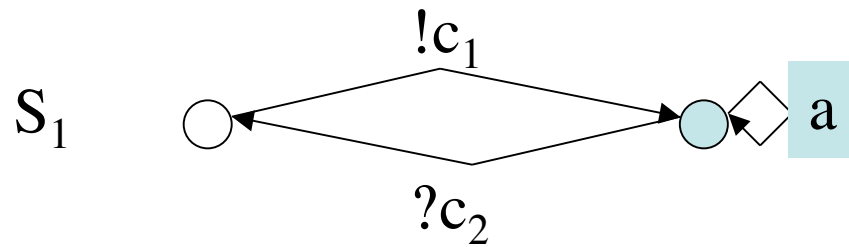
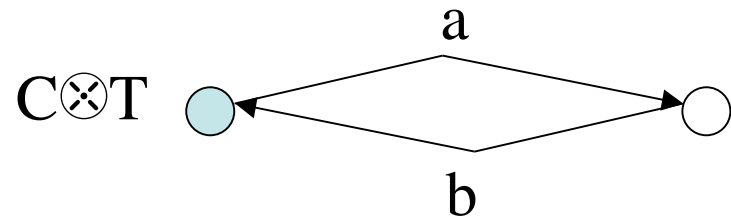
Service oriented computing



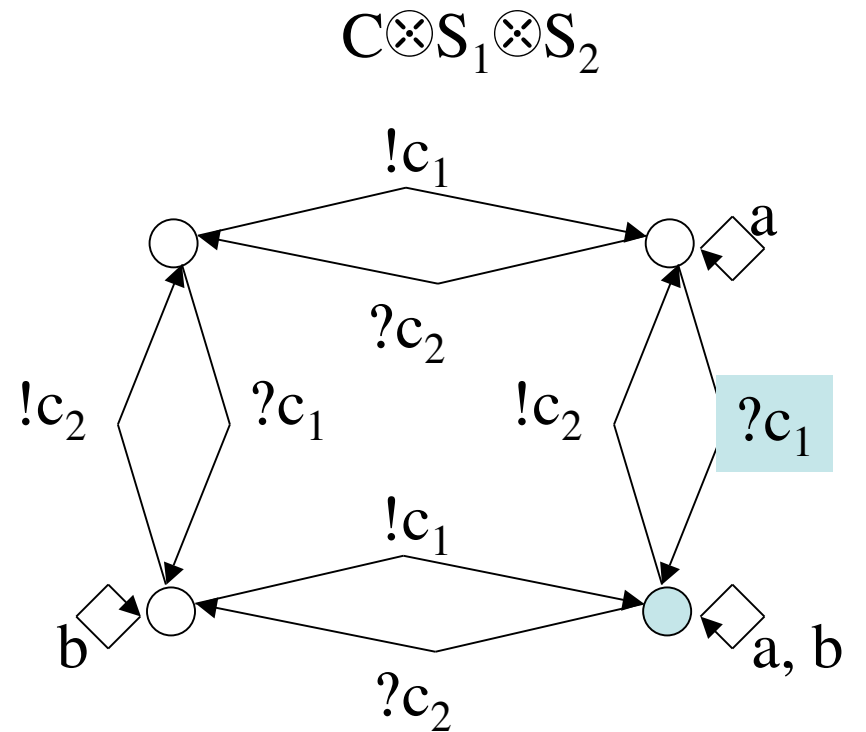
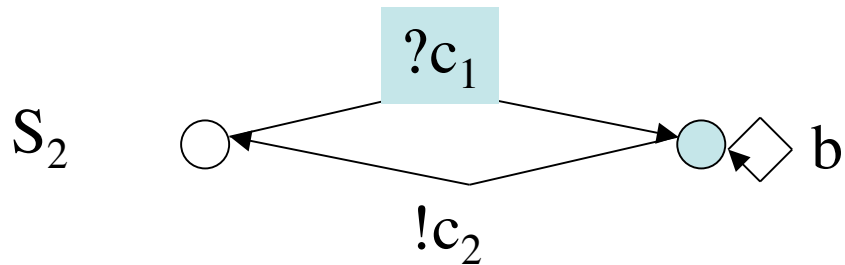
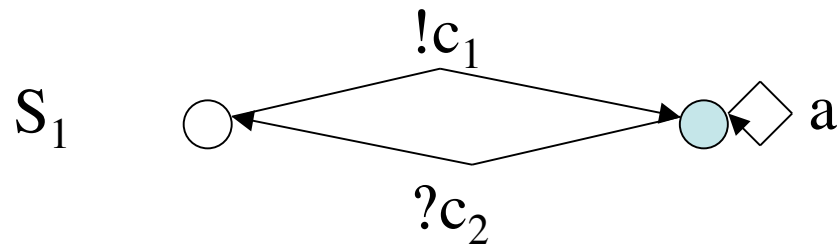
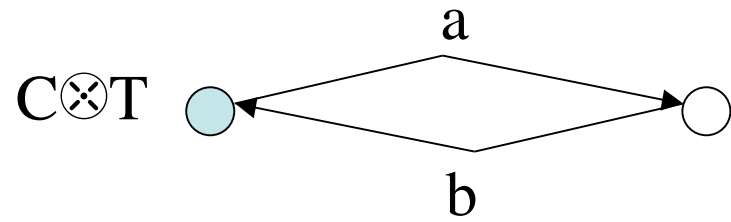
Service oriented computing



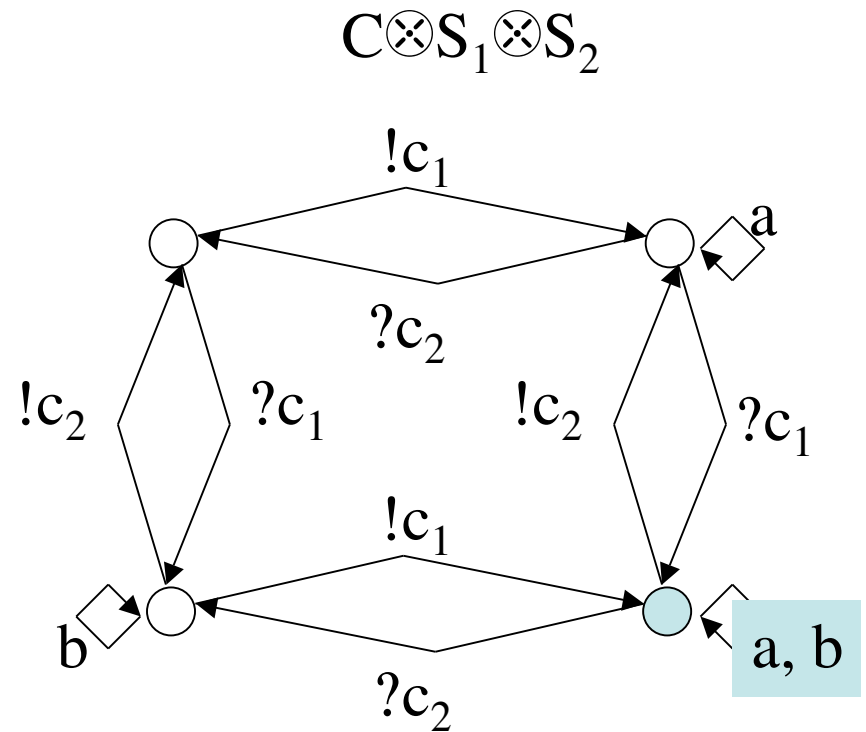
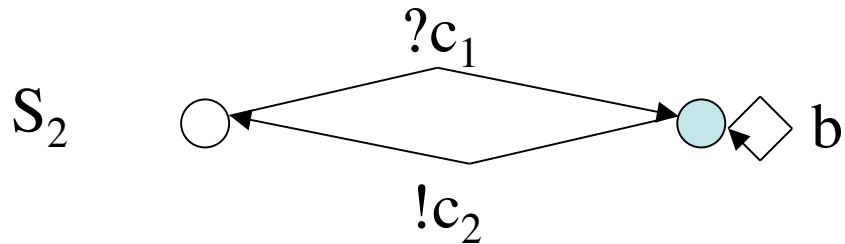
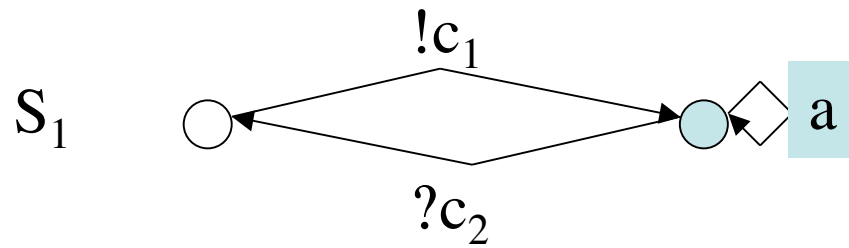
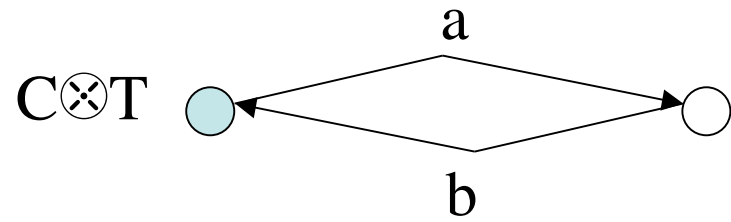
Service oriented computing



