# FOAM
# A Lightweight Method for Verification of Use-Cases

(submitted to SEAA)

http://d3s.mff.cuni.cz


Department of Distributed and Dependable Systems

CHARLES UNIVERSITY IN PRAGUE
faculty of mathematics and physics

**Viliam Šimko**
**Tomáš Bureš**
**Petr Hnětynka**
**František Plášil**

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


Plzeň, May 16, 2012

# Goals

- Requirements specification in a changing environment
- Specification of **temporal dependencies** among use-cases.
  - using **temporal formulae** encapsulated as **annotations**
- Verification on the specification level
  - using **NuSMV** model checker

# Motivation: temporal dependency

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

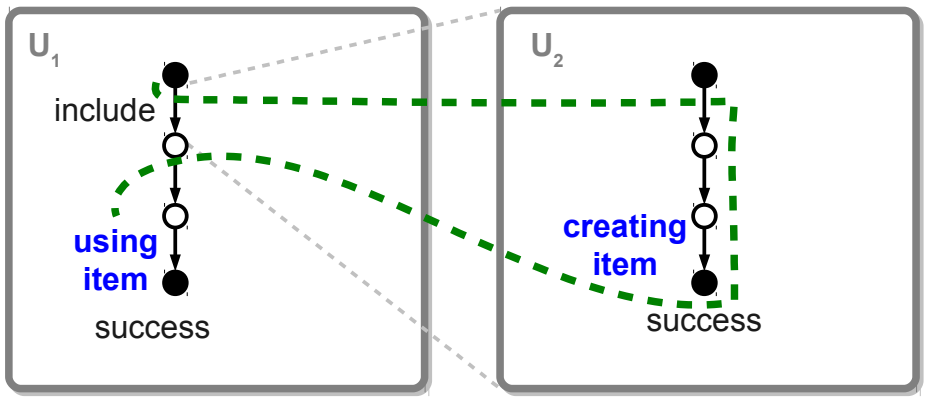
**Main success scenario:**

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:**

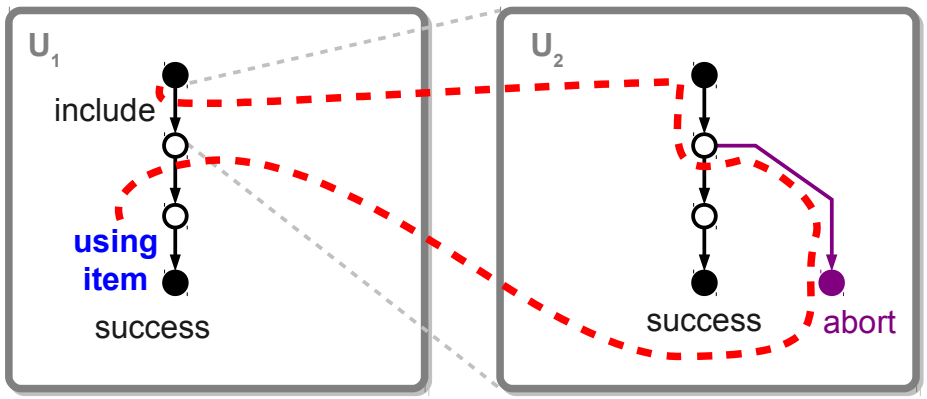
1. The buyer uses the web page to send a review item information request to the GPM.
2. The GPM displays information about the item.
3. The buyer reviews item information.

