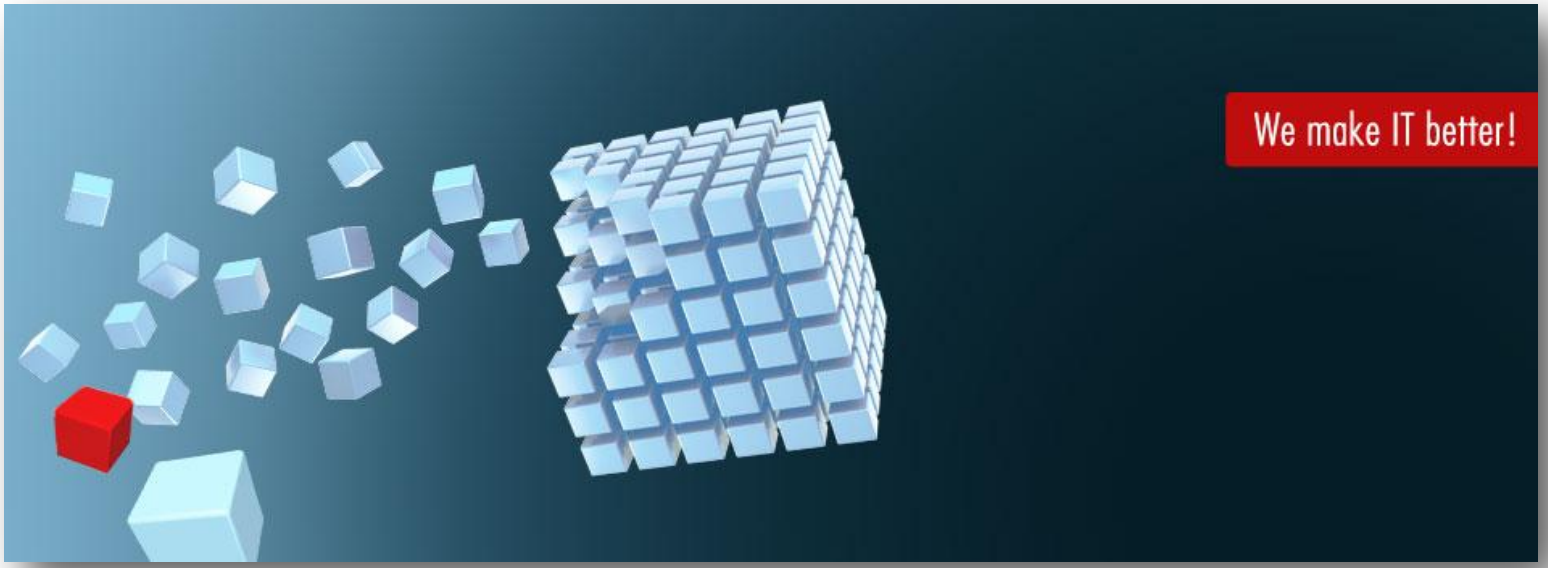


JAX 2013 – Modeling Day

Service Repository for Model-Driven SOA

Design and Implementation Aspects

Thomas Stahl, b+m Informatik AG / engineering dept.

Stefan Zeug, b+m Informatik AG / engineering dept.

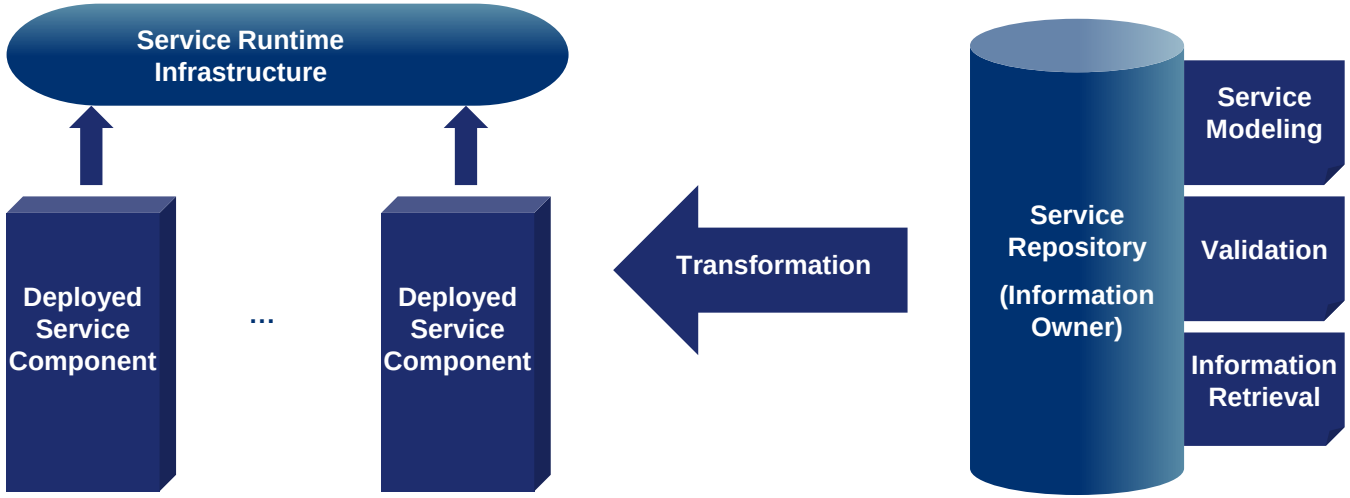
b+m business IT management
ALLGEIER GROUP

