

Bioinformatics Service Reconciliation By Heterogeneous Schema Transformation

Lucas Zamboulis^{1,2}, Nigel Martin¹, Alexandra Poulouvassilis¹

¹School of Computer Science and Information Systems, Birkbeck

²Department of Biochemistry and Molecular Biology, UCL

Project funded by the BBSRC



Data Integration in the Life Sciences, 27-29 June 2007

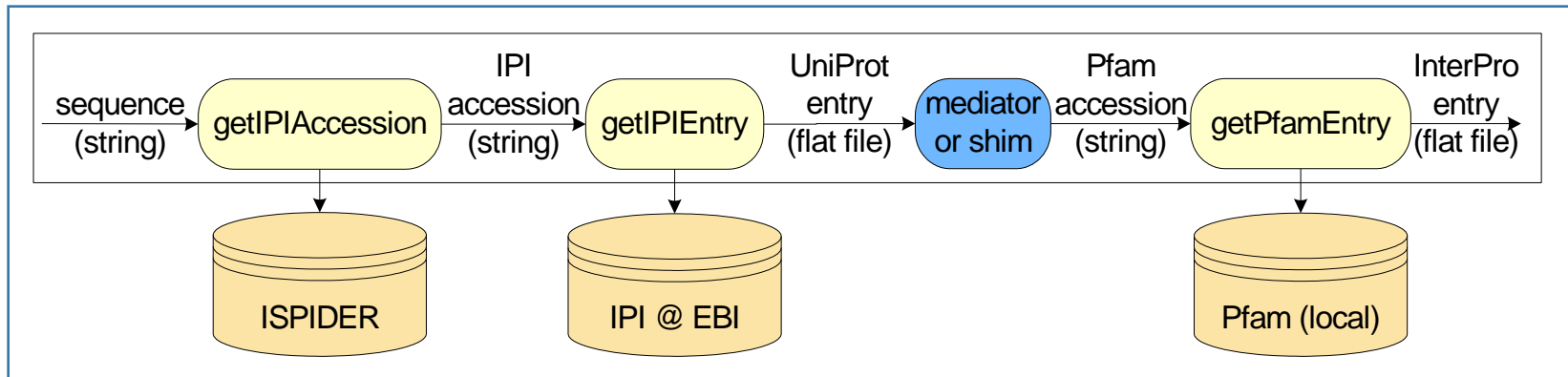


Problem Definition

- Plethora of offered bioinformatics services
 - impedes service composition
 - service discovery used to reduce the search space
- Semantically compatible services not able to interoperate:
 - service technologies
 - differences in data model, modelling, data types

→ need for **service reconciliation**.
- Reconciliation problem amplified due to:
 - simple strings used rather than complex types
 - service providers disinclined to supply annotations

Proposed Approach



- Service reconciliation via mediation using the AutoMed system
- Requirements:
 - Wide coverage of interoperability issues
 - Scalability of approach, promote reusability
 - Static/dynamic mediation

Related Work

- Service composition usually assumes service compatibility
→ services forced to handle mismatches internally with **custom code**.
- **Shims**. To minimise this issue and promote reusability, *myGrid* uses *shims*, i.e. services that act as intermediaries between services and reconcile their inputs and outputs.
- **Bowers & Ludäscher (DILS'04)** describe a framework that uses 1-1 path correspondences to one or more ontologies, possibly containing subtyping information, for reconciling services. Sample implementation uses mappings to a single ontology and generates an XQuery query as the transformation program.
- **Thakkar et al.** also use a mediator system, but for service integration, i.e. for providing an interface over other services – not for reconciling semantically compatible services that need to form a pipeline.

