HENGSOFT

Feature Interactions in Aspect-Oriented Scenario Models

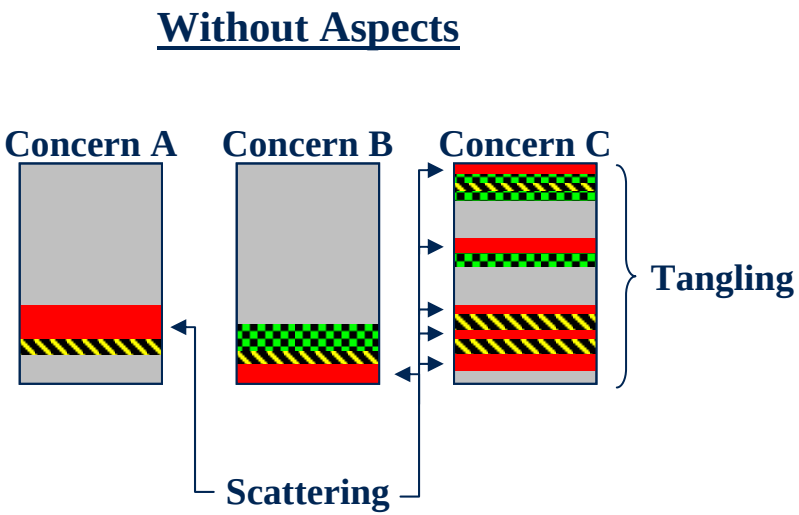
Gunter Mussbacher[1], Daniel Amyot[1],
Thomas Weigert[2], and Thomas Cottenier[3]

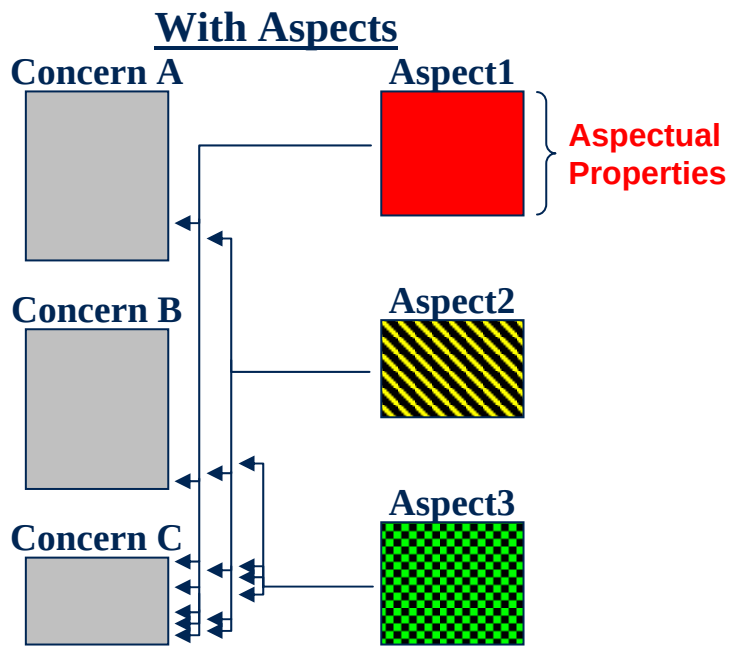
[1]SITE, University of Ottawa, Canada
[2]Missouri University of Science and Technology, Rolla, MO, USA
[3]Hengsoft LLC, Palatine, IL, USA

June 11, 2009

Background: Aspect-oriented Modeling

- Aspects address the problem of one concern crosscutting other concerns in a system or model
- Aspects can encapsulate concerns even if they are crosscutting





(each aspect contains a composition rule illustrated by the arrows that defines where to add the aspect)

… 3 Crosscutting Concerns (Aspect1, Aspect2, Aspect3)

Motivation

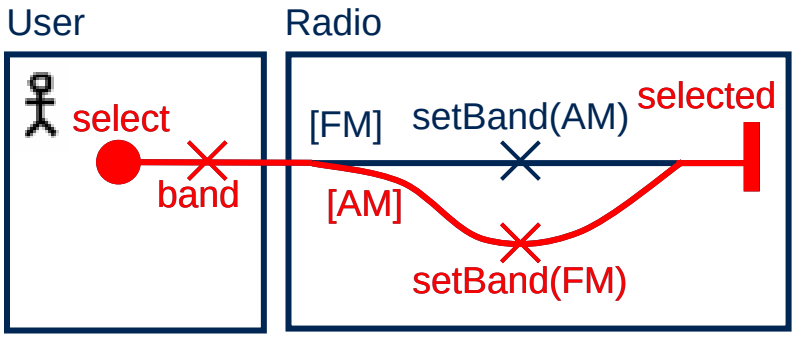
- Aspect-oriented Feature Development
  - Separation of concerns and minimization of crosscutting
  - One feature = one concern
  - Define features incrementally and independently
  - Resolve feature interactions (FI) incrementally and in a modular way
- Model features and resolve FI in a specification model
  - In this case: Use Case Maps (UCM) model
- A validation model ensures that the specification model remains compliant with feature specification and FI resolutions
  - UCM validation model is a low effort pre/postcondition approach
- Concerns should be kept separate in both models

