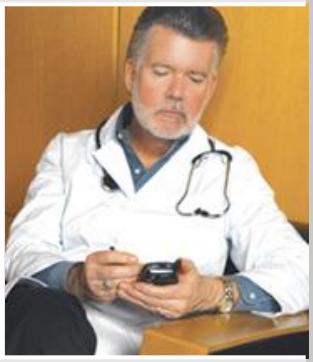
epocrates®

Integrating XML with legacy relational data for publishing on handheld devices

David A. Lee
Senior member of the technical staff
dlee@epocrates.com





Importance of correct Clinical Information

Introduction and Agenda

- Introduction
- Background “The Problem Space”
- False Starts
- “Common Object Model”
- Final Design
- Lessons Learned
- Conclusion

Introduction

- Who is Epocrates ?
- Common Terminology
- Core Application

Introduction
Who is Epocrates ?

- Epocrates is the industry leader in providing clinical references on handheld devices.
- 475,000 active subscribers
- Subscription based clinical publishing

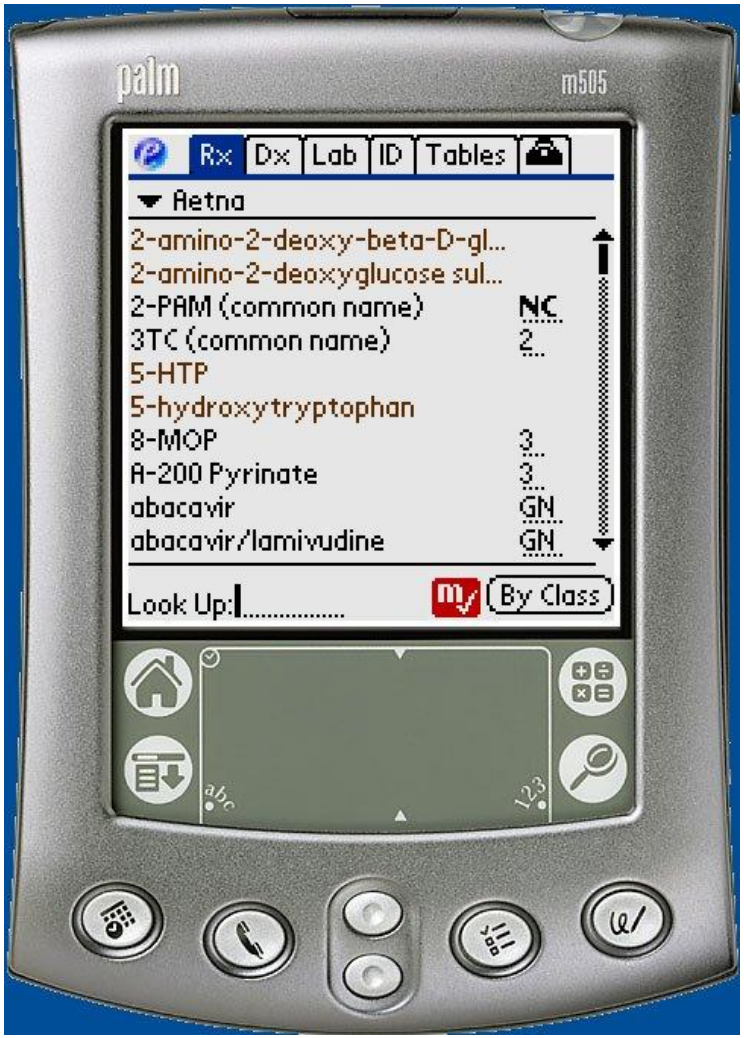
Introduction
Common Terminology

- **PDA** - "Personal Digital Assistant".
- **Monograph** - information describing a single drug, disease, lab test, preparation or other clinical entity.
- **Syncing** - The process of synchronizing a server's database with a PDA
- **PDB** - "Palm Database". A very simple variable length record format with a single 16 bit key index.

