Modeling and Simulation with Ontology Streams for Agents' Interactions

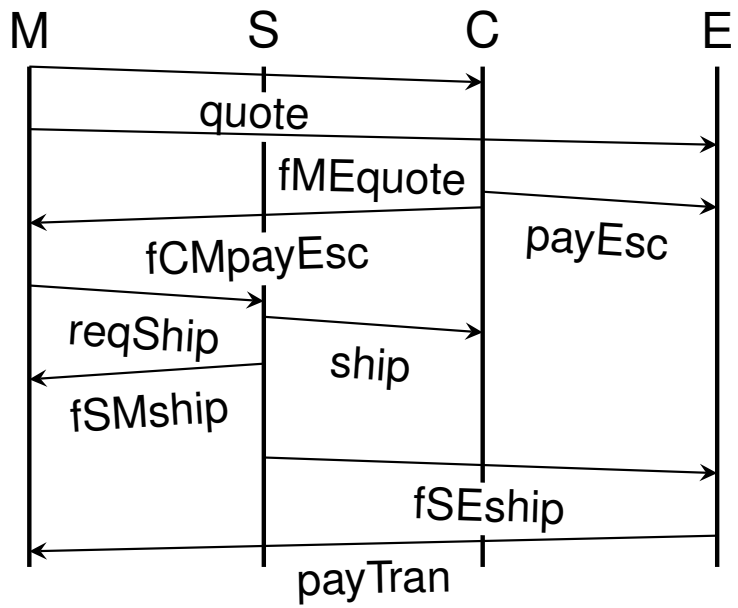
Ioan Alfred Letia and Adrian Groza

Department of Computer Science
Technical University of Cluj-Napoca, Romania
Adrian.Groza@cs.utcluj.ro

October 26, 2022

Purchase Commitments

Protocol enactment for **M**erchant, **S**hipper, **E**scrow, and **C**ustomer roles




```
1  commitment EscrowPurchase M to C
2   create quote
3   detach payEscrow[, quote + 9]
4   discharge ship[, payEscrow + 5]
```

```
1  commitment EscrowTransfer E to M
2   create payEscrow
3   detach discharged(EscrowPurchase)
4   discharge payTransfer[,
        discharged(EscrowPurchase) + 4]
```

Example (Purchase Commitments)

*EscrowPurchase M to C. A Purchase commitment from **M** to **C** is created when a quote is made. The Purchase is detached if a payment occurs within 9 time points of the quote. If the payment does not occur by the deadline, then the commitment is expired. This commitment is discharged if the shipment occurs within 5 time points of the payment and if the shipment does not happen by the deadline the commitment is violated.*

*EscrowTransfer E to M. If the creditor **M** infers its detach, then so should the debtor **E**, therefore EscrowPurchase's discharge is EscrowTransfer's detach.*