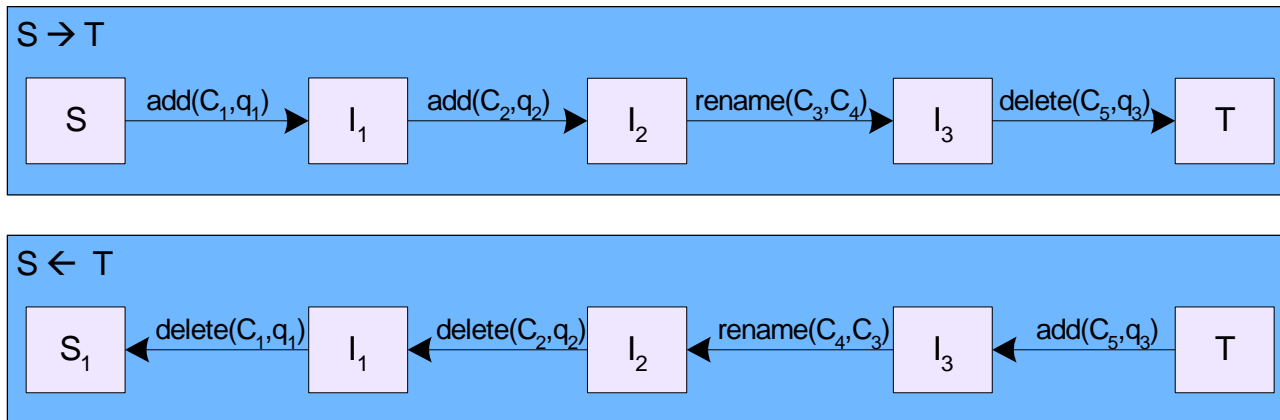
Related Work

- Requirements for our approach:
 - **Coverage:** services that produce/consume any type of data
 - **Heterogeneity:** cover all possible types
 - **Preciseness:** expressive mappings
 - **Scalability:** needed due to number of available services
 - **Applicability:** expectations from providers/users

A Brief Overview of AutoMed

- **AutoMed** is a heterogeneous data transformation & integration system that implements the **both-as-view (BAV)** approach
- Both-As-View approach:
 - Schema transformation approach
 - Can express/derive GAV, LAV, GLAV rules in/from BAV
 - Supports multiple data models (e.g. relational, object-oriented, semi-structured, XML, RDF)
 - Readily supports both global and local schema evolution
 - Can handle virtual, materialised and hybrid data integration
 - Internally uses a functional query language, IQL, for translating into and out of other query languages.

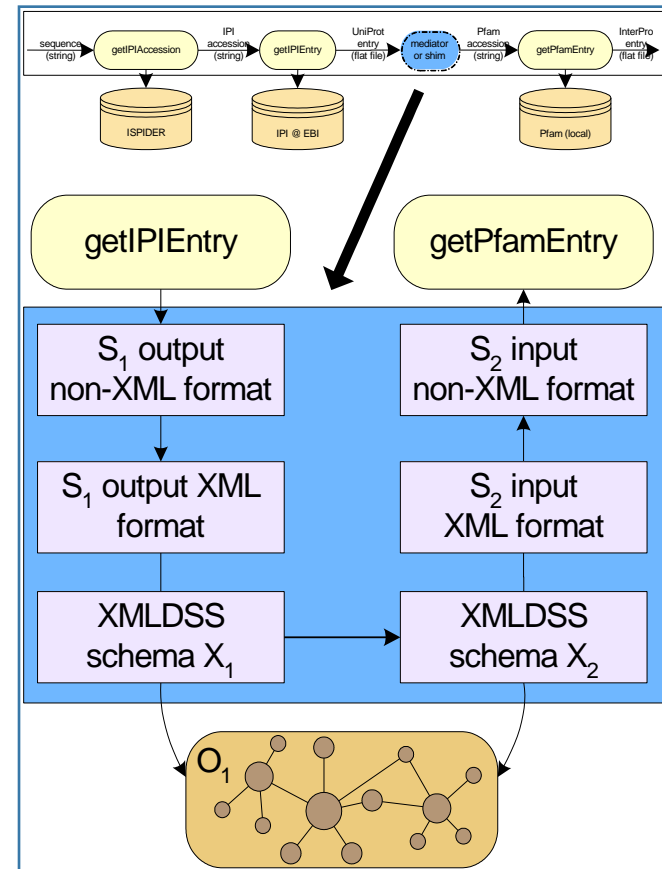
A Schema Transformation Approach



- Incrementally transform S so as to match it to T, creating pathway $S \rightarrow T$
- Each transformation adds/deletes/renames a single schema construct
- Each query defines the extent of the added/deleted construct
- Bidirectionality: queries used to automatically reverse pathway
- Pathways can be used to automatically translate both data and queries in both directions

Service Reconciliation By Schema Transformation

- Heterogeneity types
 - service technology
 - data model
 - semantic
 - schematic
 - primitive data type/scaling
- Assumption: workflow tool addresses service technology reconciliation

