# Software Architecture Fundamentals Workshop

## *Part 1: From Developer to Architect*

**Mark Richards**

**Independent Consultant**

Hands-on Enterprise / Integration Architect

Published Author / Conference Speaker

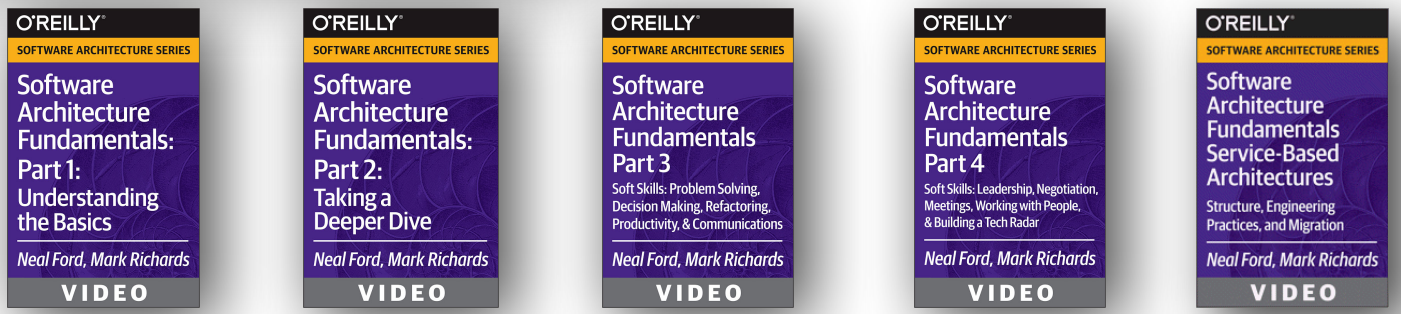
http://www.wmrichards.com

http://www.linkedin.com/pub/mark-richards/0/121/5b9

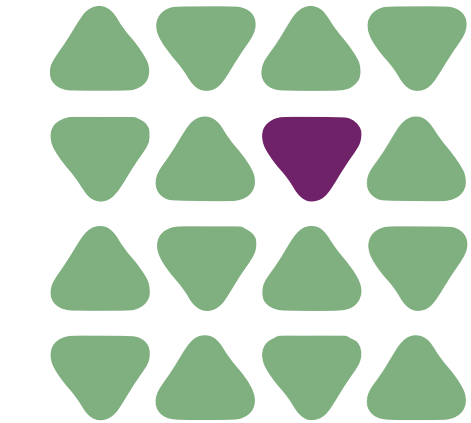

ThoughtWorks®

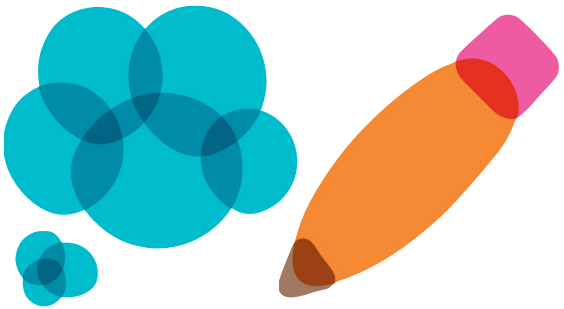
**NEAL FORD**

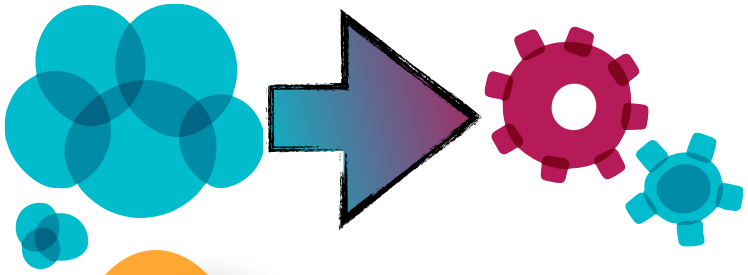
*Director / Software Architect / Meme Wrangler*

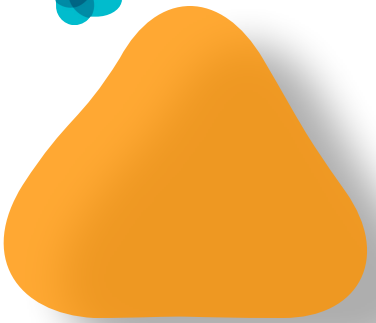


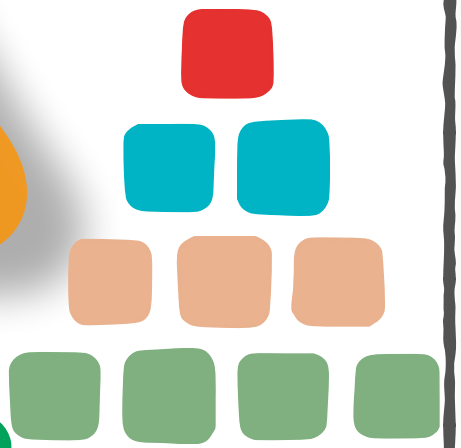
# agenda

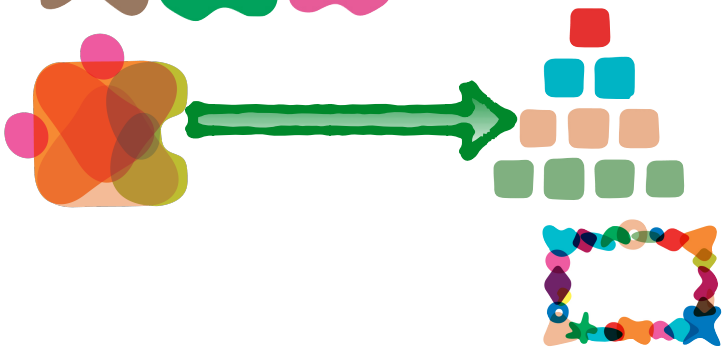


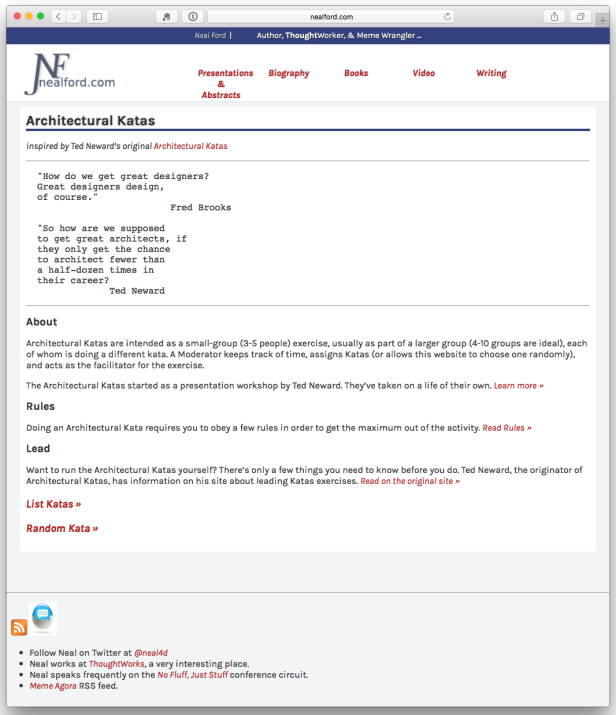













nealford.com/katas/

# Best Jobs in America

2015

CNNMoney/PayScale's top 100 careers with big growth, great pay and satisfying work.





## 1. Software Architect

NEXT

**Median pay**: $124,000
**Top pay**: $169,000
**10-year job growth**: 23%


Christopher Felpel, software architect/consultant

In the same way an architect designs a house, software architects lay out a design plan for new programs. That usually means leading a team of developers and engineers, and making sure all the pieces come together to make fully-functioning software.

**What's great:** New problems come up all the time and new technologies arise, making each day different, and keeping professionals in demand. "I'm pinged at least once or twice a week for new opportunities," said software architect Christopher Felpel. "There's just a lot of work out there, and that doesn't surprise me." --*Jillian Eugenios*

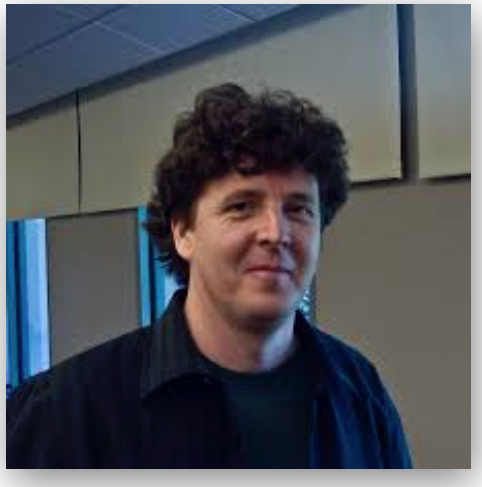
**Quality of life ratings:**
Personal satisfaction: **A** | Benefit to society: **B** | Telecommuting: **A** | Low stress: **A**

## 100 Best Jobs in America


Top 100: Full list


Top-paying jobs


Fastest-growing jobs



*“Programmers know the benefits of everything and the tradeoffs of nothing.”*

Architects must understand both.

