
**October 11-13, 2009**
**Amsterdam**

# On the Fly Transformation Into CA Gen

Mustafa Arikan, Arikan Productivity Group GesmbH
mustafa.arikan@arikan.at

Session Track 3
12th October 11:30 12:10

# Biography

Mustafa Arikan studied industrial engineering, mathematics and computer science in Istanbul and in Vienna and finished his education in 1986. He has meanwhile 29 years industrial experience in IT and operations research. He worked for vendors like IBM and as technology partner of Computer Associates for various large scale companies and won many IT awards throughout his career so far. His companies serve in Austria and Turkey and in cooperation with partners in over 10 countries mainly in software modernization.

# Agenda

Software Modernization.

Legacy Code

Program Transformation.

Goal of Legacy Transformation.

Transformation.

Modernization state-of-the-art

Metamodel Based Transformation

Documentation

Demo.

# Software Modernization

Legacy Transformation, or legacy modernization, refers to the rewriting or porting of a legacy system to a modern computer programming language, software libraries, protocols, or hardware platform. Sometimes referred to as software migration, legacy transformation aims to retain and extend the value of the legacy investment through migration to new platforms.

Some parts of this presentation are taken from WIKIPEDIA.

www.wikipedia.org

# Software Modernization – Legacy Code

A legacy code is any application based on older technologies and hardware, such as mainframes, that continues to provide core services to an organization. Legacy applications are frequently large and difficult to modify, and scrapping or replacing them often means re-engineering an organization's business processes as well. However, more and more applications that were written in so called modern languages like java are becoming legacy. Whereas 'legacy' languages such as Cobol are top on the list for what would be considered legacy, newer languages can be just as monolithic, hard to modify, and thus, be candidates of modernization projects

# Modernization – Program Transformation

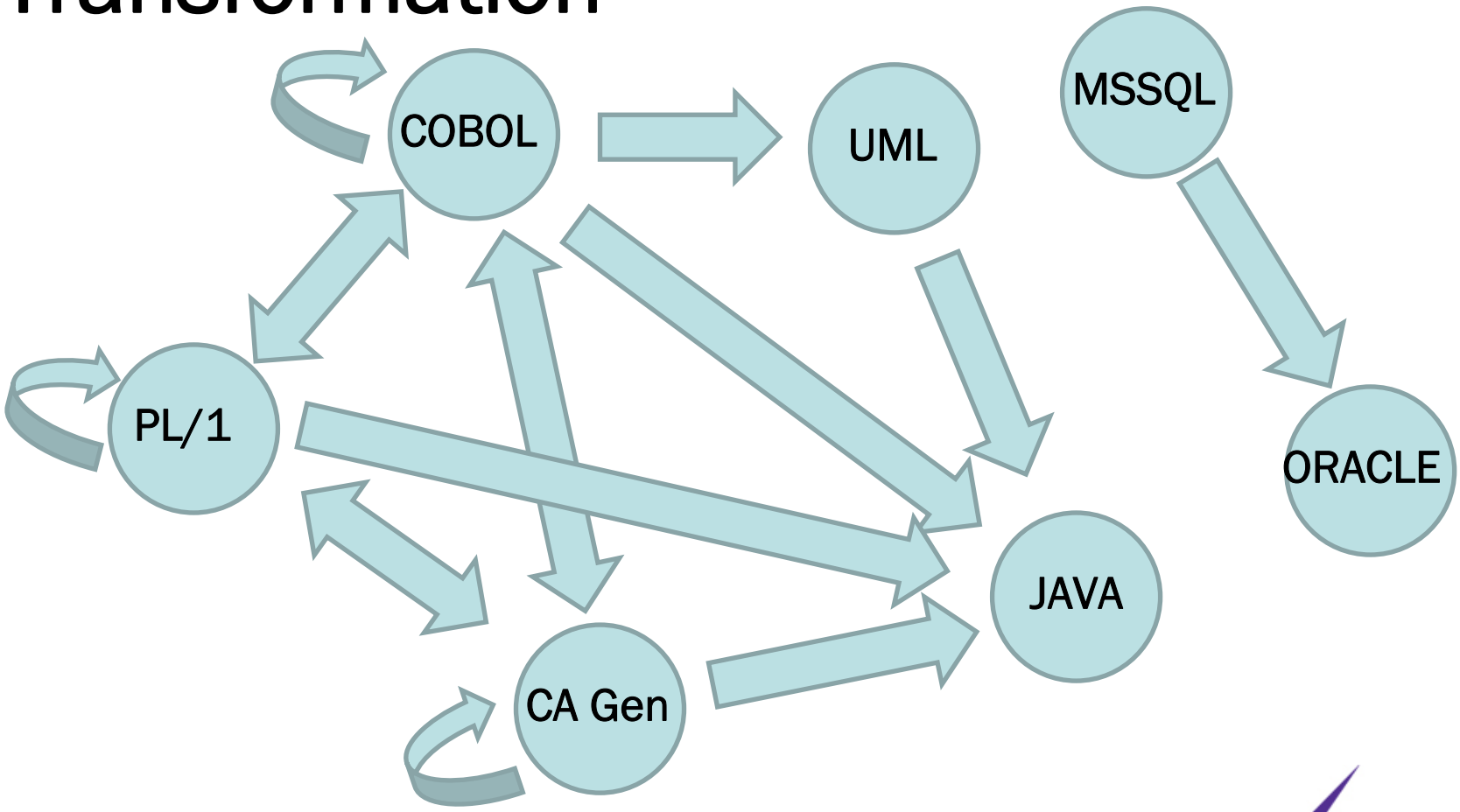
Re-implementing applications on new platforms in this way can reduce operational costs, and the additional capabilities of new technologies can provide access to functions such as web services and integrated development environments. Once transformation is complete and functional equivalence has been reached the applications can be aligned more closely to current and future business needs through the addition of new functionality to the transformed application. The recent development of new technologies such as program transformation by software modernization enterprises have made the legacy transformation process a cost-effective and accurate way to preserve legacy investments and thereby avoid the costs and business impact of migration to entirely new software.