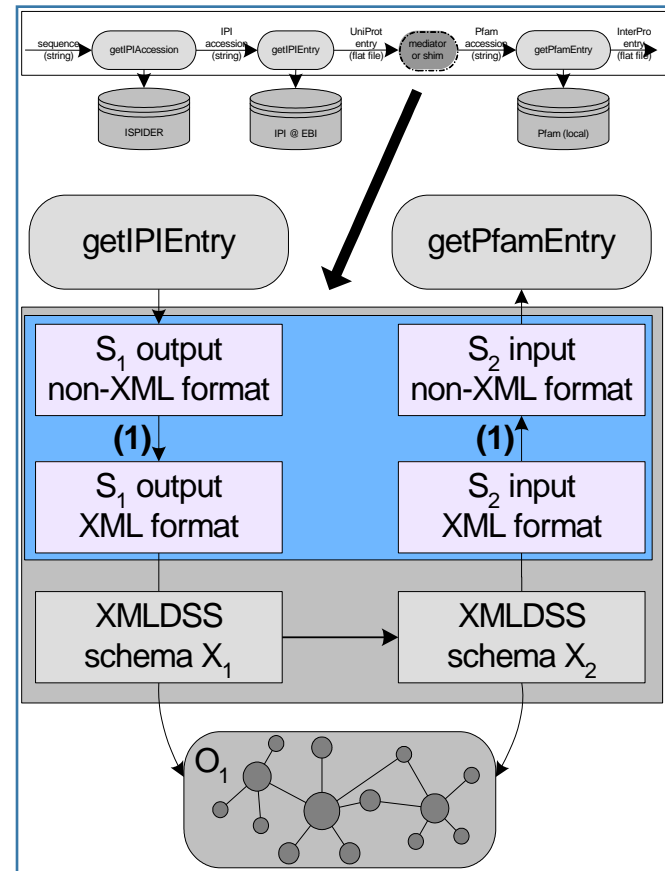


Heterogeneity Types

- **Data model heterogeneity:**
 - different data models (e.g. legacy flat files and XML)
 - different schema types (e.g. DTD and XML Schema)
 - a service producing/consuming XML data may not have an XML schema
- **Semantic heterogeneity:**
 - use of different terminology
 - describing the same information at different levels of granularity
- **Schematic heterogeneity:**
 - modelling the same information in different ways.
 - amplified in XML due to hierarchical structure and elements vs. attributes
- **Data type heterogeneity:**
 - use of different primitive data types, e.g. int and varchar
 - use of different units, e.g. miles and yards

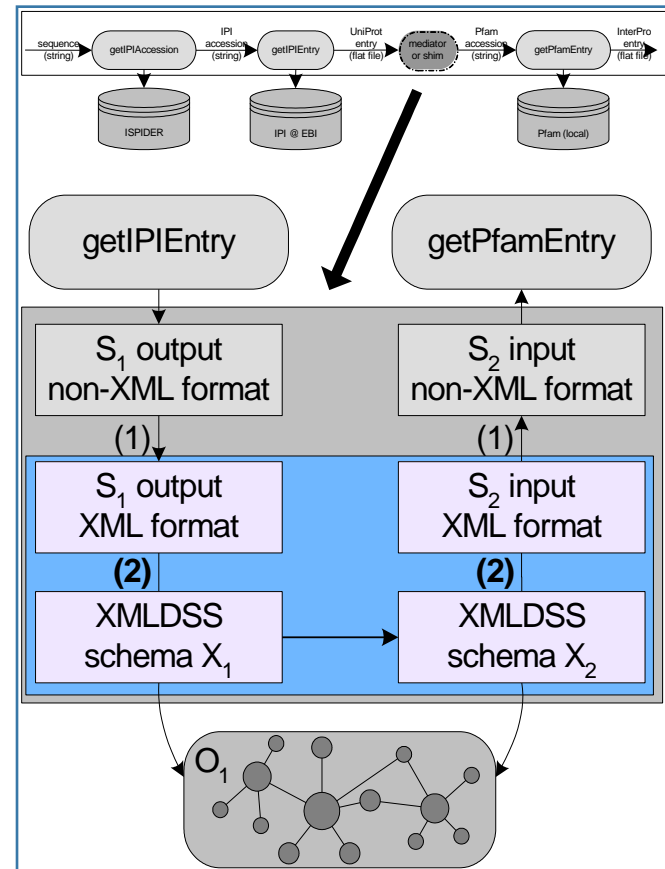
Addressing Heterogeneity

- **XML as the common representation format.**
- Addresses data model heterogeneity (data level)
- Format converters used to convert to/from XML.



