Universität Leipzig

Institut für Informatik

COMA 3.0 - Ansätze für Matching, Datentransformation und Katalogintegration

Sabine Maßmann

Web Data Integration Workshop 2011

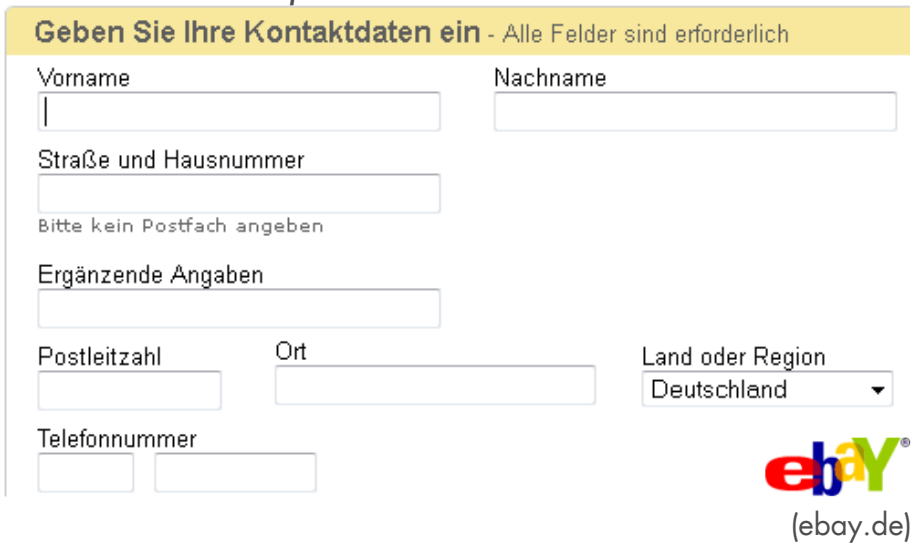
Agenda

- Motivation
- COMA 3.0
  - Matching
  - Datentransformation
  - Katalogintegration
- Ausblick

Motivation

- **Datenspeicherung** in Schemas und Einordnung in z.B. Kataloge
- z.T. **sehr verschieden** → Heterogenität in Ausdrücken, Sprachen und Konzepten
- für **Integration und Austausch von Daten** Wissen über zueinander gehörige Elemente bzw. Kategorien nötig
- **Größenordnung**: von einer Handvoll Elemente bis tausende Kategorien
- zudem regelmäßige **Veränderungen** z.B. zusätzliche Attribute → Anpassung nötig

