A Study of the Development of Students’ Visualizations of Program State during an OO Programming Course

Jorma Sajaniemi, Marja Kuittinen, Taina Tikansalo
University of Joensuu, Finland

ICER07, 15-16.9.2007, Atlanta, GA, USA
Outline

- Introduction
- Study
- Results: Examples of drawings
- Analysis
  - Overall approach
  - Content elements at the end
  - Content element development
  - Common errors
- Conclusion
Introduction

- Novices’ understanding of OO programming:
  - misconceptions of central concepts
  - understanding of dynamic aspects?
  - understanding of the **notional machine**?

- Notional machine well understood in **imperative** case
  - novices’ abilities to use visualizations are **good**

- Novices’ ability to use **OO**
  - teachers’ drawings **poor**
  - unable to draw own picture
  - **poor** ability to utilize own visualizations

- The study: students’ own visualizations of **program state**?
Study: Task

"Draw a picture that includes existing objects and methods and their relationships when the attached program is at the marked point of execution."

- Unspecific instruction → participants report what they see important and salient (Pennington, 1987; Good & Brna, 2004)
- "Program state” vs. ”existing objects …”
- Participants also guided by lectures and textbooks
- Programs 32–39 LOC (1-3 classes)
Study: Visualization Tools Used

- **Object**
- **Method call**
- **Object reference**
- **Class**

**Jeliot**

**OO metaphors**
# Study: Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Tool Use</th>
<th>Drawing Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd</td>
<td>Jeliot, prog A</td>
<td></td>
</tr>
<tr>
<td>4th</td>
<td>Jeliot, prog B</td>
<td></td>
</tr>
<tr>
<td>5th</td>
<td></td>
<td>First, prog 1</td>
</tr>
<tr>
<td>6th</td>
<td>Metaphors, prog B</td>
<td></td>
</tr>
<tr>
<td>7th</td>
<td>Metaphors, prog A</td>
<td></td>
</tr>
<tr>
<td>8th</td>
<td></td>
<td>Second, prog 2</td>
</tr>
<tr>
<td>10th</td>
<td></td>
<td>Third, prog 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Week</th>
<th>Tool Use</th>
<th>Drawing Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd</td>
<td>Metaphors, prog A</td>
<td></td>
</tr>
<tr>
<td>4th</td>
<td>Metaphors, prog B</td>
<td></td>
</tr>
<tr>
<td>5th</td>
<td></td>
<td>First, prog 1</td>
</tr>
<tr>
<td>6th</td>
<td>Jeliot, prog B</td>
<td></td>
</tr>
<tr>
<td>7th</td>
<td>Jeliot, prog A</td>
<td></td>
</tr>
<tr>
<td>8th</td>
<td></td>
<td>Second, prog 2</td>
</tr>
<tr>
<td>10th</td>
<td></td>
<td>Third, prog 2</td>
</tr>
</tbody>
</table>
Study: Participants

- Mostly first-year CS students
- Course attended by 59 students (13 females)
- Tasks part of course duties
- 29-48 participants in each drawing task
- Assignment to group depended on students’ selection of lab exercise group
Results: The First Six Drawings

- What did we expect?
  - lecture style
  - textbook style
  - Jeliot style
  - Metaphors style

- What did we see?
The 1st Drawing

"Object drive"

Class

"Object main"

Object reference as a method call
The 2\textsuperscript{nd} Drawing

All objects

Method call

All methods

Object reference
The 3rd Drawing

Active object

Active method

"Train object"

Attributes

Application domain concept
The 4th Drawing

Objects that have been active

"Main program"

Method call
The 5th Drawing

All objects

Object reference

All attributes

All methods

"main"

All objects
The 6th Drawing

Textual description of actions

List of all objects

List of all methods
Analysis

- Overall approach
- Content elements at the end
- Content element development
- Common errors
Overall Approach (1/6)

- Overall approaches characterized by
  - central concept(s)
  - subordinate concept(s)
  
  - E.g., ”existing objects with subordinate active methods”

- 8 categories (A1..A8)

- Examples ...
Overall Approach (2/6)

Existing objects with subordinate active methods (A2)
Overall Approach (3/6)

Existing objects and all methods (A3)
Classes with subordinate active code and objects (A5)
Overall Approach (5/6)

Execution path with subordinate objects and methods (A7)

```
owner a

owner b

John

Pete

Pet

50kg food

20 kg food

20 + 100 food

50 food

"Cody"

"Duke"

"Toby"

-P4

-P5

-P4

P4

"Cody"

food: -5

P4
```


## Overall Approach (6/6)

<table>
<thead>
<tr>
<th>First</th>
<th>Second</th>
<th>Third</th>
<th>Code</th>
<th>Central Concepts</th>
<th>Subordinate Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>22</td>
<td>7</td>
<td>A1</td>
<td>Existing objects</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>44</td>
<td>55</td>
<td>A2</td>
<td>-”-</td>
<td>Active methods</td>
</tr>
<tr>
<td>44</td>
<td>19</td>
<td>10</td>
<td>A3</td>
<td>-”-, all methods</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>A4</td>
<td>All methods</td>
<td>Effects on objects</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>3</td>
<td>A5</td>
<td>Classes</td>
<td>Active code &amp; objects</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>3</td>
<td>A6</td>
<td>Active object</td>
<td>Important code</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>3</td>
<td>A7</td>
<td>Execution path</td>
<td>Objects, methods</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>10</td>
<td>A8</td>
<td>Active exec. path</td>
<td>-”-</td>
</tr>
</tbody>
</table>
Analysis

- Overall approach
- **Content elements at the end**
- Content element development
- Common errors
Element Analysis Technique

Dynamic existence of an exited method (C18)

Object (C6)

Dynamic existence of a parameter, id not visible (C27)

Data type (C2)

Method’s or constructor’s belonging to its class (C21)
Content Elements, 3rd Task (1/2)

- Classes 14%, objects 100%
- Attributes and their relationship to the object frequent

- Methods (past 48%, current 72%), constructors (7%), main (0%)
  → differences in mental representations of methods vs. constructors and main
Local variables with identifiers; parameters without identifiers $\rightarrow$ differences in ...

Relationships between attributes/parameters/local variables and the object/method represented almost always, but ...

local variables of the main method never attached to main but usually to the object they referred to!
Analysis

- Overall approach
- Content elements at the end
- **Content element development**
- Common errors
**Content Element Development**

- Existing objects, **all methods** (A3) → Existing objects with subordinate **active methods** (A2):
  - **static** existence of methods 59% → 17%
  - **static** existence of constructors 20% → 3%
  - **dynamic** call graph of methods 5% → 55%

- Identifiers left out more often:
  - object references (no id 24% → 62%)
  - attributes (no id 17% → 52%)
  - parameters (no id 46% → 55%)
Analysis

- Overall approach
- Content elements at the end
- Content element development
- Common errors
Common Errors (1/3)

- Wrong location of main method’s local variables

*Where is the main method located in students’ mental representations?*
Common Errors (2/3)

- Wrong direction of the relationship between method and the associated object

Students’ mental representations seem to possess various errors in how methods deal with attributes
Common Errors (3/3)

- Wrong direction of object references

*Breaks if object references do change*
Conclusion

- Central findings:
  - large variability of approaches
  - mental representations not only grow—they also change over time
  - problems with object-method relationships
  - problems with the main method

- Future work:
  - comparison with other institutions
  - ways to improve students’ understanding of the OO notional machine