# Goal of Legacy Transformation

The goal of legacy transformation is to retain the value of the legacy asset on the new platform. In practice this transformation can take several forms. For example, it might involve translation of the source code, or some level of re-use of existing code plus a Web-to-host capability to provide the customer access required by the business. If a rewrite is necessary, then the existing business rules can be extracted to form part of the statement of requirements for a rewrite.

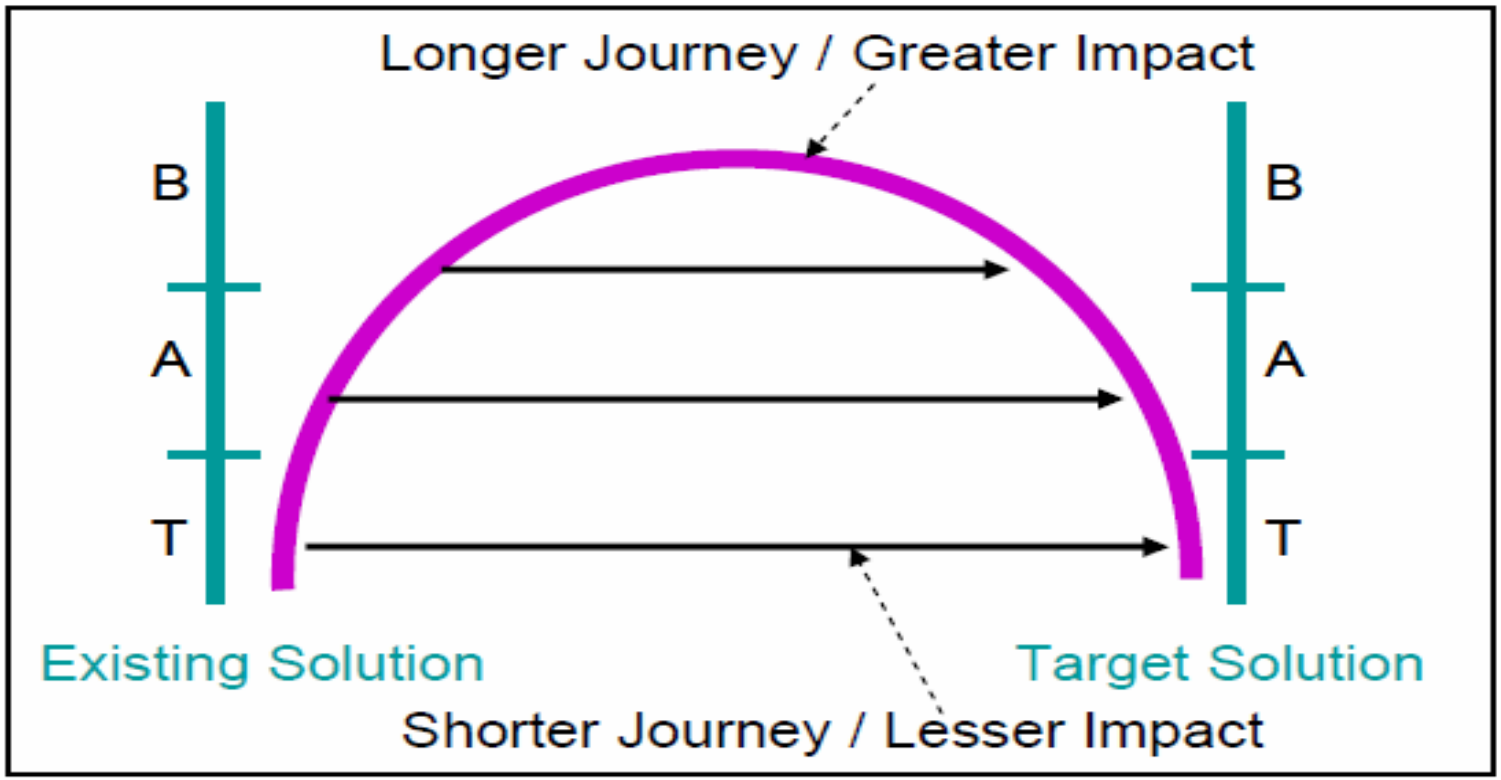
When a software migration reaches functional equivalence, the migrated application can be aligned more closely to current and future business needs through the addition of new functionality to the transformed application.

