Exploiting MDT OCL for QVT

E.D.Willink, A.Sánchez-Barbudo and C.W.Damus[3]
1: Thales Research and Technology (UK) Ltd
2: Open Canarias S.L.
3: IBM Rational Software

The Problem

OMG's QVT models and consequently tooling require EssentialOCL model extension.

Eclipse's MDT OCL supports OCL expression definition and evaluation over Ecore or UML2 models rather than EMOF.

Extension (to QVT) was not an MDT OCL design goal.

Options

The MDT OCL meta-model is parameterised with support for either Ecore or Eclipse's UML2 as the source of the underlying object model. Re-use of MDT OCL to support an EMOF object model therefore requires

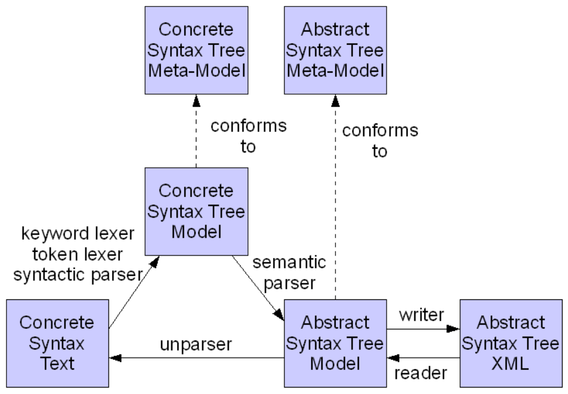
- re-parameterisation of MDT OCL using EMOF
- realising QVT with a UML2 object model to re-use the UML2 parameterisation
- realising QVT with an Ecore object model to re-use the Ecore parameterisation

Re-parameterising with EMOF was unattractive since the EMF support for EMOF is weak lacking even an `EMOF.ecore`.

Exploiting UML2 was unattractive since the UML merge of MOF packages is inconsistent with the EMOF package merge required for EssentialOCL and QVT.

The Ecore Adapters used to provide OMG compliant OCL serialisation are described in[1].

This submission describes the minor MDT OCL changes introduced in 1.2.0M3 to support Ecore-based QVT and their usage by the QVT editors available from http://www.eclipse.org/gmt/umlx/download.



Exploitation

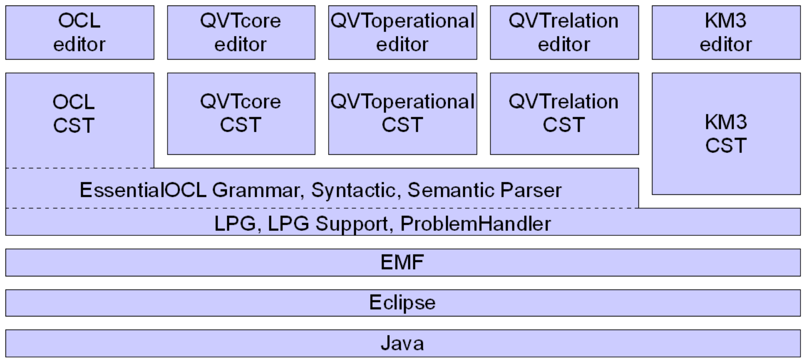
Languages such as QVT and OCL have a textual representation of a concrete syntax and an abstract syntax model typically persisted using XMI. The conversion from text to model is performed in five stages. The first three stages are performed by LPG grammars; two lexing steps to identify keywords and then tokens followed by syntactic analysis to produce the Concrete Syntax Tree. The CST is converted to the Abstract Syntax Tree in two stages, first building the tree of named AST nodes and an accompanying tree of environment nodes

[1] http://www.eclipse.org/gmt/umlx/doc/EclipseAndOMG08/EMOFAdapters.pdf

containing temporary symbol context, then the AST is completed by using the environment to resolve references to model elements. Finally the EMF writer saves the model.

The reverse conversion from XMI to concrete syntax is much simpler. The EMF reader loads the model and a tree walk emits the concrete syntax text from the AST.

The above approach has been successfully used to provide OCL, QVT and KM3 text editors. The layering of the modules for these editors is shown.



The editors are examples of tools that exploit the LPG concrete syntax tree parsing support and those aspects of MDT OCL that are relevant.

The changes that support QVT have also improved the MDT OCL functionality and modularity.

A `ProblemHandler` interface has been added to the LPG support layer to unify the previously inconsistent error and warning reports from the various parsing and validation phases. `ProblemHandler` implementations can terminate on the first error, create a log of all problems or create Eclipse Problem View markers. Usage of the `ProblemHandler` forms part of the new `BasicEnvironment` split off from the OCL `Environment`.

The `OCLParser.g` grammar has been partitioned so that `EssentialOCL.g` can be included by `OCLParser.g` or `QVTcParser.g` or `QVToParser.g` or `QVTrParser.g`.

The generated `OCLParser` class has been split so that syntactic and semantic parser stages are in distinct `OCLParser` and `OCLAnalyzer` classes. Further partitioning localizes the EssentialOCL support into an `AbstractOCLParser` for the grammar actions, and an `AbstractOCLAnalyzer` for the CST to AST conversion.

Support for an EssentialOCL-based language such as QVTx therefore involves development of

- `EqvtxCST.ecore` extending `OCLCST.ecore` to define the CST model.
- `QVTxGrammar.g` including `EssentialOCL.g` to define the grammar and provide the CST creation actions with the aid of `AbstractOCLParser`.
- `eqvtx.ecore` to define the AST model as specified by OMG.
- `QVTxAnalyzer` to convert CST to AST with the aid of `AbstractOCLAnalyzer`.
- Derived `Environment`s to support the two stage CST to AST conversion.
- An AST to CST unparser

LPG auto-generates a fully-customised `QVTxParser` that combines grammar rules and parser actions for EssentialOCL and QVTx.

Further Work

Making the OCL Standard Library an extensible model.

Revising the OCL validation to extend Ecore and support extension by QVT.

MDT OCL model and API restructuring for conformance with a revised OCL 2.x specification.