Table of Contents

- Overview of Use Case Maps (UCM)
- Overview of Aspect-oriented Use Case Maps (AoUCM)
- Overview of Approach
- Example: Radio Control Software
  - Features
  - UCM Scenario Definitions
  - Problems
  - Feature Interactions
- Conclusion and Future Work

Use Case Maps: Overview

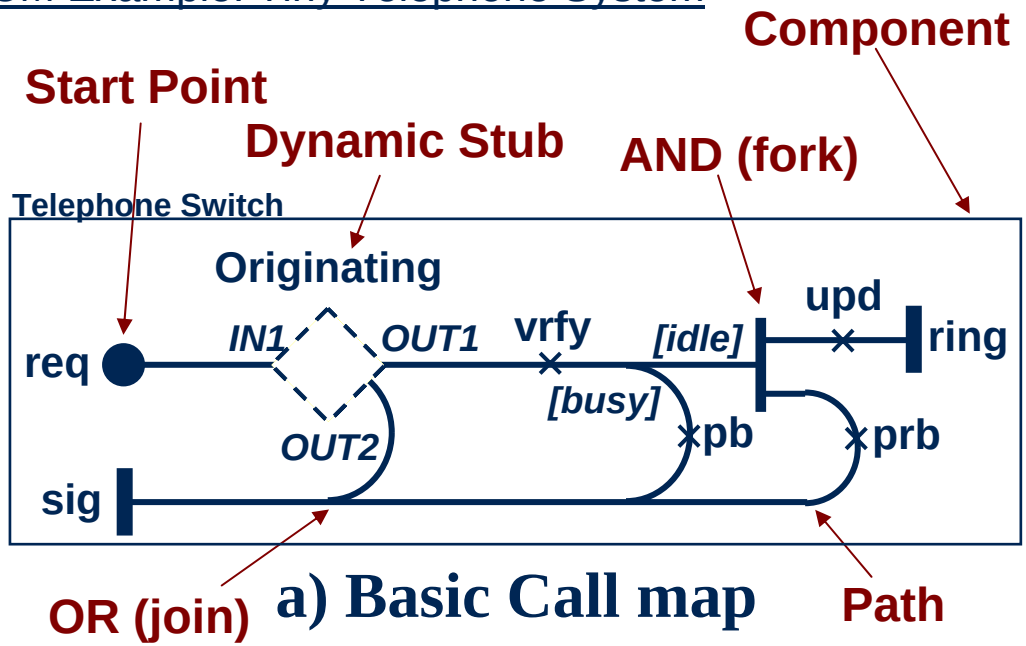
- User Requirements Notation (URN)
  - First and currently only standard which explicitly addresses goals-based and scenario-based models in one unified language
- Use Case Maps (UCMs) – scenario models



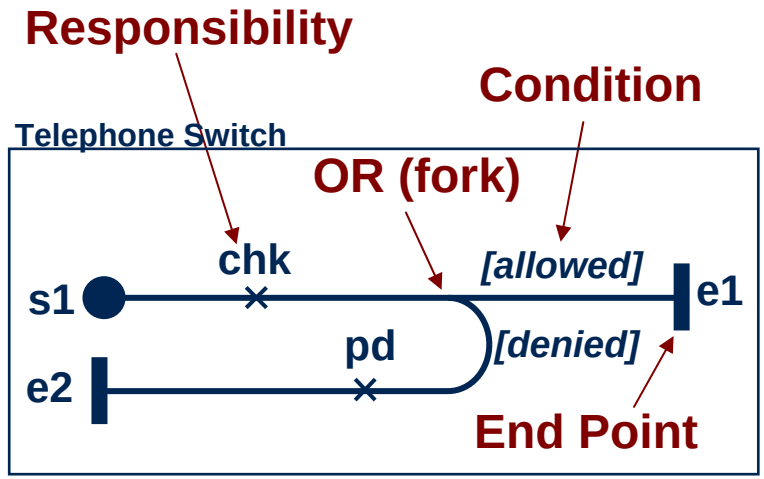
- UCM scenarios describe one path through the UCM model (only one alternative at any choice point is taken)
- jUCMNav's traversal mechanism executes the UCM model given a UCM scenario description (i.e. highlights the scenario)

Use Case Maps: Notation

UCM Example: Tiny Telephone System



a) Basic Call map



b) *OCS* plug-in map

c) *default* plug-in map

AND (join)

waiting place

Static Stub (at most one plug-in map)

