Drive Results.

Unleash the Modelling team

Trace, automate and collaborate using the IBM Rational CLM solution

Ferenc Kovács
Software Architect, Ericsson

IBM

RUC2014
Rational User Conference

Agenda

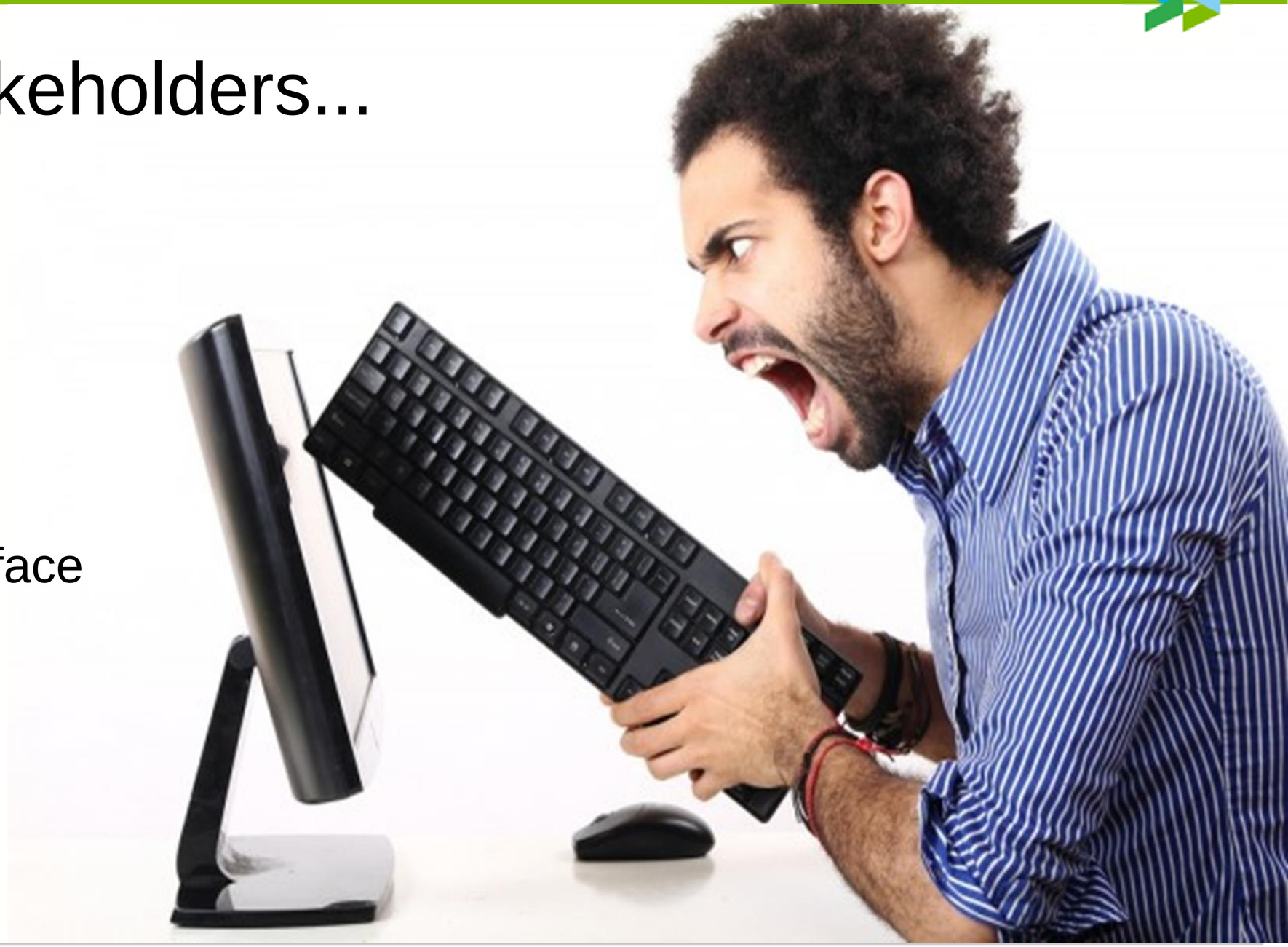
- Stakeholders and values of modeling
- Organizing our models
- Fruits of design time automation
- Sharing modeling artifacts
- Integrating and tracing
- Collaboration benefits

Stakeholder nr. one

- Wants
  - Abstraction
  - Automation
  - To spare with words
- Needs
  - Coding environment

Some other stakeholders...

