

Creating, Filling, and Using a Repository of Reusable Learning Objects for Database Courses

Jutta Mülle

Michael Klein, Khaldoun Ateyeh, Birgitta König-Ries
Institute for Program Structures and Data Organization
Universität Karlsruhe



www.ipd.uni-karlsruhe.de/SCORE



www.vikar.de

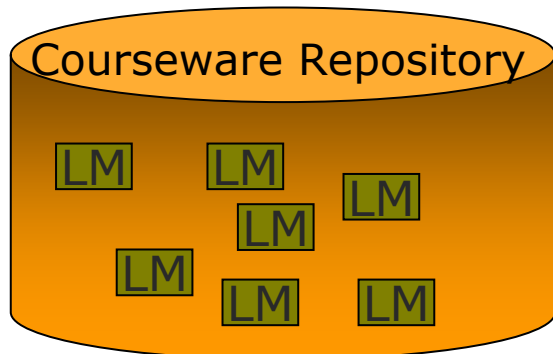
Reuse of Learning Material – Our Scenario



- Cooperate by designing and reusing learning material
- Reduce costs
- Improve quality

Community of Database Groups

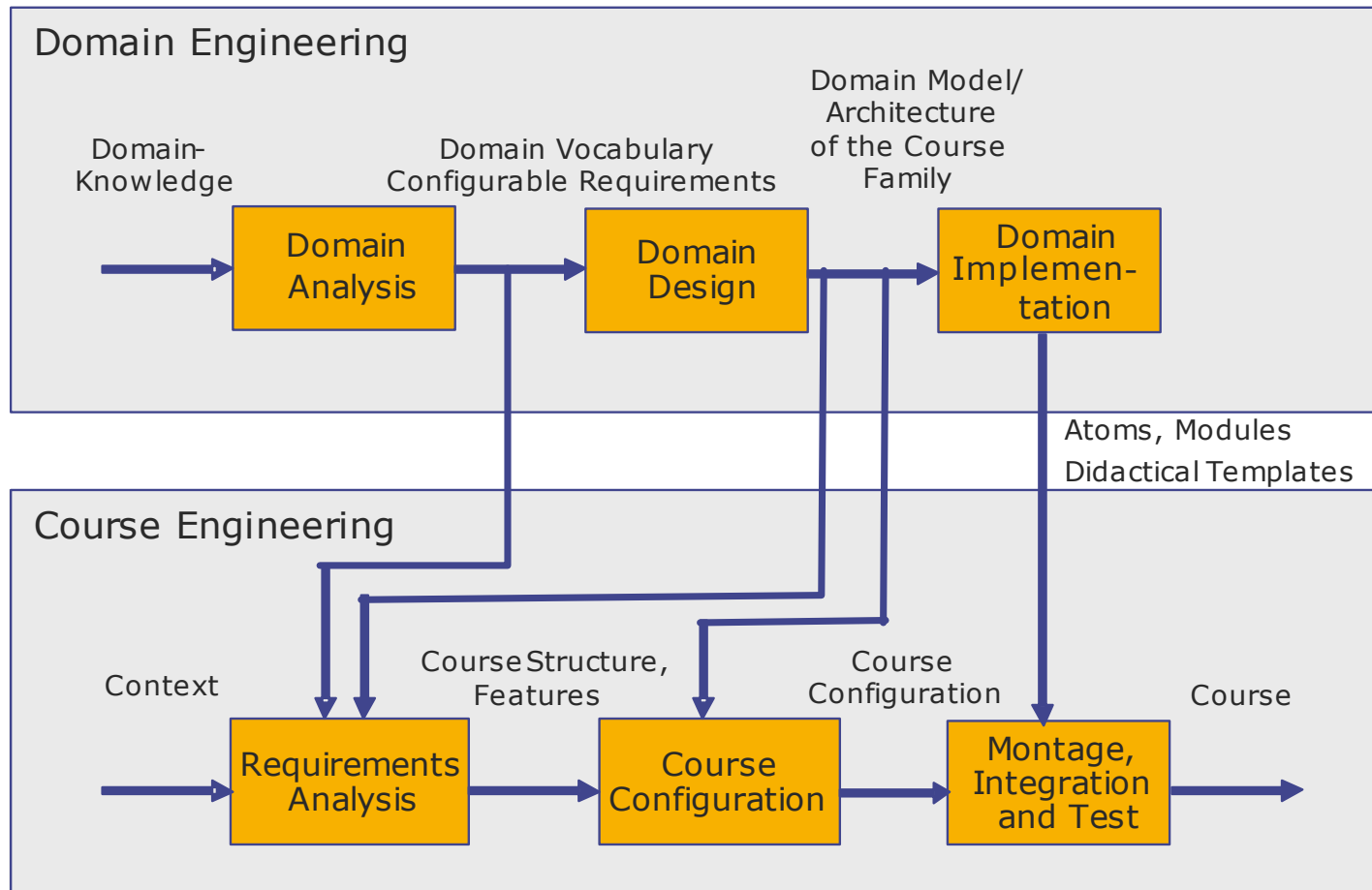
- Informatik, Universität Karlsruhe
- Wirtschaftsingenieurwesen, Universität Karlsruhe
- Wirtschaftsinformatik, Fachhochschule Karlsruhe
- Wirtschaftsinformatik, Berufsakademie Karlsruhe