# software architecture?

*"the highest level concept of a system in its environment. The architecture of a software system (at a given point in time) is its organization or structure of significant components interacting through interfaces, those components being composed of successively smaller components and interfaces."*

Rational Unified Process definition, working off the IEEE definition

http://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf

# software architecture?

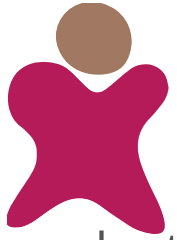
*Architecture is the highest level concept of the expert developers.*

*"In most successful software projects, the expert developers working on that project have a shared understanding of the system design. This shared understanding is called 'architecture.' This understanding includes how the system is divided into components and how the components interact through interfaces. These components are usually composed of smaller components, but the architecture only includes the components and interfaces that are understood by all the developers."*

http://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf

# software architecture?

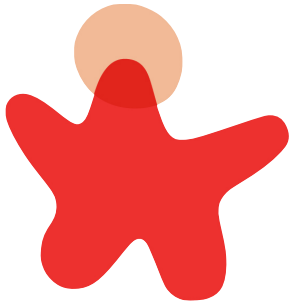
developers



product owner

*Architecture is about the important stuff.*
*Whatever that is.*

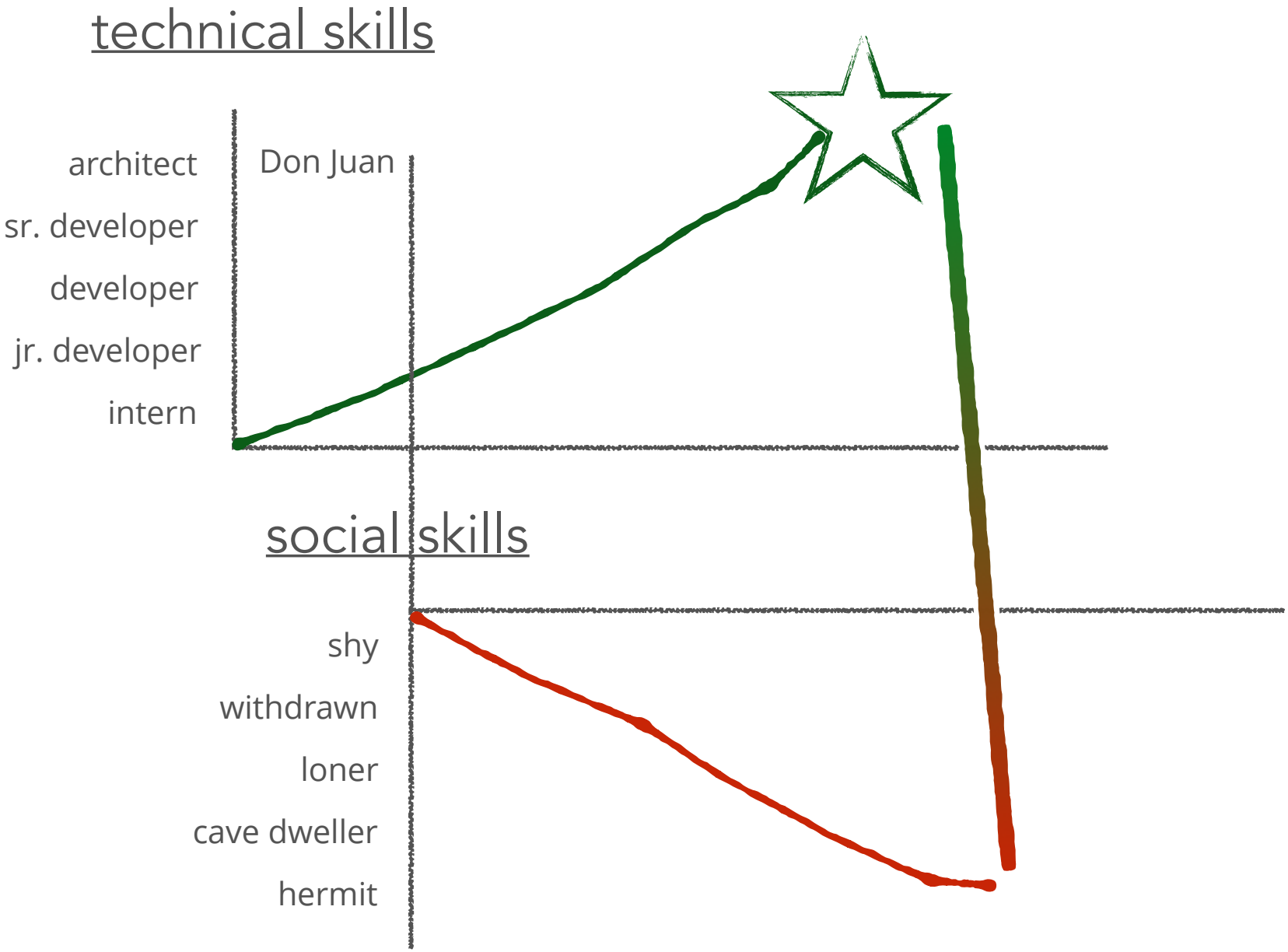
Martin Fowler



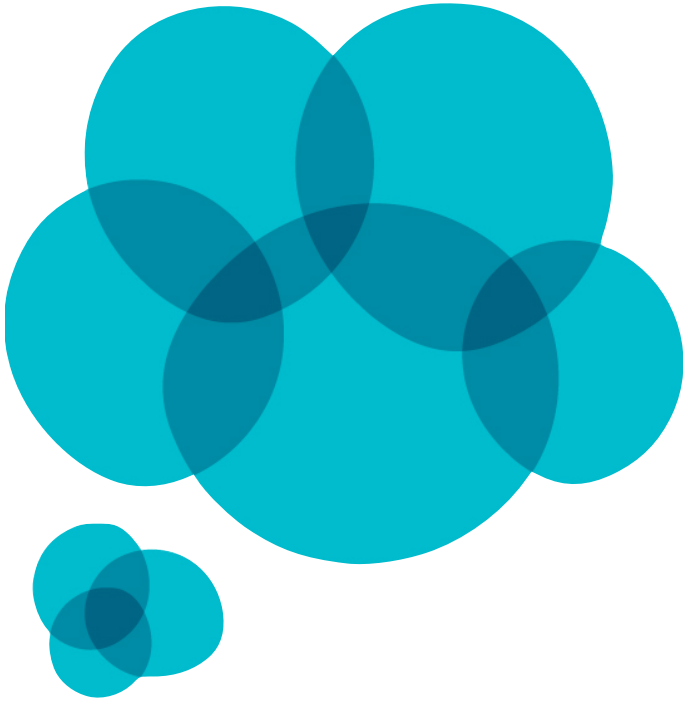
operations

http://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf

# soft skills

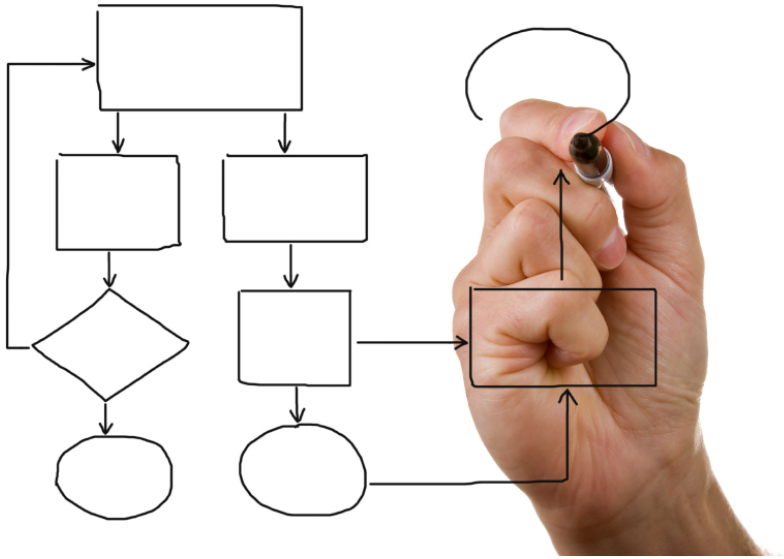

technical skills
architect
sr. developer
developer
jr. developer
intern
Don Juan
social skills
shy
withdrawn
loner
cave dweller
hermit