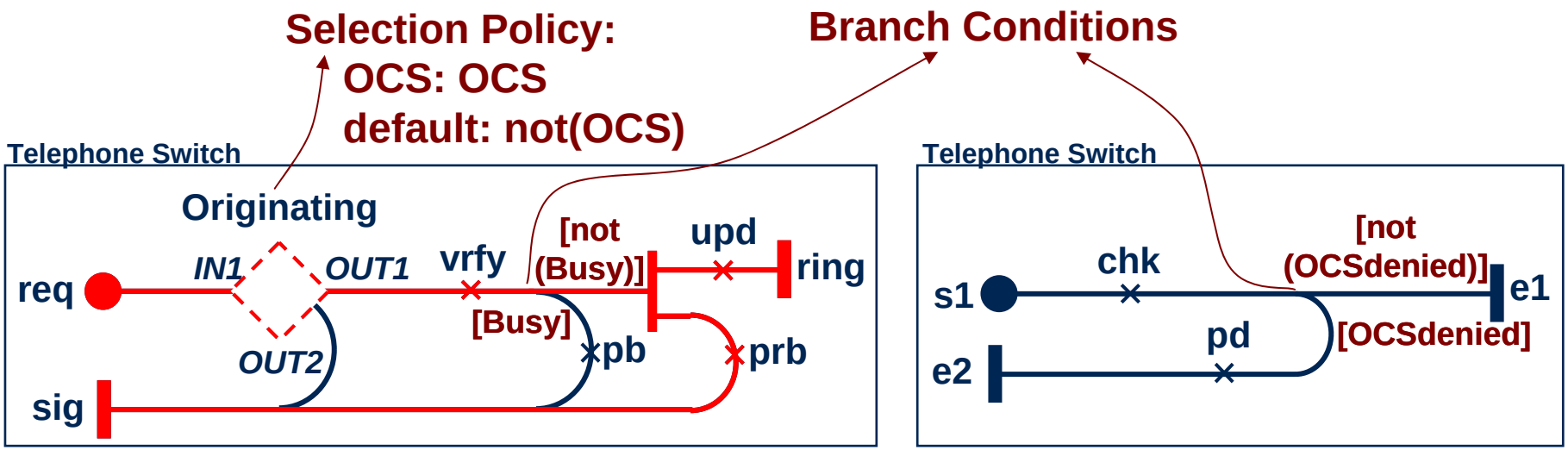
timer

Synchronizing Stub with synchronization threshold

Blocking Stub with replication indicator

Use Case Maps: Path Traversal Mechanism

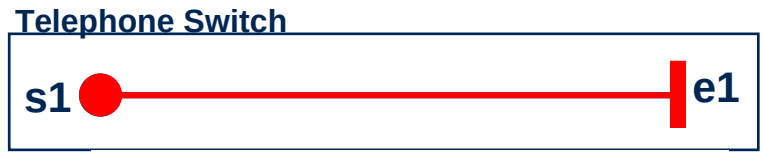
UCM Example: Tiny Telephone System



a) Basic Call map

b) *OCS* plug-in map

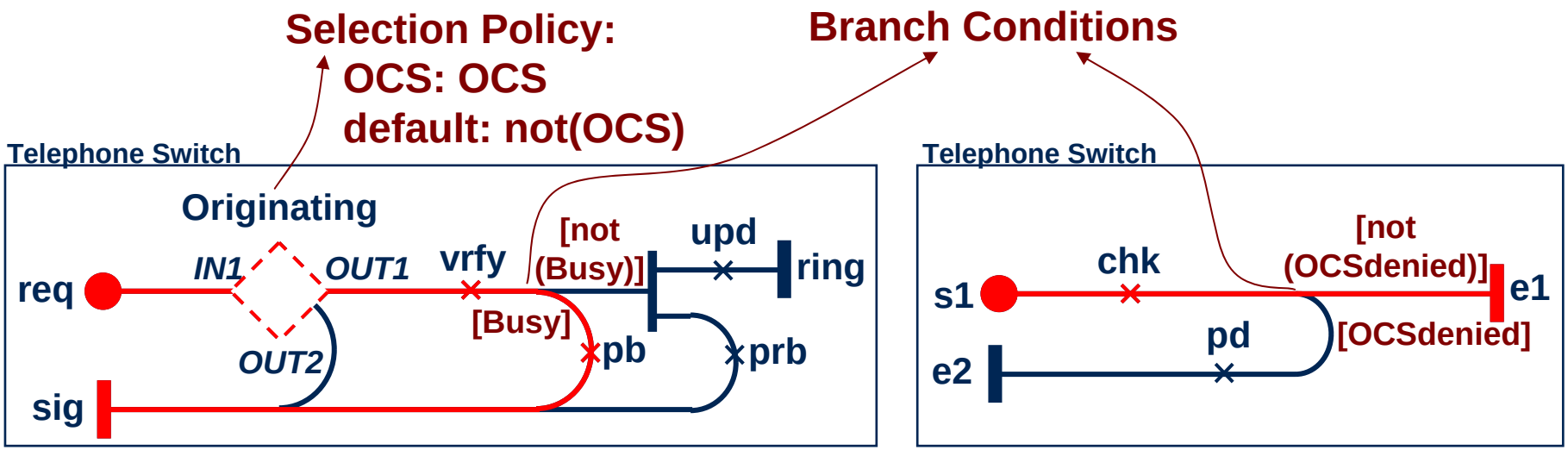
- Scenario Definition “Simple Basic Call”
  - Start point: req
  - OCS = false
  - Busy = false
  - End points: ring, sig



c) *default* plug-in map

Use Case Maps: Path Traversal Mechanism (2)