# Transformation



# Transformation

http://www.omg.org/docs/admtf/07-12-01.pdf



# Metamodel

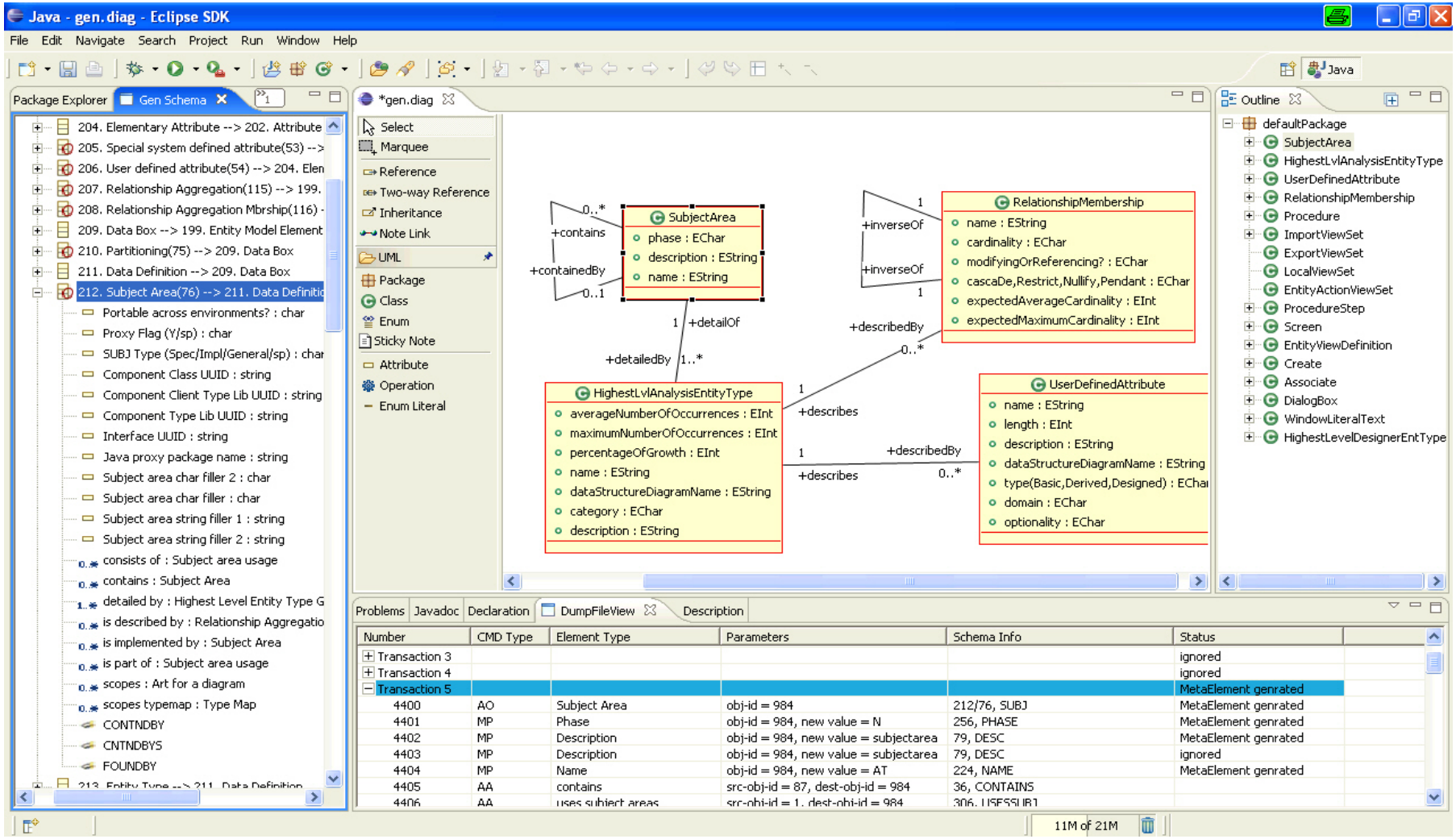
Metamodeling, or *meta-modeling* in software engineering and systems engineering among other disciplines, is the analysis, construction and development of the frames, rules, constraints, models and theories applicable and useful for modeling a predefined class of problems. As its name implies, this concept applies the notions of meta- and modeling.

