APSCC 2009, Biopolis, Singapore
10 December 2009 14:00

MORSE:
A Model-Aware Service Environment

Ta´id Holmes, Uwe Zdun, and Schahram Dustdar

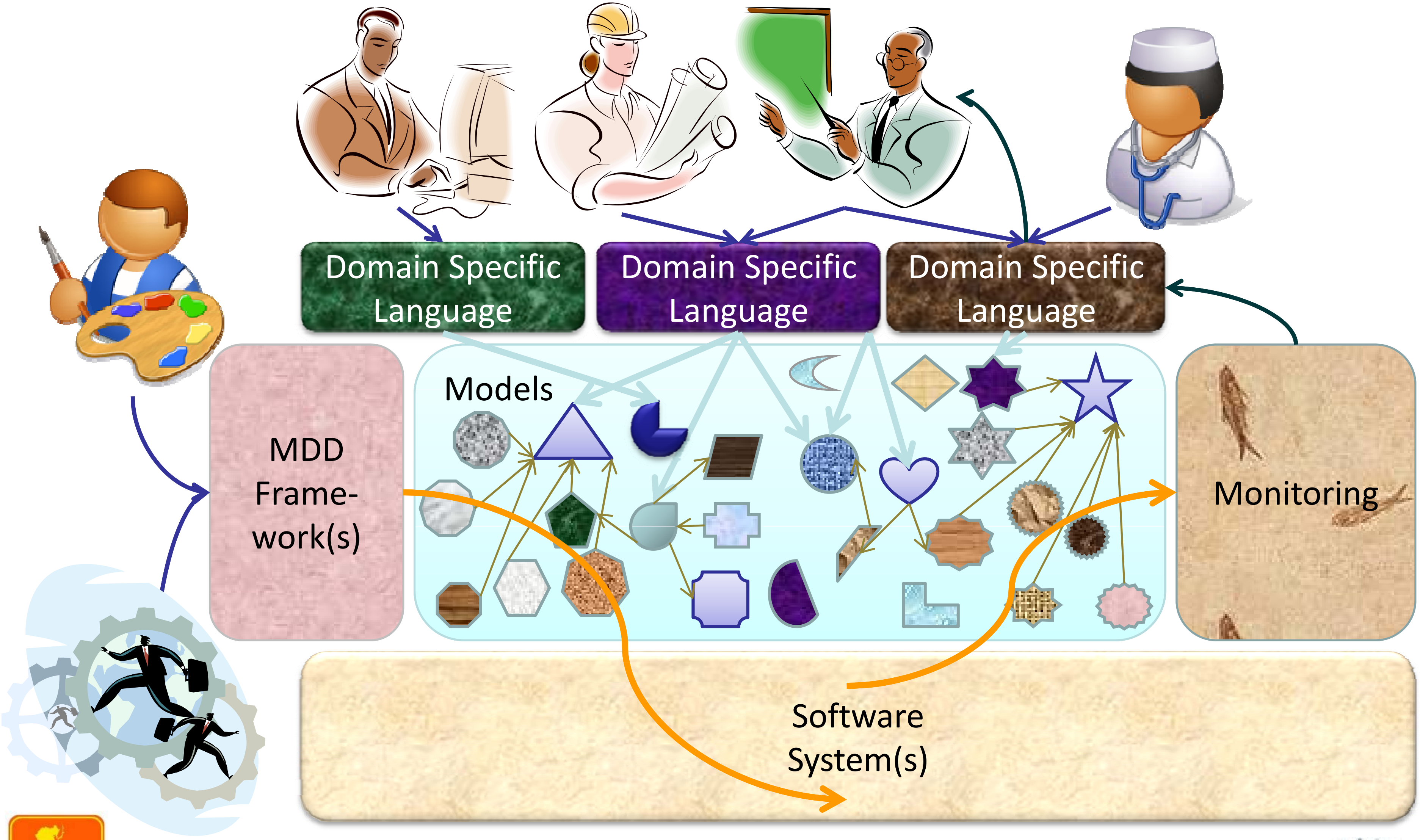
Distributed Systems Group

Institute of Information Systems

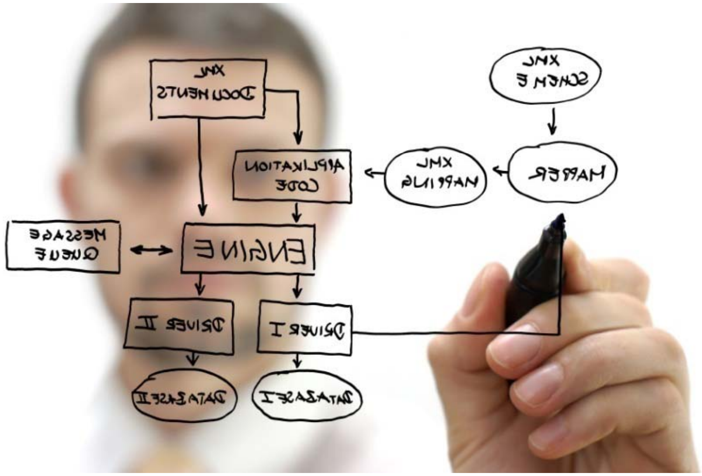
TU Wien

http://www.infosys.tuwien.ac.at

A Model-centered World



Introduction



Models

- precisely specified
- instances can be validated
- can be (d|r)efined at different abstraction levels
- are suiteable to be represented to stakeholders
- can be bound to tailored DSLs