Learning Material about Database Courses

LM = Learning Module

Process Model



Reuse of Learning Material

- **Domain Engineering:**

Explicit support of the cooperative design and reuse

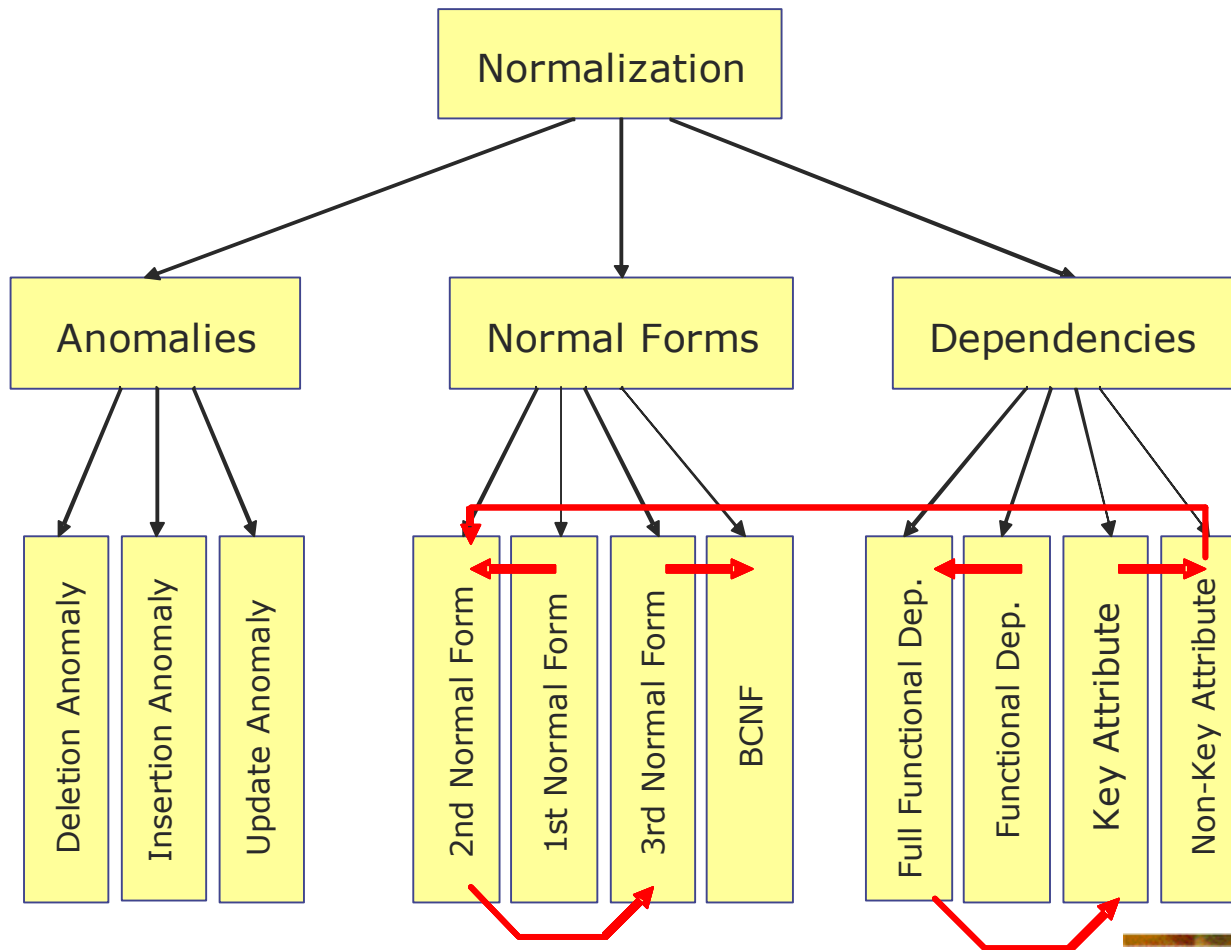
- Development for reuse (⇒ reusable modules)
- Community and domain specific ontology
- Aspect separation
- Goal: course families

- **Course Engineering:**

Reusable based course design

- Design with reuse
- Course configuration
- Course generation

A domain ontology



→
isPrerequisiteFor
(transitive)

→
isSuperTopicOf
(transitive)

Repository of Learning Modules / Classification: Relational Model

Materialsammlung ViKar TP2.3 - Microsoft Internet Explorer

Adresse <http://vikar.aifb.uni-karlsruhe.de/QISEL0203/Begriffe/begriffe.html>

Relationales Modell [DBE3.25b.ppt](#) [DBE3.26b.ppt](#) [A18.5b.ppt](#) [DBS\(FH\)4.2.ppt](#) [DBE_U2.html](#)

