Compositional Reasoning about Concurrent Objects and Futures

Crystal Chang Din, Johan Dovland, Olaf Owe

Precise Modeling and Analysis group (PMA), University of Oslo

COST Action IC0701 9th MC and WG Meeting
Darmstadt, Germany, 29 Feb. 2012
Introduction

- The Abstract Behavioral Specification Language (ABS) is an imperative and object-oriented language.
- The concurrent object model of ABS is inherently compositional.
- Object communication is only by asynchronous method calls and futures.
- Compositional reasoning is by means of class invariants over histories.
Outline

1. asynchronous method calls and futures
2. Invariants and histories
3. Reasoning system in dynamic logic
4. Summary
What is future?
We revise the four-event semantics of earlier work to deal with futures, allowing several readings of the same future, possibly by different objects.
ABS syntax and semantics for futures (kernel language)

- \( fr := o!m(e^*) \)
  The caller generates a fresh future identity and continues without waiting.

- \( v := fr? \)
  The caller blocks while waiting for the future to be resolved.

- \textbf{await} \( v := fr? \)
  The caller suspends while waiting for the future to be resolved.
Example in the ABS language

class Adm(Prof p) {
    String req() {
        Fut<String> mb, fr; String cv;
        fr := p!ask(); mb := fr?; cv := mb?; return cv
    }
}

class Prof {
    Stud s; {s := new Stud();}
    Fut<String> ask() {
        Fut<String> fr; fr := s!reply(); return fr
    }
}

class Stud {
    String reply() {
        return "cv"
    }
}
An example history (a possible sequence of events)

\[
\langle _\rightarrow a, u_1, req, \varepsilon \rangle, \langle a \rightarrow p, u_2, ask, \varepsilon \rangle, \langle a \rightarrow p, u_2, ask, \varepsilon \rangle, \\
\langle p \rightarrow s, u_3, reply, \varepsilon \rangle, \langle \leftarrow p, u_2, ask, u_3 \rangle, \langle p \rightarrow s, u_3, reply, \varepsilon \rangle, \\
\langle a \leftarrow, u_2, u_3 \rangle, \langle \leftarrow s, u_3, reply, "cv" \rangle, \langle a \leftarrow, u_3, cv \rangle, \langle \leftarrow a, u_1, req, cv \rangle
\]

class Adm(Prof p) {
    String req() {Fut<String> fr; String cv;
            fr := p!ask(); mb := fr?; cv := mb?; return cv}
}
class Prof {
    Stud s; {s := new Stud();}
    Fut<String> ask() { Fut<String> fr; fr := s!reply(); return fr}
}
class Stud {
    String reply() {return "cv"}}
Outline

1. asynchronous method calls and futures

2. Invariants and histories

3. Reasoning system in dynamic logic

4. Summary
Class invariants

\[ I_{Adm}(p)(h) \triangleq h \leq \]
\[ [\langle c \rightarrow this, d, req, \varepsilon \rangle, \langle this \rightarrow p, u, ask, \varepsilon \rangle, \langle this \leftarrow u, u' \rangle, \]
\[ \langle this \leftarrow u', cv \rangle, \langle \leftarrow this, d, req, cv \rangle. \]
\[ \text{some } c, d, u, u' \] *

\[ I_{Prof}(s, h) \triangleq h \leq \]
\[ [\langle c \rightarrow this, u, ask, \varepsilon \rangle, \langle this \rightarrow s, u', reply, \varepsilon \rangle, \langle \leftarrow this, u, ask, u' \rangle. \]
\[ \text{some } c, u, u' \] *

\[ I_{Stud}(h) \triangleq h \leq \]
\[ [\langle c \rightarrow this, u, reply, \varepsilon \rangle, \langle \leftarrow this, u, reply, "cv" \rangle. \]
\[ \text{some } c, u \] *

The order of the history events must be obeyed locally but can be interleaved in the global setting according to the mechanisms of asynchronous method calls.
Object invariants

\( l_a: \text{Adm}(p)(h_a) \triangleq h_a \leq \)
\[ \langle \_ \rightarrow a, d, \text{req}, \varepsilon \rangle, \langle a \rightarrow p, u, \text{ask}, \varepsilon \rangle, \langle a \leftarrow, u, u' \rangle, \]
\[ \langle a \leftarrow, u', \text{cv} \rangle, \langle \leftarrow a, d, \text{req}, \text{cv} \rangle. \]
\text{some } d, u, u' \]