Model-Driven Development

- technical expertise is captured in PIM → PSM transformations; eases e.g., portability, adaptation
- generation of (recurring) code; eases e.g., maintainance

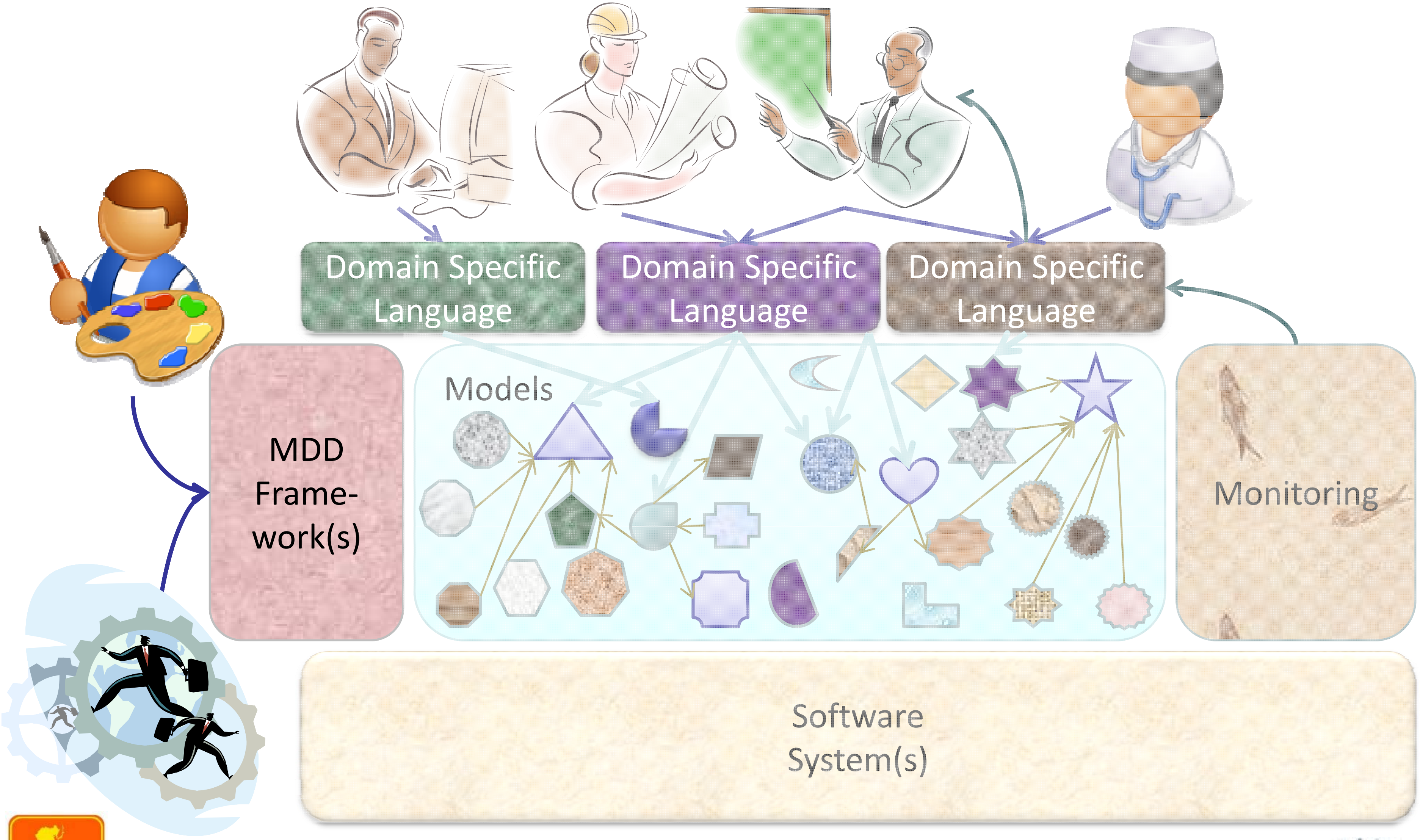
Motivation

- **foster the use of models** in software systems
- support the design activity of (model-driven) engineering processes
- support runtime infrastructures for static and dynamic model-based execution

☺ establish and employ model-aware

- ☺ components
- ☺ systems
- ☺ tools
- ☺ environments

Problems @ Design-Time



Domain Specific Language

Domain Specific Language

Domain Specific Language

MDD Frame-work(s)

