Process Integration: Look at How You Behave!

Giuseppe De Giacomo
SAPIENZA Università di Roma, Italy
degiacomo.uniromal.it
Suppose processes/services act on an integrated view of the world…

- Messages between services
- Impact on “real world” — modeled as relations or FOL
- “View” of internal process model — automata based

[BCDHM-VLDB05]
Process/service behavior of an abstract machine querying and acting on the world...

- **Local store**
- **Edge conditions based on local store (and incoming message)**
- **Edge actions**
  - Atomic Process
  - acting on the world
  - set the local store
  - Create/send message
  - Read message

```
requestOrder(payBy,cartNum,addr,price)
(payBy == PREPAID) \&\& (price \leq 10) / charge(cartNum; paymentOK)
(payBy == CC) \lor (price > 10) / ! requestCCCheck(cartNum)

requestCCCheck(approved)
(approved == F) / ! replyCCCheck(“fail”)
(approved == T) / requestShip(wh,addr; oid,date,status)

requestShipStatus(oid)
shipStatus(oid; date,status)
checkShipStatus(oid; date,status)
! replyOrder(“fail”)
```
Behavior-based composition

Key points

- No available process for the target behavior
- Must realize target behavior using fragments of available behaviors

Target behavior
Spec. of the desired process behavior expressed in terms of virtual actions

Environment
Spec. of atomic processes and data

Available behaviors
Spec. of the behavior of available processes/services expressed in terms of the environment

Actual available behaviors

[ICSOC03, VLDB05, IJCAI07, AAAI07]
Realizing target behavior via a composition

If behavior + environment is finite state …
… then we can automatically synthesize the controller!
Example

controller

NextAct:
act = a
    delegate 1
act = b
    delegate 2
act = c
    if (state2 = S21) delegate 2
    else delegate 1

NextState: --  //it’s stateless

Example

controller

NextAct:
act = a
    delegate 1
act = b
    delegate 2
act = c
    if (state2 = S21) delegate 2
    else delegate 1

NextState: --  //it’s stateless
Example

**target**

```
controller

NextAct:
- act = a
- delegate 1
- act = b
- delegate 2
- act = c
- if (state2 = S21) delegate 2
  else delegate 1

NextState: -- //it's stateless
```

---

**behavior 1**

```
act = a
delegate 1
act = b
delegate 2
act = c
if (state2 = S21) delegate 2
else delegate 1
```

---

**behavior 2**

```
NextAct:
- act = a
- delegate 1
- act = b
- delegate 2
- act = c
- if (state2 = S21) delegate 2
  else delegate 1

NextState: -- //it's stateless
```
Example

controller

NextAct:
- act = a
delegate 1
- act = b
delegate 2
- act = c
  if (state2 = S21) delegate 2

controller automatically synthesized!
Basic issues

We have all the issues of data integration but in addition …

• Behavior: description of the dynamics of the process!
• Behavior should be formally and abstractly described: conceptual modeling of dynamics (not a la OWL-S). Which?
  - Workflows community may help
  - Business process community may help
  - Services community may help
  - Process algebras community may help
  - Reasoning about actions community may help
  - DB community may help
  - …may help
• Behavior + environment should be even more abstractly described for analysis/synthesis: must be finite state.
  - CAV (Computer Aid Verification) community can help!
• How do we map behaviors one into the other?
  - ??? may help
• …