Agenda

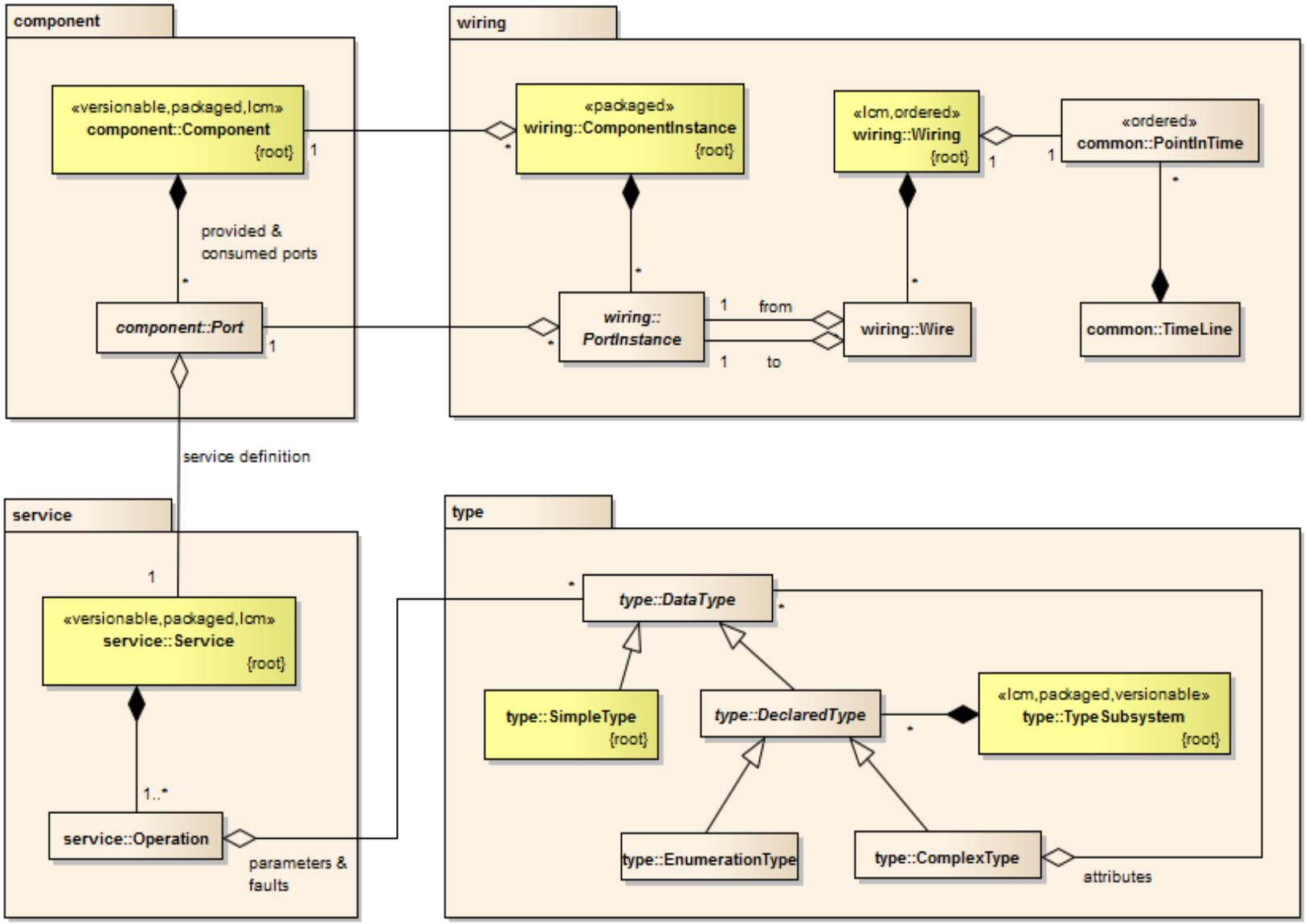
- **Context**
- **Architectural Decisions and Design Principles**
- **Specific Aspects**
  - Discussion of Design- and Implementation-Perspective
- **Prospect**

Context

- **SOA is an almost comprehended domain in the sense of DSL-Engineering and Model-Driven Software Development**
- **The tools of the Eclipse Modeling Project provide a good basis for the technical infrastructure of a SOA – especially a Service Repository**
- **This talk presents a (prototypically validated) target vision which was developed in the project MAIA of the Swiss Mobiliar (see last track)**

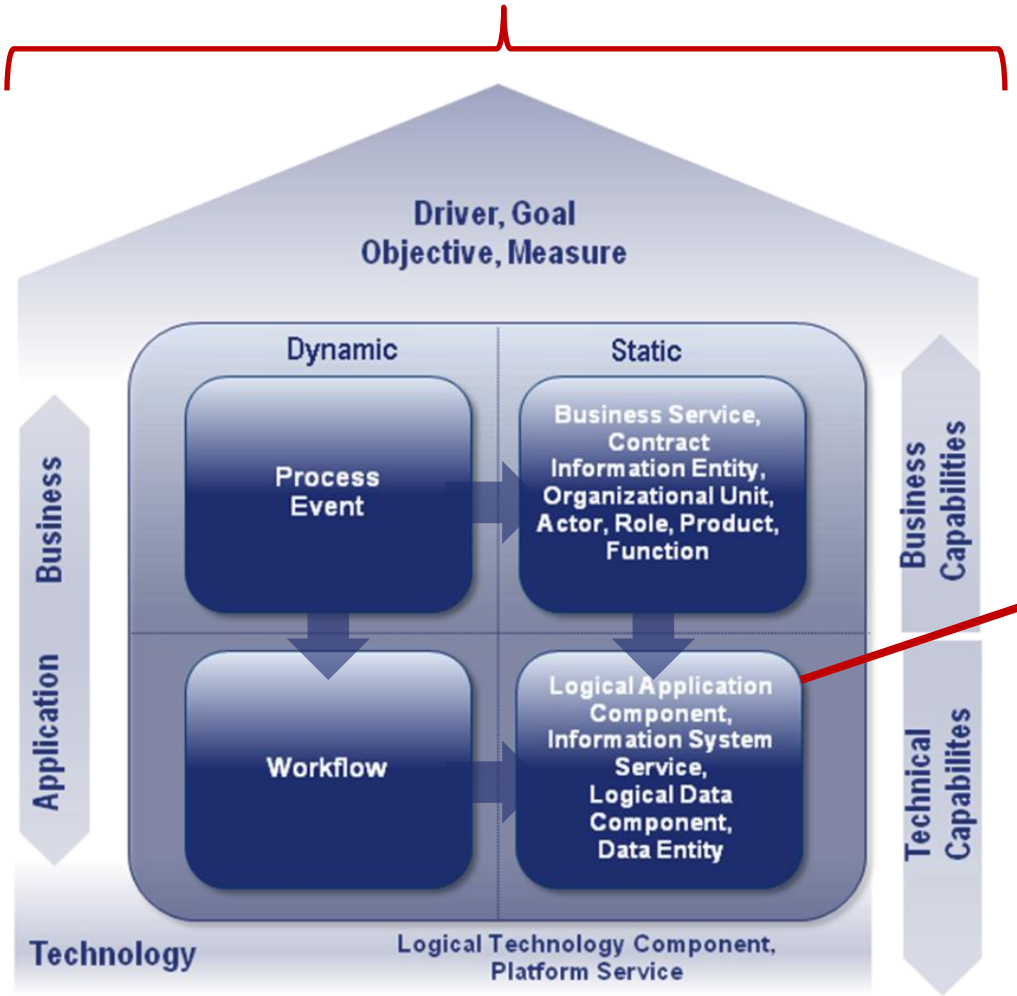


Recap: Status Quo Scope of Technical SOA Domain Model from the MAIA Project



Scope / Target Vision

Wide scope (Vision):
Enterprise-Repository - compatible with TOGAF content metamodel



Narrow scope:
Service-Repository - managing technical services and schemas to enable SOA micro governance

