A Configurable UML Based Use Case Modeling Metamodel

Ľuboš Zelinka **Valentino Vranić**

Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology, Bratislava, Slovakia
vranic@fiit.stuba.sk

ECBS-EERC 2009, September 7–8, 2009, Novi Sad, Serbia

Overview

1. Diversity of Use Case Modeling Approaches
2. Use Case Description
3. Metamodel
4. Metamodel Configuration
5. Summary

No Uniform Approach to Use Case Modeling

- UML support of use case modeling gives a false impression of uniformness
- Approaches to use case modeling are quite different
- This is so with respect to use case diagrams
- But even more with respect to the main part of use cases: their description

Tool Support Requires a Configurable Metamodel

- As with any other modeling technique, tool support is needed
- The tool support of a particular notation cannot possibly exist without making it clear what is, and what is not a part of the notation
- This can be achieved by a metamodel and is important also for a consistent notation application
- To support multiple notations, a *configurable metamodel* is needed
- Each configuration would define a particular notation
- Some use case modeling tools provide a sophisticated support of use case description (e.g. Visual Use Case and CaseComplete), but they are not configurable

Elements of Use Case Description

- A use case describes a coherent functionality that provides some result of value to a user—it is a case of a system use
- Name and brief description
- Actors
- Preconditions and postconditions
- Flows of events
- Use case relationships: include and extend
- Many different ways of describing use cases, but rooted in Jacobson's or Cockburn's notation

Jacobson's Notation (1)

Use Case: Reserve Room
Basic Flows:
B1. Reserve Room ← *multiple main flows*
The use case begins when a customer wants to reserve a room.

1. The customer selects to reserve a room.
2. The system displays the types of rooms the hotel has and their rates.
3. The customer **Check Room Cost**.

...

Alternate Flows:
A1. Duplicate Submission If in step 5 of the basic flow there is...

1. If the customer wants to continue...

Subflows:
S1. Check Room Cost

1. The customer selects his desired room type...

Extension Points:

E1. Update Room Availability The Update Room Availability extension point occurs at step 5 of the Basic Flow. ← *EP defined by a step number*

Jacobson's Notation (2)

Use Case: Handle Waiting List[1]
Extension Flows: ← *actually alternative flows in a separate UC*
EF1. Queue for Room This extension flow occurs at the extension point Update Room Availability in the Reserve Room use case when there are no Rooms of the selected type available.

1. The system creates a pending reservation with a unique identifier for the selected Room type.

   ...

[1] K. Bittner and I. Spence. *Use Case Modeling*. Addison-Wesley, 2002.

Cockburn's Notation (1)

Use Case: Edit a document[2]
Primary actor: user
Scope: Wapp
Level: user goal
Trigger: User opens the application.
Precondition: none
Main success scenario: ← *no multiple basic flows*

1. User opens a document to edit.

   ...

Use Case: Check spelling Primary actor: user
Scope: Wapp
Level: subfunction!
Precondition A document is open
Trigger: Anytime in **Edit a document** that the document is open and the user selects to run the spell checker. ← *implicit extension: no EP*
Main success scenario:

...

[2] A. Cockburn. *Writing Effective Use Cases*. Addison-Wesley, 2000.

Objectives

- Cover the diversity in use case modeling
- Provide a basis for the development of configurable use case modeling tools
- Align with the UML metamodel (not meant as a UML specification extension)

Related Work

- OMG's UML use case metamodel—a common diagrammatical use case modeling notation
- An extension of the UML metamodel to support a textual use case description[3]
- Several proposals of limited use case modeling metamodels and formalizations targeting only a single notation or specific issues
- The metamodel proposed here
  - Doesn't aim at unifying different notations
  - It attempts to map the common and variable among them and to provide a way to opt for a particular notation or their combination

[3] A. N. Veit Hoffmann, Horst Lichter. Towards the integration of UML- and textual use case modeling. *Journal of Object Technology*, 8(3):85–100, 2009.

