Avancier Methods (AM)
Data Architecture

Data Structures

It is illegal to copy, share or show this document
(or other document published at http://avancier.co.uk)
without the written permission of the copyright holder
Long before computers

- Business actors depended business information being.
  - moved in messages (data flows) – often paper
  - stored in memories (data stores) – card indexes and filing cabinets

- Human actors created, processed and moved this data using
  - pens
  - typewriters and
  - snail-mail postal organisations.
The Information Age

- Digitisation of business information using computerised IT.

- Physical forms of business data developed from
  - from human memory and speech,
  - to written records,
  - to bits stored in computer memory and persistent data storage.

- This use of IT massively increased the ability of a business to capture, move, store, process, and analyse business data.

- “Today’s CEOs know that the effective management and exploitation of information through IT is a key factor to business success.” (TOGAF 9.1)
What does an enterprise data architect do?

► Leading bank urgently seeking a proven EA to engage and lead IT projects including
  ■ the Enterprise Information Architecture
  ■ information models and flows,
  ■ data dictionaries, data standards
  ■ data quality standards and processes

► develop and maintain the logical Enterprise Information Architecture that enables seamless information interoperability of all Bank systems for efficiency and cost-effectiveness.

eutopiaonline.com
PREFACE

<table>
<thead>
<tr>
<th>Business</th>
<th>Passive Structure</th>
<th>Required Behaviour</th>
<th>Logical Structure</th>
<th>Physical Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data / Information</td>
<td>Data Entity</td>
<td>Data Flow</td>
<td>Log Data Model</td>
<td>Data Store</td>
</tr>
<tr>
<td>Applications</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Infrastructure Technology</td>
<td>Platform Service</td>
<td>Platform Interface</td>
<td>Platform Service</td>
<td></td>
</tr>
</tbody>
</table>

Copyright Avancier Limited 2007-2016
Data architecture

- EA is about business operations that create and use information.
- Data architects are concerned with:
  - Data stores
  - Data flows
  - Data qualities
Data stores – data at rest

► Businesses have to store information for future use, in persistent data stores.

► A data store contains a data structure that can be described in terms of inter-related entities.

► Concerns include:
  ■ The locations, contents and synchronisation of data stores
  ■ Physical data store forms
  ■ Data store schema standards
  ■ Centralised and distributed data storage
TOGAF terminology

- TOGAF uses the term **data component** rather than data store.
- There are two interpretations
  - A passive data structure, contained in some kind of data store.
  - An active data server (an application component) that provides read/write access to a data structure.

- Either way, the data component contains a data structure that can be described in terms of inter-related entities.
### Physical data store forms

<table>
<thead>
<tr>
<th>Conceptual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
</tr>
<tr>
<td>Physical</td>
</tr>
<tr>
<td>Real</td>
</tr>
</tbody>
</table>

*Data Store*
Physical data store forms

- Data architects are concerned with the forms of matter and energy in which information is stored.
- In theory, data architects can design non-digital stores.
- In practice most focus on digital ones.
Digital data store forms

► Magnetic disks are currently being replaced by flash storage.

► Flash is optimised for the asymmetric use cases of mobile devices, where data is written few times and read many times.

► So, if you want to find out what may replace flash memory try this:

► Architects have to research physical data storage forms as the need arises.
### Data store schemas

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Logical</th>
<th>Physical</th>
<th>Real</th>
</tr>
</thead>
</table>

A physical data schema or model defines a data structure in a way that is tuned to specific technologies and NFRs.
Data store schema varieties

- Transactional database
- Data warehouse
- Document store
- Big data store

Sometimes misleadingly called NoSQL databases

Need some kind of SQL if they are to support the coding of queries.
Transactional databases (for OLTP)

► Processing requirement
■ many short on-line update transactions (insert, update, delete).
■ short fast queries
■ maintains data integrity (despite concurrent users)
■ throughput measured by number of tpm or tps.

► The OLTP database – typically relational
■ typically a “normalised” relational database
■ typically accessed using SQL
■ holds current data in detail, but only a little history
■ structured and strongly typed business data.
■ Increasingly, solid state drives and in-memory storage
  • enable tens of thousands of transactions per second
  • greatly speed up enquiries and reports.
Data warehouses for OLAP (On-line Analytical Processing)

► Processing requirement
  ■ often aggregates or summarises a lot of transactional data
  ■ usually updated by batch data loads.
  ■ often complex queries
  ■ focus on response/cycle time.

