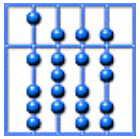


A Playful Approach to Formal Descriptions

A Field Report on Using Modeling Fundamentals in Middle School

Katharina Spies and Bernhard Schätz
Fakultät für Informatik, TU München
Werner-Heisenberg-Gymnasium Garching)

FM‘06 Workshop on Formal Methods in the Teaching Lab
McMaster University, 26. August 2006

Context: Informatics in Middle School

- Situation: Informatics at Middle School in Germany
  - Obligatory class (since summer 2004)
  - Vague curriculum
  - Focus on IT use (e.g., office tools, internet)
  - Integeration of OO techniques suggested

- Setting: Teaching for courses at 7th grade
  - 4 classes with ca. 15 pupils
  - Average age 12-13, both male and female
  - Class duration 6 month
  - Tryout period of 2 years

Three levels of informatics

- Model level: “Theoretical informatics”
  - Focus: Modeling of a problem domain
  - Approach: Providing abstract concepts to based descriptions on

- Description level: “Programming”
  - Focus: Solution of a specific problem
  - Approach: Providing formalisms for descriptions

- Method level: “Software engineering”
  - Focus: Systematic development of solutions
  - Approach: Providing guidelines for the construction of good descriptions

Goal: Teaching basic principles rather then specific techniques

Goals

Conceptual goal: Basic abstract models of computer science

1. Data models
2. Algorithmic models
3. Reactive models

Methodical goal: Advantages of using models

1. Exploiting the preciseness of models to avoid ambiguities
2. Exploiting the executability of models to implement systems
3. Exploiting the structure of models to establish quality

Side goal (curriculum): Computer experience

- Basic knowledge in standard operating and application software
- Experience in handling the computer as a tool

Teaching Approach: Playful Discovery of Basic Models

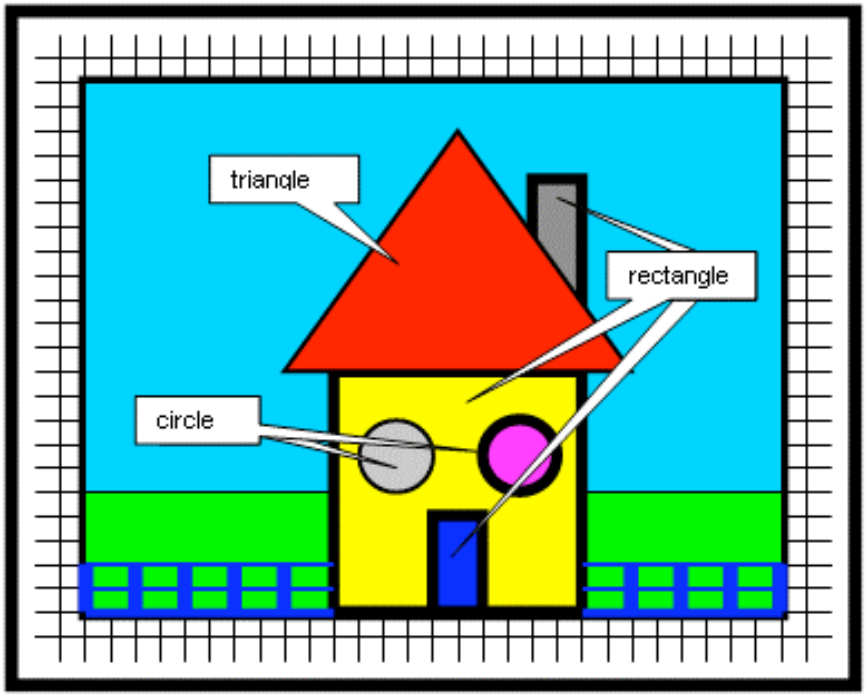
Overall motivation:

1. Lern basic models instead of technical specifica
2. Experience models as helpful tools
3. Have fun!!!

General learning path:

1. Experience problem context in a game
2. Discover modeling concepts
3. Apply modeling concepts to create solution description
4. Reflect on the advantages of the approach

