

Active Knowledge Modeling (AKM) Technology platform

John Krogstie

1



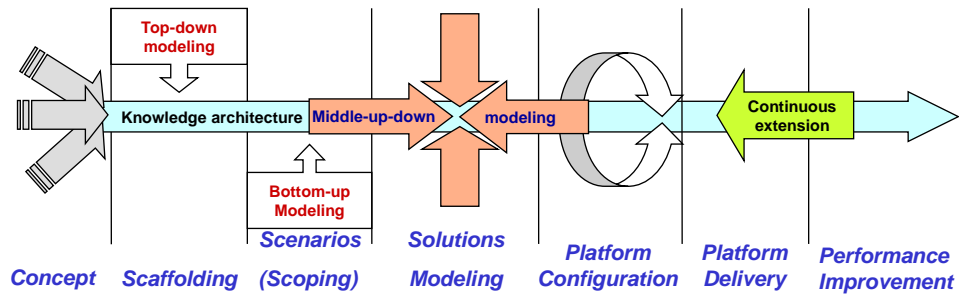
Overview of presentation

- **Based on**
 - AKM-book (parts of chapter 8 and 9, and bits from 5 and 7)
 - Tomorrow: Task Patterns and A14 + example of workplace (end of chapter 9)



2

C3S3P



3

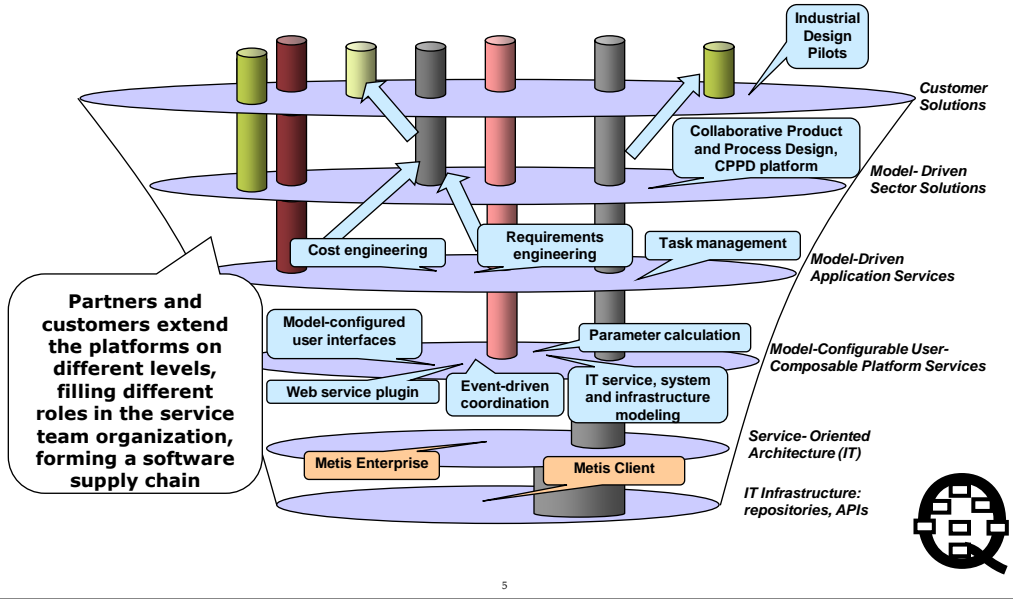