**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

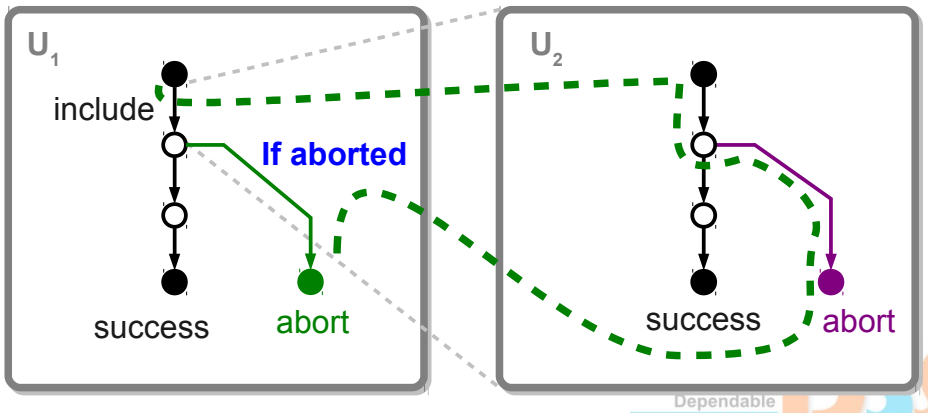
**Main success scenario:**

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.

