

# A Communication Acts Ontology for Software Agents Interoperability

Jesús Bermúdez, Alfredo Goñi, Arantza Illarramendi,  
Miren I. Bagüés y Alberto Tablado

Grupo BDI

Universidad del País Vasco

FOCA 2006. Málaga

# Where do we come from?



- Aingeru is a healthcare telemonitoring agent-based system
- Aingeru agents need to communicate with heterogeneous agents
  - Independently developed
  - Different agent communication languages

# Motivation



Right Now!

KQML-based

FIPA-based

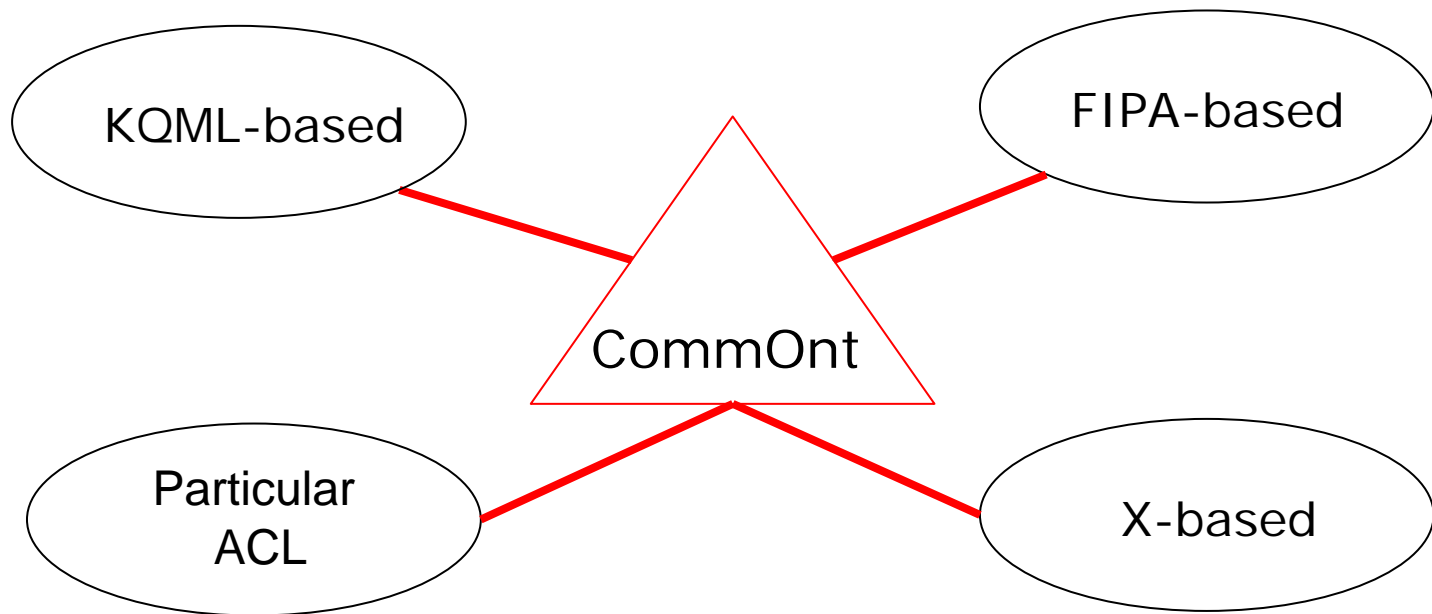
Particular  
ACL

X-based

# Motivation



Where we want to arrive at!



Exchangeability of communication acts

# CommOnt design criteria



- Speech Acts Theory
  - A communication act is basically composed of an intention and a content
- Social commitments approach
  - Objective and verifiable semantics
- Axiomatization of communication acts with Event Calculus
- Materialization of CommOnt: OWL ontology

# Speech acts



- A communication acts has two main components:
  - Attitude of the sender
  - Content
- Different kinds of attitudes and contents lead to different classes of communication acts in CommOnt

# Commitments



- Commitments
  - $C(x, y, p)$
- Conditional commitment
  - $CC(x, y, c, p)$

# Event Calculus



- First-order theory for reasoning about actions
- **Events** (actions) initiate and terminate fluents
- **Fluents** are propositions whose value is subject to change over time

# Event Calculus predicates



1.  $Initiates(a, f, t)$  means that  $f$  holds after event  $a$  at time  $t$ .
2.  $Terminates(a, f, t)$  means that  $f$  does not hold after event  $a$  at time  $t$ .
3.  $Initially_P(f)$  means that  $f$  holds from time 0.
4.  $Initially_N(f)$  means that  $f$  does not hold from time 0.
5.  $Happens(a, t_1, t_2)$  means that event  $a$  starts at time  $t_1$  and ends at  $t_2$ .
6.  $HoldsAt(f, t)$  means that  $f$  holds at time  $t$ .
7.  $Clipped(t_1, f, t_2)$  means that  $f$  is terminated between  $t_1$  and  $t_2$ .
8.  $Declipped(t_1, f, t_2)$  means that  $f$  is initiated between  $t_1$  and  $t_2$ .

