FOAM
A Lightweight Method for Verification of Use-Cases

Viliam Šimko
Tomáš Bureš
Petr Hnětynka
František Plášil

{simko, bures, hnetynka, plasil}@d3s.mff.cuni.cz

D3S Seminar, April 24, 2012
Goals

- Requirements specification in a changing environment
- Specification of \textit{temporal dependencies} among use-cases.
  - using \textit{temporal formulae} encapsulated as annotations
- Verification on the specification level
  - using \textbf{NuSMV} model checker
Motivation: temporal dependency

Use-Case $U_1$: Buyer Places Bid On Item

Main success scenario:
1. Include use-case “Buyer Reviews Item Information”.
2. The buyer notifies the GPM that he/she wants to place a bid.
3. The GPM shall respond by requesting the details about bids from the buyer.
4. The buyer sends a submit bid request to the GPM.
5. The GPM shall respond by sending a notification to the buyer.
6. The buyer sends a notification acknowledgement to the GPM.

Use-Case $U_2$: Buyer Reviews Item Information

Main success scenario:
1. The buyer uses the web page to send a review item information request to the GPM.
2. The GPM displays information about the item.
3. The buyer reviews item information.
Motivation: temporal dependency

Use-Case $U_1$: Buyer Places Bid On Item

Main success scenario:
1. Include use-case “Buyer Reviews Item Information”.
2. The buyer notifies the GPM that he/she wants to place a bid.
3. The GPM shall respond by requesting the details about bids from the buyer.
4. The buyer sends a submit bid request to the GPM.
5. The GPM shall respond by sending a notification to the buyer.
6. The buyer sends a notification acknowledgement to the GPM.

Use-Case $U_2$: Buyer Reviews Item Information

Main success scenario:
1. The buyer uses the web page to send a review item information request to the GPM.
2. The GPM displays information about the item.
3. The buyer reviews item information.

Variation:
2a. The item is not valid
   2a1. The GPM displays a message describing invalid item.
   2a2. Use-case aborted.

Counter-example
Motivation: temporal dependency

Use-Case $U_1$: Buyer Places Bid On Item

Main success scenario:
1. Include use-case “Buyer Reviews Item Information”.
2. The buyer notifies the GPM that he/she wants to place a bid.
3. The GPM shall respond by requesting the details about bids from the buyer.
4. The buyer sends a submit bid request to the GPM.
5. The GPM shall respond by sending a notification to the buyer.
6. The buyer sends a notification acknowledgement to the GPM.

Extension:
1a. The use-case “Buyer Reviews Item Information” was aborted.
1a1. The GPM displays a message “Bid cannot be placed”
1a2. Use-case aborted.

Use-Case $U_2$: Buyer Reviews Item Information

Main success scenario:
1. The buyer uses the web page to send a review item information request to the GPM.
2. The GPM displays information about the item.
3. The buyer reviews item information.

Variation:
2a. The item is not valid
2a1. The GPM displays a message describing invalid item.
2a2. Use-case aborted.
Case Study: Donald Firesmith, Global personal marketplace system requirements specification, 2003, Link
How to formalize this?

**Use-Case U₁:** Buyer Places Bid On Item

**Main success scenario:**
1. Include use-case “Buyer Reviews Item Information”.  
   
2. The buyer notifies the GPM that he/she wants to place a bid.
3. The GPM shall respond by requesting the details about bids from the buyer.
4. The buyer sends a submit bid request to the GPM.  
5. The GPM shall respond by sending a notification to the buyer.
6. The buyer sends a notification acknowledgement to the GPM.

**Extension:**
1a. The use-case “Buyer Reviews Item Information” was aborted.  
1a1. The GPM displays a message “Bid cannot be placed”
1a2. Use-case aborted. #abort

**Use-Case U₂:** Buyer Reviews Item Information

**Main success scenario:**
1. The buyer uses the web page to send a review item information request to the GPM.
2. The GPM displays information about the item.  
3. The buyer reviews item information.

**Variation:**
2a. The item is not valid
2a1. The GPM displays a message describing invalid item.
2a2. Use-case aborted. #abort

**Flow annotations** (describing the structure of an LTS, predefined fixed semantics)
- abort, include, goto, mark, guard

**Temporal annotations** (to be checked)
- create, use, ... (user-defined in TADL)
Temporal Annotation Definition Language

Annotations:
#create:city
#use:city
#create:map

TADL Template:
Annotations: create, use
$\text{CTL AG} (\text{create} \rightarrow \text{EF(use)})$ "Branch with use required after create"
$\text{CTL AG} (\text{create} \rightarrow \text{AX(AG(!create))})$ "Only one create"
$\text{CTL A}[\!\text{use} \text{create} | \!\text{EF(use)}]$ "First create then use"

A set of temporal formulae to be checked

$\text{CTL AG}(\text{create}_\text{city} \rightarrow \text{EF(use}_\text{city}))$
$\text{CTL AG}(\text{create}_\text{city} \rightarrow \text{AX(AG(!create}_\text{city}))}$
$\text{CTL A}[\!\text{use}_\text{city} \text{create}_\text{city} | \!\text{EF(use}_\text{city})]$

