

A Rule-Based, Integrated Modelling Approach for Object-Oriented Systems

Benjamin Braatz

5th International Workshop on
Graph Transformation and Visual Modeling Techniques
(GT-VMT 2006)

Sunday, 2 April 2006

Motivation

- Goal: Precise, Constructive Object-Oriented Models
(Necessary for Formal Semantics and Code Generation)
- Separation of Concerns:
Structure, Local Changes, and Control Flow
- Declarative, Rule-Based Modelling to ease
Comprehension

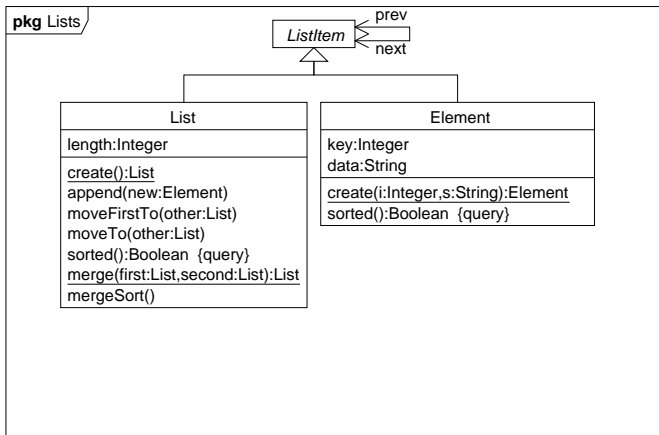
Overview

- 1 UML Foundations
- 2 Transformation Rules
- 3 Structured Flowcharts

UML Foundations

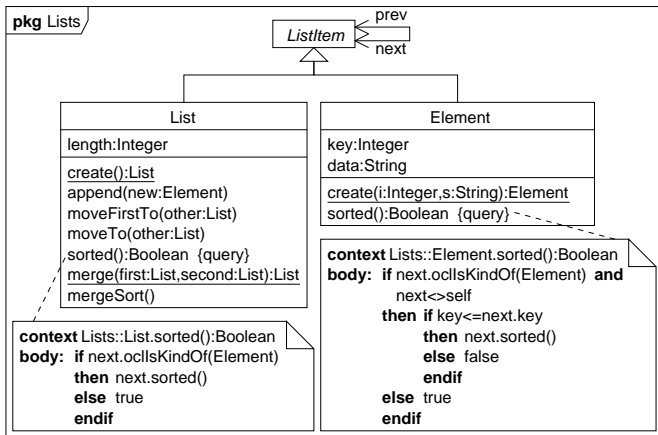
- Class Diagrams for Structure
- Declaration of Entities of the Object-Oriented System \implies
Comparable to Signatures in Algebraic Specifications
- OCL Constraints for Queries
- Stateless Language without Side Effects \implies
Comparable to Functional Programming Languages

Class Diagrams



- Classes and Associations
- Inheritance Structure
- Attributes
- Operations
- Static Operations
- Query Operations

OCL Body Expressions

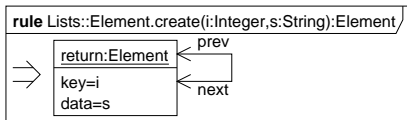
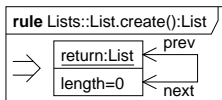


- OCL Body Expressions specify Queries
- if-then-else Constructs and Calls to other Queries
- Recursion

Transformation Rules

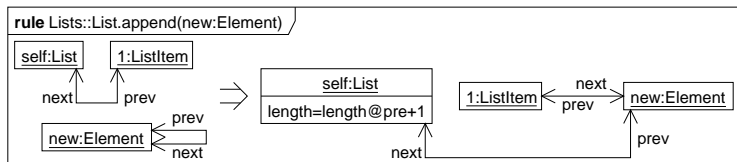
- Declarative Modelling of Local Structure Changes
- Adaption of Graph Transformation for Object Structures
- Intuitive Notation with Precise Semantics (SPO or DPO)

