

Organisation, Governance and Methodology

Oxford University
Software Engineering Programme
April 2021



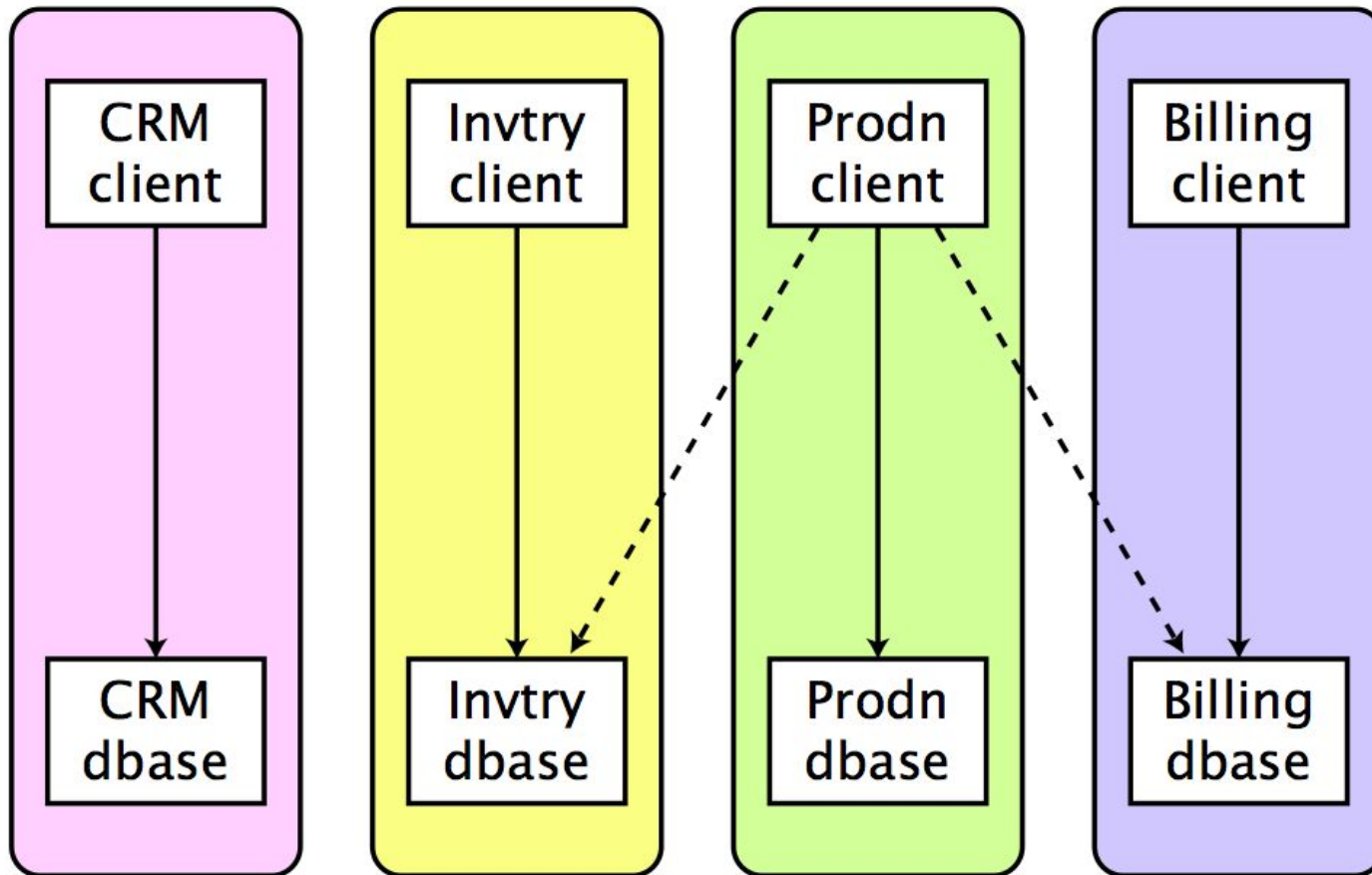
© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Contents