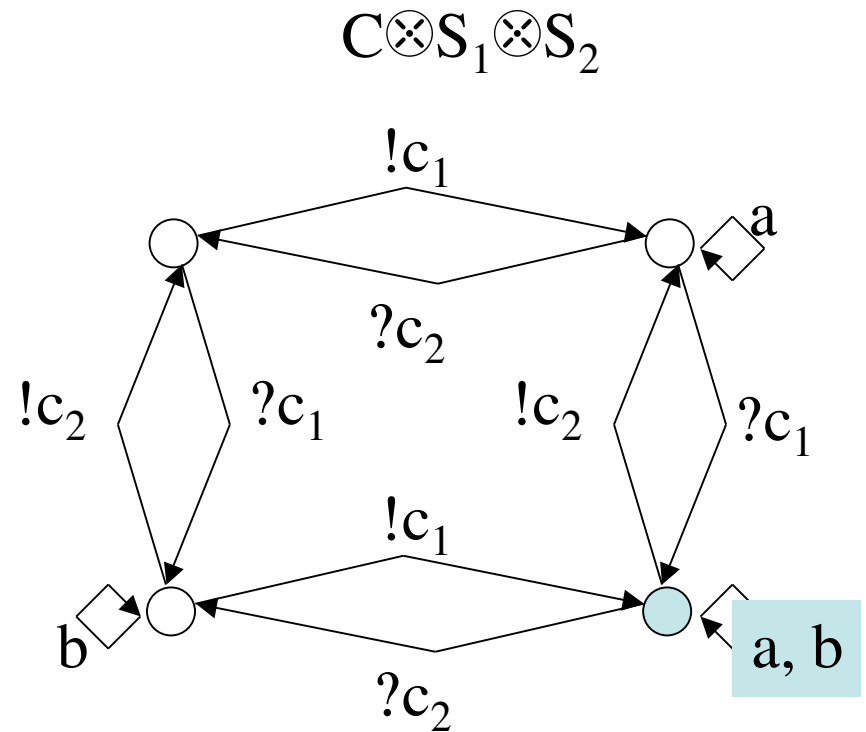
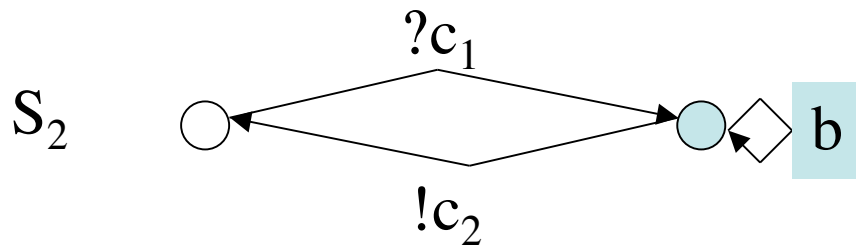
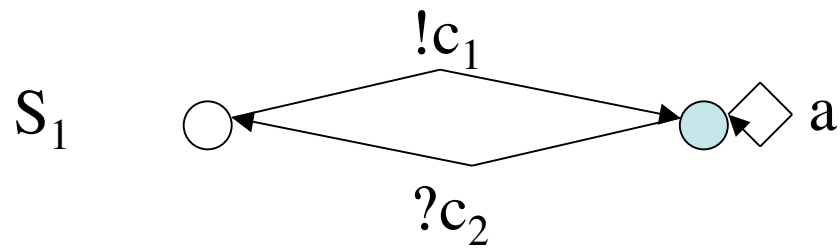
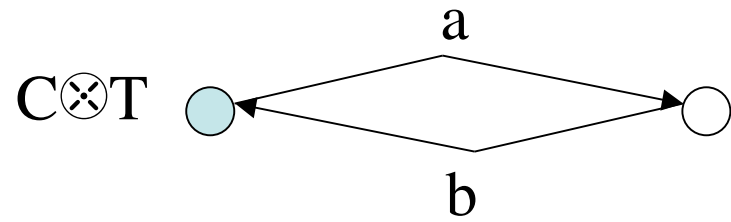
Service oriented computing



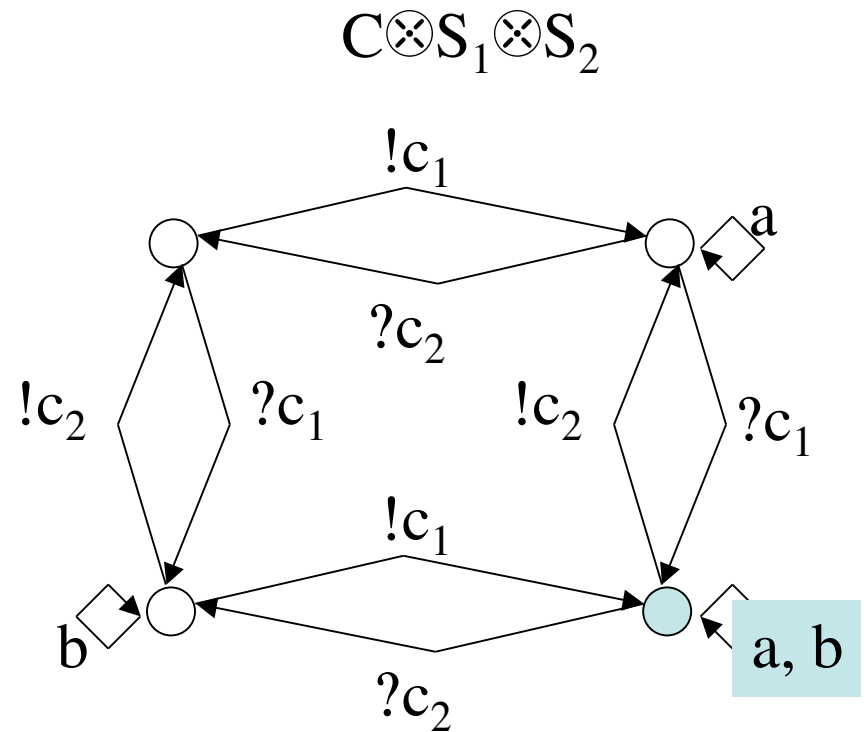
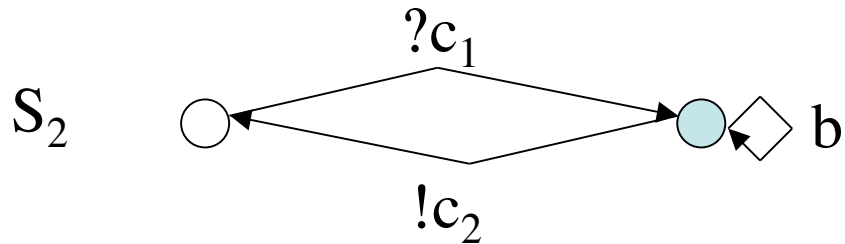
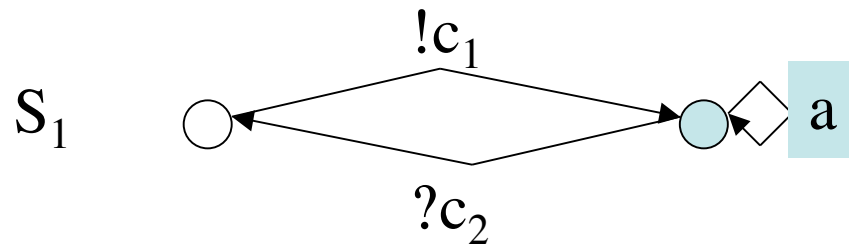
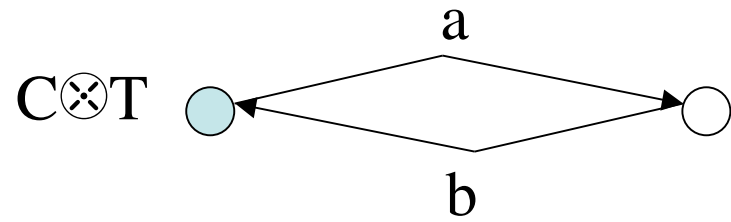
Service oriented computing