- *Beispiele* *Schema* *Katalog*

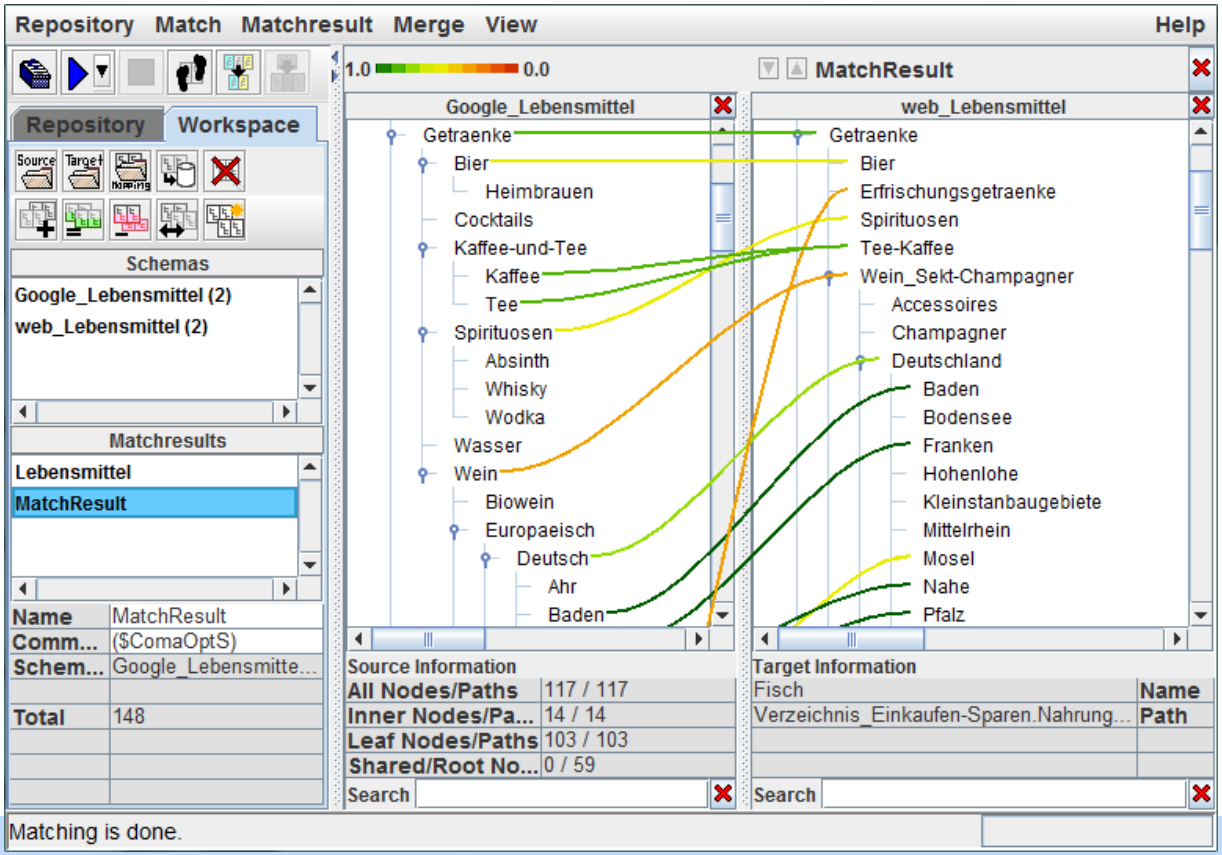


(ebay.de)

(amazon.de)

Unsere Lösung: COMA 3.0

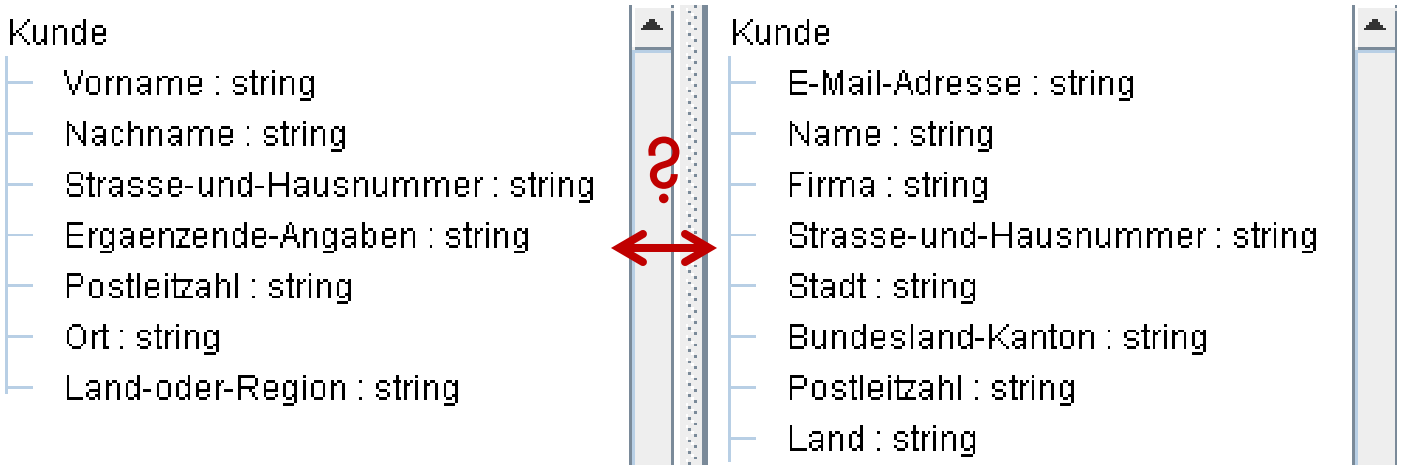
- Redesign & Weiterentwicklung des Prototyps COMA++
- Kernfunktionalitäten:
  - **Matching** von Schemas und Ontologien
  - **Katalogintegration**
  - Unterstützung der **Datentransformation**