Models

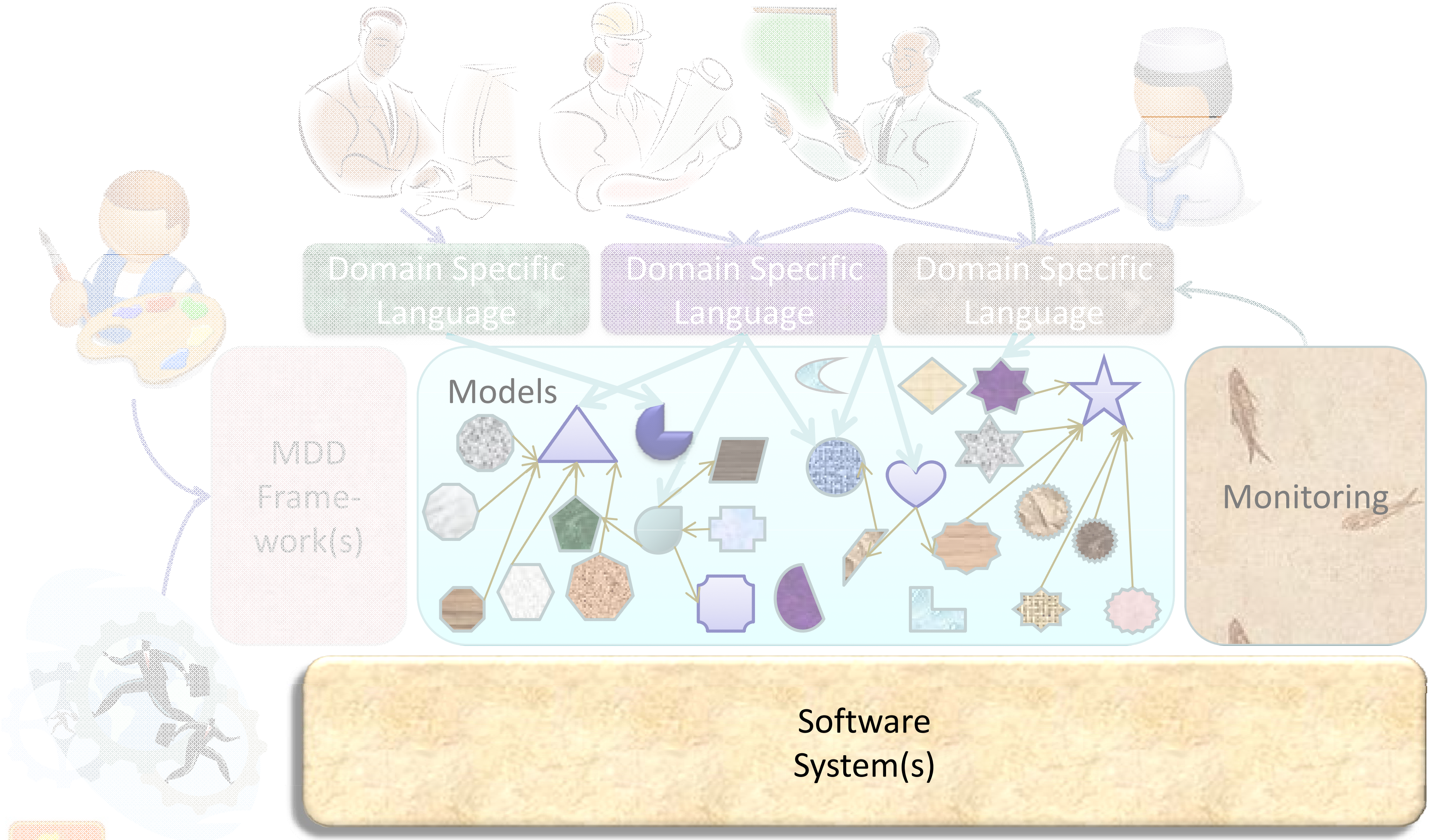
Monitoring

Software System(s)

Problems @ Design-Time

- Concurrent work
  - MDD tools offer little/no collaboration support
    - lack of integration
  - common VCSs (e.g., SVN) are too naïve for MDD
    - versioning on a model-element level is not supported
    - relationships between artifacts are not captured/managed
- Search & retrieval of models
  - missing tool support and infrastructures
  - reuse becomes difficult
  - no knowledge management

Problems @ Runtime



Problems @ Runtime

- Traceability (high-level ↔ low-level model-instances and code)
  - essential for meaningful feedback from runtime to stakeholders and for identifying and understanding the root-cause
- Model-based execution is rarely used: missing infrastructure that supports dynamic lookup of models for model-aware systems

Initial Research-Question(s)

1. How to enable various Stakeholders (⊃ Developers) to collaboratively work on MDD projects and artifacts?
2. How to enable dynamic model-based execution?

Σ = **How to facilitate various model-aware entities to concurrently work with MDD projects and artifacts?**

entities ∈ {Stakeholders, Developers, Systems}

Solutions

- Management of *Projects & Artifacts*
  - versioning
  - capturing and keeping track of relationships
- Support for model-aware *entities*
  - information retrieval service
  - resource management service
- Support for *various* stakeholders (⊂ entities)
  - appropriate model-representations (DSLs)
  - role-based access control (RBAC)
- Facilitating *collaboration*
  - federation & integration with third party services
- Dealing with *concurrency*
  - locking mechanismns
  - raising the awareness of the work of others
  - comparing and merging possibilities
  - support for resolving conflicts

MORSE

.-- .... .- -

.... .- - ....

--. --- -..