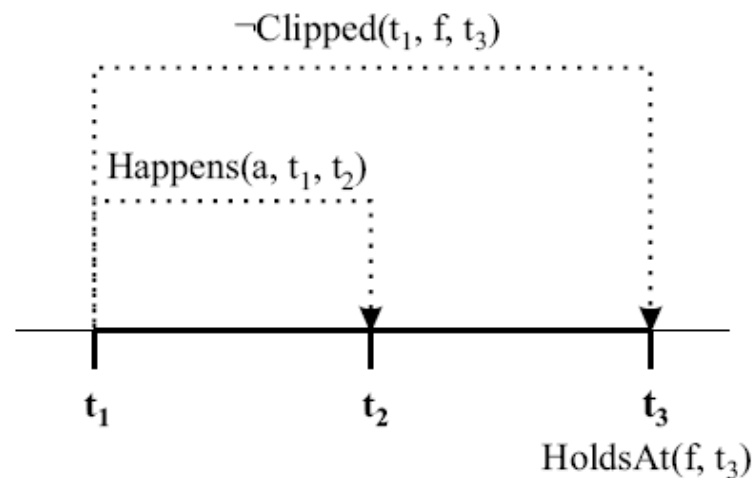
From Pinar Yolum and Munindar P. Singh

# Event Calculus axioms



$$\text{HoldsAt}(f, t_3) \leftarrow \text{Happens}(a, t_1, t_2) \wedge \text{Initiates}(a, f, t_1) \wedge (t_2 < t_3) \wedge \neg \text{Clipped}(t_1, f, t_3)$$

$\text{Initiates}(a, f, t_1)$

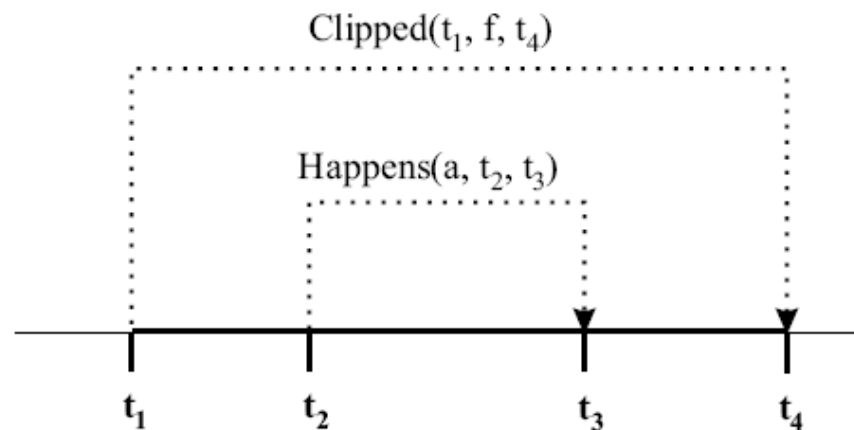


# Event Calculus axioms



$$\text{Clipped}(t_1, f, t_4) \leftrightarrow \exists a, t_2, t_3 [\text{Happens}(a, t_2, t_3) \wedge (t_1 < t_2) \wedge (t_3 < t_4) \wedge \text{Terminates}(a, f, t_2)]$$

Terminates(a, f, t<sub>2</sub>)



# Event Calculus axioms



$$\text{HoldsAt}(f, t) \leftarrow \text{Initially}_P(f) \wedge \neg \text{Clipped}(0, f, t)$$

$$\text{HoldsAt}(f, t_3) \leftarrow \text{Happens}(a, t_1, t_2) \wedge \text{Initiates}(a, f, t_1) \wedge (t_2 < t_3) \wedge \neg \text{Clipped}(t_1, f, t_3)$$

$$\text{Clipped}(t_1, f, t_4) \leftrightarrow \exists a, t_2, t_3 [\text{Happens}(a, t_2, t_3) \wedge (t_1 < t_2) \wedge (t_3 < t_4) \wedge \text{Terminates}(a, f, t_2)]$$