# Decisions

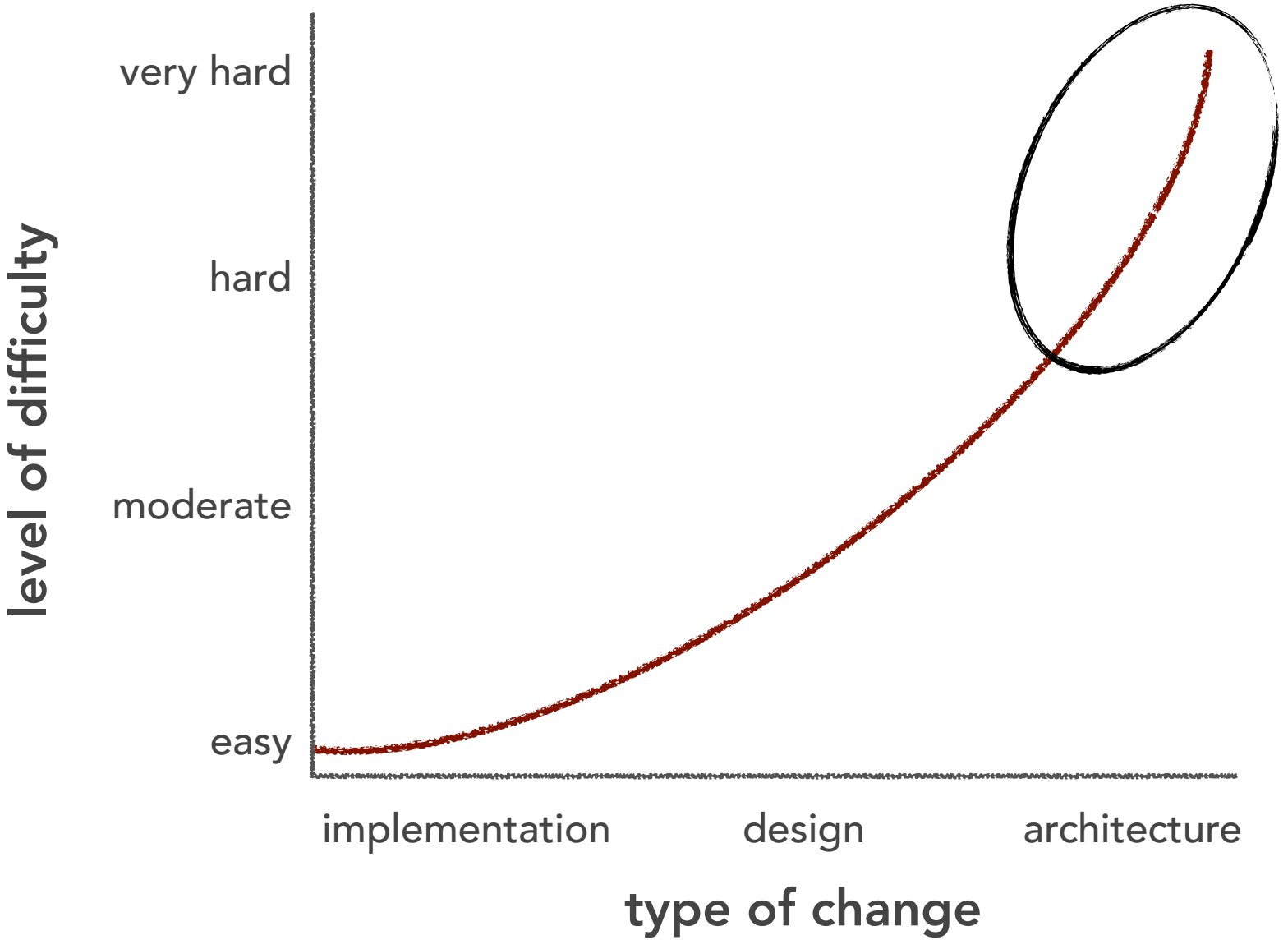


# architecture decisions

## what is an architecture decision?

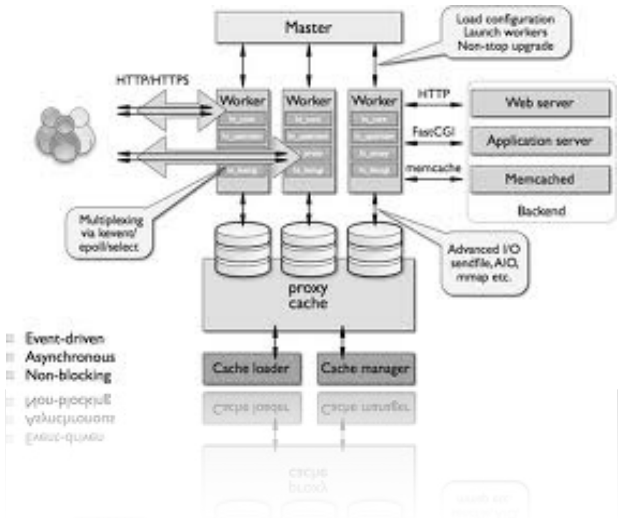


# architecture decisions



# architecture decisions

an architect is responsible for defining the architecture and design principles used to guide technology decisions



# architecture decisions


the decision to use java server faces as your web framework

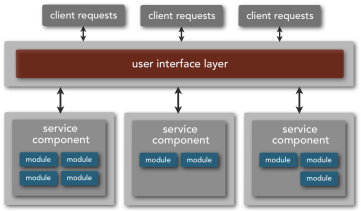
vs.


the decision to use a web-based user interface for your application

# architecture decisions


the decision to use rest to communicate between distributed components

vs.




the decision that components should be distributed remotely for better scalability

# justifying
# architecture decisions

# justifying decisions

## groundhog day anti-pattern

no one understands why a decision was made so it keeps getting discussed over and over and over...