Service Platform Architecture



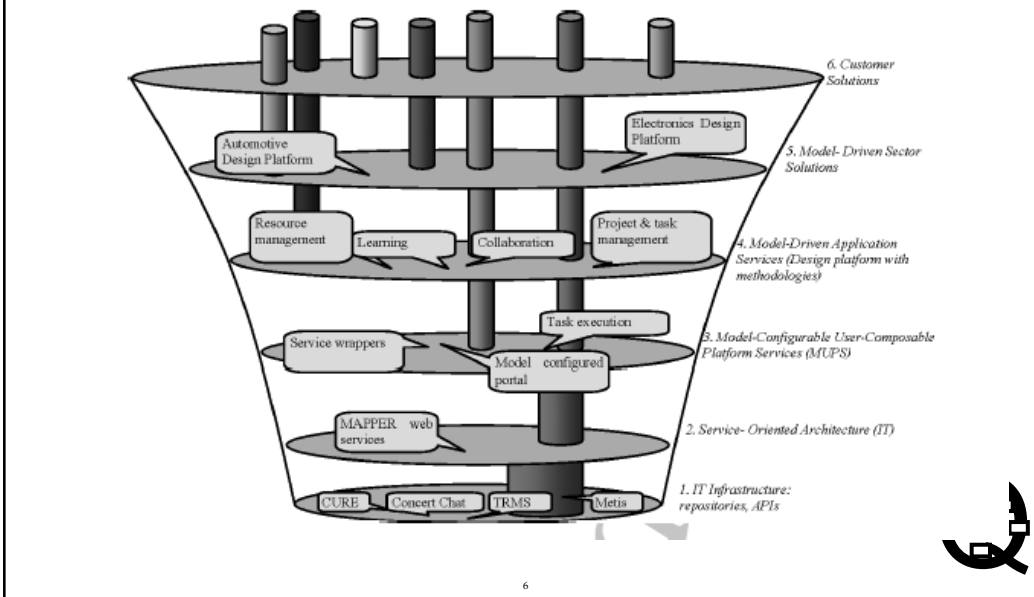
4

Layered Service Architecture



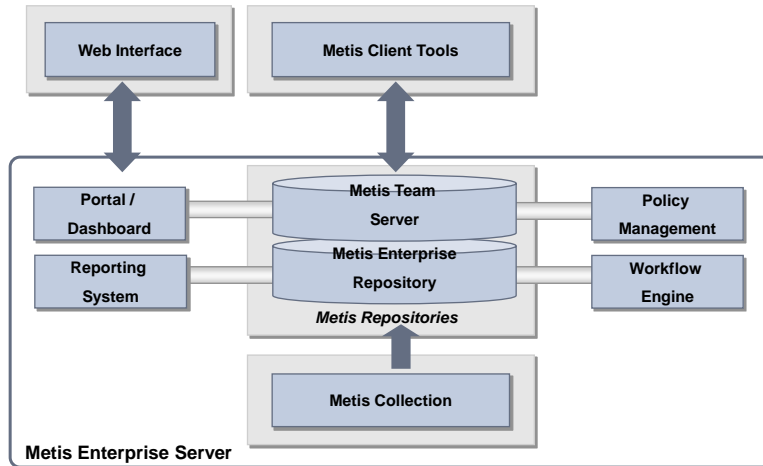
5

Example: MAPPER (chapter 8)

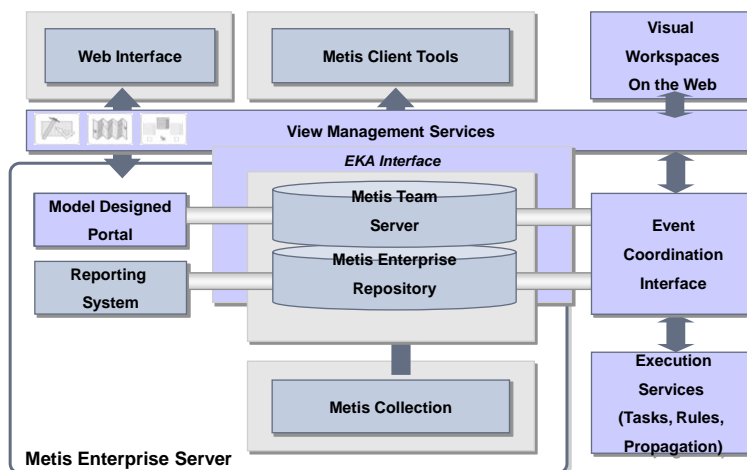


6

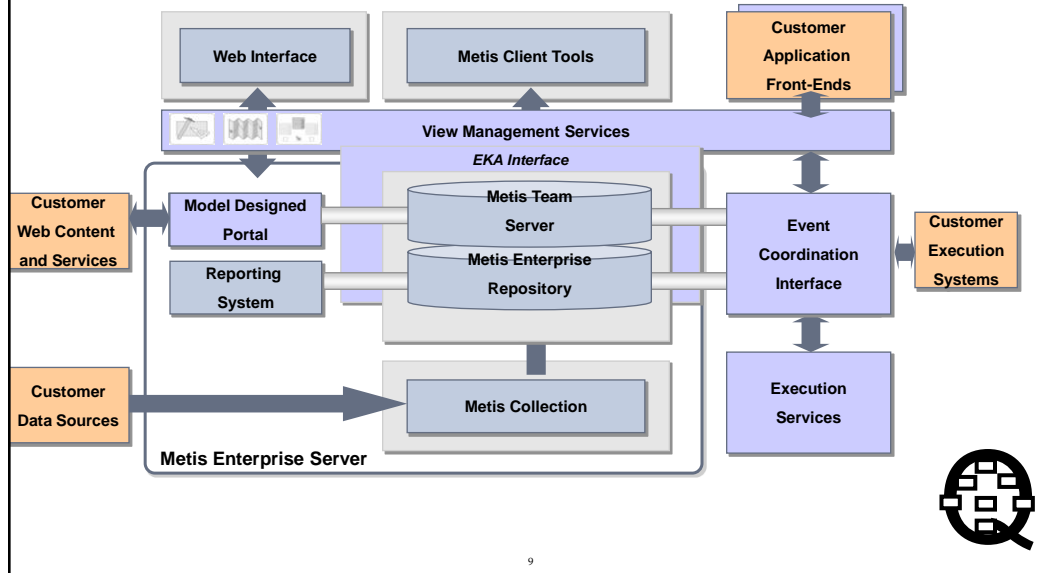
Technical Architecture – Metis Today



Technical Architecture – AKM a.s.