Some Usage Scenarios

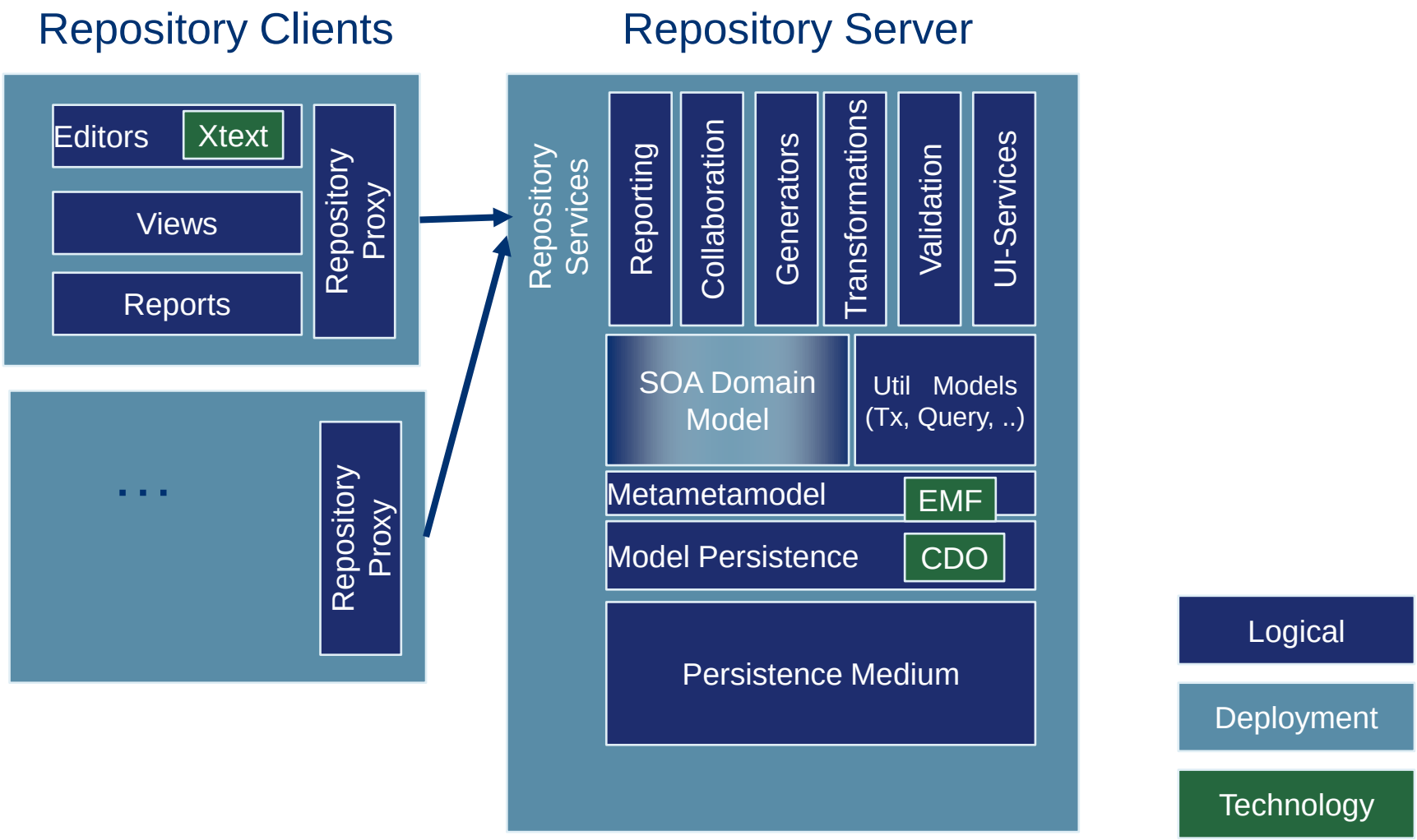
b+m business IT management
ALLGEIER GROUP

- **(Service-) Reuse**
  Contract publishing, searching and finding, reporting, navigation to associated elements and artifacts relevant for software development (like generated service stubs)
- **(Service-) Evolution**
  Life cycle management, technical versioning (snapshots), quality gates (constraint sets)
- **Dependency Management**
  Impact analysis
- **SOA Governance rules and processes**
  Design time policies, model-diffs, workflow support for life cycle management
- **Deployment Automation (Forward Engineering)**
  Consistency between deployed (service-)versions
  Consistency between (service-)model and deployed artifacts
- **Support for Business IT Alignment**
  Manage dependencies between technical services, business services, business processes and technical workflows

Architectural Decisions and Design Principles

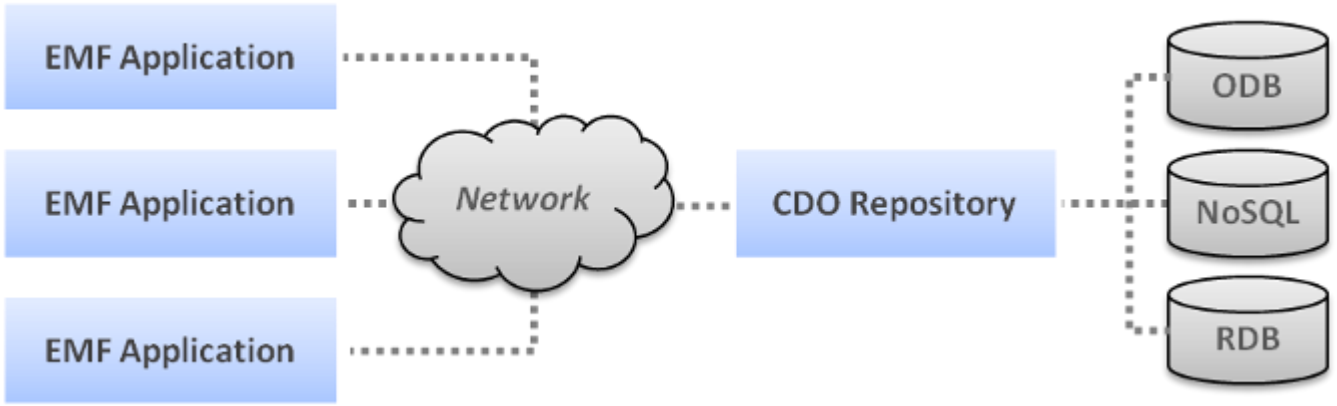
- **MDD is an established paradigm in the SOA context**
- **Start Small: Evolution of the Service Domain Model must be supported – it's a model on its own**
- **We see a SOA-Repository as a special case of a generic Model-Repository. The typical usage scenarios of a SOA-Repository can thus be mapped to**
  - Collaborative editing of (service) models
  - Persistence of (service) models
  - (Service) model diffs & validation
  - (Service) model transformations and integration of generators
  - Navigation, views, queries, reports for (service) models
- **We build upon the technologies of the Eclipse Modeling Project**
  - EMF/Ecore as metameta model
  - CDO as persistence layer
  - Xtext as DSL infrastructure

