KQML & FLBC: Contrasting agent communication languages

Scott Moore

University of Michigan Business School
Assistant Professor of Computer & Information Systems
samoore@umich.edu
www-personal.umich.edu/~samoore/

My basic proposal

- Should have a small fixed number of basic message types
- Should have a single message structure
- A message's idiosyncratic meaning builds upon the meaning implied by its type
- Content should have small reusable terms
- Two components: the language (FLBC) and the message management system (MMS)

Scenarios that take advantage of the FLBC

- Companies communicating with automated electronic communication (AEC) with no prior contact
- Companies communicating with AEC almost immediately after setup
  - Think of how systems are prototyped

The language

- Based on Speech Act Theory
- Functional structure: F(P)
  - force applied to a proposition
- Context
  - other information necessary to understand message
  - e.g., conversational structure
- Messages formatted as XML documents

An example

- KQML ask-all performative

```
(ask-all
  :sender A
  :receiver B
  :content X)
```

- Meaning
  - "A wants to know all of B's responses that make X true of B."

Example: Basic structure when using FLBC

```
<?xml … ?>
<!DOCTYPE flbcMsg SYSTEM “http://…” >
<flbcMsg msgID=“Ex#1”>
  <simpleAct speaker=“A” hearer=“B”>
  <illocAct force=“request”>
    [what it is that is requested]
  </illocAct>
  <context>
    [terms needed to interpret message]
  </context>
</flbcMsg>
```

Example: Content of the request (1/2)

```
<andMsg>
  <do>
    <thisPerson who="B"/>
    <evaluate id="Set">
      <bel>
        <thisPerson who="B"/>
        <setOf>
          <truthStatus theStatus="true">
            <predSt>X</predSt>
          </truthStatus>
        </setOf></bel></evaluate></do>
```

Example: Content of the request (2/2)

```
<do>
  <thisPerson who="B"/>
  <sendMsg>
    <flbcMsg>
      <simpleAct speaker="B" hearer="A">
        <illocAct force="inform">
          <aTerm theTerm="Set"/>
        </illocAct></simpleAct>
    </flbcMsg></sendMsg>
</do>
</andMsg>
```

Example: Context of the request

```
<context newConv="yes">
   <resources>
      <actors>
         <person id="A" … />
         <person id="B" … />
      </actors>
   </resources>
   <timeSent> … </timeSent>
</context>
```

Example: FLBC interpretation

- "A requests of B that B evaluate the set of values that B believes makes X true and to inform A of this set"
  - similar to (the same as?) KQML performative
- Could easily be modified to
  - find those values that make it false
  - find just one value or five values or …
  - send the values to someone else
  - ask them to simply evaluate and not inform

Example: Points concerning FLBC (1/2)

- Functional form F(P)
  - `request( and(do(evaluate), do(sendMsg)) )`
- This is a recursive message
  - i.e., $F_1(F_2(X))$
  - e.g., `request(inform(X))`
- Much more detailed content than KQML message

Example: Points concerning FLBC (2/2)

- Here, the act is a “request”
- Set used by FLBC hasn’t changed in 5 years
- Categorized so far
  - UN/EDIFACT EDI messages
  - AppleEvents in the AERegistry
  - A set of proprietary EDI messages

  u All 30 standard KQML performatives

Results of translating all standard KQML performatives

- 10 of the existing FLBC illocutionary forces were needed to represent the 30 KQML performatives
  - These forces (and their associated meanings) ***already existed***
- 18 of the KQML performatives were represented by an *inform* and/or a *request*
- Every content predicate but one (`subscribe`) was re-used

Different underlying communication models

- KQML
  - Preconditions imply that agents must `advertise` the messages they are interested in receiving
- FLBC
  - Sender of a message can assume only that recipient knows what sender intends for the message to accomplish
    - e.g., sender does not know if an `assert(X)` will result in a `believe(X)`

Benefits of using the FLBC

- Can do more things easily (i.e., broader participation) in the electronic marketplace
- Lower start-up costs when adding new communication partners
- Lower marginal costs when adding new capabilities

Broader participation

- Automatic reordering systems
- Join electronic value chains
- Use electronic agents for shopping
- Create a broader web of communication partners
- More automated communication among and within
  - Trading partners, vendors, customers, electronic agents

Low start-up costs for automated communication

- Large companies
  - More firms to market or sell to
  - More firms to buy from
- Small companies
  - New markets
  - New ways of doing business
- Individuals
  - New communication channel

Lower marginal costs when adding new capabilities

- Message construction from existing parts
- Very few (or **no**) new message types needed
- Message handlers are partly reusable