.-- .-. --- ..- --. .... -

Samuel F. B. Morse
May 24th, 1844

MORSE

1. **Mo**del-Aware

   because it stores models & MDD artifacts

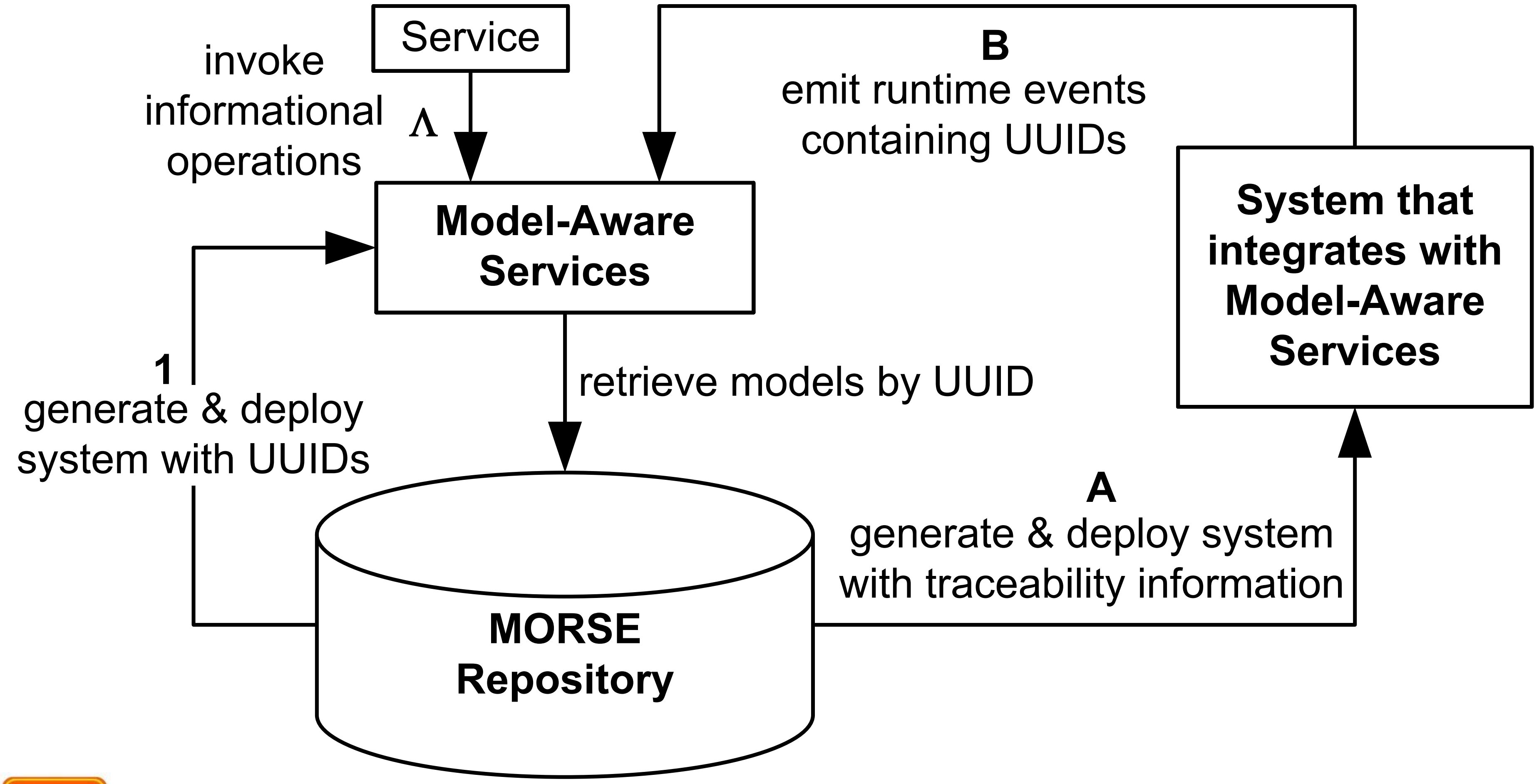
2. **R**epository

   because it supports configuration management (e.g., versioning) of MDD projects

