Instrumental Tool for Automata Based Software Development *UniMod 2*

*Kochelaev D.*, *Khasanzyanov B.*, *Yaminov B.*

*Scientific Supervisor: prof. Shalyto A.*

*SPb SU ITMO*
Overview

- Instrumental Tool *UniMod*
- Data Model
- Models' Validation
- Models' Verification
- Debugging
- Visualization
UniMod base concepts

- **UML**
- Automata based software development
- *Eclipse* platform
- Java programming language
- Open source
Programs' constituents

- Class diagrams
- State chart diagrams
- Java code
UniMod advantages

- Visual building of state chart diagrams
- Ability to interpret and compile programs
- Ability to debug programs
- Ability to validate composed models
UniMod disadvantages

- No ability to verify programs
- Non-optimal (regarding speed) validation algorithm
- Non-extendable visualization subsystem
New in *UniMod 2*

- Closer integration with *Eclipse* platform
- New validation algorithm
- Ability to verify programs
- New debugging engine
Data Model

- Designed using **EMF (Eclipse Modeling Framework)**
- Automatic creation of classes from the model
Models' Validation (1)

- Basic concepts:
  - Utilization of preliminary calculations
  - Context dependent rules checking
  - Utilization of OCL (*Object Constraint Language*) to describe rules
Models' Validation (2)

- Algorithm stages:
  1. Determination of rules those can be violated while model editing
  2. Creating set of rules to be checked on each model modification

Start model editing

Single model modification
Base concepts:
- Extensible language of Bogor verifier (extension for automata language)
- UniMod 2 as automata model interpreter
- Model is not translated into verifier's input language
Models' Verification (2)

Algorithm stages:

1. Formalization \textit{LTL} (\textit{Linear Temporal Logic}) statements to be verified

2. \textit{Bogor} runs automata model using \textit{UniMod 2} interpreter and checks for statements' violation

3. Violation trace is returned to \textit{UniMod 2}
Debugging engine has following abilities:

- Ability to add breakpoints to states and transitions
- Ability to execute program step by step
- Ability to watch value of context variables
Visualization

- Editor based on GMF (Graphical Modeling Framework)
- Ability to highlight model elements while:
  - Validation
  - Verification
  - Debugging
Redesigned and newly introduced components in instrumental tool *UniMod 2*:

- Data model representation was redesigned
- Faster validation algorithm was introduced
- Ability to verify models was introduced
- Generated editor based on *GMF*
- Unified system for visualization of actions on model was introduced