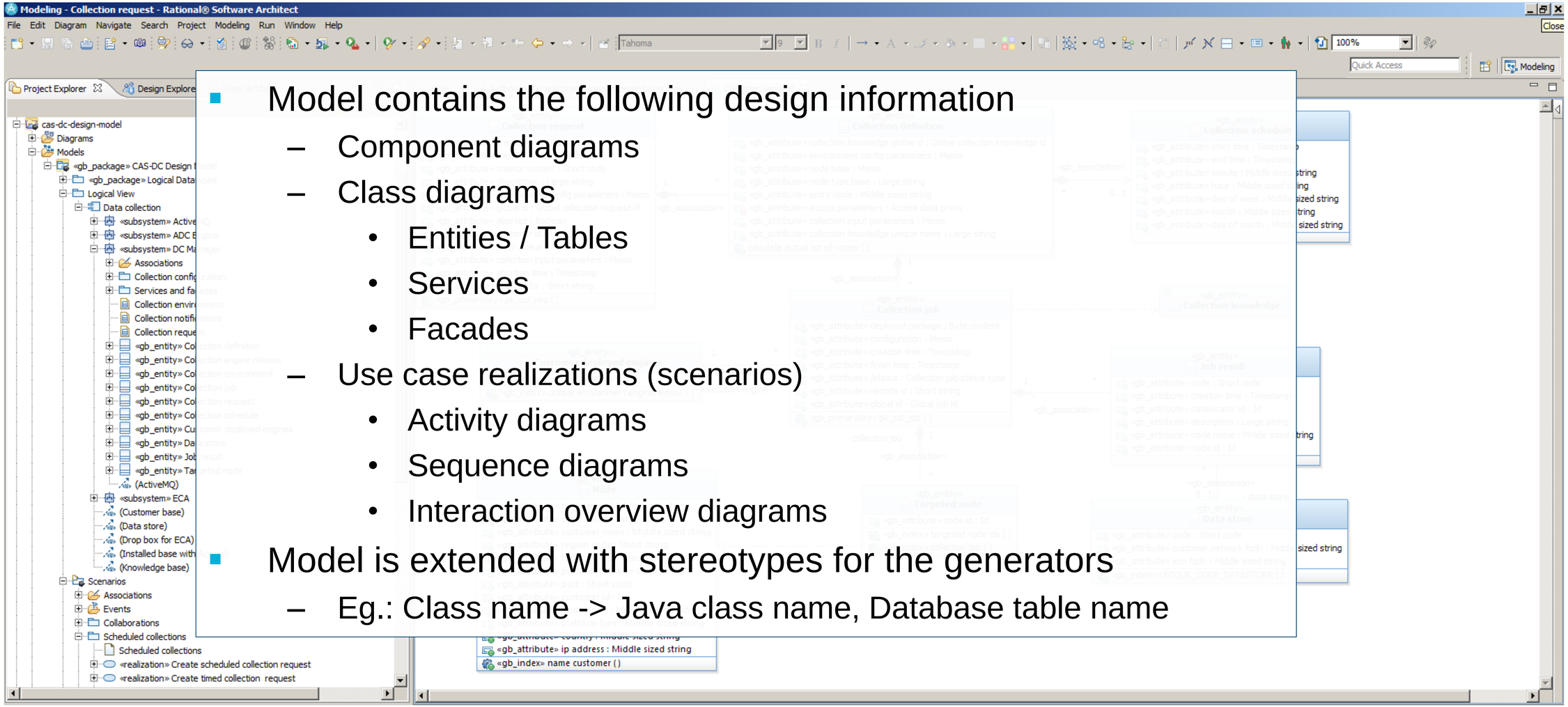
- Want
  - Documentation
  - Reports
- Need
  - Easy to use interface