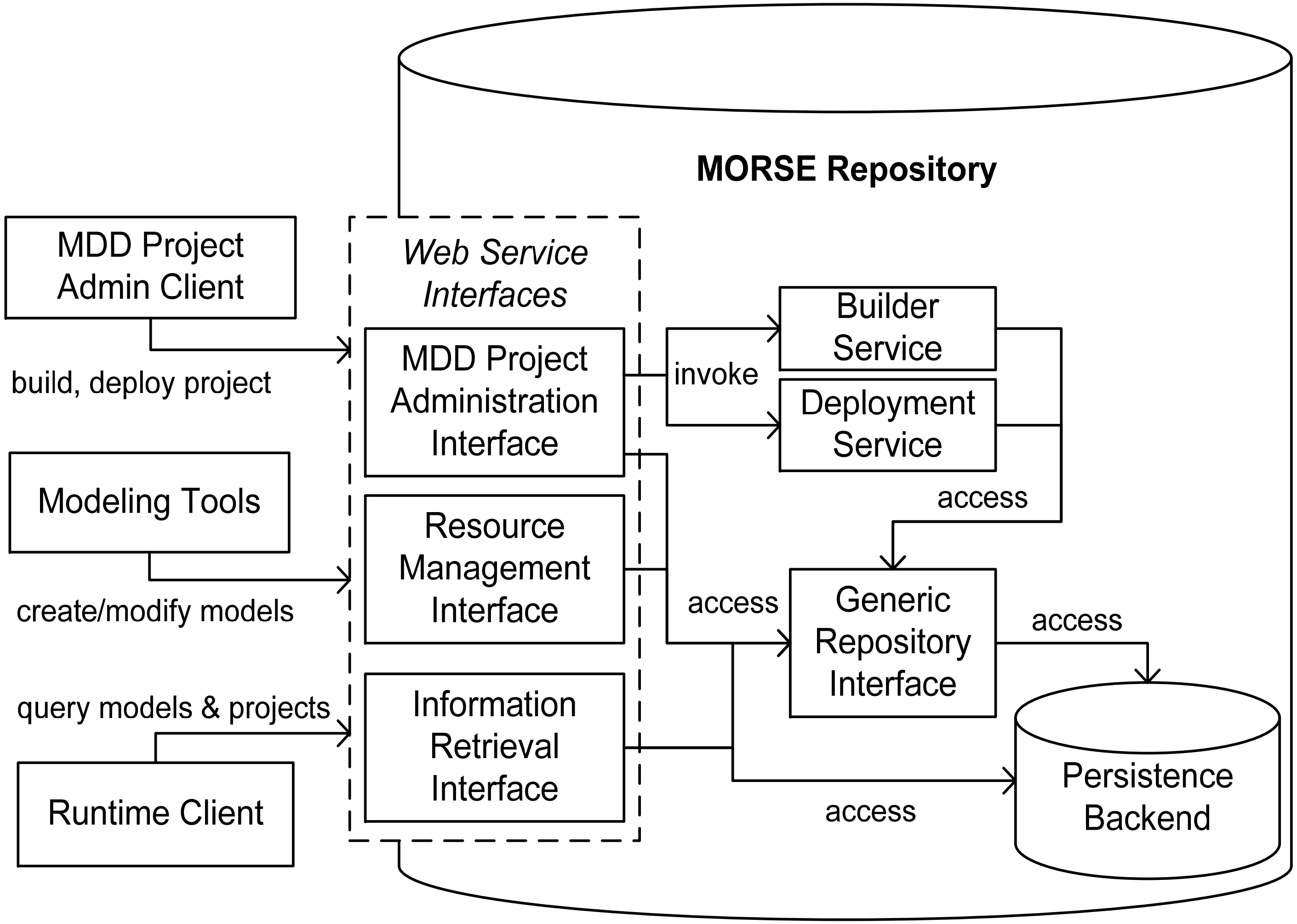
3. & **S**ervice **E**nvironment

   because it offers interfaces and integrates with other model-aware components, that cover the model-driven engineering lifecycle, and provides for specific interactions with these.