► OLAP database
  ■ may have a star or snowflake schema, a key-value store or column store
  ■ enable analysis and summary reports for management information
  ■ optimised for data retrieval by the use non-relational data structures (column stores, key value stores etc.).
  ■ practices includes cleansing, sorting, transforming, aggregating data
  ■ often associated with specific BI tools.
<table>
<thead>
<tr>
<th><strong>OLTP v. OLAP</strong> (<a href="http://www.rainmakerworks.com">www.rainmakerworks.com</a>)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OLTP System: Online Transaction Processing (Operational System)</strong></td>
</tr>
<tr>
<td><strong>Source of data</strong></td>
</tr>
<tr>
<td><strong>Purpose of data</strong></td>
</tr>
<tr>
<td><strong>What kind of data</strong></td>
</tr>
<tr>
<td><strong>Inserts and Updates</strong></td>
</tr>
<tr>
<td><strong>Queries</strong></td>
</tr>
<tr>
<td><strong>Speed</strong></td>
</tr>
<tr>
<td><strong>Storage space</strong></td>
</tr>
<tr>
<td><strong>Data structure</strong></td>
</tr>
<tr>
<td><strong>Recovery</strong></td>
</tr>
</tbody>
</table>
Document store

- Stores hierarchical data structures
  - Commonly XML or JSON documents

- (A network data model can be divided into discrete hierarchical data structures - to be discussed later.)
Big data store – 3Vs

► You can store whatever (unstructured or structured) data is captured, then glue different data sets together later

► Volume
  ■ large volume - many tera/peta bytes collected in minutes.

► Variety
  ■ unstructured and/or weakly typed data
    ● E.g. social media data
  ■ behaviour that has not traditionally been monitored or directed by business systems.
    ● E.g. 1.5 billion pieces of data generated per car per grand prix.

► Velocity
  ■ Rapidly store a high volume and variety of data
  ■ Data retrieval may require faster technologies - a variety of data structures and query languages.
Avancier customers have told me

► Housing association has 1m customers.
  ■ We collect data from sensors in houses
  ■ 100 megabytes per customer per hour

► Rolls Royce get paid for engines by "thrust hours".
  ■ So they ping their engines once an hour to capture engine data.
  ■ 2 to 3 terabytes of data per minute of thrust.
How do Big Data stores handle high data volumes?

► Functional scaling – divides data types

► Sharding – divides a large population across database instances
All data store schemas are designed to enable I/O data flows

CUSTOMER entity
- Customer number
- Customer name
- Customer address

ORDER entity
- Customer number
- Order number
- Order amount
- Product type

Customer Order History SEQUENCE
- Customer number
- Customer name
- Customer address
- Set of Orders ITERATION
  - Order SEQUENCE
    - Order number
    - Order amount
    - Product type
  - Order END
- Set of Orders END
- Customer Order History END
## Logical data models

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Logical</th>
<th>Physical</th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Logical Data Model</td>
<td>Physical Data Schema</td>
<td>Data Store</td>
</tr>
</tbody>
</table>

- **Logical**
  - Technology neutral,
  - But usually focused on one data store

### Diagram

- Logical Data Model
- Customer
- Product
- Depot
- Sale
- Stock

---

Copyright Avancier Ltd 2008-2016
Three data model notations

**UML**

- Customer
- Department
- 1
- *children
- parent

**IDEF**

- Entity
  - Entity Name
  - Relationship Name
- Relationship
  - One
  - Manages
- Many
  - Entity Name
  - Optional Existence
  - Attribute Name

**Barker**

- Woman
  - Optional
  - Multiple
  - May bear
  - Was born to
  - Child
- Coach Trip
  - Carries
  - Takes
  - Passenger
- Choir Performance
  - Needs
  - May belong to
  - Singer Performance

Copyright Avancier Limited 2007-2016
Logical data model

- A model that describes the data that must persist for the processes of a business system to work.
- A network of entities, attributes and relations between entities.
- A kind of “domain model” that defines business terms and concepts.
- A data structure that shows entities and events of importance to a business.

(Not a database schema – unless you want it to be.)

**DEPARTMENT entity**
- DeptNumber (P Key)
- DeptName
- DeptManager

**EMPLOYEE entity**
- DeptNumber (F Key)
- Empid (P Key)
- LastName
- FirstName
- Salary

**TIMESHEET entity**
- Empid (P Key + F Key)
- WeekEnding (P Key)
- HoursWorked
How to partition the network structure of a domain model?