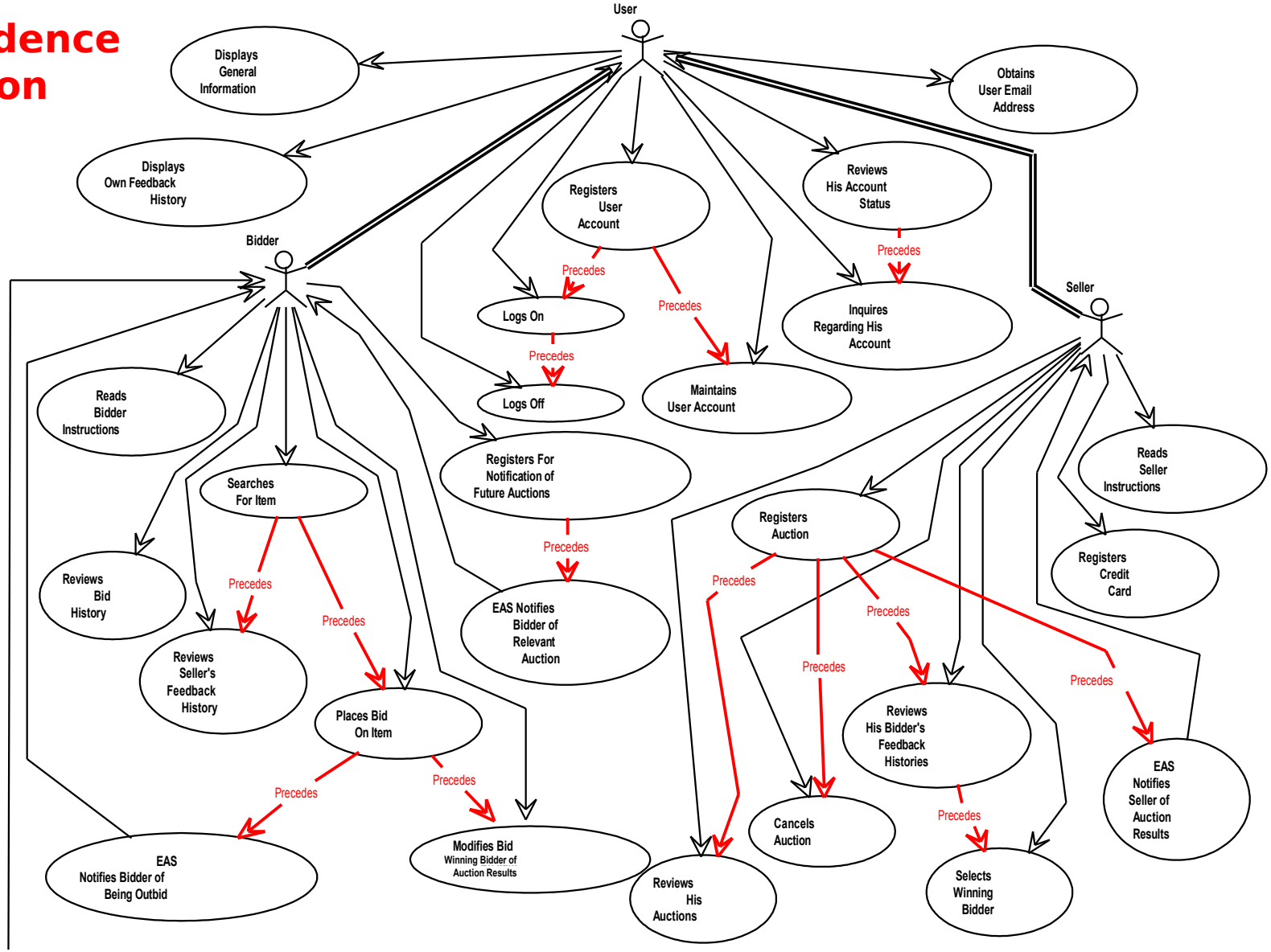
**Variation:**

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

**OK**



# Precedence Relation




**Case Study:** Donald Firesmith, Global personal marketplace system requirements specification, 2003, Link

# How to formalize this?

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

**Main success scenario:**

1. **Include** use-case **"Buyer Reviews Item Information"**. **#include:u2**
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. **#use:item**
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. **#guard:u2aborted**

1a1. The GPM displays a message "Bid cannot be placed"

1a2. Use-case aborted. **#abort**

**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. **#create: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** **#mark:u2aborted**

- **Flow annotations** (describing structure)
  - **abort, include, goto, mark, guard**
- **Temporal annotations** (to be checked)
  - **create, use, …** (defined in TADL)

# Temporal Annotation Definition Language

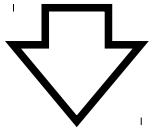
**Annotations:**

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

\+

**TADL Template:**

```
Annotations: create, use
 CTL AG( create –> EF(use) ) "Branch with use required after create"
 CTL AG( create –> AX(AG(!create)) ) "Only one create"
 CTL A[!use U create | !EF(use)] "First create then use"
```



**A set of temporal formulae to be checked**

$$\textbf{CTL AG}(\ \text{create}_{city}\ \text{–>}\ \textbf{EF}(\text{use}_{city})\ )$$
$$\textbf{CTL AG}(\ \text{create}_{city}\ \text{–>}\ \textbf{AX}(\textbf{AG}(\ !\,\text{create}_{city}))\ )$$
$$\textbf{CTL A}[\ !\,\text{use}_{city}\ \textbf{U}\ \text{create}_{city}\ |\ !\,\textbf{EF}(\text{use}_{city})]$$

$$\textbf{CTL AG}(\ \text{create}_{map}\ \text{–>}\ \textbf{EF}(\text{use}_{map})\ )$$
$$\textbf{CTL AG}(\ \text{create}_{map}\ \text{–>}\ \textbf{AX}(\textbf{AG}(\ !\,\text{create}_{map}))\ )$$
$$\textbf{CTL A}[\ !\,\text{use}_{map}\ \textbf{U}\ \text{create}_{map}\ |\ !\,\textbf{EF}(\text{use}_{map})]$$

