Can a Business Case Be Agile?

Peter Johnson, CBAP
Business Value Project, LLC

@PJohnsonCBAP
Tour Guide

Oxymorons

• ‘Waterfall risk’ and ‘Agile governance’

Value Imperative

• Failure rates and features used compared with IT performance and main challenge

Solution: Agile Business Case

• Value stream with estimating and funding cycles
• Use Case 2.0 roll-ups
• Business Case format and case history
Traditional Waterfall

Origins
• Structured programming
• Quality process control

BRUF Management
• Project control (‘Mike’s notes’)
• Internal and external audit
• Lead projects using project metrics
• Sign-offs and stage gates
Waterfall Business Case

- Project charter
- Funding approval
- Variance from budgeted cost and schedule
Waterfall Paradox

Governance perceived as ‘Low risk’

Earned value ‘Post-mortem’
Traditional Agile
Agile Paradox

Perceived as ‘Higher risk’

• Lacking pre-project budget and scope sign-off
• ‘Time’ as possible business constraint
• No Project Manager
• No BRUF, no guff
• Difficult to scale
• Culture shock
Greatest Concerns against Adopting Agile

- Lack of up-front planning: 33%
- Loss of management control: 33%
- Management opposition: 32%
- Lack of documentation: 30%
- Lack of predictability: 28%
- Lack of engineering discipline: 25%
- Dev team opposed to change: 23%
- Quality of engineering talent: 16%
- Reduced software quality: 15%
- Regulatory compliance: 14%
- Inability to scale: 14%
- No concerns: 14%
Agile Governance Proposal

Benefits perceived with ‘Business Agility’

- Faster decision making
- Better business processes
- More competitive

“… A positive relationship in the use of an agile approach with the projects in Governance is believed to be possible.”

http://www.agilegovernance.org
Waterfall / Agile Comparison

Projects studied over 18 years

- Small improvement overall
- Stark difference by method

Waterfall Results

Features delivered and used by stakeholders: ‘Always’ ‘Often’ ‘Sometimes’ ‘Rarely’ or ‘Never’

Source: XP2002 presented by Jim ohnson, Chairman, Standish Group
Value – New IT Imperative

Shift away from internal priorities
- Improve effectiveness and cost efficiency of business processes
- Provide managers with information to support decisions

Ability to help meet business objectives
- Share knowledge, create new products, gain productivity, track revenue segments, enter new markets

Effectiveness (vs managing infrastructure)
- Drive innovation, target greatest value, beat competitors

### Most Important Initiatives to Improve IT Performance?

<table>
<thead>
<tr>
<th>Initiative</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improve business’s accountability for IT-related projects</td>
<td>45</td>
</tr>
<tr>
<td>Reallocate IT budgets to focus on critical drivers of business value</td>
<td>43</td>
</tr>
<tr>
<td>Improve overall level of talent and capabilities of IT staff</td>
<td>35</td>
</tr>
</tbody>
</table>
‘New Age’ Concepts

14 Points for Management

• Teach and initiate leadership
• Optimize effort of teams
• Eliminate management by objective
• Remove barriers that rob people of pride of workmanship
• Encourage self-improvement for everyone

Profound Knowledge

• Theory of knowledge, statistics, systems, psychology
Agile Value Stream

Product vision
- 'S2E' Roadmap
- 'Minimally marketable feature set'
- Release plan
- Product backlog

- Estimate tasks
- Plan iteration
- Balance load
- Select task
- Stabilize velocity
- Deliver incremental value
- Prioritize user stories
- Update backlog
- Estimate complexity
Agile Business Case

- Business motivation
- Initial Product Backlog:
- Envisioning
- Daily standup
- Task board
- Definition of Done
- Demonstrated Functionality:
- Burndown chart
- Information radiators
- Product Backlog items:
- Sprint ‘0/1’
- Sprint ‘n’
- Selected Sprint Backlog items:
- Self-managed teams
- Release planning
- Product Owner
Agile Business Case Adds Clarity