$$\neg \text{HoldsAt}(f, t) \leftarrow \text{Initially}_N(f) \wedge \neg \text{Declipped}(0, f, t)$$

$$\neg \text{HoldsAt}(f, t_3) \leftarrow \text{Happens}(a, t_1, t_2) \wedge \text{Terminates}(a, f, t_1) \wedge (t_2 < t_3) \wedge \neg \text{Declipped}(t_1, f, t_3)$$

$$\text{Declipped}(t_1, f, t_4) \leftrightarrow \exists a, t_2, t_3 [\text{Happens}(a, t_2, t_3) \wedge (t_1 < t_2) \wedge (t_3 < t_4) \wedge \text{Initiates}(a, f, t_2)]$$

$$\text{Happens}(a, t_1, t_2) \rightarrow t_1 \leq t_2$$

# CommOnt materialization



- Axioms and reasoning in the Event Calculus **support** definitions in CommOnt
- OWL constructors and axioms provide a convenient formalism for structure description and concept relationships
- OWL reasoners help the transformation process of communication acts

# Description logics and OWL

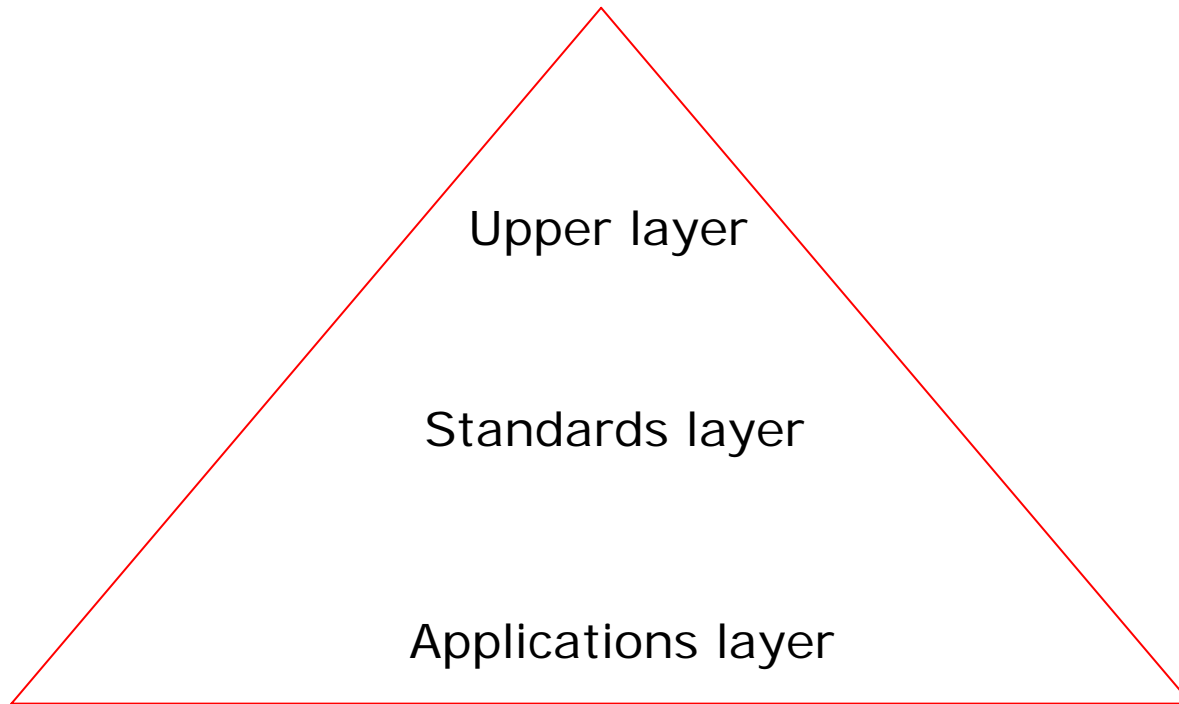


- OWL is founded on a Description Logic
- Description logics are a family of first order logics for describing concepts and properties
- OWL is the Web Ontology Language defined under the auspices of W3C
- OWL axioms:
  - $C \sqsubseteq D$
  - $C \equiv D$