Architectural Blueprint



Model Persistence

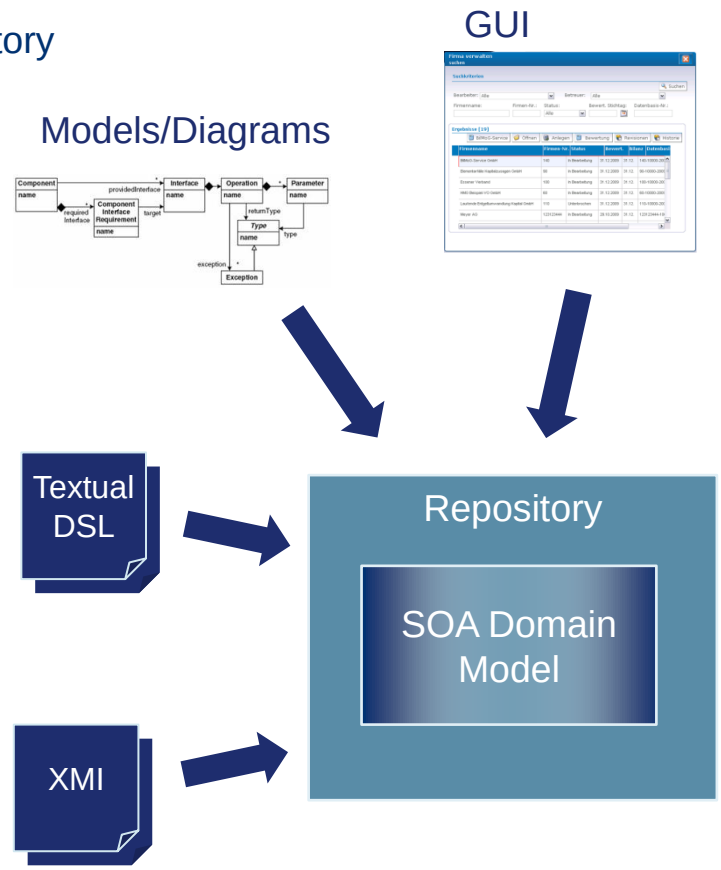
- Data ownership is located by the repository (master)
- Separation of DSL-Metamodel and Domain-Model
- Replaceable persistence layer
- CDO integration, database model, CDO session
- Prototyping results



CDO-Architecture

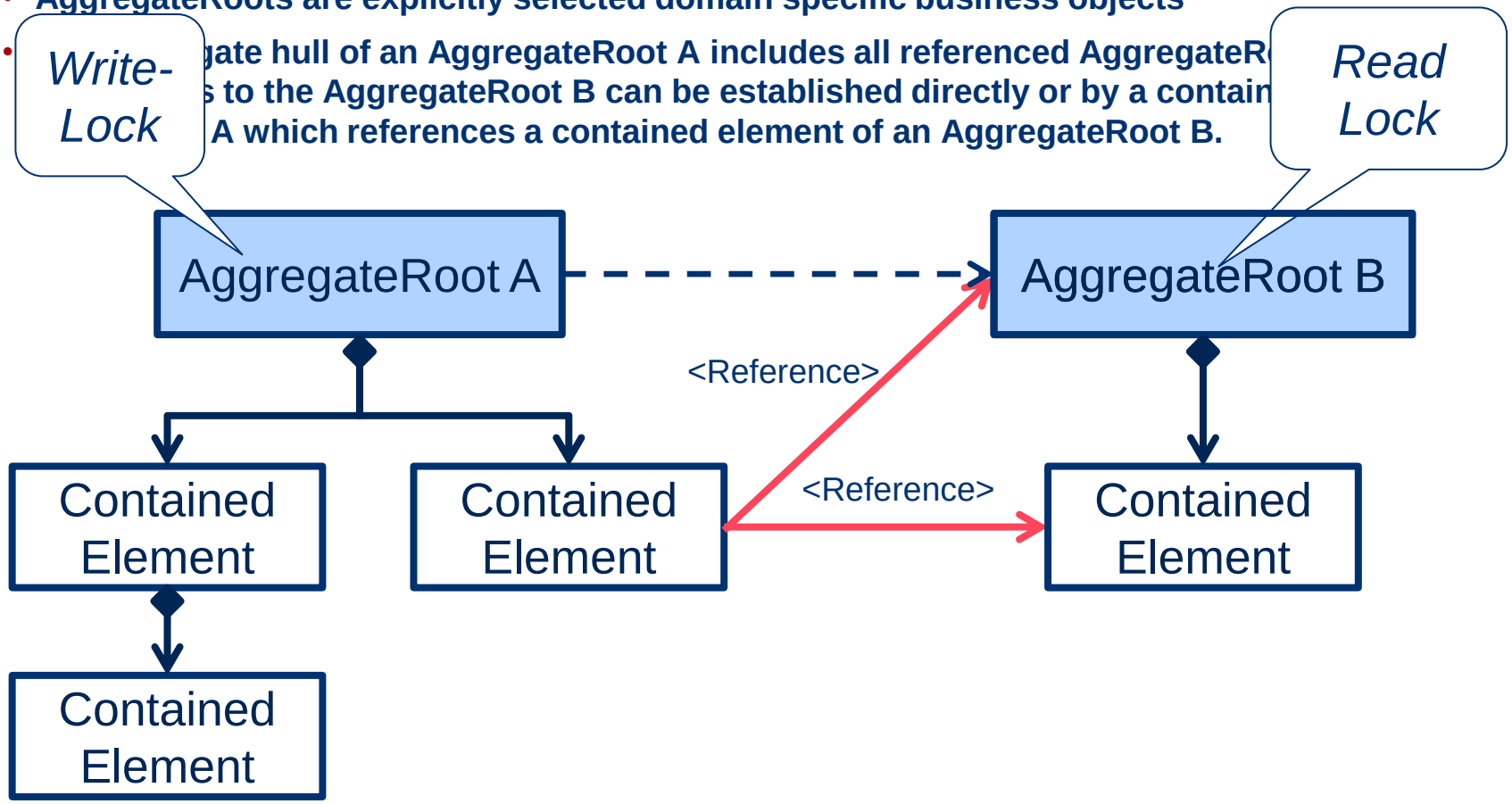
Collaboration and Editing

- No essential changes to the existing edit use cases (textual SOA-DSL of project MAIA)
- Collaboration
  - Transaction/locking model within the repository
  - Logical transactions & locking
  - Check in / check out paradigm
  - Locking granularity
  - Conflict detection and handling
- Editing
  - Textual DSL representation as snapshots
  - Fan in principle (channels)