Technical Architecture – AKM in Use



9

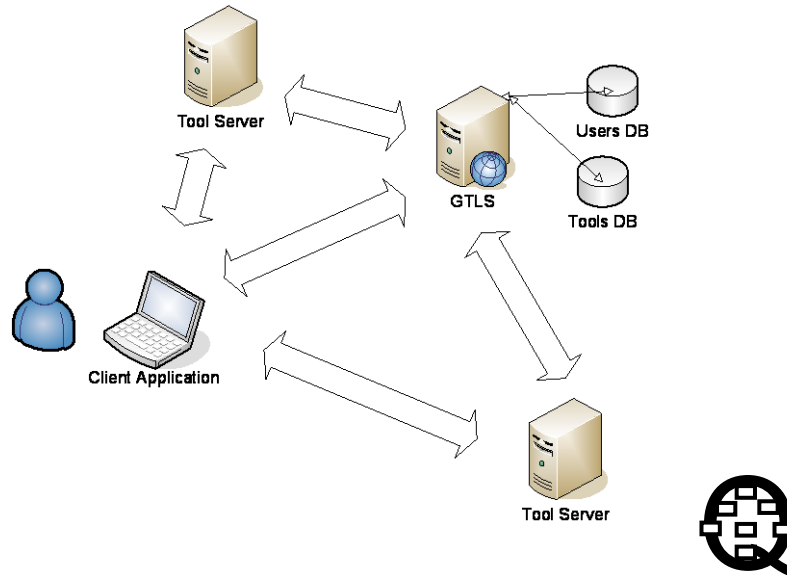
Additional tools in MAPPER

- TRMS
 - Secure invocation of tools
- CURE
 - Facilitate collaboration in distributed teams over the internet
- Concert Chat
 - Synchronus collaboration services