\( l_p: \text{Prof}(h_p) \triangleq \exists s. h_p \leq \)
\[ \langle \_ \rightarrow p, u, \text{ask}, \varepsilon \rangle, \langle p \rightarrow s, u', \text{reply}, \varepsilon \rangle, \langle \leftarrow p, u, \text{ask}, u' \rangle. \]
\text{some } u, u' \]

\( l_s: \text{Stud}(h_s) \triangleq h_s \leq \)
\[ \langle \_ \rightarrow s, u, \text{reply}, \varepsilon \rangle, \langle \leftarrow s, u, \text{reply}, " \text{cv}" \rangle. \]
\text{some } u \]
Global invariant of the objects $a$, $p$ and $s$

From the class invariants above, we can derive the following global invariant for the subsystem of $s : Stud, p : Prof, a : Adm(p)$:

$$l_{\{a, p, s\}}(H) \triangleq wf(H, \{a, p, s\}) \land l_{a:Adm(p)}(H/\{a\}) \land l_{p:Prof}(H/\{p\}) \land l_{s:Stud}(H/\{s\})$$

where wellformedness allows us to relate the different object histories.
Reasoning about the example

For the example system we may derive that the CV received by the Administrator is the one provided by the Student.

Diagram:
- Administrator
- Future
- Mailbox
- Professor
- CV
- Student

Steps:
1. Professor
2. Mailbox
3. Future
4. Administrator
Outline

1. asynchronous method calls and futures

2. Invariants and histories

3. Reasoning system in dynamic logic

4. Summary
Dynamic logic formula

\[ \psi \Rightarrow [s]\phi \]
(1) $\Gamma_C \vdash (\mathcal{H} = \langle parent(this) \to this, C, \bar{e} \rangle) \Rightarrow [\text{init}_C] l_C(\overline{w}, \mathcal{H})$

(2) $\Gamma_C \vdash (\text{wf}(\mathcal{H}) \land l_C(\overline{w}, \mathcal{H})) \Rightarrow [\mathcal{H} := \mathcal{H} \cdot \text{startEv}; \text{body}_m; \mathcal{H} := \mathcal{H} \cdot \text{endEv};](\text{wf}(\mathcal{H}) \Rightarrow l_C(\overline{w}, \mathcal{H}))$

(3) $\Gamma_C \vdash (\text{wf}(\mathcal{H}) \land P(\overline{w}, \mathcal{H})) \Rightarrow [\mathcal{H} := \mathcal{H} \cdot \text{startEv}; \text{body}_m; \mathcal{H} := \mathcal{H} \cdot \text{endEv};](\text{wf}(\mathcal{H}) \Rightarrow Q(\overline{w}, \mathcal{H}))$
Two dynamic logic rules for ABS statements

\[
\begin{align*}
\vdash \forall fr'. \{ \mathcal{H} := \mathcal{H} \cdot \langle \text{this} \rightarrow o, fr', m, \overline{e} \rangle || fr := fr' \} (wf(\mathcal{H}) \Rightarrow [s] \phi) \\
\vdash [fr = o! m(\overline{e}); s] \phi
\end{align*}
\]

\[
\begin{align*}
\vdash \forall v'. \{ \mathcal{H} := \mathcal{H} \cdot \langle \text{this} \leftarrow, fr, v' \rangle || v := v' \} (wf(\mathcal{H}) \Rightarrow [s] \phi) \\
\vdash [v := fr?; s] \phi
\end{align*}
\]
Outline

1. asynchronous method calls and futures
2. Invariants and histories
3. Reasoning system in dynamic logic
4. Summary
Conclusion

- A compositional reasoning system for concurrent objects and futures
- A reasoning system for the ABS language in Dynamic Logic
- Futures are handled by events appearing in the histories
- Classes can be specified independently from the surroundings
- Modularity is achieved
- Global specification is realized from composing local specifications
Future Work

- Implementation in KeY
- Semi-automatic verification
- Large case studies
Comparison to Related Work

- **Observable behavior of distributed systems: Component reasoning for concurrent objects.**
  - We revise the four-event semantics to deal with futures, allowing several readings of the same future, possibly by different objects.

- **A system for compositional verification of asynchronous objects.**
  - They present a compositional verification system in dynamic logic for Creol, but without futures.
  - The denotational semantics features the similar four communication events.
  - However, the reasoning system is based on the two-event semantics, which requires more complex rules.

- **A complete guide to the future.**
  - Futures are treated as visible objects, rather than our approach where futures are reflected by events in histories.
  - Global reasoning about futures is obtained by means of global invariants, rather than by compositional rules.
  - The environment of a class does not need to be known at the verification time in our system.