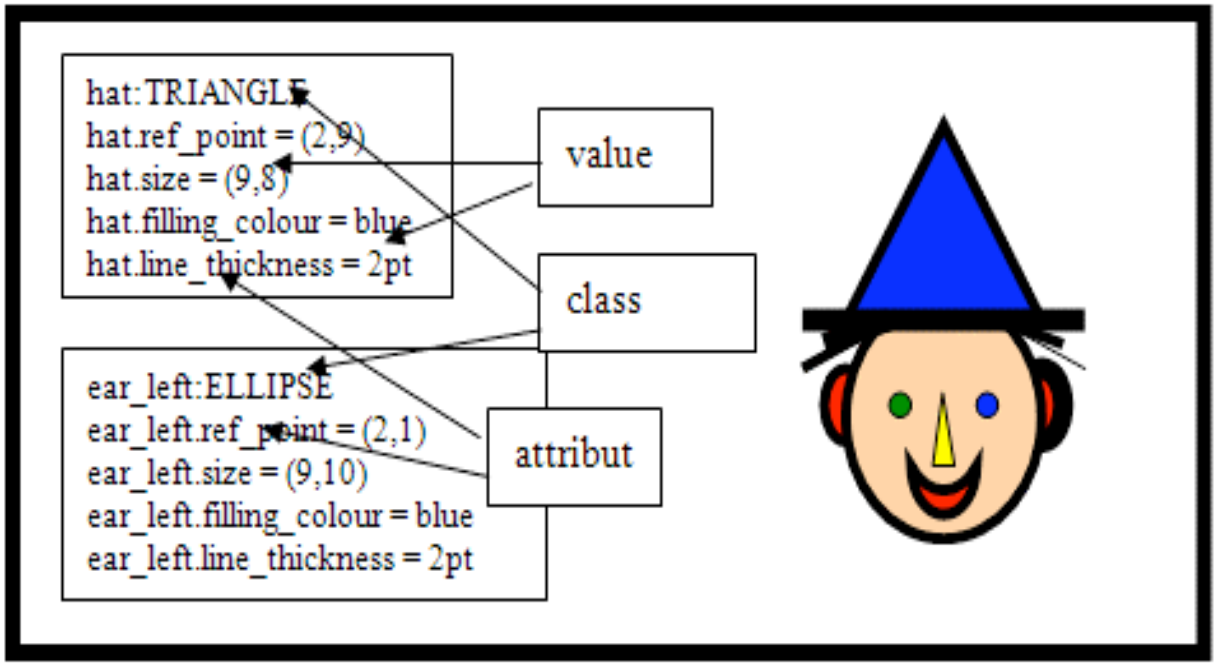
Data Descriptions: Problem Statement and Domain



Context: Description of static information

- Problem statement: Describe a picture
- Informatics domain: Data Descriptions

Data Descriptions: Concepts



Basic concepts:

1. Classes: Geometrical shapes (e.g., line, circle, triangle)
2. Attributes: Shape characteristices (e.g., color, position, size)
3. Values: Domain of characterisitics (e.g., red, blue, green)
4. Objects: Specific instances (e.g., hat.size = (9,8))