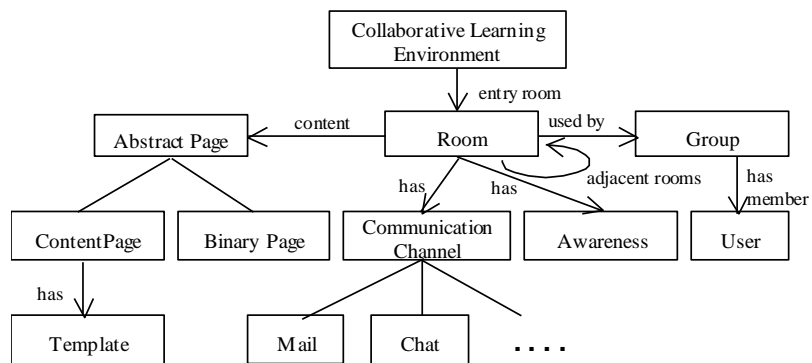


10

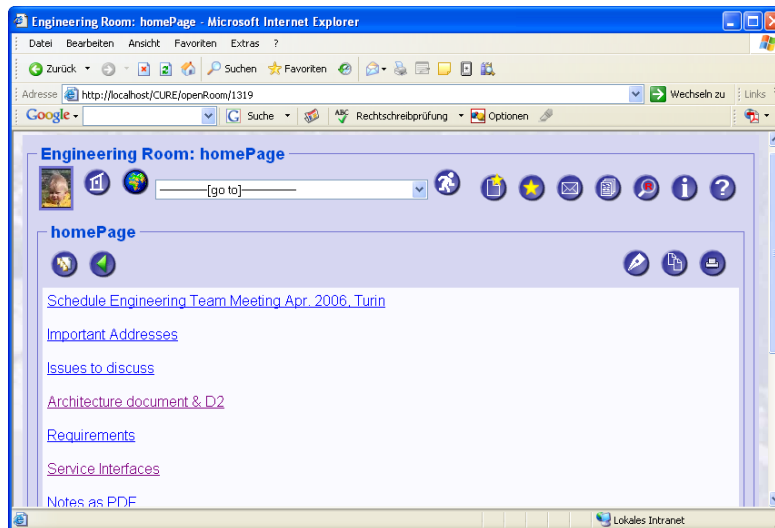
TRMS Architecture



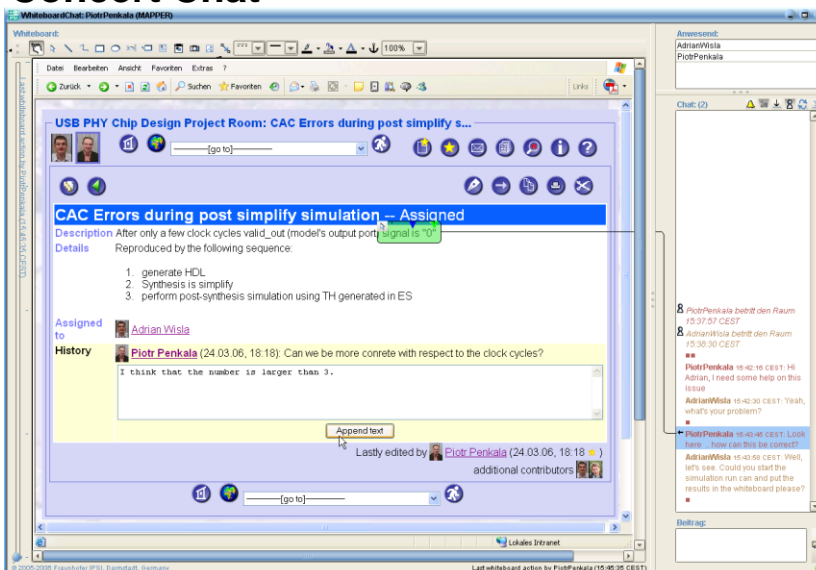
CURE – conceptual design



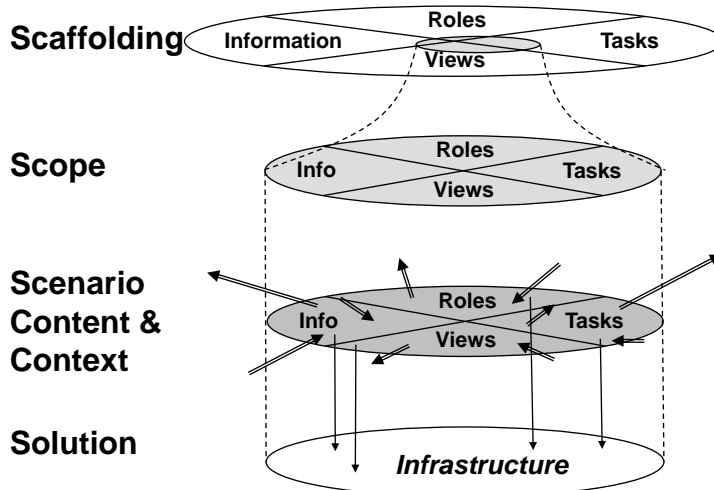
CURE Screenshot



Concert Chat



Overview of the IRTV Methodology



15

State of the Art

- **Information management**
 - Relational databases, XML
 - Transaction management
- **Role management**
 - Hierarchical access control to files
 - Corporate directories and identity management
- **Task management**
 - Business process management and workflow
 - Top-down sequential process models
- **View management**
 - Hard-coded user interfaces
 - Shallow customisation through parameterised options



16

AKM - Technological Innovation I

■ Information management

- Dynamic data formats and languages
- Tolerating mental models' incompleteness and inconsistency
- From object classes to
 - ◆ Instance templates
 - ◆ Multi-dimensional aspect-oriented models
 - ◆ Property and parameter modeling
 - ◆ Relationship management

■ Role management

- Utilises IRTV models
 - ◆ Roles given access to perform tasks on information elements in views
- Inherit specifications in multi-dimensional structures



17

AKM - Technological Innovation II

■ Task management

- Adaptive process models
 - ◆ Top-down work decomposition for management
 - ◆ Middle-out coordination and task assignment
 - ◆ Bottom-up emergent tasks, issue lists
- Structured, semi-structured and unstructured task patterns
- Automatic enactment, manual control, and interactive tasks intertwined
- Event-driven tasks controlled by other dimensions
- Automated tasks as modeled rules or coded scripts

■ View management

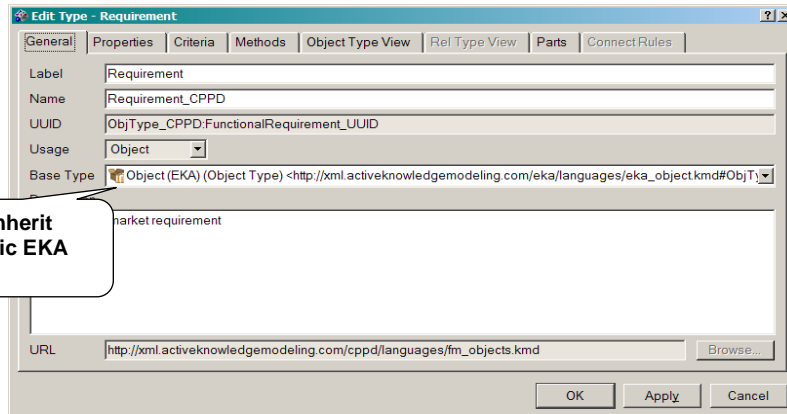
- Simplified customisation though reflecting on evolving IRT models (80% reduction)



18

Information Metamodeling

- Currently you must define new object and relationship types using metamodeling in Metis