Introduction
Core Application

"Essentials"

Handheld Clinical Reference

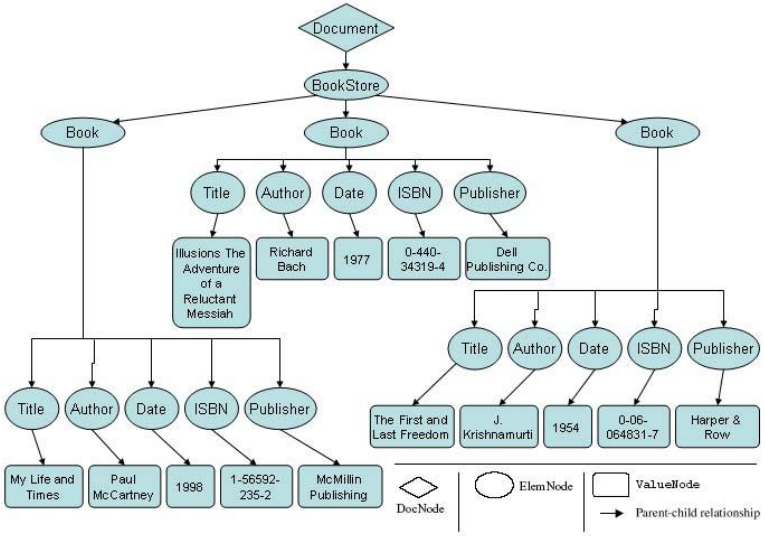


Background “The Problem Space”

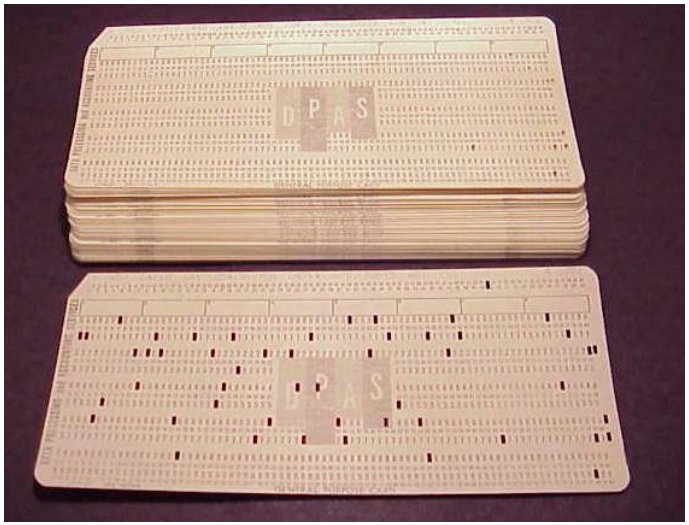
- XML vs “Legacy” data
- Characteristics of the application
- Characteristics of "legacy" data
- Characteristics of "new" XML data
- Publishing Workflow
- Workflow Requirements

Background
XML vs “Legacy” data

XML



“Legacy” data



Background
Characteristics of the Application

- Runs on handheld devices
- Limited memory capability (8MB typical)
- Simple database
- Small display
- Synchronization speed critical
- Linking of related content