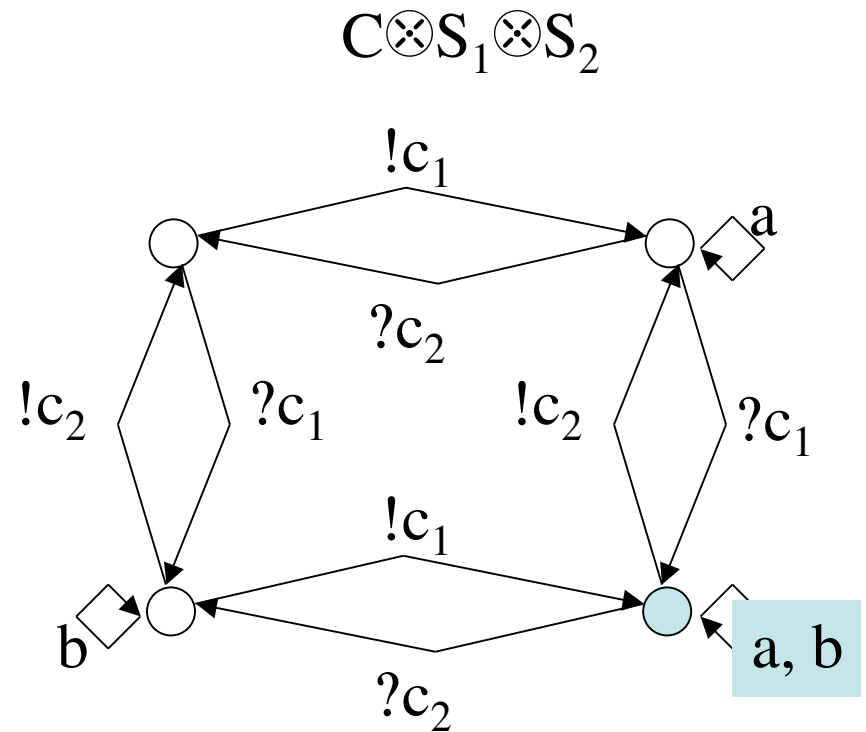
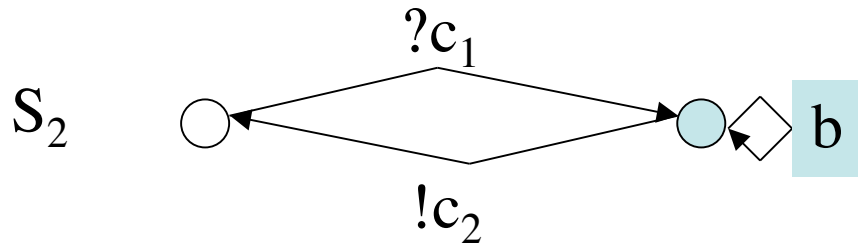
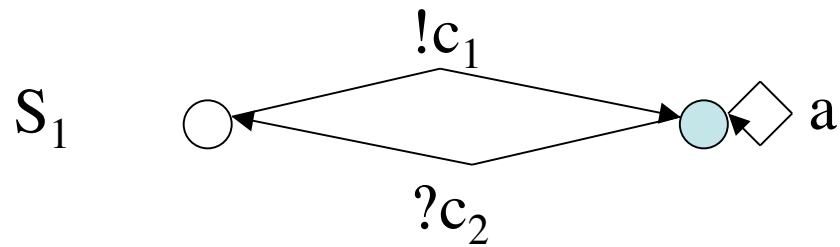
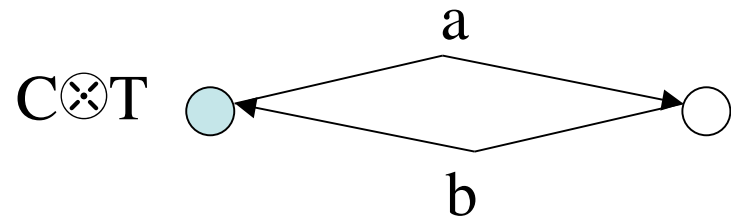
Service oriented computing



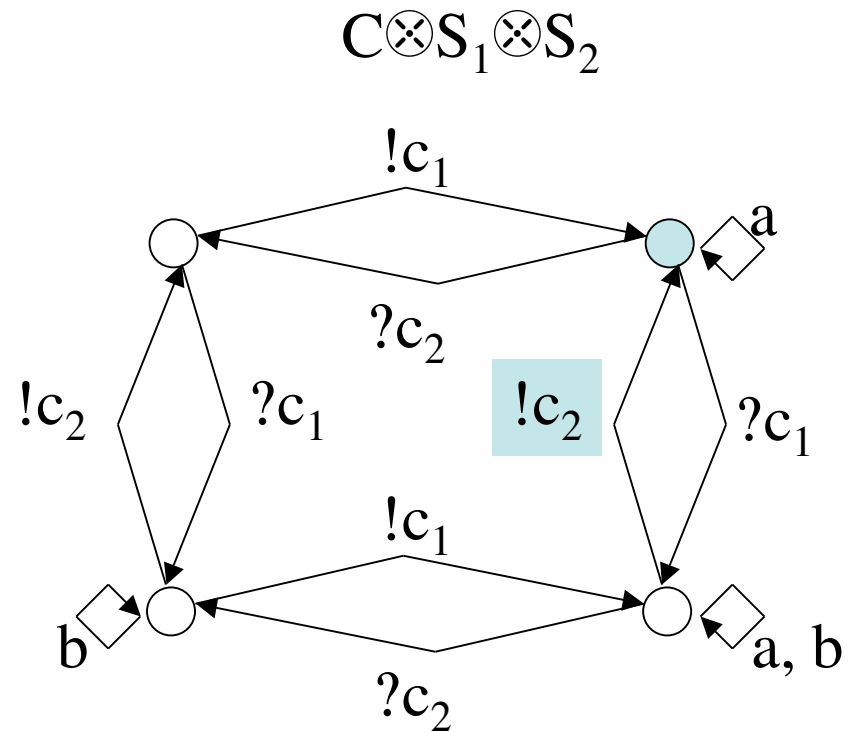
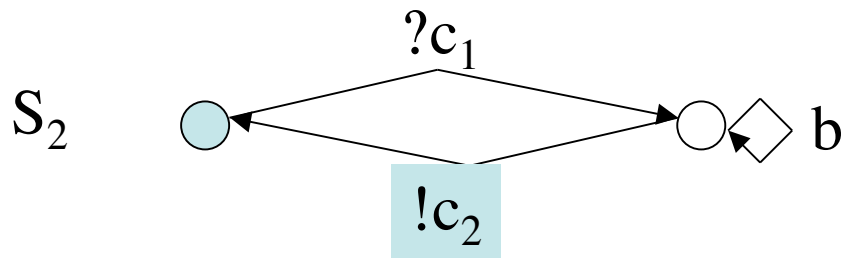
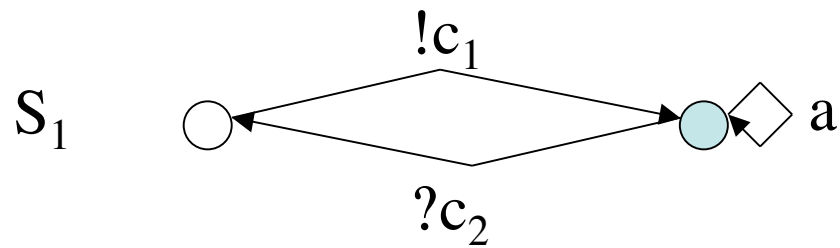
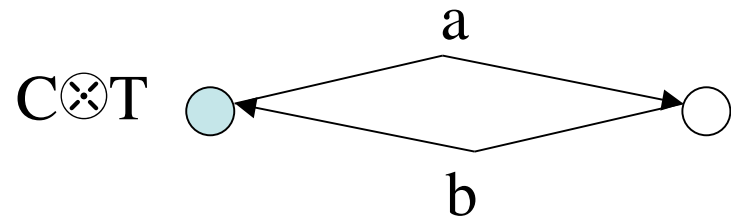
Service oriented computing



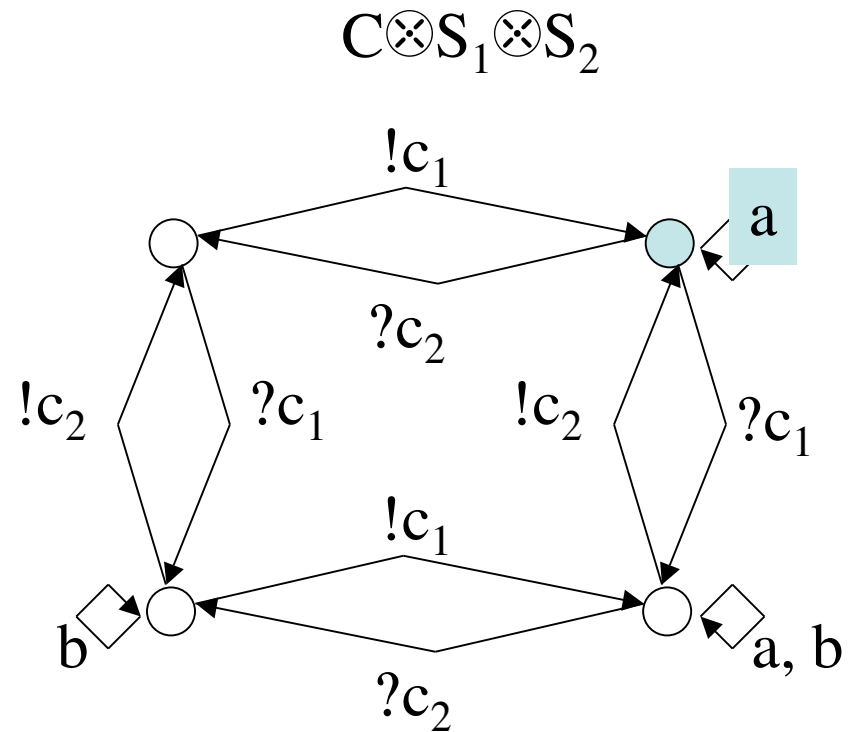
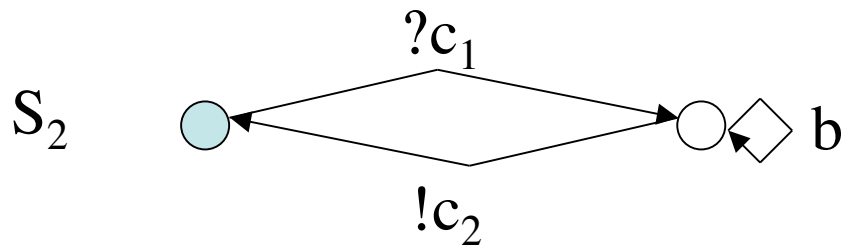
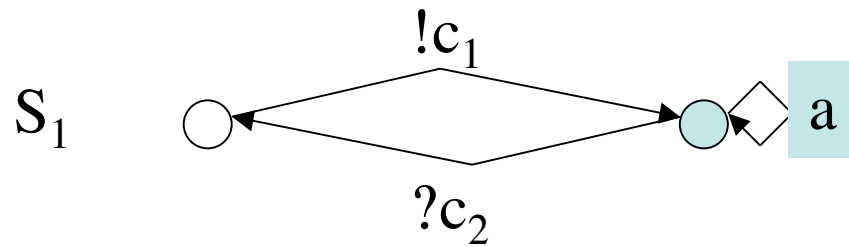
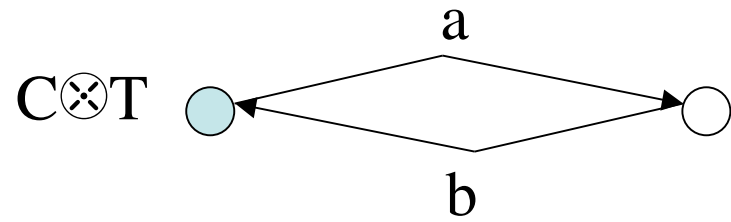
Service oriented computing