Flows

A Configurable UML Based Use Case Modeling Metamodel

Ľ. Zelinka, V. Vranić

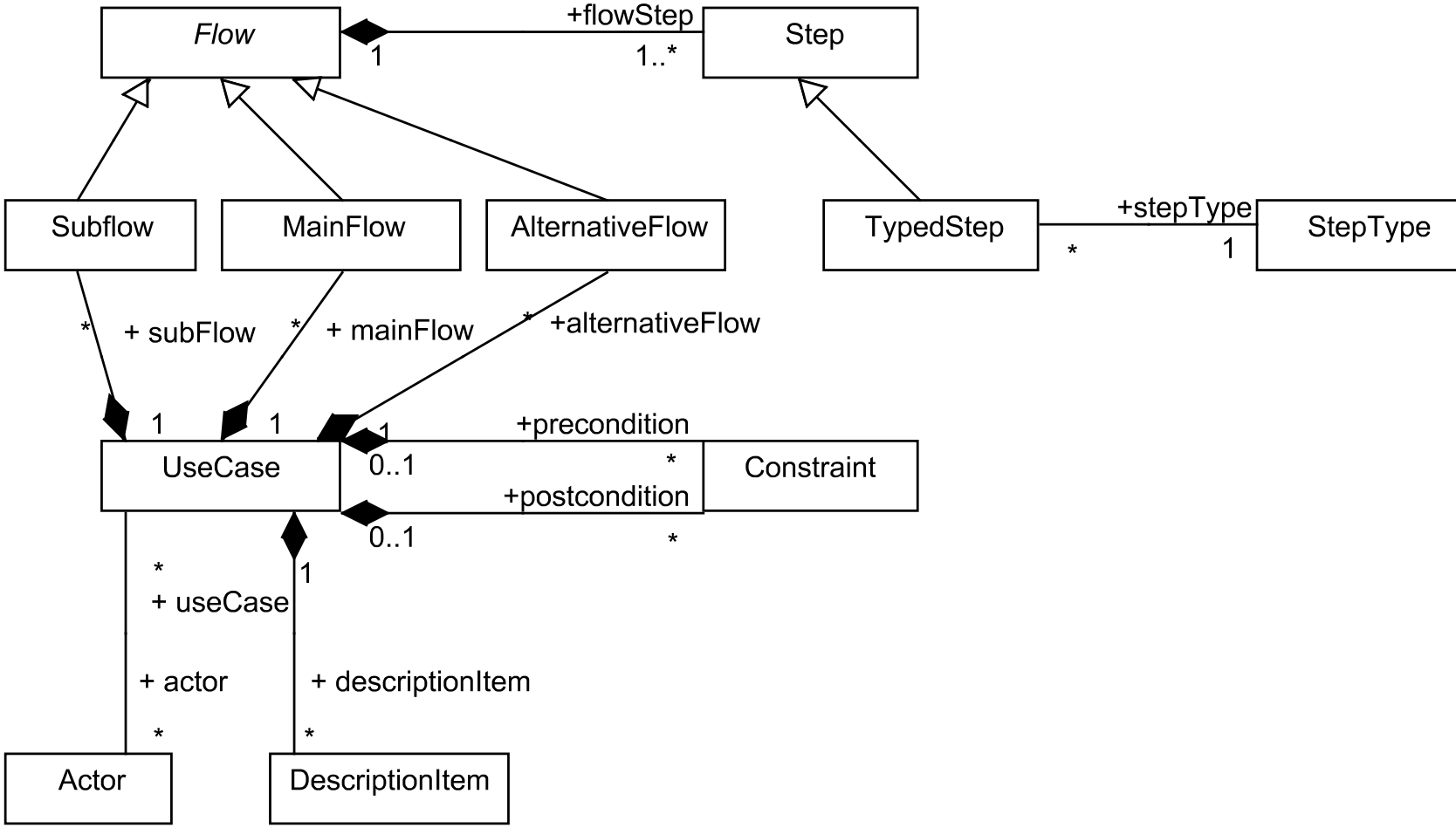
Diversity of Use Case Modeling Approaches