# Formal Grammar

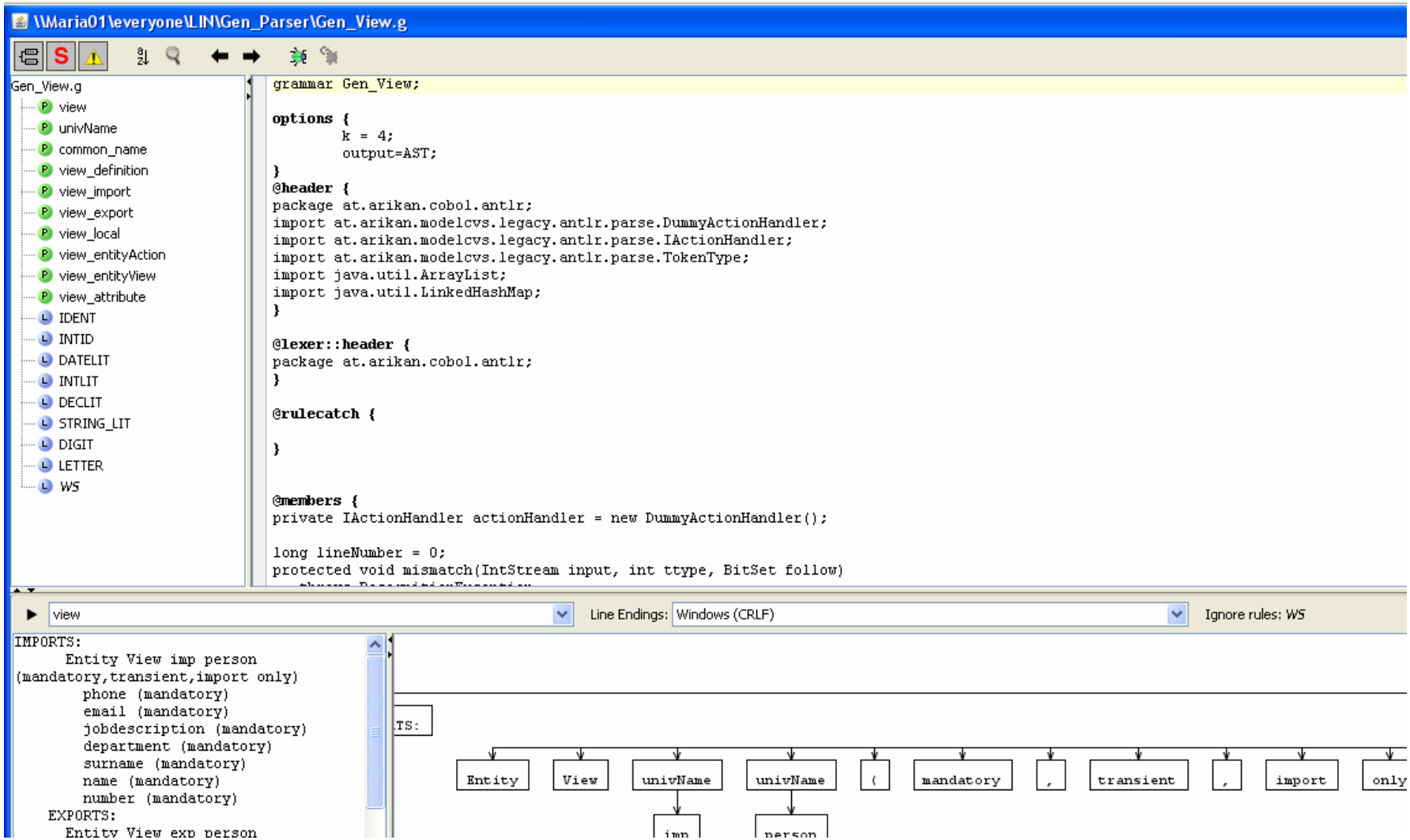
A formal language is a set of *words*, i.e. finite strings of *letters, symbols, or tokens*. The set from which these letters are taken is called the *alphabet* over which the language is defined. A formal language is often defined by means of a formal grammar (also called its formation rules); accordingly, words that belong to a formal language are sometimes called *well-formed words* (or well-formed formulas).