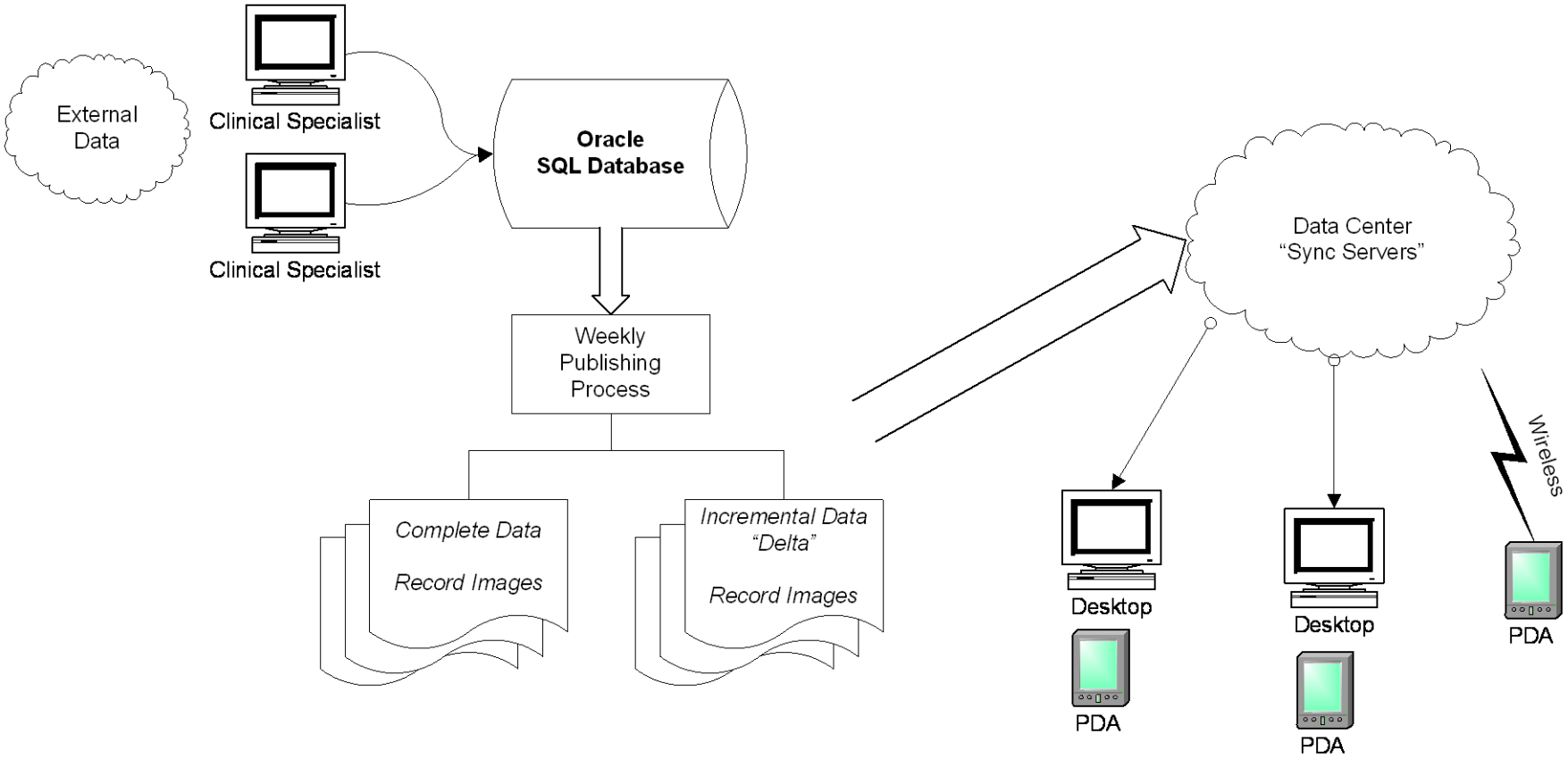
Background
Characteristics of “legacy” data

- Stored in Oracle SQL database
- Highly structured and referential
- Hard to change schema
- Data constantly changing (manual editing)
- Specifically designed schemas and content for presentation on PDAs.
- Difficult to change workflow, representation or tools
- Part of a complex workflow for publishing data to large subscriber base

Background
Characteristics of “new” XML data

- One large XML Document containing many “monographs”. 15-150MB common.
- Periodic updates with unknown amount of change.
- Schema likely to change unexpectedly
- No control over content
- Referential data within and across monographs.
- Both structural and “markup” type elements

Background
Publishing workflow



Background
Workflow Requirements

- Integrates into current workflow with minimum changes to existing process and data structures.
- Reliable change detection
- Support for deferred detection of dependancies
- Accurately manage changes when data is updated
- Resilient to XML schema changes
- Extensible design

False Starts

- One Big BLOB
- Full Normalization
- One BLOB per monograph
- XML Database

False Starts
One Big BLOB

Store entire XML as a single "BLOB"

Pros

- Very simple and easy

Cons

- Deferred almost all processing to sync server
- Impossible to detect changes
- Solves no significant problem over using the filesystem
- No relational representation
- Difficult to search with SQL
- No structure or indexing at SQL level

False Starts
Full Normalization

Fully normalize XML schema into separate DB tables for every element.

Pros

- “Ideal” relational representation, referential integrity
- Fine granularity of modification detection

Cons

- Very large number of tables (> 150)
- Difficult to implement
- Bad performance

False Starts
One BLOB per Monograph

Split each monograph into an XML document fragment and store as a BLOB.