Values and constraints summary

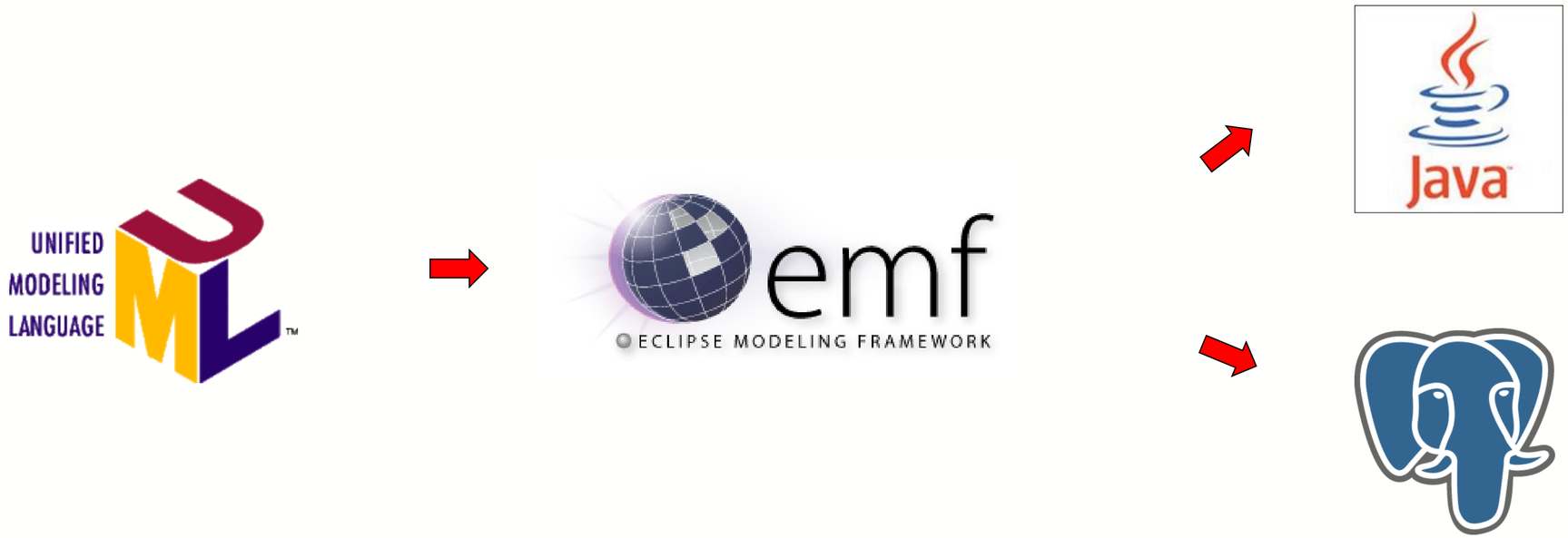
- Raising the level of abstraction
  - Less code for same functionality
  - Productivity increased
  - Number of errors decreased
- Increasing communication efficiency
  - Direct feedback
  - Visualize high level information
- Integration with project management
  - Traceability
  - Reporting
- Model & code together
  - Otherwise design will be outdated
  - Impossible without generators
  - They need the same repository (branches)
- Modeling should be as easy as coding
  - Otherwise developers simply will not use the tools
- Sharing information should be simple
  - Should not need client tool installation
  - Comment the model instead of generated docs
- Model elements should be linkable
  - To work items
  - To test cases

Structuring models in the project



- Model contains the following design information
  - Component diagrams
  - Class diagrams
    - Entities / Tables
    - Services
    - Facades
  - Use case realizations (scenarios)
    - Activity diagrams
    - Sequence diagrams
    - Interaction overview diagrams
- Model is extended with stereotypes for the generators
  - Eg.: Class name -> Java class name, Database table name