Governance based on value to product owner

Product Roadmap

- Schedule and Budget
- Estimating Risk

+ %

- %

Agile ‘Epics’

Time-boxing and Scaled Projects

T-Shirt Size

Story Points

Ideal Days

Burndown

Definition of Done

Sprint Review

Velocity

Sprint ‘n’

Task board

Release Plan
Evolving Product Backlog

When you use Use-Case 2.0 the use-case slices are the primary backlog items. The use of use-case slices ensures that your backlog items are well-formed, as they are naturally independent, valuable and testable. The structuring of the use-case narrative that defines them makes sure that they are estimable and negotiable, and the use-case slicing mechanism enables you to slice them as small as you need to support your development team.

The use cases are not put into the ordered list themselves as it is not clear what this would mean. Does it mean that this is where the first slice from the use case would appear or where the last slice from the use case would appear? If you want to place a use case into the list before slicing just create a dummy slice to represent the whole use case and insert it into the list.

When you adopt a backlog-driven approach it is important to realize that the backlog is not built and completed up-front but is continually worked on and refined, something that is often referred to as grooming or maintaining the backlog. The typical sequence of activities for a backlog-driven, iterative approach is shown in Figure 17.

© Copyright 2015 Business Value Project, Peter Johnson, LLC
The use cases and the use-case slices should also be ordered so that the most important ones are addressed first.

Figure 11 shows how these post-its can be used to build a simple product backlog on a white board. Reading from left to right you can see 1) the big picture illustrated by use-case diagrams showing the scope of the complete system and the first release, 2) the use cases selected for the first release and some of their slices which have been identified but not yet detailed and ordered, 3) the ordered list of slices ready to be developed in the release and finally 4) those slices that the team have successfully implemented and verified.

Figure 11 is included for illustrative purposes only, there are many other ways to organize and work with your requirements. For example many teams worry about their post-it notes falling off the whiteboard. These teams often track the state of their use cases and use-case slices using a simple spreadsheet including work sheets such as those shown in Figure 12 and Figure 13.
T-shirts to Story Points

A USE CASE AND ITS PROPERTIES CAPTURED ON A POST-IT NOTE

7. Browse and Shop

Priority: MUST
Release: 1
Size: Very Large
Complexity: High

SH舍PER

7. Select and Buy 1 Product
Flows: BF
Test: 1 Product, default payment, valid details

5

7.2 Select and Buy 100 Products
Flows: BF
Test: 100 Products, default payment, valid details

5

7.3 Support Systems Unavailable
Flows: BF, Ag, A10, A1, A12
Test: Select Product, Provide Information, Disconnect each system in between

13

SOME SLICES FROM THE USE CASE CAPTURED ON THEIR OWN POST-IT NOTES
Increments and Releases Capture Tangible Costs and Benefits

Use cases are a fabulous tool for release planning. Working at the use-case level allows whole swathes of related requirements to be deferred until the later releases. By making decisions at the use-case level you can quickly sketch out the big picture and use this to focus in on the areas of the system to be addressed in the next release.

Use-case diagrams, showing which use cases are to be addressed in this release and which are to be left until a later release, are a great tool for illustrating the team’s goals. They clearly show the theme of each release and look great pinned up on the walls of your war-room for everybody to see.

Use-case slices are a fabulous tool for building smaller increments on the way to a complete release. They allow you to target independently implementable and testable slices onto the increments ensuring that each increment is larger than, and builds on, the one before.

Principle 6: Adapt to meet the team’s needs

Unfortunately there is no ‘one size fits all’ solution to the challenges of software development; different teams and different situations require different styles and different levels of detail. Regardless of which practices and techniques you select you need to make sure that they are adaptable enough to meet the ongoing needs of the team. This applies to the practices you select to share the requirements and drive the software development as much as any others. For example lightweight requirements are incredibly effective when there is close collaboration with the users, and the development team can get personal explanations of the requirements and timely answers to any questions that arise. If this kind of collaboration is not possible, because the users are not available, then the requirements will require more detail and will inevitably become more heavyweight. There are many other circumstances where a team might need to have more detailed requirements as an input to development. However, what’s important is not listing all of the possible circumstances where a lightweight approach might not be suitable but to acknowledge the fact that practices need to scale.
Applying our recipe above, the use cases identify the useful things that the system will do. Select the most useful use case to find the most useful thing that the system does. To find the most central slice you will need to shed all the less important ways of achieving the goal and handling problems. You can do this by focusing on the story described by the basic flow. A slice based on the basic flow is guaranteed to travel through the entire concept from end-to-end as it will be the most straightforward way for the user to achieve their goal.