# More TADL Examples

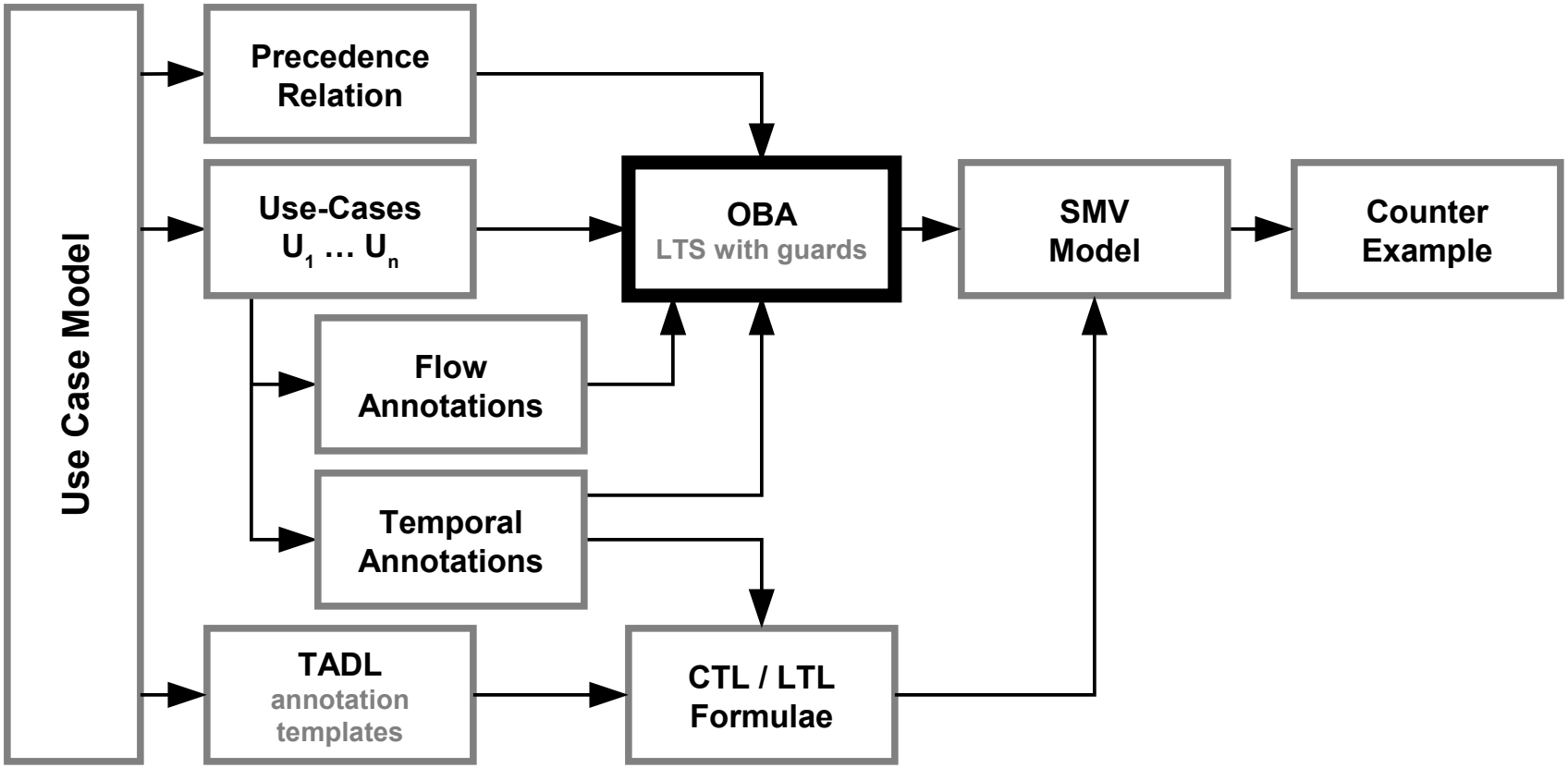
```
Annotations: create, use
 CTL AG( create -> EF(use) ) "Branch with use required after create"
 CTL AG( create -> AX(AG(!create)) ) "Only one create"
 CTL A[!use U create | !EF(use)] "First create then use"
```

```
Annotations: open, close -- strict ordering of 2 phases
 LTL G(open -> F(close)) "After open, close is required"
 CTL AG(open -> AX(A[!open U close])) "No multi-open"
 CTL AG(close -> AX(A[!close U open | !EF(close)])) "No multi-close"
 CTL A[!close U open | !EF(close)] "First open then close"
```