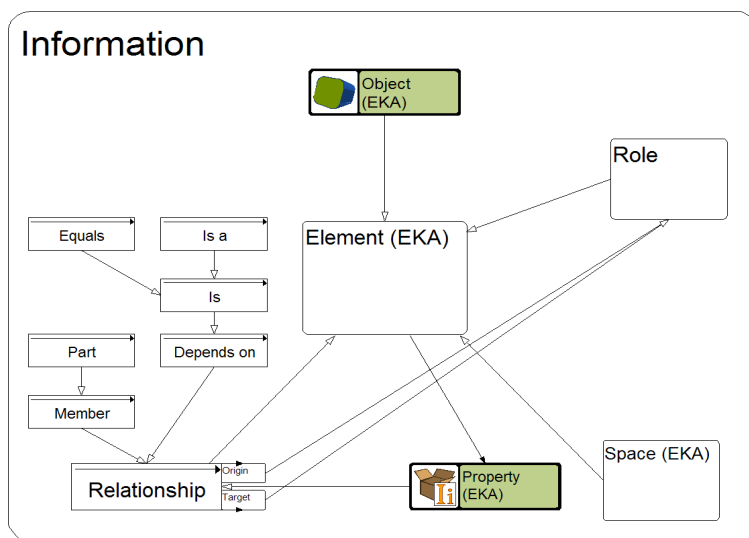


Should inherit from basic EKA types

19



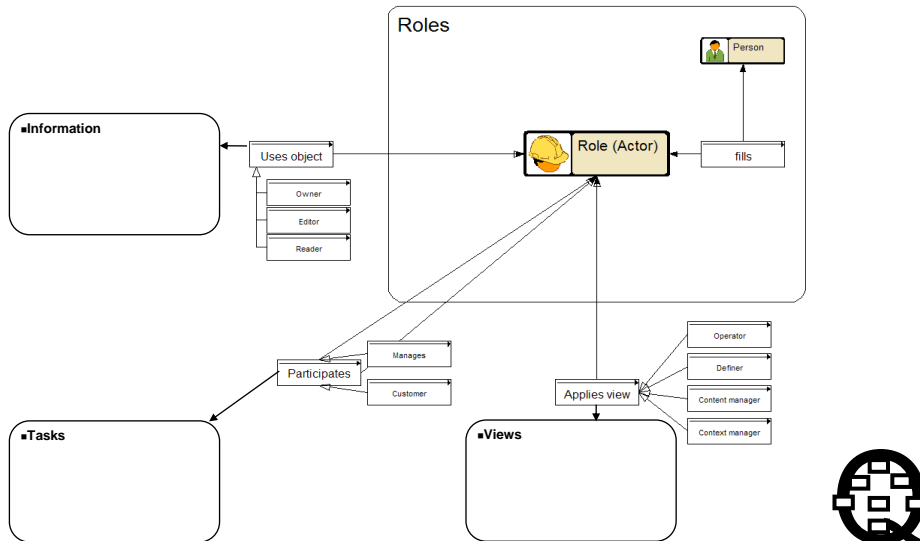
EKA Basic Types



20

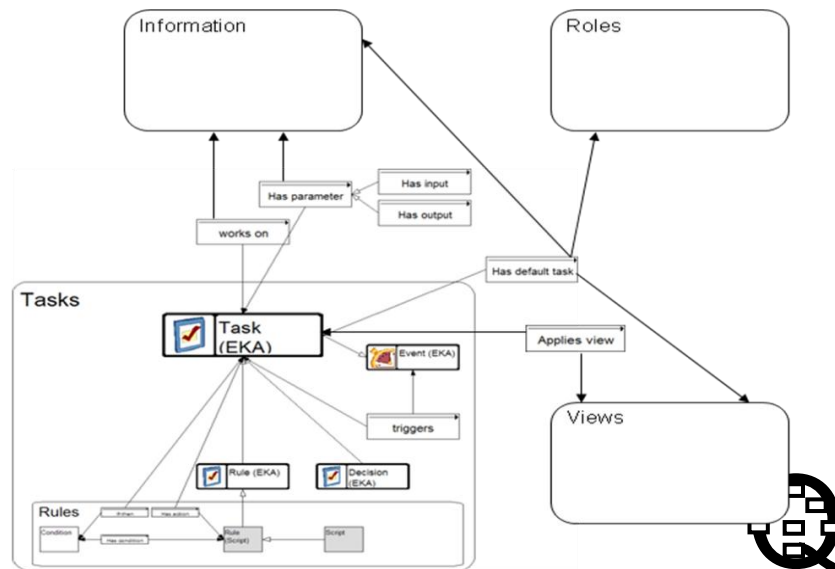


EKA Role Types



21

EKA Task Types



22

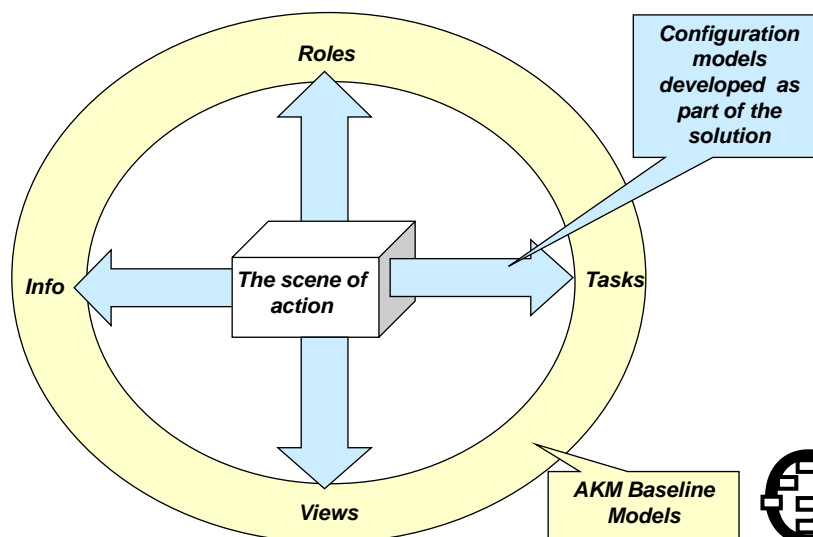
Inheritance

- **Features that determine how elements are handled, can be inherited along multiple dimensions**
 - Properties
 - Relationships to e.g. other IRTV dimensions (has default task, applies view, etc.)
- **Inheritance occurs through**
 - Explicit relationships: Is, Is a, Depends on
 - Types, all elements of type X inherits from all *type representatives* of X
- **Type representatives**
 - Define generic features for all elements of a type
 - May inherit from other objects through explicit relationships
 - All objects whose name equals the type name, and which are either members of the type, or the generic Element and Relationship-Object type, are representatives of the type
 - Objects that are part of the query, are local type representatives for the elements found by that query, inside that particular view