- **Formalisierung.** [DBE3.22.ppt](#) [A18.1.ppt](#) [A18.18.ppt](#) [DBS\(FH\)4.3.ppt](#)
 - Typsysteme. [KUD13.2.ppt](#) [KUD13.2b.ppt](#) [DBE3.1.ppt](#)
 - Nullwerte. [A18.3.ppt](#) [DBS\(FH\)4.14.ppt](#)
 - semantische Integritätsbedingungen. [KUD13.3.ppt](#) [KUD13.3b.ppt](#) [DBE3.2.ppt](#) [DBS1.6.ppt](#) [DBS1.6b.ppt](#) [DBS3.1.ppt](#) [DBS3.2.ppt](#) [DBS\(FH\)4.14.ppt](#)
 - Schlüsselbedingung. [DBE3.23.ppt](#) [DBS1.2.ppt](#)
 - Referenzielle Integrität. [DBE3.24.ppt](#) [DBS\(FH\)4.14.ppt](#)
 - Fremdschlüssel. [DBE3.24.ppt](#) [A18.6.ppt](#) [DBS\(FH\)4.9.ppt](#)
 - Relationales Schema. [KUD13.3.ppt](#) [KUD13.3b.ppt](#) [DBE3.3.ppt](#) [DBE3.3b.ppt](#) [DBE3.4.ppt](#) [A18.16.ppt](#) [A18.16b.ppt](#) [DBS1.1.ppt](#) [DBS1.1b.ppt](#) [DBS\(FH\)4.7.ppt](#) [KUD_U4.pdf](#) [KUD_U4.html](#) [KUD_U4.html](#)
 - Normalisierung. [KUD14.13.ppt](#) [KUD14.14b.ppt](#) [DBE13.1.ppt](#) [DBE13.3.ppt](#) [DBS1.3b.ppt](#) [DBS1.4.ppt](#) [DBS1.4b.ppt](#) [DBS1.5.ppt](#) [DBS\(FH\)4.15.ppt](#) [DBS\(FH\)4.16b.ppt](#)
 - Anomalien. [DBE13.2.ppt](#)
 - Einfügeanomalie.
 - Löschanomalie.
 - Änderungsanomalie.
 - Abhängigkeiten. [KUD14.15.ppt](#) [KUD_U6.pdf](#) [KUD_L6.pdf](#)
 - Funktional. [KUD14.16.ppt](#) [KUD14.16b.ppt](#) [KUD14.21.ppt](#) [DBE13.4.ppt](#) [DBE13.4b.ppt](#) [DBS1.7.ppt](#) [DBS1.8.ppt](#)
 - Schlüssel. [KUD14.24.ppt](#) [KUD14.24b.ppt](#) [DBE13.9.ppt](#) [DBE13.13.ppt](#) [A18.2.ppt](#) [A18.2b.ppt](#) [DBS1.14.ppt](#) [DBS1.14b.ppt](#) [DBS\(FH\)4.19.ppt](#)
 - Armstrong Axiome. [KUD14.17.ppt](#) [KUD14.17b.ppt](#) [DBE13.5.ppt](#) [DBS1.11.ppt](#)
 - Hüllen. [KUD14.18.ppt](#) [KUD14.18b.ppt](#) [DBE13.6.ppt](#) [DBS1.9.ppt](#) [DBS1.12.ppt](#)
 - Kanonische Überdeckung. [KUD14.19.ppt](#) [KUD14.19b.ppt](#) [DBE13.7.ppt](#) [DBE13.7b.ppt](#) [DBS1.10.ppt](#) [DBS1.13.ppt](#)
 - Vollfunktional. [KUD14.20.ppt](#) [KUD14.20b.ppt](#) [DBE13.8.ppt](#) [DBS1.15.ppt](#) [DBS1.15b.ppt](#)
 - Transitive funktionale Abhängigkeit. [DBS1.17.ppt](#)
 - Mehrwertig. [KUD14.22.ppt](#) [KUD14.22b.ppt](#) [DBE13.10.ppt](#) [DBE13.10b.ppt](#) [DBS\(FH\)4.5b.ppt](#)
 - Normalformen. [KUD14.23.ppt](#) [KUD14.25.ppt](#) [KUD14.31.ppt](#) [DBE13.11.ppt](#) [DBE13.22.ppt](#) [DBS\(FH\)4.20.ppt](#) [DBS\(FH\)4.28.ppt](#)
 - 1. Normalform. [KUD14.26.ppt](#) [DBE13.12.ppt](#) [A18.4.ppt](#) [DBS\(FH\)4.21.ppt](#)
 - 2. Normalform. [KUD14.27.ppt](#) [DBE13.14.ppt](#) [DBE13.14b.ppt](#) [DBS1.16.ppt](#) [DBS1.16b.ppt](#) [DBS\(FH\)4.22.ppt](#)
 - 3. Normalform. [KUD14.28.ppt](#) [DBE13.15.ppt](#) [DBE13.15b.ppt](#) [DBS1.18.ppt](#) [DBS1.18b.ppt](#) [DBS\(FH\)4.23.ppt](#)
 - Boyce-Codd. [KUD14.29.ppt](#) [DBE13.16.ppt](#) [DBE13.16b.ppt](#) [DBS1.19.ppt](#) [DBS\(FH\)4.17.ppt](#) [DBS\(FH\)4.18b.ppt](#)
 - 4. Normalform. [KUD14.30.ppt](#) [DBE13.17.ppt](#) [DBS\(FH\)4.24.ppt](#) [DBS\(FH\)4.26b.ppt](#)
 - 5. Normalform.
 - Relationale Algebra. [KUD13.4.ppt](#) [DBE3.5.ppt](#) [DBE3.5b.ppt](#) [KUD13.11.ppt](#) [DBE3.19.ppt](#) [A18.7.ppt](#) [A18.15.ppt](#) [KUD_U4.pdf](#) [KUD_U4.html](#) [KUD_U4.html](#)
 - Umbenennung. [DBE3.6.ppt](#)
 - Selektion. [KUD13.5.ppt](#) [KUD13.5b.ppt](#) [DBE3.11.ppt](#) [DBE3.11b.ppt](#) [A18.10.ppt](#) [DBS\(FH\)4.33b.ppt](#)
 - Projektion. [KUD13.6.ppt](#) [KUD13.6b.ppt](#) [DBE3.10.ppt](#) [DBE3.10b.ppt](#) [A18.9.ppt](#)
 - Kartesisches Produkt. [KUD13.7.ppt](#) [KUD13.7b.ppt](#) [DBE3.12.ppt](#) [DBE3.12b.ppt](#) [DBS\(FH\)4.34.ppt](#)

