



# Assessing the Applicability of Use Case Maps for Business Process and Workflow Description

Gunter Mussbacher, Daniel Amyot  
SITE, University of Ottawa  
{gunterm | damyot}@site.uottawa.ca

January 24, 2008

# Motivation – Everything Evolves...

- As any other language, Use Case Maps (UCMs) have to be reevaluated from time to time in light of new technological advances
- UCMs share many similarities with workflow description languages (WDL)
  - Structure and intent of both are similar
  - UCMs have been used for business process modeling
- Workflow Patterns have been collected for WDL but are applicable to scenario notations in general
  - Many languages and standards have been assessed based on these patterns
- Assess UCMs based on Workflow Patterns
  - Gives indication on the capabilities of UCMs as a general scenario notation
  - Shows opportunities for improvements of UCMs
  - Allows comparison of UCMs to other languages



**How ?**

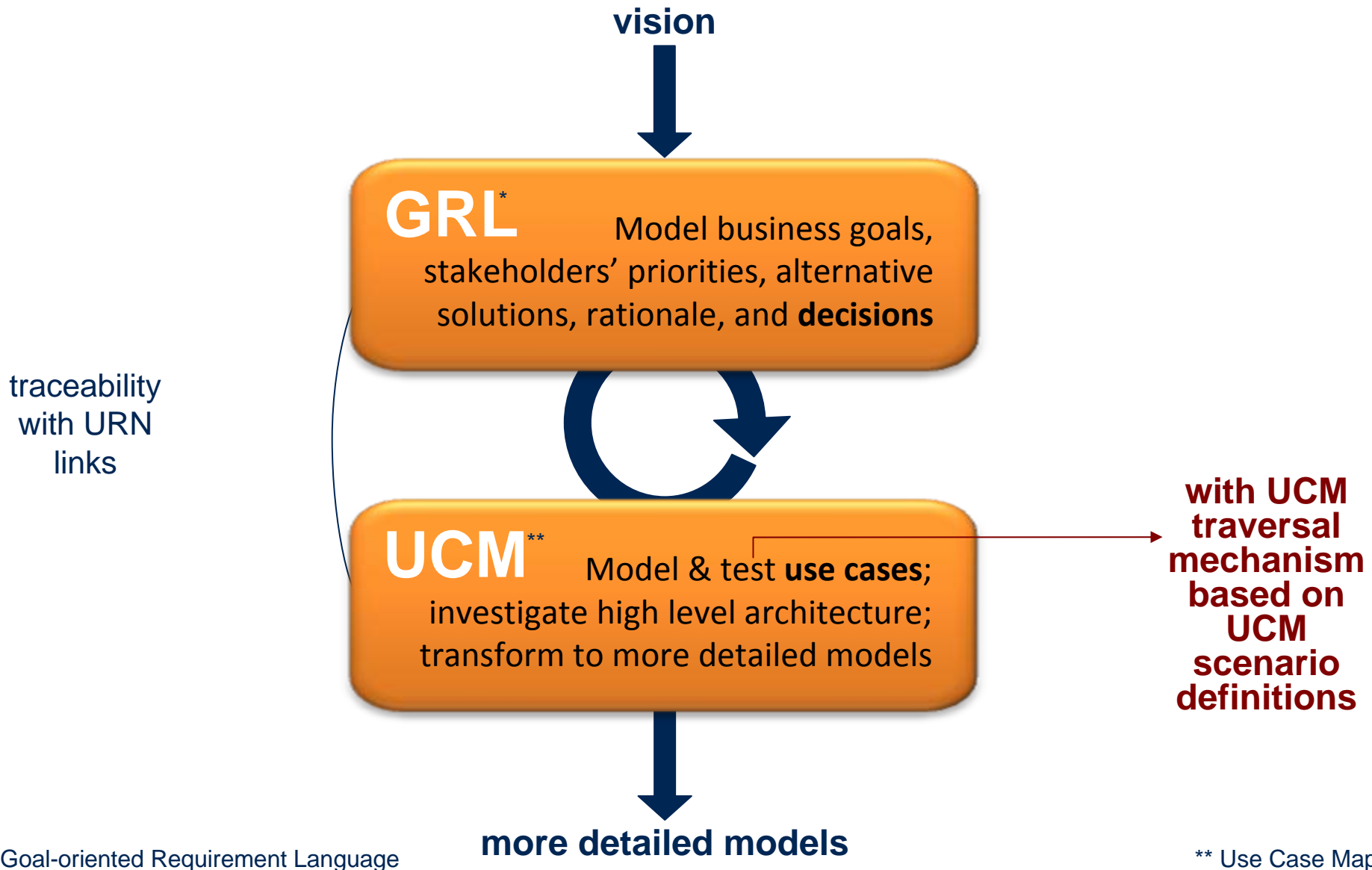


**Therefore**

# Table of Contents

- Background on Use Case Maps
- Overview of Workflow Patterns
- Assessment of UCMs based on Workflow Patterns
- Comparison of UCMs with BPMN, UML Activity Diagrams, and BPEL4WS
- Future Work

