Agile Delivery of BI Projects

Ralph Hughes
Ceregenics, Inc.
ralph.hughes@ceregenics.com
Topics

• Agile delivery and how it differs from waterfall methods?
• To what degree can Agile accelerate our BI/DW projects?
• How can fast-moving Agile BI/DW teams ensure quality?
• How to start small and grow Agile BI/DW?
• Does Agile transform other project aspects such as estimating and architecture?
<table>
<thead>
<tr>
<th>Pharma</th>
<th>Nextel</th>
<th>speed ↑ x4 cost ↓ x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Telecom</td>
<td>Qwest</td>
<td>Sony</td>
</tr>
<tr>
<td>Government</td>
<td>Sony</td>
<td>Wolters Kluwer</td>
</tr>
<tr>
<td>Oil &amp; Gas</td>
<td>Guidant Biomed</td>
<td>Feed the Children</td>
</tr>
<tr>
<td>Informatica</td>
<td>Lockheed Martin</td>
<td>Stanford Med Center</td>
</tr>
<tr>
<td>PL/SQL, Perl</td>
<td>Comcast</td>
<td>Bureau of Land Management</td>
</tr>
<tr>
<td>BObj, Cognos, Hyperion</td>
<td>AT&amp;T Broadband</td>
<td>speed ↑ x20 cost ↓ x14</td>
</tr>
<tr>
<td>QlikView</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Balanced Views Work Better

Coke vs. Pepsi

Cats vs. Dogs

PC vs. Mac
However... Waterfall *was* a Misconception!


In the 2nd half of Royce’s paper, “risky & invites failure”
Agile Flywheel & Its Wrapper

RUP Wrapper

- Ideation
- Initiation
- Inception
- Elaboration
- Construction
- Transition

Scrum Core

- Release Backlog (project requirements)
- Iteration Backlog (demo-day objectives)
- Story Conference
- “Sprint” (Multi-Week Development Cycle)
- Daily Cycle
- Stand-Ups
- Potentially Shippable Objects

Copyright (c) 2010, Ceregenics, Inc.
Key Innovations of Generic Agile

Put business & coders in the same room ➔ Co-located Teammates
...give them three weeks to deliver ➔ Time-Boxed Development
...let them work as best they know ➔ Self-Organized Teams
...figuring things out as they go ➔ Just-in-Time Requirements
...using known references ➔ Size-Based Estimation
...and a tough definition of “done” ➔ Test-Led Development
...that’s applied daily ➔ Automated Integration Testing
...with only two, paper tools for guidance ➔ PM-Light
<table>
<thead>
<tr>
<th>Story by Priority</th>
<th>Tasks Waiting</th>
<th>Tests Written</th>
<th>Under Development</th>
<th>Waiting Validation</th>
<th>Ready to Demo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>#1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td></td>
<td>Business provides “User Stories” (est. in story points”)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 sp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 hr / 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 hr / 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>#2</strong></td>
<td>Code the...</td>
<td>Test...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td>24 hr</td>
<td>List...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 sp</td>
<td></td>
<td>Tally...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compare...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 hr / 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>#3</strong></td>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td>12 hr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 sp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Team provides coding tasks (est. in labor hours)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PM Lite: Burndown Chart

“Tech Debt”
Agile’s Iterative Approach

- Emphasizes small efforts frequently reviewed
- De-emphasizes “getting it perfect the first time”
- “Fail fast & fix quickly”
- Recursive 80/20 rule
- Maximizes learning

Offers the best chance of success when facing significant unknowns
Agile DWBI in Action
What About....?

1. Analysis & Design?
2. Architecture?
3. Controlling Scope?
4. Quality Control?
5. Remote team members?
Piplined Delivery Squads

1. Analysis & Design

“The Pipeline” (advance one station per iteration)

Takes three iterations to fill the pipeline instead of one...

...but Product Owner soon sees high-quality packages emerge, one per cycle.
**Agile Development Is Only One Piece**

1. **Departmental or Industry Stds**
   - Requirements management gets you successfully into a project....

2. **Agile Requirements Management**
   - Agile development delivers the code quickly...

3. **Agile Development**
   - Potentially-shippable code

4. **Scope**
   - Shippable Application

5. **Quality Control**
   - ...and Agile QA gets you successfully out

**Good architecture is a pre-requisite, as with any project....**

Copyright (c) 2010, Ceregenics, Inc.
“Estimating Poker” Cards

- Cards have story points on them
- Use only the Fibonacci sequence
- Use reference stories from a past sprint

For each new story:

- Compares to previous sprints’ stories
- Coders pick cards with new estimate
- Show them simultaneously
- Discuss the outliers
- Revote till consensus
- Most stories: 3 to 5 min
Here they could see they were slipping.
“Send over a team, Mike will work with them to create the revenue data mart.”

“Go ask Mike how to add the subsidiary billings to the revenue data mart.”

“Add the 3rd party revenue flat file to the staging layer.”

“Send over a programmer, we’ll tell them what to build.”

Index card with one or two lines:

“As a <user role>, I want to <action>, so that <benefit>.”