A Didactical Template

Overview

Motivation

Explanation

or

Definition

Exercise

or

Example

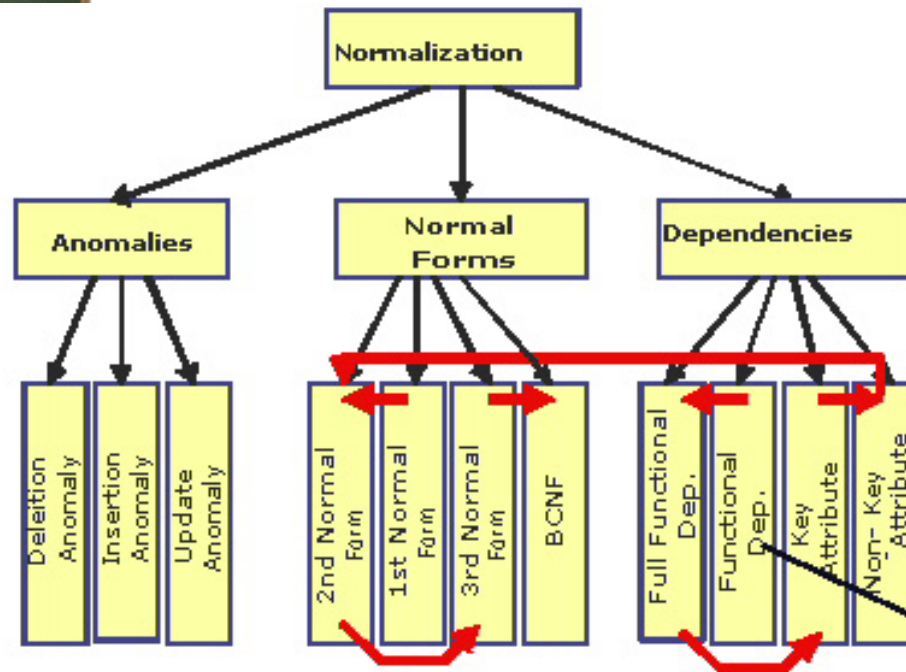
} *

Concluding Example

Exercise

Solution

Learning Atoms



Sentence	Proof
Example	Definition
Explanation	Exercise
Motivation	Solution

Abhängigkeiten

- Normalisierungskalkül beruht auf **Abhängigkeiten** zwischen Attributen als speziellen Konsistenzbedingungen.
- Varianten:
 - Funktionale Abhängigkeiten** als Verallgemeinerung von Schlüsselbedingungen.
 - Mehrwertige Abhängigkeiten** als weitere Verallgemeinerung funktionaler Abhängigkeiten.
 - Inklusionsabhängigkeiten** als Verallgemeinerung von Fremdschlüsselbedingungen (spielen untergeordnete Rolle).
- Wichtig: alle Formen sind echte Konsistenzbedingungen, d.h. Anforderungen an spätere Relationsinstanzen, nicht zufällig erfüllte Eigenschaften.

Funktionale Abhängigkeiten (2)

Funktionale Abhängigkeiten (1)

- Formale Definition:**
 - Sei $R(A_1, A_2, \dots, A_n)$ Relationstyp mit Attributen A_1, A_2, \dots, A_n .
 - Eine **funktionale Abhängigkeitsbedingung** (kurz **FD**) für R ist ein Ausdruck $X \rightarrow Y$ mit $X, Y \subseteq \{A_1, A_2, \dots, A_n\}$.
 - Sei Z die Menge der restlichen Attribute. R **erfüllt** die bedingung $X \rightarrow Y$ wenn für R in jedem Zustand gilt: Sind t_1, t_2 zwei Tupel mit $t_1[X] = t_2[X]$ und $t_1[Z] \neq t_2[Z]$, so existiert ein bestimmter Wert unter Y findet man, also in jedem Tupel, in dem dieser Wert vorkommt, denselben Wert unter Y .

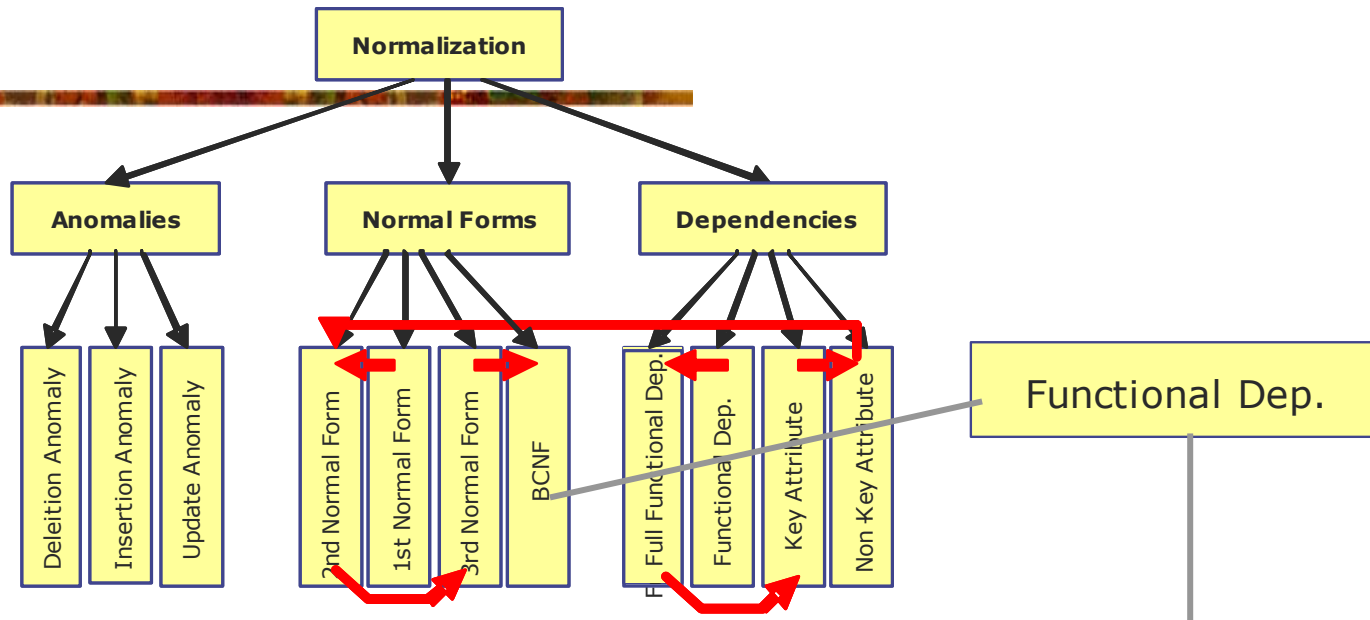
Funktionale Abhängigkeiten: Beispiele

Relation FLUGINFO soll folgende FDs erfüllen:

flugNr \rightarrow von	von,nach \rightarrow entfernung
flugNr \rightarrow nach	ticketNr \rightarrow name
flugNr \rightarrow abflugzeit	flugNr, ticketNr \rightarrow platzCode
flugNr \rightarrow ankunftszeit	flugNr, ticketNr \rightarrow datum
flugNr \rightarrow flypid	flugNr, platzCode, datum \rightarrow ticketNr
flugNr \rightarrow wochentage	

Zusatzinformation: Auf jedem Flughafen darf zu jeder Zeit nur eine einzige Maschine in eine bestimmte Richtung starten:
von,nach,anunftszeit \rightarrow toger

A Learning Module



Module „Functional Dependencies“

Funktionale Abhängigkeiten (2)

Funktionale Abhängigkeiten (1)

Sei $R(A_1, A_2, \dots, A_n)$ Relationstyp mit Attributen A_1, A_2, \dots, A_n .
 Eine funktionale Abhängigkeitsbedingung (kurz: FD) für R ist ein Ausdruck $X \rightarrow Y$ mit $X, Y \subseteq \{A_1, A_2, \dots, A_n\}$.
 Sei Z die Menge der restlichen Attribute. R erfüllt die Bedingung $X \rightarrow Y$ wenn für R in jedem Zustand gilt: Sind t_1 und t_2 Tupel mit $t_1[X] = t_2[X]$ und $t_1[Z] = t_2[Z]$, so $t_1[Y] = t_2[Y]$. Zu einem bestimmten Wert unter X findet man also in jedem Tupel, in dem dieser Wert vorkommt, den selben Wert unter Y .