Use Case Description

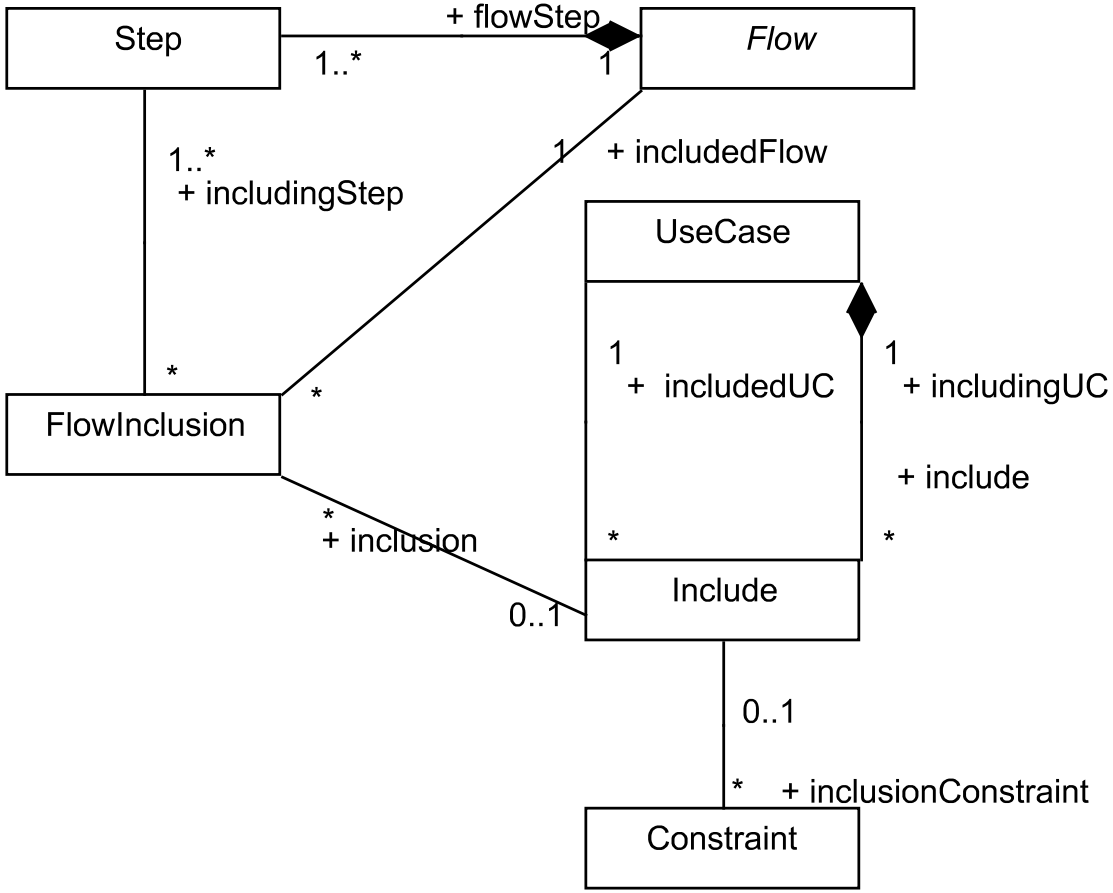
Metamodel

Metamodel Configuration

Summary



Include


Step
+ flowStep
1..*
1
Flow
1..*
+ includingStep
1
+ includedFlow
UseCase
*
*
1
+ includedUC
1
+ includingUC
+ include
FlowInclusion
*
+ inclusion
*
*
0..1
Include
0..1
*
+ inclusionConstraint
Constraint