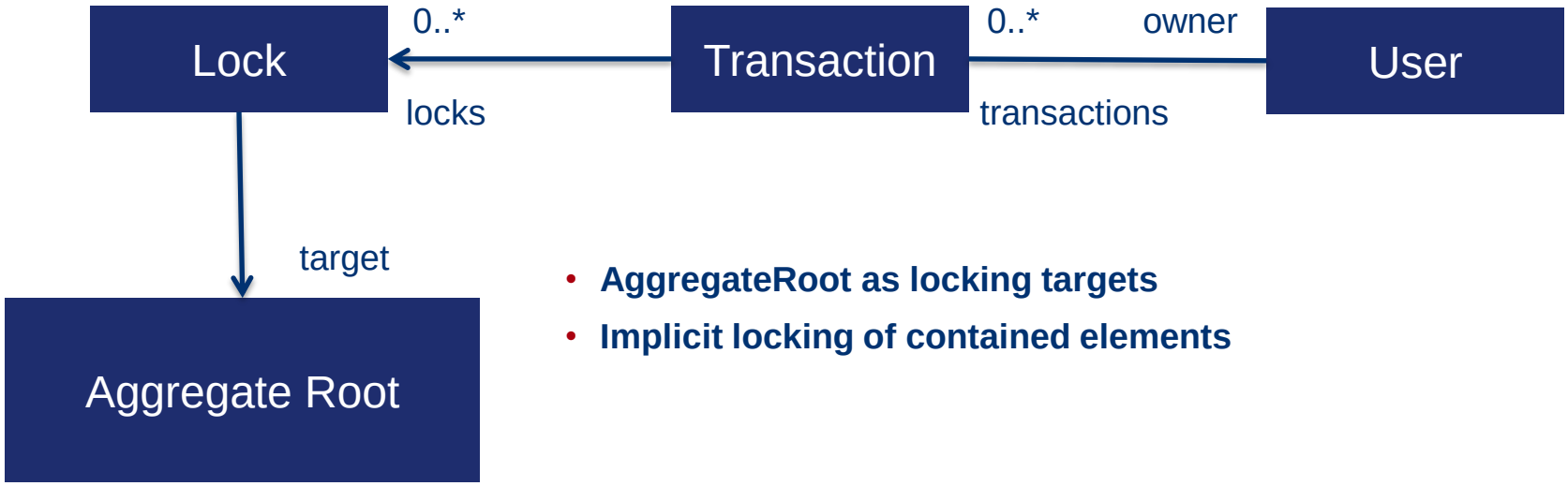
Collaboration and Editing – „Aggregate Hull“

- **AggregateRoots are explicitly selected domain specific business objects**
- **gate hull of an AggregateRoot A includes all referenced AggregateR**
  s to the AggregateRoot B can be established directly or by a contain
  A which references a contained element of an AggregateRoot B.



Collaboration/Transaction Metamodel

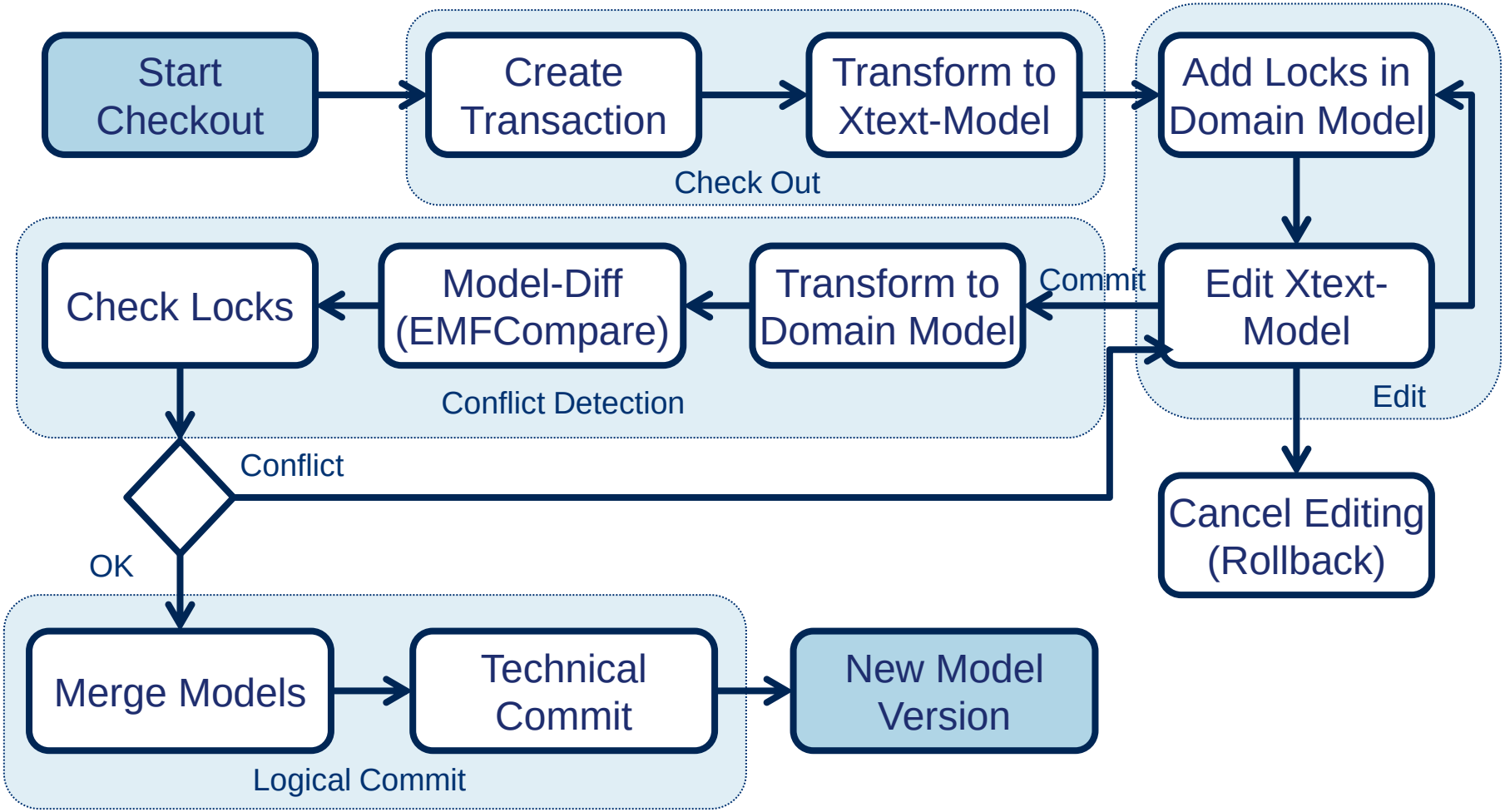
- **Main elements of a simple metamodel for collaboration/transaction**
  - Transaction with status (open/closed)
  - Lock with type (write / shared-read)
    - Only pessimistic locking
  - User management



- **AggregateRoot as locking targets**
- **Implicit locking of contained elements**