Estimate the slice and start to build it. Additional slices can then be taken from the use case until there are enough slices to provide this particular user with a usable solution. The same can then be done for any other use cases you need to complete a usable system.

A use-case slice doesn’t need to contain an entire flow and all its test cases – the first slice might just be the basic flow and one test case. Additional slices can then be added to complete the flow and address all the test cases. The slicing mechanism is very flexible enabling you to create slices as big or small as you need to drive your development.

The slices are more than just requirements and test cases. When we build the system in slices it is not enough to just slice up the requirements. Although use cases have traditionally been used to help understand and capture requirements, they have always been about more than this. As shown in Figure 3, the use-case slices slice through more than just the requirements, they also slice through all the other aspects of the system and its documentation.

On the left of Figure 3 you can see the use-case slice, this is a slice taken from one of the use cases shown in the next column. The slice then continues through the design showing the design elements involved, and through the implementation where you can see which pieces of the code actually implement the slice. Finally the slice cuts through the test assets, not just encompassing the test cases, but also the test scripts used to execute the test cases and the test results generated.

As well as providing traceability from the requirements to the code and tests, thinking of the slices in this way helps you develop the right system. When you come to implement a slice you need to understand the impact that the slice will have on the design and implementation of the system. Does it need new system elements to be introduced? Can it be implemented by just making changes to the existing elements? If the impact is too great you may even decide not to implement the slice! If you have the basic design for the system this kind of analysis can be done easily and quickly, and provides a great way to understand the impact of adding the slice to the system.
ValPak: Agile Enterprise Portfolio
ValPak: Agile Business Cases

### Sales Tools Revamp

<table>
<thead>
<tr>
<th>Executive Sponsor:</th>
<th>Lisa Sullivan</th>
<th>Planned Quarter:</th>
<th>2Q15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teams:</td>
<td>CMCM, CRM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Investment Theme:</td>
<td>Increase Franchise Efficiencies</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value Statement:</td>
<td>Create content library and templates for franchises to build their own tools. Eliminates multiple tools.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### New Mobile Apps

<table>
<thead>
<tr>
<th>Executive Sponsor:</th>
<th>Nancy Cook</th>
<th>Planned Quarter:</th>
<th>1Q15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teams:</td>
<td>M&amp;M, BI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Investment Theme:</td>
<td>Operations Excellence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value Statement:</td>
<td>• Create new app that improves user experience and drive greater app usage and downloads.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Update app to embrace brand (localize look and feel)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Inbound content to ensure usability to</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ValPak: Agile Value Criteria

- Business model approved
- Resources/teams available to build/execute
- Success metrics established
New Roles for ‘business analysis’

Join committed team

• Queue pre-shop resources, remote teams, vendors
• Facilitate root cause, process improvement, modeling
• Optimize stakeholder value, develop test criteria

Join product owner’s team

• PBI pruning, grooming, business case inputs and updates

Join business architect’s team

• Change strategy, funding, compliance
What Can You Commit To Regarding the Business Case?

Engage early
• Position yourself as a consultant
• Regardless of development methodology
• Lead the discussion among stakeholders on value

Develop tangible benefits aligned with cost
• Create a sustainable format for on-going discussion
• Use business case to guide decision-making

Report back
• Change Stories, challenges, success, templates,
  • Pjohnson_CBAP@verizon.net
Comments, Questions

Thank you!

PJohnson_CBAP@verizon.net

@PJohnsonCBAP