# OWL constructors



OWL constructor name	Logic notation	Semantics
concept name	$A$	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
intersectionOf	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
unionOf	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
complementOf	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
oneOf	$\{o_1, \dots, o_n\}$	$\{o_1, \dots, o_n\}^{\mathcal{I}} = \{o_1^{\mathcal{I}}, \dots, o_n^{\mathcal{I}}\}$
allValuesFrom	$\forall R.C$	$\{d \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(d) \subseteq C^{\mathcal{I}}\}$
someValuesFrom	$\exists R.C$	$\{d \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(d) \cap C^{\mathcal{I}} \neq \emptyset\}$
minCardinality	$\geq nR$	$\{d \in \Delta^{\mathcal{I}} \mid \#(R^{\mathcal{I}}(d)) \geq n\}$
maxCardinality	$\leq nR$	$\{d \in \Delta^{\mathcal{I}} \mid \#(R^{\mathcal{I}}(d)) \leq n\}$
property name	$P$	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
inverseOf	$P^{-}$	$(P^{-})^{\mathcal{I}} = (P^{\mathcal{I}})^{-}$



# CommOnt upper layer



CommunicationAct  $\sqsubseteq$   $\forall$ hasSender.Actor  $\sqcap$  =1.hasSender  $\sqcap$   
 $\forall$ hasReceiver.Actor  $\sqcap$   
 $\forall$ hasContent.Content

Main subclasses:

Assertive, Directive, Commissive, Expressive and Declarative

# CommOnt upper layer



*Initiates*(*Assertive*( $s, r, P$ ),  $C(s, r, P)$ ,  $t$ )

`Assertive`  $\equiv$  `CommunicationAct`  $\sqcap$   
 $\exists$ `hasContent.Proposition`  $\sqcap$   
 $\exists$ `hasCommit.AssertiveCommitment`

*Initiates*(*Directive*( $s, r, P$ ),  $CC(r, s, accept(r, s, P), P)$ ,  $t$ )

`Directive`  $\equiv$  `CommunicationAct`  $\sqcap$   
 $\exists$ `hasContent.Action`  $\sqcap$   
 $\exists$ `hasCommit.DirectiveCommitment`

# CommOnt upper layer



*Initiates(Commissive( $s, r, C, P$ ), CC( $s, r, C, P$ ),  $t$ ))*

Commissive  $\equiv$  CommunicationAct  $\sqcap$   
 $\exists$ hasContent.Action  $\sqcap$   
 $\forall$ hasCondition.Proposition  $\sqcap$   
 $\exists$ hasCommit.CommissiveCommitment

*Initiates(Expressive( $s, r, P, S$ ), CC( $s, r, P, S$ ),  $t$ ))*

Expressive  $\equiv$  CommunicationAct  $\sqcap$   
 $\exists$ hasContent.Proposition  $\sqcap$   
 $\exists$ hasState.PsyState  
 $\exists$ hasCommit.ExpressiveCommitment

# CommOnt upper layer



*HoldsAt(empowered(s), t) → Initiates(Declarative(s, r, P), P, t)*

Declarative  $\sqsubseteq$  CommunicationAct  $\sqcap$   
 $\exists$ hasContent.Proposition

# CommOnt: content ontology



Proposition  $\sqsubseteq$  Content  
    PsyState  $\sqsubseteq$  Proposition  
    Action  $\sqsubseteq$  Content  
CommunicationAct  $\sqsubseteq$  Action  
    RefExpression  $\sqsubseteq$  Content  
    ReportAct  $\sqsubseteq$  Action  $\sqcap$   
         $\exists$ hasQuery.Proposition  $\sqcup$  RefExpression  
    Command  $\sqsubseteq$  Action

# CommOnt upper layer (cont.)



- It is interesting to define subclasses of the main five

`Inquiry ≡ Directive ⊓ ∃hasContent.ReportAct`

`Request ≡ Directive ⊓ ∃hasContent.Command`

`Responsive ≡ Assertive ⊓ ∃inReplyTo.Inquiry`

# CommOnt standards layer



FIPA-Inform  $\sqsubseteq$  Assertive  
FIPA-Confirm  $\sqsubseteq$  Assertive  
FIPA-Disconfirm  $\sqsubseteq$  Assertive  
FIPA-Request  $\sqsubseteq$  Directive

FIPA-Query-If  $\equiv$  Inquiry  $\sqcap$  FIPA-Request  $\sqcap$   
=1.hasContent  $\sqcap$   $\forall$ hasContent.FIPA-Inform-If  
FIPA-Inform-If  $\equiv$  ReportAct  $\sqcap$   
 $\exists$ hasContent<sup>-</sup>.FIPA-Request  $\sqcap$   
 $\forall$ hasQuery.Proposition

# CommOnt standards layer



KQML-Tell  $\sqsubseteq$  Assertive  
KQML-Ask-If  $\sqsubseteq$  Inquiry  $\sqcap$   
 $\forall$ hasContent. ( $\forall$ hasQuery.Proposition)