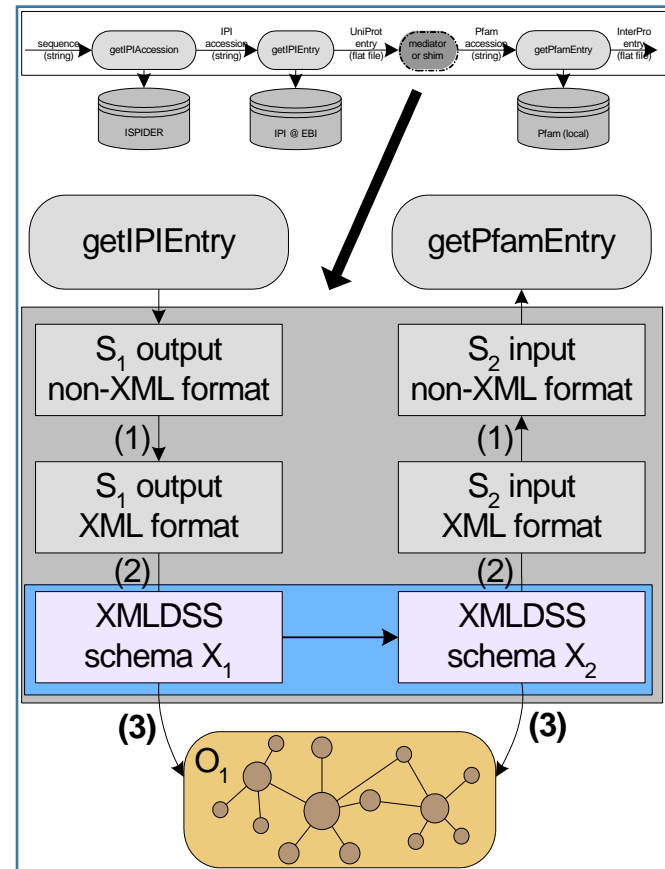
Addressing Heterogeneity

- **XMLDSS as the schema type.**
- Addresses data model heterogeneity (schema level)
- We use our own XMLDSS schema type as the common schema type for XML.
- Can be automatically derived from DTD/XML Schema, if available.
- Can be automatically extracted from an XML document.



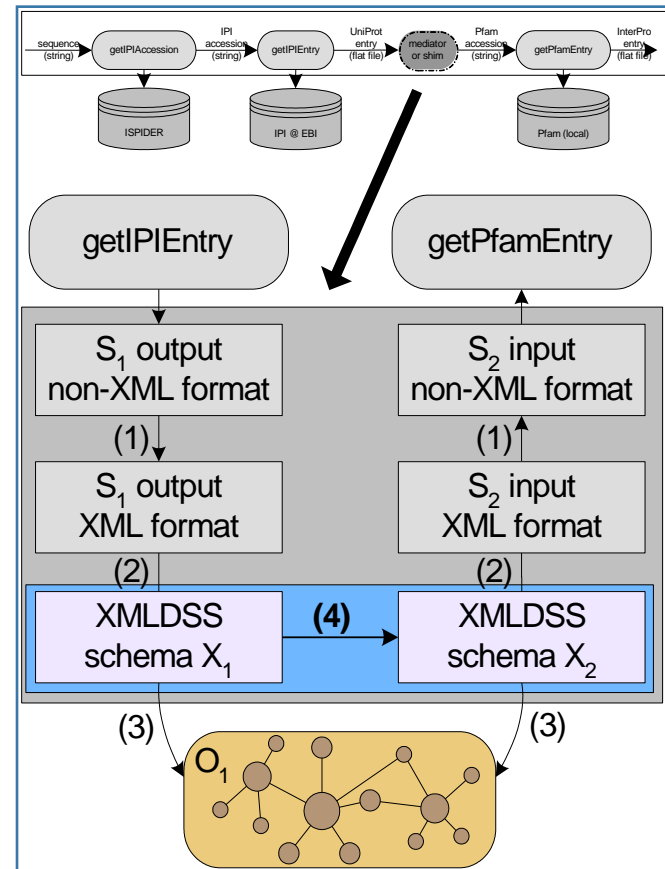
Addressing Heterogeneity

- **Correspondences to typed ontologies.**
- Addresses semantic and data type heterogeneities.
- One set of GLAV corr. between an XMLDSS schema and a typed ontology
 - An element maps to a concept/path
 - An attribute (or element + text node) maps to a literal-valued property/path
 - >1 corr. for elements/attributes