Semantic Web Vision

"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

Tim Barners Lee

Semantic Web Vision

"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

Tim Barners Lee

Social Web

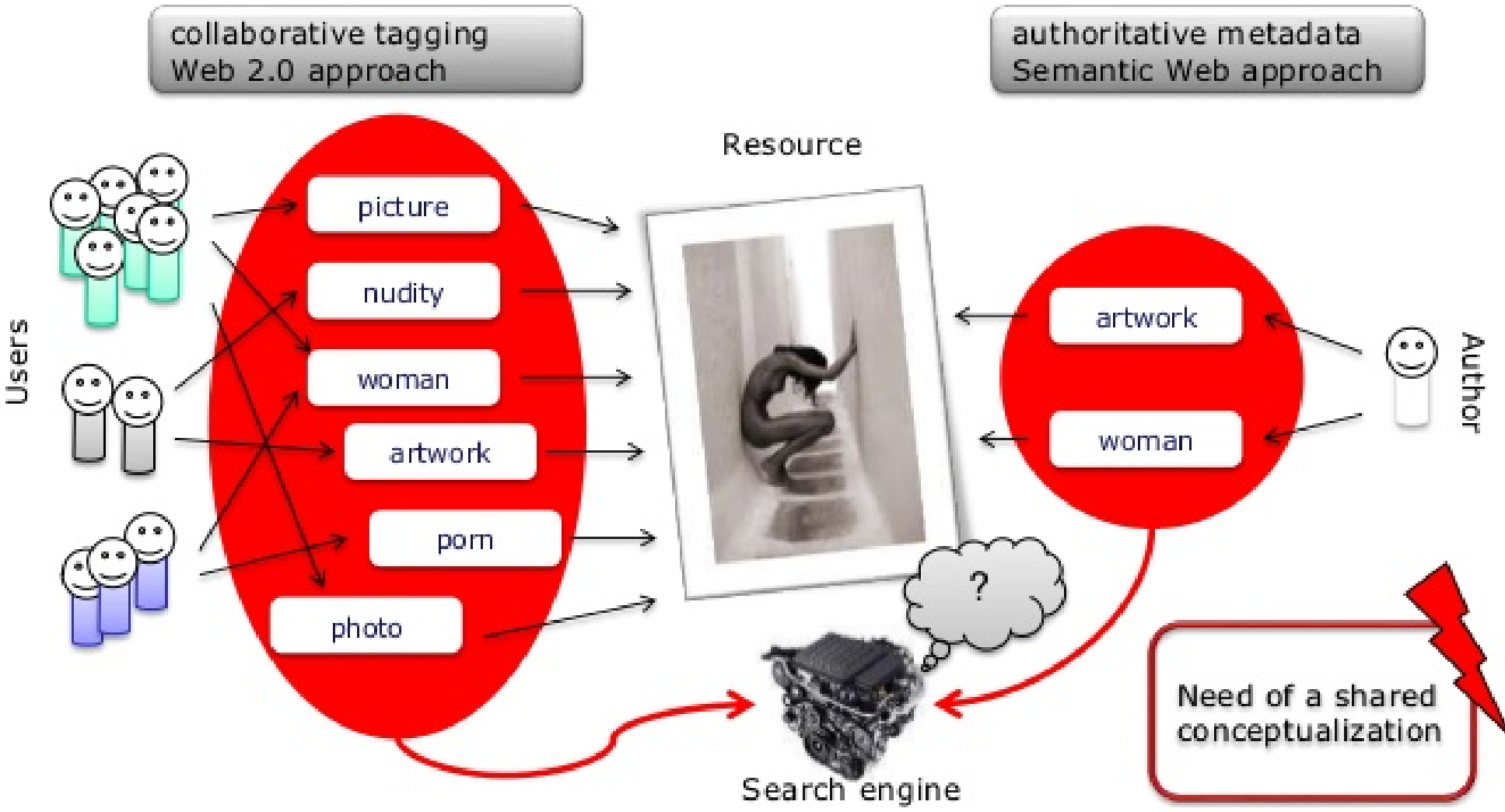
Semantic Web Vision

"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

Tim Barners Lee

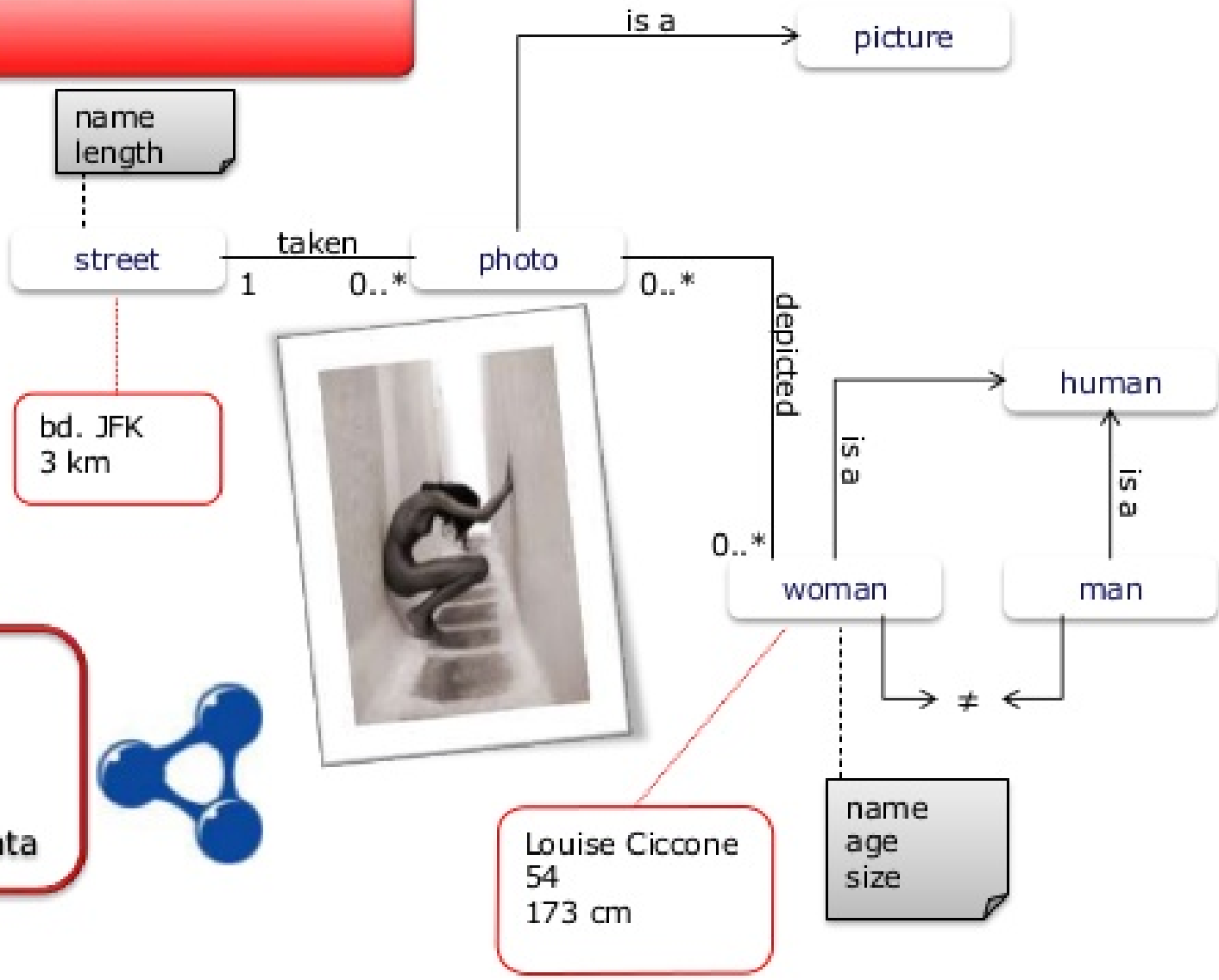
Social Web

Semantic Web


collaborative tagging
Web 2.0 approach
authoritative metadata
Semantic Web approach
Resource
Users
picture
nudity
woman
artwork
porn
photo
artwork
woman
Author
?
Search engine
Need of a shared
conceptualization

Conceptualization

- concepts
- relations between concepts
- attributes
- instances



Using the same ontology allows two different systems to communicate and to reason over (meta)data

Description Logic $\mathcal{EL}^{++}$

- A signature Σ, denoted by $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$: $\top \mid \bot \mid A \mid C \sqcap D \mid \exists r.C \mid \{a\}$.
