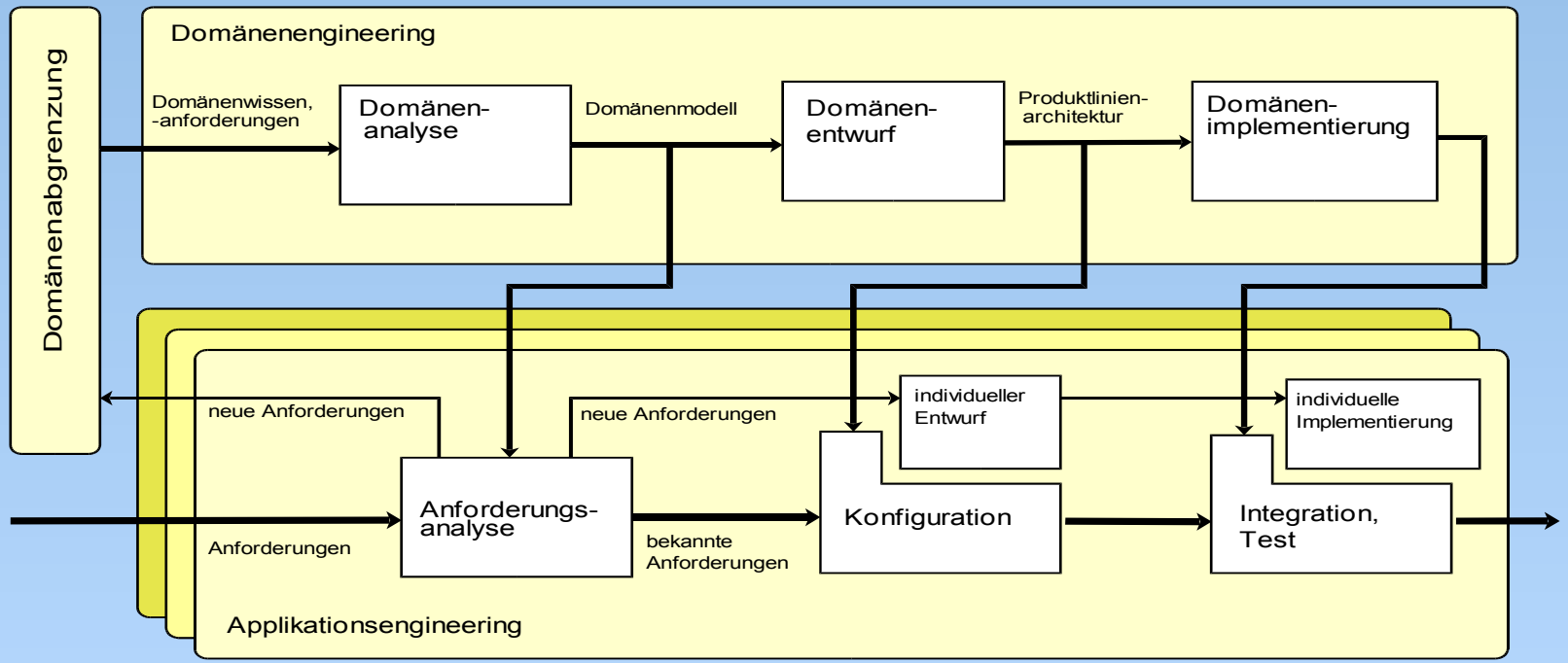
RepGen

- Eine Generierungsplattform für Versionierungsdienste

Christian Gebauer <gebauer@gmx.com>

Software-Produktlinien

- *domänenbasierter* Softwareentwicklungsansatz
- SW-Produktlinie: Menge von Systemen mit:
  - ähnlichen Anforderungen
  - ähnlichen Implementierungen
  - gemeinsamer *Softwarearchitektur*
- effektivere *Wiederverwendung*

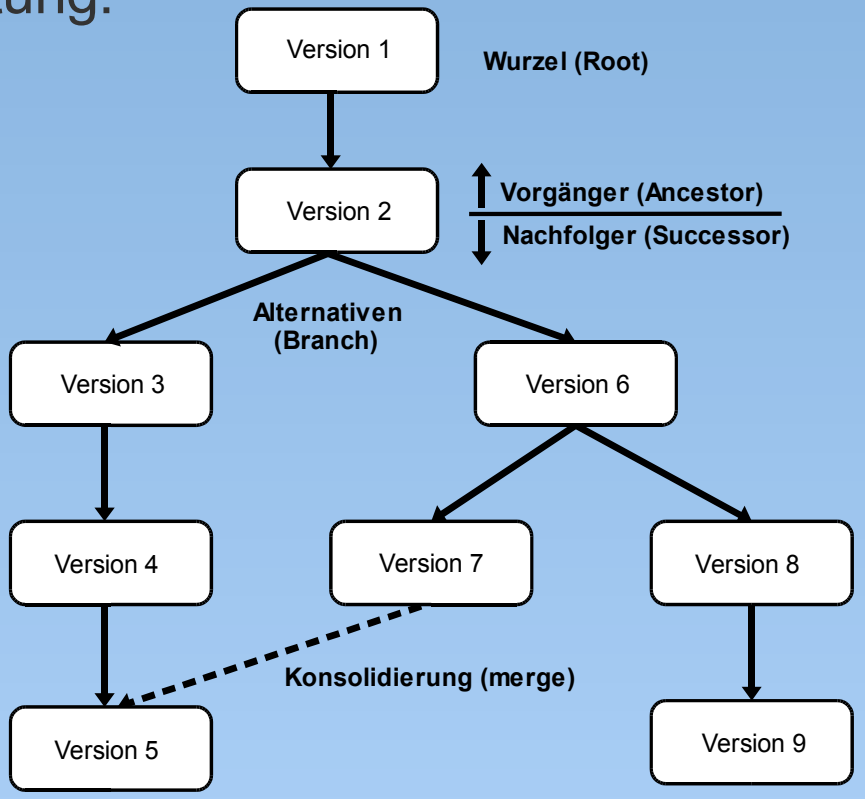


Gliederung

- **Versionierungssysteme**
- Domänenanalyse
- Spezifikationssprache
- Softwarearchitektur
- RepGen
- Softwaremetriken
- Leistungsuntersuchungen