Matching

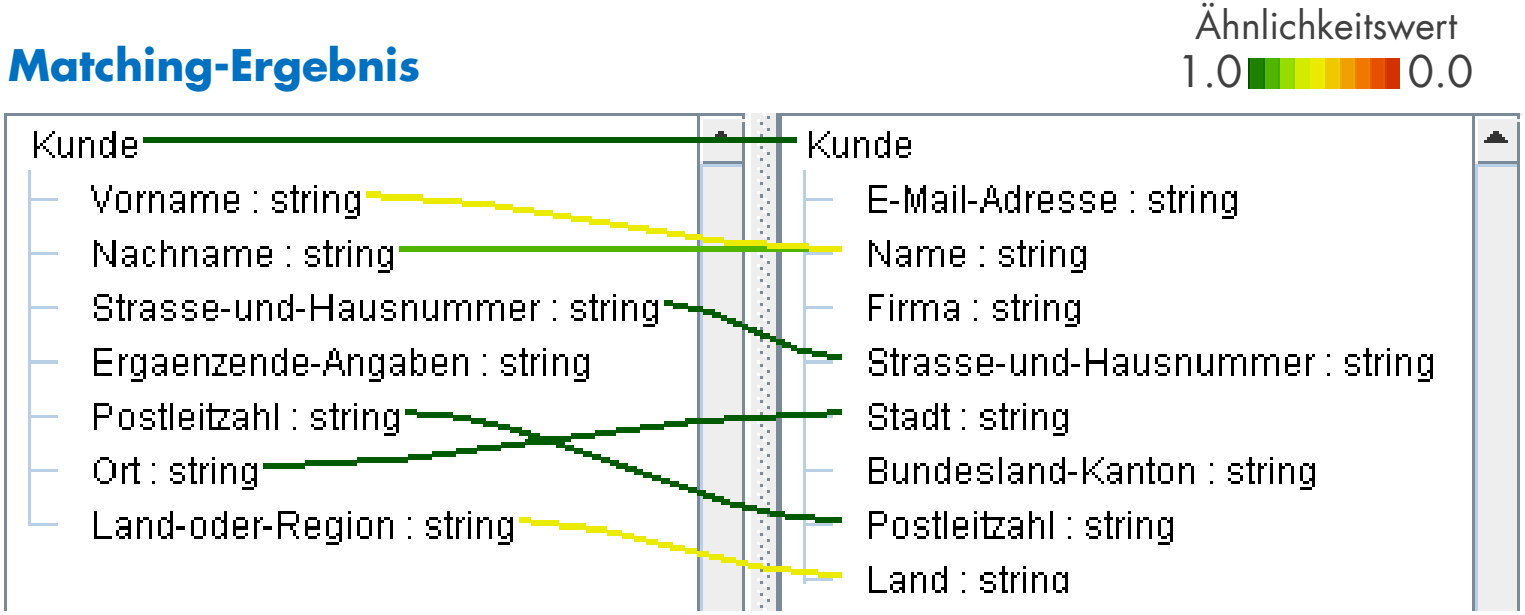
- **Auffinden** von gleichen bzw. einander entsprechenden Elementen in Schemas und Ontologien
- Automatischer Prozess
  - Ausnutzung der gegebenen **Information** z.B. Namen und Struktur
  - Implementierung verschiedener **Matchalgorithmen** z.B. für kurze Zeichenketten Trigram, für Dokumente TFIDF
  - Unterstützung mehrerer **Matchstrategien** z.B. für große Kataloge Zerlegung in Teile, Wiederverwendung früherer Ergebnisse für neue Versionen

Ausgangsschemas



Matching

- **Ähnlichkeitswert** als Hilfe bei evtl. Nachbearbeitung
- **Evaluierung** mit Datensets aus u.a. Bestellungen, Webverzeichnissen und Anatomie-Ontologie
- erfolgreiche **Standard-Konfiguration**



Datentransformation - Motivation

- Unternehmen haben individuelle **Datenbanken** für z.B.
  - Kunden
  - Produkte
  - Bestellungen

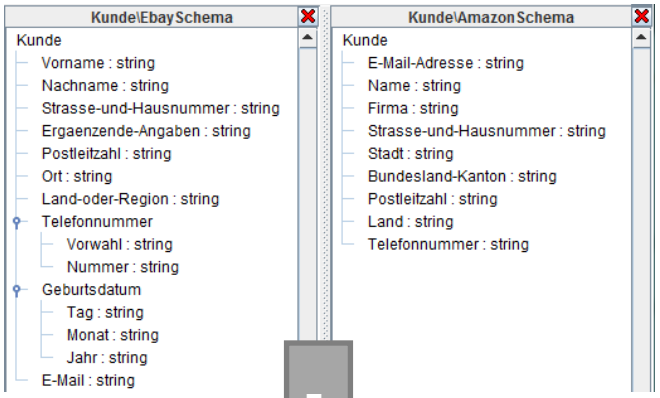
- trotz gleicher Anwendung **unterschiedlicher Aufbau** möglich, bei z.B. Kundendaten
  - Vorname und Nachname getrennt oder zusammen
  - Erhebung von Geburtsdatum oder auch nicht
  - Kontostand in verschiedenen Währungen angegeben

- **Transformation** nötig bei z.B. Kooperation, Fusion oder Informationsaustausch

