XML Database Modelling (Part-I)

Unit-II

XML, Objects and Relational Databases

- Relational databases are standard for storing, managing, and querying large amounts of data
- Object-oriented programming is the paradigm for developing applications (client or server, Web or non-Web)
- XML is serves as a "common ground" for sharing information

- The question is,
- How can we build object-oriented software that accesses relational databases or information in XML format?

Mismatch at multiple levels

- Relational model =! object model =! XML — but they'll all hold the same information!
- One approach was to unify the models — e.g., develop an object-oriented database or define a "serialized" object format
- But each paradigm has its own strengths
- Another option is to ***manually map information*** *across* these boundaries — but it is tedious, error prone, not reusable
- Solution: automation, at all levels

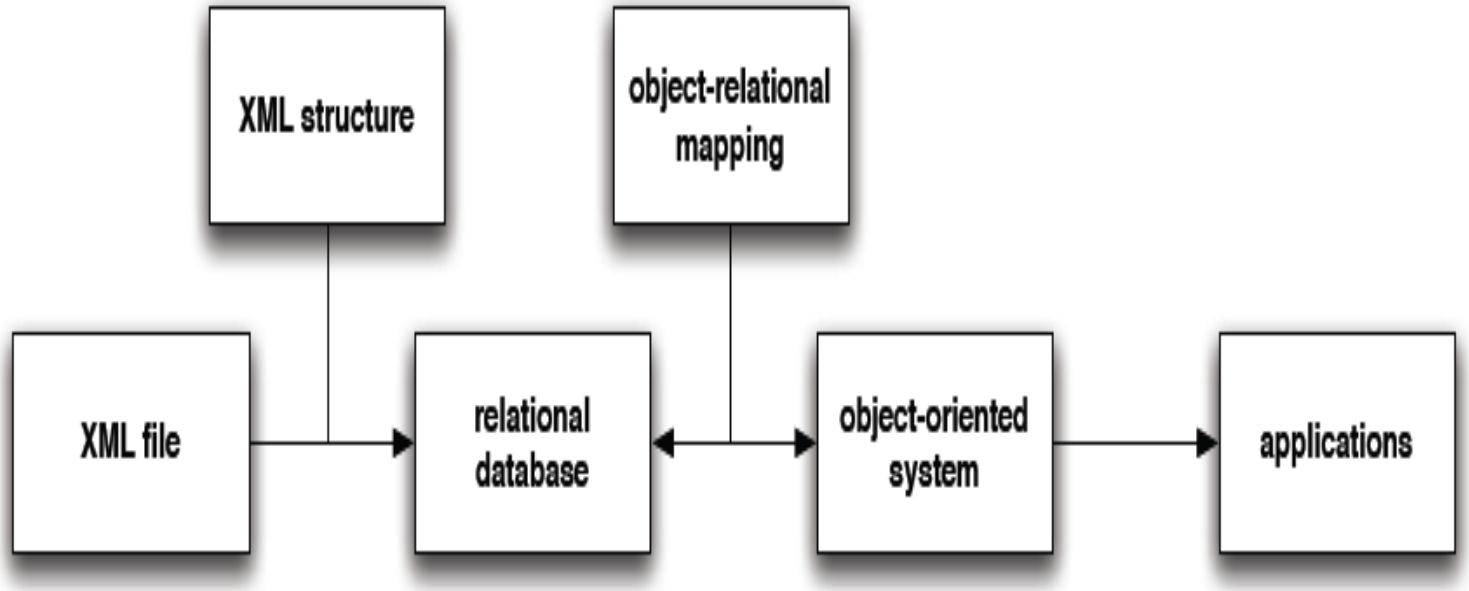
Two technologies towards unifying

- Two key technologies here:
- The XML standard includes mechanisms that can *specify the structure of a given XML file (DTD, XSD)*

- Techniques/conventions have been developed for translating a relational structure to an object structure

  — an activity called *object-relational mapping or ORM*

Contd...


XML structure
object-relational mapping
XML file
relational database
object-oriented system
applications

Specifying XML Structure

- Originally, the XML standard provided for a *document type definition (DTD) file, which specified the structure* for XML files
- This standard evolved into XML Schema
- XML files can specify the DTD or XSD to which they conform
- Validators have been developed to compare a given file against a DTD or XSD to see whether that file "complies" with the claimed structure

From XML to Objects

- One can assume that XSD is analogous to defining a *class in an object-oriented* programming language
- Thus, an XML file can be viewed as containing one or more *instances of the class defined* by an XSD
- Based on this manifestation, there can be programs that reads an XSD file and *generates Java* classes that correspond to that file
- Eg. For Automated tools - Castor, **JAXB**, XGen, XMLBeans, etc.