# justifying decisions

## the scenario


internal client

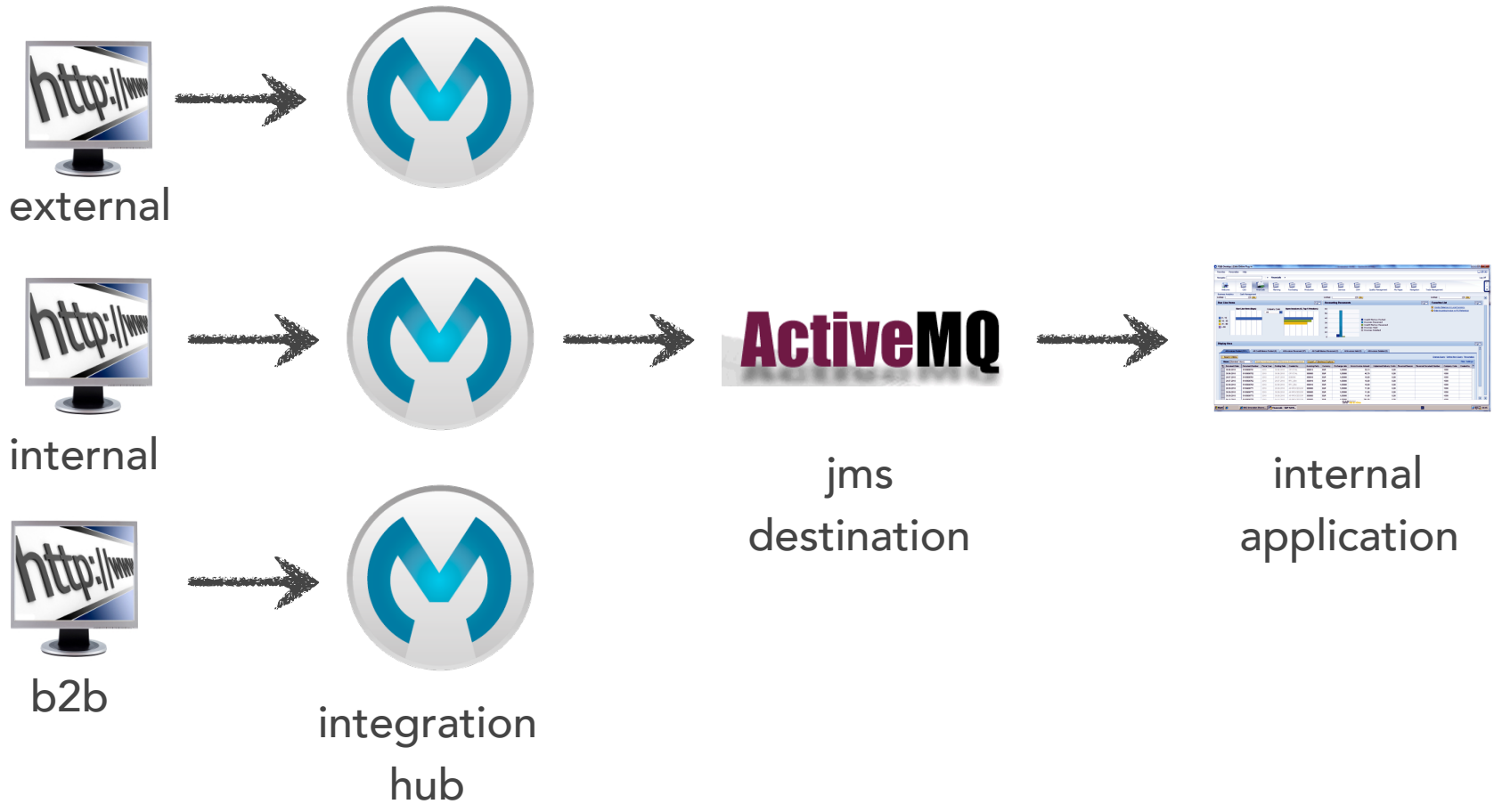
integration hub

jms destination

internal application

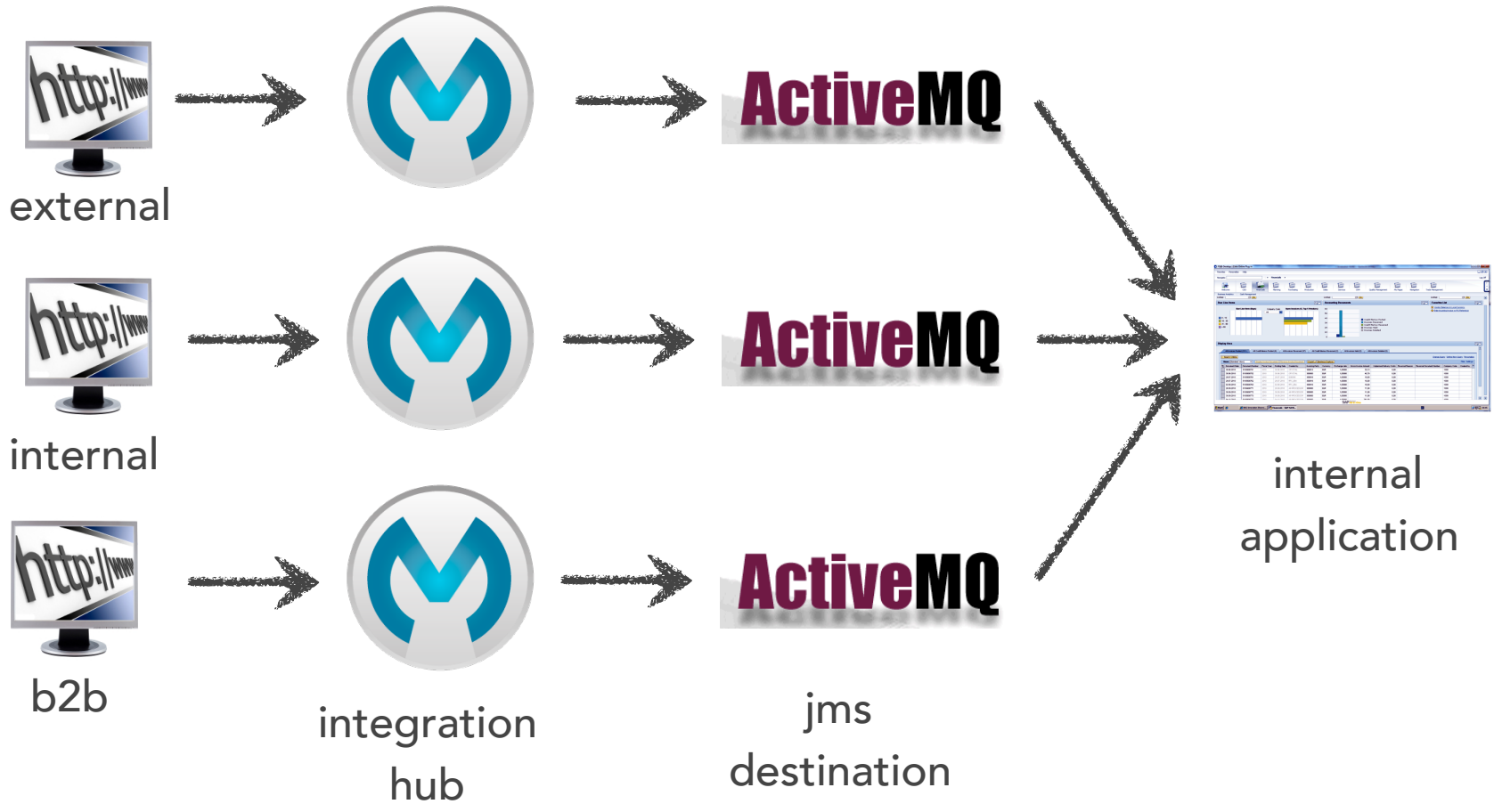
# justifying decisions

## the requirement: you need to federate the hub



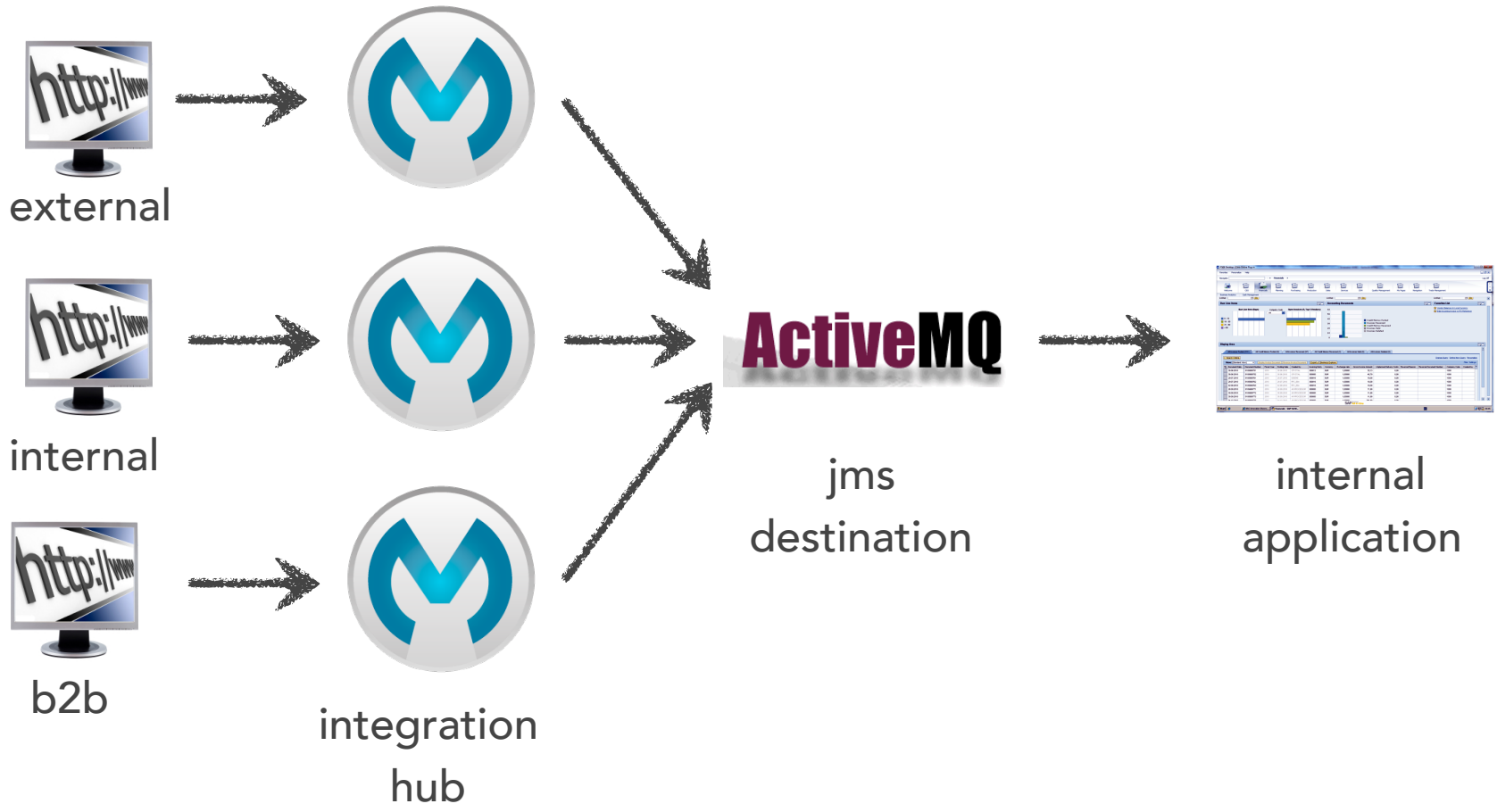
# justifying decisions

## the decision: dedicated broker instances?



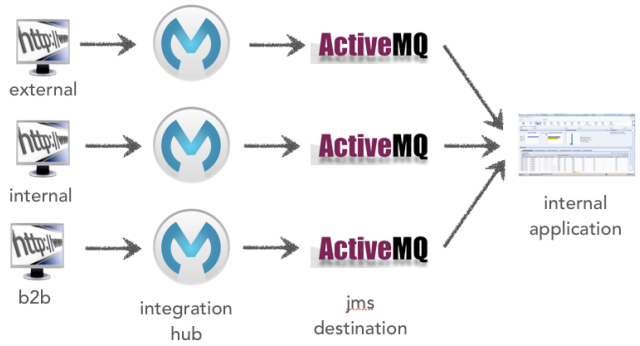
# justifying decisions

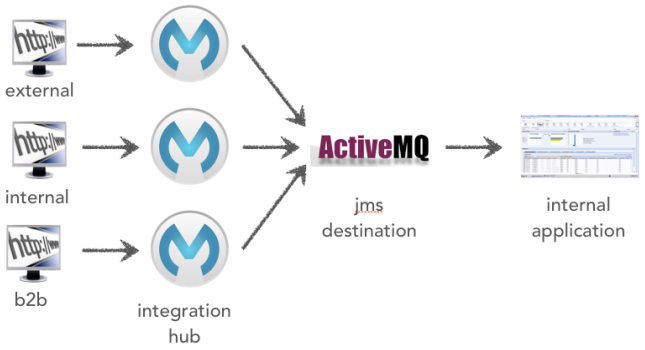
## the decision: centralized broker



# justifying decisions

## identify the conditions and constraints




**conditions and constraints:**

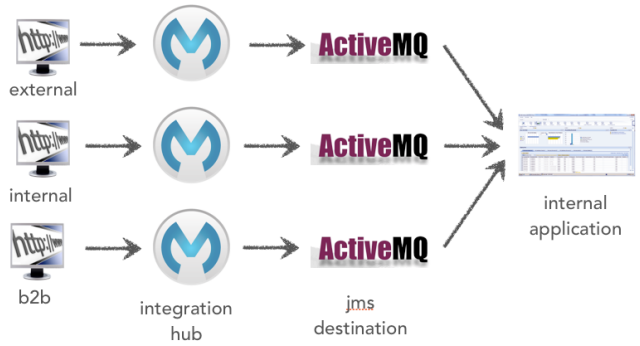
broker only used for hub access

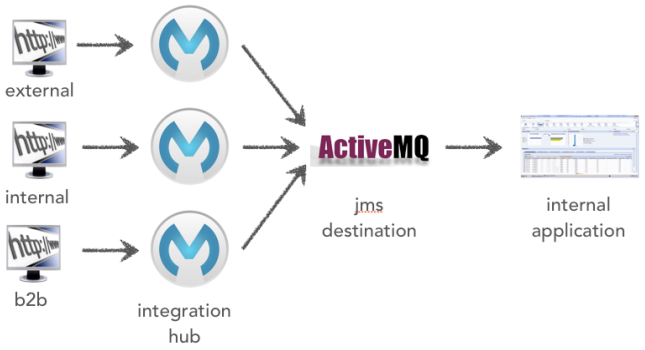
low transaction volumes expected

application logic may be shared between different types of client applications (e.g., internal and external)