Data Descriptions: Exercises and Experiences

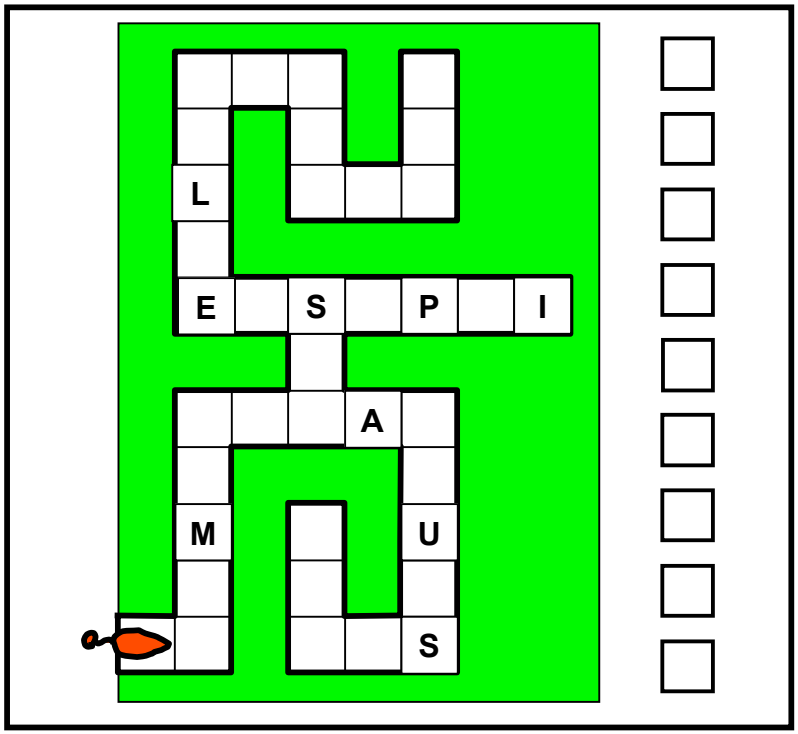
Exercises: Informal text vs. structured description

1. Construct description from picture using informal language
2. Construct picture from description using informal language
3. Construct description from picture using modeling concepts
4. Construct picture from description using modeling concepts

Experiences: Preciseness of descriptions

1. Compact, structured description
   $\Rightarrow$ eases construction of description
2. Unambiguous, precises description
   $\Rightarrow$ makes models interchangable/reusable

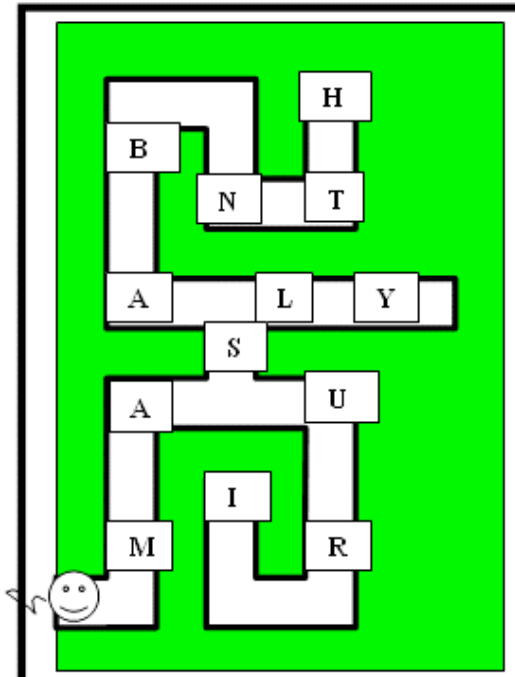
Algorithmic Descriptions: Problem Statement and Domain



Context: Description of change of static information

- Problem statement: Direct a mouse through a labyrinth
- Informatics domain: Algorithmic descriptions

Algorithmic Descriptions: Concepts



Mouse. run (1)	Mouse. right	Mouse. run (4)	Mouse. run (2)
Mouse. left	Mouse. run (3)	Mouse. right	Mouse. left
Mouse. run (4)	Mouse. left	Mouse. run (2)	Mouse. run (2)
Mouse. right	Mouse. left	Mouse. right	Mouse. right
Mouse. run (4)	Mouse. run (3)	Mouse. run (2)	Mouse. run (4)
Mouse. left	Mouse. left	Mouse. left	Mouse. right
Mouse. left	Mouse. run (5)	Mouse. left	Mouse. run (2)
Mouse. run (2)	Mouse. left	Mouse. run (2)	Mouse. right
Mouse. right	Mouse. left	Mouse. left	Mouse. run (2)
Mouse. run (2)	Mouse. run (3)	Mouse. run (2)	Mouse. right
Mouse. right	Mouse. left	Mouse. left	Mouse. run (2)
Mouse. run (1)	Mouse. run (2)	Mouse. run (4)	Mouse. left
Mouse. left	Mouse. left	Mouse. left	Mouse. run (2)
Mouse. left	Mouse. run (2)	Mouse. run (2)	
Mouse. run (3)	Mouse. right	Mouse. right	

Basic concepts:

1. State: Assignment of values (e.g., position = (1,1), orientation = up)
2. State transformation: Change of state (e.g., position = (1,1)→(1,2))
3. Program: Collection of state changes (e.g., left = down→right, right →up)

Algorithmic Descriptions: Exercises and Experiences

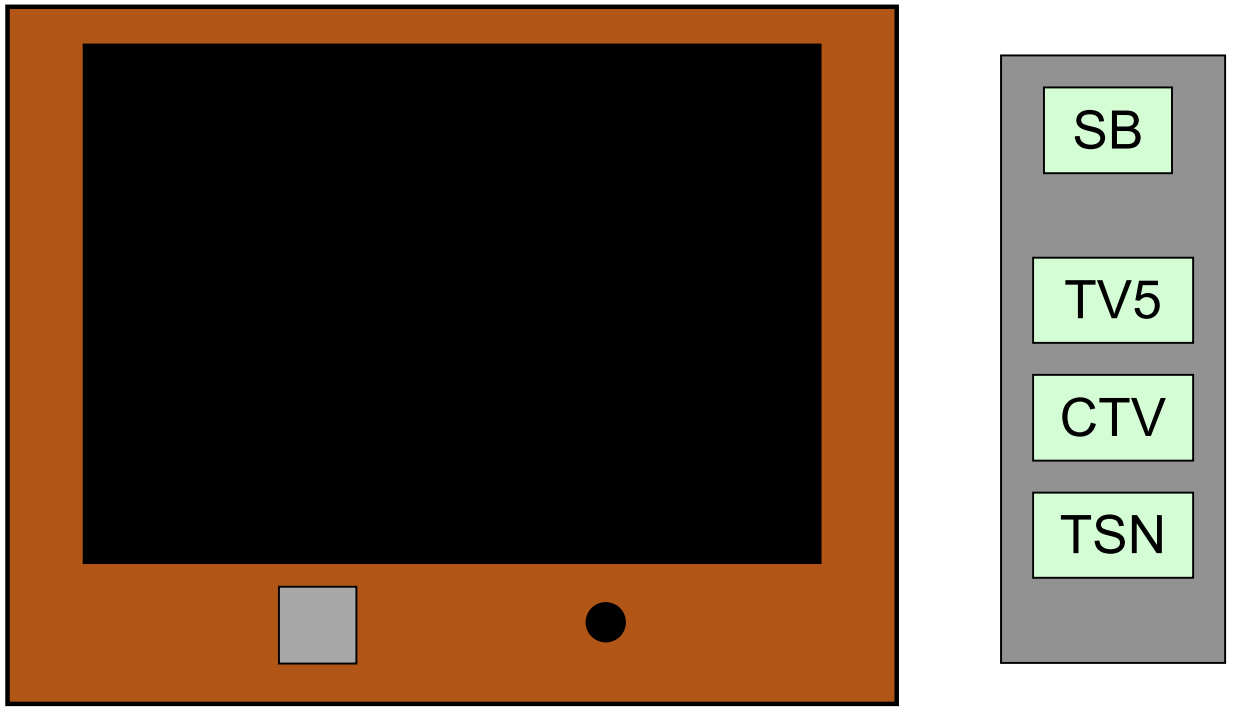
Exercises: Description of problem solution

1. Construct description for solution using informal language
2. Obtained detailed description for individual steps
3. Perform execution of detailed description
4. Compare different solution descriptions

Experiences: Executability of descriptions

1. Detailed, precise description
   $\Rightarrow$ allows automatic execution of description
2. Composable description
   $\Rightarrow$ allows different solution descriptions to the same problem