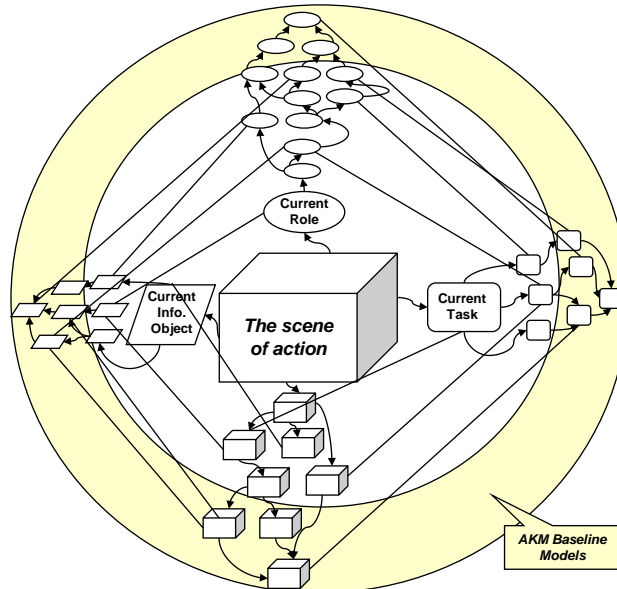
23

Inheritance and Configuration

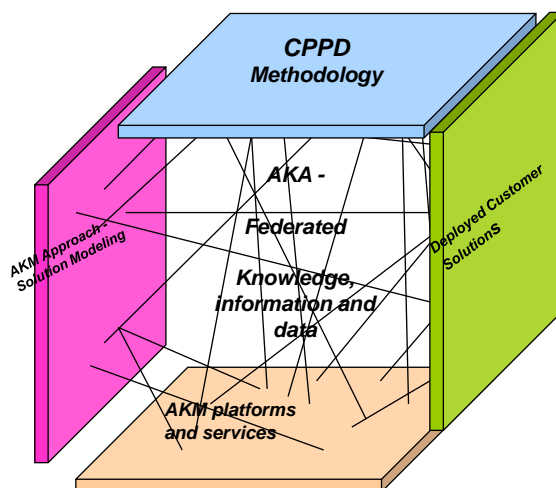


24

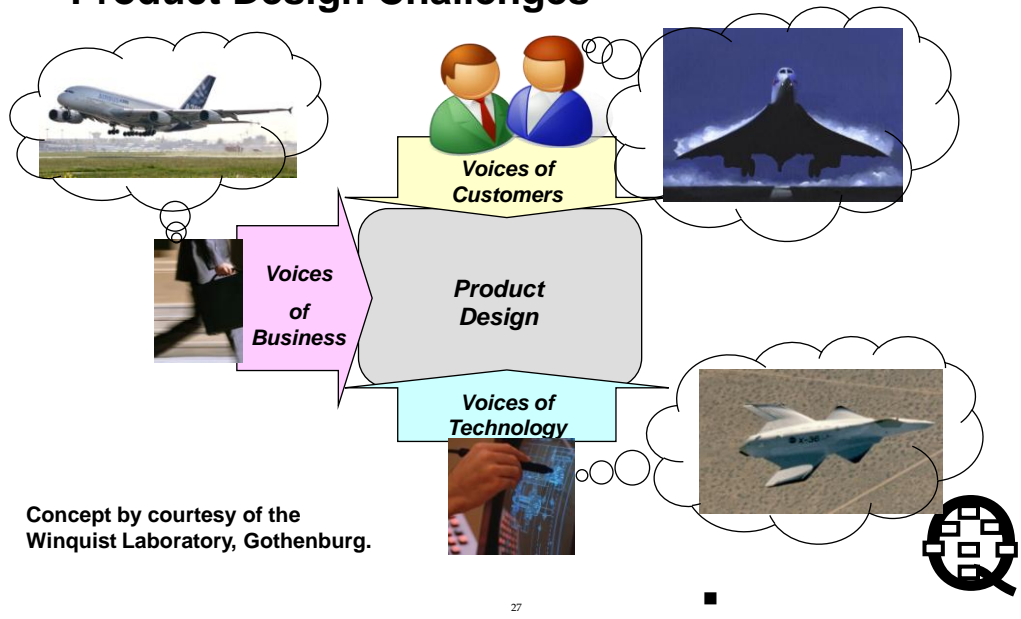
Inheritance – Web in 4 Dimensions



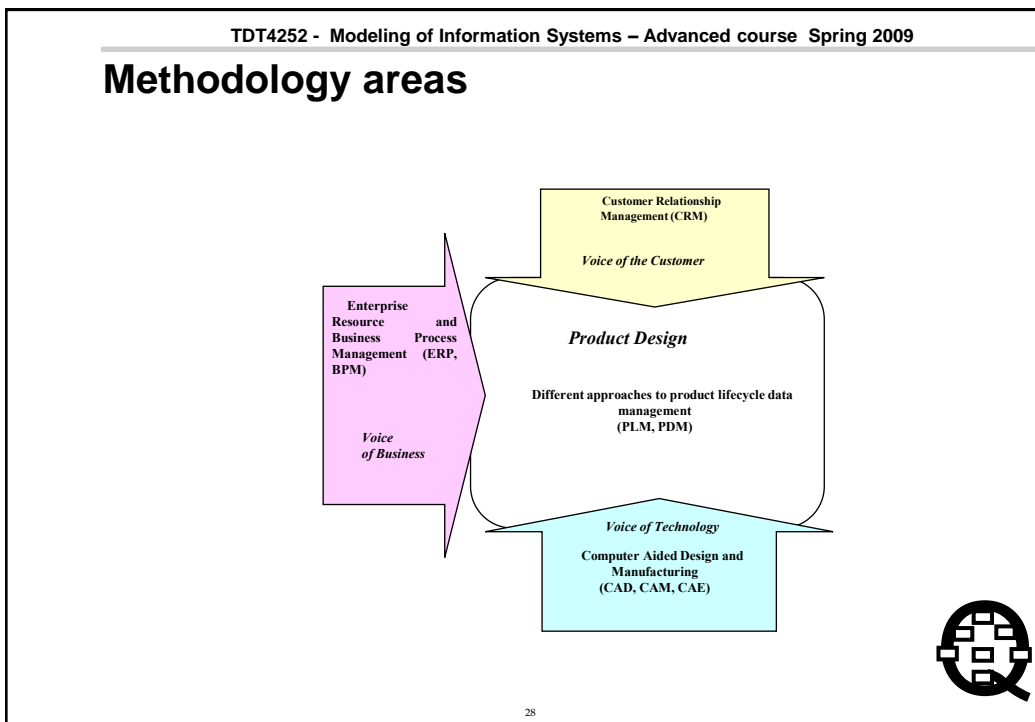
CPPD – Collaborative Product and Process Design



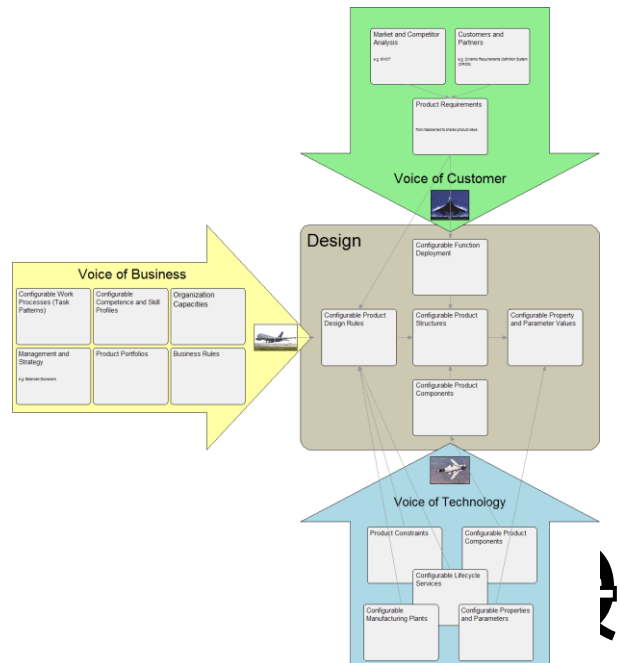
Product Design Challenges



Methodology areas



CPPD Components in product design



CPPD – components - I

■ Design

- CPS: Configurable Product Structure, an early design support language for generic model and services.
- CPC: Configurable Product Components, capturing parameterized variants, shapes and materials
- CPP: Configurable Property and Parameter-sets, making it possible to handle properties, parameters separately by each engineering or business discipline
- CFD: Configurable Function Deployment, to correlate requirements and constraints with product properties and features