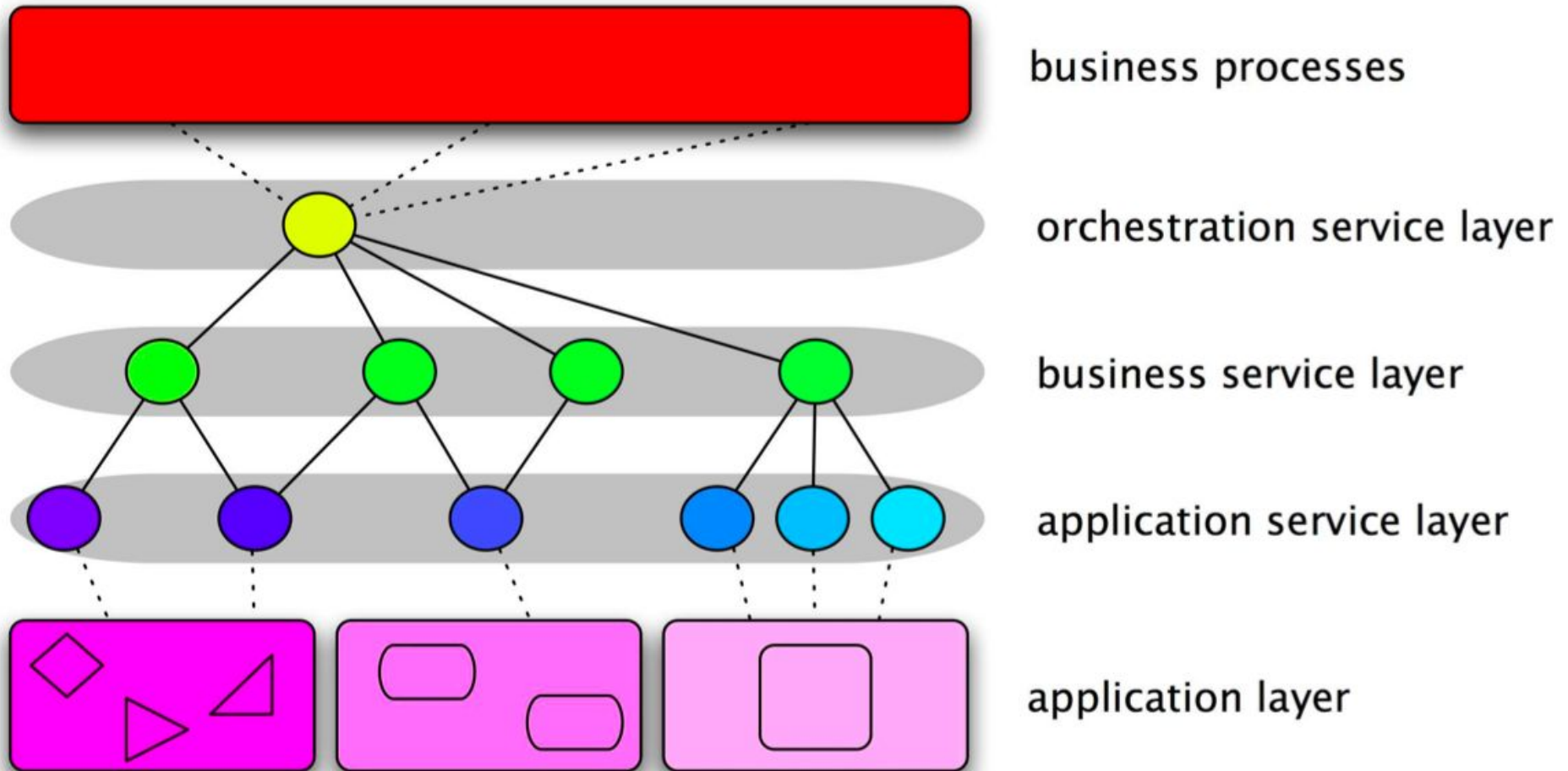
- Software Development Lifecycle
- Registries
- Design Governance
- Runtime Governance