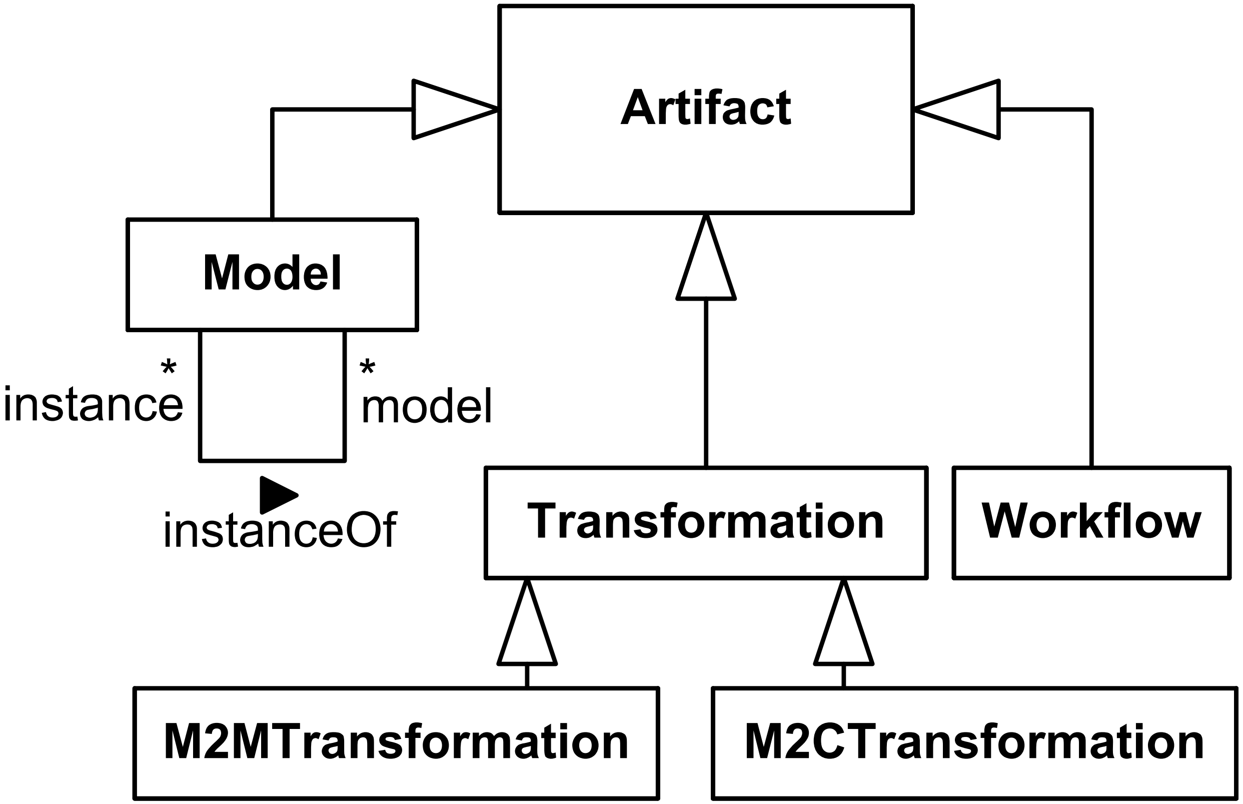
Overview of MORSE


Service
invoke informational operations
Λ
Model-Aware Services
B
emit runtime events containing UUIDs
System that integrates with Model-Aware Services
1
generate & deploy system with UUIDs
retrieve models by UUID
A
generate & deploy system with traceability information
MORSE Repository