# justifying decisions

## analyze each option based on conditions




**considerations:**

broker usage and purpose

overall message throughput

internal application coupling

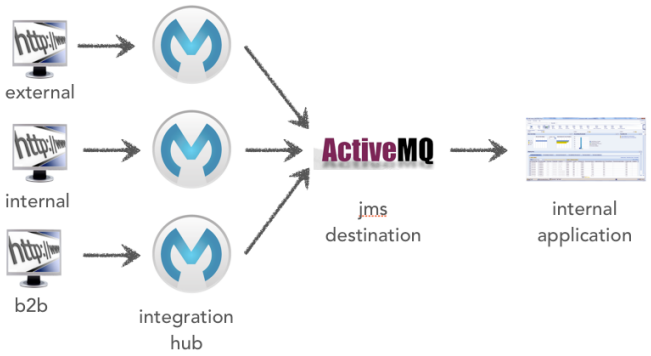
single point of failure

performance bottleneck

# justifying decisions

## architecture decision:

centralized broker



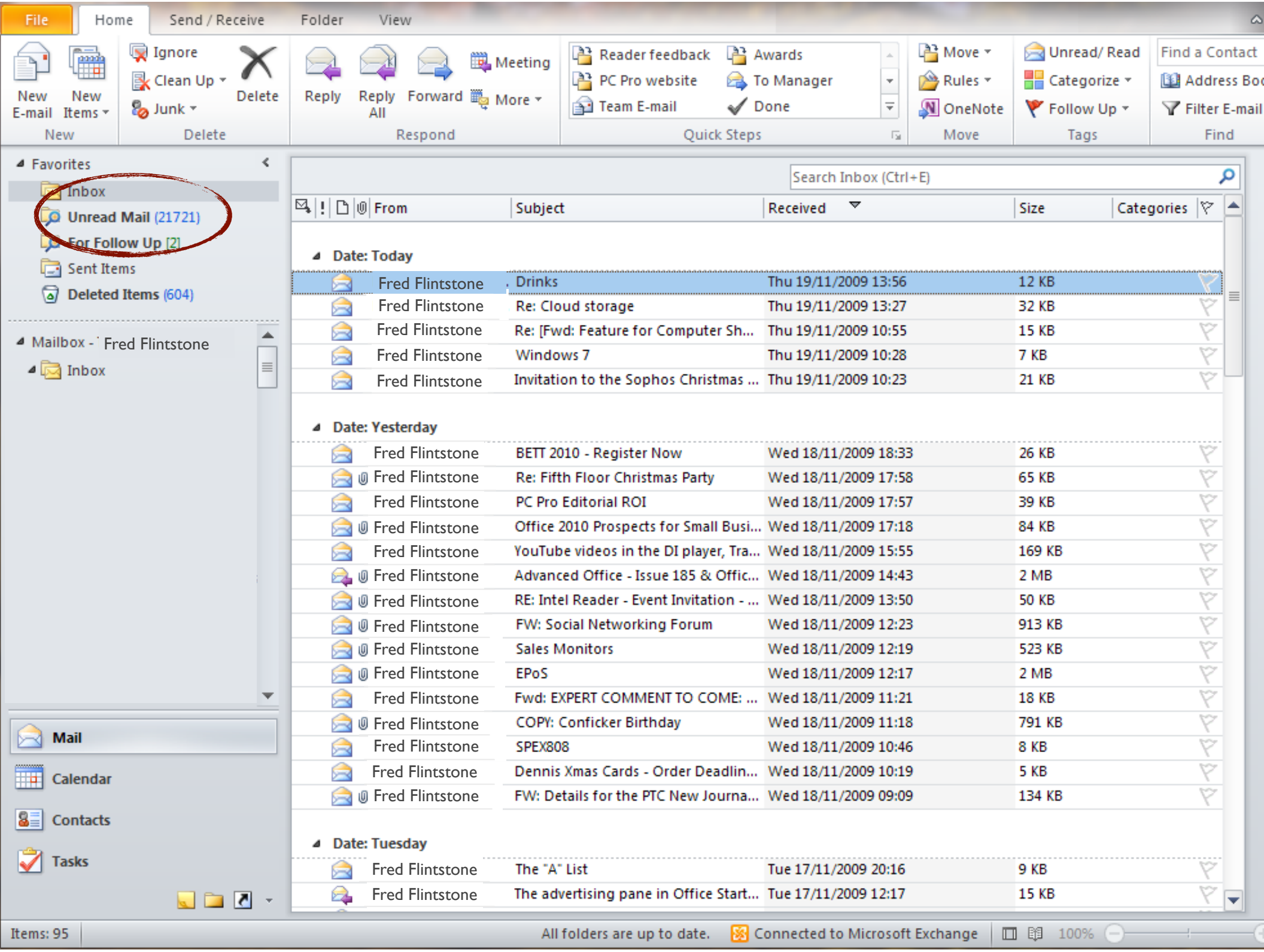

## justification:

the internal applications should not have to know from which broker instance the request came from.

only a single broker connection is needed, allowing for the expansion of additional hub instances with no application changes.

due to low request volumes the performance bottleneck is not an issue; single point of failure can be addressed through failover nodes or clustering.

# documenting and communicating architecture decisions


File
Home
Send / Receive
Folder
View
New E-mail
New Items
Ignore
Clean Up
Junk
Delete
Reply
Reply All
Forward
Meeting
More
Reader feedback
PC Pro website
Team E-mail
Awards
To Manager
Done
Move
Rules
OneNote
Unread/ Read
Categorize
Follow Up
Find a Contact
Address Boo
Filter E-mail
New
Delete
Respond
Quick Steps
Move
Tags
Find
Favorites
Inbox
Unread Mail (21721)
For Follow Up [2]
Sent Items
Deleted Items (604)
Mailbox - Fred Flintstone
Inbox
Search Inbox (Ctrl+E)
From
Subject
Received
Size
Categories
Date: Today
Fred Flintstone Drinks Thu 19/11/2009 13:56 12 KB
Fred Flintstone Re: Cloud storage Thu 19/11/2009 13:27 32 KB
Fred Flintstone Re: [Fwd: Feature for Computer Sh... Thu 19/11/2009 10:55 15 KB
Fred Flintstone Windows 7 Thu 19/11/2009 10:28 7 KB
Fred Flintstone Invitation to the Sophos Christmas ... Thu 19/11/2009 10:23 21 KB
Date: Yesterday
Fred Flintstone BETT 2010 - Register Now Wed 18/11/2009 18:33 26 KB
Fred Flintstone Re: Fifth Floor Christmas Party Wed 18/11/2009 17:58 65 KB
Fred Flintstone PC Pro Editorial ROI Wed 18/11/2009 17:57 39 KB
Fred Flintstone Office 2010 Prospects for Small Busi... Wed 18/11/2009 17:18 84 KB
Fred Flintstone YouTube videos in the DI player, Tra... Wed 18/11/2009 15:55 169 KB
Fred Flintstone Advanced Office - Issue 185 & Offic... Wed 18/11/2009 14:43 2 MB
Fred Flintstone RE: Intel Reader - Event Invitation - ... Wed 18/11/2009 13:50 50 KB
Fred Flintstone FW: Social Networking Forum Wed 18/11/2009 12:23 913 KB
Fred Flintstone Sales Monitors Wed 18/11/2009 12:19 523 KB
Fred Flintstone EPoS Wed 18/11/2009 12:17 2 MB
Fred Flintstone Fwd: EXPERT COMMENT TO COME: ... Wed 18/11/2009 11:21 18 KB
Fred Flintstone COPY: Conficker Birthday Wed 18/11/2009 11:18 791 KB
Fred Flintstone SPEX808 Wed 18/11/2009 10:46 8 KB
Fred Flintstone Dennis Xmas Cards - Order Deadlin... Wed 18/11/2009 10:19 5 KB
Fred Flintstone FW: Details for the PTC New Journa... Wed 18/11/2009 09:09 134 KB
Date: Tuesday
Fred Flintstone The "A" List Tue 17/11/2009 20:16 9 KB
Fred Flintstone The advertising pane in Office Start... Tue 17/11/2009 12:17 15 KB
Mail
Calendar
Contacts
Tasks
Items: 95
All folders are up to date.
Connected to Microsoft Exchange
100%

# communicating decisions

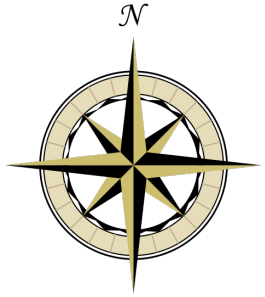
## email-driven architecture

people forget, lose, or don't know an architecture decision was made, and therefore don't implement the architecture correctly

# communicating decisions

## documenting your architecture decisions

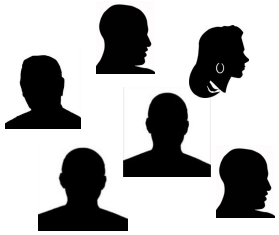
document all of your architecture decisions in a ***central document or wiki*** rather than multiple files spread throughout a crowded shared drive




establish early on ***where*** your decisions will be documented and make sure every team member knows where to go to find them