A formal grammar (sometimes simply called a grammar) is a set of rules for forming strings in a formal language. These rules that make up the grammar describe how to form strings from the language's alphabet that are valid according to the language's syntax. A grammar does not describe the meaning of the strings—only their location and the ways that they can be manipulated

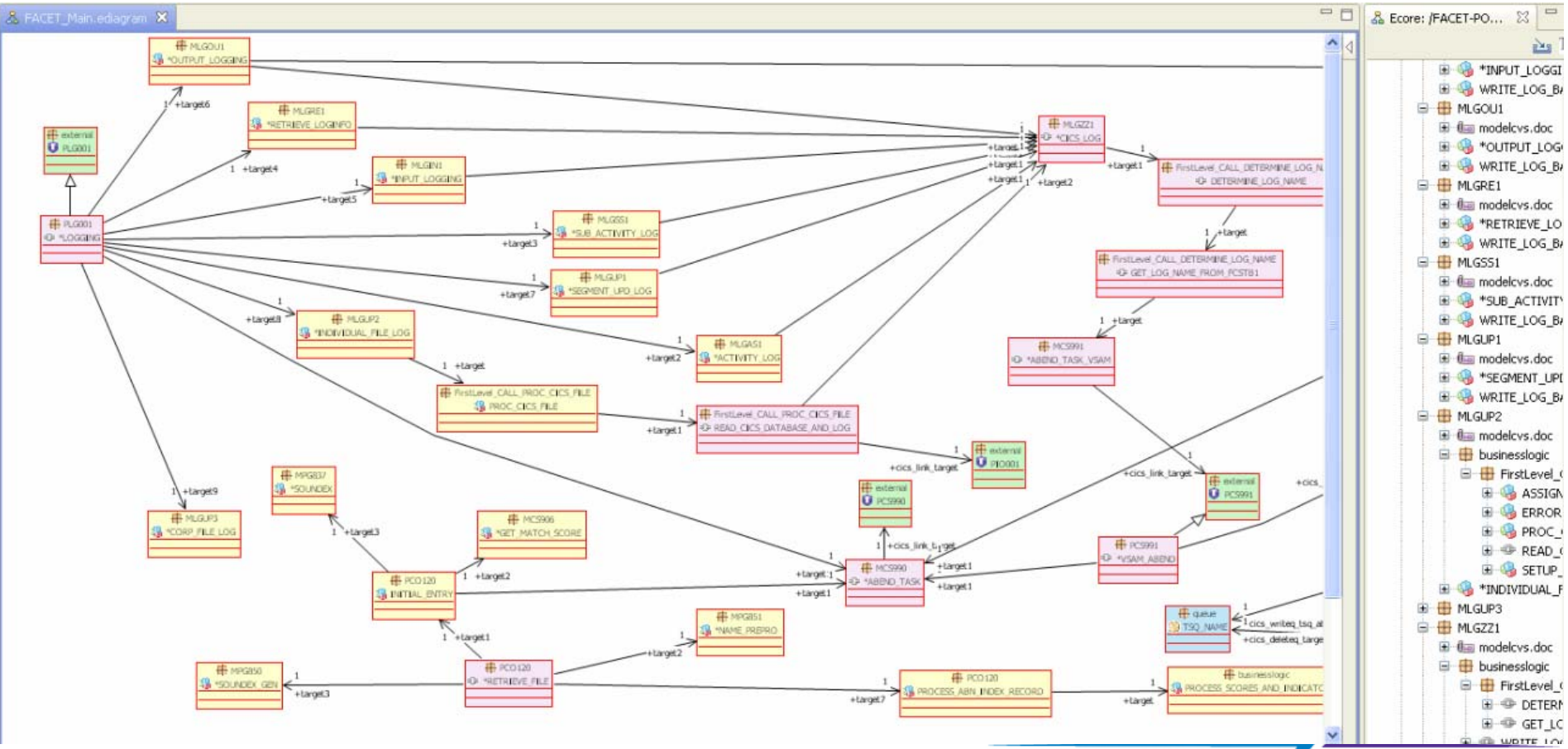
# Metamodel Creation from Tool MM



# Metamodel Generation from Grammar



# Documentation



# DEMO

# Q&A


October 11-13, 2009
Amsterdam