Pros

- Fairly simple to implement
- Granularity maps well to device DB structure

Cons

- Difficult to search via SQL
- Referential data not exposed at SQL level
- Significant processing deferred to sync server

False Starts
XML Database

Use a native XML Database

Pros

- Efficient and architecturally clean XML storage

Cons

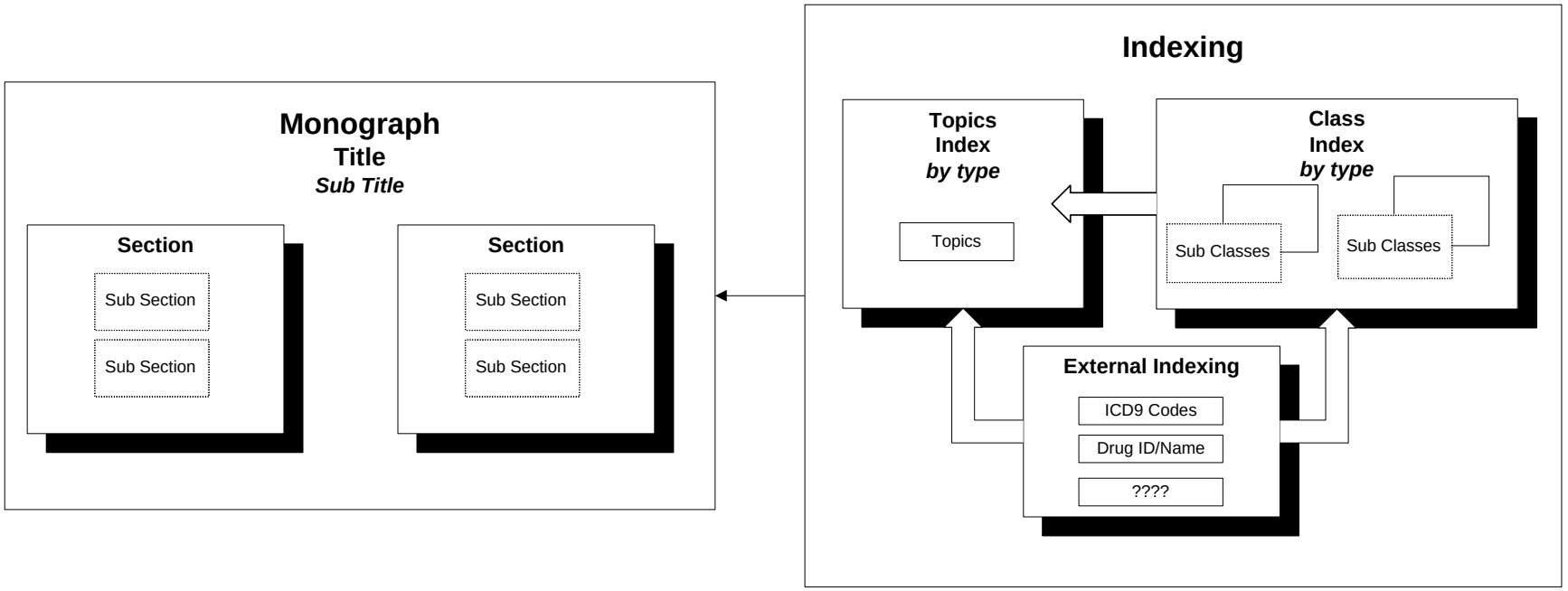
- NO in-house experience
- Difficult to integrate with existing tools
- “Locked In” to DB provider
- XML largely processed in-memory