# communicating decisions

## communicating your architecture decisions

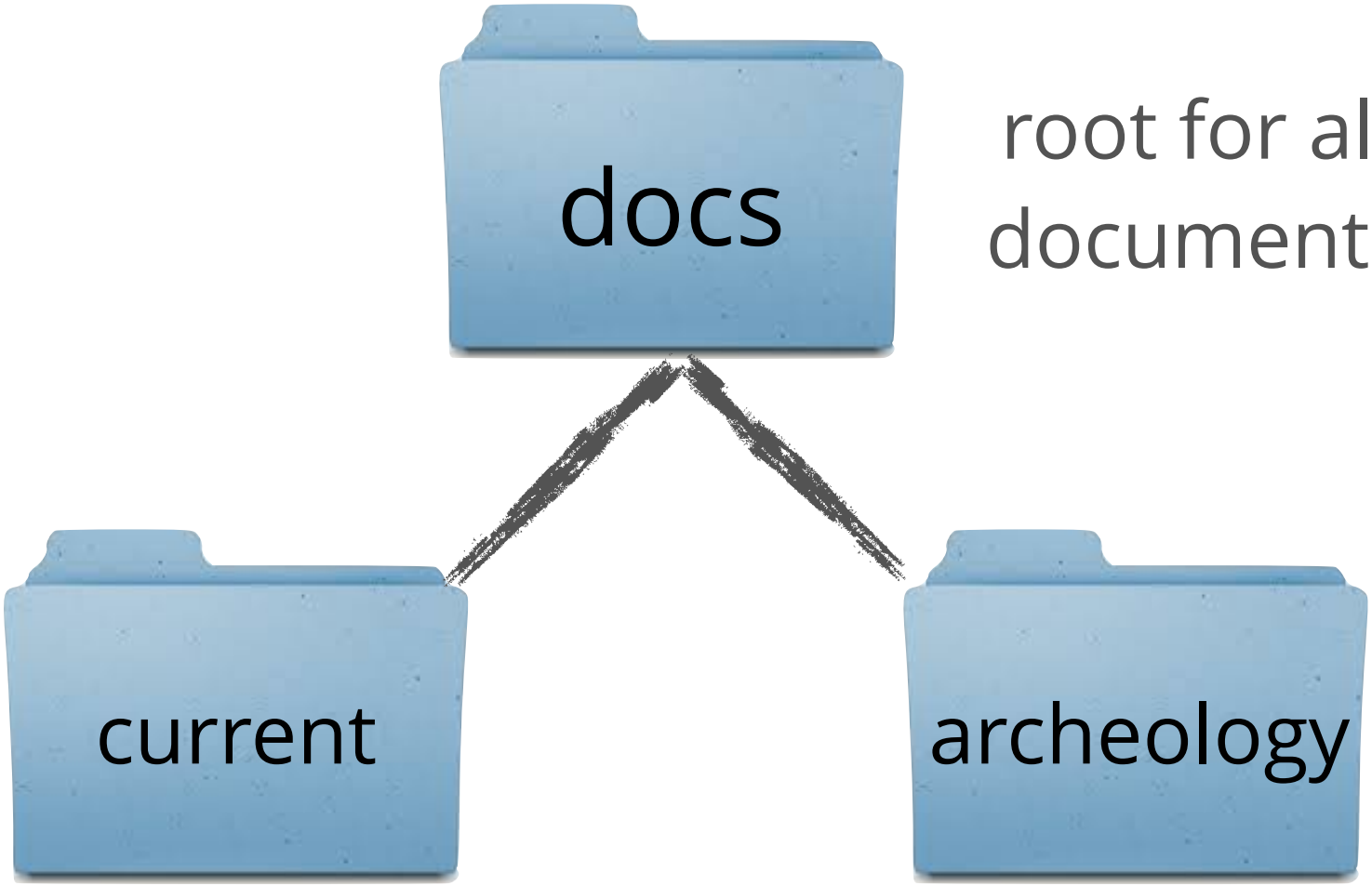


for critical architecture decisions make sure the right stakeholders know about the decision

hold periodic whiteboard sessions with key stakeholders that can then communicate your decisions to others

https://github.com/stuarthalloway/presentations/wiki/Architectural-Briefings

# Archeology



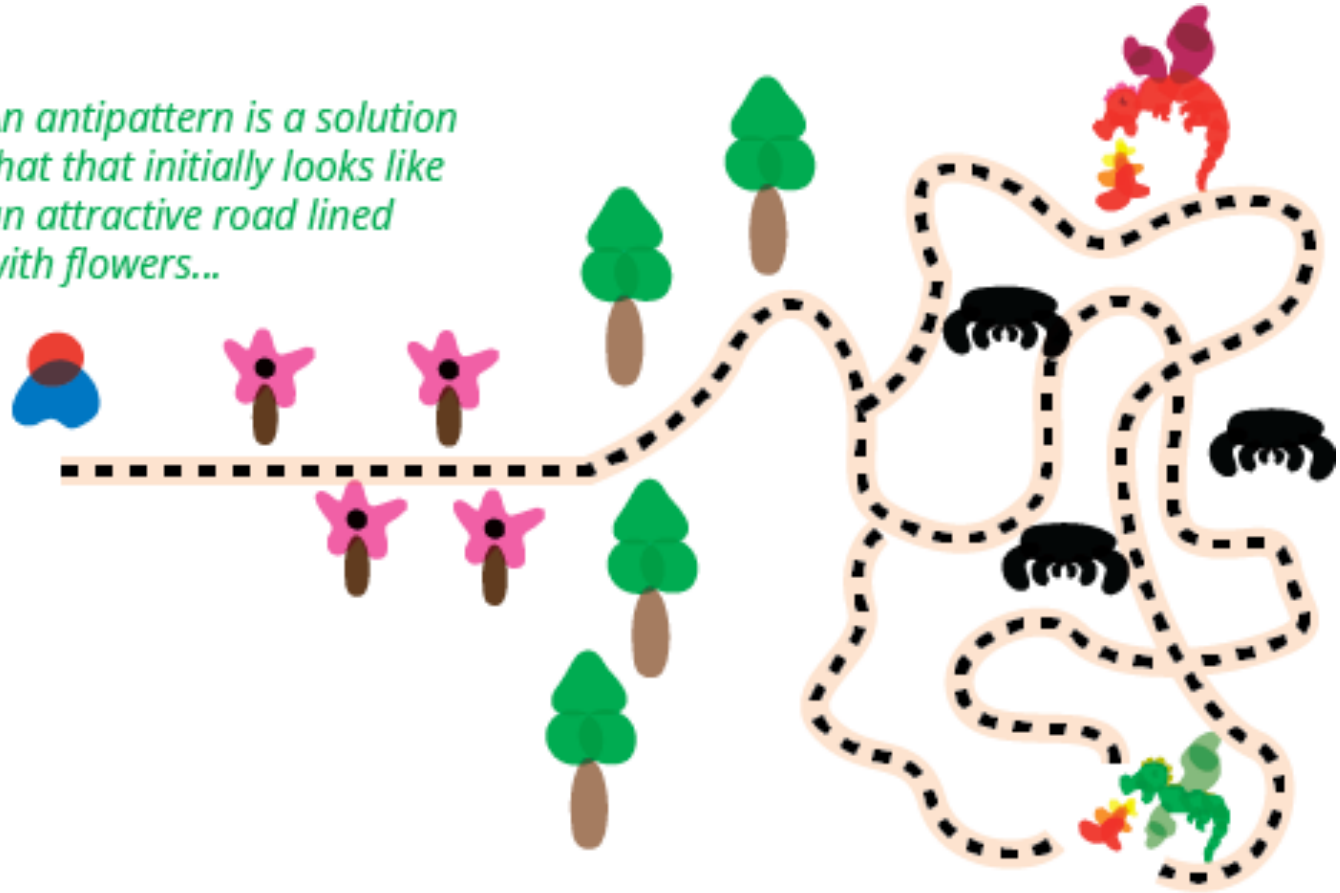

root for all documents

useful enough to update

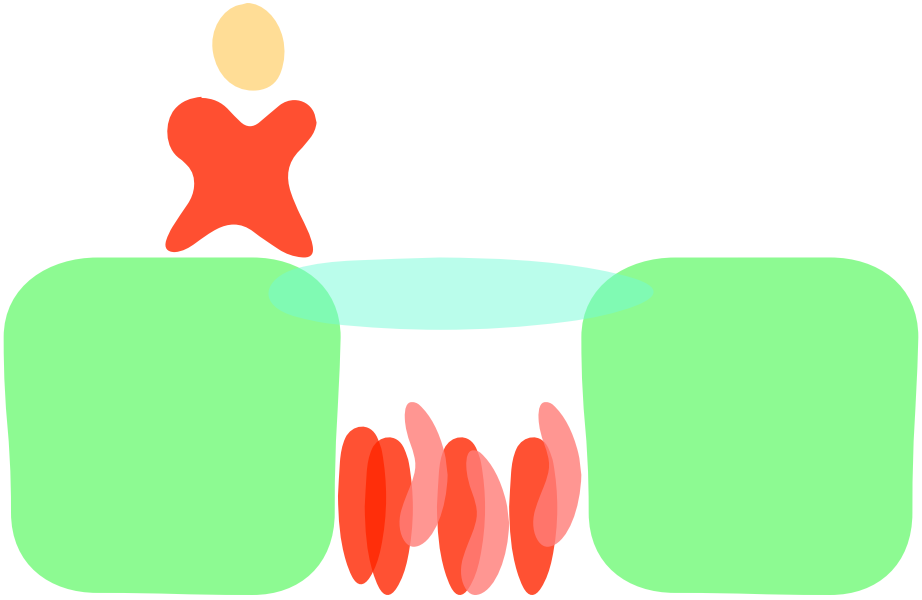
interesting historical artifacts

# anti-pattern

*An antipattern is a solution that that initially looks like an attractive road lined with flowers...*