Definition

Funktionale Abhängigkeiten: Beispiele

Relation FLUGINFO soll folgende FDs erfüllen:

flugNr \rightarrow von	von.nach \rightarrow entfernung
flugNr \rightarrow nach	ticketNr \rightarrow name
flugNr \rightarrow abflugzeit	flugNr.ticketNr \rightarrow platzCode
flugNr \rightarrow ankunftszeit	flugNr.ticketNr \rightarrow datum
flugNr \rightarrow fltypid	flugNr.platzCode.datum \rightarrow ticketNr
flugNr \rightarrow wochentage	

Zusatzforderung: Auf jedem Flughafen darf zu jeder Zeit nur eine einzige Maschine in eine bestimmte Richtung starten:
 von.nach.abflugzeit \rightarrow flugNr

Example

1.4.2 Eigenschaften funktionaler Abhängigkeiten (16)

Lemma 1.1: (Regeln für \rightarrow bzw. \twoheadrightarrow (f^*))
 Vor.: $r: (U \mid F)$, $F \subseteq \Phi(U)$, $A, B, C, D \subseteq U$

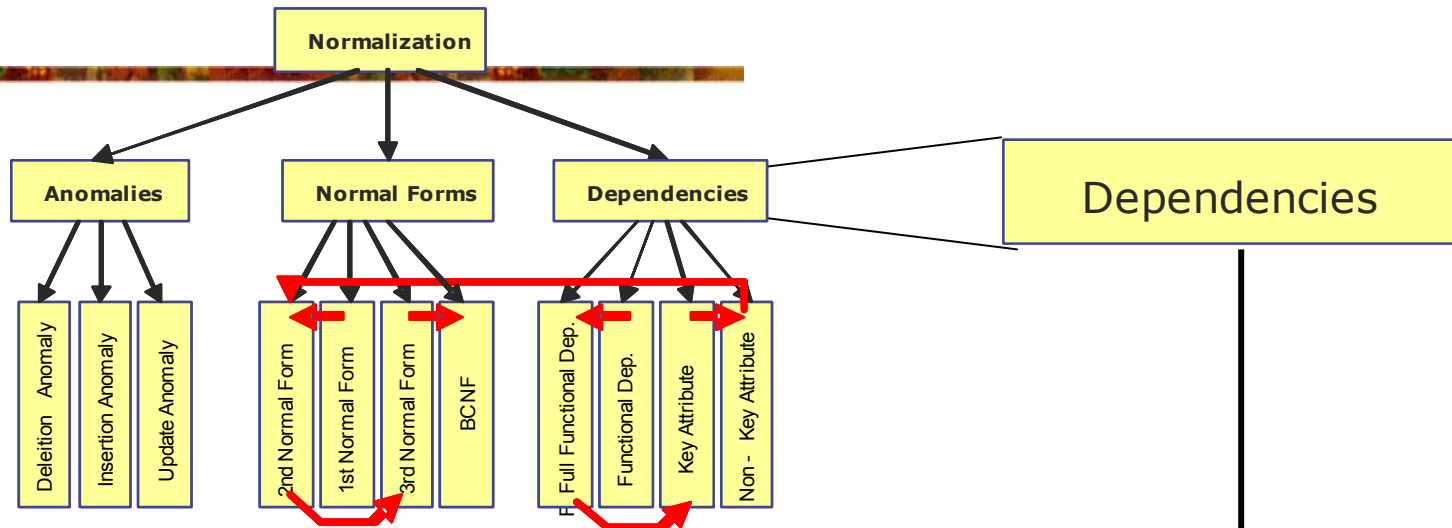
Dann gelten die folgenden Eigenschaften:

(1)	$B \subseteq A \rightarrow A \rightarrow B$ (f)	(Reflexivität/Projektivität)
	bzw. $B \subseteq A \twoheadrightarrow A \rightarrow B$	(insb. gilt immer: $A \rightarrow A$)
(2)	$A \rightarrow B \wedge AC \rightarrow BC$	(Erweiterungsregel)
(3)	$A \rightarrow B, B \rightarrow C \twoheadrightarrow A \rightarrow C$	(Transitivität)
(4)	$A \rightarrow B, A \rightarrow C \twoheadrightarrow A \rightarrow BC$	(Vereinigungsregel)
(5)	$A \rightarrow B, BC \rightarrow D \twoheadrightarrow AC \rightarrow D$	(Pseudotransitivität)
(6)	$A \rightarrow B, C \subseteq B \twoheadrightarrow A \rightarrow C$	(Zerlegungsregel)

Anderer Beweis von Eigenschaften (4)-(6)

Sentence

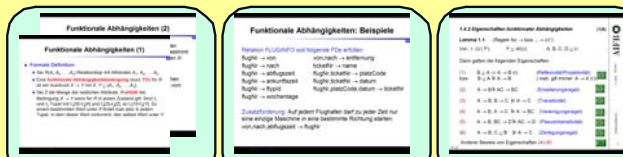
A Recursive Learning Module



Module „Dependencies“

Module

„ Functional Dependencies “



Abhängigkeiten

- Normalisierungskalkül beruht auf Abhängigkeiten zwischen Attributen als speziellen Konsistenzbedingungen.
- Varianten:
 - Funktionale Abhängigkeiten als Verallgemeinerung von Schlüsselbedingungen.
 - Mehrwertige Abhängigkeiten als weitere Verallgemeinerung funktionaler Abhängigkeiten.
 - Inklusionsabhängigkeiten als Verallgemeinerung von Fremdschlüsselbedingungen (spielen untergeordnete Rolle).
- Wichtig: alle Formen sind echte Konsistenzbedingungen, d.h. Anforderungen an spätere Relationsinstanzen, nicht zufällig erfüllte Eigenschaften.