Code generation workflow



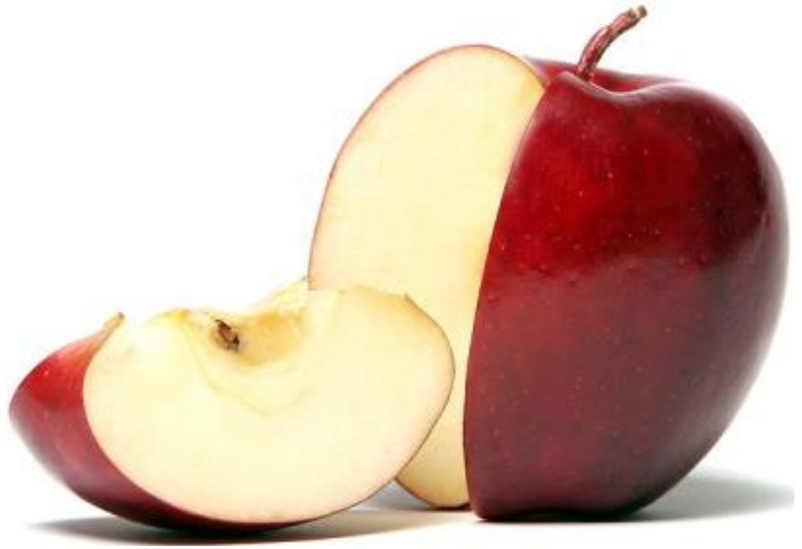
extended uml model (stereotypes)

intermediate model

service, façade, db, entity, dto

Fruits of design time automation

- 25% of java files + all ddls generated
  - adaptive to frequent changes
  - easy to make errors reduced
  - stronger architectural control
  - up to date documentation



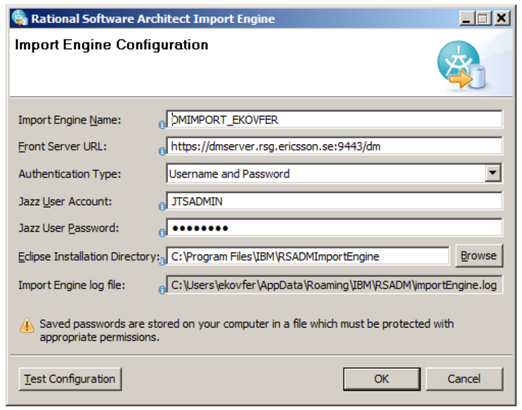
Collaboration pillar - design management pilot

- Beginning: February 3, 2014
- Objectives:
  - Connecting requirements, work items with the model
    - Work items referencing model elements
    - Model elements referencing requirements
    - Ability to query for traceability
  - Tool Support for Collaboration
    - Enable live model review and commenting between geographically separated teams
  - Interface with Rational Team Concert (RTC)
    - Model version control should remain inside RTC
- Conclusion – concept proven successfully: May 22, 2014

Sharing model artifacts

Importing models into the server repository

Local model files
(CVS, SVN, RTC, …)