- Very important for the interoperability goal

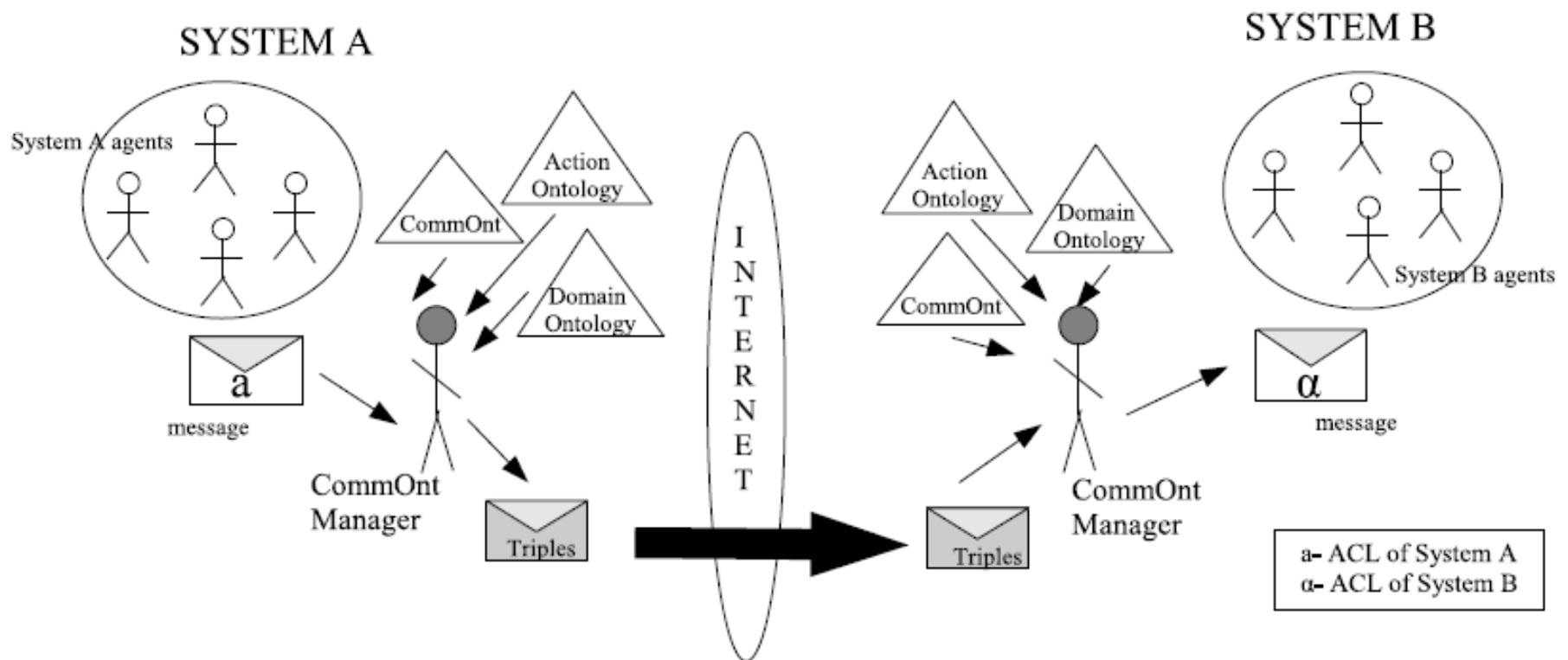
KQML-Ask-If  $\equiv$  FIPA-Query-If  
KQML-Achieve  $\sqsubseteq$  FIPA-Request  
FIPA-Inform  $\equiv$  KQML-Tell  
KQML-Achieve  $\equiv$  FIPA-Request  $\sqcap$   $\exists$ hasContent.Achieve

# CommOnt applications layer



```
MedicineModify ≡ Request ⊓ =1.hasContent ⊓  
                ∀hasContent.(Overwrite ⊓ ∃hasSubject.Medicine)  
LocationQuery  ≡ Inquiry ⊓ =1.hasContent ⊓  
                ∀hasContent.(  
                ∀hasQuery.(RefExpression ⊓ ∃hasSubject.Location))  
VitalSignInform ≡ Responsive ⊓ =1.hasContent ⊓  
                ∀hasContent.(Proposition ⊓ ∃hasSubject.VitalSignData)
```

# Communication process



# Conclusions



- CommOnt provides a reference framework for relating communication acts.
- CommOnt is an extensible ontology
- CommOnt is adaptable to particular systems



Thank you,  
for your attention!