Explanation

Reuse of Learning Material

- **Domain Engineering:**

Explicit support of the cooperative design and reuse

- Development for reuse (⇒ reusable modules)
- Community and domain specific ontology
- Aspect separation
- Goal: course families

- **Course Engineering:**

Reusable based course design

- Design with reuse
- Course configuration
- Course generation

A Course Structure

1 Normalization

1.1 Anomalies

Deletion Anomaly

Insertion Anomaly

Update Anomaly

1.2 Dependencies

Functional Dep

Full FunctionalDep

Key Attribute

Non - Key Attribute

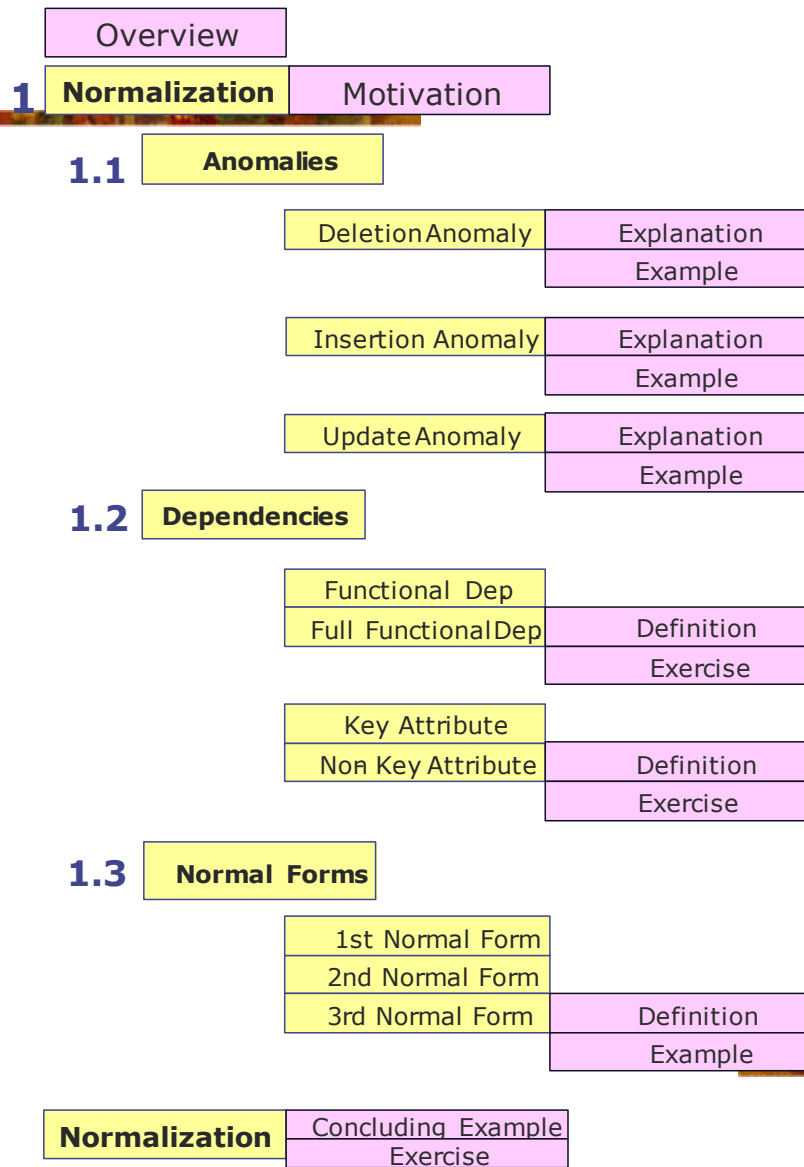
1.3 Normal Forms

1st Normal Form

2nd Normal Form

3rd Normal Form

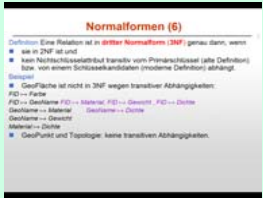
A Complete Course Structure



Course Configuration


Found Atoms

Course Structure



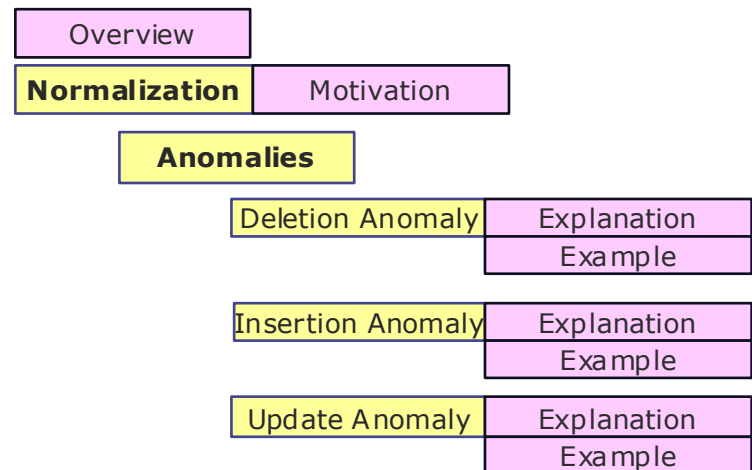
3rd Normal Form
Definition

...