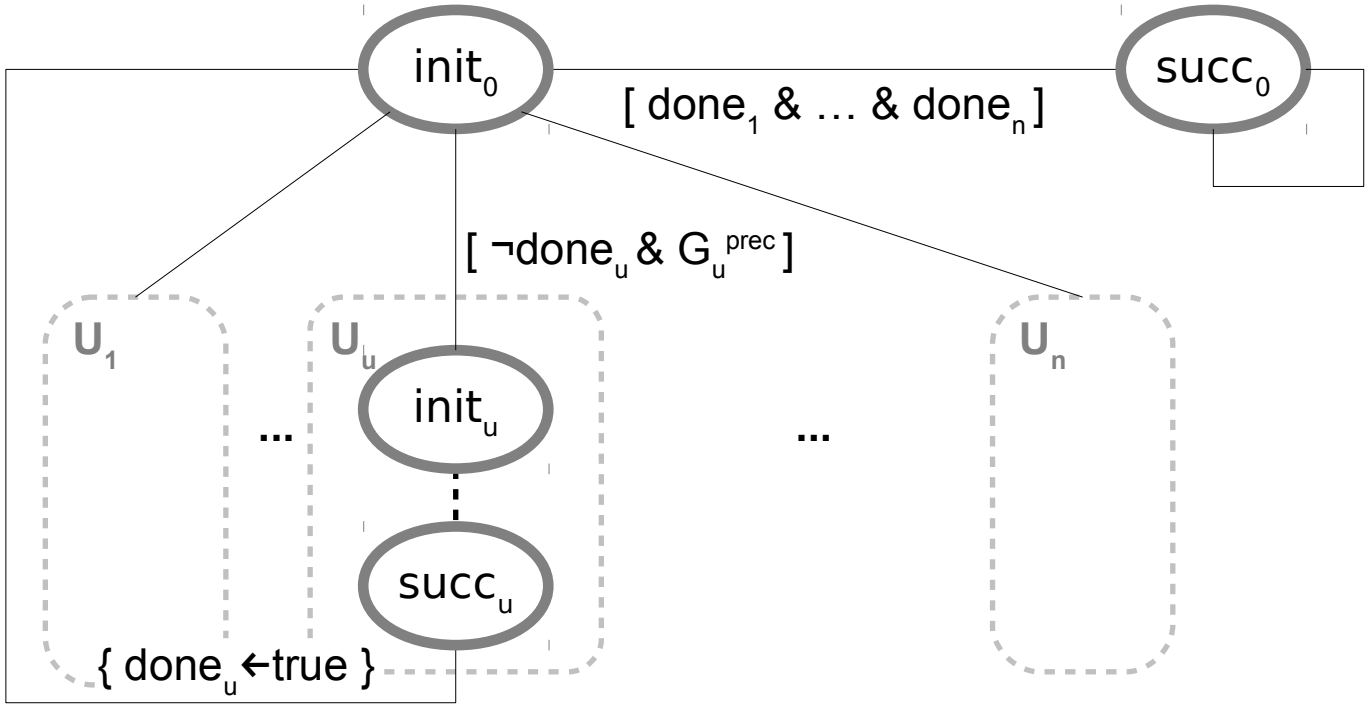
```
Annotations: init, process, release -- strict ordering of 3 phases
 -- init → process
 CTL A[!process U init | !EF(process)] "First init then process"
 CTL AG(init -> AF(process)) "After init there should always be process"
 CTL AG(init -> AX(A[!init U process])) "No multi-init without process"
 CTL AG(process -> AX(A[!process U init | !EF(process)]))
     "No multi-process without init"
 -- process → release
 CTL A[!release U process | !EF(release)] "First process then release"
 CTL AG(process -> AF(release)) "After process, release is required"
 CTL AG(process -> AX(A[!process U release]))
     "No multi-process without release"
 CTL AG(release -> AX(A[!release U process | !EF(release)]))
     "No multi-release without process"
```

# Transformation: Overview

*TADL : **T**emporal **A**nnotation **D**efinition **L**anguage (templates)*
*OBA : **O**verall **B**ehavior **A**utomaton*



# Overall Behavior Automaton (OBA)



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

**1. Representing steps**

$$\frac{u \in U_M, x \in S_u}{x^{\mathsf{in}} \to x^{\mathsf{var}} \to x^{\mathsf{jump}} \to x^{\mathsf{ext}} \to x^{\mathsf{out}}}$$

**2. Representing scenarios**

$$\frac{u \in U_M, w \in W_u, x_1 \leq_w \ldots \leq_w x_n}{(x_1^{\mathsf{out}} \to x_2^{\mathsf{in}}), \ldots, (x_{n-1}^{\mathsf{out}} \to x_n^{\mathsf{in}})}$$

**3. Connecting variations**

$$\frac{u \in U_M, w \in W_u, w = \{y_1, \ldots, y_n\}, Var_u(w) = x, \quad G_V = \{g \mid \langle \text{guard:}g \rangle \in Flow_u(y_1)\}}{x^{\mathsf{var}} \xrightarrow{[G_V]} y_1^{\mathsf{in}}}$$