UCM Example: Tiny Telephone System



a) Basic Call map

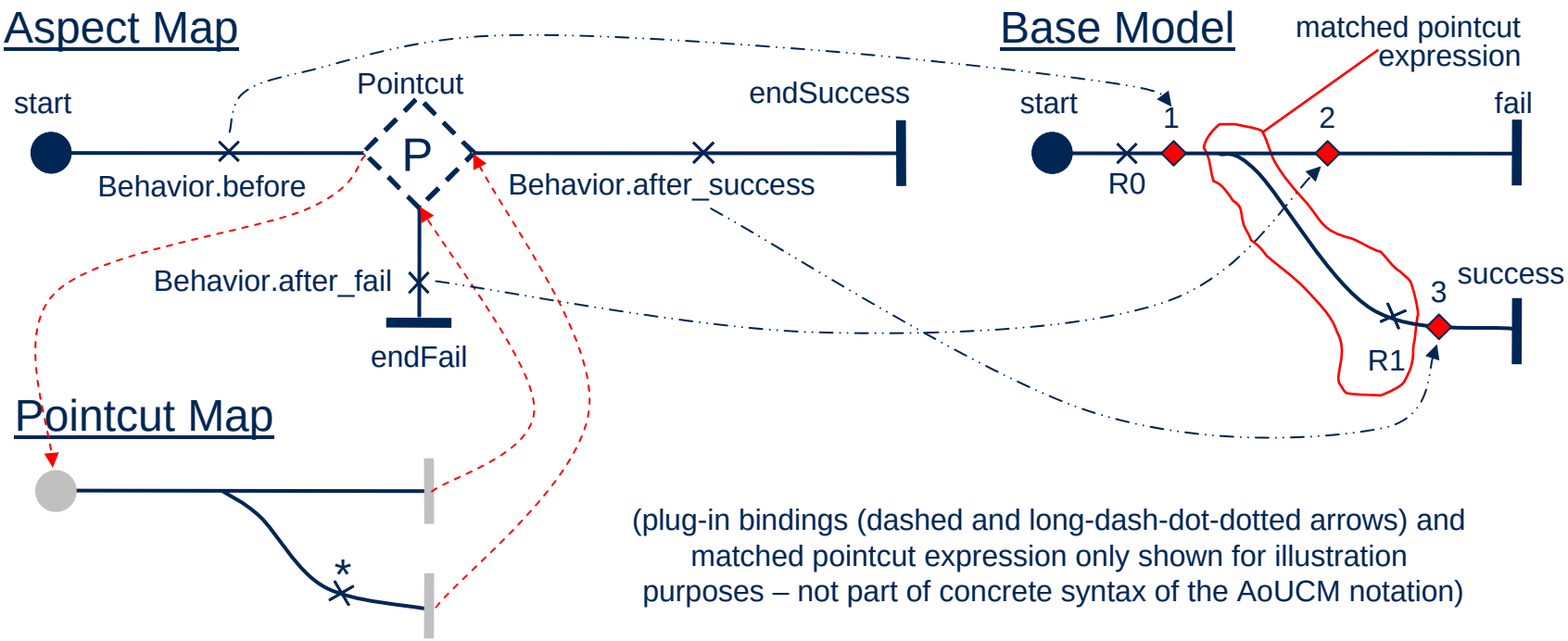
b) *OCS* plug-in map

- Scenario Definition “Busy Call + OCS”
  - Start point: req
  - OCS = true
  - OCSdenied = false
  - Busy = true
  - End point: sig

c) *default* plug-in map

Aspect-oriented Use Case Maps

- An aspect defines its structure/behavior and a pattern called pointcut expression for its composition rule stating where the aspect is to be applied in a model



(plug-in bindings (dashed and long-dash-dot-dotted arrows) and matched pointcut expression only shown for illustration purposes – not part of concrete syntax of the AoUCM notation)

Pointcut stub:

Aspect marker:

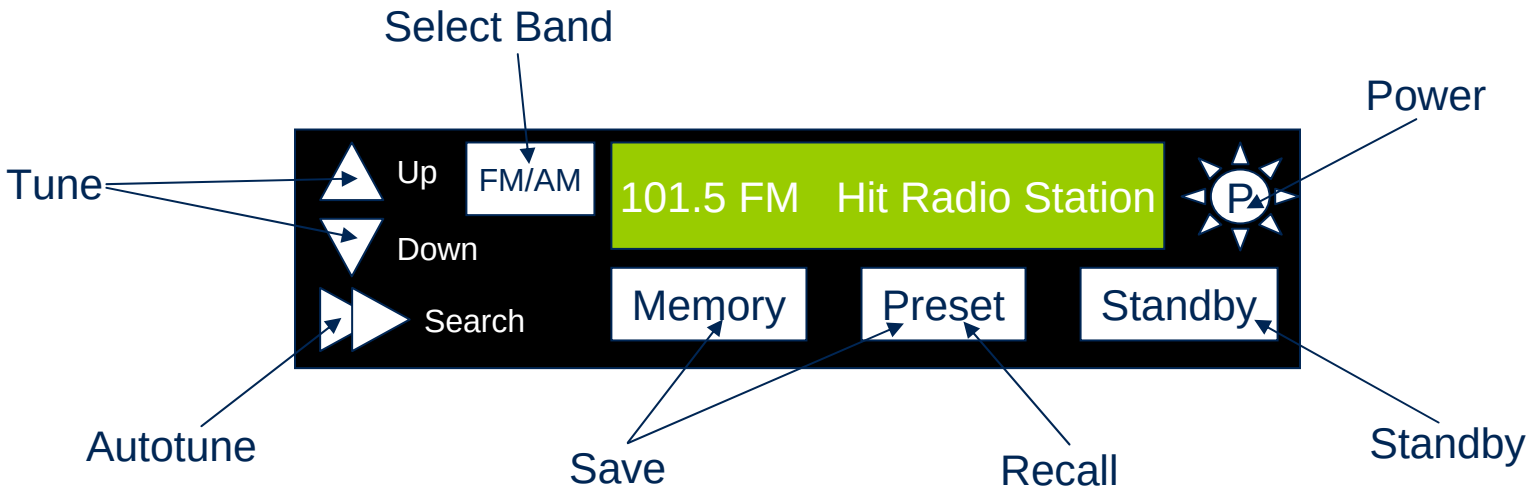
AoUCM … Aspect-oriented Use Case Maps

Overview of Approach

- Step 1: Model each feature as a concern (aspect)
- Step 2: Create scenario definitions (= validation model)
  - Complete branch coverage for each feature
  - Global variables, formalization of conditions for choice points, pseudo-code for responsibilities
- Step 3: Model FI resolutions with aspects
  - Model as part of existing feature concerns
- Step 4: Create scenario definitions for FI resolutions
  - Reuse scenario definitions of individual features as much as possible
  - Requires composition rules for aspect-oriented scenario definitions

Features of Radio Device

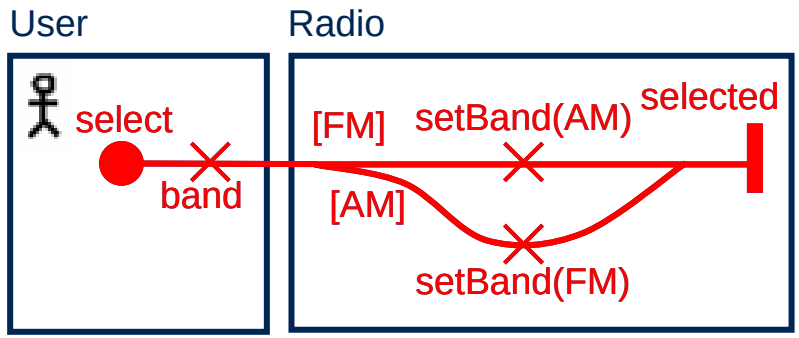
- 10 Features
  - Select Band, Tune, Autotune, Save, Recall, Traffic News, Power On / Off, Standby On / Off



  - Update Display
  - Remember Settings
    - Radio device remembers last band and frequency settings when turned off, defaults to these settings when turned on

Example of Features: Select Band

- Step 1: Model feature



- Step 2: Create scenario definitions
  - Global enumeration variable: Band = { AM , FM }
  - Formalize conditions: [FM] … Band == FM, [AM] … Band == AM

Select Band	Scenario One	Scenario Two
Included Scenarios	n/a	n/a
Start Points	select	select
Initialization	Band = FM	Band = AM
End Points	selected	Selected
Postcondition	Band == AM	Band == FM

  - Code for responsibilities changes the value of variable Band
    - setBand(AM) … Band = AM; setBand(FM) … Band = FM

UCM Scenario Definitions

- Scenario definitions may be included in other scenario definitions
  - Union of start points, end points, and pre/postconditions
  - Start points ordered by scenario include order
  - Initializations: applied in scenario include order before scenario starts – later ones may override earlier ones

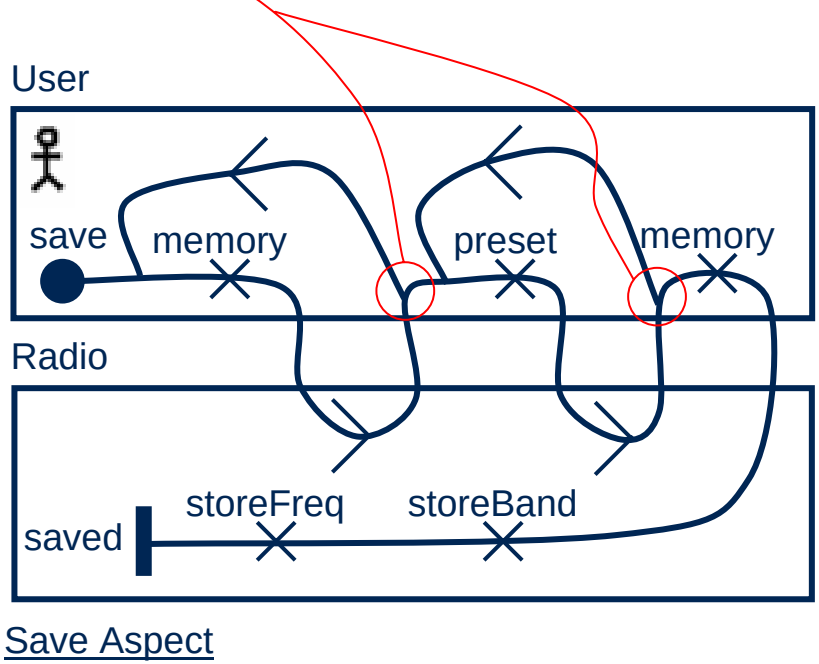
Scenario Definition	Scenario One	Scenario Two	Combined Scenario
Included Scenarios	n/a	n/a	One, Two
Start Points	start1	start2	start1, start2
Initialization	var1 = X, var2 = Y	var1 = Z	var1 = Z, var2 = Y
End Points	end1	end2	end1, end2
Postcondition	var2 = A	var1 = B	var1 = B, var2 = A