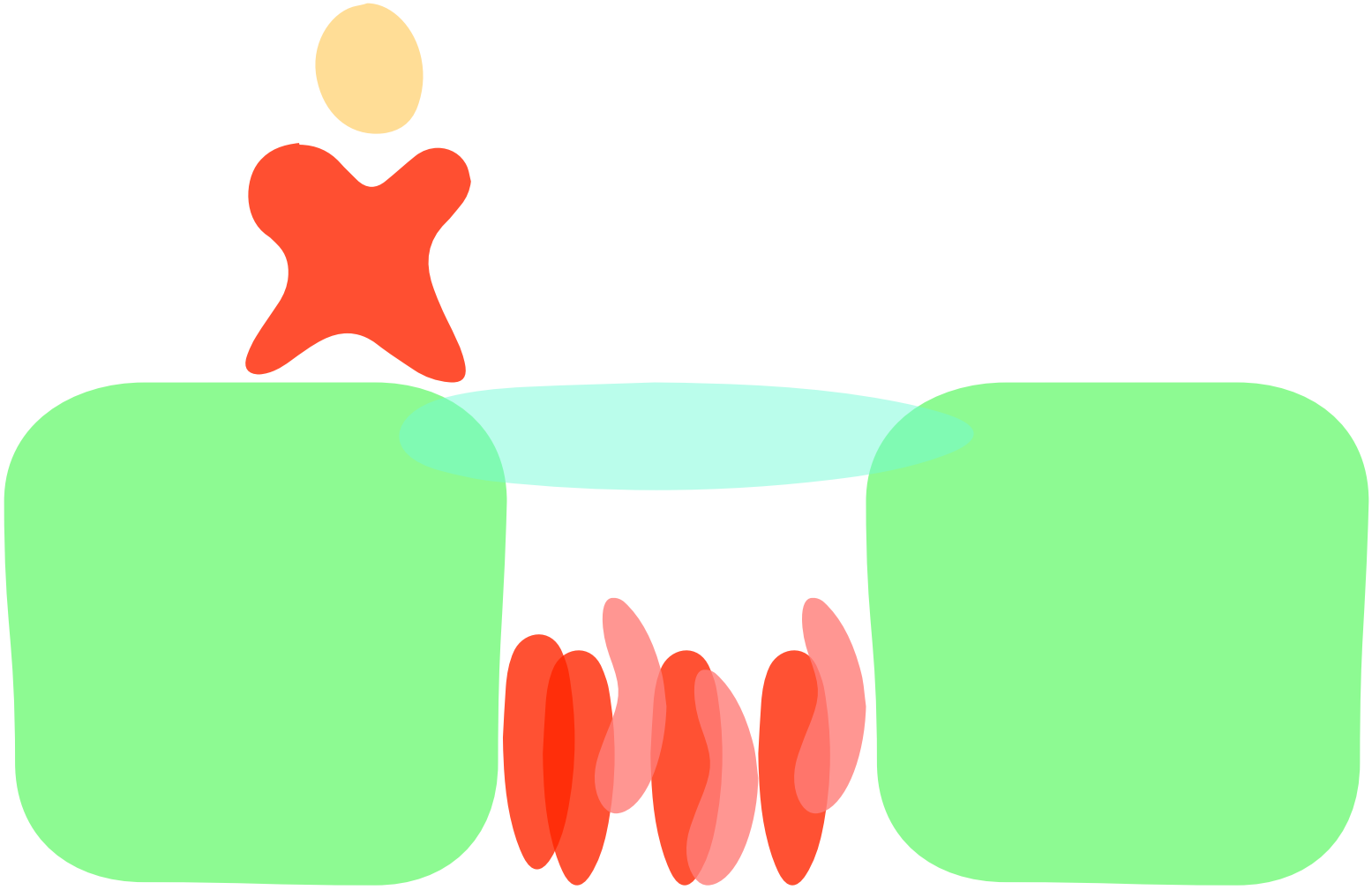
*...but further on leads you into a maze filled with monsters*



## pitfall

A pitfall looks like a safe path but immediately puts you in danger.

# architecture pitfall

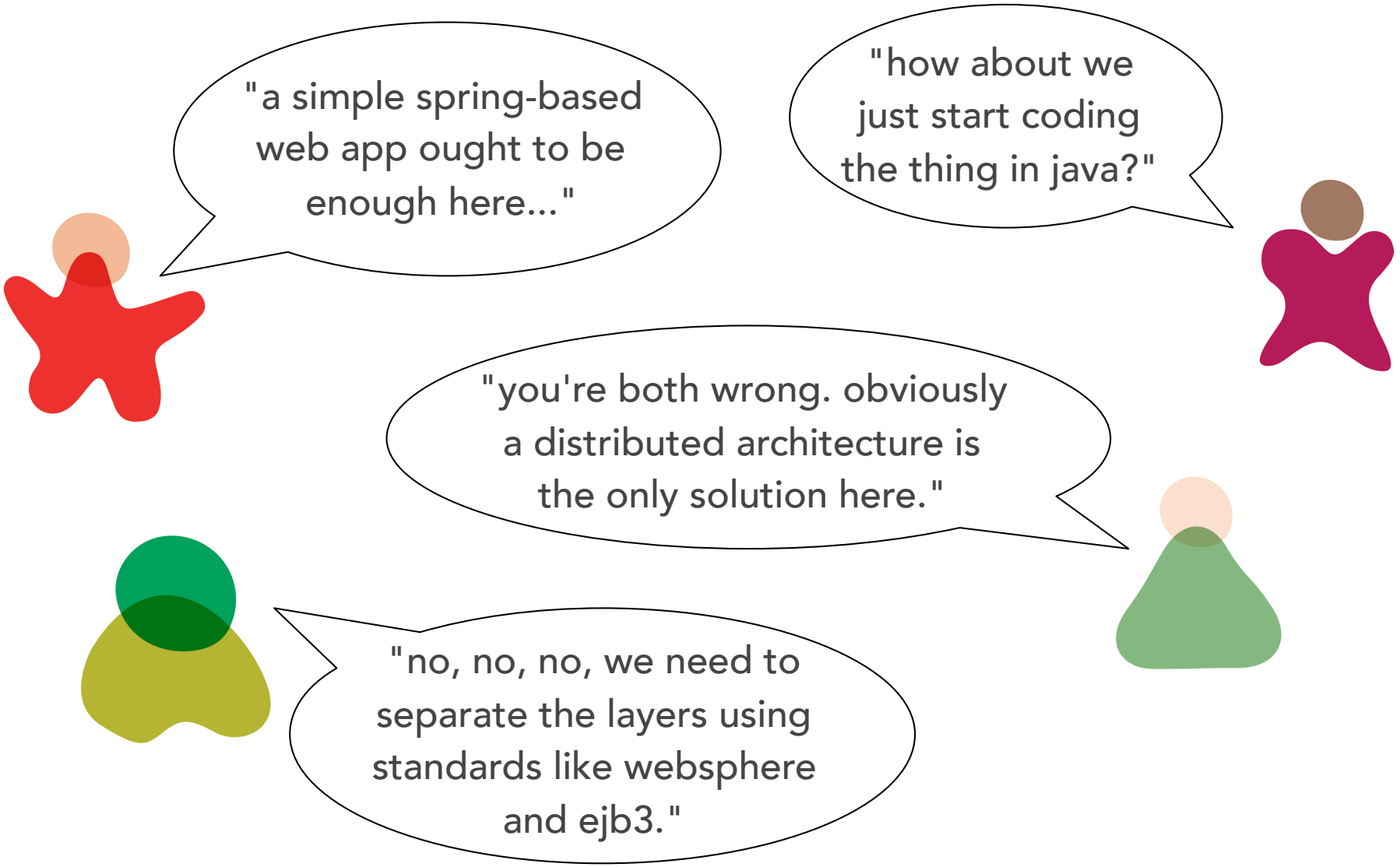


# witches brew architecture

architectures designed by groups resulting in a complex mixture of ideas and no clear vision


Bloor St W
Avenue Rd
VANBOTS

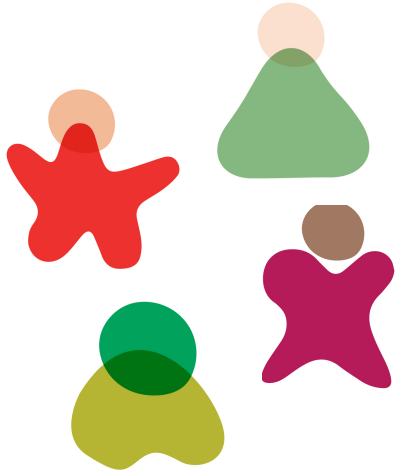
# the problem

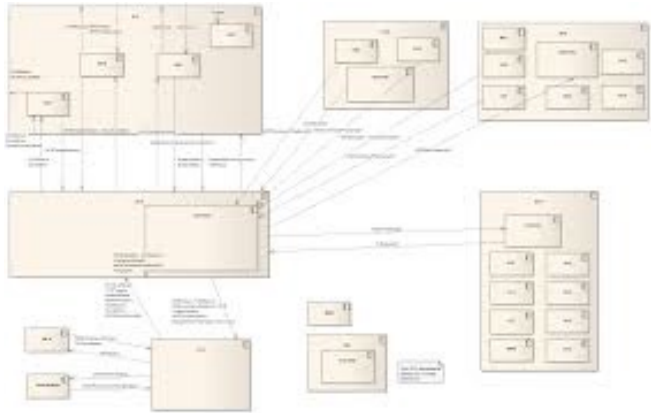

"a simple spring-based web app ought to be enough here..."
"how about we just start coding the thing in java?"
"you're both wrong. obviously a distributed architecture is the only solution here."
"no, no, no, we need to separate the layers using standards like websphere and ejb3."

# the problem


great. i'm working with a bunch of egotistical idiots...
hmmm, where should i go for lunch today?...
these people are about as useful as a back pocket on a shirt...
oh, and i suppose 'spring' solves world hunger as well, huh?