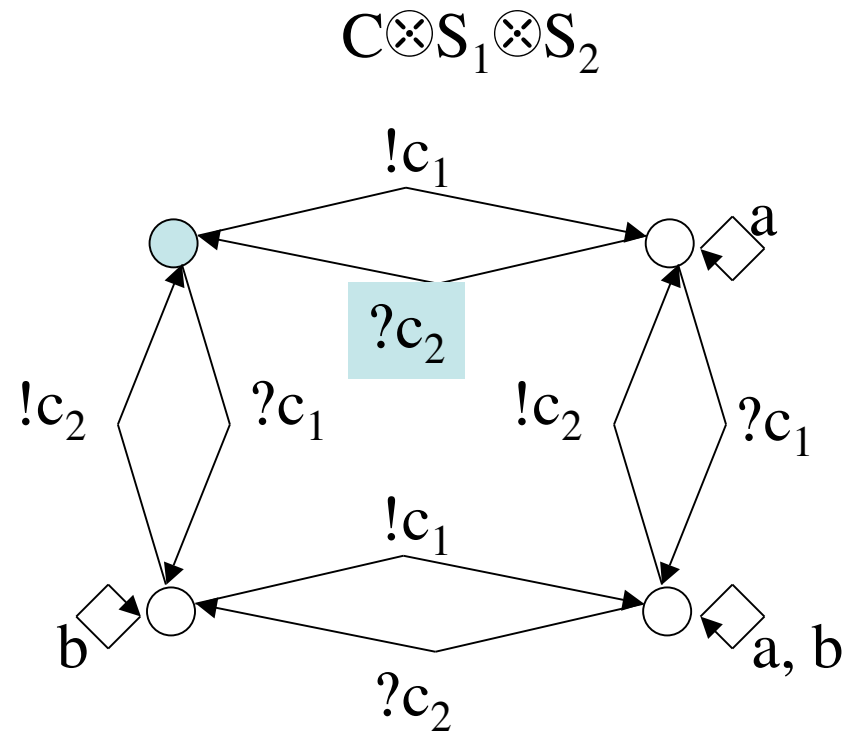
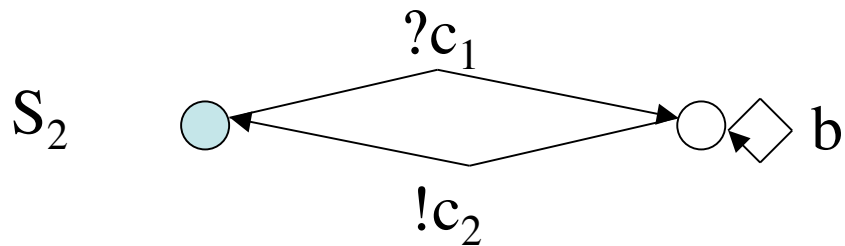
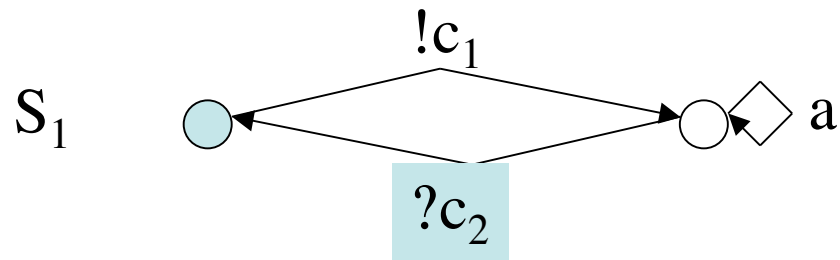
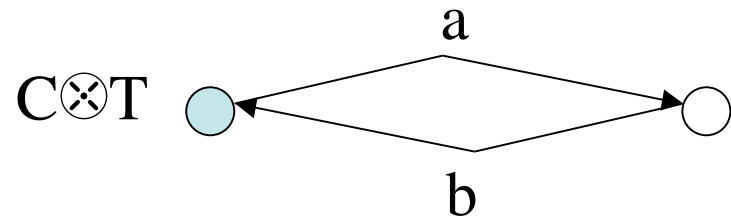
Service oriented computing



Service oriented computing

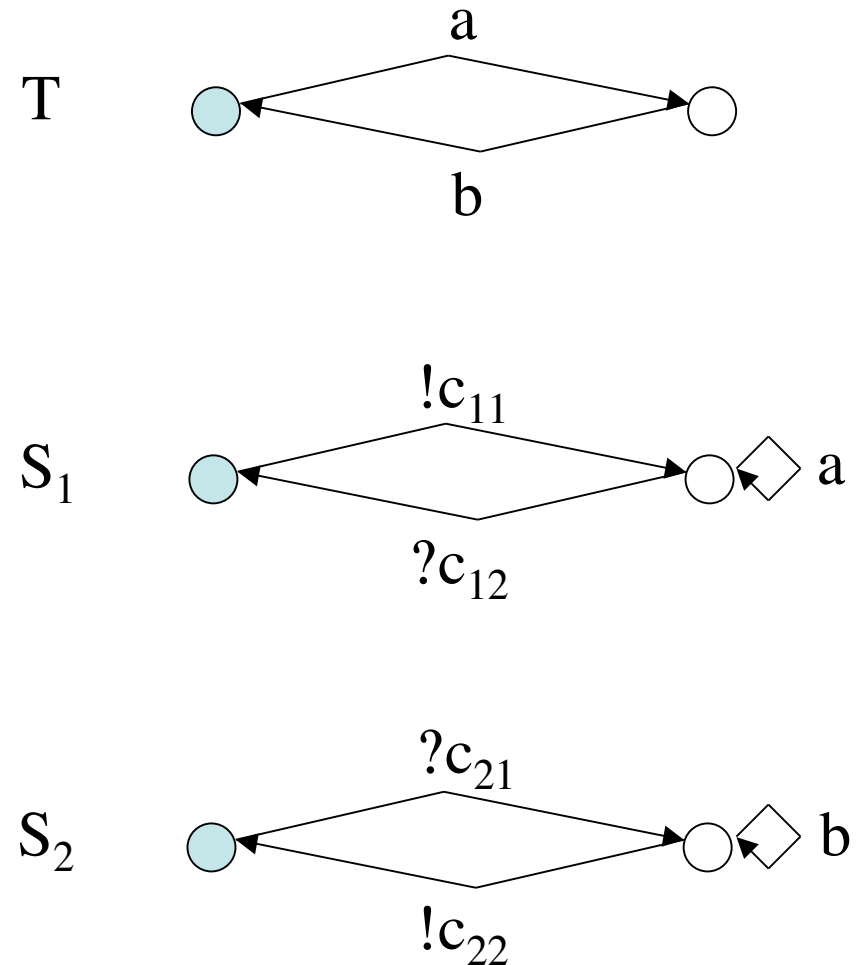


Service oriented computing

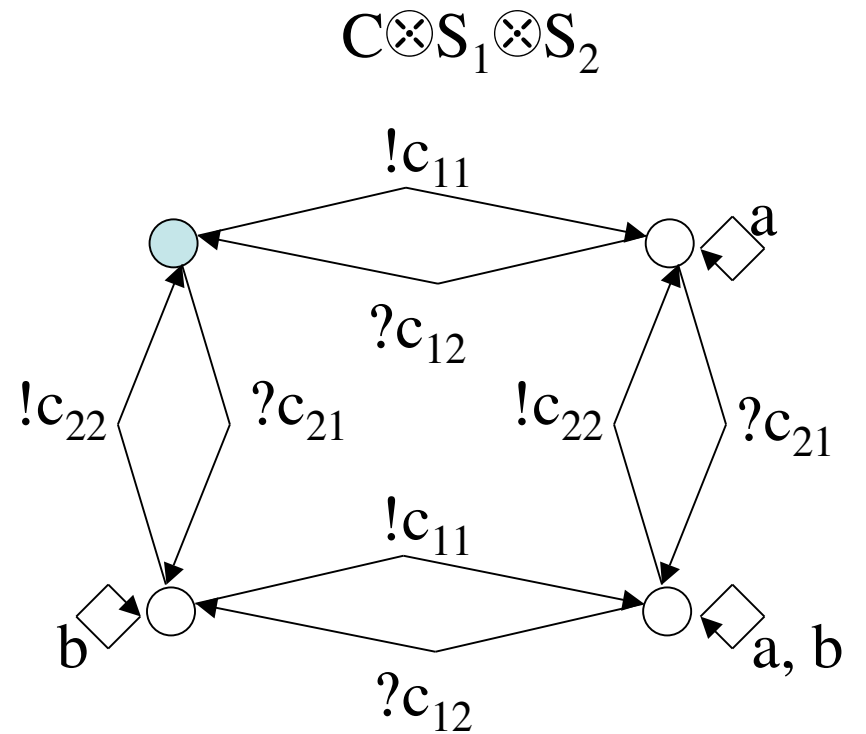
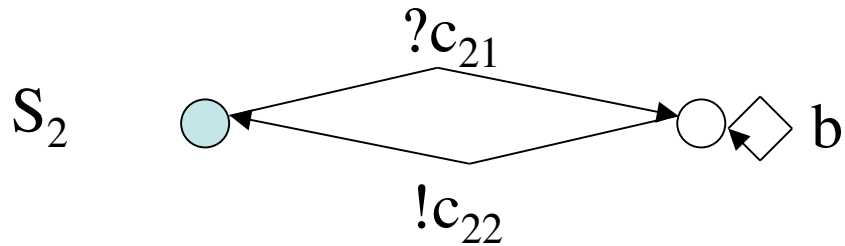
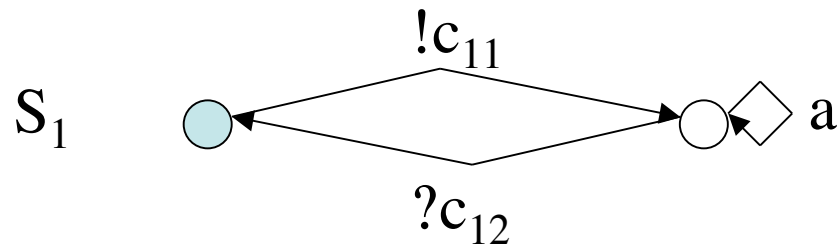
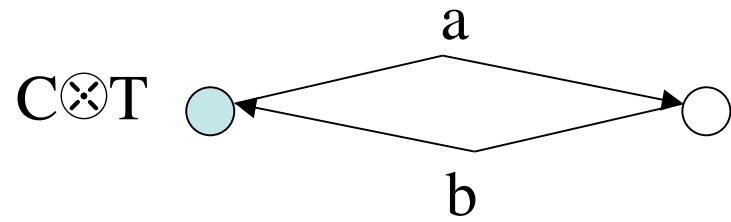