**4. Connecting extensions**

$$\frac{u \in U_M, w \in W_u, w = \{y_1, \ldots, y_n\}, Ext_u(w) = x, \quad G_E = \{g \mid \langle \text{guard:}g \rangle \in Flow_u(y_1)\}}{x^{\mathsf{ext}} \xrightarrow{[G_E]} y_1^{\mathsf{in}}}$$

**5. Continuation from scenarios**

$$\frac{u \in U_M, w \in W_u, x = Var_u(w) \vee x = Ext_u(w), \quad w = \{y_1, \ldots, y_n\}, \langle \text{abort} \rangle \notin Flow_u(y_n), \quad \forall_{s \in S_u} \langle \text{goto:}s \rangle \notin Flow_u(y_n)}{y_n^{\mathsf{out}} \to x^{\mathsf{out}},}$$

**6. Handling GOTO annotations**

$$\frac{u \in U_M, x \in S_u, \langle \text{goto:}y \rangle \in Flow_u(x)}{x^{\mathsf{out}} \to y^{\mathsf{jump}},}$$

**7. Handling ABORT annotations**

$$\frac{u \in U_M, x \in S_u, \langle \text{abort} \rangle \in Flow_u(x)}{x^{\mathsf{out}} \to x^{\mathsf{out}}}$$

Department of Distributed and Dependable Systems

# Construction of OBA using Inference Rules (2/2)

**8. Handling INCLUDE (procedure call)**

$$\frac{u, c \in U_M, x \in S_u, \langle \text{include:}c \rangle \in Flow_u(x), \quad w_c^m = \{y_1, \ldots, y_n\}}{x^{\text{jump}} \xrightarrow{\{incl_{u,c} \leftarrow true\}} y_1^{\text{in}}, x^{\text{jump}} \xrightarrow{[false]} x^{\text{ext}},}$$

**9. Handling INCLUDE (return)**

$$\frac{u, c \in U_M, x \in S_u, \langle \text{include:}c \rangle \in Flow_u(x), \quad w_c^m = \{y_1, \ldots, y_n\}, \forall_{s \in S_c} \langle \text{goto:}s \rangle \notin Flow_c(y_n)}{y_n^{\text{out}} \xrightarrow{\{incl_{u,c} \leftarrow false\}} x^{\text{ext}},}$$

**10. Scheduler**

$$\frac{u \in U_M^P, w_u^m = \{x_1, \ldots, x_n\}, \quad G_u^{\text{prec}} = \{done_v | \exists_{v \in U_M^P} (v, u) \in Prec_M\}}{init_0 \xrightarrow{[G_u^{\text{prec}}, \neg done_u]} x_1^{\text{in}}, x_n^{\text{out}} \xrightarrow{\{done_u \leftarrow true\}} init_0}$$