Automatische Datentransformation

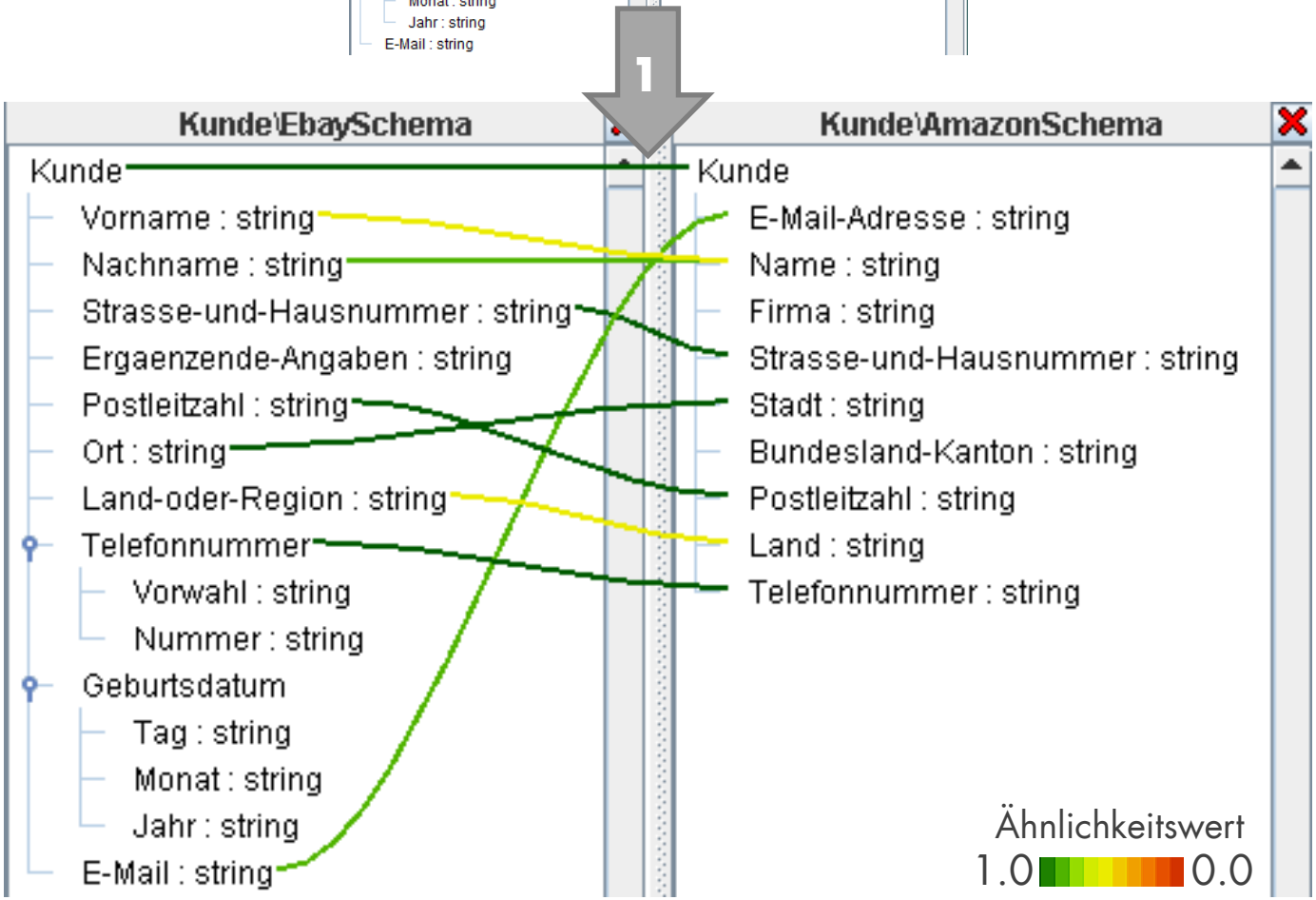
- Ein **Skript** (auch Query genannt) zur Datentransformation erstellen
- Dies entweder **ausführen** und Daten direkt transformieren oder z.B. in Applikationen **einbinden** (mehrmalige Nutzung)
- **Ziele** sind:
  - Information bleibt im Kontext
    z.B. Straße gehört zu Hausnummer
  - Unterstützung einfacher
    z.B. Postleitzahl=Postleitzahl
    und komplexer Beziehungen
    z.B. Vorname Nachname=Name
  - Vorschlag für Transformationsfunktion
    z.B. Preis in 1.3 · € = Preis in $