Common Object Model

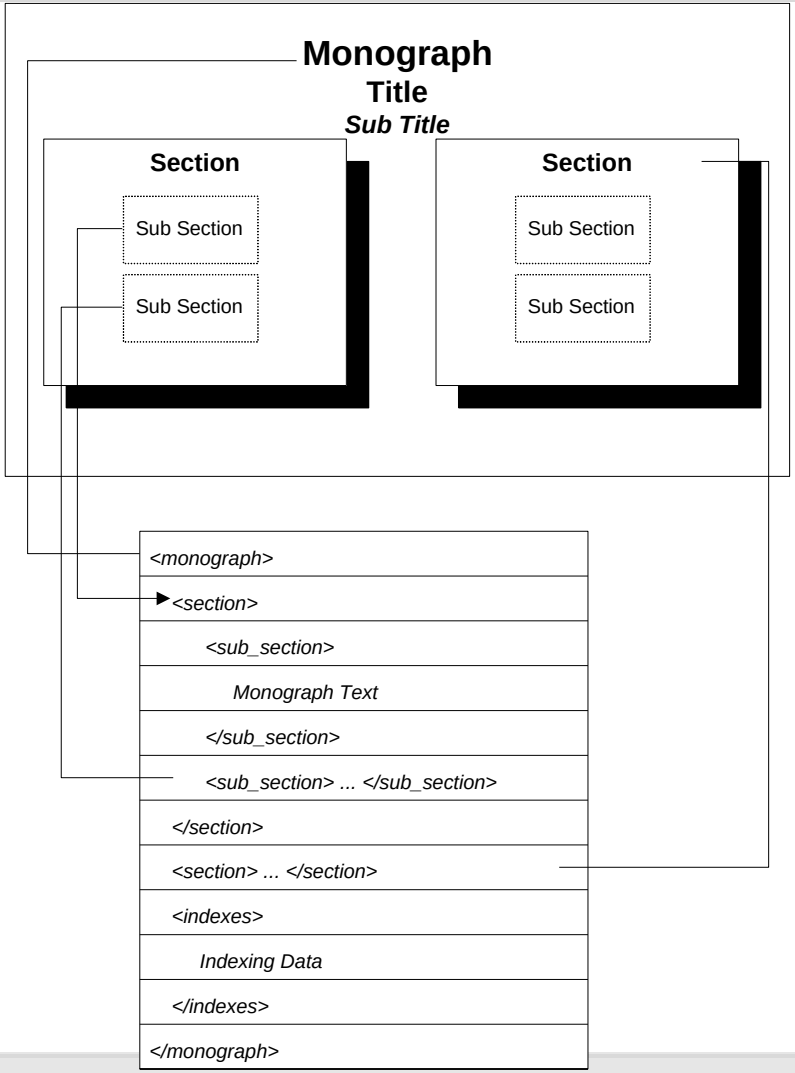
Abstract design pattern for modeling content with mappings to concrete representations.