Versionierungssysteme

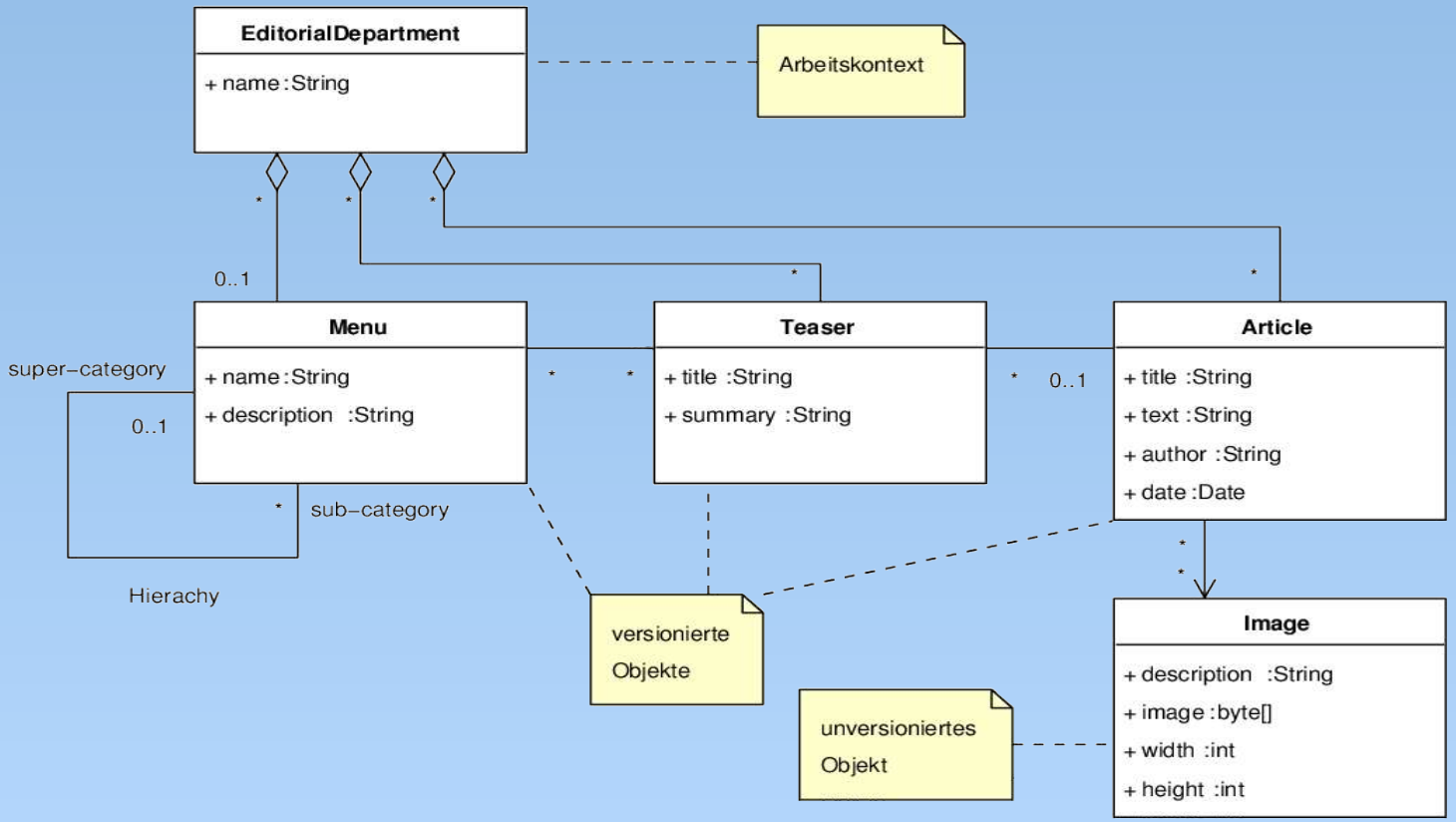
- versionierte Datenverwaltung:
  - Speicherung von Zwischenständen
  - Speicherung von Alternativen
  - Navigation im Versionsgraph



- Anwendungsbeispiele:
  - Engineering-Repositories
  - Content-Management-Systeme

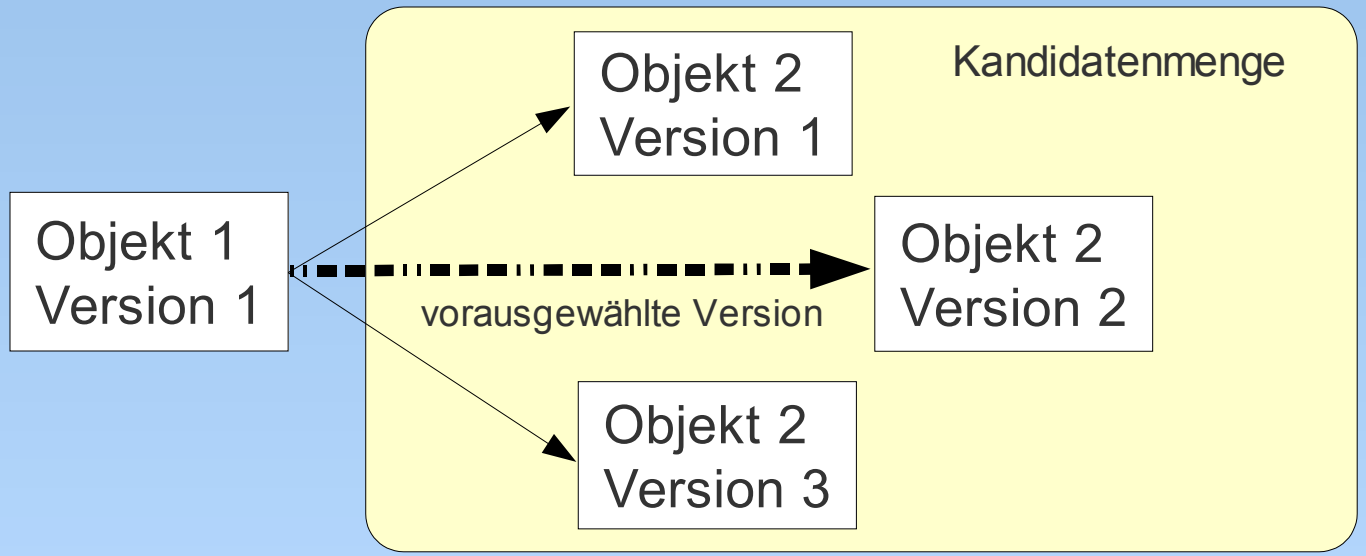
Objekt-orientierte Versionierungssysteme

- Informationsmodell
  - Objekttypen, versioniert und unversioniert
  - Arbeitskontexttypen (Konfigurationsverwaltung)
  - Beziehungstypen



Verwaltung von Objektbeziehungen

- viele Variationen
- komplexe Semantik:
  - *gleitende* Beziehungsenden
    - Verwaltung der *Kandidatenmengen*
    - Vorauswahl von Objektversionen (Pinning)
    - regelbasierte Auswahl
  - Propagierung von Operationen

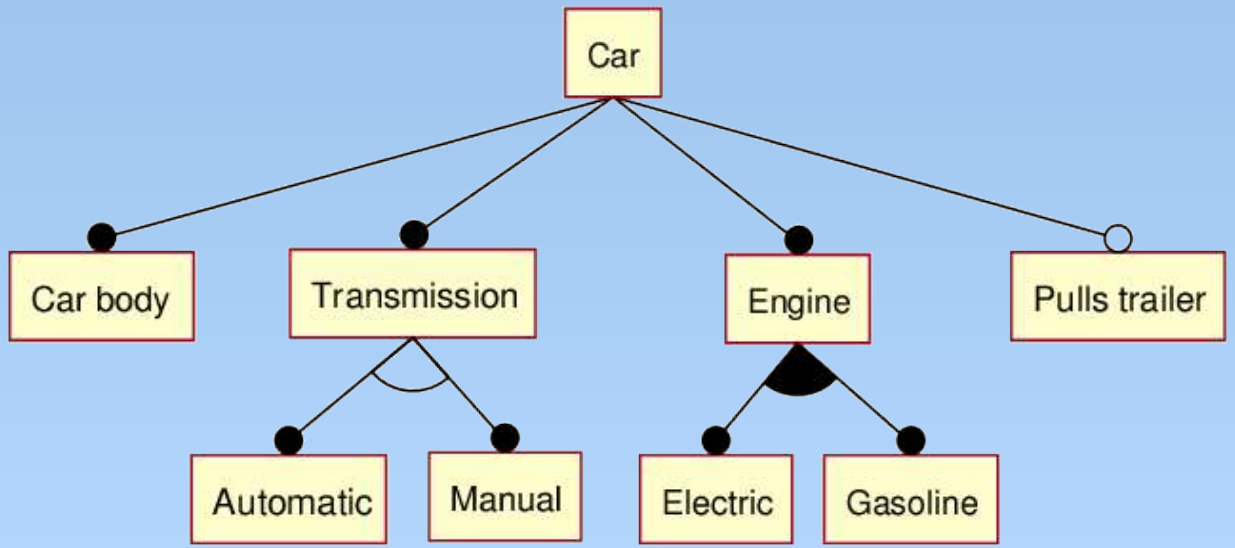


Gliederung

- Versionierungssysteme
- **Domänenanalyse**
- Spezifikationssprache
- Softwarearchitektur
- RepGen
- Softwaremetriken
- Leistungsuntersuchungen