- A DL ontology $\mathcal{O} \doteq \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$.
  - The TBox contains concepts and role axioms, e.g $C \sqsubseteq D$, $r \sqsubseteq s$
  - The ABox consists of concept and role assertions, e.g. $C(a)$, $R(a, b)$
- $\mathcal{EL}^{++}$ TBox completion rules

R_1 If X $\sqsubseteq$ A, A $\sqsubseteq$ B then X $\sqsubseteq$ B
R_2 If X $\sqsubseteq A_1, \cdots A_n$, $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq$ B then X $\sqsubseteq$ B
R_3 If X $\sqsubseteq$ A, A $\sqsubseteq \exists$r.B then X $\sqsubseteq \exists$r.B
R_4 If X $\sqsubseteq \exists$r.A, A $\sqsubseteq$ A', $\exists$r.A' $\sqsubseteq$ B then X $\sqsubseteq$ B
R_5 If X $\sqsubseteq \exists$r.A, A $\sqsubseteq \bot$, then X $\sqsubseteq \bot$
R_6 If X $\sqsubseteq \exists$r.A, r $\sqsubseteq$ s, then X $\sqsubseteq \exists$s.A
R_7 If X $\sqsubseteq \exists r_1$.A, A $\sqsubseteq \exists r_2$.B, $r_1 \circ r_2 \sqsubseteq r_3$ then X $\sqsubseteq \exists r_3$.B