Before SOA



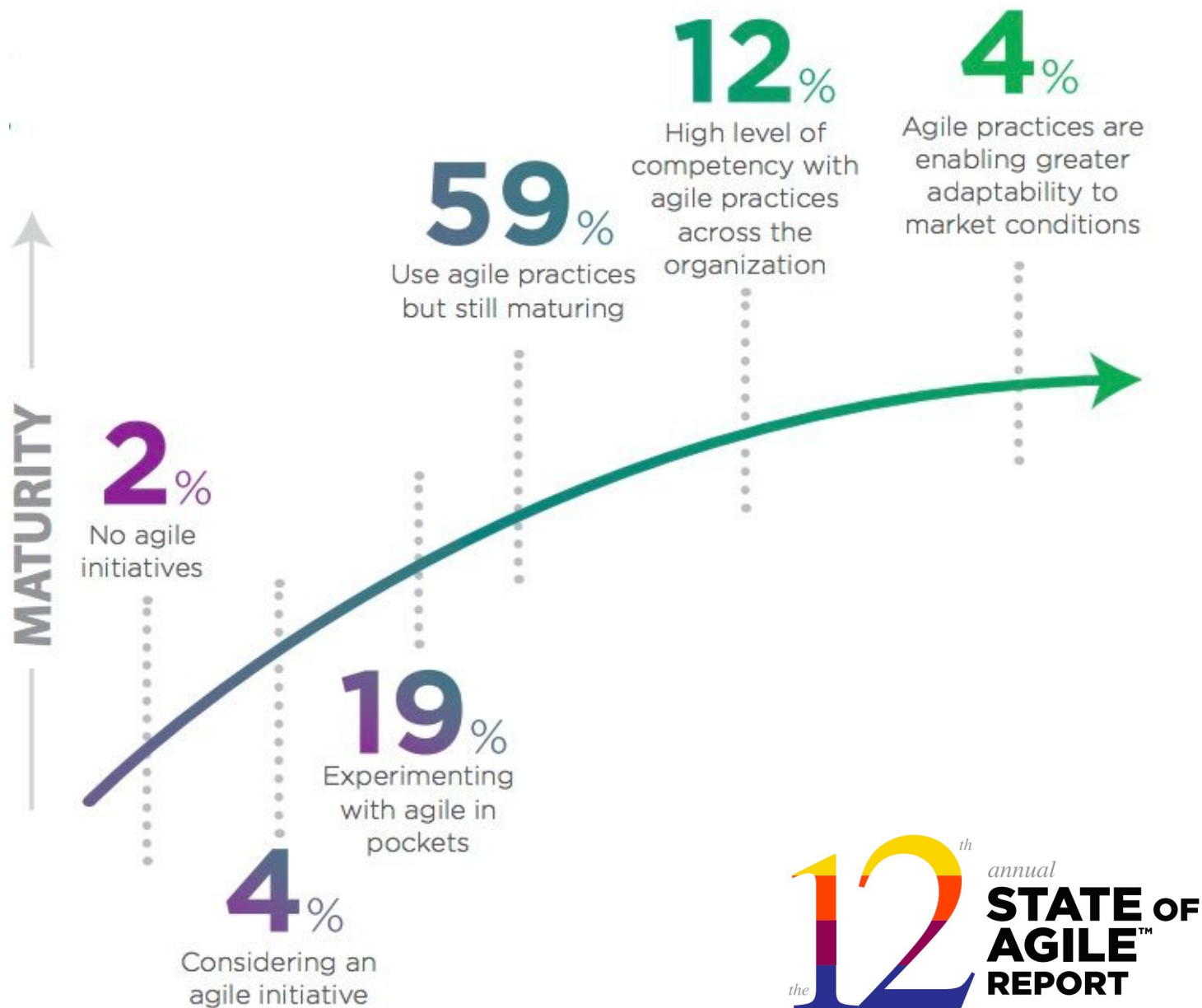
With SOA



SOA has an impact on organization

- Refactoring of fiefdoms:
 - backend departments
 - cross-domain departments – frontend departments
 - “solutions managers”
- Requires collaboration and trust
- May change the funding model
 - That will pull in resistance







© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>



Developer Flow



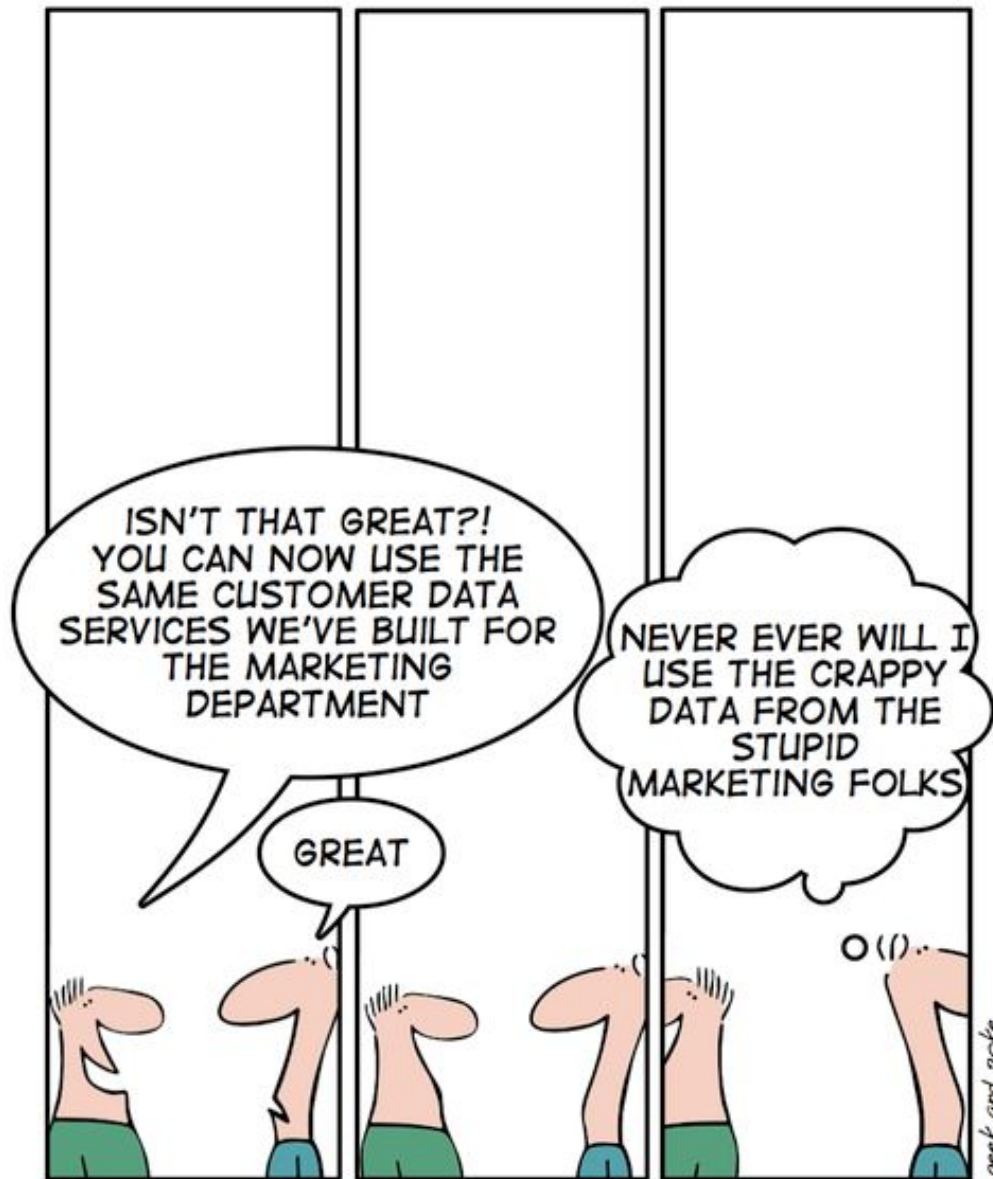
© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>



The wrong organization interrupts flow



© Paul Fremantle 2016 except where noted elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>



THE BENEFITS OF A SOA



Donnie Berkholz ✓

@dberkholz

Following



You say "center of excellence," I hear
"new silo."

8:01 PM - 23 Feb 2018

40 Retweets 118 Likes



16



40



118



Tweet your reply



Donnie Berkholz ✓ @dberkholz · Feb 23



We're creating Communities of Practice at CWT. Bringing together expertise from across the company, with people sharing practices, patterns and code.



