

REVaMP²

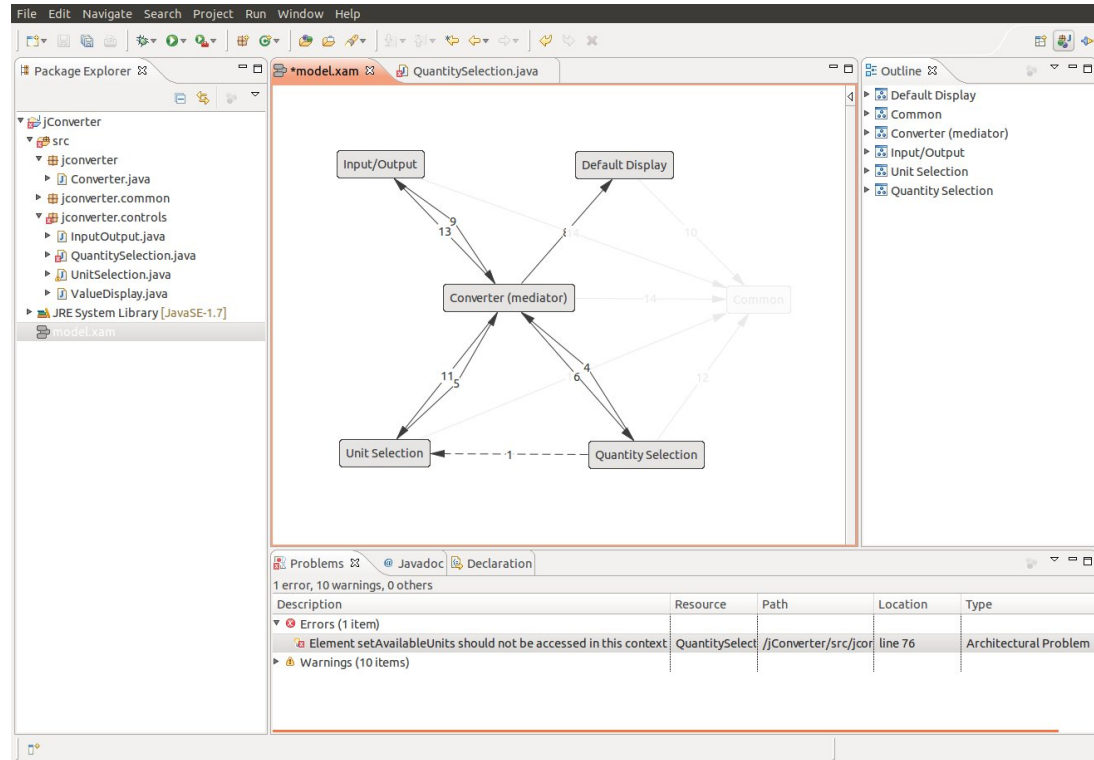
Jittac

Just-in-Time Tool for Architecture Consistency
Karlstad University

- Software Quality and Digital Modernization (SQuaD) Research Group in Department of Mathematics & Computer Science at **Karlstad University**
- Research Focus
 - design and evolution of security- and privacy-critical systems
 - quality assurance techniques, such as testing, for long-living systems
 - **detection of and mitigation against software architecture degradation and other forms of technical debt**

- JITTAC is an Eclipse based plug-in that helps check the **consistency between a software's architectural design and its implementation.**
- Original Jittac is developed as open source tool by Lero - The Irish Software Research Centre at the University of Limerick
- It **detects architectural violations** using a method based on Reflexion Modelling and prompts the developer/architect accordingly
- Gives programmers **real time feedback on architectural violations** they might introduce to the codebase, AS they introduce them into the codebase.

- This increases **programmers' architectural awareness** in a timely fashion and lessens the likelihood that architectural violations will become entrenched in the codebase over time.
 - **Java** codebase



The screenshot displays an IDE interface with a dependency graph for a Java project. The graph shows the following nodes and their relationships:

- Input/Output** and **Default Display** both depend on **Converter (mediator)** (edges 9 and 13).
- Unit Selection** and **Quantity Selection** both depend on **Converter (mediator)** (edges 11 and 5).
- Quantity Selection** depends on **Unit Selection** (edge 1, dashed).
- Converter (mediator)** depends on **Common** (edge 14).
- Common** depends on **Input/Output** (edge 10) and **Quantity Selection** (edge 12).

The 'Problems' window at the bottom shows the following error:

Description	Resource	Path	Location	Type
Element setAvailableUnits should not be accessed in this context	QuantitySelect	/Converter/src/convert.java	line 76	Architectural Problem

- Used by SQuaD
 - for software architecture degradation research
 - for module dependency analysis for single systems & software product lines
 - for feature dependency analysis for single systems & software product lines
 - to generate software dependency metrics
 - as a software codebase-to-architecture mapping tool
- For related research interests or industrial case study analysis contact
 - <https://www.kau.se/en/cs/research/research-areas/squad>

- The tool can be used for
 - software architecture degradation & consistency checking
 - module / feature dependency analysis
 - codebase-to-architecture mapping
 - software dependency metrics generation
- **Partners involved**
 - Karlstad University, Department of Mathematics & Computer Science, SQuaD Research Group
 - **Contact:** <https://www.kau.se/en/cs/research/research-areas/squad>
 - Sebastian Herold (sebastian.herold@kau.se)
 - Zipani Tom Sinkala (tom.sinkala@kau.se)