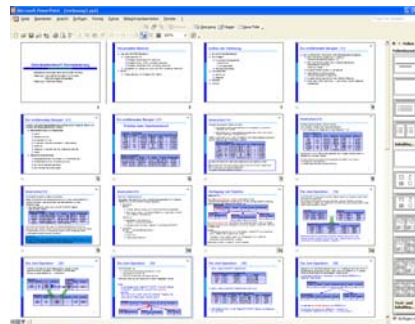
3rd Normal Form
Example

+



Generator

Course



Conclusions (1)

- Basic Problem:
 - Courseware development is extremely costly and time consuming
- Desirable:
 - Reusable courseware, i.e. create courseware in a way that allows to use parts of it in other contexts, for other audiences and by other educators
 - this is prevented by monolithic courses
 - that do not separate contents from structure and presentation
 - do not identify semantic units of teaching

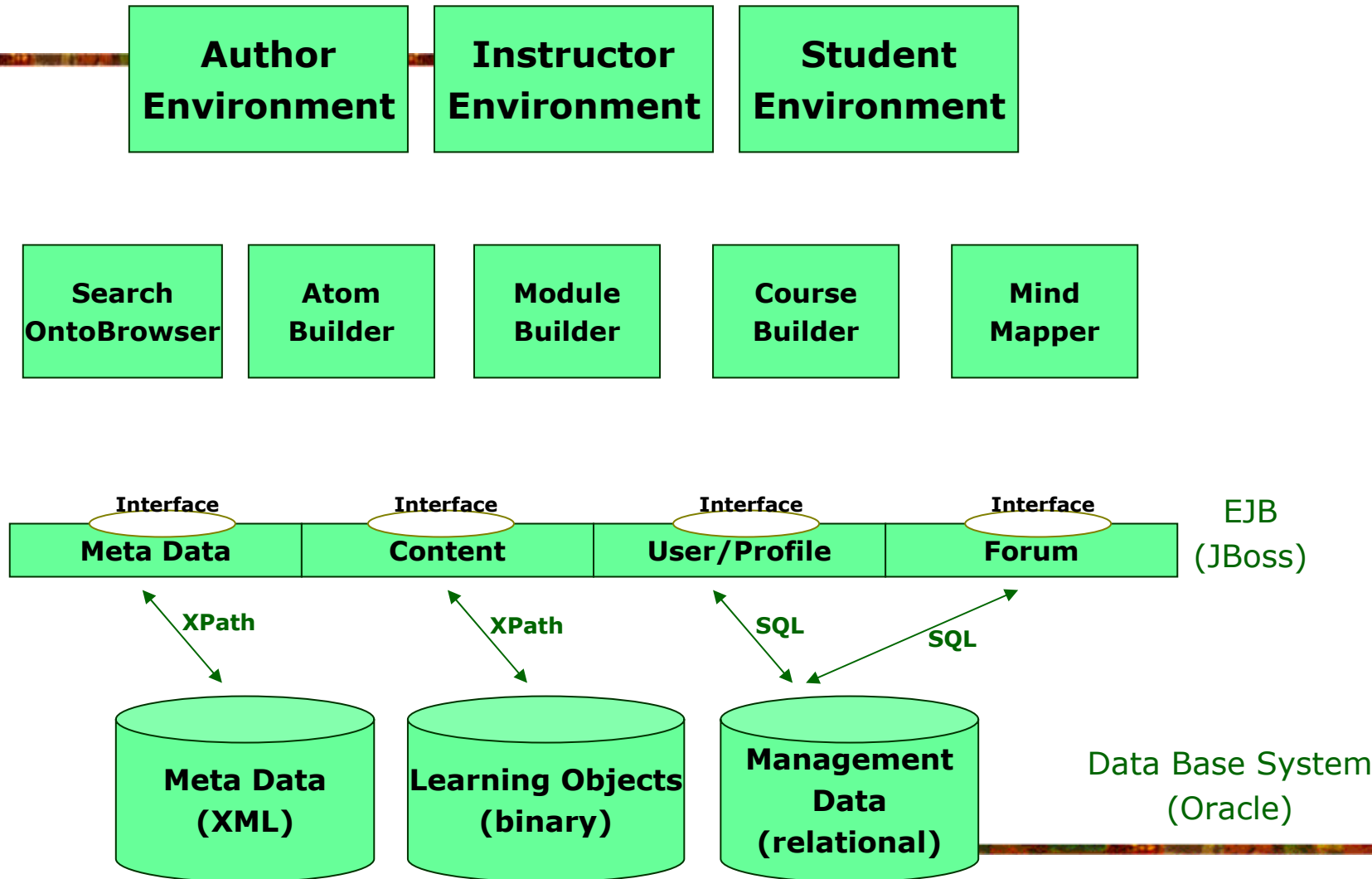
Conclusions (2)

- Our application domain: Database Courses
- Our approach:
 - divide contents into smallest semantic units
 - combine related units to form modules
 - separate different aspects, in particular: contents, structure, presentation
 - provide tools to help
 - devise a course structure
 - find existing materials that cover the topics needed
 - structure contents according to individual needs
 - adapt material to presentation requirements.

Thank you!

