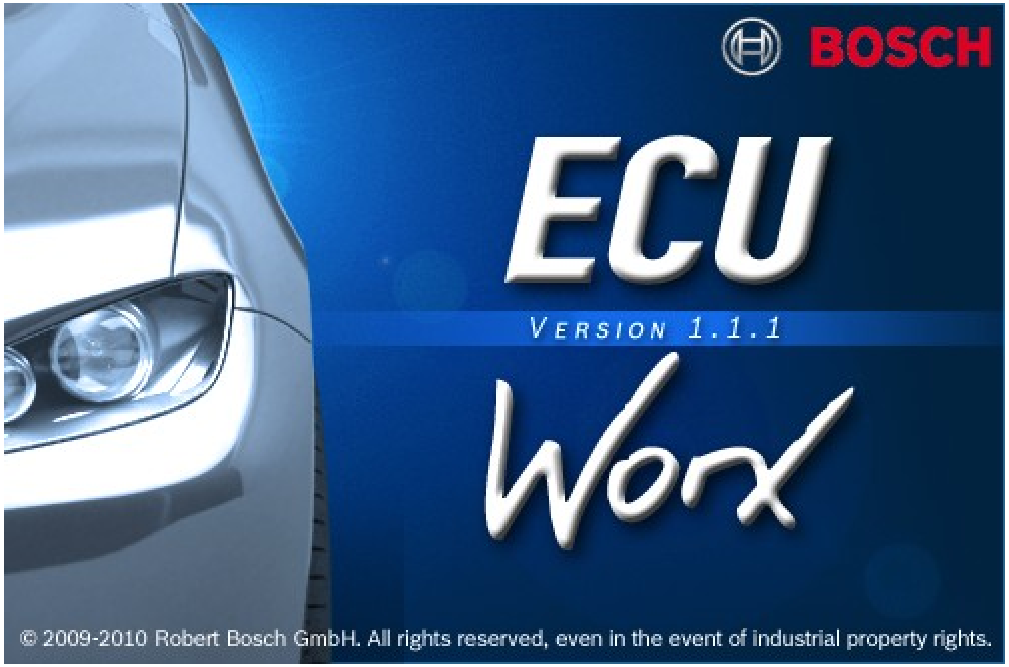
# Models in an embedded automotive IDE


BOSCH
ECU
VERSION 1.1.1
Worx
© 2009-2010 Robert Bosch GmbH. All rights reserved, even in the event of industrial property rights.

Dr. Lars Geyer-Blaumeiser
Robert Bosch GmbH
Cross Divisional Tool Development Department
lars.geyer-blaumeiser@de.bosch.com