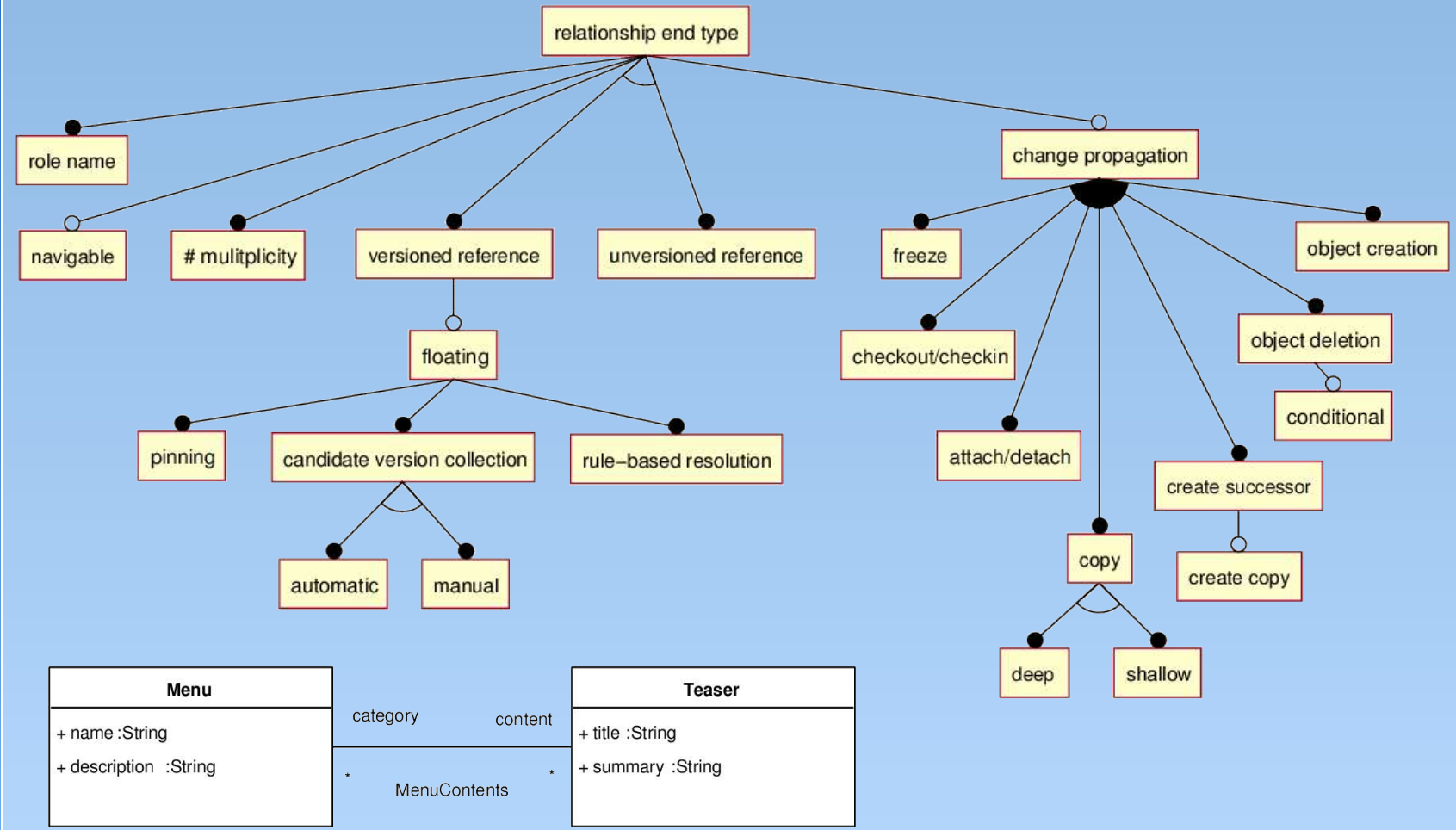
Merkmalanalyse

- Teil der Domänenanalyse (FODA)
- Analyse der Domänenkonzepte
  - Aufgliederung in *Merkmale*
  - Unterscheidung von vorgeschriebenen und optionalen Merkmalen
  - Darstellung der Abhängigkeiten in Merkmaldiagrammen
  - Kompositionsregeln, Merkmaldefinitionen, Auswahlrichtlinien
  - **keine** Berücksichtigung von strukturellen Eigenschaften