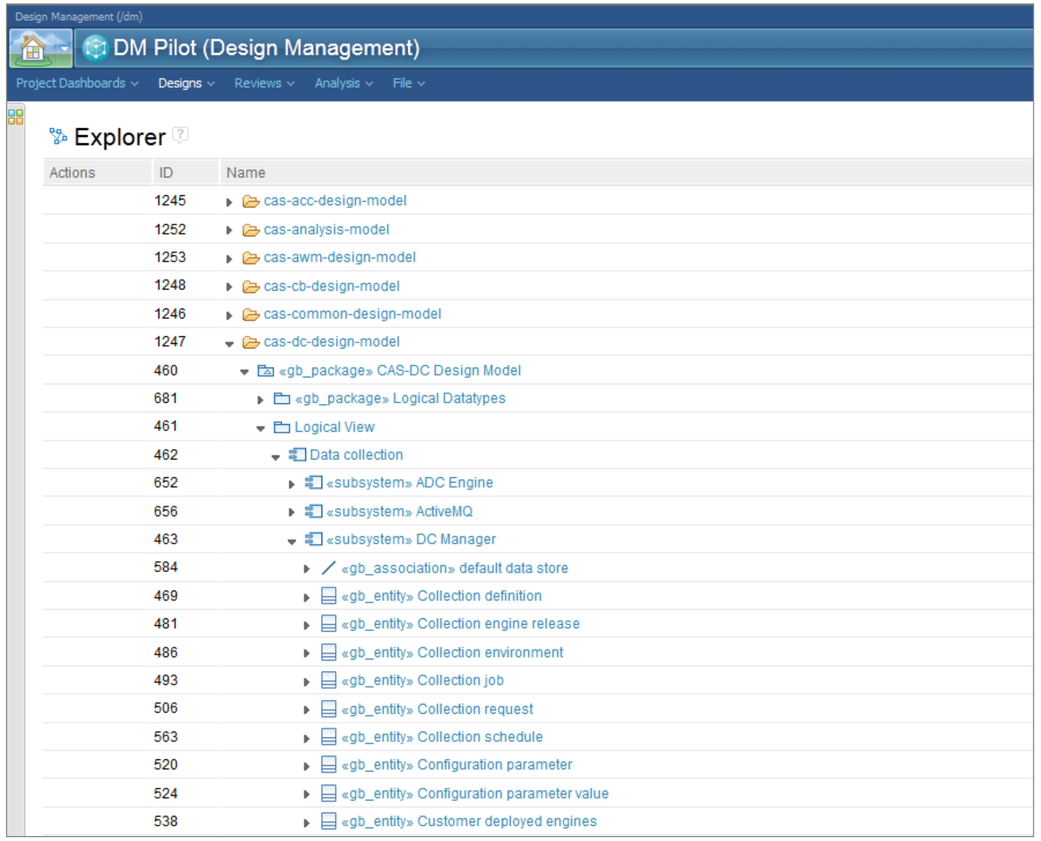


Import engine

Client machine

reads

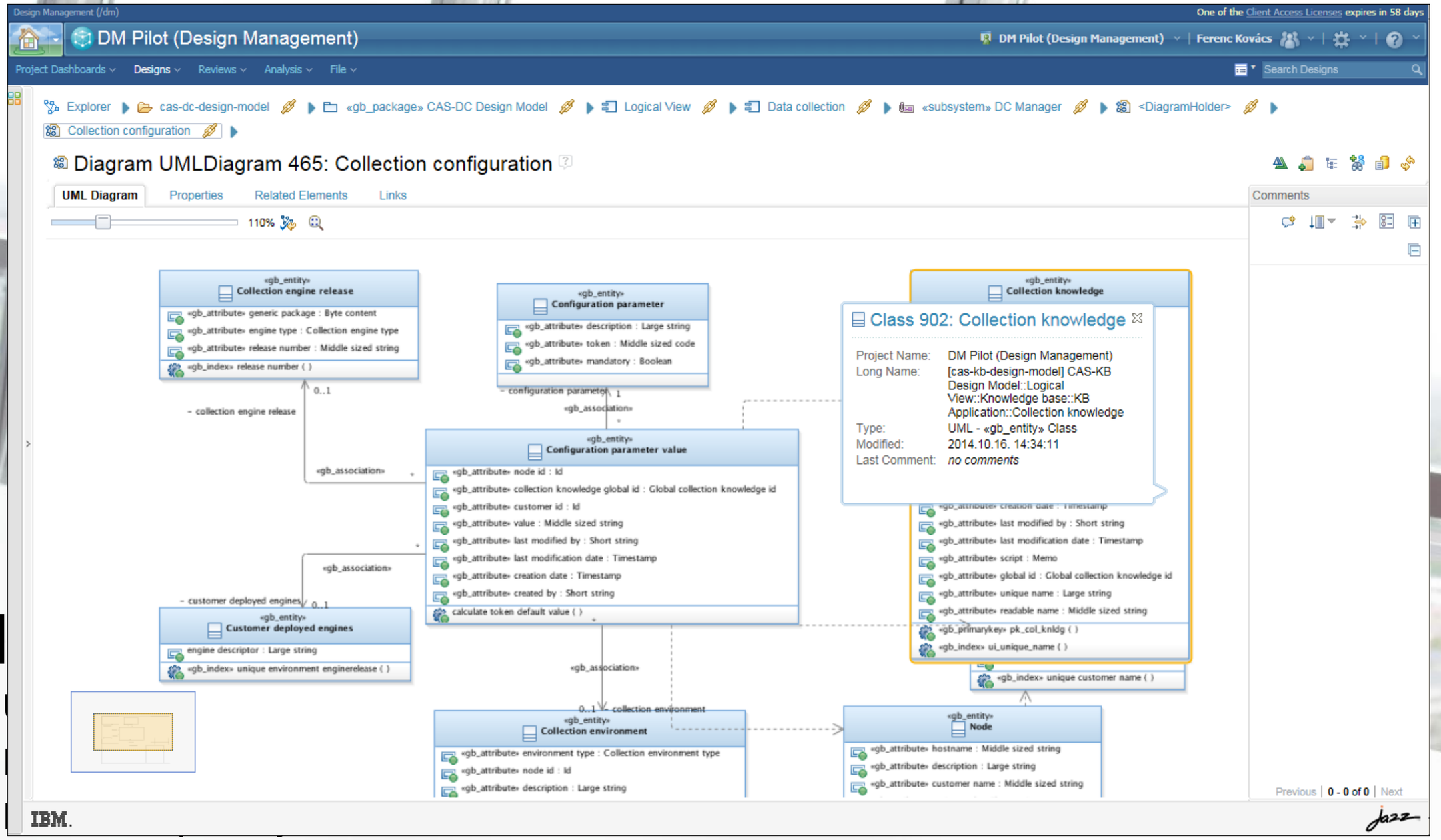
imports



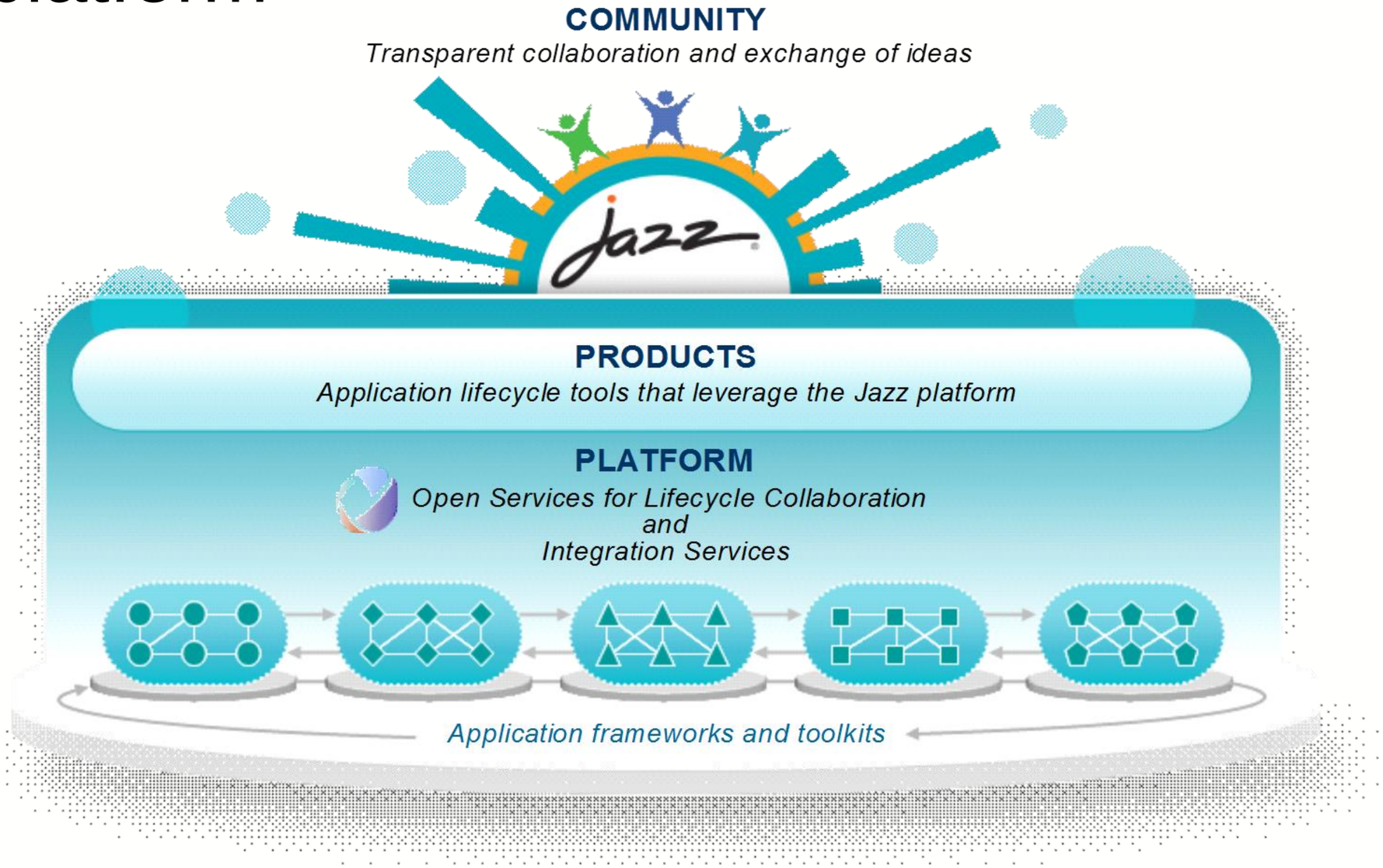
Design manager Server

Coll

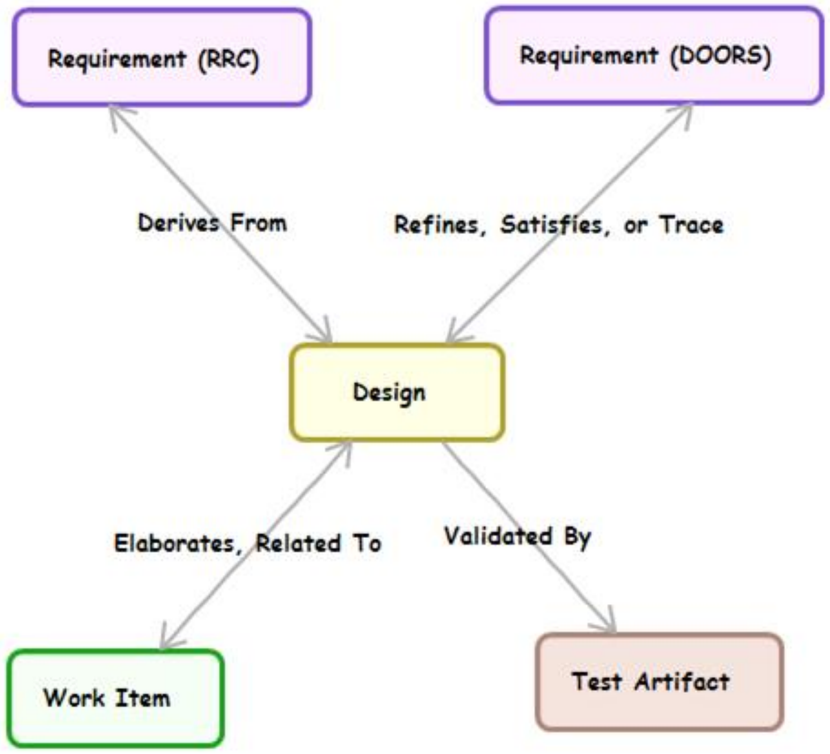
-
-
-


Design Management (/dm)
One of the Client Access Licenses expires in 58 days
DM Pilot (Design Management)
DM Pilot (Design Management)
Ferenc Kovács
Project Dashboards
Designs
Reviews
Analysis
File
Search Designs
Explorer
cas-dc-design-model
«gb_package» CAS-DC Design Model
Logical View
Data collection
«subsystem» DC Manager
<DiagramHolder>
Collection configuration
Diagram UMLDiagram 465: Collection configuration
UML Diagram
Properties
Related Elements
Links
110%
Comments
«gb_entity»
Collection engine release
«gb_attribute» generic package : Byte content
«gb_attribute» engine type : Collection engine type
«gb_attribute» release number : Middle sized string
«gb_index» release number ( )
«gb_entity»
Configuration parameter
«gb_attribute» description : Large string
«gb_attribute» token : Middle sized code
«gb_attribute» mandatory : Boolean
«gb_entity»
Collection knowledge
Class 902: Collection knowledge
