Mappings Make Data Processing Go ’Round
An Inter-paradigmatic Mapping Tutorial

Ralf Lämmel and Erik Meijer

Microsoft Corp., Data Programmability Team, Redmond, USA

Abstract. Whatever programming paradigm for data processing we choose, data
has the tendency to live on the other side or to eventually end up there. The ma-
jor paradigms for data processing are Cobol, object, relational and XML; each
paradigm offers many facets and many versions; each paradigm provides specific
forms of data models (object models, relational schemas, XML schemas, etc.).
Each data-processing application depends on a horde of interrelated data models
and artifacts that are derived from data models (such as data-access layers). Such
conglomerations of data models are challenging due to paradigmatic impedance
mismatches, performance requirements, loose-coupling requirements, and others.
This ubiquitous problem calls for a good understanding of techniques for map-
pings between data models, actual data, and operations on data. This tutorial lists
and discusses mapping scenarios, mapping techniques, impedance mismatches
and research challenges regarding mappings.

Keywords: Data processing, Mapping, XML data binding, Object-XML map-
ping, Object-relational mapping, Cross-paradigm impedance mismatch, Data
modeling, Data access, Loose coupling, Software evolution.

1 Introduction

We steal the beginning of our tutorial from elsewhere: “Once upon a time it was pos-
sible for every new programmer to quickly learn how to write readable programs to
Create, Read, Update and Delete business information. These so-called CRUD appli-
cations, along with reporting, were pervasive throughout business and essentially de-
finite IT or MIS as it was called in those days.” [92] (Dave Thomas: “The Impedance
Imperative Tuples + Objects + Infosets = Too Much Stuff!”).

Instead, today we face the following diversity:

– Cobol applications with keyed files are still developed and they make sense.
– Relational databases have fully matured and they are unarguably omnipresent.
– OO databases innovate, perhaps at a slow pace, but they must be taken seriously.
– The XML hype is over. XML types and XML documents are everywhere now.
– All these paradigms have triggered a myriad of query languages and 4GL tools.
– Much current CRUD development is done with OO languages with various APIs.

This tutorial is about the challenges implied by such diversity in data modeling and
data processing. Either there are respectable, perhaps fundamental reasons for all this
diversity, or it is just plain IT reality. No matter what, we need to map amongst these
paradigms, and everyone is trying to do that anyhow. According to a designated online resource,\(^1\) there are roughly 60 established products for X/O mapping, also known as XML data binding, i.e., XML schemas or DTDs are mapped to object models. We reckon that practice is ahead of foundations in this area, but this surely implies ad-hoc approaches with unnecessary limits and complexities. We need basic and applied research on inter- and intra-paradigm mappings.

**What Is a Mapping Anyway?**

We should make more precise what we mean by ‘mapping’. We have to disappoint those readers looking for a detailed or even formal definition. Instead we offer the following explanation and the illustration in Fig. 1.

- Mapping is essentially about the transformation of values between data models.
- The data models typically involve different paradigms (Cobol, OO, relational, XML).
- Fig. 1 opts for a type-based mapping (described at the type level).
- By contrast, instance-based mappings directly define value transformations.
- Other mappings may implicitly define data models for source and target.
- CRUD operations may need transcription from the source to the target or vice versa.
- There may be more levels than those in the figure, e.g., the level of protocols.

**Road-Map for the Tutorial**

- Sec. 2 presents diverse illustrative mapping examples.
- Sec. 3 is an attempt to collect (some) mapping concepts.

---

\(^1\) [http://www.rpmourret.com/xml/XMLDataBinding.htm](http://www.rpmourret.com/xml/XMLDataBinding.htm)

\(^2\) The term ‘data model’ is ambiguous as it may refer to both the general data model of a paradigm such as the ‘relational model’; it may also refer to domain/application-specific data models such as a particular ‘relational schema’ or ‘object model’; [http://en.wikipedia.org/wiki/Data_model](http://en.wikipedia.org/wiki/Data_model) In this tutorial, we favor the latter meaning.