The Logical Data Model in Agile and Model-Driven Development

Session Code ET01
Larry Burns has worked in IT for more than 25 years as a database developer, application developer, consultant and teacher. He was a contributing author and editor of DAMA International’s Data Management Body of Knowledge (DAMA-DMBOK), and writes a quarterly series of articles for TDAN.com.

He has been an instructor and advisor in the certificate program for Data Resource Management at the University of Washington in Seattle. He is also the author of "Building The Agile Database", published by Technics Publications (www.technicspub.com).
In today's Agile software development environment, the importance of a good logical data model is often overlooked, or derided as an example of "Big Design Up Front" (BDUF). Agile DBA Larry Burns will explain how the logical data model can be used as an Agile model, and can actually drive application development in a model-driven development (MDD) process. He will also explain the importance of keeping the logical and physical data models separate and distinct, and describe how each of these models are developed in an Agile project.
Today’s discussion will cover:

— What is a logical data model?
— What is a physical data model?
— What is Model-Driven Development (MDD)?
— How can the logical and physical models be used in MDD?
— What is an Agile model?
— How can the logical and physical models be used in Agile development?
First, we need to understand what the logical model is (and is not):

**The logical model IS:**

- The business requirements view of data
- Contains business data definitions, rules, relationships, and constraints for an area of the business
- Changes only in response to changes in business data or business rules (or our understanding of them)
- Implementation-independent and application-neutral; supports multiple implementations and uses of data
What is a logical model?

- Database (relational)
  Application 1

- Database (hierarchical)
  Application 2

- Database (object)
  Application 3

- XML Schema (EDI)

- End-user reports
What is a logical model?
What is a logical model?

- Database (relational) Application 1
- Database (hierarchical) Application 2
- Database (object) Application 3
- XML Schema (EDI)
- End-user reports

The Logical Data Model in Agile and Model-Driven Development
What is a logical model?
What is a logical model?

**Logical Model**

- Database (relational) Application 1
- Database (hierarchical) Application 2
- Database (object) Application 3
- XML Schema (EDI)
- End-user reports

---

The Logical Data Model in Agile and Model-Driven Development
First, we need to understand what the logical model is (and is not):

**The logical model IS NOT:**

- The database design
- A specification of what data will be implemented (or how it will be implemented)
- Application-specific
Why is this distinction important?

— Logical data models that are nothing more than the specification of the database design are NOT Agile models

— They can’t be extended to support multiple applications, or multiple uses of the data

— They make refactoring the database schema more difficult and time-consuming

— They put the cart before the horse!
In Model-Driven Development (MDD):

— Models are used not just to understand or describe problems, but to implement solutions to them

— Models are regarded as intellectual property, as important as the end product

— Models are used pervasively and continually in the development process

— Models are used as input to parameterized implementation generators
Models used in MDD:

- Capture information not captured elsewhere
- Describe a limited subset of the problem space
- Follow a standard and support design patterns
- Are unambiguous, clear and easy to understand
- Can be tested and validated
- Are used as input to parameterized application generators
<table>
<thead>
<tr>
<th>TaskID</th>
<th>IDENTITY</th>
<th>not null</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskTitle</td>
<td>varchar</td>
<td>null</td>
</tr>
<tr>
<td>TaskDesc</td>
<td>varchar</td>
<td>null</td>
</tr>
<tr>
<td>ProjectNo</td>
<td>int</td>
<td>null</td>
</tr>
<tr>
<td>ProjectMgr</td>
<td>varchar</td>
<td>null</td>
</tr>
<tr>
<td>EmployeeNo</td>
<td>int</td>
<td>null</td>
</tr>
<tr>
<td>EmployeeName</td>
<td>varchar</td>
<td>null</td>
</tr>
<tr>
<td>WeekNo</td>
<td>tinyint</td>
<td>null</td>
</tr>
<tr>
<td>EstHours1</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>ActHours1</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>EstHours2</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>ActHours2</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>etc, etc, etc...</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>EstHours7</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>ActHours7</td>
<td>smallint</td>
<td>null</td>
</tr>
<tr>
<td>OverTimeHours</td>
<td>smallint</td>
<td>null</td>
</tr>
</tbody>
</table>
Agile models must:

— Be created and updated incrementally
— Allow the entire team input into the development process
— Be used to communicate and discuss requirements
— Capture only the necessary amount of information to achieve a specific purpose
— Continually move the development effort forward
— Deliver maximum value for stakeholder investment
In Data Development, we:

- Capture a set of requirements
- Update the conceptual and/or logical model
- Update the physical model
- Update physical data structures
- Update virtual data objects
- Test and validate changes
- Repeat above steps as needed!
Example:

<table>
<thead>
<tr>
<th>Widget Category</th>
<th>Logical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Widget Category Code</td>
<td></td>
</tr>
<tr>
<td>Widget Category Description</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WidgetCategory</th>
<th>Physical</th>
</tr>
</thead>
<tbody>
<tr>
<td>WidgetCategoryCode: char(1) NOT NULL</td>
<td></td>
</tr>
<tr>
<td>WidgetCategoryDescription: varchar(255) NOT NULL</td>
<td></td>
</tr>
</tbody>
</table>

Widget Category Code values are (‘A’, ‘B’, ‘C’, ‘D’)
In Agile Data Development, we must:

— Be an integral member of the project team
— Be involved from the very beginning
— Model and develop iteratively
— Model and develop collaboratively
— Automate the process as much as possible
— Communicate proactively
— Be responsive to change
Thank You!

— Please send comments and feedback to me at: Larry_Burns@comcast.net.
Thanks!
© Copyright CA 2011. All rights reserved. All trademarks, trade names, service marks and logos referenced herein belong to their respective companies. No unauthorized use, copying or distribution permitted.

THIS PRESENTATION IS FOR YOUR INFORMATIONAL PURPOSES ONLY. CA assumes no responsibility for the accuracy or completeness of the information. TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENT “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. In no event will CA be liable for any loss or damage, direct or indirect, in connection with this presentation, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised of the possibility of such damages.

Certain information in this presentation may outline CA’s general product direction. This presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described in this presentation remain at CA’s sole discretion.

Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA may make such release available (i) for sale to new licensees of such product; and (ii) in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and if-available basis.