1



2



10



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Complex processes interrupt flow

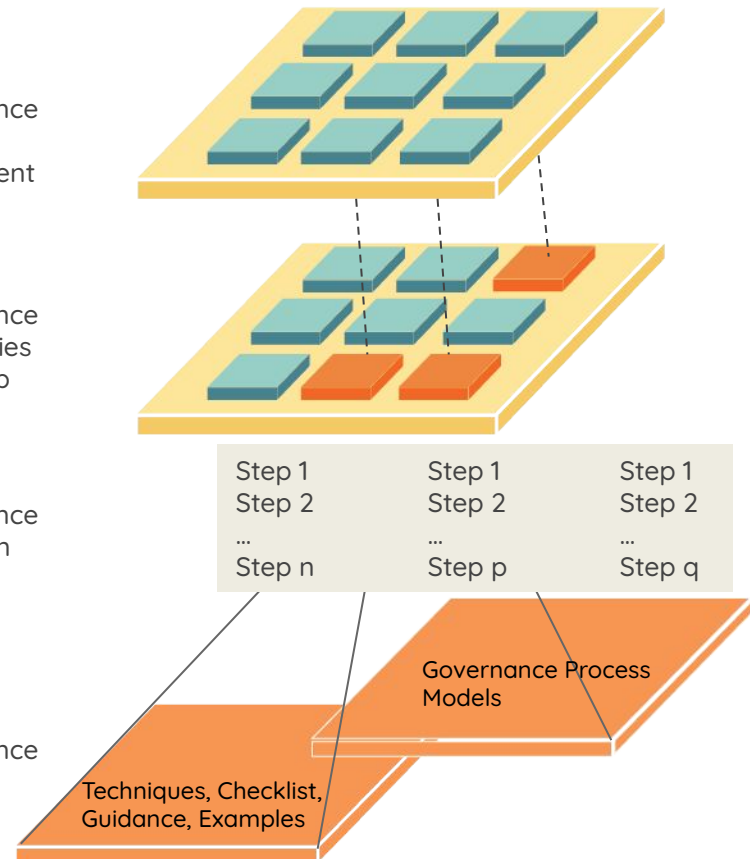
SGMM

SOA
Governance
Planning
Assessment

SOA
Governance
Capabilities
Heat Map

SOA
Governance
Transition
Plan

SOA
Governance
Assets



The Agile Manifesto

The best architectures, requirements, and designs emerge from self-organizing teams.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Deliver working software frequently with a preference to the shorter timescale.



What is a “self-organizing” team?

A team which:

- Manages its own work
- Pulls work
- Doesn't require “command and control”
- Communicates effectively with each other
- Is not afraid to ask questions
- Continuously evolves skills and capabilities

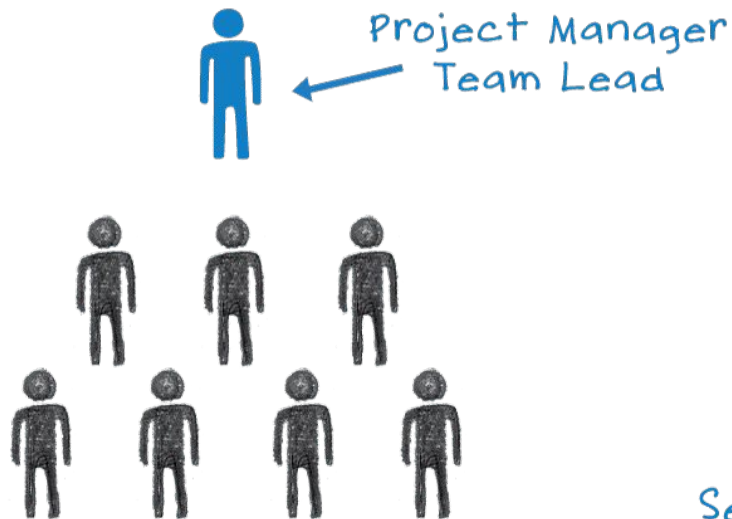
<https://www.scrumalliance.org/community/articles/2013/january/self-organizing-teams-what-and-how>