- Document Structure
- XML Mapping
- Database Mapping
- Application Mapping

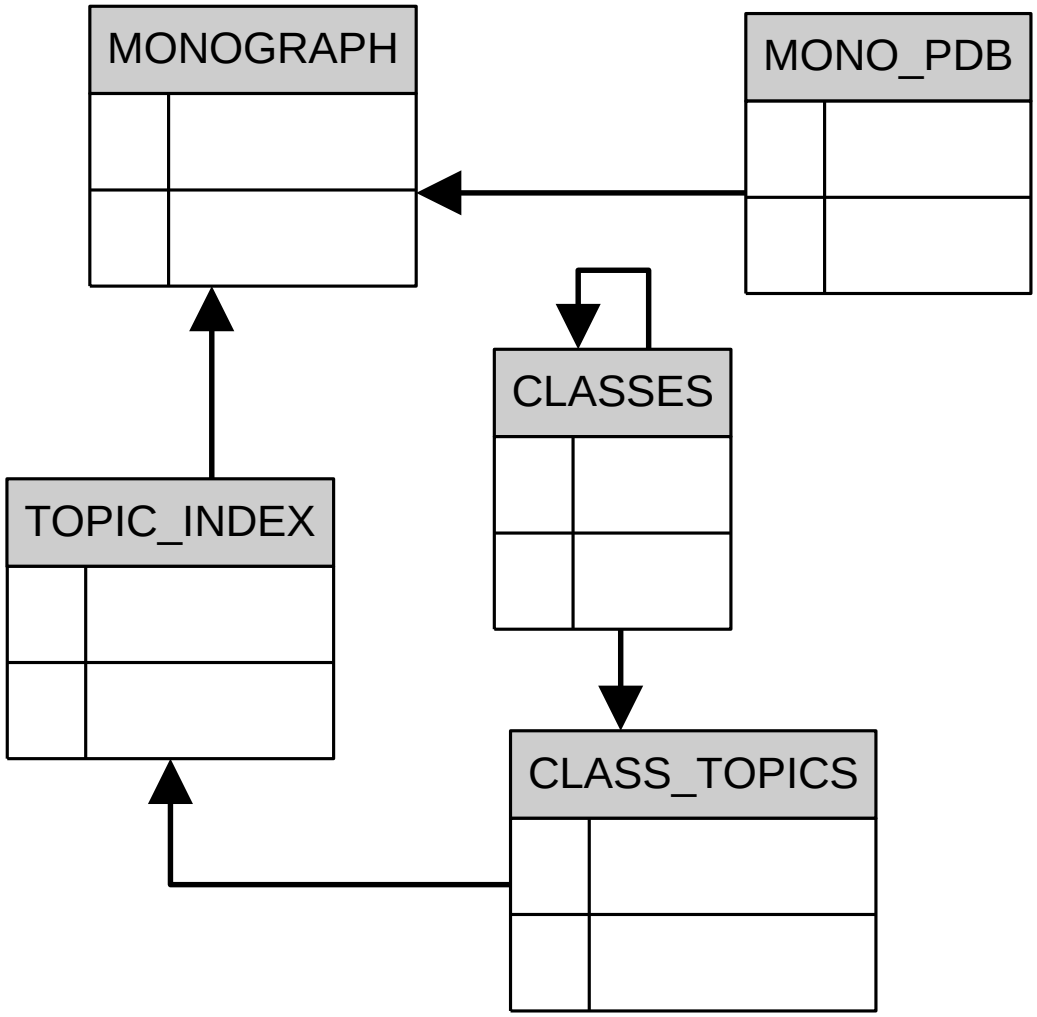
Common Object Model
Document Structure



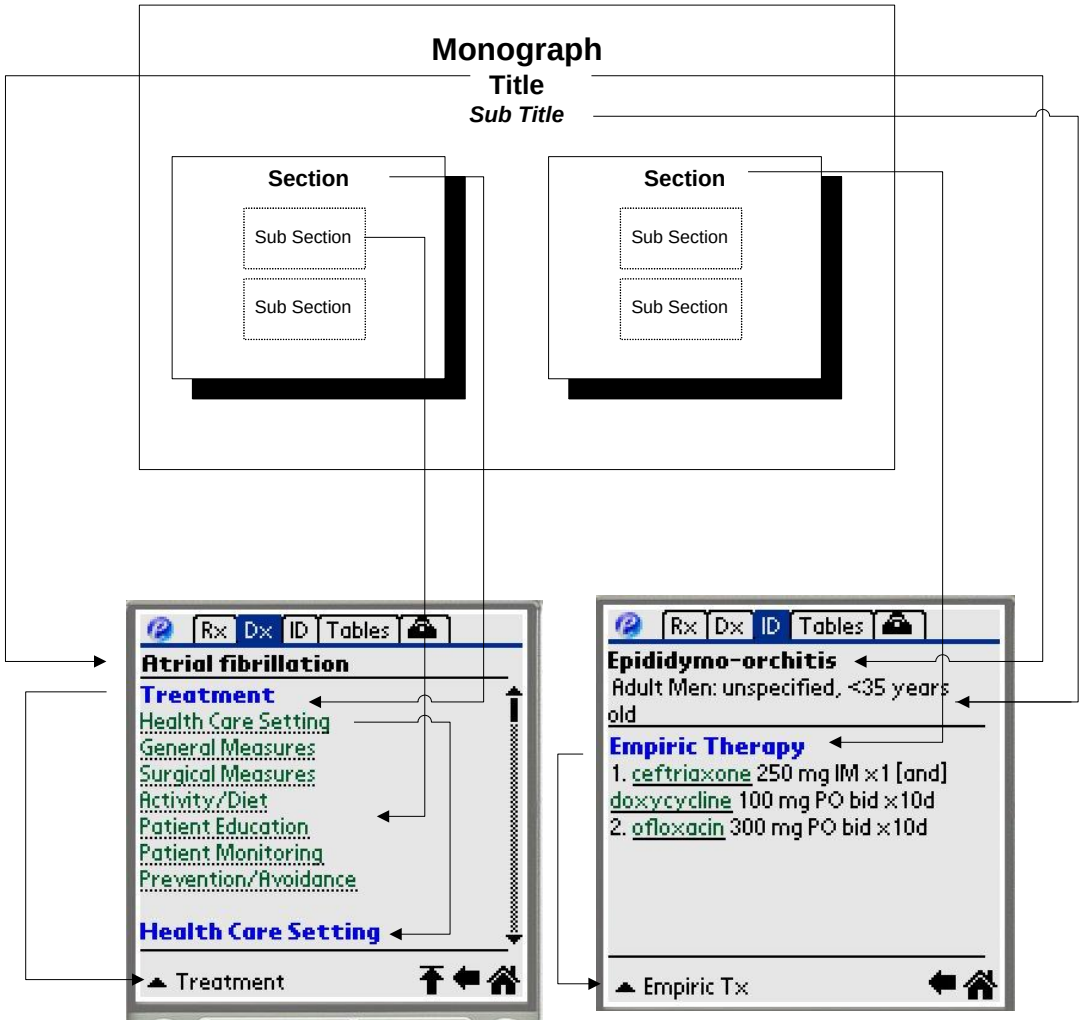
Common Object Model
XML Mapping



Common Object Model
Database Mapping



Common Object Model
Application Mapping

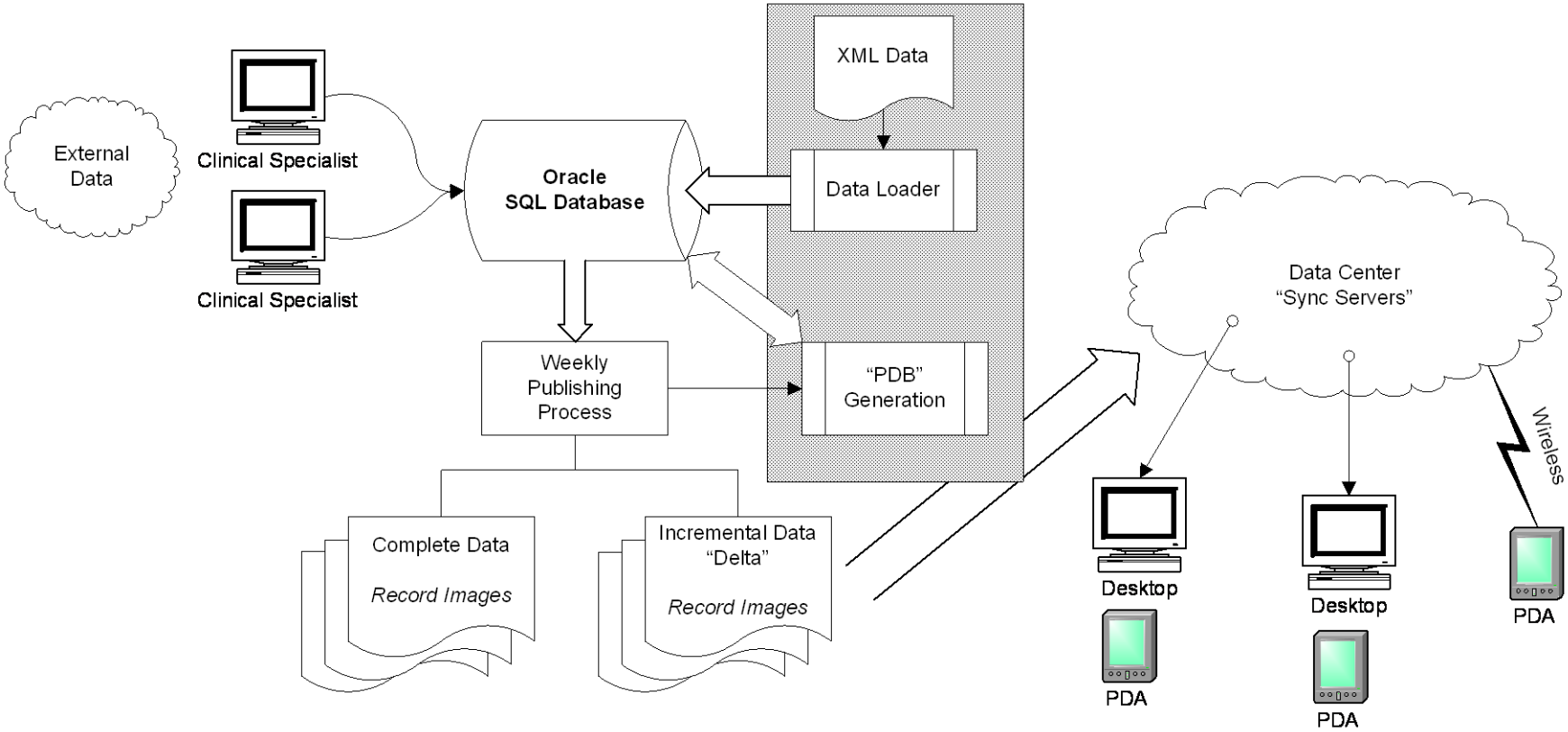


Final Design

Final Design comprises a model based on the Common Object model as a Design Pattern.

- One BLOB per Monograph
  - XML Document Fragment
- Normalized referential data
  - Key fields as table fields
- Separate “compiled” BLOB per monograph

Final Solution
Modified Workflow Processes



Lessons Learned

- Split up large XML files
- Don’t assume “All or Nothing”
- Process XML with a programming language
- Look for the distinction between “Structure” and “Markup”
- The ‘Real World’ is a compromise.

Conclusion