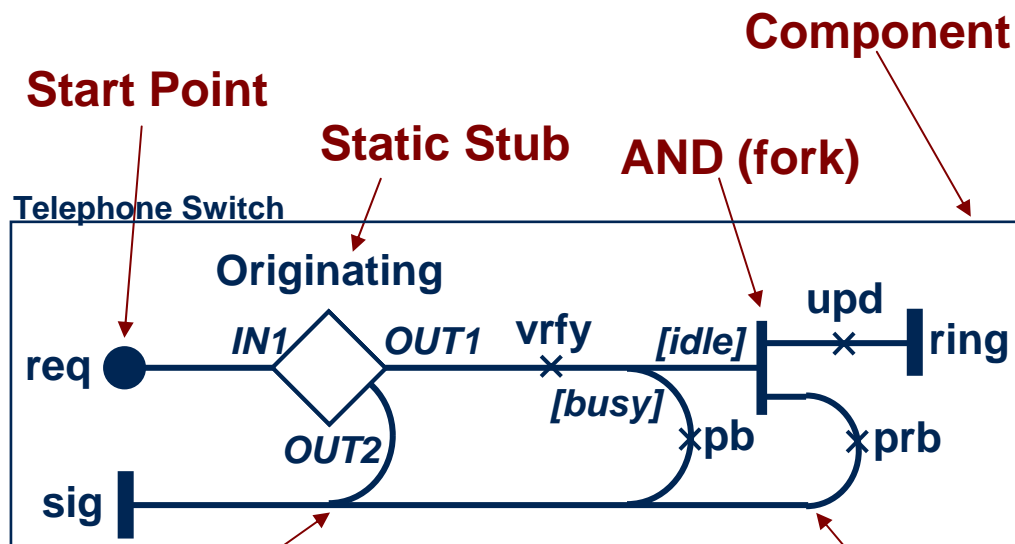
# User Requirements Notation (URN)