From Objects to Relations

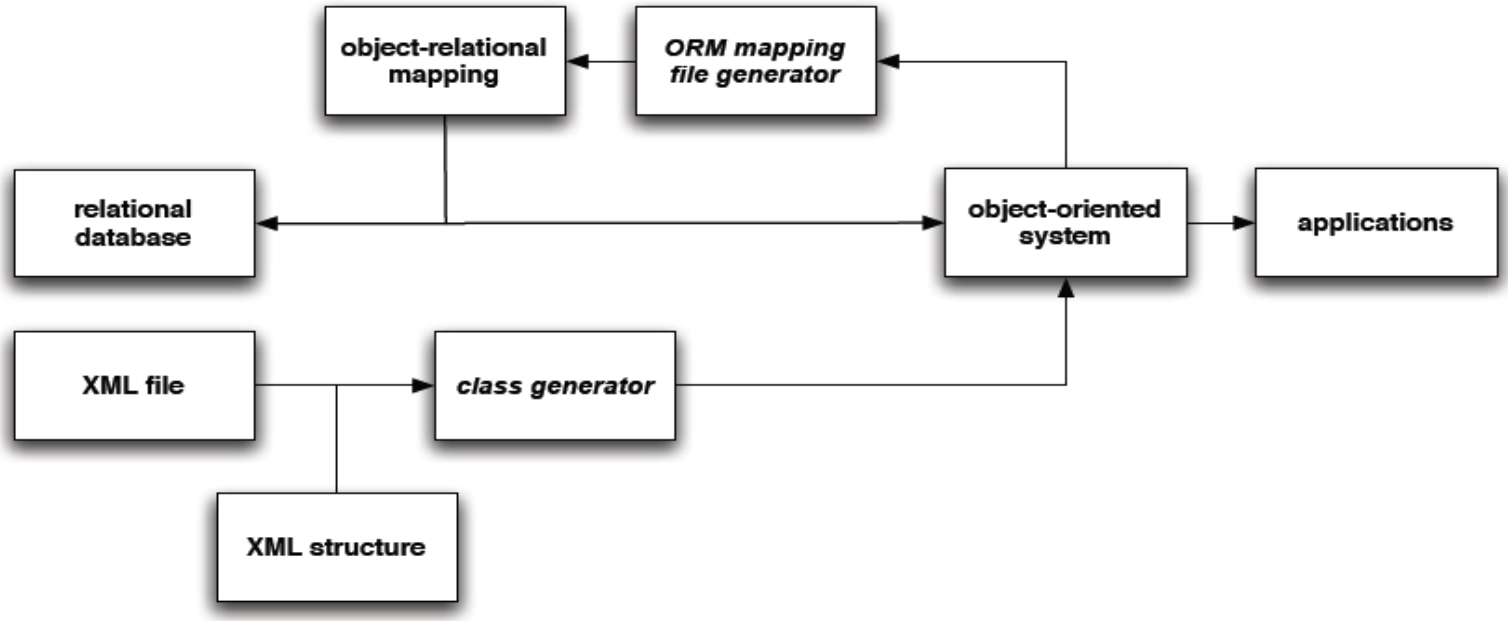
- An initial approach **toward automated object-relational mapping** followed this process:

  1. Define the objects/classes and relations
  2. Build a *mapping file between the objects and relations*

- Tools for doing this - Hibernate, JDO, iBatis
- Easier than writing custom code to read/write objects from a relational database
- Drawback — if either the class or relation changes, the mapping file has to be modified as well

Making ORM easier

- Xdoclet, can automate the generation of a mapping file: just change the annotations in the source
- Xdoclet can read *annotations in Java code, then* generate new files (Java, XML, HTML, whatever) based on those annotations



Contd...

- **What we needed is, classes we are mapping are themselves automatically generated from the XML schema definition**
- **So that when XML schema changes, we would have to regenerate the classes and reinstitute the ORM annotations**

- A tool that does *both* — *generate the classes with the ORM map*

- Solution - HyperJAXB, a tool that bridges the gap between JAXB, Hibernate, and Xdoclet

Unifying technologies towards full automation