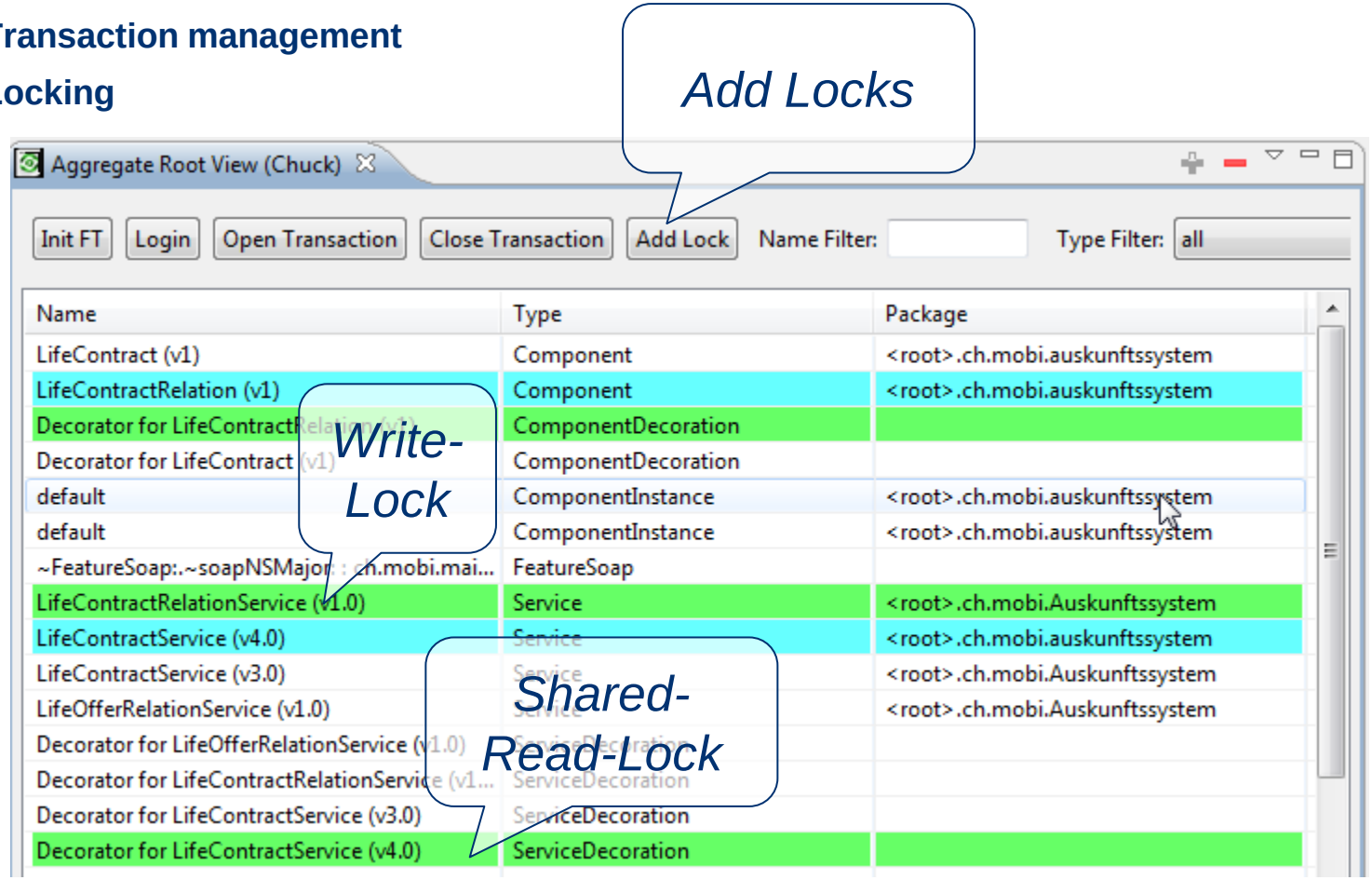
Checkout-Edit-Commit

Edit model:



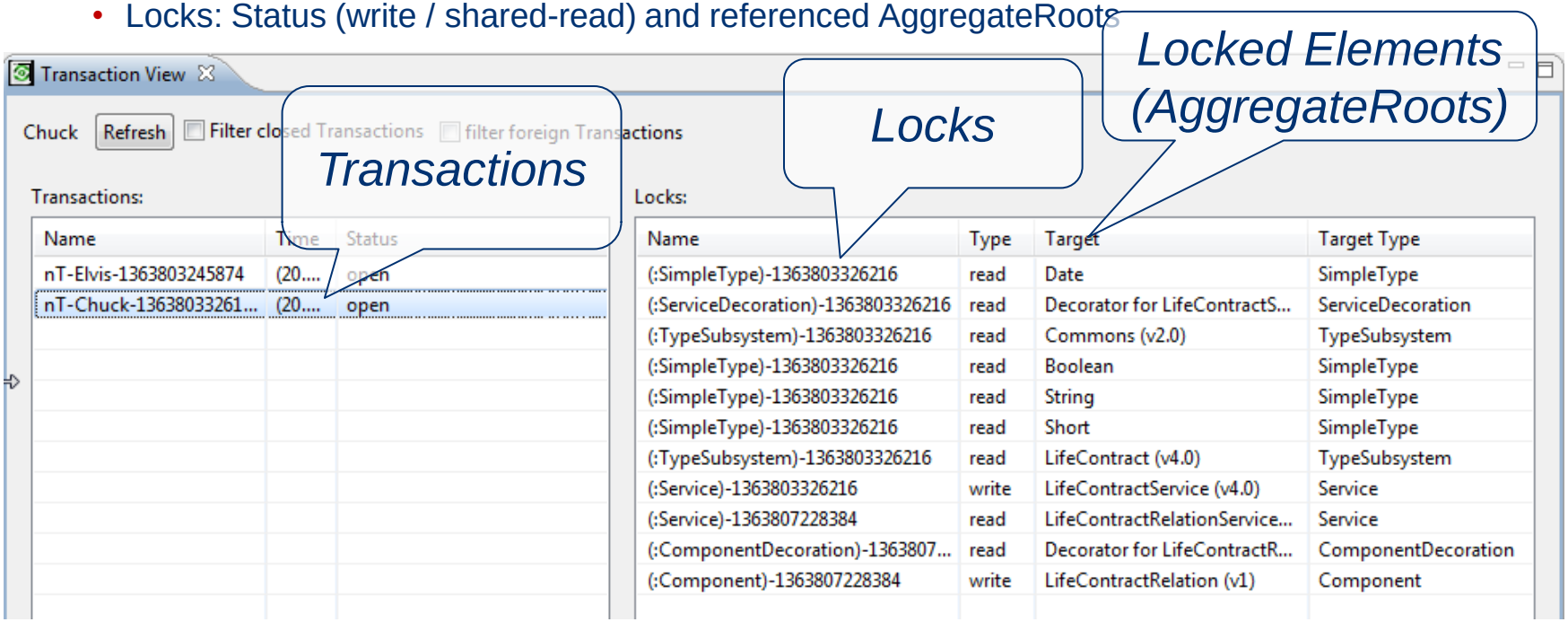
Aggregate Root View

- **Transaction management**
- **Locking**



Transaction and Collision View

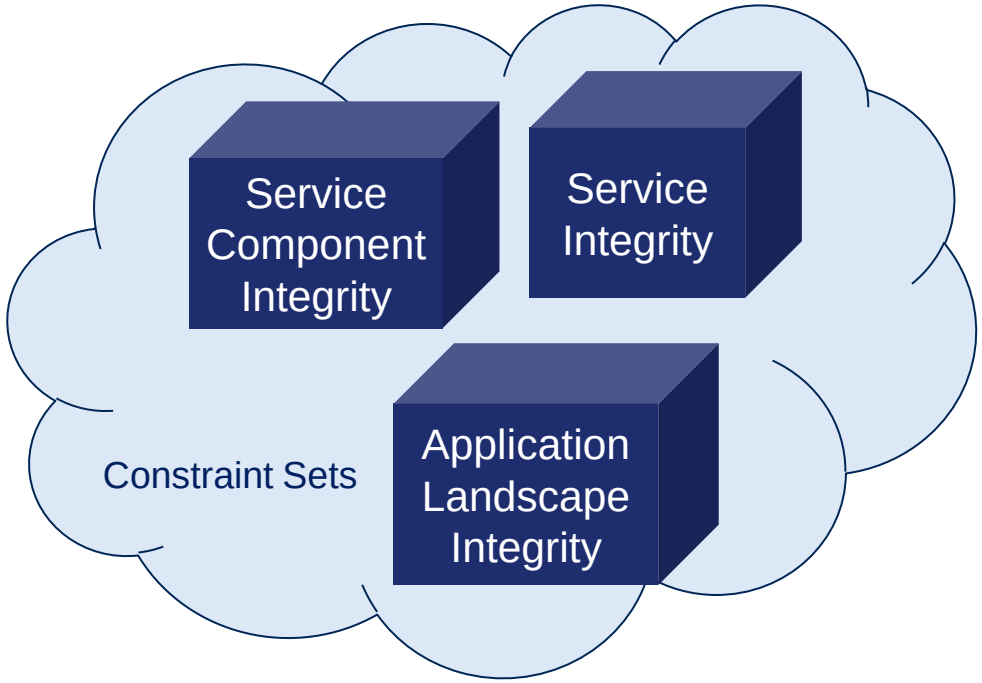
- **Transaction View:**
  - Overview of transaction and locks in Domain Model
  - Locks: Status (write / shared-read) and referenced AggregateRoots



- **Collision View:**
  - Show conflicts if any
  - Show type of conflicts (write/write, write/read, read/write)

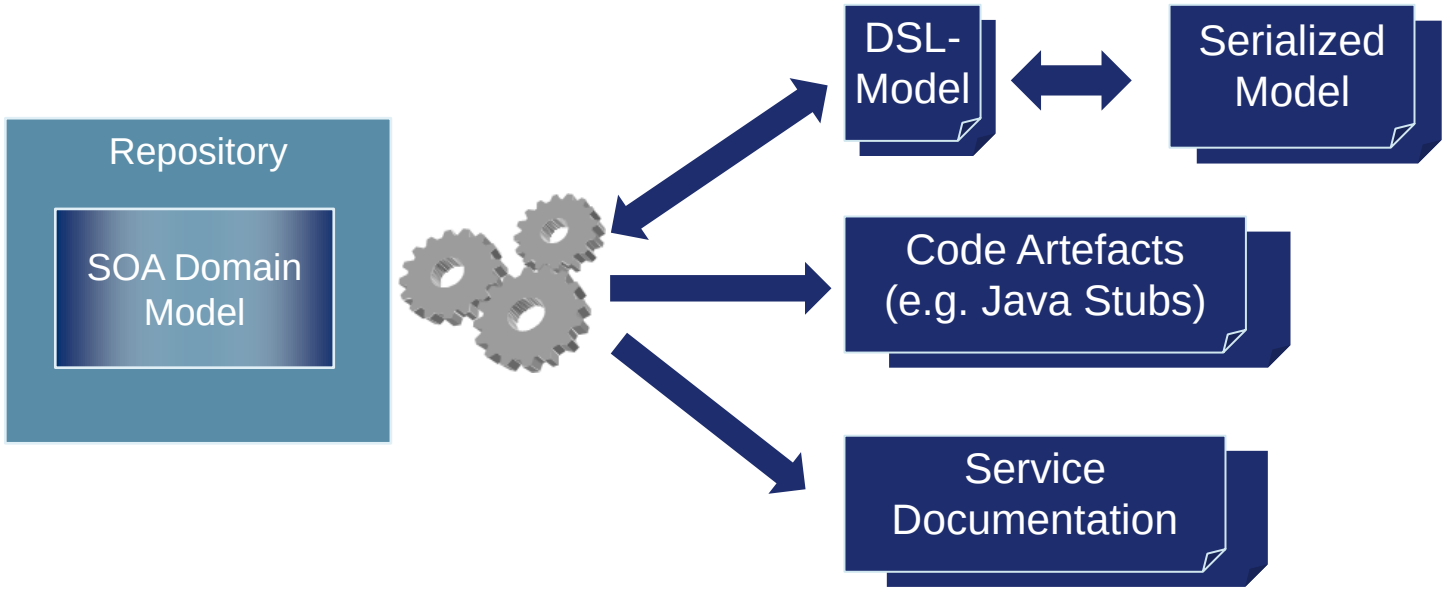
Validation

- Distinguished constraint sets with different grades of wellformedness
- Validation is based on Domain-Model - including rules referring previous (service) versions (life cycle management) - independent of the textual DSL
- Constraint language
  - Specific DSL vs. OCL
  - Technical integration



