Model Transformation goes declarative

QVT Relations in practice

Hajo Eichler
[eichler@ikv.de]
who we are

ikv++ technologies
- located in Berlin (de) and Yokohama (jp)
- developing model-driven solutions
- focus on automotive domain
- eclipse user

medini
- umbrella of our base technology
- integration and customization

medini QVT
- OMG’s QVT Relations implementation
- just another part of our tech pool
- started 1,5 years ago
medini QVT engine - facts

OMG’s QVT Relations implementation with compliance level: 

**syntax execution compliance**
- pattern matching of object templates
- multiple domains – multiple models
- relation calls via *when* and *where* clauses
- OCL queries
- key concept
- in-place transformations
- black-box through user operations
- relation import
- (no object templates)

**trace management**
- enables incremental updates on re-transformation
- enables backward transformation (depends on rule)

**direct implementation (no QVT Core mapping)**
medini QVT debugger

like the Java debugger
- step into, step over, step return and resume
- breakpoints (with conditions)
- call stack view
- variable view (changing variable bindings)
- expressions view (watch points with code assistant)

let’s have a look...
source code available under the Eclipse Public License (EPL) at

http://projects.ikv.de/qvt

we welcome your feedback...

TRAC/Subversion system for
- downloads
- help and QA
- discussion forum
- tutorial
- “how can I use the engine in my application?”
- browse & checkout source code
- ticket system (bugs & features)