Beispiel: Matching

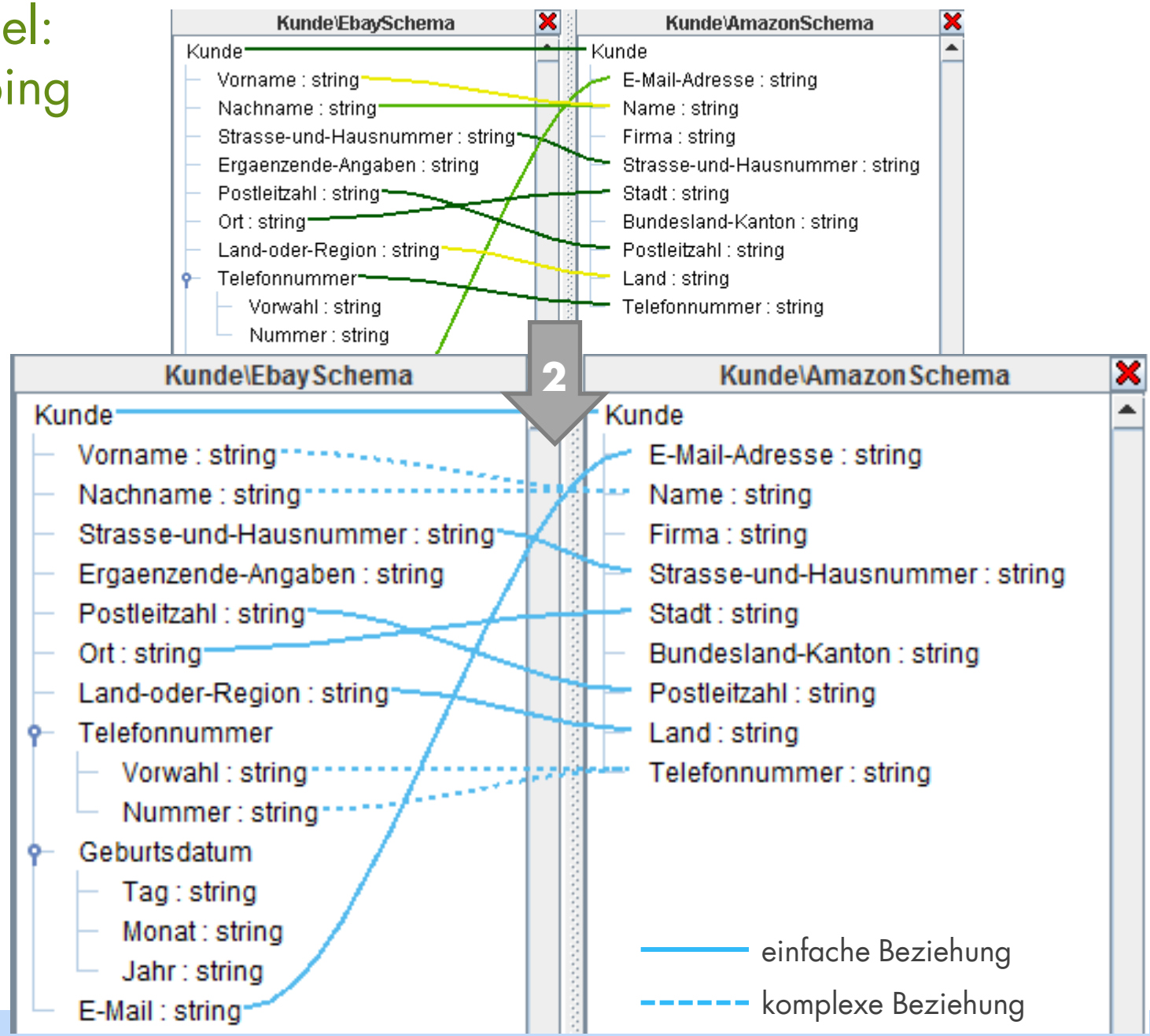


(ebay.de)

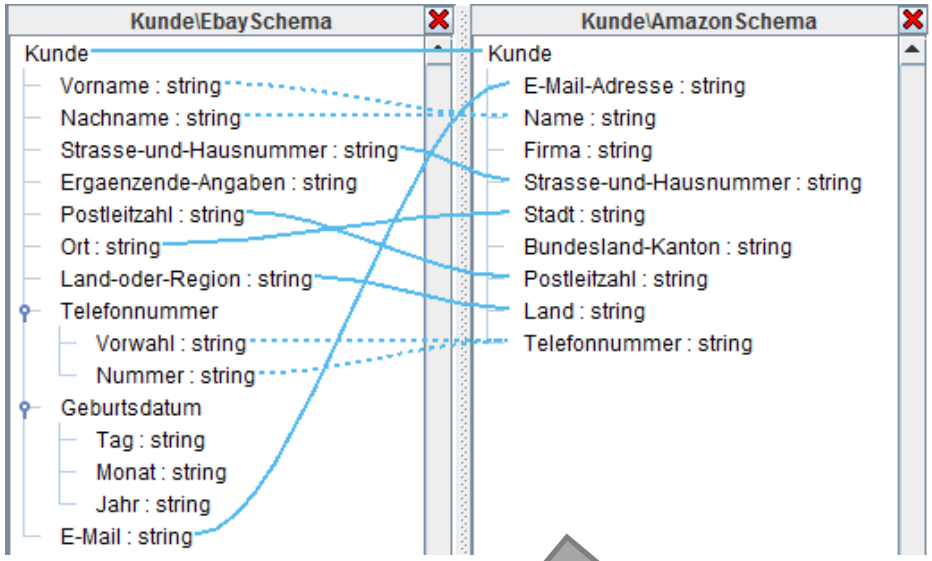
(amazon.de)



Beispiel: Mapping



Beispiel: Transformations-Skript




```
<Amazon> {
    for $Kunde in /Ebay/Kunde return
     <Kunde>
        <E-Mail-Adresse>{$Kunde/E-Mail/text()}</E-Mail-Adresse>
        <Name> {concat($Kunde/Vorname/text()," ",
          $Kunde/Nachname/text())}</Name>
        <Strasse-und-Hausnummer> $Kunde/Strasse-und-Hausnummer/text()}
          </Strasse-und-Hausnummer>
        <Stadt> {$Kunde/Ort/text()}</Stadt>
        <Postleitzahl>{$Kunde/Postleitzahl/text()}</Postleitzahl>
        <Land>{$Kunde/Land_oder_Region/text()}</Land>
        <Telefonnummer>{concat($Kunde/Telefonnummer/Vorwahl/text()," ",
          $Kunde/Telefonnummer/Nummer/text())} </Telefonnummer>
    </Kunde>}
</Amazon>
```

