

The Big Ball of Mud

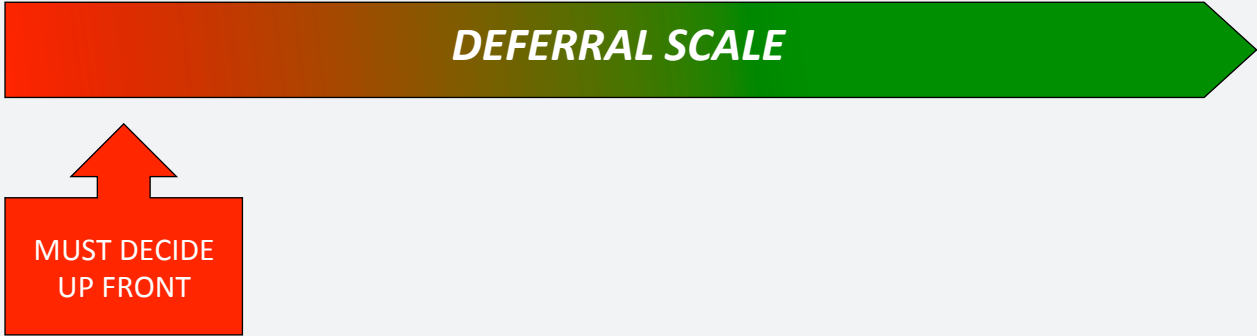
- This was originally a paper presented by Brian Foote and Joseph Yoder to the Fourth Conference on Patterns Languages of Programs (PLoP /EuroPLoP) in 1997
- They define a Big Ball of Mud as:

“... a haphazardly structured, sprawling, sloppy, duct-tape-and-baling-wire, spaghetti-code jungle. These systems show unmistakable signs of unregulated growth, and repeated, shared promiscuously system, often to the important information The overall structure of been well defined. If it beyond recognition. architectural sensibility Only those who are architecture, and, perhaps, are comfortable with the inertia of the day-to-day chore of patching the holes in these failing dikes, are content to work on such systems”



- While many successful systems are actually Big Balls of Mud [*citation needed*], such systems are usually hard to change, expensive to run and hated by everyone

Example Decision 1: Choice of Primary Implementation Language

Context	Any new software system being written in a 3GL
Decision	Which primary implementation language should be used to build the system?
Forces	<ul style="list-style-type: none"> • Language maturity and feature-richness • Existing corporate technical standards • Extent to which language is established in the industry • Availability of development and support skills
Early-Decision Risks	<ul style="list-style-type: none"> • A different choice could improve productivity • A different choice could improve the stability, scalability, security of built code
Late-Decision Risks	<ul style="list-style-type: none"> • Would have to rewrite a large amount of code in the alternative language
Decision	 <p><i>DEFERRAL SCALE</i></p> <p>MUST DECIDE UP FRONT</p>

Example Decision 2: User Interface Layout



Context	Any new software system with a substantial UI component
Decision	How should UI elements be laid out on screen?
Forces	<ul style="list-style-type: none"> • User experience and skills • Process workflows and interaction models • Physical characteristics of devices (form factor) • Availability of design, prototyping and build tools
Early-Decision Risks	<ul style="list-style-type: none"> • Little is known about screen elements so will probably have to be changed • Specifying UIs in detail early on is rarely a productive use of anyone’s time
Late-Decision Risks	<ul style="list-style-type: none"> • The users who should be involved in the decision may not be available • The UI may rely on features not available in the architecture (eg microservices)
Decision	<p>The diagram illustrates a 'DEFERRAL SCALE' as a horizontal arrow pointing to the right, with a color gradient from red on the left to green on the right. Below this arrow is a green box with an upward-pointing arrow, containing the text 'CAN LEAVE TILL LATE'.</p>



QUALITY. PRODUCTIVITY. INNOVATION.



Today's Session

Goals of Today's Session



- In this workshop you will attempt to apply the concept of Last Responsible Moment to some real-world design decisions
- We will work in groups to identify interesting design decisions from current and past projects and map each decision to the right point on the scale
 - For the "early-on decisions," we will consider whether there are any practical ways we can defer these decisions until later
 - For those decisions which fall nearer the other end of the time scale, we will look at the risks of leaving these decisions late, and practical ways to mitigate those risks
- There are two exercises:
 - In Exercise 1 we will share some interesting design decisions
 - In Exercise 2 we will brainstorm some tactics for deferring early decisions until later, and if time allows, look at the risks of leaving them late
- **To start, please form yourself into small groups**

QUALITY. PRODUCTIVITY. INNOVATION.