Eclipse Enterprise Modeling Day
28.10.2010
Zürich

## Bosch Cross Divisional Tool Development

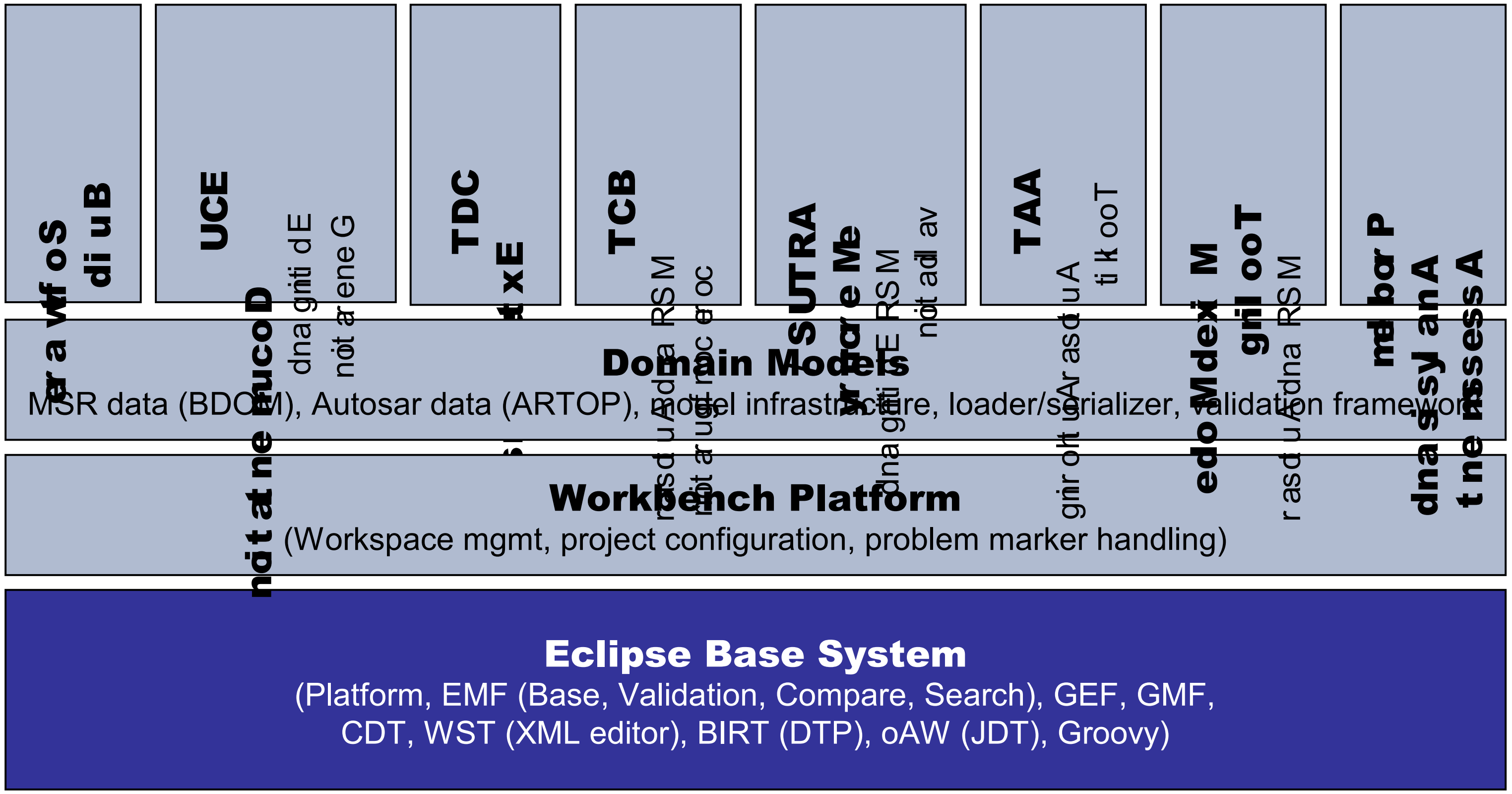
- Goal: Provide common development methods and tools for ECU software development in automotive business units
- Support of several product lines (Engine control, traction control, body computers, …)
  - Spread throughout the world
- Eclipse based systems in place for several years for single business units
  - Large data models in use from the beginning
  - Past: Command line centered approach with Eclipse RCPs as tools for special purposes
  - Future: Eclipse based IDE including most functionalities

## Technologies

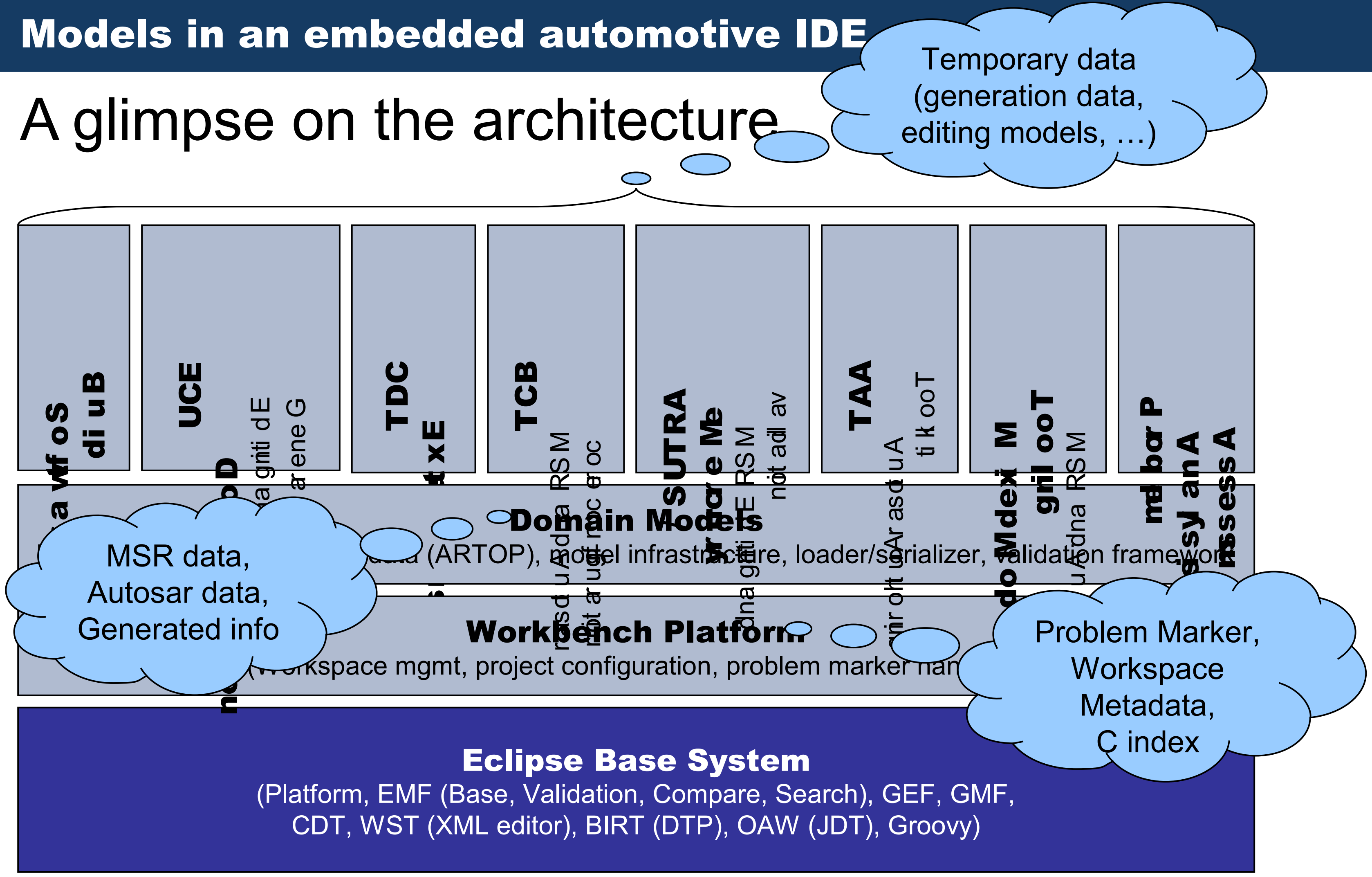
- ➔ Application Software is modeled with Tools like Ascet or Simulink
  - C code is generated
- ➔ Metadata (interfaces, architecture, system composition, data types, calibration parameters, documentation) in MSR or Autosar
- ➔ Core Software in C which is generated to a large degree
  - According to hardware and application requirements
  - Perl or oAW as generation techniques
  - Configuration done in MSR or Autosar
- ➔ Standard mechanisms for software build (Make, Scons, …)
- ➔ Special tool chains for documentation generation
- ➔ Projects requirements up to
  - Before compilation: 400MB (only hex relevant data)
  - Compiled: 2GB (generated code, object files, temporary files, …)