Creation Rules



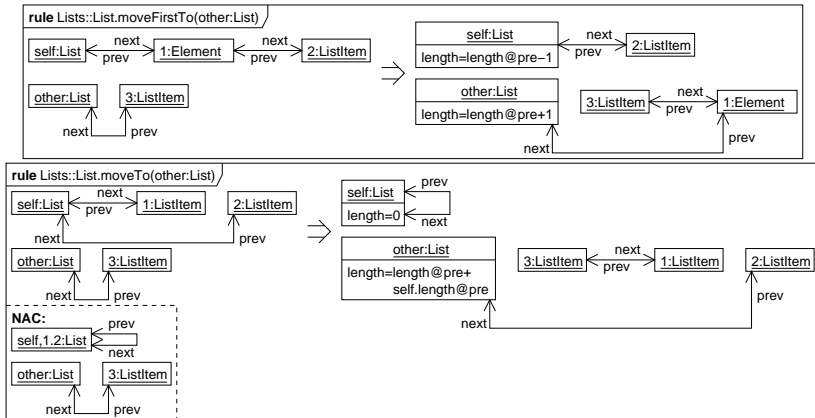
- Constructors are Static Operations
 ⇒ Left-Hand-Sides are empty
- Right-Hand-Sides contain
 Constructed Object (return)
- Right-Hand-Sides use Parameters
 of Operation

Modification Rules (1)



- Left-Hand-Sides contain Parameters and self
- Left-Hand-Sides navigable from Parameters \implies Match completely determined by Parameters
- Right-Hand-Sides contain modified Structure
- Right-Hand-Sides use OCL Expressions for Attributes

Modification Rules (2)

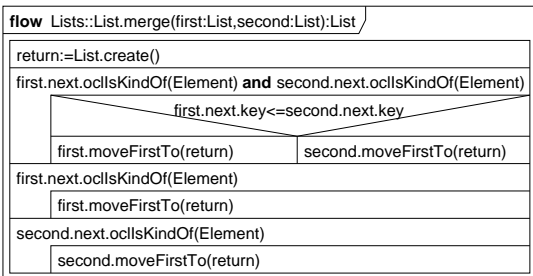


- Negative Application Conditions to prohibit Structures

Structured Flowcharts

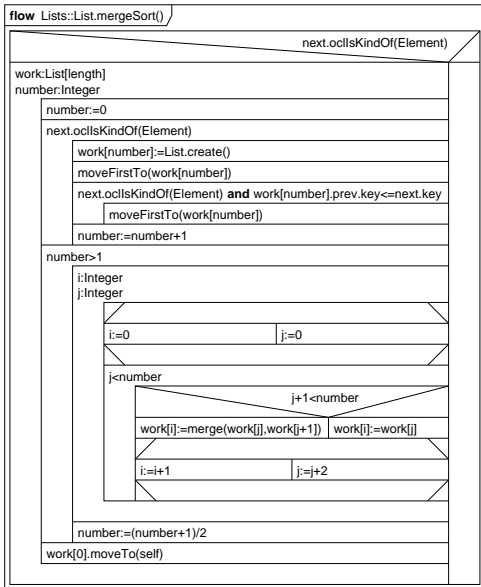
- Need for Control Layer on Top of Transformation Rules
- Nassi-Shneidermann-Diagrams: Well-Known Technique for Modelling Control Structures
- Context-Free Block Structure \implies
Close to Imperative Programming Languages and
Suitable for Top-Down Modelling

Merge Operation



- Variable Assignment with Creation Rule Call
- While-Loop over OCL Query
- Decision on OCL Query
- Modification Rule Calls

Mergesort Operation



- Local Variable Declaration
- Parallel Execution of Control Blocks

Summary

- Integrated Modelling Approach with Separation of Concerns
- Structure specified by Class Diagrams
- Queries specified by OCL Constraints
- Local Modifications specified by Transformation Rules
- Control Flow specified by Structured Flowcharts

Future Work

- Formal Syntax given by Meta-Model and Typed, Attributed Graph Transformation
- Formal Semantics
- Extension by Descriptive Techniques with Verification Methods
- Refinement Concept

The End

Thank You!