An Extensible System for Merging Two Models

Rachel Pottinger
University of Washington

Supervisors: Phil Bernstein and Alon Halevy

Model Management

A **model management system** consists of three key abstractions:

- **Models** – a formal description of any complex structure that describes how data is organized (e.g., schemas, interface definitions)
- **Mappings** – which describe how two models are related
- **Operators** – which manipulate models and mappings as atomic objects. The main operators are:
  - **Match** – create a mapping between two given models
  - **Merge** – combine two given models based on a given mapping between them
  - **Apply** – apply a function to all of the items in a given model
  - **Compose** – given a mapping between models A and B and a mapping between models B and C, create the mapping between A and C
  - **Difference** – given two models and a mapping between them, return a model consisting of all items in the first model and not in the second model
  - **Import/Export** – translate a model to/from our representation

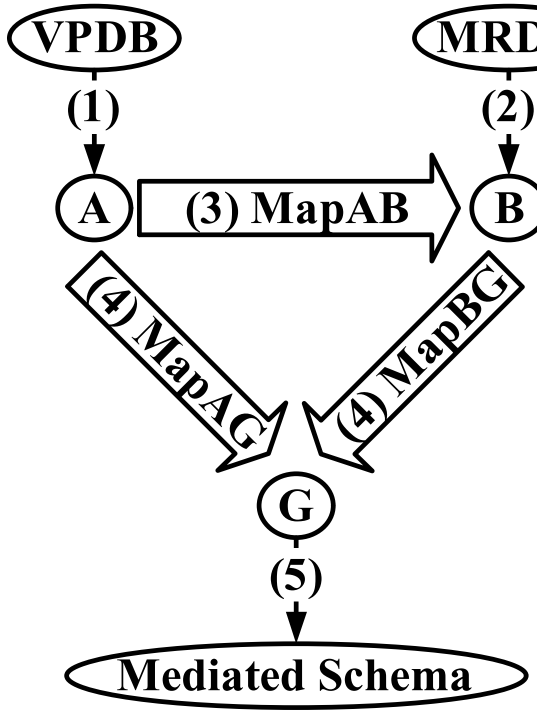
Data Integration Scenario

Video Place, which sells videos through its website, has purchased the Movie Review Database, a website that lists movie facts and reviews. Each company has its own database (VPDB and MRDB respectively) about movies and an application (the website) that runs on its database.

Process to create mediated schema

1. Import(VPDB)   Model A
2. Import(MRDB)   Model B
3. Match(A, B)   MapAB
4. Merge(A, B, MapAB)

   (Model G, MapAG, MapBG)
5. Export(G)   Mediated Schema



Merge

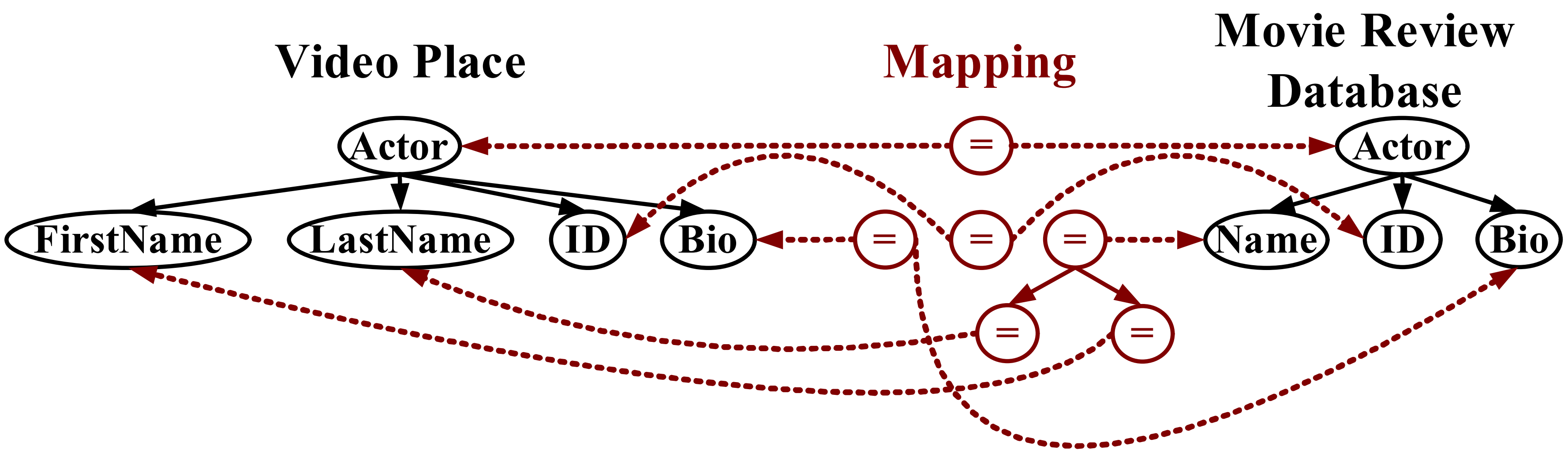
Goal:

Create a generic Merge operator useful for many different data models and schema management problems.

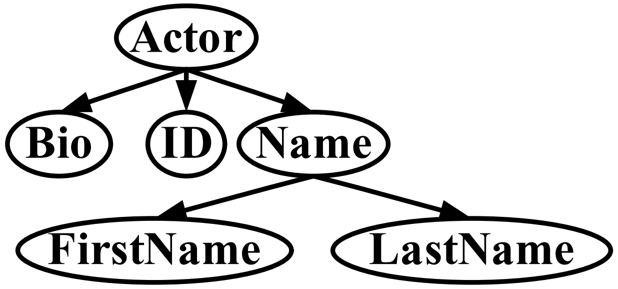
Process:

1. Study many applications that merge models of various data models
2. Specify a generic Merge operator including:
   - Merge inputs and outputs
   - Properties of the merged model
   - Conflict resolution strategy
3. Augment Merge to exploit semantics and application-specific usage including:
   - Data Integration
   - View Integration
   - Data Warehousing
   - Ontology Merging

Merge Example From Scenario



Our mapping contains structural information, so it can represent more complex relationships between the schemas than could be represented by direct correspondences between the input schemas. E.g., the structure in the mapping above allows us to specify the merged result on the bottom



Generic Merge Specification

Input: model A, model B, and mapping MapAB (which expresses equality or similarity relationships between elements in A and B)

Output: merged model G, mapping MapAG between A and G, and MapBG between B and G. G has the following properties:

- Elements equated by MapAB correspond to the same element in G
- Elements declared similar by MapAB are retained as distinct elements in G along with an un-interpreted element describing the relationship
- Elements in A or B unmapped by MapAB map to distinct elements in G
- For each relationship (x,y) in A or B, if x and y remain distinct in G, then a corresponding relationship exists in G
- Each element in G includes the disjoint union of all the properties

Applications that Use Merged Models

Merge must retain enough semantic information to use the merged model:

- **Data Integration**: Translate queries over mediated schema (merged model) to queries over source schemas (original models)
- **View Integration**: Translate queries over local user views (original models) to queries over global database schema (merged model)
- **Data Warehousing**: Translate data from original databases to data warehouse (merged model). Possibly requires data lineage tracing
- **Ontology Merging**: Make inferences over the global knowledge representation (merged model) including data instances