Agile Database Management & Development

Greg Akins
MCAD & JCP
Software Engineer
Sony Electronics, Inc

What's the Problem?

Can't we all just get along?

- Communication problems
- Political Barriers
- Competing Job Definitions
- Expectations

Agile Solution: Agile Database Techniques “Ambler, Scott”
What is Agile?

- Individuals and Interactions over processes and tools
- Working Software over Comprehensive Documentation
- Customer Collaboration over Contract Negotiation
- Responding to Change over Following a Plan

What's not Agile?

What limits our ability to adapt to change?

DBA Expectations of Developers

- Develop!
- Allow adequate time for DBA's to review
- Stored Procedures
- Data Modeling (First?)
Developer Expectations of DBAs

- Database Modeling (Whenever?)
- Database Code review
  - SQL Review
- Performance Tuning
  - Statement and Instance
- Backup and Recovery
- Security

Overlapping Responsibilities

- Data Modeling
- Stored Procedure Development
  - SQL Statement Tuning
Developers

- Develop against different database versions?
- Develop against Production databases?

Script Arsenal
- Sql Scripts
- Database Build Scripts
- Test Scripts

Enterprise Roles

- Database Administrators
- Software Developers
- Enterprise Administrators
- Enterprise Architects
Initiating Change

- Grassroots movement
- Tools that facilitate agile communication

How do you typically work with your peer dba/developers?

- Change Control
- Email

DBA Changes

- Dropping and re-adding tables
  - Data?
  - Permissions?
  - Verify?
- Script Arsenal
- Unit Tests

```java
private void testAddLogin(Login login, String username) {
    login.setUsername(username);
    login.setPassword("newPassword");
    dao.saveID(login);
    Login id = dao.getIDByName(login.getUsername());
    assertEquals(login, id);
    assertNotNull(id.getId());
}
```
Tools

- Scripts
  - Windows Scripting Host, Batch, DTS
- Command-Line Tools
  - Osq1, sqsh, console applications
- Test frameworks
  - Nunit, Junit, DBUnit
- Build Tools
  - Ant, Nant, Maven, Make

Ant Script

```xml
<target name="local_seedData"
     description="Runs the database build for seed data and constraints"
     <exec executable="osql" output="local_build_UDFs.log">
        <arg value="-U"/>
        <arg value="${db.username}" />
        <arg value="-P" />
        <arg value="${db.password}"/>
        <arg value="-S" />
        <arg value="${db.server}"/>
        <arg value="-i" />
        <arg value="${app.home}\database\build_seedData">
    </exec>
</target>
```
Now the communication begins...

- Source Control
  - SourceSafe, CVS, Subversion
- Scripts
- Database review/ Code review

Review Process

- More communication
- Review DDL
  - Change, test, promote

- Critical that we...
  - Allow time for review
  - Use clean build environment
Let's complicate things

- Legacy Changes

Altering an existing database

- Developers shouldn't be working against a production database
- Scripts should recreate development database everytime and populate with appropriate test data
- Production integration should be tested ahead of time
- Production integration should be a separate set of scripts

Changing a production database

- Tools to produce a “diff”
  - Red Gate, AdeptSql
- Create “alter”scripts for each product release
- Rollback scripts, Database Lineage...
- This is where developers really need DBAs
  - Seems like extra work for DBAs... But will make life much easier over the long term.
Database Abstraction

- Object Relational Mapping
- Data Access Layer
- Separation of Concerns

Object Relational Mapping

- Object and Relational concepts are different
- Typically Object -> RDBMS is not a one to one mapping
- Even if it is, there are RDBMS doesn't implement behavior
Mapping Tools

Tools that facilitate ORM

- Hibernate, Ibatis, Toplink, etc...
- Object Mapping, SQL Mapping, Code generator

<statement id="getEmployee" resultClass="Employee">
    select emp_id as id dept_id as deptId, Name...
    from Employee
</statement>
<statement id="insertEmployee" >
    insert into Employee values (#id#, #deptId#, #name#...
</statement>

Review Process

- Allow the DBA’s to easily view DML and make appropriate changes
  - Change, test, promote

- Critical that we...
  - Allow time for review
  - Use clean build environment
Continuous Integration

- Automated Builds in a clean build environment
- Immediate Feedback
- Verifiable results, and consistent metrics

- CruiseControl, DamageControl, Gump

- Allows DBAs & Developers to make changes at their own pace

Architecture & Design

- Patterns
- Reference Architectures
- Anti-Patterns
Architectural Decisions

- The best architecture is the one in which the most likely changes are the easiest to make

Grapical Database Modeling

- Getting Models to Code
  - ErWin - forward generates DDL
  - Visio - forward generates DDL
  - AndroMDA xmi -> database

- Model “keeper” - One person to maintain the model
- Central Model – Everyone updates the model
  - Use the model to generate DDL
The Joel Test (6/12)

- Do you use source control?
- Can you make a build in one step?
- Do you make daily builds?
- Do you have a bug database?
- Do you fix bugs before writing new code?
- Do you have an up-to-date schedule?
- Do you have a spec?
- Do programmers have quiet working conditions?
- Do you use the best tools money can buy?
- Do you have testers?
- Do new candidates write code during their interview?
- Do you do hallway usability testing?


Agile Database Techniques

Do Agile Database Techniques make sense?
Planning for Testing

- Estimate Testing time appropriately
- Plan for all types of tests
- Build commitment to ensure tests are carried out
- Detect previously undetected problems
- Distinguish and prioritize important tests
- Ensure that important tests are run, and run early

Types of Testing

- Unit
- Check-in
- Functional
- Build Verification
- Regression
- Configuration
- Integration
- Load
- Documentation and Help file tests
Testing Patterns

- Separate Test Interface
  - White Box Testing vs. Black Box Testing
- Error Simulator
- Call Stack Tracer

Any program feature without an automated test case simply doesn’t exist
- Kent Beck

Automated Testing Tools

- code analyzers - monitor code complexity, adherence to standards, etc.
- coverage analyzers - these tools check which parts of the code have been exercised by a test, and may be oriented to code statement coverage, condition coverage, path coverage, etc.
- memory analyzers - such as bounds-checkers and leak detectors.
- load/performance test tools - for testing client/server and web applications under various load levels.
- web test tools - to check that links are valid, HTML code usage is correct, client-side and server-side programs work, a web site’s interactions are secure.
- other tools - for test case management, documentation management, bug reporting, and configuration management.
Testing Tools

- CompuWare
  - DevPartner (Code Analysis)
  - QA Center (Coverage Analysis and test execution)
- Mercury Interactive
  - WinRunner (Test Execution)
  - LoadRunner (Performance Testing)
- NUnit (Unit Testing)
- xUnit (HttpUnit)

Resources

- www.martinfowler.com/articles/evodb.html
- groups.yahoo.com/group/agilemodeling
- www.agiledata.org