Additional information can be found at
<http://www.ipd.uni-karlsruhe.de/SCORE>

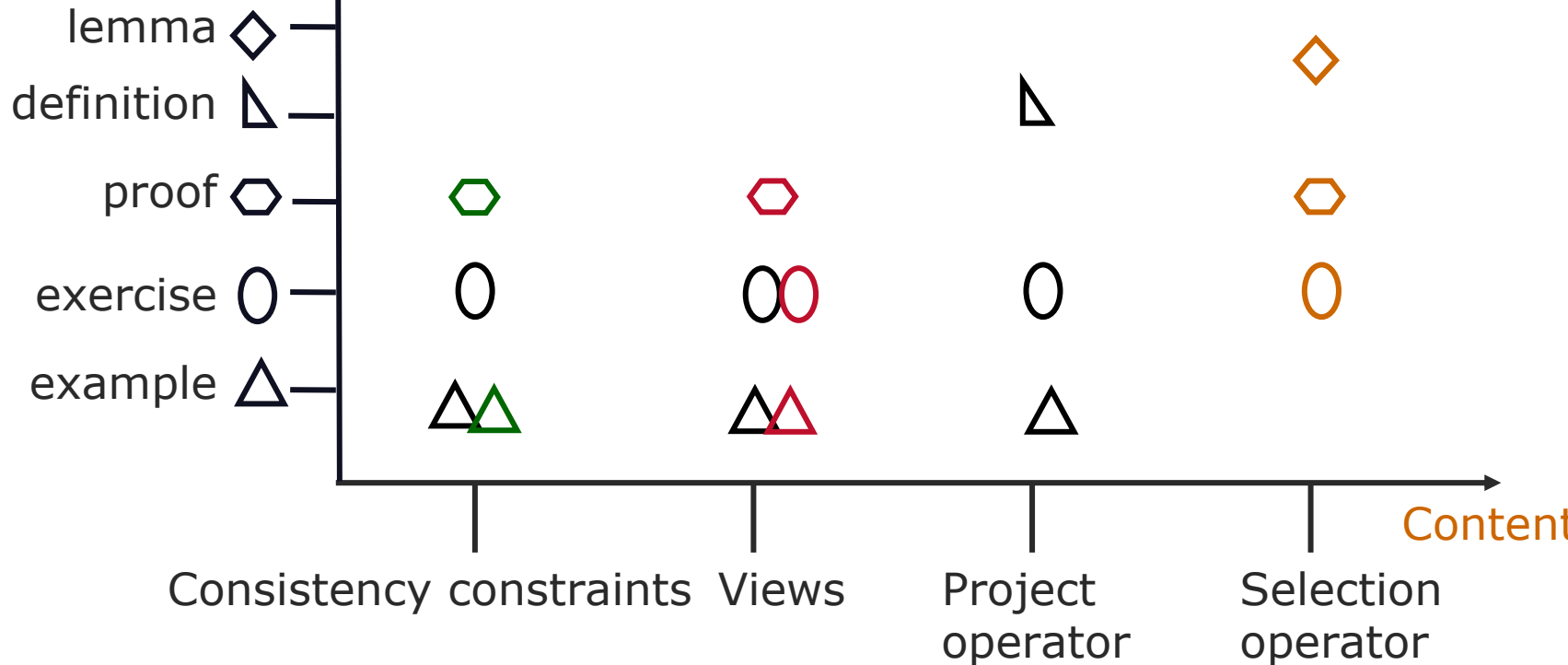
SCORE System Architecture



Aspects to Organize Learning Material

Didactic

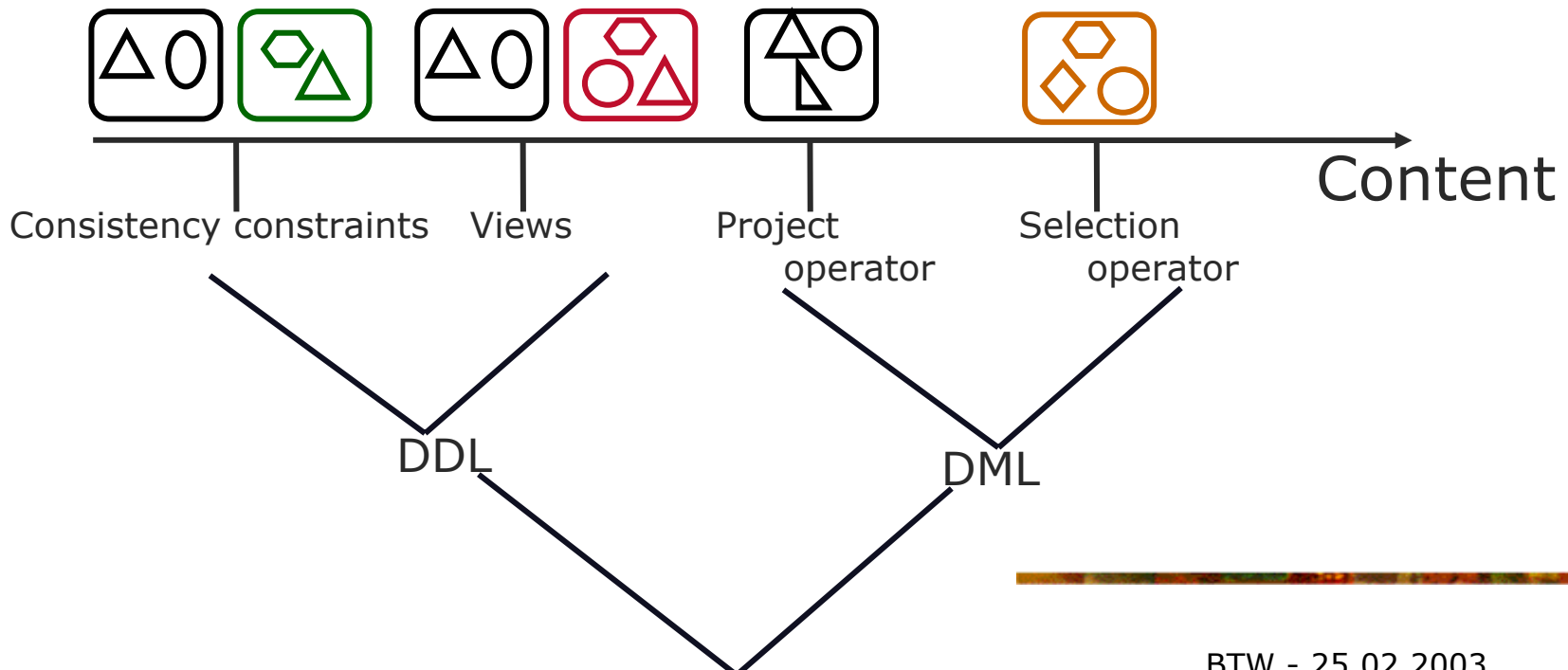
(material type)



→ other orthogonal aspects exist, e.g., layout (represented by different colors)

Problems for Reuse

- Modularization along the content
- Other aspects are mixed
→ Modular (re-)use is difficult



Aspects of Learning Material

- content
 - terms of the semantic ontology
- presentation
 - layout
 - notation
 - ...
- didactic
 - material type, e.g., motivation, example, definition
 - learning model
 - level of detail
 - ...