  - Elements may be added to combined scenario definitions
- Next start point of scenario definition is only considered if the traversal of the previous start point is finished or stuck

Problem 3

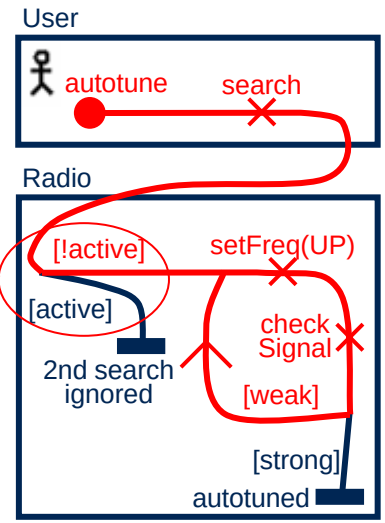
3) Control variables for loops (counters)

- NEW Traversal mechanism exposes element hit count
- _hitCount



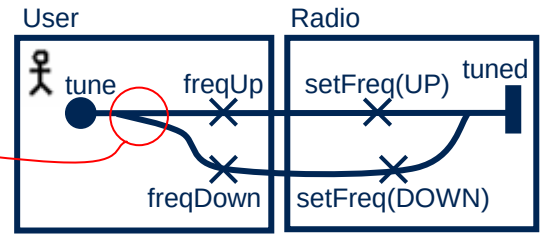
Problems 1, 2, and 4

1) Scenario stops at path element that is not an end point (e.g. OR-fork of infinite loop)
   - NEW Allow any UCM element to be end of scenario
2) <feature>Active variables
   - NEW Traversal mechanism exposes activity status
   - _activeCount



Autotune Aspect

4) Same feature several times in a row
   - NEW Support array type variables
   - TuneDirection = { UP, UP, DOWN, UP, DOWN, UP… }



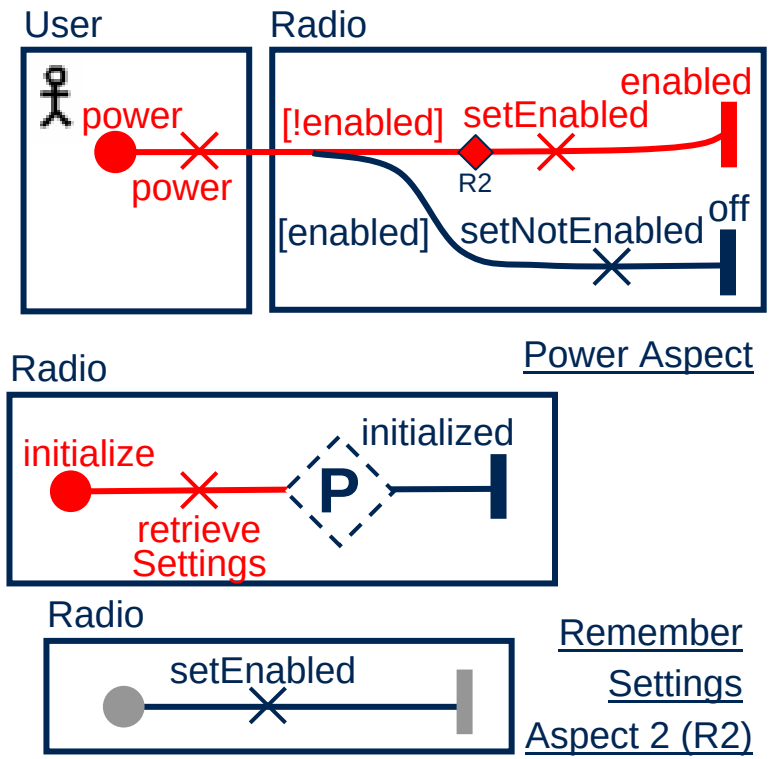
Tune Aspect

Problem 5

5) Redundant start points

Feature	Start Points
Power On	power
Remember Settings	initialize
Interaction	power, initialize

- The Remember Settings map is traversed another time after the enabled end point was reached
- NEW declare start point initialize as redundant