Merkmalanalyse der Beziehungsenden

- resultierendes Merkmaldiagramm für das Konzept „Beziehungsendtyp“



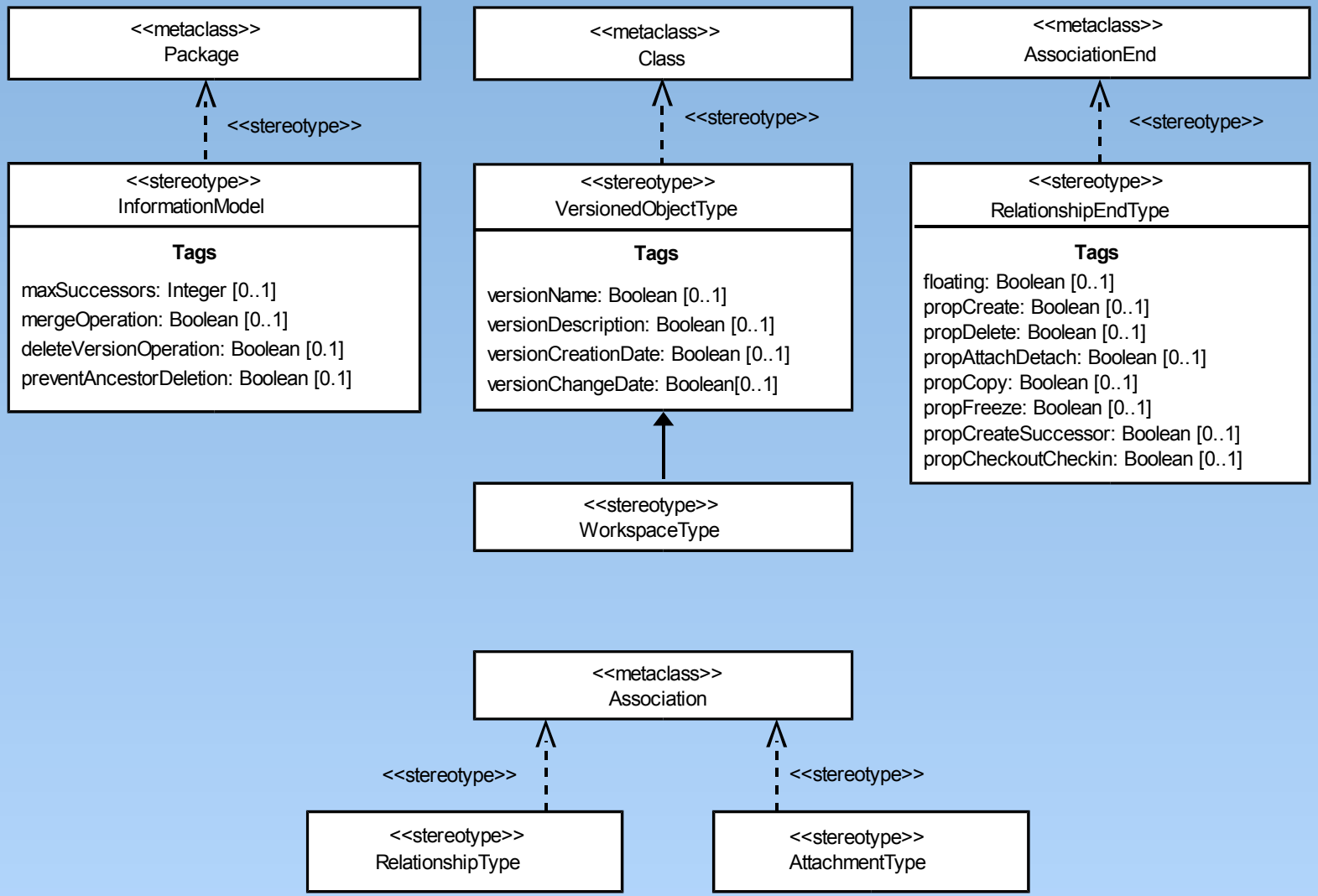
Gliederung

- Versionierungssysteme
- Domänenanalyse
- **Spezifikationssprache**
- Softwarearchitektur
- RepGen
- Softwaremetriken
- Leistungsuntersuchungen

Spezifikationssprache

- Festlegung der *Konfiguration* eines Versionierungssystems
- *plattformunabhängige*, *domänenspezifische* Darstellung erhöht Abstraktionsgrad:
  - keine Implementierungsaspekte
  - Konzentration auf variable Systemanteile
  - kompakte Darstellung
- UML als Basis
- Erweiterung der UML-Semantik mit einem *UML-Profil:*
  - Stereotypes
  - Tagged Values

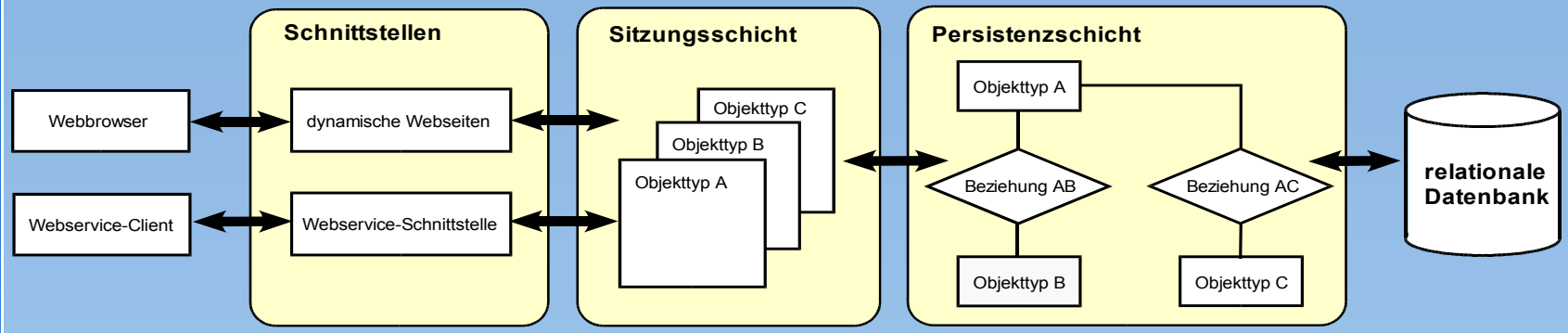
UML-Profil für Versionierungssysteme



Gliederung

- Versionierungssysteme
- Domänenanalyse
- Spezifikationssprache
- **Softwarearchitektur**
- RepGen
- Softwaremetriken
- Leistungsuntersuchungen

Softwarearchitektur



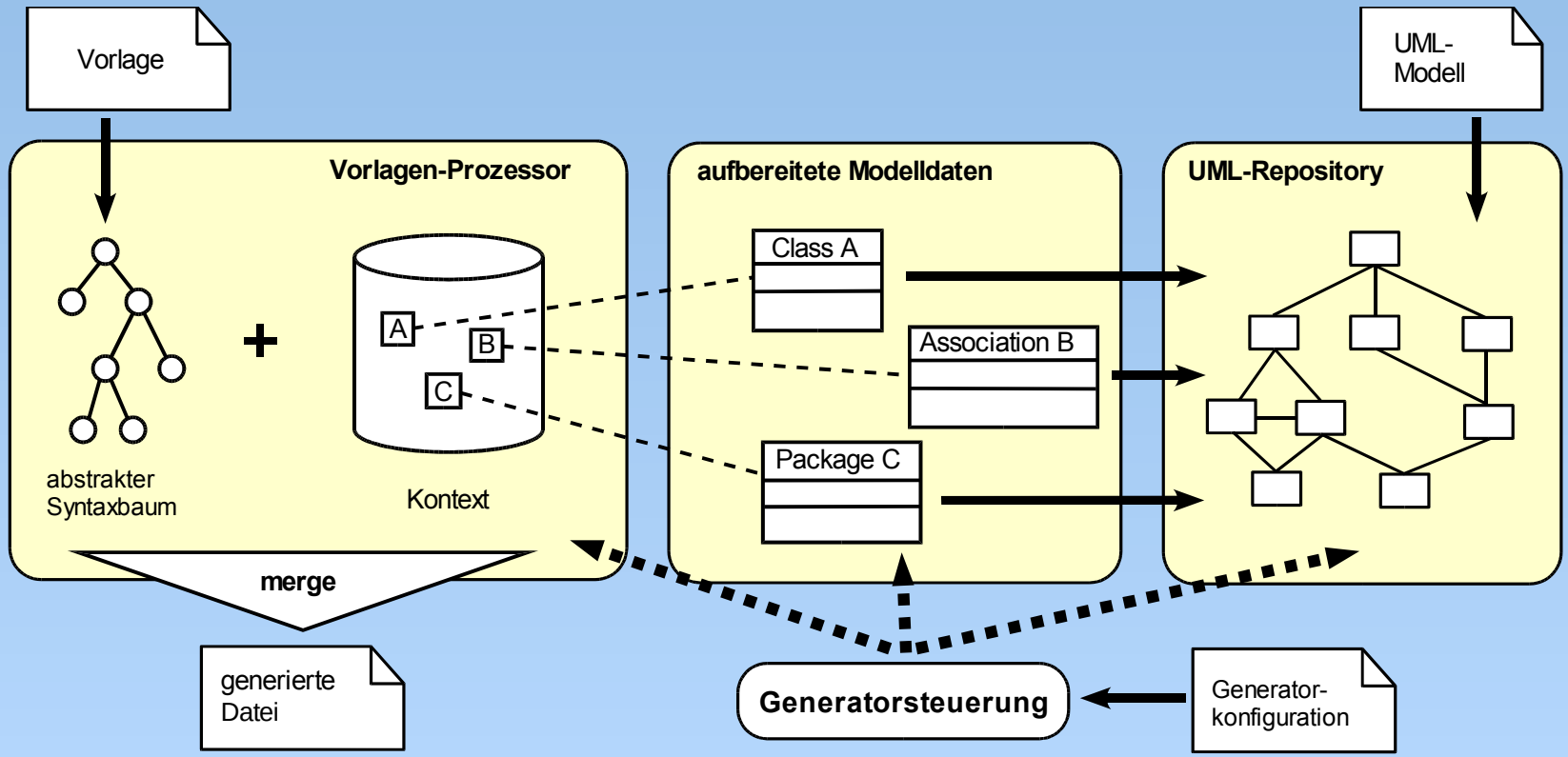
- Middleware-basierte Implementierung
- Persistenzschicht:
  - Entity-Komponenten
  - Objektorientierte Zugriffsschicht
  - effiziente Daten-Zwischenspeicherung durch den Anwendungserver
- Sitzungsschicht:
  - Sitzungskomponenten
  - Abbildung auf prozedurale Schnittstelle
  - Verwaltung des Sitzungszustands, insb. Transaktionen