System Descriptions: Problem Statement and Domain



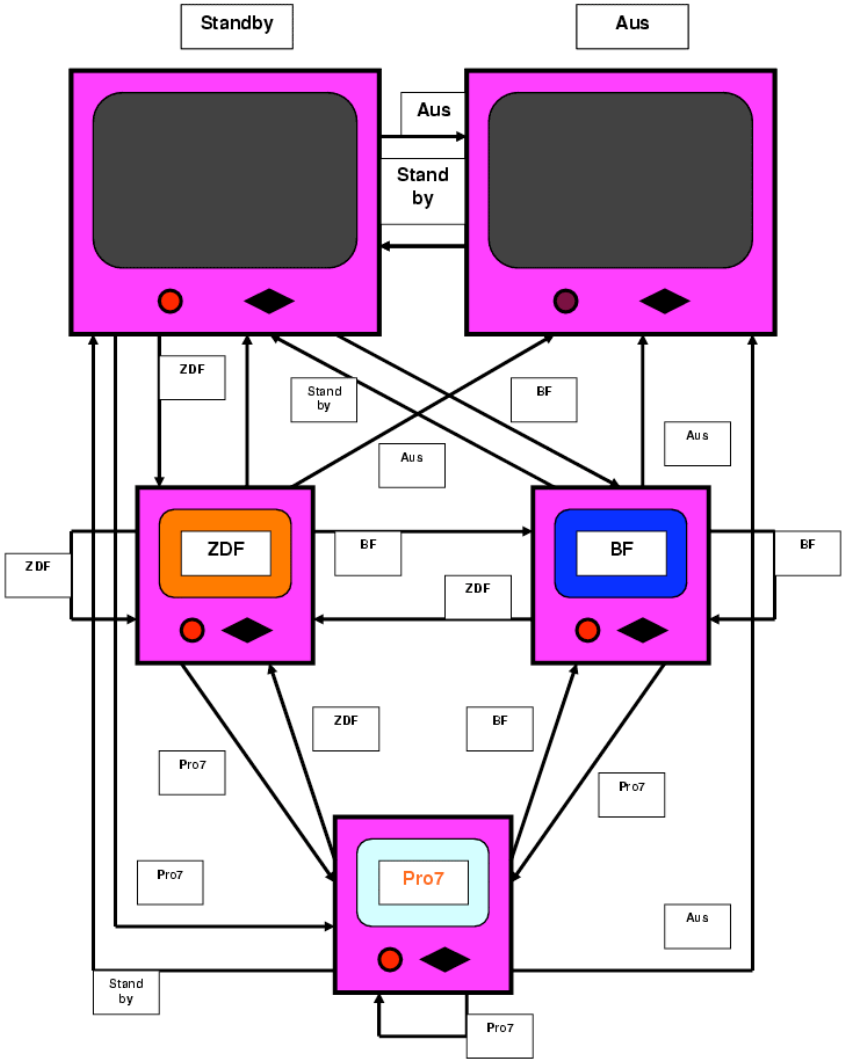
Context: Description of reactive behavior

- Problem statement: Describe the behavior of a TV set
- Informatics domain: Systems description

System Descriptions: Concepts

Basic concepts:

1. Modus: Characterization of state of system (e.g., turned off, stand by, channel 1)
2. Event: Interaction with system (e.g., pressing button 1)
3. Transition: Change of modus, triggered by event (e.g., from stand by to channel 1 via button 1)



System Descriptions: Exercises and Experiences

Exercises: Create description of TV set behavior

1. Describe the behavior of the system informally
2. Describe the interfaces and the states of the system
3. Describe the transitions of the system
4. Check the behavior of the system with respect to interfaces/states

Experiences: Analyzability of descriptions

1. Compact, structured description
   $\Rightarrow$ eases systematic construction of description
2. Unambiguous, precise description
   $\Rightarrow$ makes models analyzable

Experiences (1)

- Basic computer science models suitable for middle school
  - Simplified but correct models
  - Models generally easily understood
  - Approaches transferrable to new problems

- Middle graders are interested in basic topics
  - Strong active participation
  - No distinction between male/female participants
  - Sustained interest after courses

- Motivation is achieved by
  - “Self-discovery” of concepts
  - “Self-experience” of methods
  - Playful scenarios as problem statements

Experiences (2)

- Applied Tools:
  - Class can be performed with office applications
  - Additional specialized tools to enhance learning experience/motivation (e.g., EOS, LTSA)

- Current curriculum:
  - Focus on description level (OO-programming)
  - Little emphasis on model level (concepts of OO)
  - No emphasis on methodical level

  - Problem: Teachers generally have no informatics background