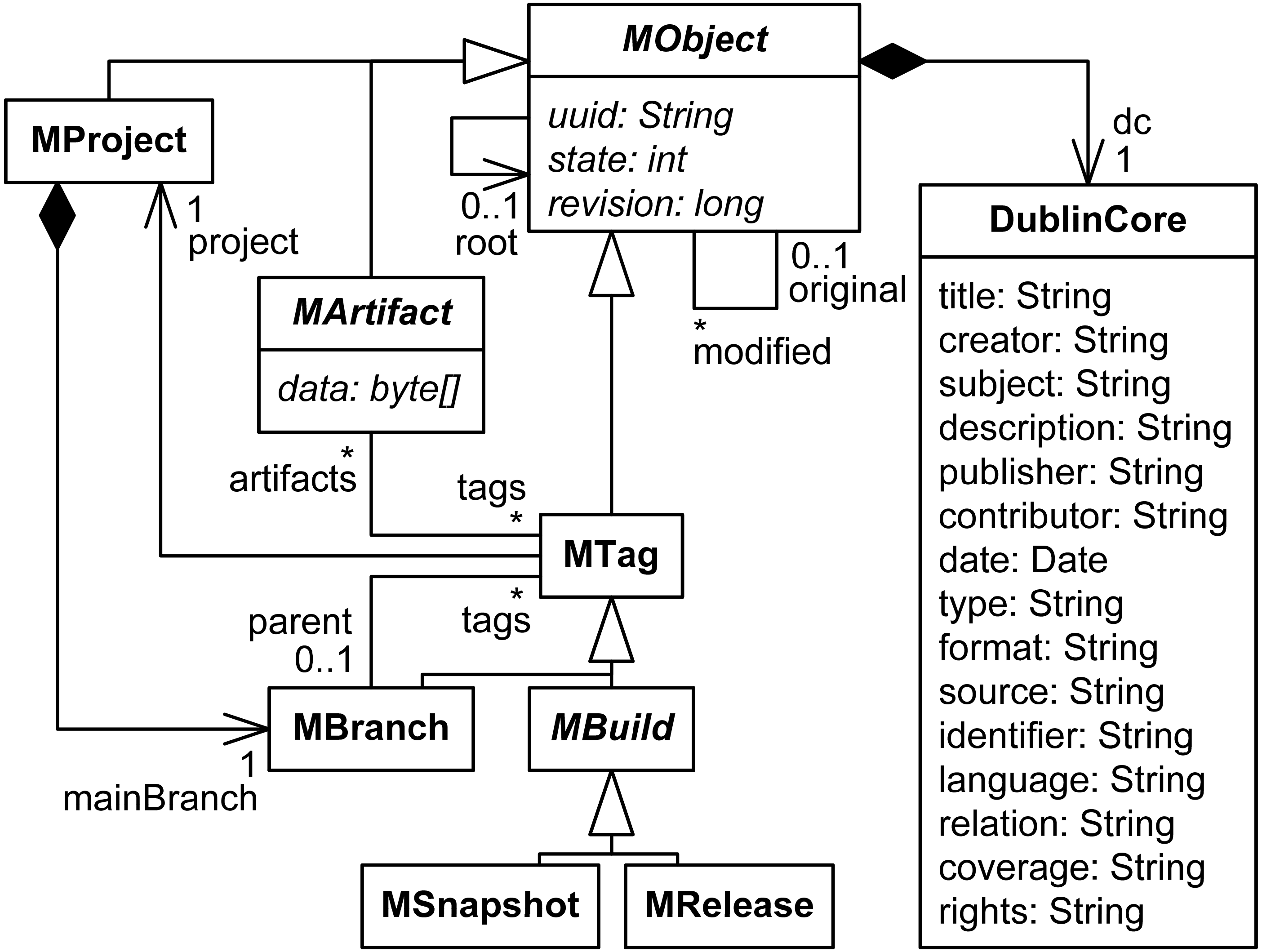
MORSE Architecture



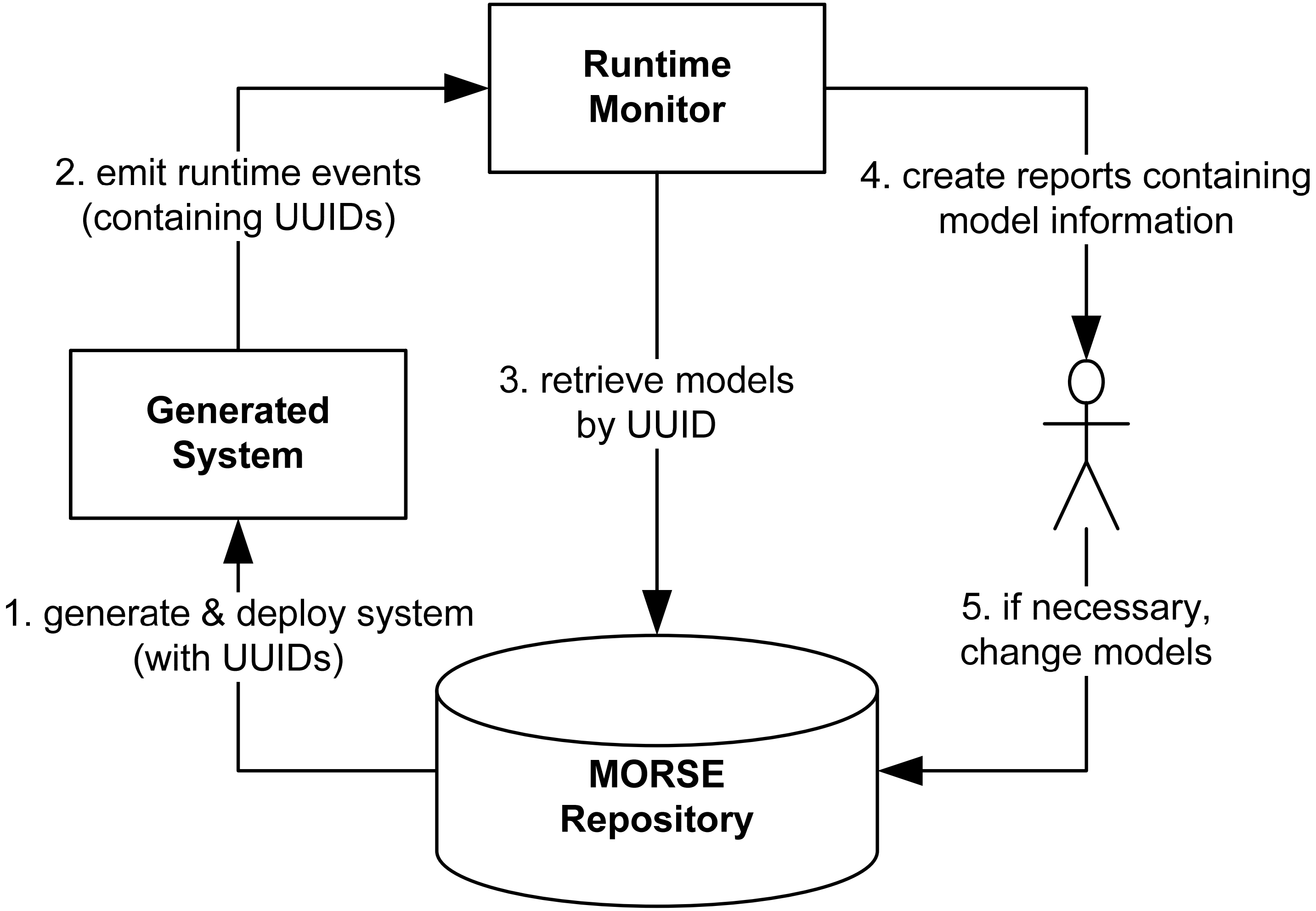
Model of MDD-Artifacts



Model of Projects



Model-Aware Monitoring


Runtime Monitor
2. emit runtime events (containing UUIDs)
4. create reports containing model information
Generated System
3. retrieve models by UUID
1. generate & deploy system (with UUIDs)
5. if necessary, change models
MORSE Repository