## Models in an embedded automotive IDE

# A glimpse on the architecture


Software Build
UCE Documentation Editing and Generation
TDC Ext
TCB MSR and Autosar core configuration
ARTUS Me MSR EE Editing and validation
AAT Autosar Authoring Toolkit
Mixed Mode Tooling MSR and Autosar
Problem Analysis and Assessment
Domain Models
MSR data (BDOM), Autosar data (ARTOP), model infrastructure, loader/serializer, validation framework
Workbench Platform
(Workspace mgmt, project configuration, problem marker handling)
Eclipse Base System
(Platform, EMF (Base, Validation, Compare, Search), GEF, GMF, CDT, WST (XML editor), BIRT (DTP), oAW (JDT), Groovy)

## A glimpse on the architecture


Temporary data (generation data, editing models, …)
UCE
TDC
TCB
TAA
Domain Models
data (ARTOP), model infrastructure, loader/serializer, validation framework
MSR data, Autosar data, Generated info
Workbench Platform
(Workspace mgmt, project configuration, problem marker han
Problem Marker, Workspace Metadata, C index
Eclipse Base System
(Platform, EMF (Base, Validation, Compare, Search), GEF, GMF, CDT, WST (XML editor), BIRT (DTP), OAW (JDT), Groovy)

BOSCH

## Use of models in the IDE

- Fast editing and validation
  - Editing is model based
  - Validation of editing within editor
  - Efficient searching
  - Coarse grained validations on the whole model
  - Support of text editors (code completion, hyperlinking) from model

- Processing is model driven
  - Software build uses model data for code generation
  - Documentation build generates documentation items out of the model

- Achieved goal: Improve the performance and consistency of development items (front loading)

## Challenges (from a data structure perspective)

- ➔ Getting the right model
  - When to use EMF?
  - XML is a difficult source for EMF models
  - Pure data vs. transformed data
  - Performance  vs. memory consumption
  - Different views, e.g., processing vs. editing
- ➔ Getting the infrastructure right
  - Loading/Serializing/Persistence
  - Synchronization
  - Performance/User Feedback
- ➔ The java.lang.String issue
- ➔ Memory consumption
  - A 32bit virtual machine has 1,5 GB memory, period

## Conclusion

- ➔ We are convinced from the idea of a model based IDE
  - Use cases and user support require fast data access

- ➔ Challenges are demanding ➢ No easy solutions
  - We test the boundaries of scalability

- ➔ User experience is at stake
  - Long waiting times
  - Use cases cannot be realized due to lack of resources
  - Parallel work on one machine (Build, IDE, Outlook …)

- ➔ Memory consumption is the biggest issue
  - Especially with the constraints of the Java VM
  - Intelligent load/unload mechanisms with memory as cache?