Gliederung

- Versionierungssysteme
- Domänenanalyse
- Spezifikationssprache
- Softwarearchitektur
- **RepGen**
- Softwaremetriken
- Leistungsuntersuchungen

Vorlagenbasierte Generierung

- Vorlagen enhalten:
  - statische Anteile (z.B. Java-, HTML-Code)
  - dynamische Anteile (Metaprogramm)
    - Referenzen auf Modelldaten
    - Kontrollstrukturen
- Trennung von Steuerung, Datenaufbereitung und Darstellung



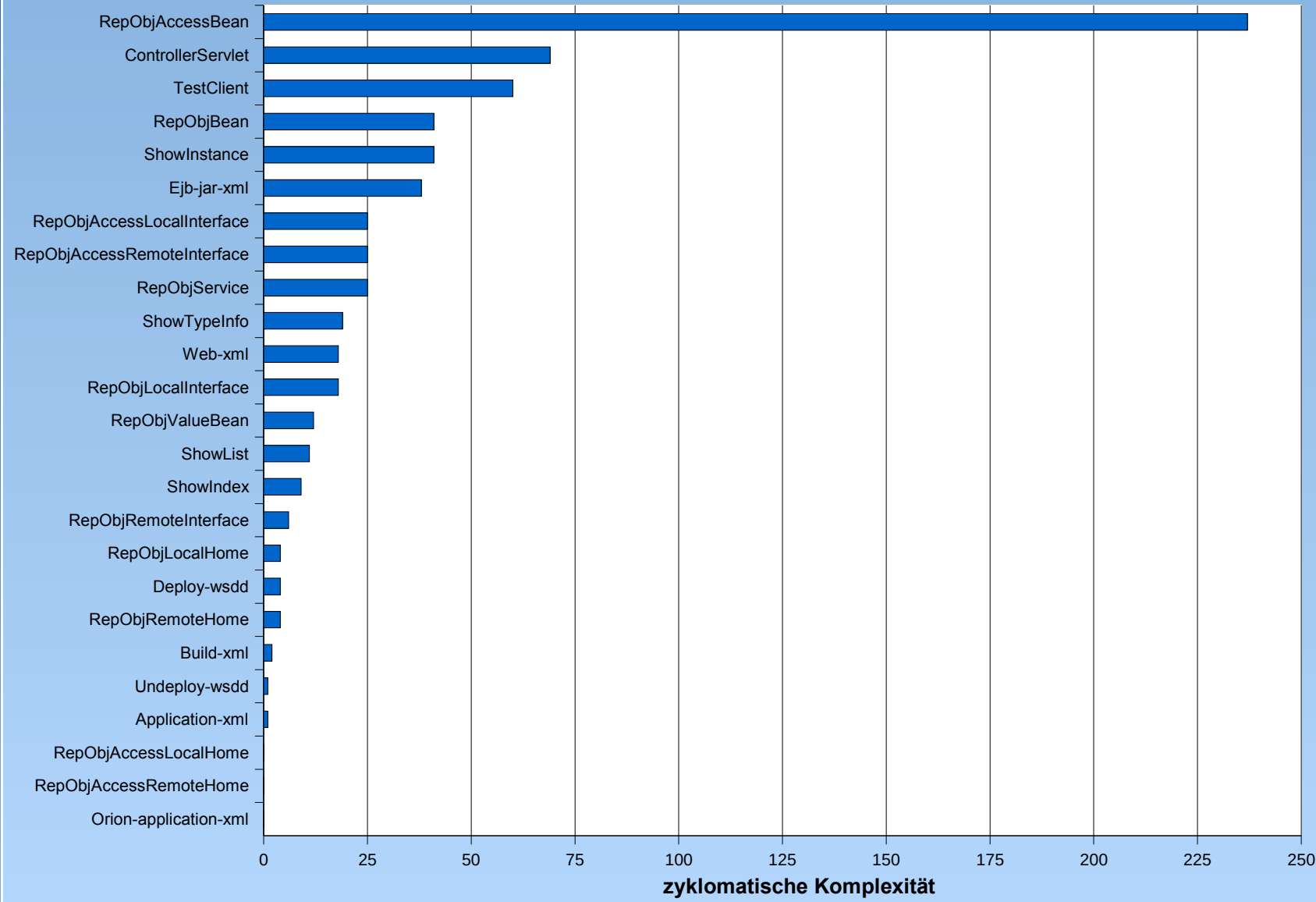
Gliederung

- Versionierungssysteme
- Domänenanalyse
- Spezifikationssprache
- Softwarearchitektur
- RepGen
- **Softwaremetriken**
- Leistungsuntersuchungen