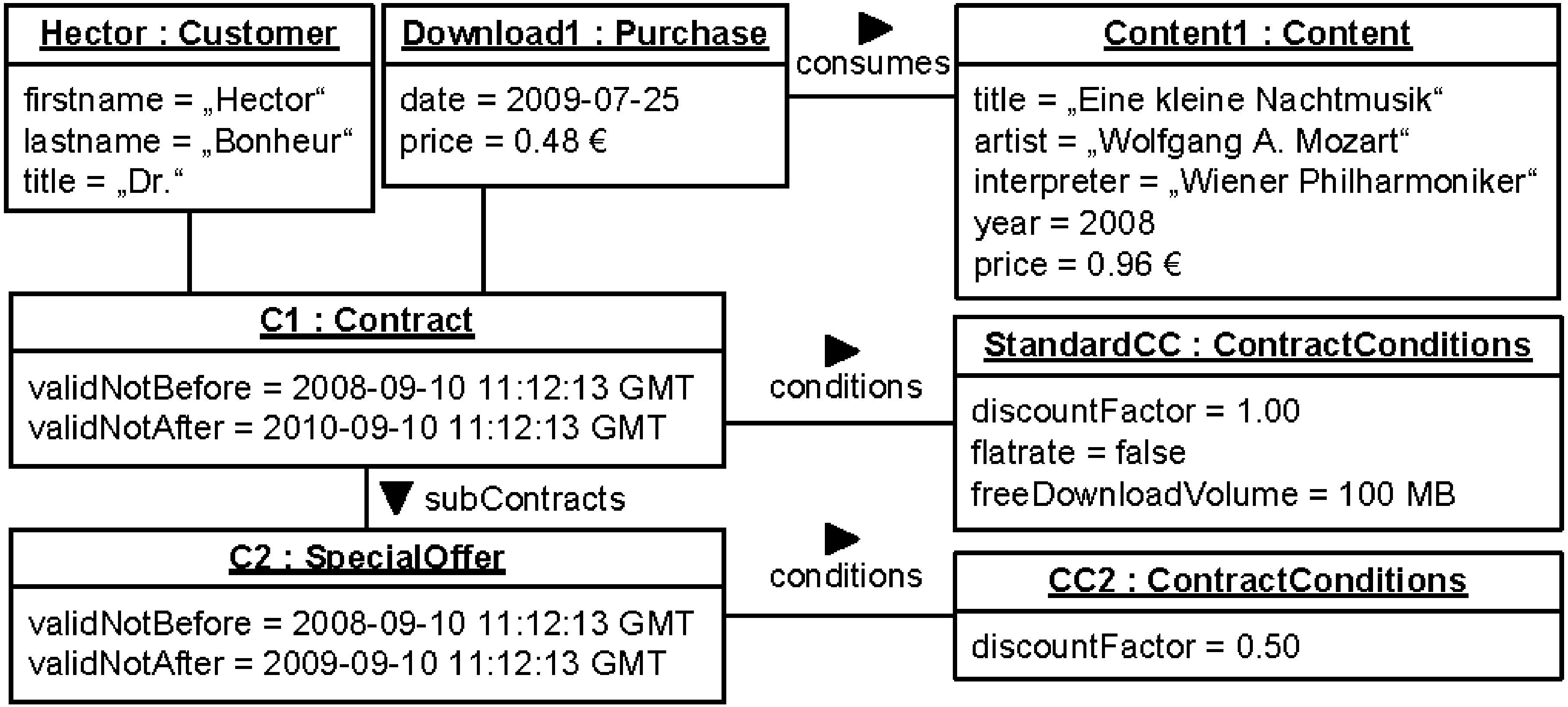
Case-Study

Context

- european telecommunication company
- rich multimedia services
  - customers can subscribe to video or audio streams
  - e.g., download music albums; watch movies
- licensing model: copyright, origin of requestor
- payment model: price, contract, special offers

Case-Study

Model-Instances



Hector : Customer

firstname = „Hector“
lastname = „Bonheur“
title = „Dr.“

Download1 : Purchase

date = 2009-07-25
price = 0.48 €

▶ consumes

Content1 : Content

title = „Eine kleine Nachtmusik“
artist = „Wolfgang A. Mozart“
interpreter = „Wiener Philharmoniker“
year = 2008
price = 0.96 €

C1 : Contract

validNotBefore = 2008-09-10 11:12:13 GMT
validNotAfter = 2010-09-10 11:12:13 GMT

▶ conditions

StandardCC : ContractConditions

discountFactor = 1.00
flatrate = false
freeDownloadVolume = 100 MB

▼ subContracts

C2 : SpecialOffer

validNotBefore = 2008-09-10 11:12:13 GMT
validNotAfter = 2009-09-10 11:12:13 GMT

▶ conditions

CC2 : ContractConditions

discountFactor = 0.50

Case-Study

Payment Algorithm

```
Input: r ε Content, c ε Contract
Output: price ε Price
begin
    cc $\leftarrow$ c.conditions;
    for special 2 c.subContracts!forAll(sc|isValid(sc)) do
        applyConditions(special.conditions, cc);
    if cc.flatrate then
        return 0;
    if 0 $\neq$ c.purchases!forAll(p|p.date+24h>now() $\wedge$ p.content=r) then
        return 0;
    if getTotalDownloadVolume(c) < c.conditions.freeDowloadVolume then
        return 0;
    else
        return r.price * cc.discountFactor;
end
```

Case-Study

Results

- Services do not have to be modified or redeployed: e.g., special offers can be introduced as a new model-relation.
- Access is granted as specified in the effective licensing model. The price calculation considers e.g., the contract and special offers.
- For analyzing customer services the corresponding processes are monitored and related to their originating models.
- The user can be provided with detailed information that he can retrace by reflecting on the models, e.g., before payment or in case of an access violation.

Further Work

- Model-Aware Monitoring
  - Requirements-Monitoring:

    Managing and correlating requirements & system models
  - Recognizing and raising Business Events from low-level events through Complex Event Processing
  - Evolution, Adaptation, Compensation, Synchronization of e.g., Business Processes

Opportunities

- MORSE add-on services
  - validation, provisioning
- Integration with model-aware systems
  - third party resources, services
    - e.g., model-federation, licensing
- (IDE) tool-support
  - collaboration features
    (e.g., awareness of the work of others)
  - information retrieval & resource management
  - reuse & knowledgemanagement

Summary

MORSE

- services can dynamically reflect on
  - models, model-elements, and model-relationships
- model repository
  - management of MDD projects
  - versioning
  - information retrieval
- traceability for business processes

Thanks for your attention!

Ta´id Holmes
Distributed Systems Group
Institute of Information Systems
TU Wien

http://www.infosys.tuwien.ac.at