“Reminder to have a conversation”
Problem: Business could insist IT start at any point in these chains

Solution: Be ready to start at any point and retrospectively draft the RUP docs as the project proceeds.

RUP / Traditional

Program Concept Brief
“We going to resell Verizon cell phones, and we’ll need to see the revenue by week”

Stakeholder Request
“We can’t make the partner products program profitable if we can’t see the gateway data”

Vision Document
“Give us an estimate on all the projects we’ve talked about so we can pick which ones to do next year”

Solution Use Case

Iterative

Epic

“Send over a team, Mike will work with them to create the revenue data mart.”

Theme

“Go ask Mike how to add the subsidiary billings to the revenue data mart.”

User Story

“Add the 3rd party revenue flat file to the staging layer.”

Solution: Be ready to start at any point and retrospectively draft the RUP docs as the project proceeds.

Developer Story

Application Use Case

Mike: “I know exactly how you should build the allocation module.”

Send over a programmer, we’ll tell them what to build.”
## Test-Driven Development

### 4. Quality Control, cont.

<table>
<thead>
<tr>
<th>Story by Priority</th>
<th>Tasks Waiting</th>
<th>Tests Written</th>
<th>Under Development</th>
<th>Waiting Validation</th>
<th>Ready to Demo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>#1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td></td>
<td>List...</td>
<td></td>
<td>Code the... 10 hr / 6</td>
<td>Code the... 5 hr / 2</td>
</tr>
<tr>
<td>2 sp</td>
<td></td>
<td>Tally...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compare...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>#2</strong></td>
<td>Code the...</td>
<td>List...</td>
<td></td>
<td>Code the... 8 hr / 10</td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td>24 hr</td>
<td>Tally...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 sp</td>
<td></td>
<td>Compare...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>#3</strong></td>
<td>Code the...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User needs to...</td>
<td>12 hr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 sp</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Automated Regression / Integration Testing

Nominal Data

Incremental load

Dirty Data

Incomplete

Resource Failure

Staging

Integration 3NF

Ent. Dimensional

Dept. Dimensional

Dirty Data

Incomplete

Resource Failure
5. Remote Team Members

- Consider co-located sub teams
  - Architects and solutions architects
  - Analysts and developers
- Co-locate everyone for an iteration or two
- Re-collocate them for “top-of-cycle” phases
- Invest in something better than a conference phone
  - individually miking and muting a must
  - easy and dependable web conferencing
  - web cam a nicety
- Tabloid-sized scanner and printers on both ends
  - supports fast, hand-drawn diagramming
### Starting a Team

**Option 1: Gradual Improvement**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Iterations Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>0: Generic Scrum</td>
<td>Fast 1, Slow 2</td>
</tr>
<tr>
<td>1: Pipelined Delivery</td>
<td>Fast 1, Slow 2</td>
</tr>
<tr>
<td>2: Sized-Based Estimation &amp; Release Plan</td>
<td>Fast 2, Slow 4</td>
</tr>
<tr>
<td>3: Reference Model</td>
<td>Fast 2, Slow 4</td>
</tr>
<tr>
<td>4: Test-Led Development</td>
<td>Fast 3, Slow 6</td>
</tr>
<tr>
<td>5: Requirements Mgt &amp; Automated Testing</td>
<td>Fast 3, Slow 6</td>
</tr>
<tr>
<td><strong>Total Iterations</strong></td>
<td><strong>Fast 12, Slow 24</strong></td>
</tr>
</tbody>
</table>

**Elapsed Time** (2- and 3-week iterations)

- **Fast**: 24 – 36 wks
- **Slow**: 48 – 72 wks

---

**Option 2: Fast, Test-Driven Cultural Change**

- 3: Reference Model
  - Option 2: 2, Fast 4, Slow 4
- 4: Test-Led Development
  - Option 2: 3, Fast 6, Slow 6

---

*Note: The table outlines the stages of team development, the number of iterations required for each stage, and the total elapsed time for both fast and slow implementations.*
Agile vs. Really Tough DW Challenges

Situation:

• Team asked to add the newly acquired company’s billing data to the warehouse
• Don’t know the data’s structure, attributes, or quality
• Management wants to know how long will it take and how much is it going to cost...
• ...so they can decide what degree of integration they’ll sponsor
There are no silver bullets...

...but monsters are real.
Response:

- Agile doesn’t mean you can’t use all your smarts and tools!
  - *Do* profile the data, calculate a ROM estimate, etc.

- “Don’t know the data’s structure, attributes, or quality....”
  - Agile is the best way to proceed when facing a large unknown
  - We get started after one-fifth as much analysis (i.e., 80/20 rule)
  - We visualize & demo data as we go, for developer and business
  - The problems will bubble up to the surface much faster

- “Management wants to know how long will it take and how much”
  - Give us two iterations against the real problem, and we’ll give you an estimate based upon real knowledge, not speculation
Summary: If it’s going to be “mission impossible”
...tackle it with Agile,
...because disaster will happen in iteration 1 or 2
...rather than leaving all the problems to suddenly appear in user acceptance testing.