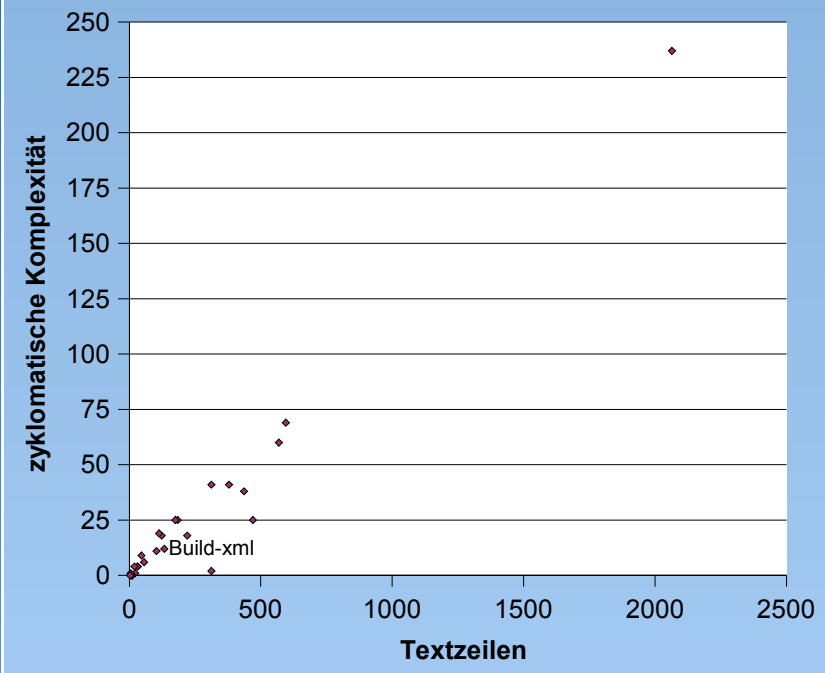
Bewertung durch Softwaremetriken

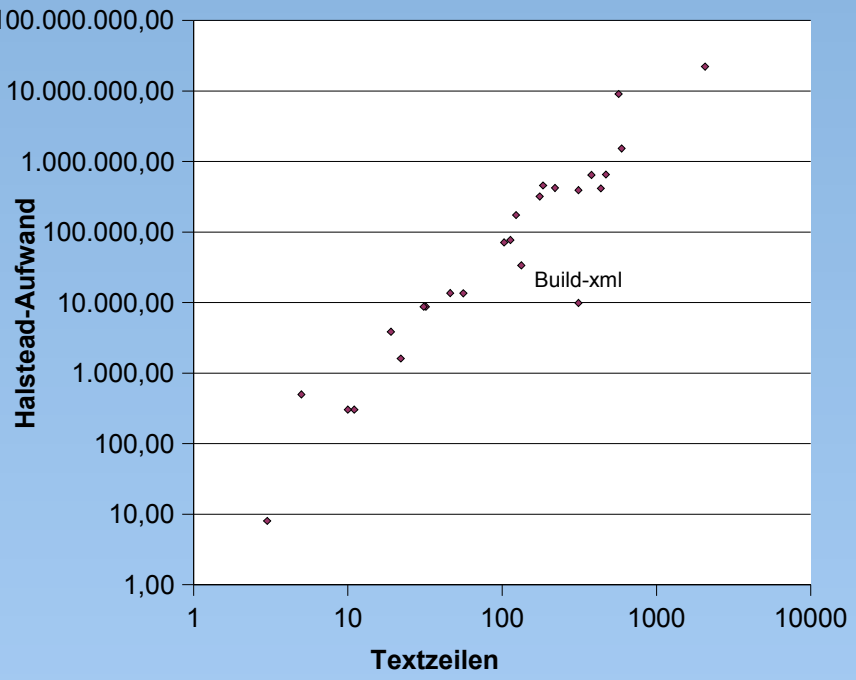
- Softwaremetriken
  - Ableitung von Messwerten aus dem Quellcode bzw. der Spezifikation eines Systemes
  - Aussagen über Aufwand, Wartbarkeit, Risiko, ...
- verwendete Metriken:
  - Anweisungsanzahl
  - zyklomatische Komplexität
  - Halstead-Aufwand
- durchgeführte Untersuchungen:
  - Beurteilung der Qualität der Generatorvorlagen
  - Ermittlung der erreichten Wiederverwendungsfaktoren
  - quantitative Analyse der generierten Softwaresysteme