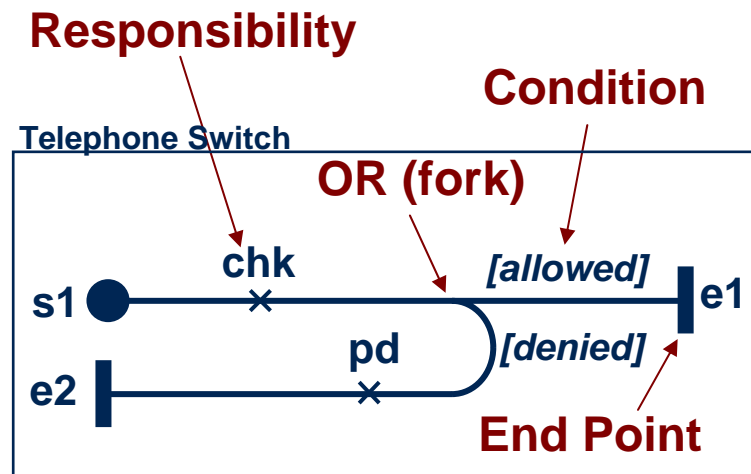
\* Goal-oriented Requirement Language

\*\* Use Case Maps

# Use Case Maps: Notation



OR (join) a) Basic Call map Path



b) OCS plug-in map

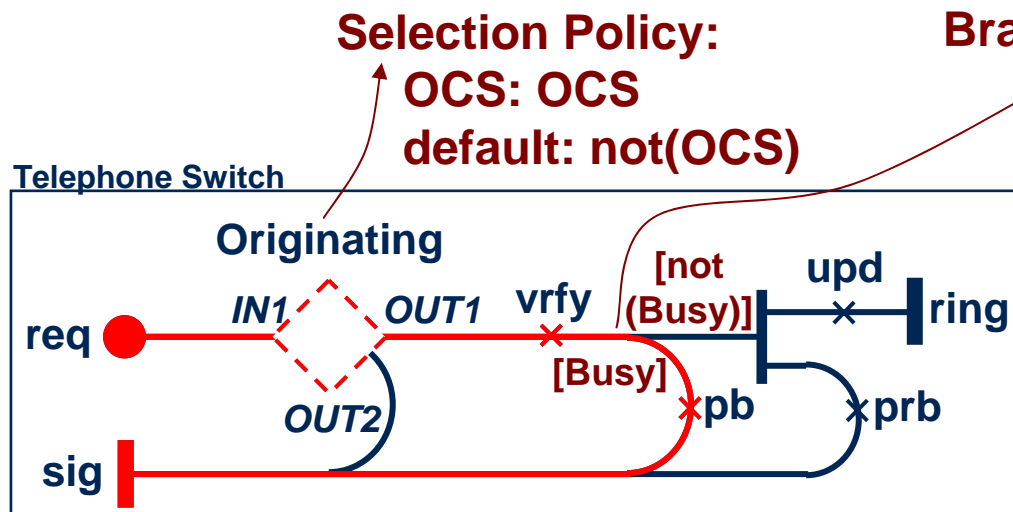
## UCM Example: Tiny Telephone System



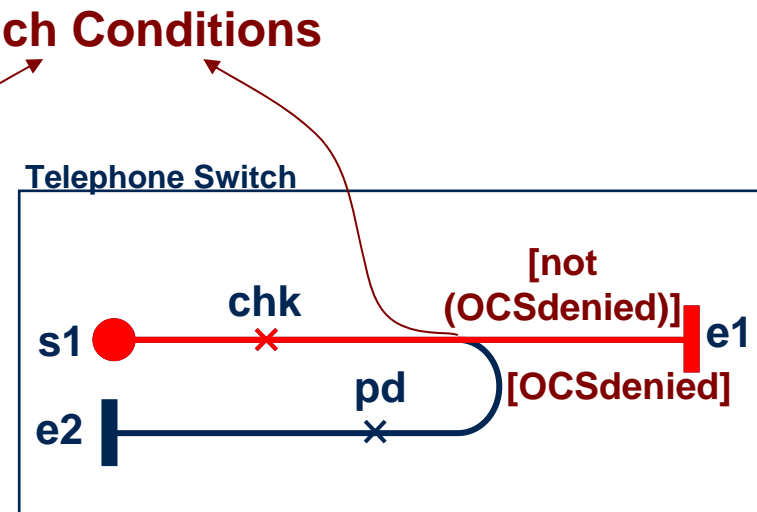


# Use Case Maps: Traversal Mechanism

## UCM Example: Tiny Telephone System



a) Basic Call map



b) OCS plug-in map

- Scenario Definition “Busy Call + OCS”

- Start point: req
- OCS = true
- OCSdenied = false
- Busy = true
- End point: sig



c) default plug-in map

# Use Case Maps: Traversal Mechanism

- jUCMNav's Traversal Mechanism executes a UCM model given UCM scenario descriptions (i.e. highlights the scenarios)
- Two options
  - Deterministic (only one alternative at any choice point can be enabled)
  - Non-deterministic (randomly choose an alternative from all enabled ones)
- Boolean, Integer, and Enumeration variables are evaluated and can be changed by responsibilities during the traversal of the UCM model
  - These variables are used in expressions for any alternative of a choice point
- Straightforward, **intuitive** interpretation of Use Case Maps
  - Exclusive OR for OR-fork
  - No synchronization on OR-joins
  - AND-fork explores all outgoing paths in parallel
  - AND-join requires all incoming paths to arrive
  - Stubs contain an exclusive OR for the selection of a single plug-in map

# Workflow Patterns

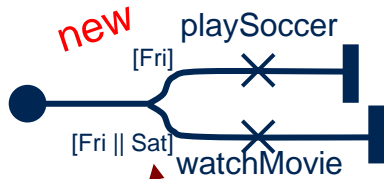
- 43 Workflow Patterns in eight groups
  - 5 Basic Control Flow Patterns
  - 14 Advanced Branching and Synchronization Patterns
  - 7 Patterns involving Multiple Instances
  - 5 State-Based Patterns
  - 5 Cancellation and Force Completion Patterns
  - 3 Iteration Patterns
  - 2 Termination Patterns
  - 2 Trigger Patterns
- <http://www.workflowpatterns.com>



# Advanced Branching and Synchronization Patterns

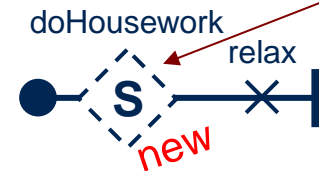
**only a single active plug-in, no synchronization, threshold and blocking not considered \***