einfache Beziehung

komplexe Beziehung mit Transformationsfunktion

Katalogintegration - Motivation

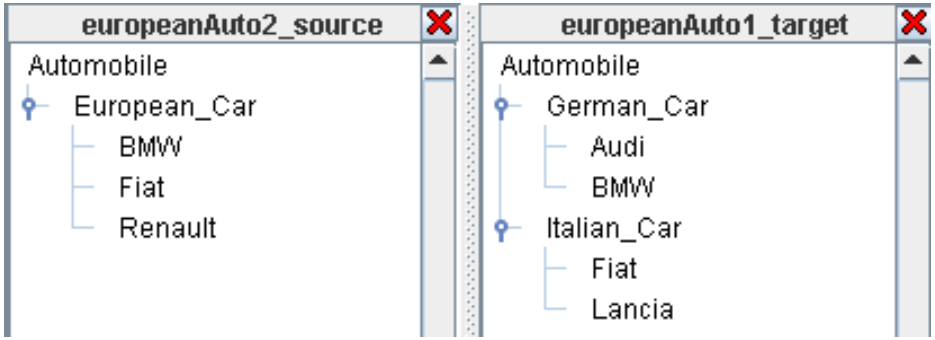
- **Portale und Shops** verwenden Produktkataloge
- **Produktkatalog**
  - systematisch sortierte Sammlung von Produkt- oder Service-Information
  - Klassifikation anhand gemeinsamer Merkmale z.B. Feature, Farbe

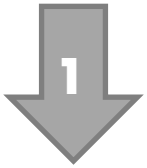
- Portale benötigen einen Produktkatalog, der alle Produkte von den Händlern umfasst → Abdeckung aller Produktkategorien

Automatisches Integrieren

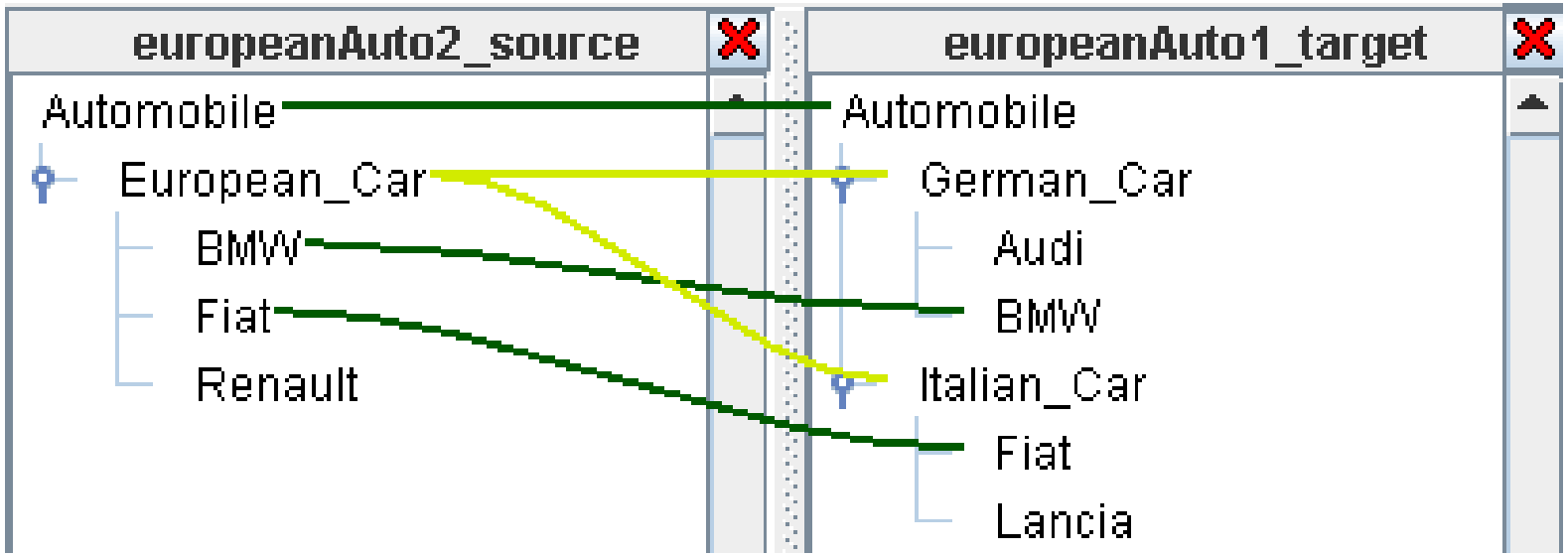
- einen **Katalog erstellen**, der die Kategorien von mehreren Ausgangskatalogen in sich vereinigt (Integrieren, Mergen)
- Algorithmus verarbeitet die Korrespondenzen aus dem Matching und **generiert Typen** anhand der Bezeichnungen (gleich, spezieller, allgemeiner)
- **Ziele** sind, dass :
  - Kategorien und Struktur des Zielkatalogs *beibehalten* werden
  - Kategorien, die in *beiden* Katalogen vorkommen, im Ergebnis nur *eine* Kategorie darstellen sollen
  - Kategorien, die neu hinzukommen, an richtiger Stelle *eingefügt* werden
  - Redundanzen *vermieden* werden (=„Aufblähen“ des Katalogs vermeiden)
- Optimierung für **große Datensets**