Qualität der Generatorvorlagen



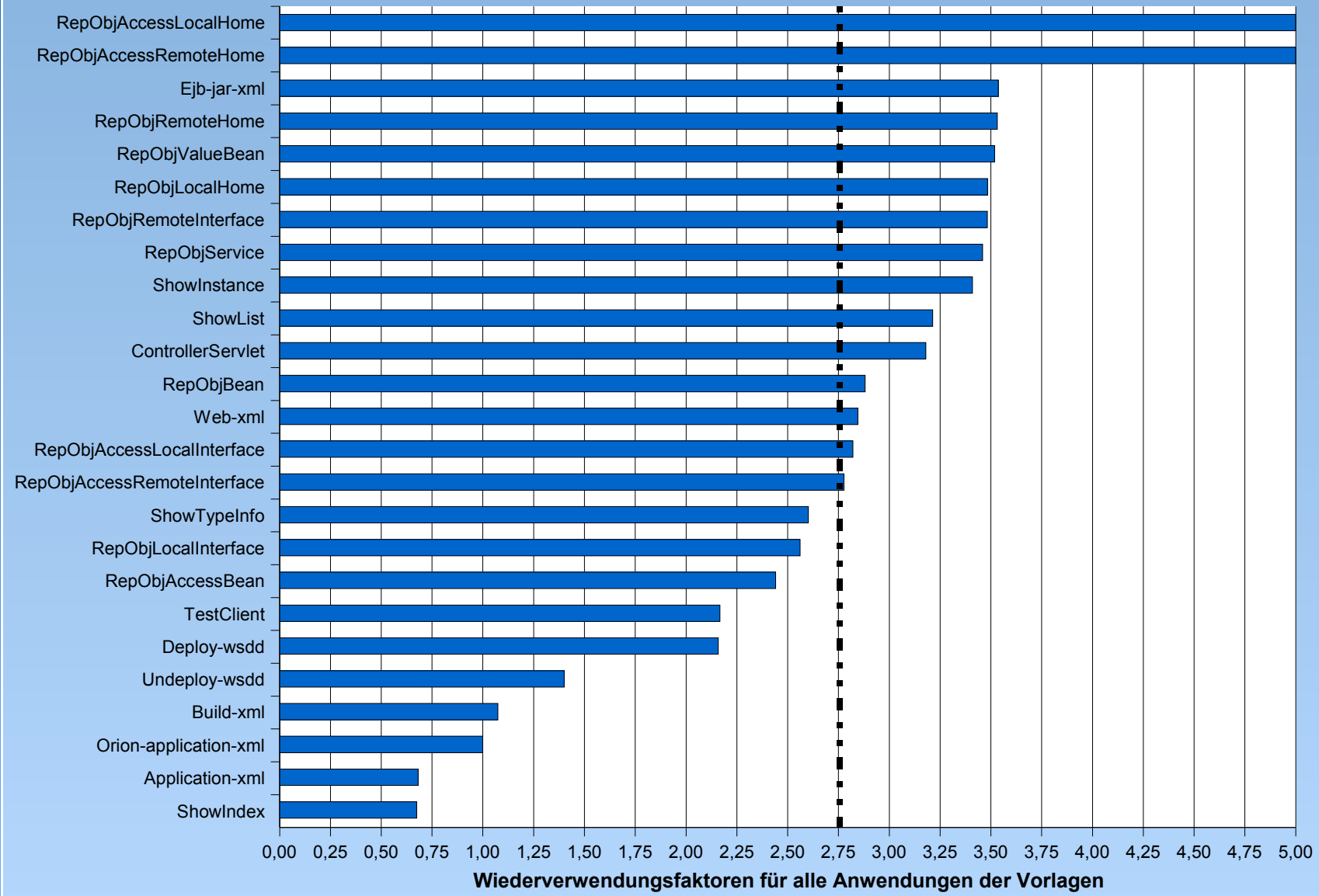
Qualität der Generatorvorlagen





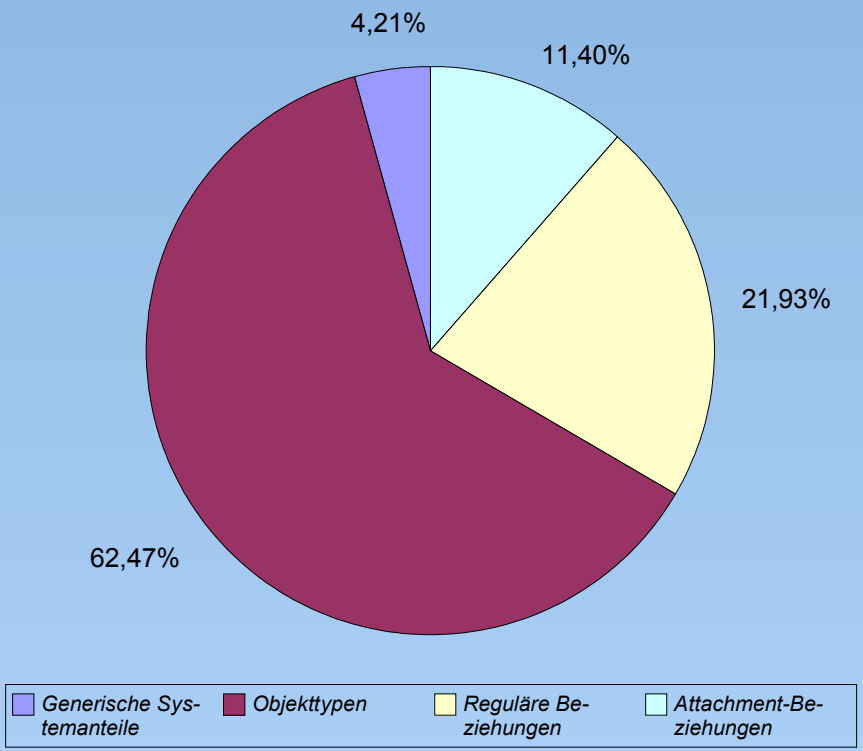
- Fazit:
  - Umfang und Komplexität ungleich verteilt
  - konstantes Verhältnis zwischen Umfang und Komplexität
  - einige Vorlagen zu komplex
  - Aufteilung der Vorlagen sinnvoll

Wiederverwendungsfaktoren



Zusammensetzung der CMS-Implementierung

Systembestandteil	Textzeilen	Prozentualer Anteil
Generische Systemanteile	750	4,21%
Objekttypen	11139	62,47%
Article	2559	14,35%
EditorialDepartment	2372	13,30%
Image	1583	8,88%
Menu	2325	13,04%
Teaser	2300	12,90%
Reguläre Beziehungen	3910	21,93%
Hierachy	1149	6,44%
MenuContents	1268	7,11%
TeaserArticle	1178	6,61%
Illustration	315	1,77%
Attachment-Beziehungen	2032	11,40%
MenuAttachment	657	3,68%
TeaserAttachment	674	3,78%

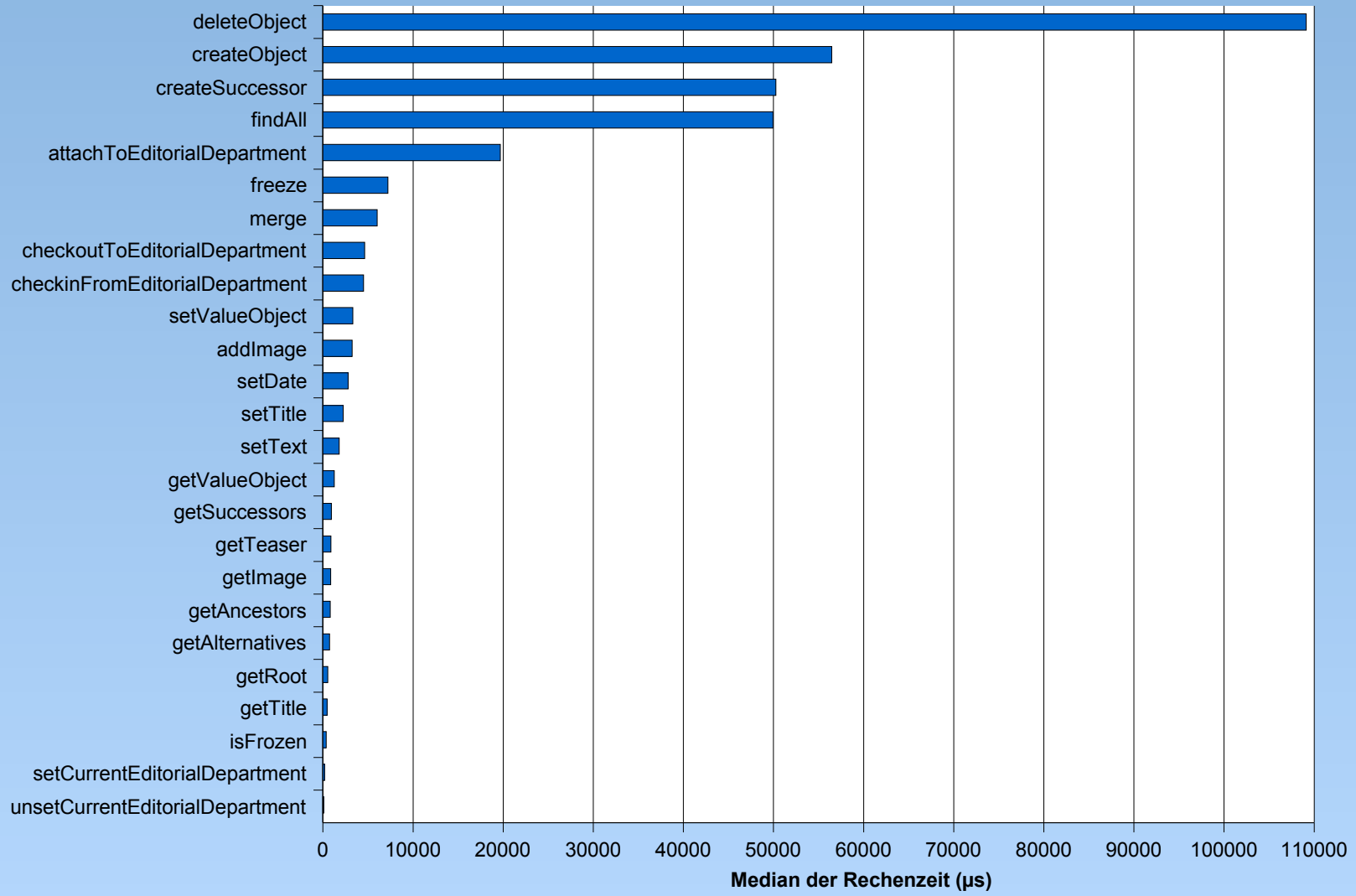


Gliederung

- Versionierungssysteme
- Domänenanalyse
- Spezifikationssprache
- Softwarearchitektur
- RepGen
- Softwaremetriken
- **Leistungsuntersuchungen**