► Note that over time
  ■ Subtypes tend to become roles
  ■ Aggregations become associations
  ■ 1-1 associations become 1-N
  ■ 1 to N become N-to-N w link entities

► The result?
► A network of 1-to-N associations, as in this Salesforce.com domain model
Salesforce.com domain model redrawn (cf. ZF column 1?)

Kernel entities
Relationships between them (conceptual model?)
Attributes and characteristic entities (logical model?)
Mapping logical to physical

- TOGAF’s “logical data component”
  - is a logical definition of the data in a data store
  - can be documented as a logical data model
  - that is, an entity-attribute-relationship model.

- TOGAF’s “physical data component”
  - is a vendor/technology specific realisation of a logical data component.
  - could be database, data warehouse, document store, web information server and transaction log
  - has a technology-specific data schema.
Mapping logical data models to technical-specific data schema

<table>
<thead>
<tr>
<th>Logical data component</th>
<th>Physical data component</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODASYL database schema</td>
<td>Relational database schema</td>
</tr>
<tr>
<td>XML schema (footnote 2)</td>
<td>OData-compliant web information server.</td>
</tr>
</tbody>
</table>

- **Entities**: Records, Tables, Complex types, Entities
- **Attributes**: Fields, Columns, Contained elements, Properties
- **Relationships**: Address pointers, Foreign keys, Contained elements, Navigation properties
The challenges of incorporating data structures from often very complex ERP systems into architectures are well known by Enterprise Architects and Data Managers alike. The sheer number of database tables is enough to cause headaches, and coupled with customizations and multiple languages it has a real impact on the effectiveness of your Enterprise Architecture.

We’ll demonstrate how to surface and exploit data models from large, customized systems from SAP® and Oracle® using automatic discovery and intuitive scoping tools.

Presented by Martin Owen (CEO, Corso), Nick Porter and Roland Bullivant (Silwood Technology)
ODdata - the access protocol for a web-based information server

Provides a generic way to organize and describe the data structure of any remote data store as a logical data model.

- An **Entity Type** (Customer, Employee, etc.) is a data structure type consisting of named and typed Properties and with a key.
- An **Entity** is an instance of an Entity Type.
- An **Entity Key** (CustomerId, OrderId etc.) is formed from a subset of Properties of the Entity Type.
- An **Association** defines a relationship between instances of Entity Types (for example, Employee WorksFor Department).
- An Association can be 1-to-1 or 1-to-many, uni-directional or bi-directional.
- A **Navigation Property** is property of an Entity Type bound to a specific association, which can be used to refer to associations of an entity.
Using OData,

- the physical data structure of a remote data server is its own business.
- All that matters to a client is that data server returns a logical data model in reply to a request saying "get meta data".
- The client can then proceed to invoke create, read, update and delete operations on entities in that data model.
### Business data model

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Logical</th>
<th>Physical</th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Business Data Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logical Data Model</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Physical Data Schema</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Store</td>
<td></td>
</tr>
</tbody>
</table>

A conceptual data model that defines entities a business wants to remember, regardless of computing.
TOGAF: Conceptual Data Diagram (aka business data model)

► to depict the relationships between critical data entities within the enterprise.
► developed to address the concerns of business stakeholders.
Difficult to draw

Systems are about roles rather than humans

Enterprise/Business/Conceptual Data Model (MIT)
An enterprise might have scores of overlapping Logical Data Models

Impractical to build a Business Data Model with relationships between the entities

INSTEAD

List important and common data entities and group them under headings

To help people to consider and describe data in new systems as they arise

Products
- products, services

Properties
- maintained resources, offices, vehicles, assets

Promotions
- campaigns, adverts, mailings

Processes
- transactions, events, orders, payments

Places
- areas, invoicing and delivery addresses

Pipes
- routes, networks

Parties and people
- customers, suppliers, organisations, employees

Points in time
- calendar, dates, times

Pounds and Pennies
- accounts, budgets, currencies

Papers
- documents
“Dear graham,

Today's business users are accessing data from more sources than ever before which increases potential duplicate records, outdated information and allows for simple mistakes to slip through the cracks.

More and more organisations are realising the negative impact that inaccurate and duplicate records cause within daily operations, and the expense of having to eliminate and manage this data.