CPPD – components II

- **Voice of Business**
 - CWP: Configurable Work Processes, managing dependencies between tasks
 - CCP: Configurable Competence and skill Profiles, for visual competency management
- **Voice of Customer**
 - CIB: Configurable Idea Bank , capturing and relating design ideas, principles, requirements, sketches, constraints and stakeholder views for more effective innovation



31

CPPD – components III

- **Solutions development**
 - CVW: Configurable Visual Workplaces, designing and generating user workplaces
 - CWW: Configurable Web Workplaces, designing and generating workplaces on the web
 - CWI: Configurable Web Service Integration, interfacing legacy systems as web services
 - CDL: Configurable Design Language, linking conceptual EKA to sketches illustrating fundamental and innovative product concepts
 - CCS: Configurable Collaboration Spaces, configuring roles, tasks and views



32

5. Platform Integration

■ Data Integration

- Model-configured import through Metis Data Interchange Format (DIF)
- Databases
- XML

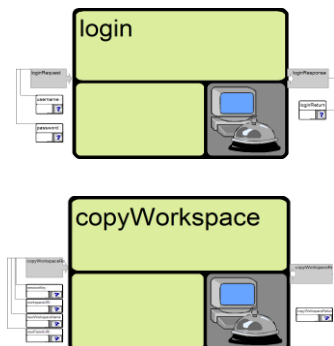
■ Application Integration

- Scripts that utilise APIs
 - ◆ E.g. Excel spreadsheets for user-defined calculations
- Web service import to models
- Portal content and services
- Documents
- Automatic invocation as tasks
- With rich parameter binding support



33

Web Service Integration

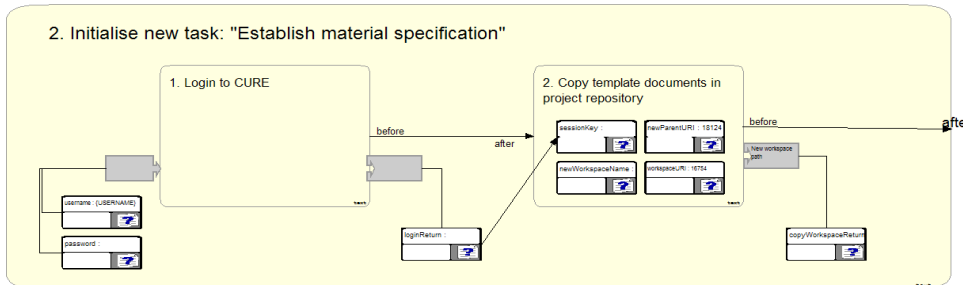


- Web services with parameters are imported into models from WSDL files



34

Web Service Integration II

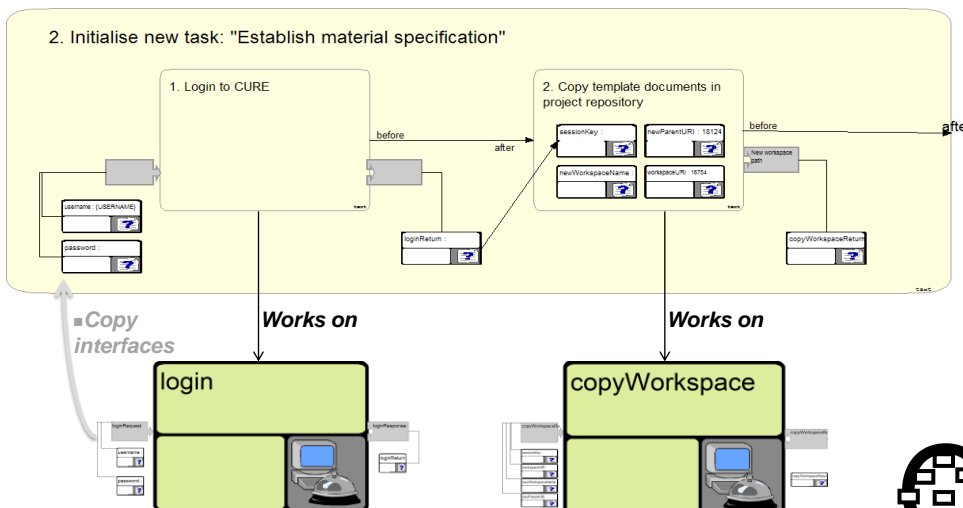


- Task patterns define sequencing and parameters



35

Linking Tasks to Services



36

6. Solution Deployment

- Install Metis client
- Deploy metamodels
- Training
- Personalisation of workplaces
- Super users



37

Packaging the Solution

- Create views of the elements in the content model
- Separate the content of your solution from the structures used for producing it.
 - Future automatic service
- Decide which elements are the main templates
 - Projects, Tasks, ...
- Decide which elements are to be manipulated locally for each occurrence
 - Make them parts of the templates
 - Typically roles and tasks, but not views
 - Information basis or framework, but not templates
- Package the solution model into a reusable submodel
- Create workplace models for each of the supported roles



38

Active Knowledge Modeling (AKM) Technology platform

John Krogstie