Service oriented computing

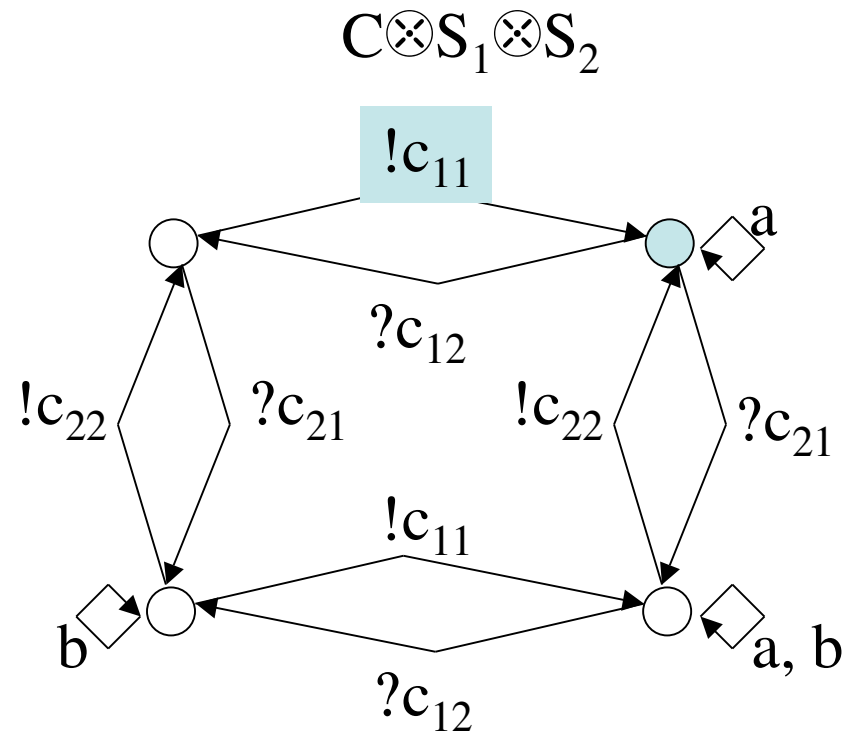
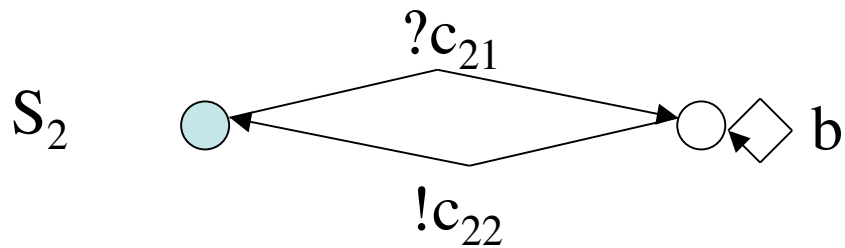
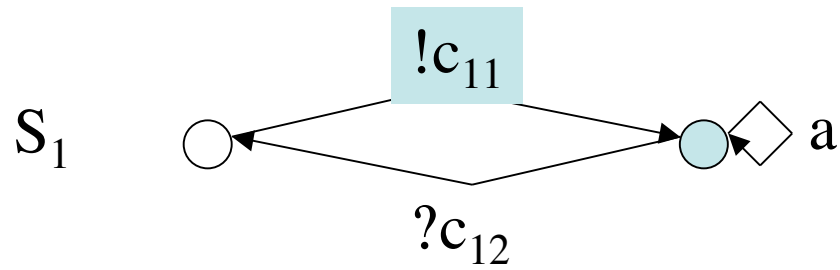
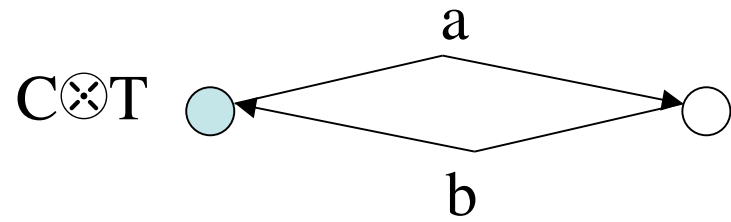
- Community ontology
 - $\Sigma = \{a,b\}$, $Ch = \{c_{11},c_{12},c_{21},c_{22}\}$
- Client service
 - $C = (Q_C, q_{0C}, \delta_C)$
- Target service
 - $T = (Q_T, q_{0T}, \delta_T)$
- Available services
 - $S_1 = (Q_1, q_{01}, \delta_1)$
 - $S_2 = (Q_2, q_{02}, \delta_2)$



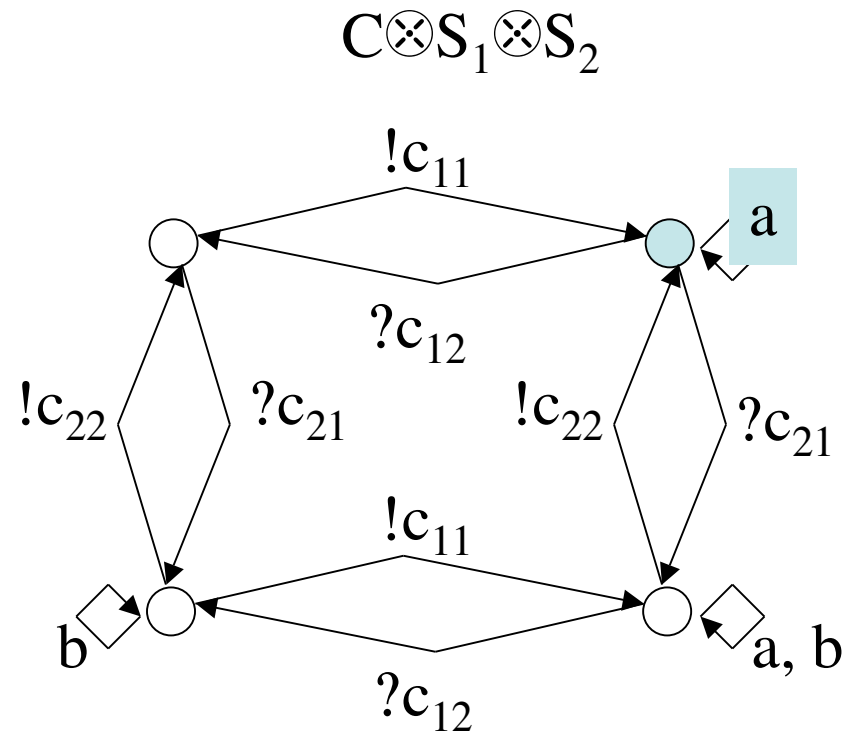
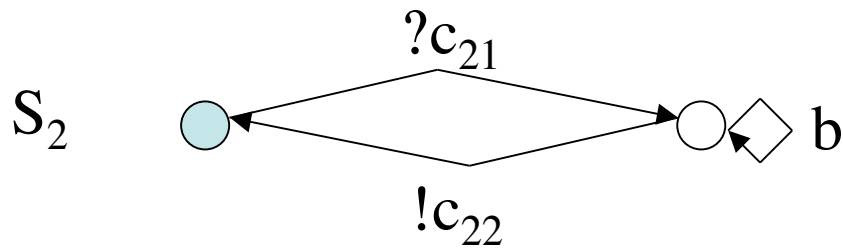
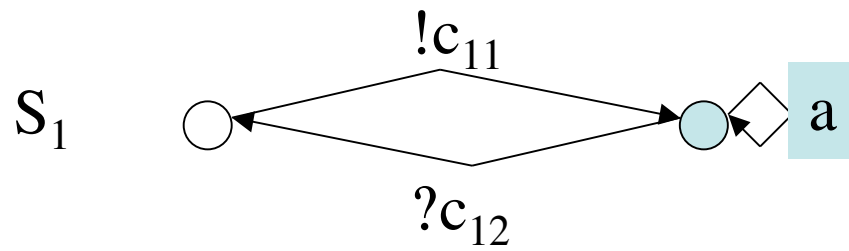
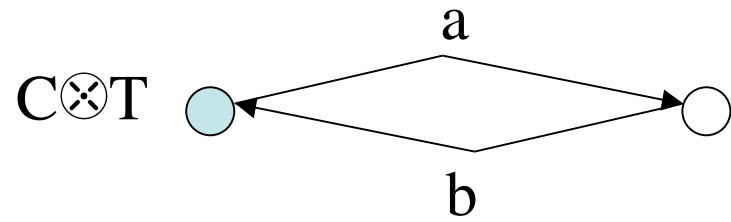
Service oriented computing