$\text{CTL AG}(\text{create}_\text{map} \rightarrow \text{EF(use}_\text{map}))$
$\text{CTL AG}(\text{create}_\text{map} \rightarrow \text{AX(AG(!create}_\text{map}))}$
$\text{CTL A}[\!\text{use}_\text{map} \text{create}_\text{map} | \!\text{EF(use}_\text{map})]$
More TADL Examples

Annotations: create, use

\[ \text{CTL AG}( \text{create} \rightarrow \text{EF}(\text{use})) \] "Branch with use required after create"
\[ \text{CTL AG}( \text{create} \rightarrow \text{AX}(\text{AG}(\neg \text{create}))) \] "Only one create"
\[ \text{CTL A}[\neg \text{use} \text{ U create} \mid \neg \text{EF}(\text{use})] \] "First create then use"

Annotations: open, close — strict ordering of 2 phases

\[ \text{LTL G}(\text{open} \rightarrow \text{F}(\text{close})) \] "After open, close is required"
\[ \text{CTL AG}(\text{open} \rightarrow \text{AX}(\text{A} \mid \neg \text{open} \text{ U close})) \] "No multi—open"
\[ \text{CTL AG}(\text{close} \rightarrow \text{AX}(\text{A} \mid \neg \text{close} \text{ U open} \mid \neg \text{EF}(\text{close}))) \] "No multi—close"
\[ \text{CTL A}[\neg \text{close} \text{ U open} \mid \neg \text{EF}(\text{close})] \] "First open then close"

Annotations: init, process, release — strict ordering of 3 phases

— init \rightarrow process

\[ \text{CTL A}[\neg \text{process} \text{ U init} \mid \neg \text{EF}(\text{process})] \] "First init then process"
\[ \text{CTL AG}(\text{init} \rightarrow \text{AF}(\text{process})) \] "After init there should always be process"
\[ \text{CTL AG}(\text{init} \rightarrow \text{AX}(\text{A} \mid \neg \text{init} \text{ U process})) \] "No multi—init without process"
\[ \text{CTL AG}(\text{process} \rightarrow \text{AX}(\text{A} \mid \neg \text{process} \text{ U init} \mid \neg \text{EF}(\text{process}))) \]
"No multi—process without init"

— process \rightarrow release

\[ \text{CTL A}[\neg \text{release} \text{ U process} \mid \neg \text{EF}(\text{release})] \] "First process then release"
\[ \text{CTL AG}(\text{process} \rightarrow \text{AF}(\text{release})) \] "After process, release is required"
\[ \text{CTL AG}(\text{process} \rightarrow \text{AX}(\text{A} \mid \neg \text{process} \text{ U release})) \]
"No multi—process without release"
\[ \text{CTL AG}(\text{release} \rightarrow \text{AX}(\text{A} \mid \neg \text{release} \text{ U process} \mid \neg \text{EF}(\text{release}))) \]
"No multi—release without process"
**Transformation: Overview**

**TADL**: Temporal Annotation Definition Language (templates)

**OBA**: Overall Behavior Automaton

---

**Diagram**:
- **Use Case Model**
  - Precedence Relation
  - Use-Cases $U_1 \ldots U_n$
  - Flow Annotations
  - Temporal Annotations
  - TADL annotation templates
  - CTL / LTL Formulae
- **OBA**
  - LTS with guards
- **SMV Model**
- **Counter Example**
Overall Behavior Automaton (OBA)

init_0 → succ_0

[ done_1 & ... & done_n ]

[ ¬done_u & G_u^{prec} ]

U_1 → init_u → succ_u → U_n

{ done_u \leftarrow true }

...
Construction of OBA using Inference Rules (1/2)

1. Representing steps

\[
u \in U_M, x \in S_u \\
x^{\text{in}} \rightarrow x^{\text{var}} \rightarrow x^{\text{jump}} \rightarrow x^{\text{ext}} \rightarrow x^{\text{out}}
\]

2. Representing scenarios

\[
u \in U_M, w \in W_u, x_1 \leq_w \ldots \leq_w x_n \\
(x_1^{\text{out}} \rightarrow x_2^{\text{in}}), \ldots, (x_{n-1}^{\text{out}} \rightarrow x_n^{\text{in}})
\]

3. Connecting variations

\[
u \in U_M, w \in W_u, w = \{y_1, \ldots, y_n\}, Var_u(w) = x, \\
G_V = \{g | \langle \text{guard:}g \rangle \in Flow_u(y_1)\} \\

x^{\text{var}} \xrightarrow{G_V} y_1^{\text{in}}
\]

4. Connecting extensions

\[
u \in U_M, w \in W_u, w = \{y_1, \ldots, y_n\}, Ext_u(w) = x, \\
G_E = \{g | \langle \text{guard:}g \rangle \in Flow_u(y_1)\} \\

x^{\text{ext}} \xrightarrow{G_E} y_1^{\text{in}}
\]