Addressing duplicate records efficiently requires a clear strategy and the tools to carry this out, such as:

1. Identifying all internal systems that may contain duplicate records within the organisation;
2. Establish rules that define what a duplicate record is;
3. Defining actions that occur for duplicate data;
4. Listing all duplicate records based on the criteria;
5. Building real-time or scheduled processes to eliminate duplicate records re-entering the systems.”
Data dissemination view/matrix

Conceptual

<table>
<thead>
<tr>
<th>Data Entity</th>
<th>CRM</th>
<th>ERP</th>
<th>Billing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>M</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Sale</td>
<td>M(1)</td>
<td>C</td>
<td>M(2)</td>
</tr>
<tr>
<td>Invoice</td>
<td></td>
<td></td>
<td>M</td>
</tr>
</tbody>
</table>

Logical

Physical

Real

Data dissemination view/matrix

Business Data Model

Data dissemination view/matrix

Data Entity

CRM

ERP

Billing

Customer

Sale

Invoice

Logical Data Model

Physical Data Schema

Data Store

Copyright Avancier Ltd 2008-2016
TOGAF: Data Dissemination Diagram

- to show the relationship between data entity, business service, and application components.
- shows how the logical entities are to be physically realized by application components
- allows effective sizing to be carried out and the IT footprint to be refined.
- can indicate the business criticality of application components
- may show **data replication and application ownership** of the master reference for data.
- can show two copies and **the master-copy relationship between them.**
AM: Data Dissemination diagram: illustration

- Tabulate data entities against data components (sources / stores)
- Define which data component is the master for each entity.

<table>
<thead>
<tr>
<th>Application database</th>
<th>Data entity</th>
<th>CRM database</th>
<th>ERP database</th>
<th>Billing database</th>
<th>Data warehouse</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Customer</td>
<td>Master</td>
<td>Copy</td>
<td>Copy</td>
<td>Copy</td>
</tr>
<tr>
<td></td>
<td>Sale</td>
<td>Master (1)</td>
<td>Copy</td>
<td>Master (2)</td>
<td>Copy</td>
</tr>
<tr>
<td></td>
<td>Invoice</td>
<td></td>
<td>Master</td>
<td></td>
<td>Copy</td>
</tr>
</tbody>
</table>

(1) until Order Closed  
(2) after Order Closed.

- Note that attributes of an entity may be mastered in different data stores.
- Gaps (columns and rows with no entries) may indicate where further analysis is needed.
Data store models

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Business Data Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
<td>Logical Data Model</td>
</tr>
<tr>
<td>Physical</td>
<td>Physical Data Schema</td>
</tr>
<tr>
<td>Real</td>
<td>Data Store</td>
</tr>
</tbody>
</table>

Data dissemination view/matrix
Aside: More data model layers?

- Eskil Swende has related Data Models to the Zachman Framework

<table>
<thead>
<tr>
<th>Abstraction</th>
<th>Question</th>
<th>What</th>
<th>How</th>
<th>Where</th>
<th>Who</th>
<th>When</th>
<th>Why</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Definition</td>
<td></td>
<td>Overall BDM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Representation</td>
<td></td>
<td>Detailed BDMs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification</td>
<td></td>
<td>LDMs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration</td>
<td></td>
<td>PDMs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instantiation</td>
<td></td>
<td>Data stores</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data flows – data in motion

- Businesses have to move information from one place to another, between business actors and data stores.

- Data architects are concerned with the capture and transport of information in data structures.

- Concerns include:
  - The sources, destinations and contents of data flows
  - Physical data flow forms
  - Data flow format standards
  - Semantic interoperability
## Data flow models

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Defines the ideal or common types for data items in data flows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
<td>The proper form to define the logical structure of a data flow is a regular expression</td>
</tr>
<tr>
<td>Physical</td>
<td>A data flow’s content can be defined in various forms</td>
</tr>
<tr>
<td>Real</td>
<td>At the bottom-level of a communication stack is the physical medium wires, Microwaves, Sound waves</td>
</tr>
</tbody>
</table>

- **Canonical Data Model**
- **Data Flow Structure**
- **CSV, JSON, XML**
Data qualities

- Meta data = data about data

- Data quality concerns include
  - Data types
  - Data standards
  - Data confidentiality, integrity and availability (CIA)
  - Data owners and stewards