Exercise 1

Interesting Design Decisions

35 minutes

Exercise 1

- In this exercise we would like you to share interesting design decisions from current and past projects in your teams
- Categorise each decision on the spectrum from "Must Decide Up Front" to "Can Leave Till Quite Late"
- Think about what it is about these decisions that led you to place them on the Deferral Scale where you did
 - Balance the benefits of late decision-making against potential need for rework and replanning
 - Balance the benefits of early decision-making against the risk of making the wrong decision
- Use the proformas to record your ideas
- **You have 35 minutes for this exercise, followed by a 15-minute break**



Exercise 2

Deferral Tactics

35 minutes

Exercise 2

Group Discussion

- What sorts of decision did you talk about in Exercise 1?
- Were they “Must Decide Up Front,” “Can Leave Till Quite Late,” or somewhere in the middle?

Exercise 2

- In this exercise we would like you to consider tactics you can use for your "early-on" decisions
- Discuss whether there are ways we can architect and design systems to enable you to make these decisions later
- We have a few suggestions on the next slide to get you started
- Use the proformas to record your ideas

- If you have time, we would also like you to think about your “late” decisions, the risks of leaving them late, and how you might mitigate these risks

- **You have 30 minutes for this exercise, followed by a presentation to the rest of the group**

Some Example Deferral Tactics



Abstract	Hide the decision behind an abstraction to allow later change e.g. use a data access library
Renegotiate	Change the decision to be made by renegotiating time, scope, constraints e.g. persuade the users to allow late changes to the user interface
Redesign	Avoid the decision by redesigning your system to avoid the problematic element e.g. alter the structure to avoid connections to elements needing early decisions
Variation Point / Bind Late	Explicitly add a mechanism to allow the decision to be changed later e.g. implement late binding of components via configuration
<i>etc.</i>	

QUALITY. PRODUCTIVITY. INNOVATION.

Group Presentations and Conclusions



Thank you



Nick Rozanski

nick@rozanski.org.uk

Chris Cooper-Bland

Chris.Cooper-Bland@endava.com

Eoin Woods

Eoin.Woods@endava.com

Endava

125 Old Broad Street
London

QUALITY. PRODUCTIVITY. INNOVATION.

Appendices



Decision Proforma



Context	
Decision	
Notes	
Position on Deferral Scale	<p>The diagram shows a horizontal arrow labeled "DEFERRAL SCALE" pointing to the right. Below the left end of the arrow is a box containing the text "MUST DECIDE UP FRONT" with an upward-pointing arrow. Below the right end of the arrow is a box containing the text "CAN LEAVE TILL LATE" with an upward-pointing arrow.</p>
Deferral Tactics	

Links



Last Responsible Moment (Glenn Ballard, Todd Zabelle)

- <http://www.cp.berkeley.edu/designresearch/My%20Webs/Contents/projectdefinition.pdf>

Who Needs an Architect (Martin Fowler)

- <http://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf>

NATS National Air Traffic Control System

- <http://www.computerweekly.com/feature/A-brief-history-of-an-air-traffic-control-system>

Vasa (Swedish warship)

- [http://en.wikipedia.org/wiki/Vasa_\(ship\)](http://en.wikipedia.org/wiki/Vasa_(ship))

Big Ball of Mud (Brian Foote, Joseph Yoder)

- <http://www.joeyoder.com/PDFs/mud.pdf>

Pompidou Project (Richard Rogers)

- <http://www.dezeen.com/2013/07/26/richard-rogers-centre-po>

NHS Spine Project

- <http://www.computerweekly.com/opinion/Six-reasons-why-the-NHS-National-Programme-for-IT-failed>

Software Systems Architecture (Nick Rozanski, Eoin Woods)

- <http://www.viewpoints-and-perspectives.info>

Endava

- <http://www.endava.com/en/Services/Application-Development/Solution-and-Enterprise-Architecture.aspx>