Agile can’t boil the ocean any better than waterfall can, but it fail to do it faster.
Agility through Back Filling the Architecture

1: Staging
2: Historical Archive
3: Integration
4: Dimensional
5: OLAP
6: Department
7: End User

Access
Trends
360° Vision
Single Version of the Truth
Dashboards

1% of data
5% of data
10% of data
25% of data
Full data

Copyright (c) 2010, Ceregenics, Inc.,
www.ceregenics.com
Agile Tools Trims DW Architecture

Front End Tools

- Presentation
- Integration
- Staging
- Source

Data Warehouse Appliances

Change Data Capture

Maximize the objects not built

ETL $$$

Source Archive

(c) Ceregenics 2009 29
Summary

• High-level view of how Agile DWBI works
• How Agile DWBI is reducing time-to-market
• Estimating and quality get better under Agile
• Start-up path for a DWBI team or shop
For More Information

Ralph Hughes
Ceregenics, Inc.
ralph.hughes@ceregenics.com
999 18th St., Ste 3000,
Denver Co 80202
720.951.2100

Available via
Amazon.com and Barnes & Noble
Advanced Topics

• Agile & Project Risk
• Agile Resource Planning
• Agile Architecture?
• Bidding an Agile Project
Agile & Project Risk

Risks

- Poor business requirements
- Misunderstood requirements
- Needless requirements
- Institutionalized scope creep
- Under-employed developers
- Untested product

Mitigation

- Requirements Management via
  - Early starts & task board
  - Iterative & automated testing
<table>
<thead>
<tr>
<th>Further Risks</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rework</td>
<td>Team leaders’ docs developed during iterations -1 and 0:</td>
</tr>
<tr>
<td></td>
<td>• Vision Doc</td>
</tr>
<tr>
<td></td>
<td>• Initial Data Model</td>
</tr>
<tr>
<td></td>
<td>• Initial Ref. Model</td>
</tr>
<tr>
<td>ETL sensitive to data model changes</td>
<td></td>
</tr>
<tr>
<td>Refactoring for re-usability</td>
<td></td>
</tr>
<tr>
<td>Team disruption</td>
<td></td>
</tr>
<tr>
<td>Re-assignment and production support</td>
<td></td>
</tr>
<tr>
<td>Geographically-remote teammates</td>
<td></td>
</tr>
<tr>
<td>Possible if performed carefully</td>
<td></td>
</tr>
</tbody>
</table>
Advantages of Agile

- Size-based estimation fast & more accurate
- 20% LOE estimates easier to derive
- Faster to spin-up projects & minimize bench time
- Staff planning becomes “We run three teams, that’s what we do”
- Agile is fast to train
- Quick, thorough, and standard means to manage staff augmentation

Disadvantages of Agile

- Requires “agile” temperament, esp. in leadership positions
- Greatest velocity from consistent team rosters
- Emphasizes co-located teams
Bidding an Agile Project

Challenges

• Need 2 to 3 iterations to establish a velocity
• Product owner in command of backlog

Possible solutions

• Two-part bids: “Establish Velocity” and “Remaining Deliverables”
• Reference Architecture
• “Reference” Agile team to backlog and story point a new project
• Analyze as waterfall and reduce proportionally for Agile approach
• Requirements management to constrain backlog churn
• Any new requirement affecting scope, budget, or duration → CR
• Customer’s ability to conclude early can lead a fixed-price project to T&M
  ...puts you on the same side of the table with them
Agile Architecture?

Concerns

- “Incremental optimization” can equal “muddling through”
- “How to avoid an Apollo 13-type situation?”

Mitigations

- Wrapper process includes requirements management
- Architecture comes from *departmental standards*
- New & smaller programs can externally acquire an architecture
- Reserve 20% of team bandwidth for non-functional requirements
Req. Mgt: Hierarchy of DWBI Needs

Prediction

Analysis

Research

Reporting (incl KPIs)

Data Access

Copyright (c) 2010, Ceregenics, Inc.
Agile’s Five-Levels of Planning

Product Vision
- Vision Statement
  - Product Owner & Execs, 1 to 2 x / yr

Product Roadmap
- Product Roadmap (calendarized)
  - Product Owner & Team; 3 to 4 x / yr

Release
- Release Backlog (user stories)
  - Product Owner & Team; every 3 to 4 weeks, Iteration Backlog (user stories)

Iteration
- Tasks and Burndown Review
  - Team; every day

Daily
- (three standard questions)
Agile is Evolution – Not Revolution

Waterfall

T-Shirt ➔ ROM ➔ Analysis & Design ➔ Build ➔ Intgrn Test ➔ Systm Test ➔ Deplo

Agile

T-Shirt ➔ ROM ➔ 1 ➔ 2 ➔ 3 ➔ 4 ➔ 5 ➔ System Test ➔ Deplo

Dept Arch

Here the project “out drives its headlights”

Copyright (c) 2010, Ceregenics, Inc.
Better Estimates: Reference Model