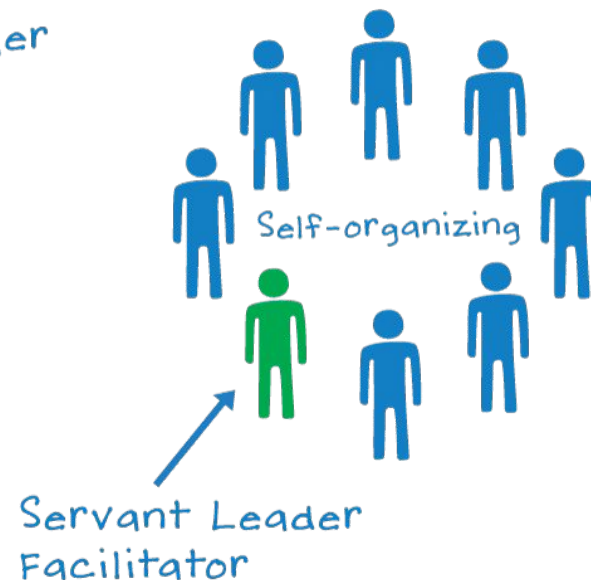
© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Self Organizing Teams

Traditional Teams



Agile Teams





© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Why individuals in larger teams perform worse

Jennifer S. Mueller*

University of Pennsylvania, The Wharton School, Management Department, 3620 Locust Walk, Suite 3014 SHDH, Philadelphia, PA 19104, United States

ARTICLE INFO

Article history:

Received 15 July 2009

Accepted 17 August 2011

Available online 29 September 2011

Accepted by Linn Van Dyne

Keywords:

Team size

Individual performance

Perceived social support

Appraisal theory

Multi-level theory

Process loss

Coordination

ABSTRACT

Research shows that individuals in larger teams perform worse than individuals in smaller teams; however, very little field research examines why. The current study of 212 knowledge workers within 26 teams, ranging from 3 to 19 members in size, employs multi-level modeling to examine the underlying mechanisms. The current investigation expands upon [Steiner's \(1972\)](#) model of individual performance in group contexts identifying one missing element of process loss, namely relational loss. Drawing from the literature on stress and coping, relational loss, a unique form of individual level process, loss occurs when an employee perceives that support is less available in the team as team size increases. In the current study, relational loss mediated the negative relationship between team size and individual performance even when controlling for extrinsic motivation and perceived coordination losses. This suggests that larger teams diminish perceptions of available support which would otherwise buffer stressful experiences and promote performance.

© 2011 Elsevier Inc. All rights reserved.

$$\frac{n(n-1)}{2}$$



Conway's Law

- Any organization that designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure.