Modelling relations in the Purchase protocol

Domain	Range	Role subsumption
$\exists payEscrow.\top \sqsubseteq Customer$	$\top \sqsubseteq \forall payEscrow.Escrow$	$payEscrow \sqsubseteq action$
$\exists ship.\top \sqsubseteq Shipper$	$\top \sqsubseteq \forall ship.Customer$	$ship \sqsubseteq action$
$\exists payTransfer.\top \sqsubseteq Escrow$	$\top \sqsubseteq \forall payTransfer.Merchant$	$payTransfer \sqsubseteq action$
$\exists quote.\top \sqsubseteq Merchant$	$\top \sqsubseteq \forall quote.Customer$	$quote \sqsubseteq message$
$\exists reqShip.\top \sqsubseteq Merchant$	$\top \sqsubseteq \forall reqShip.Shipper$	$regShip \sqsubseteq message$
$\exists fMEquote.\top \sqsubseteq Merchant$	$\top \sqsubseteq \forall fMEquote.Escrow$	$fMEquote \sqsubseteq message$
$\exists fCMpayEscrow.\top \sqsubseteq Customer$	$\top \sqsubseteq \forall fCMpayEscrow.Merchant$	$fCMpayEscrow \sqsubseteq message$
$\exists fSMship.\top \sqsubseteq Shipper$	$\top \sqsubseteq \forall fSMship.Merchant$	$fSMship \sqsubseteq message$
$\exists fSEship.\top \sqsubseteq Shipper$	$\top \sqsubseteq \forall fSEship.Escrow$	$fSEship \sqsubseteq message$

Domain	Range	Inverse role
$\exists hasDebtor.\top \sqsubseteq Commitment$	$\top \sqsubseteq \forall hasDebtor.Agent$	$hasDebtor^- \equiv isDebtorIn$
$\exists hasCreditor.\top \sqsubseteq Commitment$	$\top \sqsubseteq \forall hasDebtor.Agent$	$hasCreditor^- \equiv isCreditorIn$

Modelling states of the EscrowPurchase commitment

EP is a commitment between a debtor of type *Customer* and a creditor *Merchant* agent that has conveyed a quote towards everybody ($\top$):

$$EP \equiv Commitment \sqcap \exists hasDebtor.Customer \sqcap \exists hasCreditor.(Merchant \sqcap \exists quote.\top)$$

DetachEP is an *EP* commitment that has a debtor an agent that has paid escrow towards an *Escrow* agent:

$$DetachEP \equiv EP \sqcap \exists hasDebtor.(\exists payEscrow.Escrow)$$

DischargeEP is a *DetachEP* commitment and has as debtor someone (i.e. customer) that has received a shipment from a shipper agent:

$$DischargeEP \equiv DetachEP \sqcap \exists hasDebtor.(\exists ship^{-}.Shipper)$$

Note that: $DischargeEP \sqsubseteq DetachEP \sqsubseteq EP \sqsubseteq Commitment$
The current state of a specific $c : Commitment$ is the most specific concept to which it belongs.

Modelling states of the EscrowPurchase commitment

EP is a commitment between a debtor of type *Customer* and a creditor *Merchant* agent that has conveyed a quote towards everybody ($\top$):

$$EP \equiv Commitment \sqcap \exists hasDebtor.Customer \sqcap \exists hasCreditor.(Merchant \sqcap \exists quote.\top)$$

DetachEP is an *EP* commitment that has a debtor an agent that has paid escrow towards an *Escrow* agent:

$$DetachEP \equiv EP \sqcap \exists hasDebtor.(\exists payEscrow.Escrow)$$

DischargeEP is a *DetachEP* commitment and has as debtor someone (i.e. customer) that has received a shipment from a shipper agent:

$$DischargeEP \equiv DetachEP \sqcap \exists hasDebtor.(\exists ship^{-}.Shipper)$$

Note that: $DischargeEP \sqsubseteq DetachEP \sqsubseteq EP \sqsubseteq Commitment$
The current state of a specific $c : Commitment$ is the most specific concept to which it belongs.