**11. Final state**

$$\frac{G = \{done_u | u \in U_M^P\}}{init_0 \xrightarrow{[G]} succ_0}$$

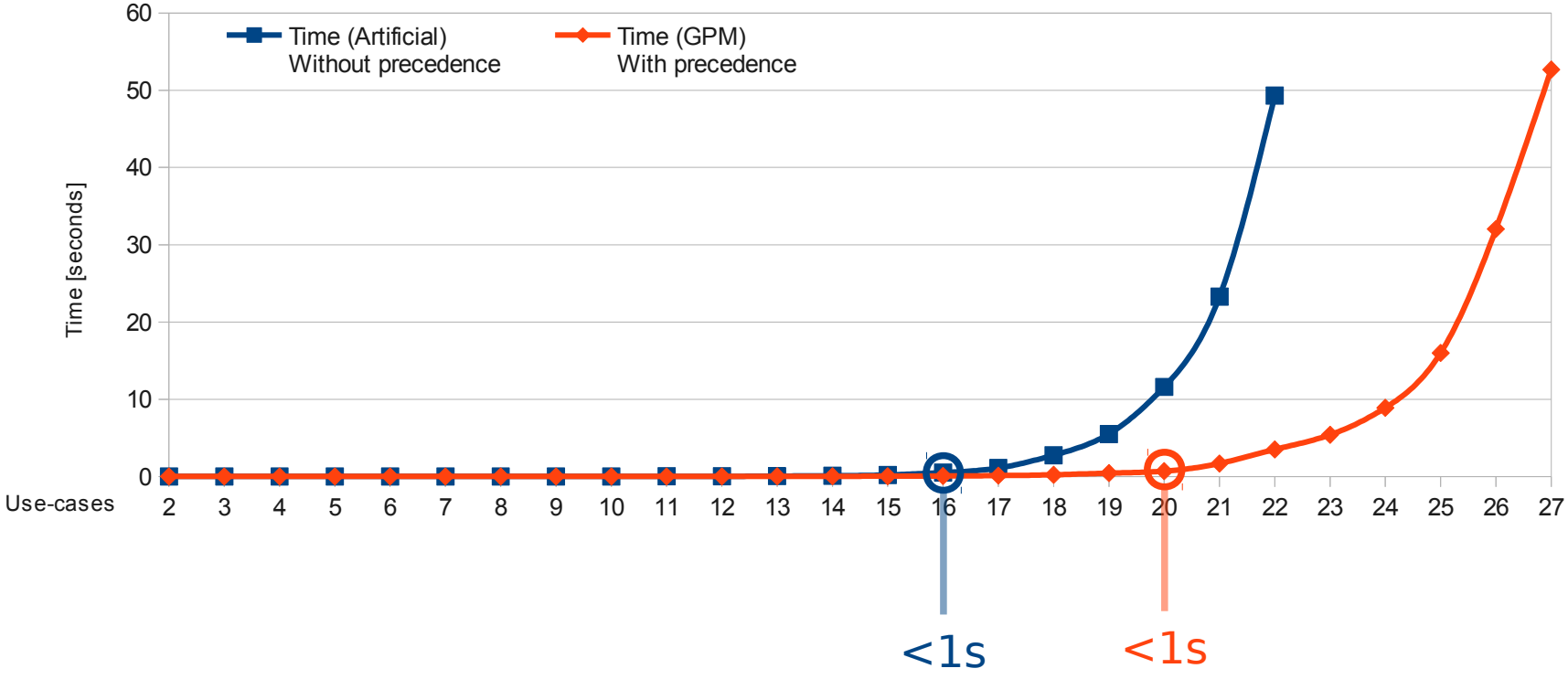
**12. Atomic propositions**

$$\frac{x \in S_u, u \in U_M}{Lab(x^{\text{jump}}) = Temp_u(x)}$$

# OBA represented in NuSMV

**MODULE** main
  **VAR** state : $\{s_1, ..., s_n\}$ -- *all states of OBA*
  **ASSIGN init**(state) := init_0; -- *initial state of OBA*
    **next**(state) := **case**
      state=$x$ : $\{y_1, ..., y_n\}$; -- *transitions* $x \rightarrow y_1, ..., x \rightarrow y_n$
      state=$y_i$ & !($g$) : $x$; ... -- *guarded transition* $x \xrightarrow{g} y_i$
    **esac**;
  **FAIRNESS** ! guardloop -- *avoids infinite loops when testing guards*
  **DEFINE** guardloop := state **in** $\{x_1, ..., x_m\}$ -- *states in guards*
  **VAR** $v$ : **boolean**; -- *variable* $v$ *from OBA*
  **ASSIGN init**($v$) := **FALSE**; -- *valuation function* $Val_A$
    **next**($v$) := **case**
      state = $s^v$ : $b^v$; ... -- *assigns value* $b^v$ *to* $v$ *in state* $s^v$
      **TRUE** : $v$; -- *preserves the current value of* $v$
    **esac**;
  -- *LTL/CTL formula* $f \in F_A$ *which uses variables* $t_1, \ldots, t_j$
  **LTLSPEC** $f(t_1, \ldots, t_j)$ ... **CTLSPEC** $f(t_1, \ldots, t_j)$

# Evaluation




- Unfortunately the current approach does not scale well

# Future work

- Reduction of the problem
  - Partial order reduction
  - Independent groups of use-cases
  - Construction of a specific OBA per annotation group
- 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/