Untersuchung der generierten Implementierung

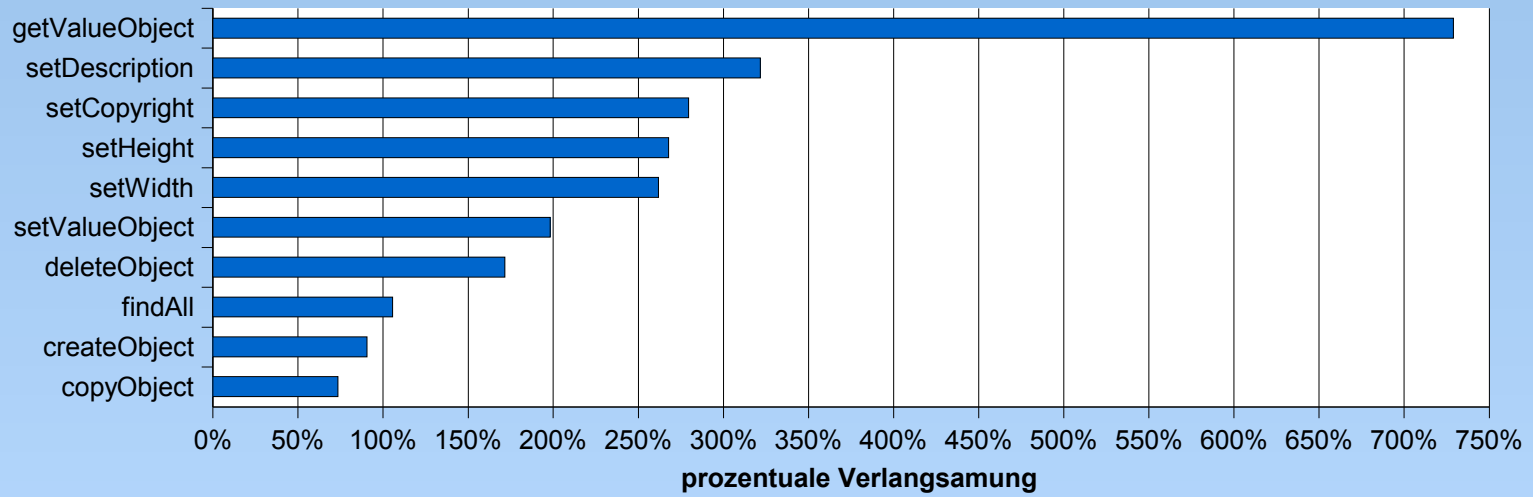
- Messwerte für den Objekttyp „Article“



Untersuchung der Webservice-Schnittstelle

- Messwerte für den Objekttyp „Image“

Operation	Aufrufdauer für nativen Client (µs)	Aufrufdauer für Webservice-Client (µs)	Verlangsamung	prozentuale Verlangsamung
findAll	10764,0	22128,0	11364,0	105,6%
setCopyright	2794,0	10602,5	7808,5	279,5%
setWidth	2975,5	10765,5	7790,0	261,8%
setHeight	2884,0	10606,5	7722,5	267,8%
setDescription	2399,5	10117,5	7718,0	321,7%
createObject	8130,5	15487,5	7357,0	90,5%
getValueObject	938,5	7780,0	6841,5	729,0%
setValueObject	3151,0	9393,5	6242,5	198,1%
copyObject	7879,5	13672,0	5792,5	73,5%



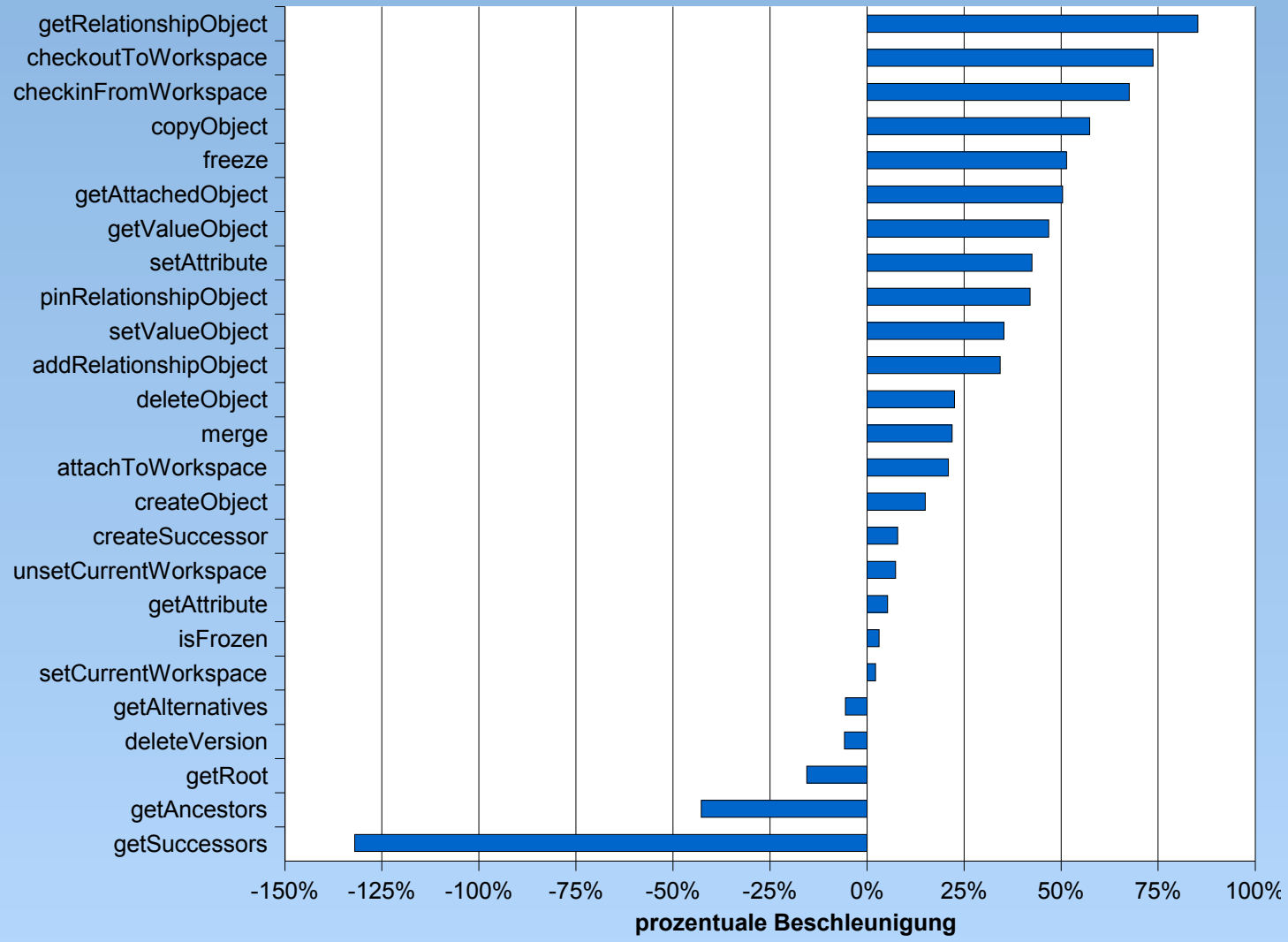
Eine generische Vergleichsimplementierung

- mittels eines generischen *Frameworks* realisiert
- Systemschnittstellen und Funktionalität entsprechen der generierten Implementierung
- die generische Implementierung ist mit den üblichen kommerziellen Lösungen vergleichbar

	generische Implementierung	**generierte Implementierung**
Verwaltung von Versions- und Beziehungsdaten	zentrale, generische Tabellen, keine Berücksichtigungen der Kardinalitäten	separate Speicherung, integiert in die Objekttabellen, Anpassung an die Kardinalitäten des Informationsmodells
unversionierte Objekttypen	nein	ja
nicht-gleitende Beziehungsenden	nein	ja
Operations-propagierung	dynamisch anhand von Konfigurationsdaten	durch fest vorgegebene Logik in den Sitzungskomponenten
Implementierung der Sitzungs-komponenten	Sitzungskomponenten erben Anwendungslogik von generischen Superklassen des Framework	Duplizierung und Anpassung der Anwendungslogik durch den Generator

Beschleunigung durch die Generierung

- Beschleunigung insgesamt: **26,6%**



the end

- Fragen?