Modelling states of the EscrowTransfer commitment

ET is a commitment between: i) a creditor *Escrow* agent that has received a *payEscrow* from a customer, and ii) a debtor *Merchant* agent that is creditor in a *DetachEP* commitment:

$$ET \equiv Commitment \sqcap \exists hasCreditor.(Escrow \sqcap \exists pasEscrow^{-}.Customer) \sqcap \exists hasDebtor.(Merchant \sqcap \exists isCreditorIn.DetachEP)$$

DetachET is an *ET* commitment that has as debtor someone who paid an escrow towards an *Escrow* agent:

$$DetachET \equiv ET \sqcap \exists hasDebtor.(\exists isCreditorIn.DischargeEP)$$

DischargeET is a *Detach* commitment that has as creditor someone who payed a transfer towards a *Merchant* agent

$$DischargeET \equiv DetachET \sqcap \exists hasCreditor.(\exists payTransfer.Merchant)$$

Abox streams in the Purchase protocol

Abox stream	Input knowledge	Inferred knowledge
Init	c_1:*Customer*, m_1:*Merchant*, e_1:*Escrow*, s_1:*Shipper*, (ep_1, m_1):*hasCreditor*, (ep_1, c_1):*hasDebtor*, (et_1, e_1):*hasCreditor*, (et_1, m_1):*hasDebtor*	ep_1:*Commitment*, et_1:*Commitment*
A_0	(m_1, c_1):*quote*	ep_1 : *EP*
A_1	(c_1, e_1):*payEscrow*	ep_1:*DetachEP*, et_1:*ET*
A_2	(s_1, c_1):*ship*	ep_1:*DischargeEP*, et_1:*DetachET*
A_3	(e_1, m_1):*payTransfer*	et_1:*DischargeEP*

Tbox signature

```
(full-reset)
(undefine-all)
(in-tbox EscrowPurchaseProtocol)
(set-unique-name-assumption t)
(signature
 :atomic-concepts
    (Agent Customer Merchant Escrow  Shipper
    Commitment  EP DetachEP DischargeEP  ET  DetachET DischargeET)
 :roles
   ((quote  :domain Merchant   :range Customer :parents action)
    (payEscrow     :domain Customer   :range Escrow   :parents action)
    (ship          :domain Shipper    :range Customer :parents action)
    (payTransfer   :domain Escrow     :range Merchant :parents action)
    (reqShip       :domain Merchant   :range Shipper  :parents action)
    (fMEquote      :domain Merchant   :range Escrow   :parents message)
    (fCMpayEscrow :domain Customer   :range Merchant :parents message)
    (fSMship       :domain Shipper    :range Merchant :parents message)
    (fSEship       :domain Shipper    :range Escrow   :parents message)

    (hasDebtor     :domain Commitment :range Agent :inverse isDebtorIn)
    (hasCreditor   :domain Commitment :range Agent :inverse
        isCreditorIn))
  :attributes (window)
  :individuals (c1 m1 e1 s1 ep1 et1))
```

Terminological box for the Escrow Protocol

- Partition desing pattern

```
(equivalent Agent (or Customer Merchant Escrow Shipper))
(disjoint Customer Merchant Escrow Shipper)
(disjoint Agent Commitment)  ;simplifies reasoning
```

- EP is a commitment between a creditor Merchant agent that has conveyed a quote towards everybody (top) and a debtor of type Customer

```
(equivalent EP (and Commitment
                    (some hasCreditor (and Merchant (some quote top)))
                    (some hasDebtor Customer)))
```

- DetachEP is an EP commitment that has a debtor an agent that has paid escrow towards an Escrow agent

```
(equivalent DetachEP (and EP
                          (some hasDebtor (some payEscrow Escrow))))
```

- DischargeEP is a DetachEP commitment and has as debtor someone (the customer) that received a shipment from a shipper agent

```
(equivalent DischargeEP (and DetachEP
                             (some hasDebtor (some (inv ship) Shipper))))
```