Melvin Conway,

How Do Committees Invent?,

Datamation Apr 1968,

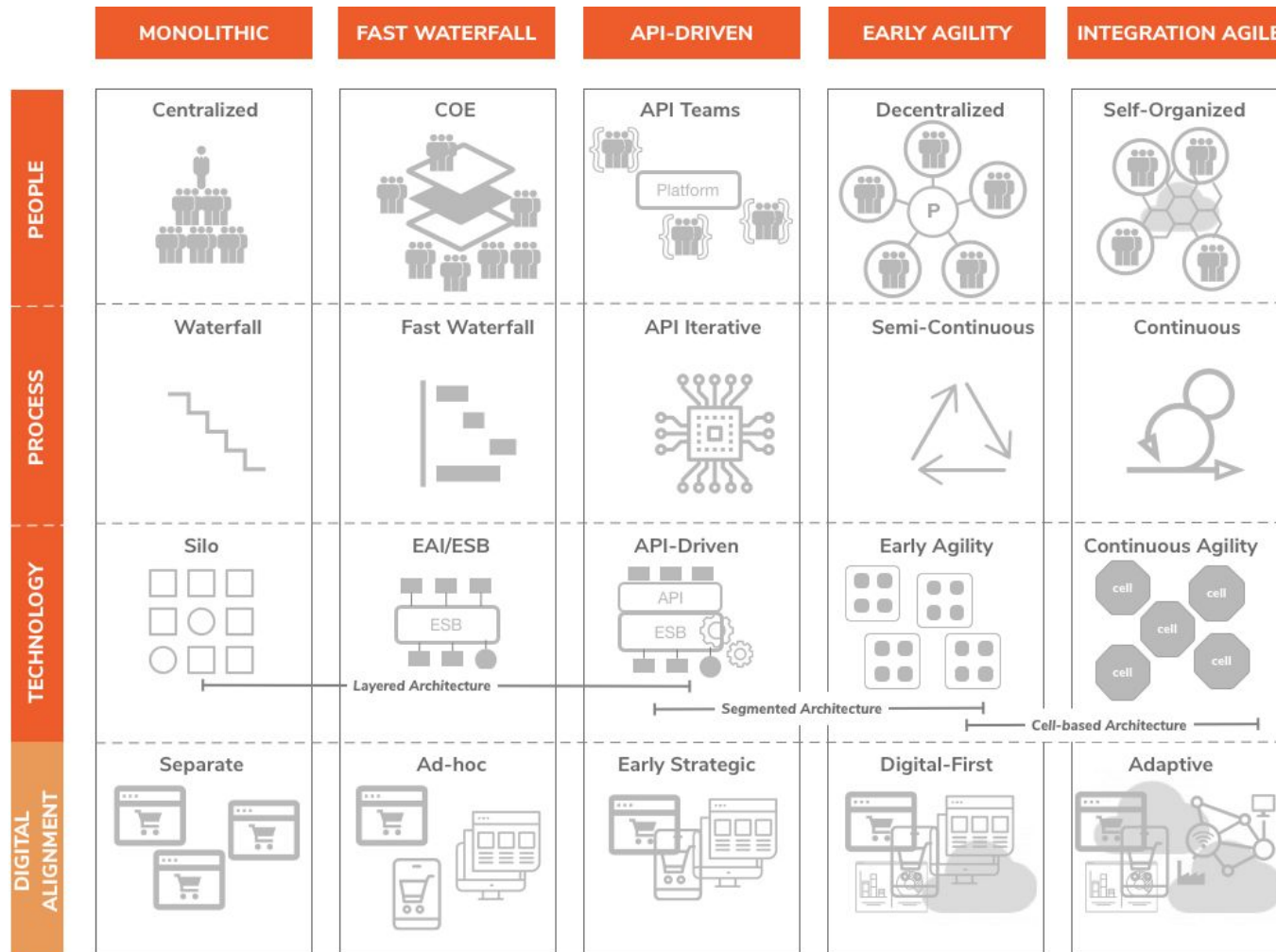
<http://www.melconway.com/law/>

- Popularized and named by Fred Brooks in
 - *The Mythical Man-Month*
- Eric Raymond:
 - “If you have four groups working on a compiler, you’ll get a 4-pass compiler.”

