

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

Benjamin Braatz

TU Berlin, TFS, Forschungskolloquium

16 January 2006

# Motivation

- Goal: Complete Code Generation from Models
- Need for Precise, Constructive Modelling Approach
- Separation of Concerns: Structure, Local Changes, and Control Flow
- Declarative, Rule-Based Modelling to ease Comprehension

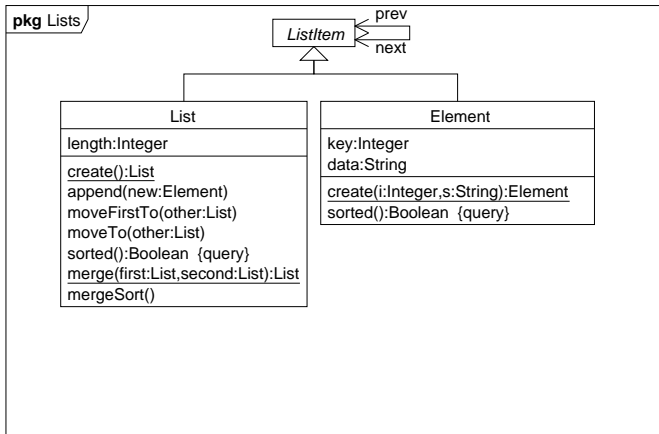
# Overview

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

# UML Foundations

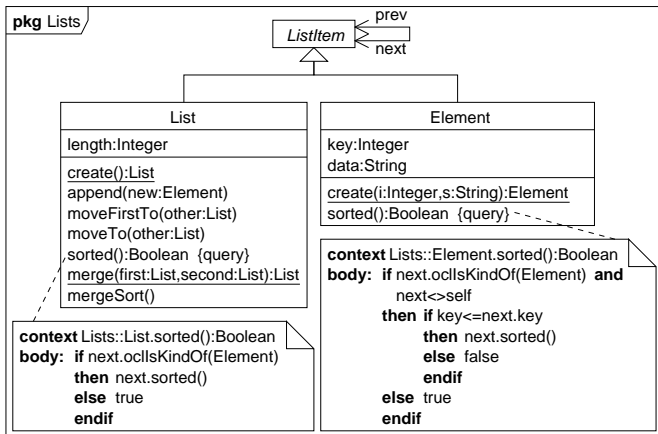
- Class Diagrams for Structure
- Declaration of Entities of the Object-Oriented System  $\implies$  Comparable to Signatures in Algebraic Specifications
- OCL Constraints for Queries
- Stateless Language without Side Effects  $\implies$  Comparable to Functional Programming Languages

# Class Diagrams



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

# OCL Body Expressions

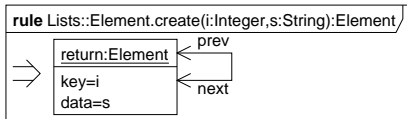
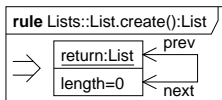


- OCL Body Expressions specify Queries
- if-then-else Constructs and Calls to other Queries
- Recursion

# Transformation Rules

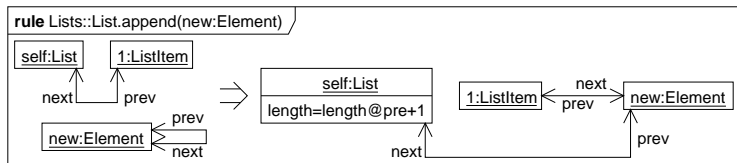
- Declarative Modelling of Local Structure Changes
- Adaption of Graph Transformation for Object Structures
- Intuitive Notation with Precise Semantics (SPO or DPO)

# Creation Rules



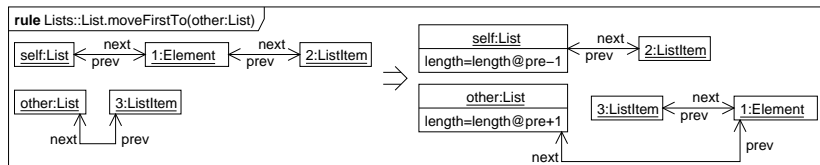
- Constructors are Static Operations  
⇒ Left-Hand-Sides are empty
- Right-Hand-Sides contain Constructed Object (return)
- Right-Hand-Sides may use Parameters of Operation

# Modification Rules (1)



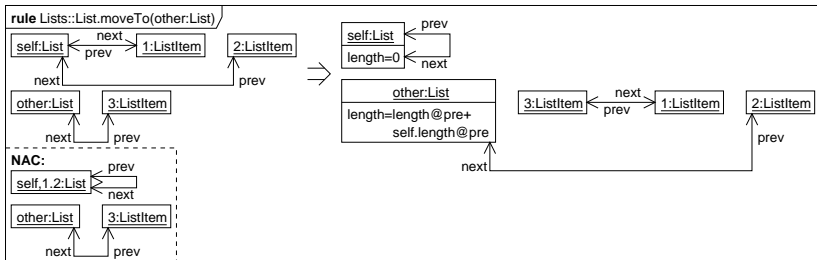
- Left-Hand-Sides contain Parameters and self
- Left-Hand-Sides navigable from Parameters  $\implies$  Match completely determined by Parameters
- Right-Hand-Sides contain modified Structure

## Modification Rules (2)



- Right-Hand-Sides may use OCL Expressions for Attributes

# Modification Rules (3)

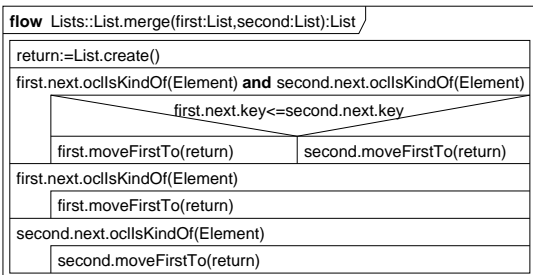


- Negative Application Conditions may be used to prohibit Structures

# Structured Flowcharts

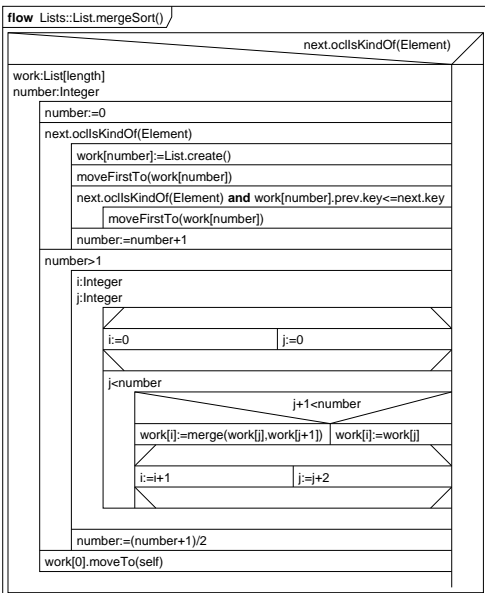
- Need for Control Layer on Top of Transformation Rules
- Nassi-Shneidermann-Diagrams: Well-Known Technique for Modelling Control Structures
- Context-Free Block Structure  $\implies$  Close to Imperative Programming Languages and Suitable for Top-Down Modelling

# 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 given by Object-Oriented Transformation System
- Extension by Descriptive Techniques with Verification Methods
- Extension by other Behavioural UML Techniques (Sequence Diagrams, State Machines)