Problems 6 and 7

6) Changes to scenario definition

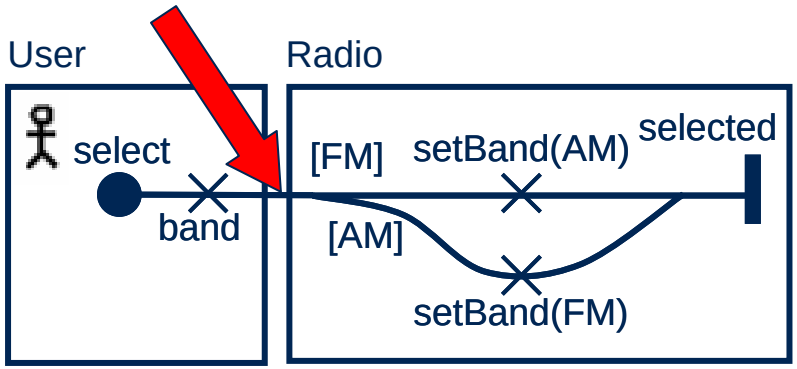
- NEW Elements may be deleted from scenario definitions

7) Interrupting a scenario

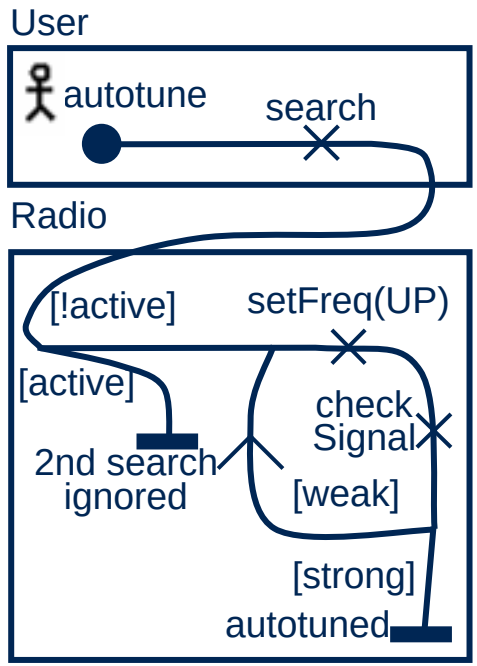
- NEW Allow interruption of a scenario before/after a path element if a condition evaluates to true

Example of Feature Interaction: Abort Autotune

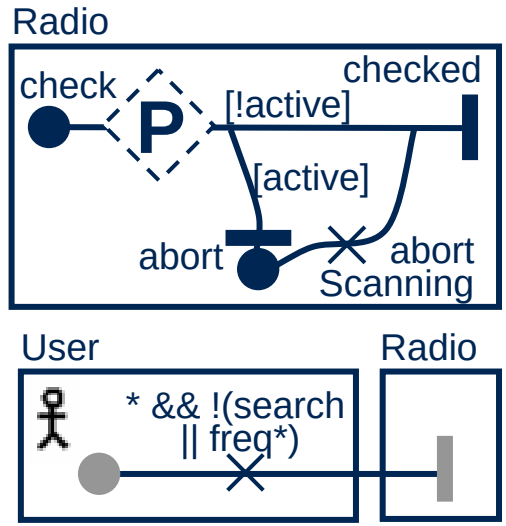
- Step 3: Model FI resolution
  - Autotune must be aborted when Select Band, Save, Recall, Power Off, or Standby Off are activated
  - Modeled as part of the Autotune feature concern



Select Band Aspect



Autotune Aspect



Autotune Aspect – Abort (Aa)

Example of Feature Interaction: Abort Autotune

- Step 4: Create scenario definitions for FI resolution
  - Interaction Scenario Definition:

    include Autotune, Select Band

    interrupt after OR-join in Autotune if _hitCount == 50

    delete OR-join**; add** abort

Scenario Definition	Autotune	Select Band	Interaction
Included Scenarios	n/a	n/a	Autotune, Select Band
Start Points	autotune	select	autotune, select
Initialization	Freq = 32, StrongFreq = -1	Band = AM	Freq = 32, StrongFreq = -1 Band = AM
End Points	OR-join	selected	~~**OR-join**~~, **abort**, selected
Postcondition	Freq != StrongFreq	Band == FM	Freq != StrongFreq, Band = FM

