Runtime Verification of Contracts for Java Programs

Christian Colombo
Gordon J. Pace
Gerardo Schneider

FLACOS - November 2008
RV & Contracts

- In SOA we are concerned with security and trust.
- Model checking is not scalable.
- Testing lacks coverage.
- Particular behaviour only emerges during normal use of service composition.
- Runtime verification monitors the behaviour during runtime, scales up.
- Real-time properties / overheads.
- Contracts may have conflicts.
Runtime Verification

MONITORED SYSTEM

FEEDBACK

EVENTS

VERIFYING SYSTEM

Specification

---

2008
Runtime Verification

- MONITORED SYSTEM
- FEEDBACK
- EVENTS
- VERIFYING SYSTEM
- Contracts
Runtime Verification

MONITORED SYSTEM

FEEDBACK

EVENTS

VERIFYING SYSTEM

CONTRACT CONFLICT ANALYSIS

Deontic Contracts

---
Dynamic Automata with Timers & Events (DATE)

- Communicating symbolic automata enriched with **events** and **timers**.
- Automata are automatically replicated according to context: hence **dynamic**.
- Supports:
  - Conditions and actions on transitions
  - Real-time
  - Communication between automata
An Example (1)
An Example (2)

```
interact\treset();

logged in

goodlogin\treset();

logged out

logout\tc = 0;
t@30*60\tlogout();c = 0;

badlogin\tc += 2
\tc = 0;blockUser();
```
LARVA - Architecture

SYSTEM

AspectJ Matching method names

FEEDBACK

EVENTS

REPORT

USER

MONITORING SYSTEM
(SYMBOLIC AUTOMATON EXECUTION)
Contract Language Example

\[ \overline{\text{login}}^* F(request) \]
\[ \land [1^*][\text{logout}][\text{login}^*] F(request) \]
\[ \land F(request) \]
Contract Language to Automata
Contract Language to LARVA

EVENTS {
    login = {*.login()}
    logout = {*.logout()}
    request = {*.requestItem()}
}

TRANSITIONS {
    Init -> S1 [login]
    Init -> V [request]
    Init -> S2 [logout]
    S1 -> S1 [login]
    S1 -> S1 [request]
    S1 -> S2 [logout]
    S2 -> S2 [logout]
    S2 -> V [request]
    S2 -> S1 [login]
}

PROPERTY clcontract {
    STATES {
        BAD { V }
        NORMAL { S1 S2 }
        STARTING { Init }
    }
}

2008
Contradictions in Contracts

\[ [\text{login}^*]F(\text{request}) \]
\[ \land [1^*][\text{logout}][\text{login}^*]F(\text{request}) \]
\[ \land F(\text{request}) \]

\( O(\text{request}) \)

Contradiction Detected!
Ongoing Work

- Working closely with industry
- Guarantees on the effect of monitoring – memory and time
- Identifying better notations
- Investigating compensable actions
Conclusions

- Mathematical framework – DATE
- Implemented useable tool – LARVA
- Highly expressive (incl. real-time)
- Evolving theory with practical guarantees
- Can monitor contracts
- Find contradictions in contracts
- Future prospects of collaboration and improvement of current framework
Questions

?