WCP-06) Multi-Choice

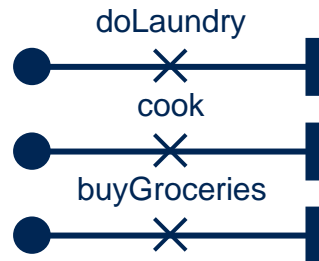


**only one path  
or random \***

WCP-07) Structured  
Synchronizing Merge



Plug-ins:



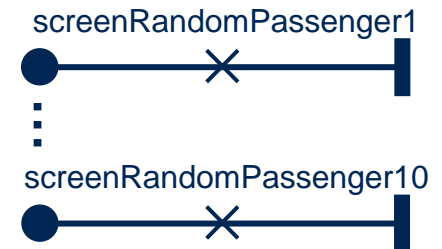
Selection Policy:

first plug-in: [Wed || Sat]  
second plug-in: [any day]  
third plug-in: [Sat || Sun]

WCP-31) Blocking Partial Join



Plug-ins:



Selection Policy:

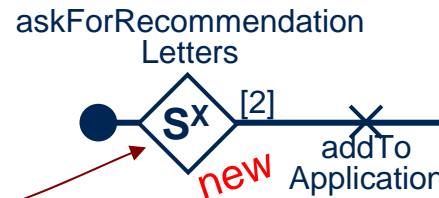
true for all plug-ins  
Threshold: 9  
Blocking: true

**\* current traversal mechanism**

# Patterns Involving Multiple Instances (MI)

**only a single active plug-in, no synchronization, replication factor and threshold not considered \***

WCP-34) Static Partial Join for MI



Plug-in:



Selection Policy: true  
Replication factor: 5  
Threshold: 2

WCP-12) MI without synchronization

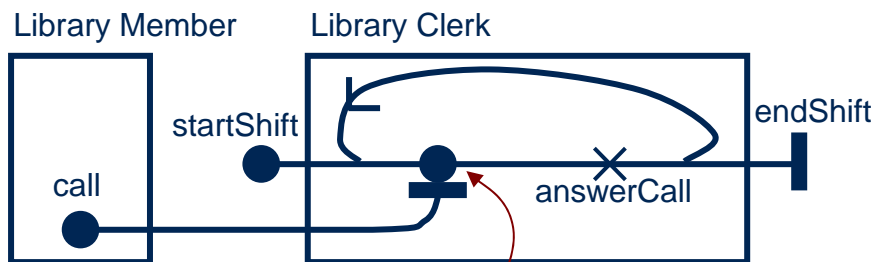


**replication factor is not taken into account \***

**\* current traversal mechanism**

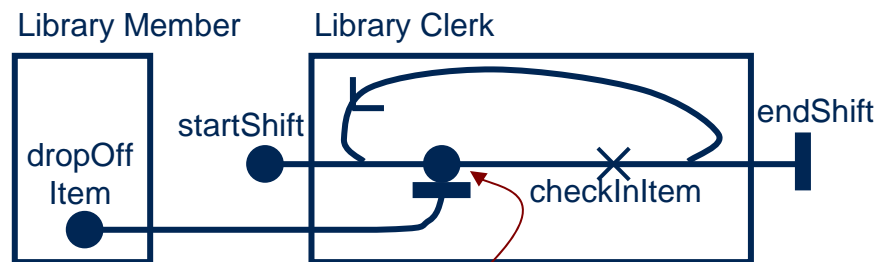
# Trigger Patterns

WCP-23) Transient Trigger



*Waiting Place Property:  
waitType: transient*

WCP-24) Persistent Trigger



*Waiting Place Property:  
waitType: persistent*

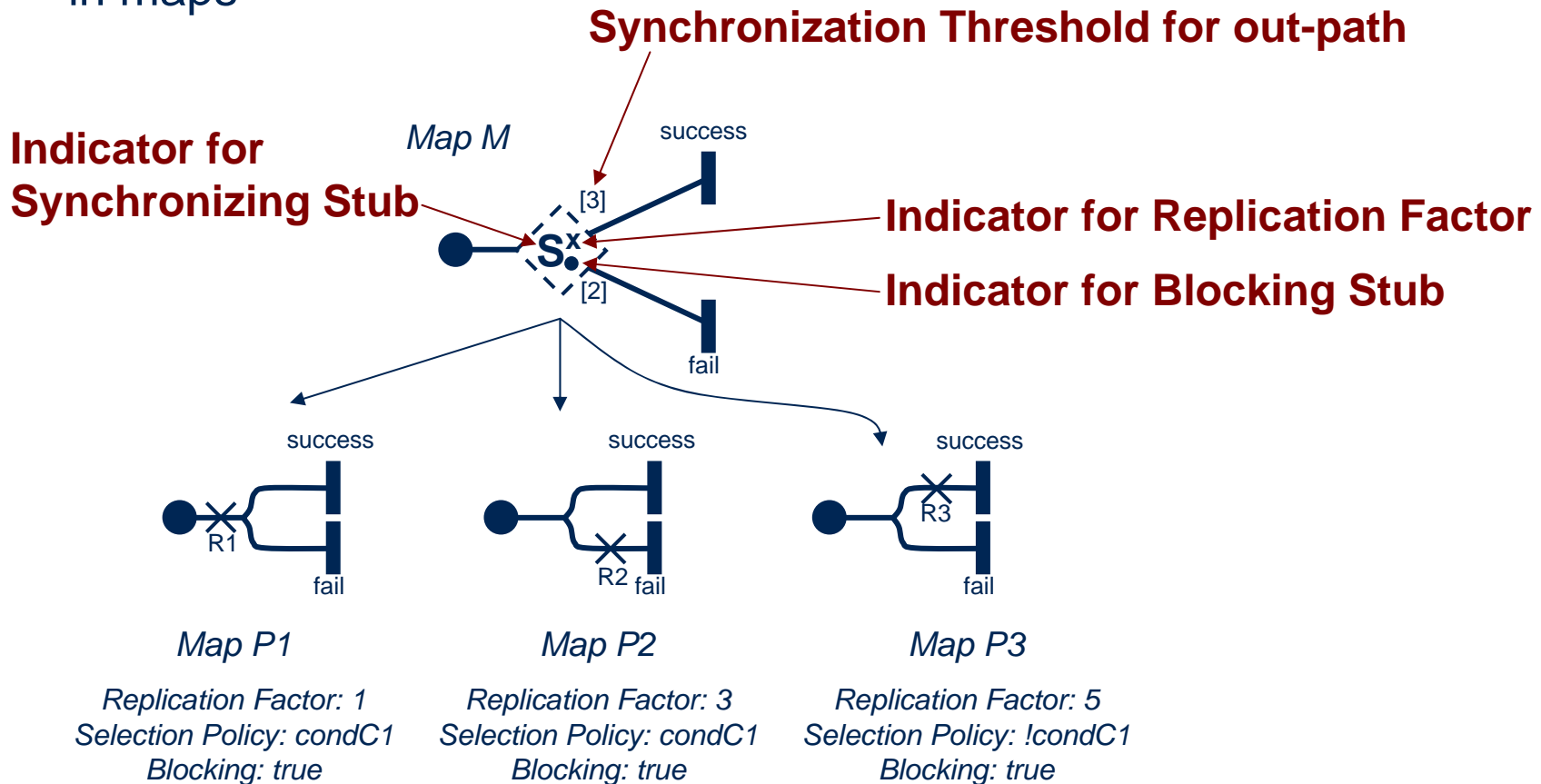
**property waitType not considered \***

\* Timers can also be used instead of waiting places

**\* current traversal mechanism**

# Summary of Synchronizing Stub

- Synchronizing Stub allows parallel execution and synchronization of plug-in maps



# Comparison with BPMN, Activity D., BPEL4WS

- BPMN 1.0: 24 (9)\* out of 43 workflow patterns
- UML 2.0 Activity Diagrams: 25 (5)\* out of 43 workflow patterns
- BPEL4WS 1.1: 17 (4)\* out of 43 workflow patterns
- Use Case Maps: 27 (2)\* out of 43 workflow patterns
  - 8 out of the 14 not supported patterns deal with cancellation scenarios
  - Assuming new notational element
    - Synchronizing stub
  - Assuming an improved traversal mechanism
    - E.g., take multiple-choice OR-forks, replication factor, and waitType into account
  - Only 11 out of 43 workflow patterns with current notation and traversal mechanism

\* full support (partial support)



# Future Work

- Implement enhancements in jUCMNav (the most popular URN modeling tool)
- Cancellation Patterns
  - All compared notations support them
  - Some very limited support available in UCM notation but no support available in jUCMNav
- Address longstanding issues with the UCM notation
  - Map Instances
  - Relationship of components on parent and plug-in maps
  - Component instances and types
- Aspect-oriented URN for communication patterns
  - Use aspect-oriented techniques to model communication patterns and identify where communication patterns need to be added