Terminological box for the Escrow Protocol

ET is a commitment between a creditor Escrow agent that has received a payEscrow from a Customer and a debtor Merchant agent that is creditor in a DetachEP commitment

```
(equivalent ET (and Commitment
     (some hasCreditor (and Escrow (some (inv payEscrow) Customer)))
     (some hasDebtor   (and Merchant (some isCreditorIn DetachEP)))))
```

DetachET is an ET commitment that has as debtor someone who paid an Escrow towards an Escrow agent

```
(equivalent DetachET (and ET
                          (some hasDebtor (some isCreditorIn DischargeEP))))
```

DischargeET is a Detach commitment that has as creditor someone who payed a transfer towards a Merchant agent

```
    (equivalent DischargeET (and DetachET (some hasCreditor (some
        payTransfer Merchant))))
```

Assertional boxes

```
(in-abox INITABOX EscrowPurchaseProtocol)
    (instance c1 Customer)          (instance m1 Merchant)
    (instance e1 Escrow)            (instance s1 Shipper)
    (instance ep1 Commitment)       (instance et1 Commitment)
    (related ep1 m1 hasCreditor) (related ep1 c1 hasDebtor)
    (related et1 e1 hasCreditor) (related et1 m1 hasDebtor)

 (clone-abox INITABOX :new-name A0)
    (related m1 c1 quote)
    (racer-read-file "queries.racer")

 (clone-abox A0 :new-name A1)
    (related c1 e1 payEscrow)
    (racer-read-file "queries.racer")

 (clone-abox A1 :new-name A2)
    (related s1 c1 ship)
    (racer-read-file "queries.racer")

 (clone-abox A2 :new-name A3)
    (related e1 m1 payTransfer)
    (racer-read-file "queries.racer")
```

Queries

```
(current-abox)
(abox-consistent?)
(evaluate (format t "Queries"))
(concept-instances EP)
(concept-instances DetachEP)
(concept-instances DischargeEP)
(concept-instances ET)
(concept-instances DetachET)
(concept-instances DischargeET)

(individual-types ep1)
(individual-types et1)
(individual-direct-types ep1)  ; current state of the ep1 commitment
(individual-direct-types et1)  ; current state of the et1 commitment
```

```
(full-reset)                                  --> OKAY-FULL-RESET
(in-tbox EscrowPurchaseProtocol)              --> EscrowPurchaseProtocol
(set-unique-name-assumption T)                --> T
(tbox-coherent?)                              --> T
(tbox-cyclic?)                                --> NIL
(in-abox initabox EscrowPurchaseProtocol)     --> initabox
```

Ontology streams

```
(clone-abox initabox new-name A0) --> A0
(current-abox)                    --> A0
(abox-consistent?)                --> T
(concept-instances EP)            --> (EP1)
(concept-instances DETACHEP)      --> NIL
(concept-instances DISCHARGEEP)   --> NIL
(concept-instances ET)            --> NIL
(concept-instances DETACHET)      --> NIL
(concept-instances DISCHARGEET)   --> NIL
(individual-types EP1)            --> ((EP) (COMMITMENT) (*TOP* TOP))
(individual-types ET1)            --> ((COMMITMENT) (*TOP* TOP))
(individual-direct-types EP1)     --> ((EP))
(individual-direct-types ET1)     --> ((COMMITMENT))
```

```
(clone-abox A0 new-name A1)       --> A1
(current-abox)                    --> A1
(abox-consistent?)                --> T
(concept-instances EP)            --> (EP1)
(concept-instances DETACHEP)      --> (EP1)
(concept-instances DISCHARGEEP)   --> NIL
(concept-instances ET)            --> (ET1)
(concept-instances DETACHET)      --> NIL
(concept-instances DISCHARGEET)   --> NIL
(individual-types EP1)            --> ((DETACHEP) (EP) (COMMITMENT) (*TOP* TOP))
(individual-types ET1)            --> ((ET) (COMMITMENT) (*TOP* TOP))
(individual-direct-types EP1)     --> ((DETACHEP))
(individual-direct-types ET1)     --> ((ET))
```