Feature Interactions

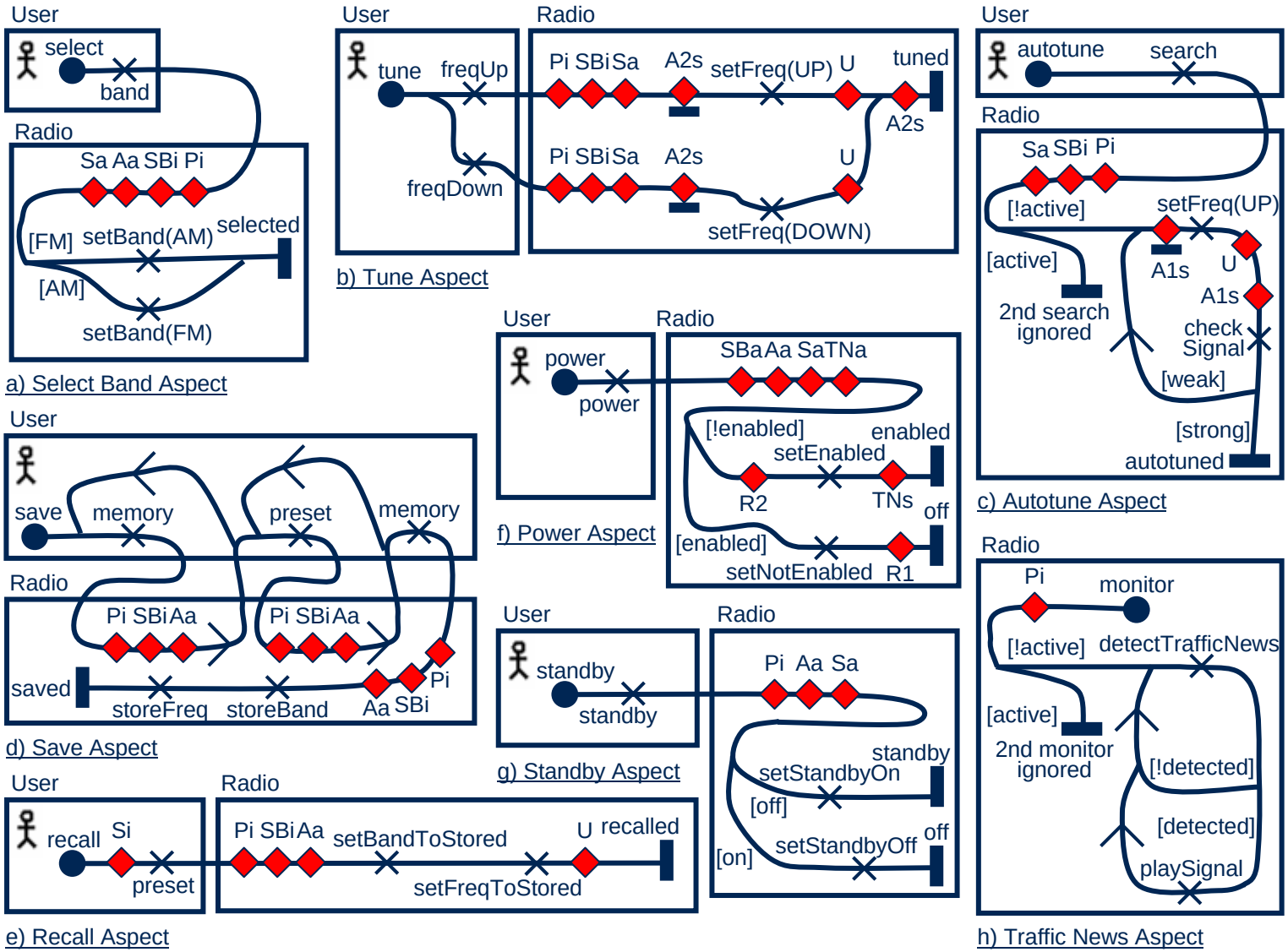
		Next									
#	**Current**	Sel. B.	Tune	Autotune	Save	Recall	Tr. N.	P. On	P. Off	S. On	S. Off
1	Select Band							Imp.			Imp.
2	Tune							Imp.			Imp.
3	Autotune	Abort	Synergy	Ignore	Abort	Abort		Imp.	Abort	Abort	Imp.
4	Save	Abort	Abort	Abort		Ignore		Imp.	Abort	Abort	Imp.
5	Recall							Imp.			Imp.
6	Traffic N.						Ignore	Imp.	Abort		Imp.
7	Power On						Synergy	Imp.			
8	Power Off	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore		Imp.	Ignore	Ignore
9	Standby On	Ignore	Ignore	Ignore	Ignore	Ignore		Imp.	Abort	Imp.	
10	Standby Off							Imp.			Imp.

#	**Features**	Sel. B.	Tune	Autotune	Save	Recall	Tr. N.	P. On	P. Off	S. On	S. Off
11	Update Display		ASF	ASF		ASF					
12	Remember Settings							ABF	BEF		

ASF…After setting frequency; ABF…After beginning of feature; BEF… Before end of feature

- 12 Abort: one feature aborts another
- 14 Ignore: one feature is ignored because of another feature
- 2 Synergy: creates something useful and new for user
- 3 for Display, 2 for Remember Settings
- Impossible: due to radio device interface constraints

Complete AoUCM Model



Conclusion

- Integrated aspect-oriented specification and validation scenario models at the requirements stage
- Incremental development of features
- Composition rules for aspect-oriented validation models
  - Include, add, delete, interrupt, redundant
- Reduced complexity of AoUCMs
- Improved scalability
- Feature concerns and FI resolution concerns are properly encapsulated
- Specification model is kept strictly separate from the validation model (thus further improving separation of concerns)

Future Work

- It may be possible to infer composition rules for the validation model from the aspect-oriented composition rules of the specification model
- It may be useful to provide more internal information of the path traversal mechanism
- Context information still needs to be addressed in scenario definitions
  - Requires the introduction of concrete component instances
  - Requires a rethinking of the global data model
  - Requires the introduction of scoping rules