5. Continuation from scenarios

\[
u \in U_M, w \in W_u, x = Var_u(w) \lor x = Ext_u(w), \\
w = \{y_1, \ldots, y_n\}, \langle \text{abort} \rangle \notin Flow_u(y_n), \\
\forall s \in S_u \langle \text{goto:}s \rangle \notin Flow_u(y_n) \\
y_n^{\text{out}} \rightarrow x^{\text{out}},
\]

6. Handling GOTO annotations

\[
u \in U_M, x \in S_u, \langle \text{goto:}y \rangle \in Flow_u(x) \\
x^{\text{out}} \rightarrow y^{\text{jump}},
\]

7. Handling ABORT annotations

\[
u \in U_M, x \in S_u, \langle \text{abort} \rangle \in Flow_u(x) \\
x^{\text{out}} \rightarrow x^{\text{out}}
\]
Construction of OBA using Inference Rules (2/2)

8. Handling INCLUDE (procedure call)
\[ u, c \in U_M, x \in S_u, \langle \text{include}:c \rangle \in \text{Flow}_u(x), \]
\[ w^m_c = \{y_1, \ldots, y_n\} \]
\[ x_{\text{jump}} \xrightarrow{\{\text{incl}_u, c \leftarrow \text{true}\}} y^1_{\text{in}}, x_{\text{jump}} \xrightarrow{[\text{false}]} x_{\text{ext}}, \]

9. Handling INCLUDE (return)
\[ u, c \in U_M, x \in S_u, \langle \text{include}:c \rangle \in \text{Flow}_u(x), \]
\[ w^m_c = \{y_1, \ldots, y_n\}, \forall s \in S_c (\langle \text{goto}:s \rangle \notin \text{Flow}_c(y_n) \]
\[ y^1_{\text{out}} \xrightarrow{\{\text{incl}_u, c \leftarrow \text{false}\}} x_{\text{ext}}, \]

10. Scheduler
\[ u \in U^P_M, w^m_u = \{x_1, \ldots, x_n\}, \]
\[ G^\text{prec}_u = \{\text{done}_v | \exists v \in U^P_M (v, u) \in \text{Prev}_M\} \]
\[ \xrightarrow{\{G^\text{prec}_u, \neg \text{done}_u\}} \]
\[ \xrightarrow{\text{init}_0} \]

11. Final state
\[ G = \{\text{done}_u | u \in U^P_M\} \]
\[ \xrightarrow{[G]} \text{succ}_0 \]

12. Atomic propositions
\[ x \in S_u, u \in U_M \]
\[ \xrightarrow{\text{Lab}(x_{\text{jump}})} = \text{Temp}_u(x) \]
MODULE main

VAR state : \{s_1, \ldots, s_n\} \quad \text{--- all states of OBA}

ASSIGN init(state) := init_0; \quad \text{--- initial state of OBA}

next(state) := case

state=x : \{y_1, \ldots, y_n\}; \quad \text{--- transitions } x \rightarrow y_1, \ldots, x \rightarrow y_n

state=y_i \& !(g) : x; \ldots \quad \text{--- guarded transition } x \xrightarrow{g} y_i

esac;

FAIRNESS ! guardloop \quad \text{--- avoids infinite loops when testing guards}

DEFINE guardloop := state in \{x_1, \ldots, x_m\} \quad \text{--- states in guards}

VAR \upsilon : \text{boolean}; \quad \text{--- variable } \upsilon \text{ from OBA}

ASSIGN init(\upsilon) := FALSE; \quad \text{--- valuation function } Val_a

next(\upsilon) := case

state = s^\upsilon : b^\upsilon; \ldots \quad \text{--- assigns value } b^\upsilon \text{ to } \upsilon \text{ in state } s^\upsilon

TRUE : \upsilon; \quad \text{--- preserves the current value of } \upsilon

esac;

\text{--- LTL/CTL formula } f \in F_A \text{ which uses variables } t_1, \ldots, t_j

\text{LTLSPEC } f(t_1, \ldots, t_j) \quad \text{--- CTLSPEC } f(t_1, \ldots, t_j)
- BDD construction
- Unfortunately the current approach does not scale well
Future work

- Reduction of the problem
  - Partial order reduction
  - Independent groups
    - Multiple OBAs (one OBA per annotation)
    - Automatic detection of annotations from text
- Sequencing of use-cases vs parallelism
Summary

- Method for verification of textual use-cases
  - against temporal properties (CTL/LTL)
  - under all use case orderings implied by precedence relation
  - custom temporal annotations (TADL)
- Supports collaborative work and iterative development
- Paper submitted to SEAA 2012
  - 38th Euromicro Conference on Software Engineering and Advanced Applications, September 5-8, 2012, Cesme, Izmir, Turkey
Thank you for attention

See also

http://code.google.com/a/eclipselabs.org/p/reprotool/