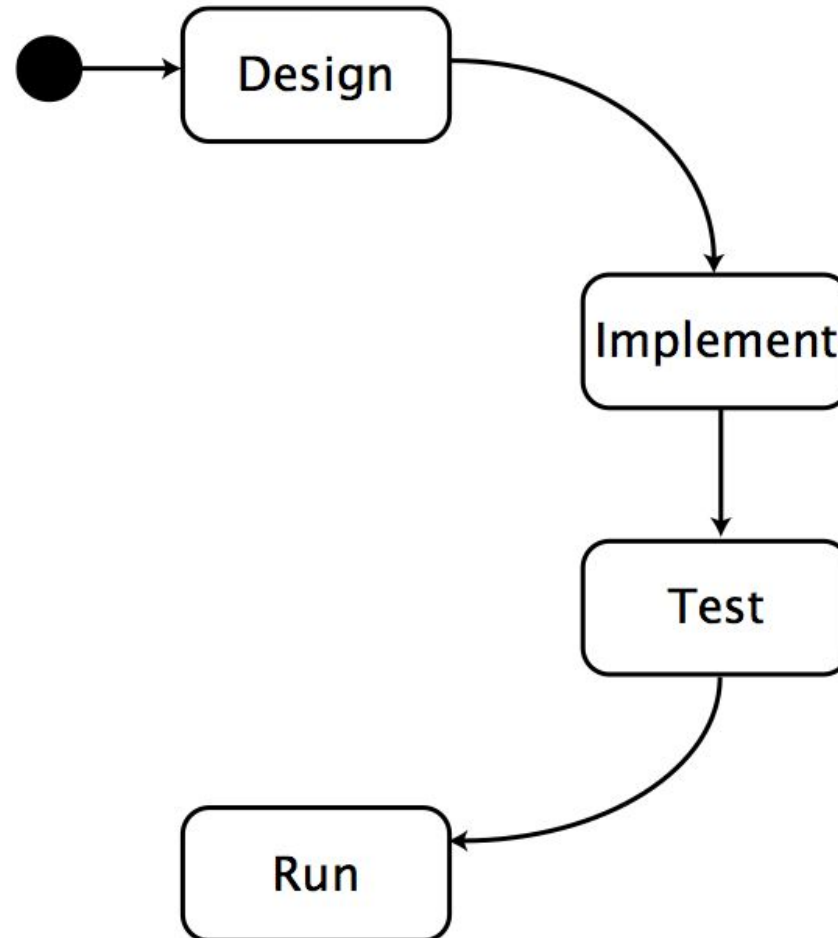


Reference Methodology

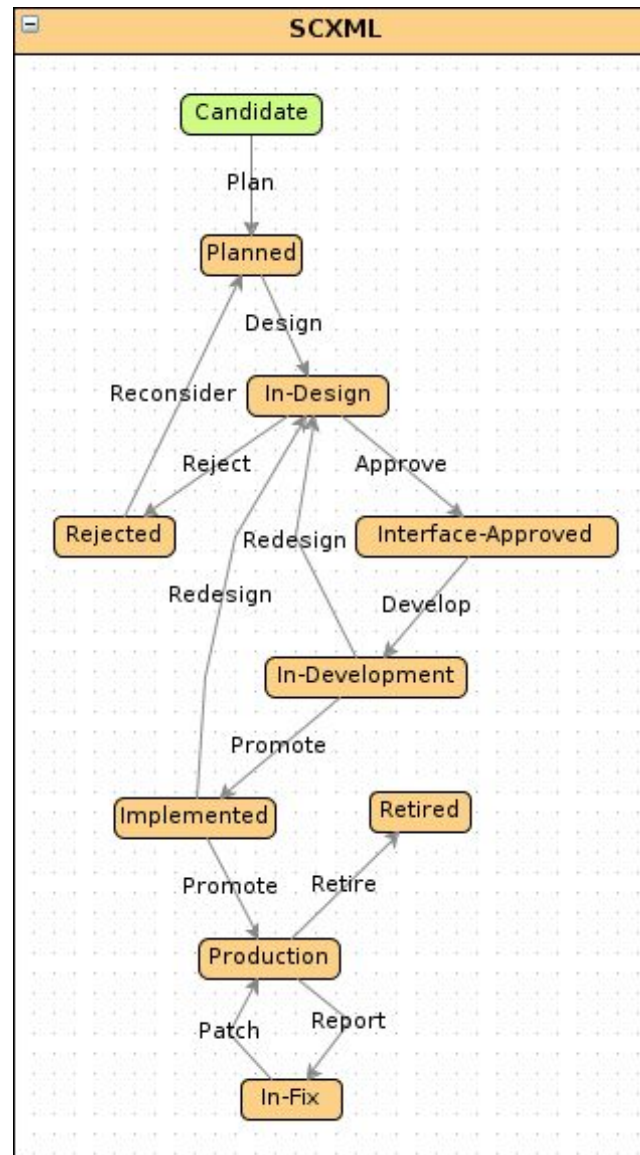
<https://github.com/wso2/reference-methodology>



Software Development Lifecycle



Not that simple!



High level governance

- *Visions, objectives, business case, funding model*
 - Why are we doing this? How will we pay for it?
 - *Reference architecture*
Fundamental decisions: preferred technology, message exchange patterns, metamodel, etc
- *Rules and responsibilities*
 - who drives and cares about issues
- *Policies, standards, formats, processes, lifecycles*
 - decide and document, in standard notations



Technical Governance

- *Documentation*
 - important for transparency; promotes non-technical issues
- *Service management*
 - repositories and registries for services and contracts
- *Monitoring*
 - conformance to policies, meeting SLAs, preparing for withdrawal
- *Change and configuration management*
 - Code lifecycle, DevOps, SOA, the intersection



Establishing a services approach

- *Developer-driven, grass-roots*
 - leads to technological experience; likely to be uncoordinated
- *Business-driven*
 - proof of concept helps adoption; limited benefit from early projects
- *IT-driven*
 - effective for infrastructure; focus on technical aspects
- *Management-driven*
 - top-down coordinated, driven by business priorities; expensive, disruptive, risky



“Shift to the Left”

- Start each project with:
 - Git / SCM
 - Build / Test
 - DevOps
 - Cloud Orchestration
 - Observability
 - GitOps
 - etc
- Then start writing code...



Questions?



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>