Extend

A Configurable UML Based Use Case Modeling Metamodel

Ľ. Zelinka, V. Vranić

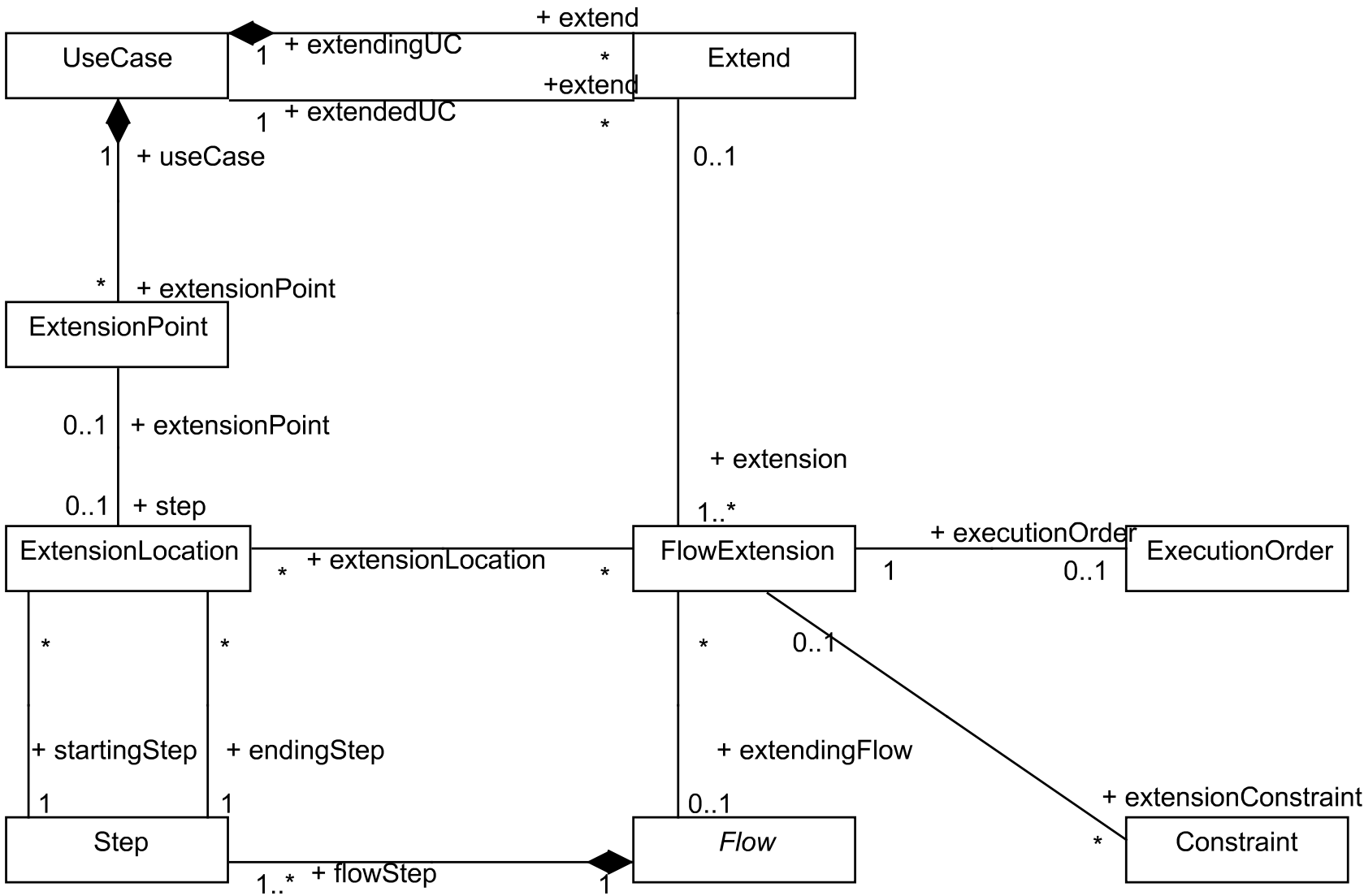
Diversity of Use Case Modeling Approaches

Use Case Description

Metamodel

Metamodel Configuration

Summary



Metamodel Configuration

- The proposed metamodel is configurable
  - By including or omitting some of its elements
  - Posing some constraints on their use
  - By constraining multiplicity of relationships among them
- Configurations represent individual use case modeling notations
  - As a basis for tool development and configuration
  - Defines the syntax that should be followed manually

Jacobson's and Cockburn's Notation Profiles

Property	J	C
Single-Step Extension Points	Y	N
Range Extension Points	Y	N
Mandatory Main Flow	Y	Y
Multiple Main Flows	Y	N
Subflows	Y	Y
Subflows in Main Flows	Y	Y
Subflows in Alternative Flows	Y	Y
Subflows in Subflows	Y	Y
Alternative Flows	Y	Y
Alternative Flows in Main Flows	Y	Y
Alternative Flows in Subflows	Y	Y
Alternative Flows in Alternative Flows	Y	Y
Extension	Y	N
Multiple Extension Locations in an Extension	N	N
Extension by a Specific Flow	Y	N
Extension Flow Constraint	Y	N
Extension Flow Execution Order	Y	N
Inclusion	Y	Y
Inclusion Flow Constraint	N	N
Inclusion of a Specific Flow	N	N

Evaluation

A Configurable UML Based Use Case Modeling Metamodel

Ľ. Zelinka, V. Vranić

Diversity of Use Case Modeling Approaches

Use Case Description

Metamodel

Metamodel Configuration

Summary

- A prototype of a configurable use case modeling tool has been developed
- Supports the textual part, not diagrams
- The goal: to test the metamodel and choice of configuration options in practice
- A possibility of having inconsistent configurations of options has been identified
- This is actually a feature interaction problem: points to feature modeling

Summary

- A use case modeling metamodel has been proposed
- Based on UML metamodel
- Embraces mainly Jacobson's and Cockburn's use case notation
- The metamodel is configurable
- A prototype tool developed
- Further work
  - Embrace other possibilities of use case modeling in the metamodel
  - Explore the possibility of employing feature modeling to express configurability
  - Integration into the UML metamodel is a part of our