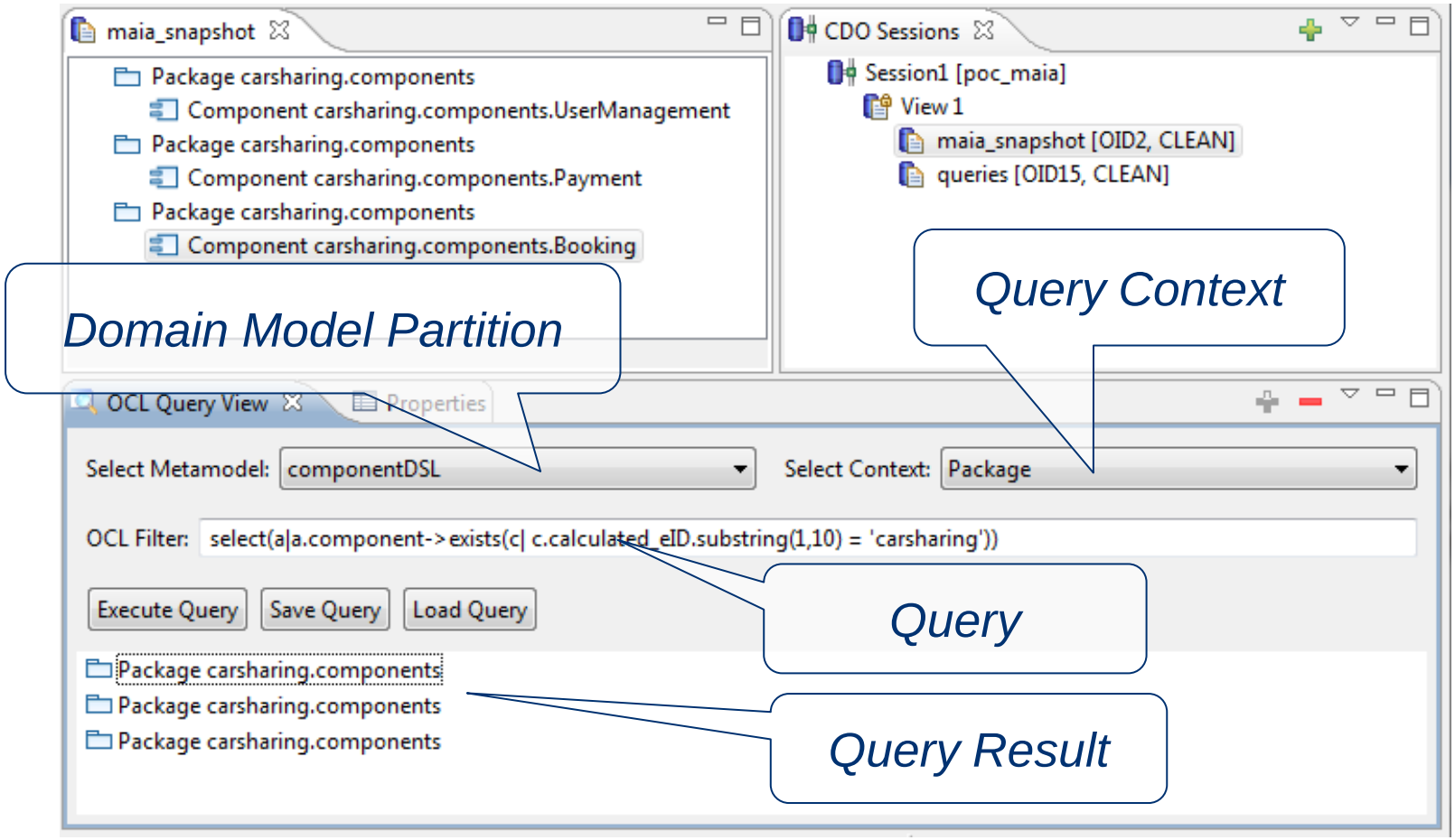
Transformations & Generators

- Bidirectional transformation between DSL-Model and Domain-Model
- Fan out principle
- Generator integration (prototype vs. target vision)



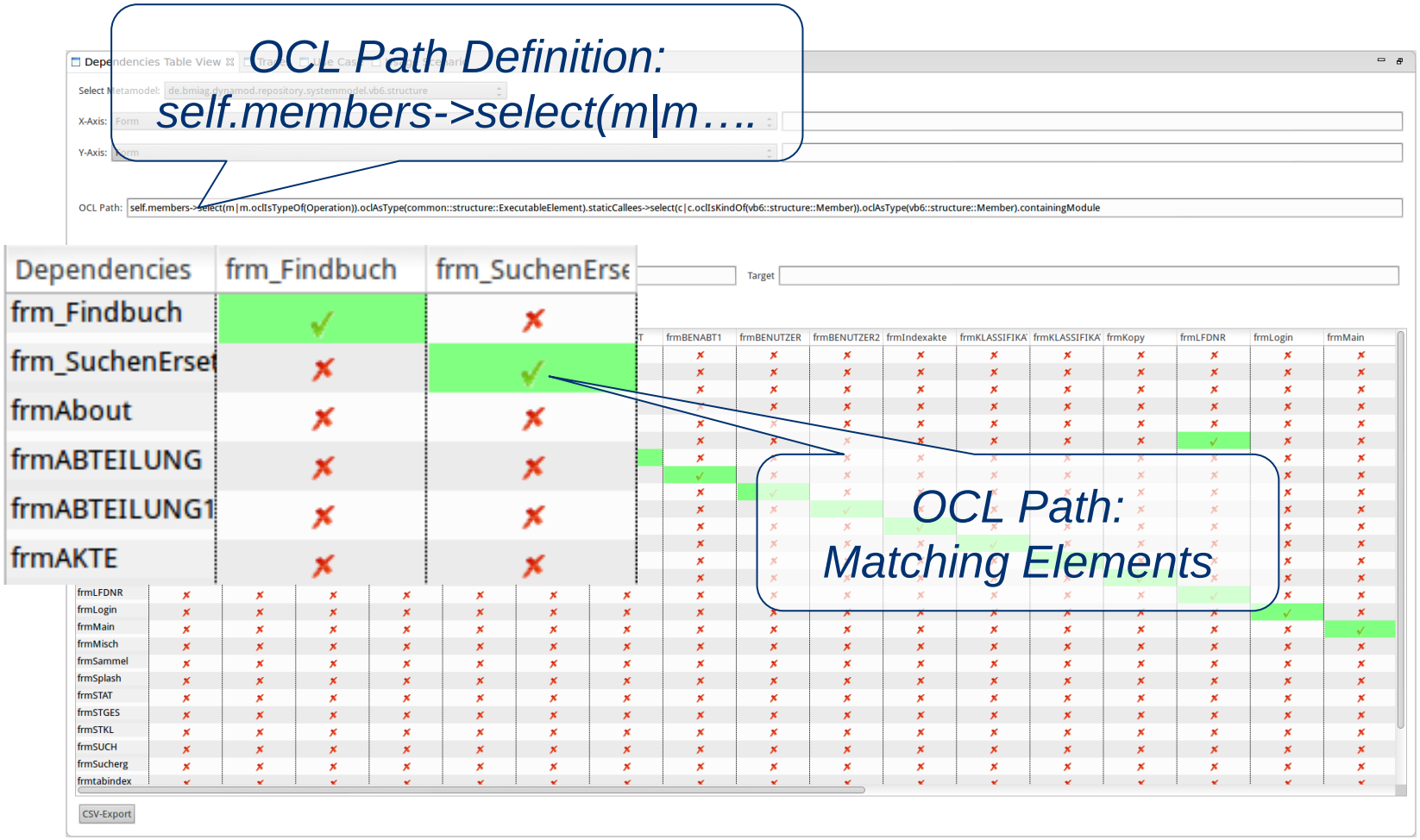
Queries & Reports (1/2)

- Generic query language as a base (e.g. OCL)
- Ad-hoc queries based on Domain-Model
- Queries can be stored in the repository (saved queries)



Queries & Reports (2/2)

- Table visualization of elements connected by a metamodel path



Conclusion & Prospect

- **Conclusion**
  - Advantages of modeling technologies in the context of a SOA repository
    - Reuse of generic building blocks => efficiency
    - Consistency from model to runtime and documentation
    - Metamodel oriented construction => flexibility, adaptability
  - Shift of information ownership from textual model representation into the repository (model runtime)
    => Enhanced support of usage scenarios
    => Enhanced support of different users / roles
    => Improved governance support
    => Multi channel editing (fan in)

- **Prospect**
  - Schema Evolution
  - Alternative persistence media
    - GraphDB, …

Contact

Tom Stahl, Stefan Zeug

b+m Informatik AG
Rotenhofer Weg 20
24109 Melsdorf

t.stahl@bmiag.de, stefan.zeug@bmiag.de

T +49 4340 404- 0
F +49 4340 404-111

b+m business IT management
ALLGEIER GROUP