Service oriented computing



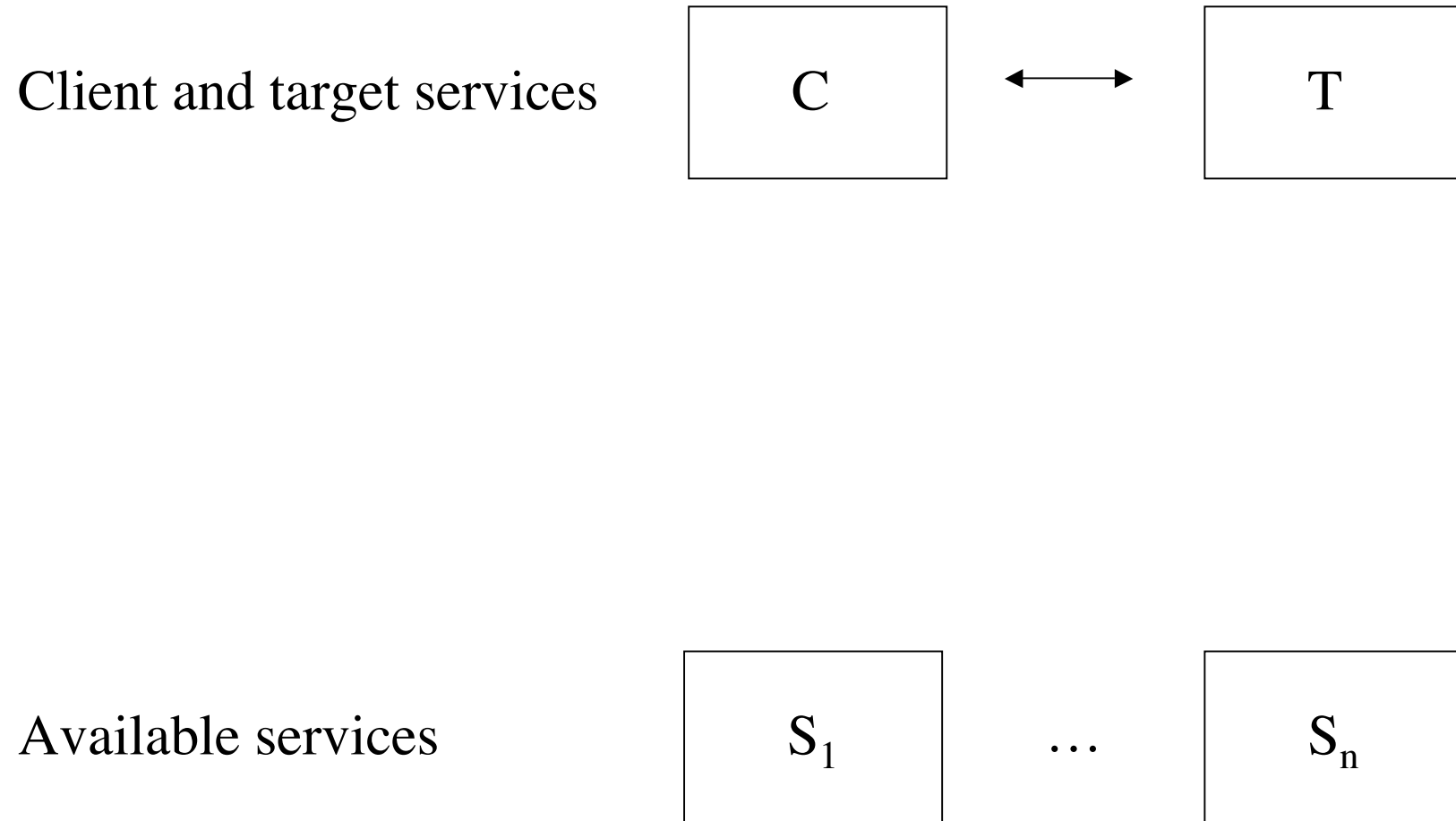
Service oriented computing



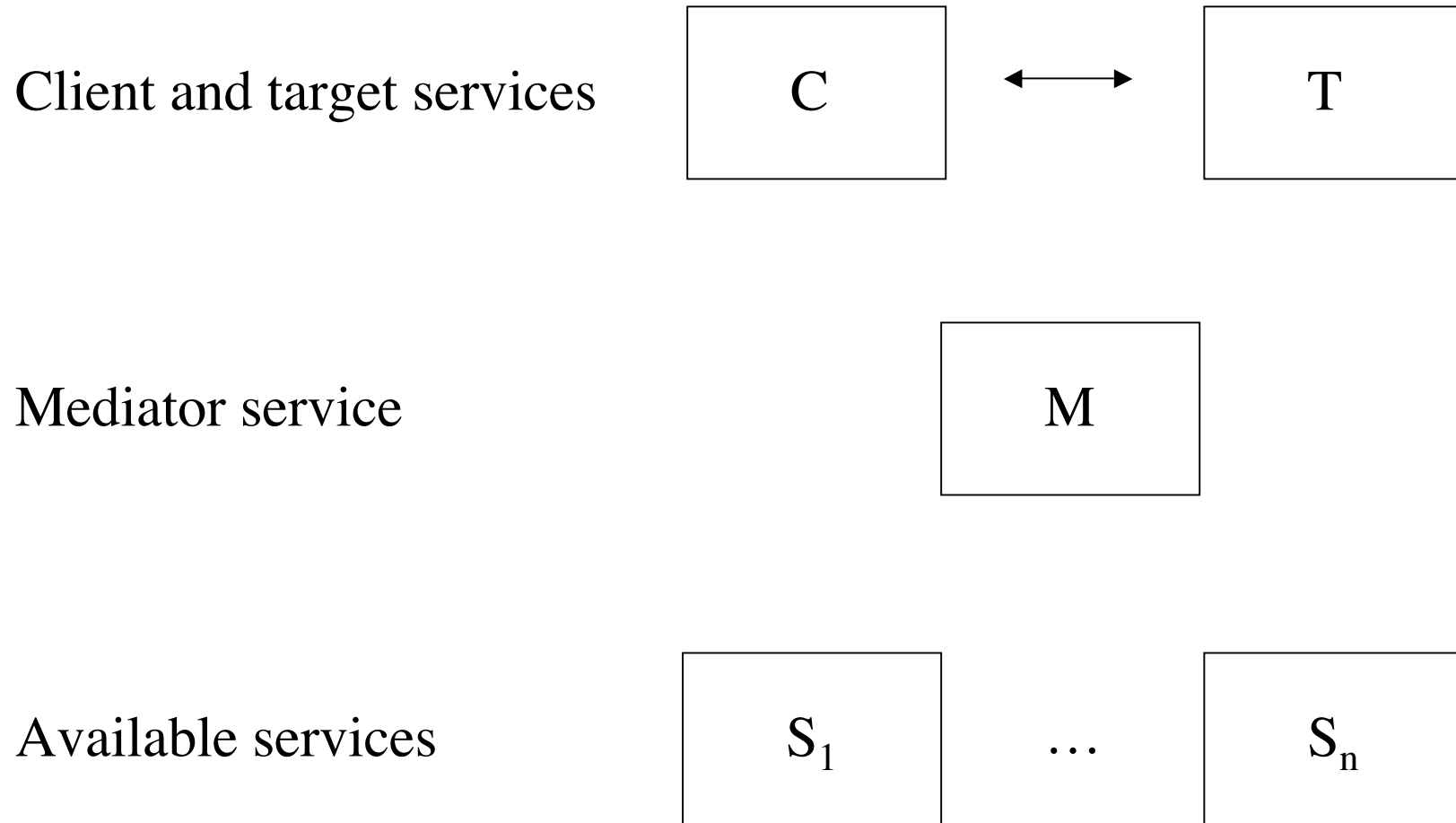
Outline

- Service oriented computing
- **Composition problem**
- Controller synthesis
- Modal logic
- Composition of interactive Web services
- Variants and open problems

Composition problem



Composition problem



Composition problem

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $Ch = \{c_1, \dots, c_{|Ch|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - Determine whether there exists a mediator service M coordinating the client service C and the available services S_1, \dots, S_n into a pattern that provides an outcome « able to satisfy » the target service T

Composition problem

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - Determine whether there exists a mediator service M such that
 - $\text{trace}(\text{fa}(C \otimes T)) \subseteq \text{trace}(\text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n))$
 - $\text{trace}(\text{fa}(C \otimes T)) = \text{trace}(\text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n))$
 - $\text{fa}(C \otimes T) \leq \text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n)$
 - $\text{fa}(C \otimes T) \equiv \text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n)$

Composition problem

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - Determine whether there exists a mediator service M such that
 - $\text{trace}(\text{fa}(C \otimes T)) \subseteq \text{trace}(\text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n))$
 - $\text{trace}(\text{fa}(C \otimes T)) = \text{trace}(\text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n))$
 - $\text{fa}(C \otimes T) \leq \text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n)$
 - **$\text{fa}(C \otimes T) \equiv \text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n)$**

Outline

- Service oriented computing
- Composition problem
- **Controller synthesis**
- Modal logic
- Composition of interactive Web services
- Variants and open problems

Controller synthesis

- Basic control problem [Ramadge and Wonham, 1989]
 - Process : automaton S over an alphabet Σ of events
 - Controllable events (Σ_{con}) and uncontrollable events (Σ_{unc})
 - Observable events (Σ_{cobs}) and unobservable events (Σ_{uno})
 - Controller : process M such that
 1. For any state q of M and for any uncontrollable event a , there is a transition from q labelled by a
 2. For any state q of M and for any unobservable event a , if there is a transition from q labelled by a then this transition is a loop over q
 - Supervised system : product $M \times S$

Controller synthesis

- Basic control problem [Ramadge and Wonham, 1989]
 - Given a process S and a set T of behaviours, does there exist a controller M satisfying (1) and (2) such that the behaviours of the supervised system $M \times S$ are all in T ?