Ontology streams

```
(clone-abox A1 new-name A2)          --> A2
(current-abox)                       --> A2
(abox-consistent?)                   --> T
(concept-instances EP)               --> (EP1)
(concept-instances DETACHEP)         --> (EP1)
(concept-instances DISCHARGEEP)      --> (EP1)
(concept-instances ET)               --> (ET1)
(concept-instances DETACHET)         --> (ET1)
(concept-instances DISCHARGEET)      --> NIL
(individual-types EP1)               --> ((DISCHARGEEP) (DETACHEP) (EP) (COMMITMENT) (*TOP* TOP))
(individual-types ET1)               --> ((DETACHET) (ET) (COMMITMENT) (*TOP* TOP))
(individual-direct-types EP1)        --> ((DISCHARGEEP))
(individual-direct-types ET1)        --> ((DETACHET))
```

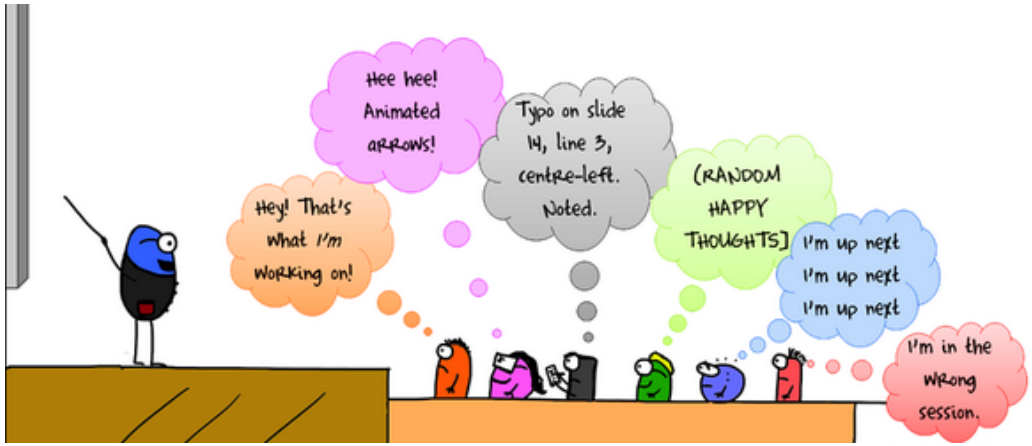
```
(clone-abox A2 new-name A3)          --> A3
(current-abox)                       --> A3
(abox-consistent?)                   --> T
(concept-instances EP)               --> (EP1)
(concept-instances DETACHEP)         --> (EP1)
(concept-instances DISCHARGEEP)      --> (EP1)
(concept-instances ET)               --> (ET1)
(concept-instances DETACHET)         --> (ET1)
(concept-instances DISCHARGEET)      --> (ET1)
(individual-types EP1)               --> ((DISCHARGEEP) (DETACHEP) (EP) (COMMITMENT) (*TOP* TOP))
(individual-types ET1)               --> ((DISCHARGEET) (DETACHET) (ET) (COMMITMENT) (*TOP* TOP))
(individual-direct-types EP1)        --> ((DISCHARGEEP))
(individual-direct-types ET1)        --> ((DISCHARGEET))
```

Conclusion

- We modelled interaction protocols in Description Logics
  - the transition between states is modelled as a navigation in the knowledge graph from generic to specific concepts
  - the current state of the commitment is the most specific concept
  - the formalisation is straithforward simulated in Racer reasoner
- The agents can
  - keep their domain knowledge and interaction protocols in DL
  - in case of inconsistency, the agent can signal to the human agent the reason of inconsistency

Conclusion

- We modelled interaction protocols in Description Logics
  - the transition between states is modelled as a navigation in the knowledge graph from generic to specific concepts
  - the current state of the commitment is the most specific concept
  - the formalisation is straithforward simulated in Racer reasoner
- The agents can
  - keep their domain knowledge and interaction protocols in DL
  - in case of inconsistency, the agent can signal to the human agent the reason of inconsistency



$\exists participates.\{thisSession\} \sqsubseteq \exists thank^{-}.\{I\}$