2. Beispiel: Matching

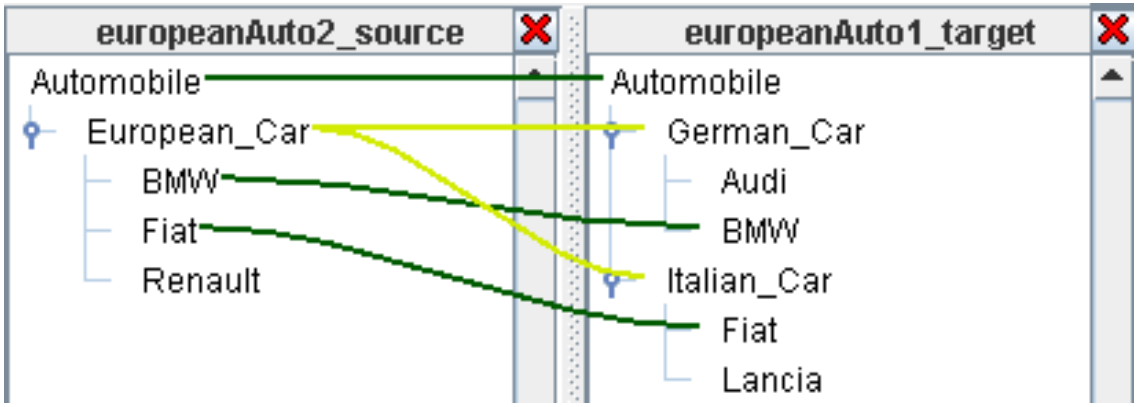




Ähnlichkeitswert
1.0 0.0



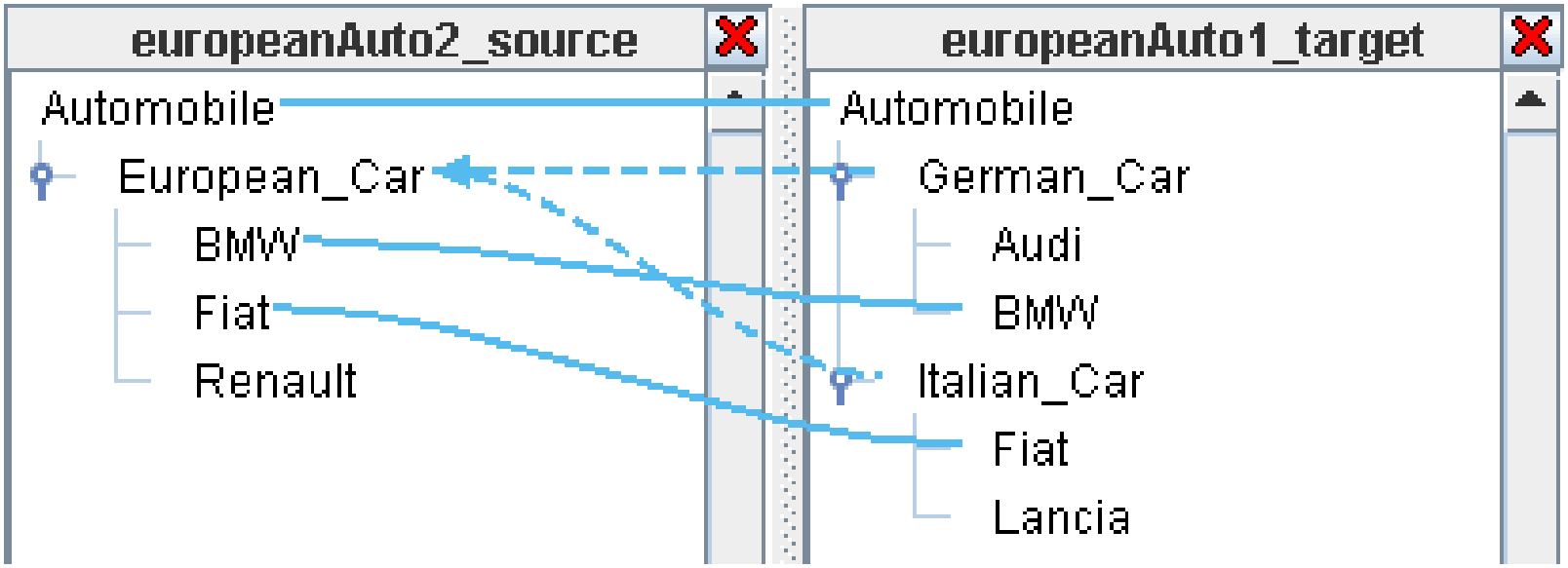
2. Beispiel: Mapping



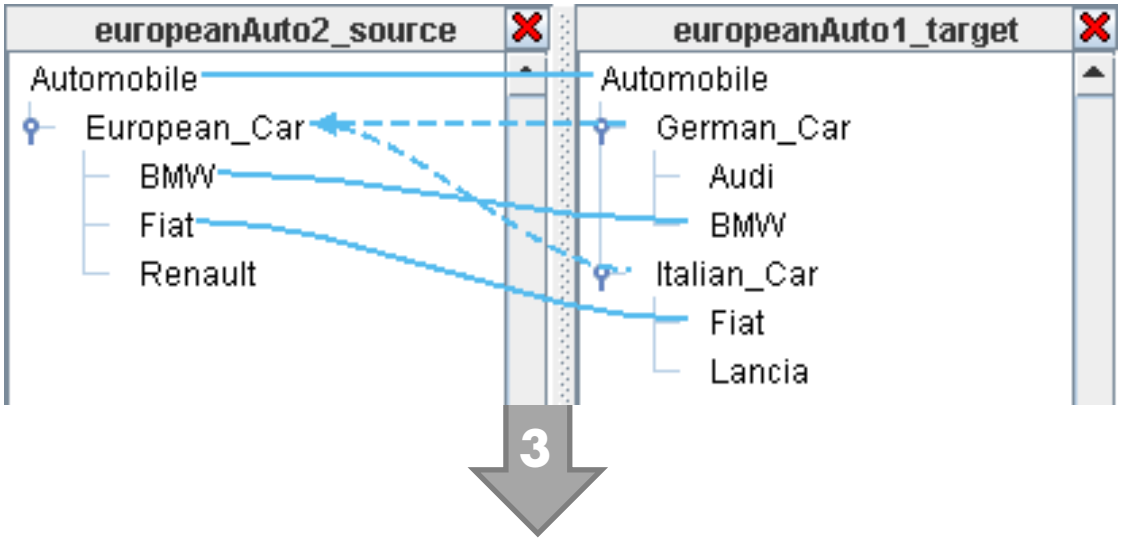
gleich

spezieller/Teilmenge

allgemeiner/Übermenge



2. Beispiel: Integrierter Katalog

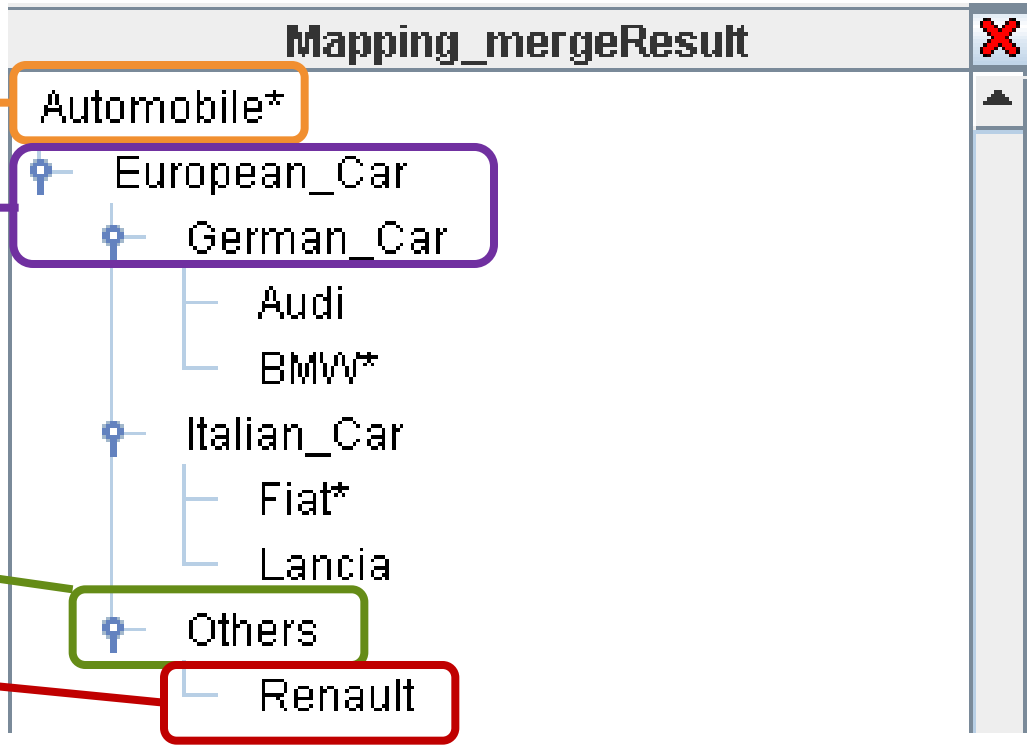


Kategorie in beiden Katalogen gleich

eine Kategorie ist Teilmenge der anderen

neue Kategorie

Kategorie in Hierarchie eingeordnet



Ausblick

- COMA 3.0 **Community Edition** verfügbar im Januar 2012
  - Open Source, kostenlos
  - Kernfunktionalität Matching
  - Nutzer können
    - experimentieren
    - vorhandene Matchalgorithmen und –strategien testen
    - neue Datenquellen anbinden
    - um eigene Algorithmen für z.B. Matching, Kombination und Selektion erweitern
    - Funktionalität in eigene Anwendungen einbinden (via API)

- **Software Suite** mit voller Funktionalität von COMA 3.0 in einem Modul
  - für Business-Anwendungen