Project Name: DM Pilot (Design Management)
Long Name: [cas-kb-design-model] CAS-KB Design Model::Logical View::Knowledge base::KB Application::Collection knowledge
Type: UML - «gb_entity» Class
Modified: 2014.10.16. 14:34:11
Last Comment: no comments
«gb_entity»
Configuration parameter value
«gb_attribute» node id : Id
«gb_attribute» collection knowledge global id : Global collection knowledge id
«gb_attribute» customer id : Id
«gb_attribute» value : Middle sized string
«gb_attribute» last modified by : Short string
«gb_attribute» last modification date : Timestamp
«gb_attribute» creation date : Timestamp
«gb_attribute» created by : Short string
calculate token default value ( )
«gb_attribute» last modified by : Short string
«gb_attribute» last modification date : Timestamp
«gb_attribute» script : Memo
«gb_attribute» global id : Global collection knowledge id
«gb_attribute» unique name : Large string
«gb_attribute» readable name : Middle sized string
«gb_primarykey» pk_col_knldg ( )
«gb_index» ui_unique_name ( )
«gb_index» unique customer name ( )
«gb_entity»
Customer deployed engines
engine descriptor : Large string
«gb_index» unique environment enginerelease ( )
«gb_entity»
Collection environment
«gb_attribute» environment type : Collection environment type
«gb_attribute» node id : Id
«gb_attribute» description : Large string
«gb_entity»
Node
«gb_attribute» hostname : Middle sized string
«gb_attribute» description : Large string
«gb_attribute» customer name : Middle sized string
«gb_association»
0..1
1
Previous | 0 - 0 of 0 | Next
IBM.
jazz

Jazz platform



Unique traceability



- Design and Requirements
  - Analysis: link designs that provide the next level of details for requirements
  - Coverage: link designs that implement requirements to ensure coverage
- Design and Change Management (Work Items)
  - Planning: link design tasks to related designs
  - Design Changes: link design change sets to related work items
  - Implementation: link implementation tasks to related designs making it easier for developers to find design
  - Defects: link design defects back to defective design
- Designs and Test Artifacts (one-way only)
  - Test coverage of designs: Link from designs to test artifacts that validate the design to ensure test coverage

Conclusion/Summary

- Even though “modeling” doesn’t really sound cool today, we posses both the knowledge and the tools to create an environment, which delivers the benefits of design automation and collaboration for the whole project team, by providing flexibility, boosting collaboration and thus letting us focus on our real tasks by saving time and energy.

Future steps

- Getting the DM Server hosted in Aachen ;-)
- Introducing build-time model generation + validation
- Textual based modeling
- Instant refactoring

Acknowledgements

- Thanks to the system and quality architects of the CAS project team for supporting me in setting up and piloting the solution
- Thanks to Ibm ECAT for providing substantial assistance during and after the pilot.
- Special thanks to Roland Revsater for his patience and support over these slides.
- Thanks to Ericsson DU Hungary management for supporting my work related to the modeling environment

References

- Marian Petre: UML in practice
- Jon Whittle, John Hutchinson, Mark Rouncefield: The State of Practice in Model-Driven Engineering
- https://jazz.net/help-dev/dm/index.jsp
- http://pic.dhe.ibm.com/infocenter/rsahelp/v9/index.jsp
- http://www.eclipse.org/modeling/emf/