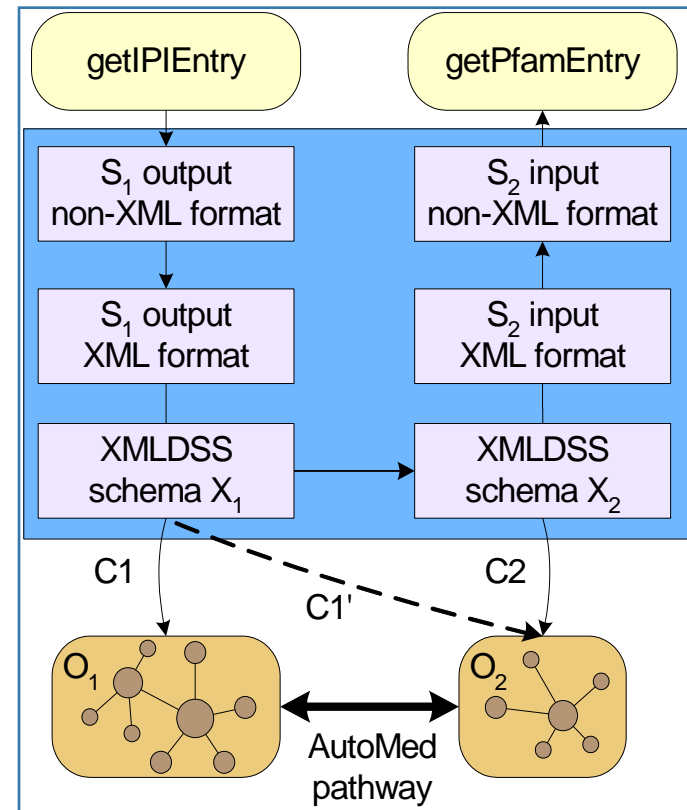
Addressing Heterogeneity

- **Schema and data transformation.**
- AutoMed used to automatically transform X_1 to X_2
- Conformance step: corr. used to create $X_1 \leftrightarrow X_1'$ and $X_2 \leftrightarrow X_2'$
- Restructuring step: create $X_1' \leftrightarrow X_2'$ (addresses schematic heterogeneity)
- Overall: $X_1 \leftrightarrow X_1' \leftrightarrow X_2' \leftrightarrow X_2$



Multiple Ontologies

- In a setting where:
 - X1 corresponds to O1 using C1
 - X2 corresponds to O2 using C2
 - there is a (direct or indirect) AutoMed pathway $O1 \leftrightarrow O2$
- Automatically produce new set of correspondences C1' for X1 and O2 (using query reformulation)
- Setting is now identical to single ontology setting.
- Proviso: C1' syntax must conform to our correspondences language.



Architectures

- A workflow tool can use our approach either dynamically or statically:
- **Mediation service.**
 - Workflow tool invokes service S1 and receives its output
 - Workflow tool submits output of S1, the schema of S2 and the two sets of correspondences to an AutoMed service.
 - The AutoMed service transforms the output of S1 to a suitable input for consumption by S2.
- **Shim generation.**
 - AutoMed is used to generate a shim for services S1 and S2.
 - XMLDSS schema transformation algorithm (currently) tightly coupled with AutoMed
→ functionality exported as single XQuery query.

Approach Advantages

- Support for both static and dynamic reconciliation
- Support for model, semantic, schematic and type heterogeneity
- Semi-automatic approach
- Correspondences:
 - Expressive
 - Incrementally built → no need for complete sets of correspondences
 - Reusable since each set applies to a single service, not pair
 - Reusable through the use of one or more ontologies
- Restructuring algorithm avoids loss of information by analysing the hierarchical nature of the source and target schemas and by using subtype information provided by the ontologies.

Future Work

- Extend expressiveness of correspondences
- Further investigate use of multiple ontologies (necessary conditions for the proviso to hold)
- Evaluate approach with a workflow tool in real-world scenarios
- Evolution of correspondences if schemas/ontologies evolve

Further Information

- Links
 - ISPIDER: <http://www.ispider.manchester.ac.uk>
 - AutoMed: <http://www.doc.ic.ac.uk/automed>
- ISPIDER project members
 - Birkbeck, University of London
 - University College London
 - University of Manchester
 - European Bioinformatics Institute