

UML CASE Tools



Your friendly assistant

(<name>@inf.ethz.ch)

Chair of Programming Methodology

Agenda for Today

1. Why do we need UML tools?
2. Main UML tools
3. StarUML
4. Requirements Analysis exercise

Brief re-cap of UML

- Modeling language used in **analysis**, **design** and **implementation** phases.
- **Textual** and **graphical** notation to document specification.
- Main diagram types:
 - **structural**: **class**, component, deployment
 - **behavioral**: **use case**, **sequence**, **statechart**

Why do we need UML tools?

- Advantages of “e-design” over “paper-design”
 - modifications made easier
 - communication between designers easier
 - coordination and handling of large projects easier



Why do we need UML tools?

- Advantages of UML tools over drawing tools
 - uniform notations
 - certain checks provided

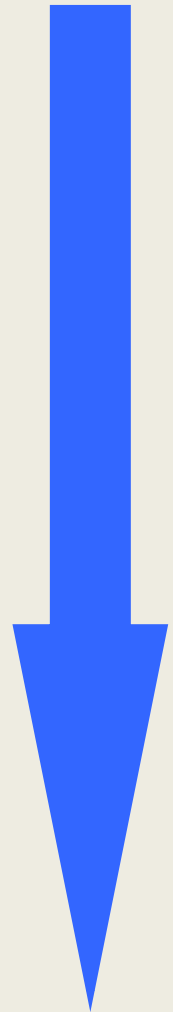


- UML tools provide add-ons

Add-ons

- Document generation
 - HTML, PDF, etc.
- Round-trip engineering
 - code generation from diagram (forward engineering)
 - diagram generation from code (reverse engineering)
- Test generation
- Simulation
- Model validation and verification

Level of sophistication



UML tools

- **Big competition** among UML tool vendors
 - <http://plg.uwaterloo.ca/~migod/uml.html>
 - http://objectsbydesign.com/tools/umltools_byCompany.html
- Many **free tools/editions** with limited capabilities
- **Compatibility** ensured by XML dialect
 - XMI – XML Metadata Interchange
 - may contain diagram layout info (Diagram Interchange Standard)

Main UML tools

- **Commercial**
 - Rational Rose (IBM)
 - Together Designer (Borland)
 - Rhapsody (I-Logix)
 - Poseidon (Gentleware)
- **Free**
 - UMLet
 - EclipseUML
 - Visual Paradigm (Visual Paradigm)
- **Open source**
 - ArgoUML (Tigris)
 - StarUML (Sourceforge)
- **And many, many more...**



Choosing a UML tool

- Platform support
- UML 2.0 and XMI support
- Support all diagram types
- Intuitive and clear GUI
- Add-ons
- Actively maintained
- Document generation and printing support

StarUML

- Pros
 - Open-source
 - UML 2.0 and XMI
 - All diagram types
 - Code generation (C++, C#, Java)
 - Diagram exporting and printing
- Cons
 - **Windows only**
 - Not maintained since 2005



<http://staruml.sourceforge.net/>

Other tools you could use

- Any drawing tool (e.g., Paint or Gimp)
- NetBeans' or Eclipse's UML plug-ins
- ArgoUML:
 - Pros: Platform independent and small
 - Cons: no undo feature and sometimes tedious to use
- UMLet: use for small models only

Tool demo via a Case Study

- Digital sound-recorder (Dictaphone)
- Based on:
 - *Ivan Porres Paltor, Johan Lilius:*
 - **Digital Sound Recorder: A case study on designing embedded systems using the UML notation**
- For next week: download diagrams from website (either StarUML or PDF file)



Main features

- Capacity for 10 different messages, each of max. 2 minutes
- Messages can be recorded, played back and deleted.
- Messages can be locked/unlocked to prevent unwanted deletion/overwriting .
- LCD display

Problem Statement

1. The recorder stores up to 10 messages.
2. Each message is max. 2 minutes long.
3. The user can record messages.
4. Recording of a message ends after 2 minutes or when the user stops recording.
5. Recording destroys the original message at the chosen slot.
6. Sufficient level of battery is checked before recording message.
7. Message of a given slot can be replayed.
8. Sufficient level of battery is checked before replaying message.
9. Messages can be locked/unlocked.
10. Locked messages cannot be deleted or over-written by recording to the same slot.
11. The user uses a LCD display and buttons to interact with the recorder.

Next Week

- **Topic:** Requirements Analysis exercise (*continued*)
- **Homework:**
 - Prepare questions regarding analysis phase of the project. Please note that the next exercise session is the only chance for you to clarify issues about the analysis phase.
 - Be prepared to make a presentation on your first deliverable.
 - We are going to complete the “Dictaphone” case study. Look through the “Dictaphone” requirements so we are able to proceed with the case study.

Student Projects in our group

- Research interests:
 - program verification
 - specification of object-oriented programs
 - software development tools
 - advanced type systems
 - test automation
 - abstract interpretation
- Topics:
<http://www.pm.inf.ethz.ch/education/theses>
- Contact person: peter.mueller@inf.ethz.ch

BACKUP

Problem Statement

1. The **recorder stores** up to 10 **messages**.
2. Each message is max. 2 minutes long.
3. The **user** can **record** messages.
4. Recording of a message **ends** after 2 minutes or when the user **stops** recording.
5. Recording **destroys** the original message at the chosen slot.
6. Sufficient level of **battery** is **checked** before recording message.
7. Message of a given slot can be **replayed**.
8. Sufficient level of battery is checked before replaying message.
9. Messages can be **locked/unlocked**.
10. Locked messages cannot be **deleted** or **over-written** by recording to the same slot.
11. The user uses a LCD **display** and **buttons** to **interact** with the recorder.