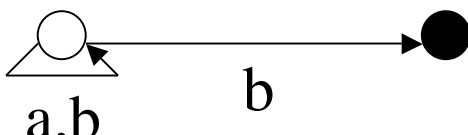
Controller synthesis

- Basic control problem [Ramadge and Wonham, 1989]
 - Given a process S and a set T of behaviours, does there exist a controller M satisfying (1) and (2) such that the behaviours of the supervised system $M \times S$ are all in T ?

- Example

- $\Sigma = \{a,b,c\}$, $\Sigma_{\text{con}} = \{a,b\}$, $\Sigma_{\text{unc}} = \{c\}$, $\Sigma_{\text{obs}} = \{b\}$, $\Sigma_{\text{uno}} = \{a,c\}$

- $T = \{b, ab, aab, \dots\}$

- $S =$ 

Controller synthesis

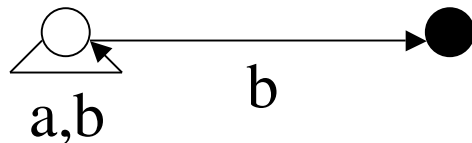
- Basic control problem [Ramadge and Wonham, 1989]
 - Given a process S and a set T of behaviours, does there exist a controller M satisfying (1) and (2) such that the behaviours of the supervised system $M \times S$ are all in T ?

- Example

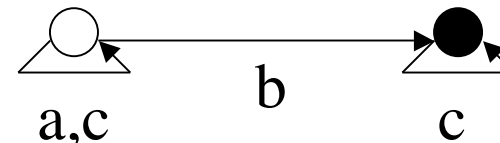
- $\Sigma = \{a,b,c\}$, $\Sigma_{\text{con}} = \{a,b\}$, $\Sigma_{\text{unc}} = \{c\}$, $\Sigma_{\text{obs}} = \{b\}$, $\Sigma_{\text{uno}} = \{a,c\}$

- $T = \{b, ab, aab, \dots\}$

- $S =$



- $M =$



Controller synthesis

- First extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and a modal formula ϕ , does there exist a controller M satisfying (1) and (2) such that the supervised system $M \times S$ satisfies ϕ ?

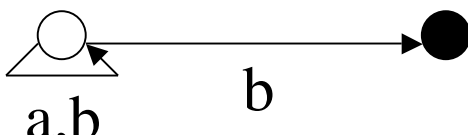
Controller synthesis

- First extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and a modal formula ϕ , does there exist a controller M satisfying (1) and (2) such that the supervised system $M \times S$ satisfies ϕ ?

- Example

- $\Sigma = \{a,b,c\}$, $\Sigma_{\text{con}} = \{a,b\}$, $\Sigma_{\text{unc}} = \{c\}$, $\Sigma_{\text{obs}} = \{b\}$, $\Sigma_{\text{uno}} = \{a,c\}$

- $\phi = [U][c]\text{false} \wedge [a^*](\langle a \rangle \text{true} \wedge \langle b \rangle \text{true})$

- $S =$ 

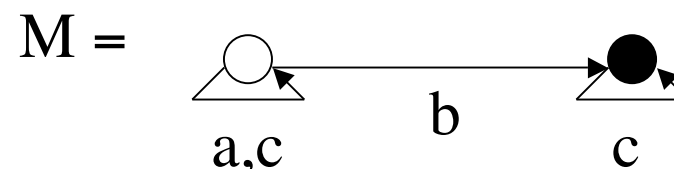
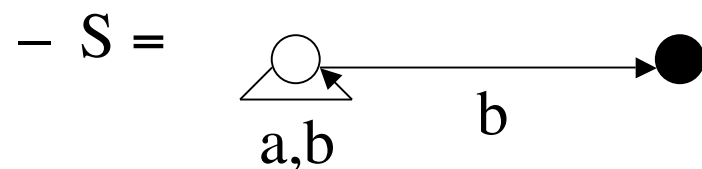
Controller synthesis

- First extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and a modal formula ϕ , does there exist a controller M satisfying (1) and (2) such that the supervised system $M \times S$ satisfies ϕ ?

- Example

- $\Sigma = \{a,b,c\}$, $\Sigma_{\text{con}} = \{a,b\}$, $\Sigma_{\text{unc}} = \{c\}$, $\Sigma_{\text{obs}} = \{b\}$, $\Sigma_{\text{uno}} = \{a,c\}$

- $\phi = [U][c]\text{false} \wedge [a^*](\langle a \rangle \text{true} \wedge \langle b \rangle \text{true})$



Controller synthesis

- Second extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and modal formulas ϕ' , ϕ'' , does there exist a controller M satisfying ϕ' such that the supervised system $M \times S$ satisfies ϕ'' ?

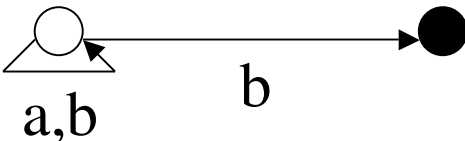
Controller synthesis

- Second extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and modal formulas ϕ' , ϕ'' , does there exist a controller M satisfying ϕ' such that the supervised system $M \times S$ satisfies ϕ'' ?

- Example

- $\Sigma = \{a,b\}$, $\phi' = [U]\langle c \rangle \text{true} \wedge [U][a \cap \neq] \text{false} \wedge [U][c \cap \neq] \text{false}$

- $\phi'' = [U][c] \text{false} \wedge [a^*](\langle a \rangle \text{true} \wedge \langle b \rangle \text{true})$

- $S =$ 

Controller synthesis

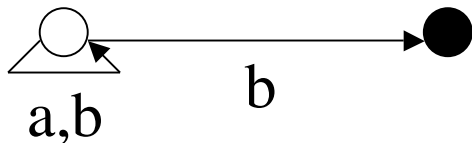
- Second extension of the basic control problem [Arnold *et al.*, 2003]
 - Given a process S and modal formulas ϕ' , ϕ'' , does there exist a controller M satisfying ϕ' such that the supervised system $M \times S$ satisfies ϕ'' ?

- Example

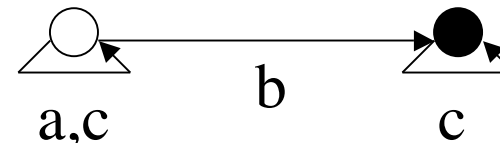
- $\Sigma = \{a,b\}$, $\phi' = [U]\langle c \rangle \text{true} \wedge [U][a \cap \neq] \text{false} \wedge [U][c \cap \neq] \text{false}$

- $\phi'' = [U][c] \text{false} \wedge [a^*](\langle a \rangle \text{true} \wedge \langle b \rangle \text{true})$

- $S =$



- $M =$



Outline

- Service oriented computing
- Composition problem
- Controller synthesis
- **Modal logic**
- Composition of interactive Web services
- Variants and open problems

Modal logic

- Syntax
 - Modal formulas : $\phi ::= \text{false} \mid \neg\phi \mid (\phi' \vee \phi'') \mid [a]\phi$

Modal logic

- Syntax
 - Modal formulas : $\phi ::= \text{false} \mid \neg\phi \mid (\phi' \vee \phi'') \mid [a]\phi$
 - Abbreviation : $\langle a \rangle\phi ::= \neg[a]\neg\phi$

Modal logic

- Syntax
 - Modal formulas : $\phi ::= \text{false} \mid \neg\phi \mid (\phi' \vee \phi'') \mid [a]\phi$
 - Abbreviation : $\langle a \rangle\phi ::= \neg[a]\neg\phi$
- Semantics
 - $S = (Q, q_0, \delta)$

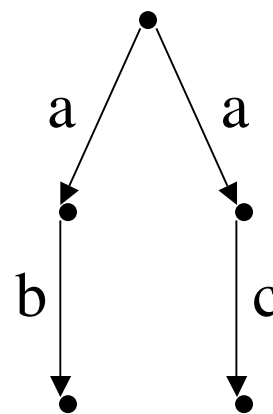
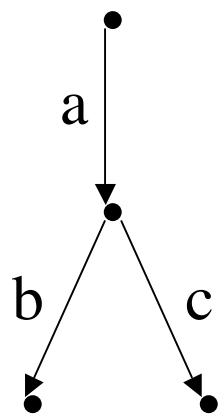
Modal logic

- Syntax
 - Modal formulas : $\phi ::= \text{false} \mid \neg\phi \mid (\phi' \vee \phi'') \mid [a]\phi$
 - Abbreviation : $\langle a \rangle\phi ::= \neg[a]\neg\phi$
- Semantics
 - $S = (Q, q_0, \delta)$
- Satisfiability
 - Non S sat false
 - S sat $\neg\phi$ iff non S sat ϕ
 - S sat $\phi' \vee \phi''$ iff S sat ϕ' or S sat ϕ''
 - S sat $[a]\phi$ iff for all $q_0' \in \delta(q_0, a)$, (Q, q_0', δ) sat ϕ

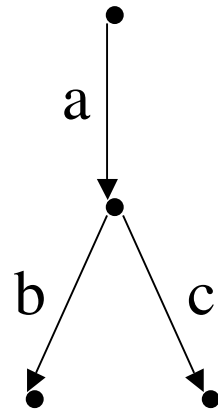
Modal logic

- Syntax
 - Modal formulas : $\phi ::= \text{false} \mid \neg\phi \mid (\phi' \vee \phi'') \mid [a]\phi$
 - Abbreviation : $\langle a \rangle\phi ::= \neg[a]\neg\phi$
- Semantics
 - $S = (Q, q_0, \delta)$
- Satisfiability
 - Non S sat false
 - S sat $\neg\phi$ iff non S sat ϕ
 - S sat $\phi' \vee \phi''$ iff S sat ϕ' or S sat ϕ''
 - S sat $[a]\phi$ iff for all $q_0' \in \delta(q_0, a)$, (Q, q_0', δ) sat ϕ
 - S sat $\langle a \rangle\phi$ iff there exists $q_0' \in \delta(q_0, a)$ such that (Q, q_0', δ) sat ϕ

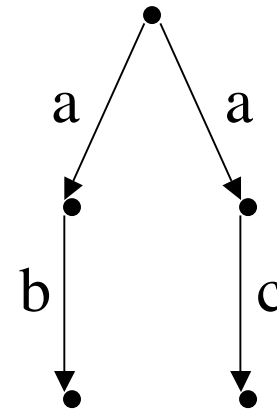
Modal logic



Modal logic



$\langle a \rangle (\langle b \rangle \text{true} \wedge \langle c \rangle \text{true})$



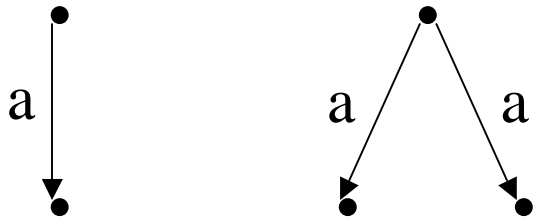
$\langle a \rangle (\langle b \rangle \text{true} \wedge [c] \text{false})$

$\langle a \rangle ([b] \text{false} \wedge \langle c \rangle \text{true})$

Modal logic

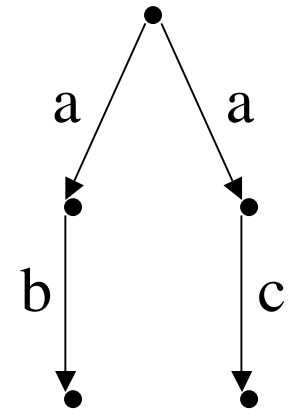
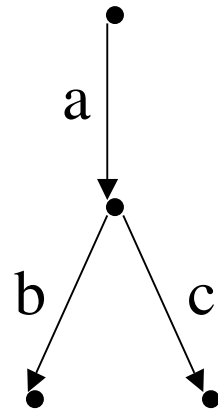
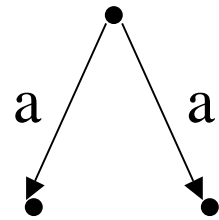
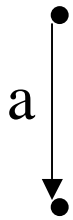
- Bisimulation
 - $S = (Q, q_0, \delta)$
 - $S' = (Q', q_0', \delta')$
 - $Z \subseteq Q \times Q'$ is a bisimulation between S and S' iff
 - $q_0 Z q_0'$
 - If $q_1 Z q_1'$ and $q_2 \in \delta(q_1, a)$ then there exists $q_2' \in \delta'(q_1', a)$ such that $q_2 Z q_2'$
 - If $q_1 Z q_1'$ and $q_2' \in \delta'(q_1', a)$ then there exists $q_2 \in \delta(q_1, a)$ such that $q_2 Z q_2'$

Modal logic



Bisimilar

Modal logic



Bisimilar

Non bisimilar

Modal logic

- Bisimulation theorem
 - Let $S = (Q, q_0, \delta)$, $S' = (Q', q_0', \delta')$ be services. If there exists a bisimulation between S and S' then for all modal formulas ϕ , $S \text{ sat } \phi$ iff $S' \text{ sat } \phi$
- Hennessy-Milner theorem
 - Let $S = (Q, q_0, \delta)$, $S' = (Q', q_0', \delta')$ be services. If for all modal formulas ϕ , $S \text{ sat } \phi$ iff $S' \text{ sat } \phi$ then there exists a bisimulation between S and S'

Modal logic

- Input
 - Service : $S = (Q, q_0, \delta)$
- Output
 - A modal formula ϕ_S such that the service S « is characterized by » the modal formula ϕ_S

Modal logic

- Input
 - Service : $S = (Q, q_0, \delta)$
- Output
 - A modal formula ϕ_S such that for all services $S' = (Q', q_0', \delta')$, there exists a bisimulation between S and S' iff S' sat ϕ_S

Modal logic

- Input
 - Service : $S = (Q, q_0, \delta)$
- Output
 - A modal formula ϕ_S such that for all services $S' = (Q', q_0', \delta')$, there exists a bisimulation between S and S' iff S' sat ϕ_S
- Solution
 - [Browne *et al.*, 1988] : existence of ϕ_S is proved within the context of CTL
 - [Baltag, 1998] : existence of ϕ_S is proved within the context of modal logic
 - [Balbiani and Herzig, 2008] : ϕ_S is effectively built within the context of CTL or modal logic

Outline

- Service oriented computing
- Composition problem
- Controller synthesis
- Modal logic
- **Composition of interactive Web services**
- Variants and open problems

Composition of interactive Web services

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - Determine whether there exists a mediator service M such that
 - $\text{fa}(C \otimes T) \equiv \text{fa}(C \otimes M \otimes S_1 \otimes \dots \otimes S_n)$

Composition of interactive Web services

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - $L = (Q_L, q_{0L}, \delta_L)$ being the largest mediator for C and S_1, \dots, S_n , determine whether there exists a service S such that
 - $\text{fa}(C \otimes T) \equiv \text{fa}(C \otimes (S \times L) \otimes S_1 \otimes \dots \otimes S_n)$

Composition of interactive Web services

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - $L = (Q_L, q_{0L}, \delta_L)$ being the largest mediator for C and S_1, \dots, S_n , determine whether there exists a service S such that
 - $\text{fa}(C \otimes T) \equiv \text{del}^\circ(\text{fa}^\circ(C \otimes L \otimes S_1 \otimes \dots \otimes S_n) \times S)$

Composition of interactive Web services

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - $L = (Q_L, q_{0L}, \delta_L)$ being the largest mediator for C and S_1, \dots, S_n and $\phi_{C \otimes T}$ being the modal formula associated to $\text{fa}(C \otimes T)$, determine whether there exists a service S such that
 - $\text{fa}^\circ(C \otimes L \otimes S_1 \otimes \dots \otimes S_n) \times S \text{ sat } \phi_{C \otimes T}$

Composition of interactive Web services

- Input
 - Community ontology : $\Sigma = \{a_1, \dots, a_{|\Sigma|}\}$, $\text{Ch} = \{c_1, \dots, c_{|\text{Ch}|}\}$
 - Client service : $C = (Q_C, q_{0C}, \delta_C)$
 - Target service : $T = (Q_T, q_{0T}, \delta_T)$
 - Available services : $S_1 = (Q_1, q_{01}, \delta_1), \dots, S_n = (Q_n, q_{0n}, \delta_n)$
- Output
 - $L = (Q_L, q_{0L}, \delta_L)$ being the largest mediator for C and S_1, \dots, S_n and $\phi_{C \otimes T}$ being the modal formula associated to $C \otimes T$, determine whether there exists a service S such that
 - $\text{fa}^\circ(C \otimes L \otimes S_1 \otimes \dots \otimes S_n) \times S \text{ sat } \phi_{C \otimes T}$
- Solution
 - Apply the procedure described in [Arnold *et al.*, 2003]

Composition of interactive Web services

- Algorithm
 - Compute (in PTIME) the service $C \otimes T$
 - Compute (in PTIME) the modal formula $\phi_{C \otimes T}$
 - Compute (in PTIME) the largest mediator $L = (Q_L, q_{0L}, \delta_L)$
 - Compute (in EXPTIME) the service $C \otimes L \otimes S_1 \otimes \dots \otimes S_n$
 - Apply (in 2EXPTIME) the procedure described in [Arnold *et al.*, 2003] to determine whether there exists a service S such that
 - $\text{fa}^\circ(C \otimes L \otimes S_1 \otimes \dots \otimes S_n) \times S \text{ sat } \phi_{C \otimes T}$
- This algorithm can be given a 2EXPTIME implementation

Outline

- Service oriented computing
- Composition problem
- Controller synthesis
- Modal logic
- Composition of interactive Web services
- **Variants and open problems**

Variants and open problems

- The exact complexity of the composition problem for bisimulation is still unknown
- Consider the composition problem for trace inclusion, trace equivalence or simulation
- Consider communication channels that can contain more than one message at a time
- Consider deterministic services
- Consider services that can exchange real messages, i.e. first-order terms