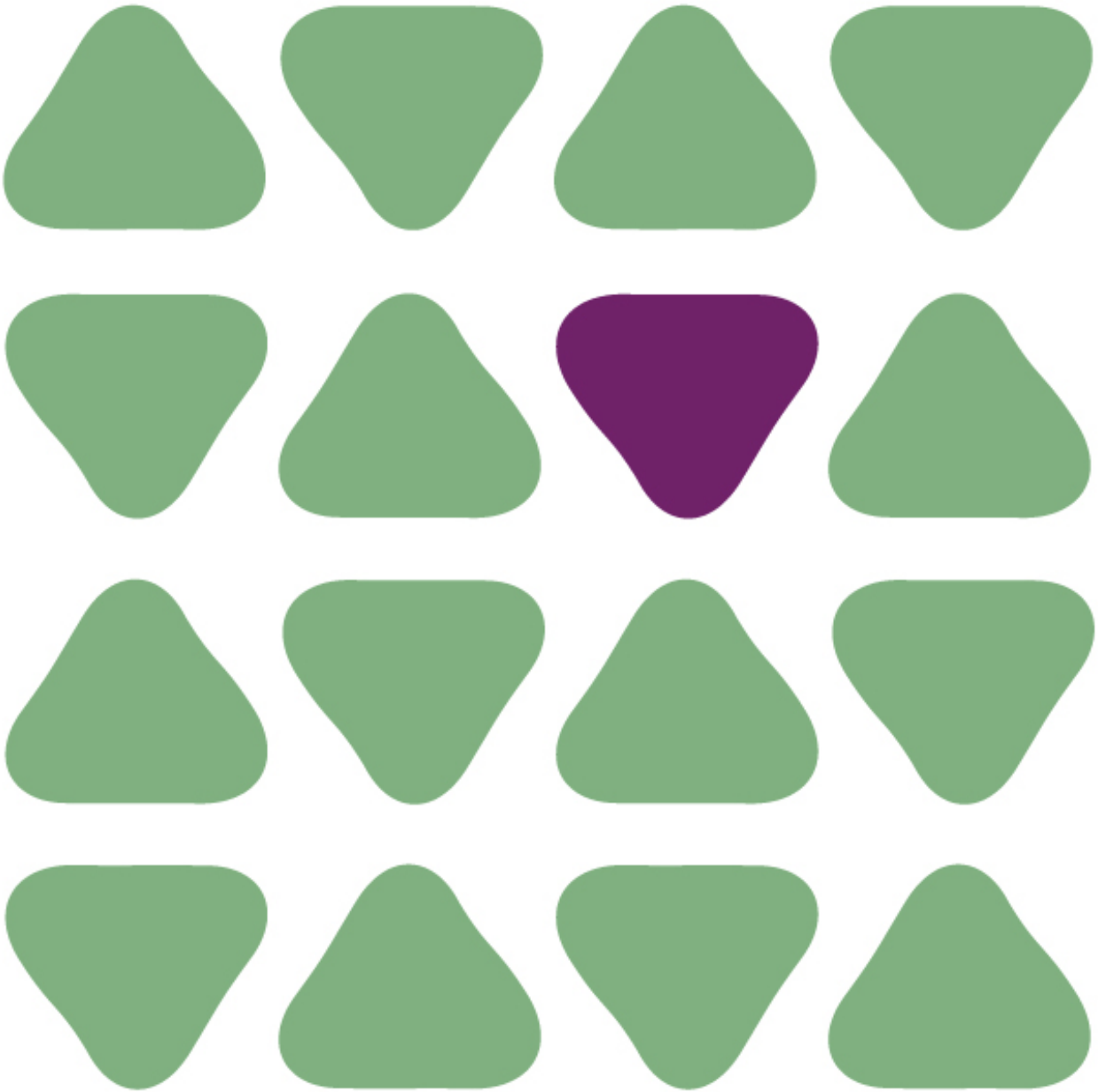
# the goal



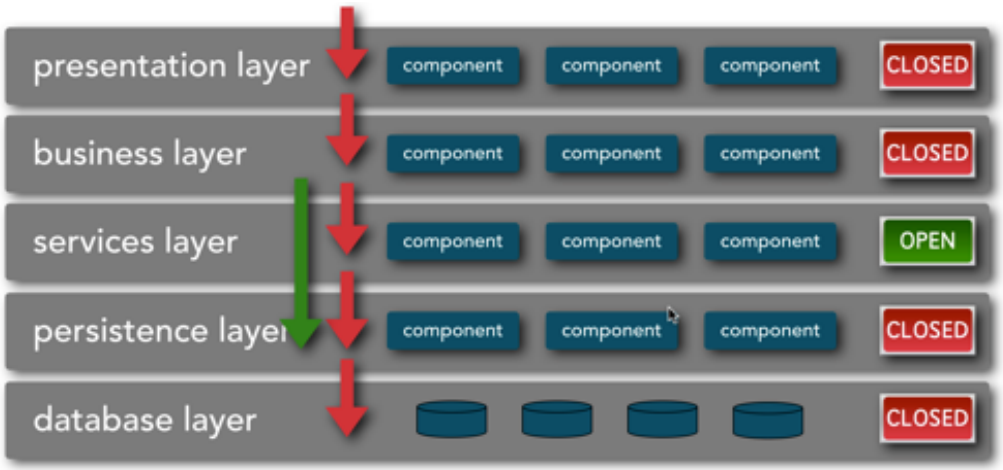


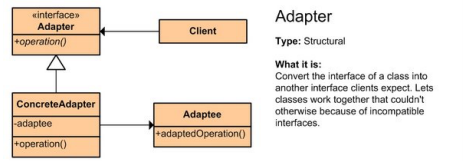
using collective knowledge and experience to arrive at a unified vision for the architecture

# Architecture Patterns



# Components vs Classes

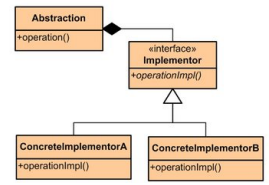



## Adapter

**Type:** Structural

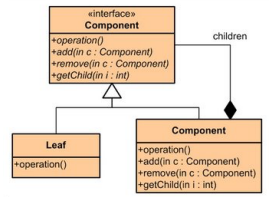
**What it is:**
Convert the interface of a class into another interface clients expect. Lets classes work together that couldn't otherwise because of incompatible interfaces.




## Bridge

**Type:** Structural

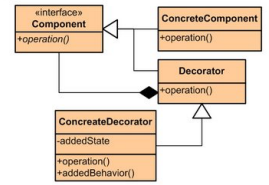
**What it is:**
Decouple an abstraction from its implementation so that the two can vary independently.




## Composite

**Type:** Structural

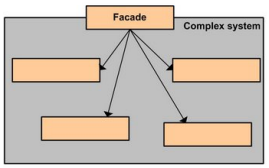
**What it is:**
Compose objects into tree structures to represent part-whole hierarchies. Lets clients treat individual objects and compositions of objects uniformly.




## Decorator

**Type:** Structural

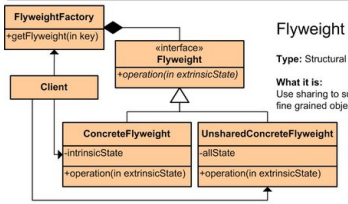
**What it is:**
Attach additional responsibilities to an object dynamically. Provide a flexible alternative to sub-classing for extending functionality.




## Facade

**Type:** Structural

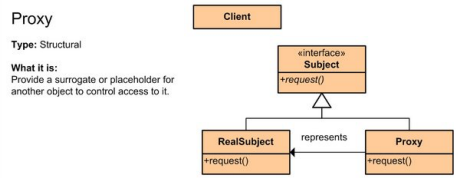
**What it is:**
Provide a unified interface to a set of interfaces in a subsystem. Defines a high-level interface that makes the subsystem easier to use.




## Flyweight

**Type:** Structural

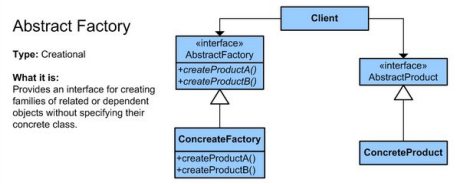
**What it is:**
Use sharing to support large numbers of fine grained objects efficiently.




## Proxy

**Type:** Structural

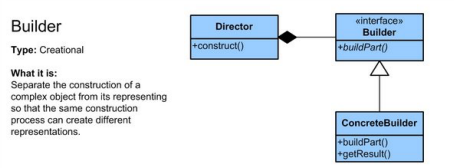
**What it is:**
Provide a surrogate or placeholder for another object to control access to it.




## Abstract Factory

**Type:** Creational

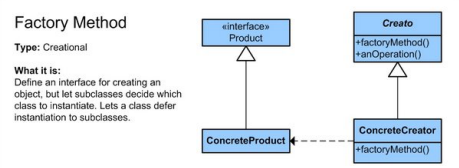
**What it is:**
Provides an interface for creating families of related or dependent objects without specifying their concrete class.




## Builder

**Type:** Creational

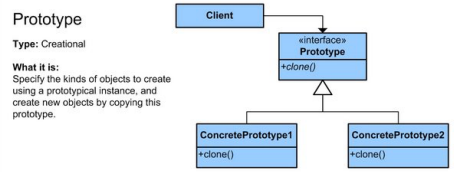
**What it is:**
Separate the construction of a complex object from its representing so that the same construction process can create different representations.




## Factory Method

**Type:** Creational

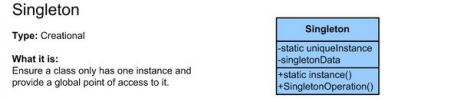
**What it is:**
Define an interface for creating an object, but let subclasses decide which class to instantiate. Lets a class defer instantiation to subclasses.




## Prototype

**Type:** Creational

**What it is:**
Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype.



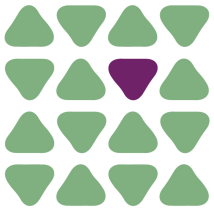

## Singleton

**Type:** Creational

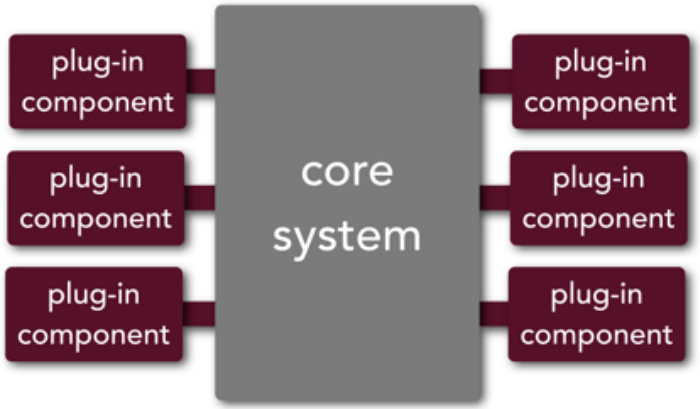
**What it is:**
Ensure a class only has one instance and provide a global point of access to it.

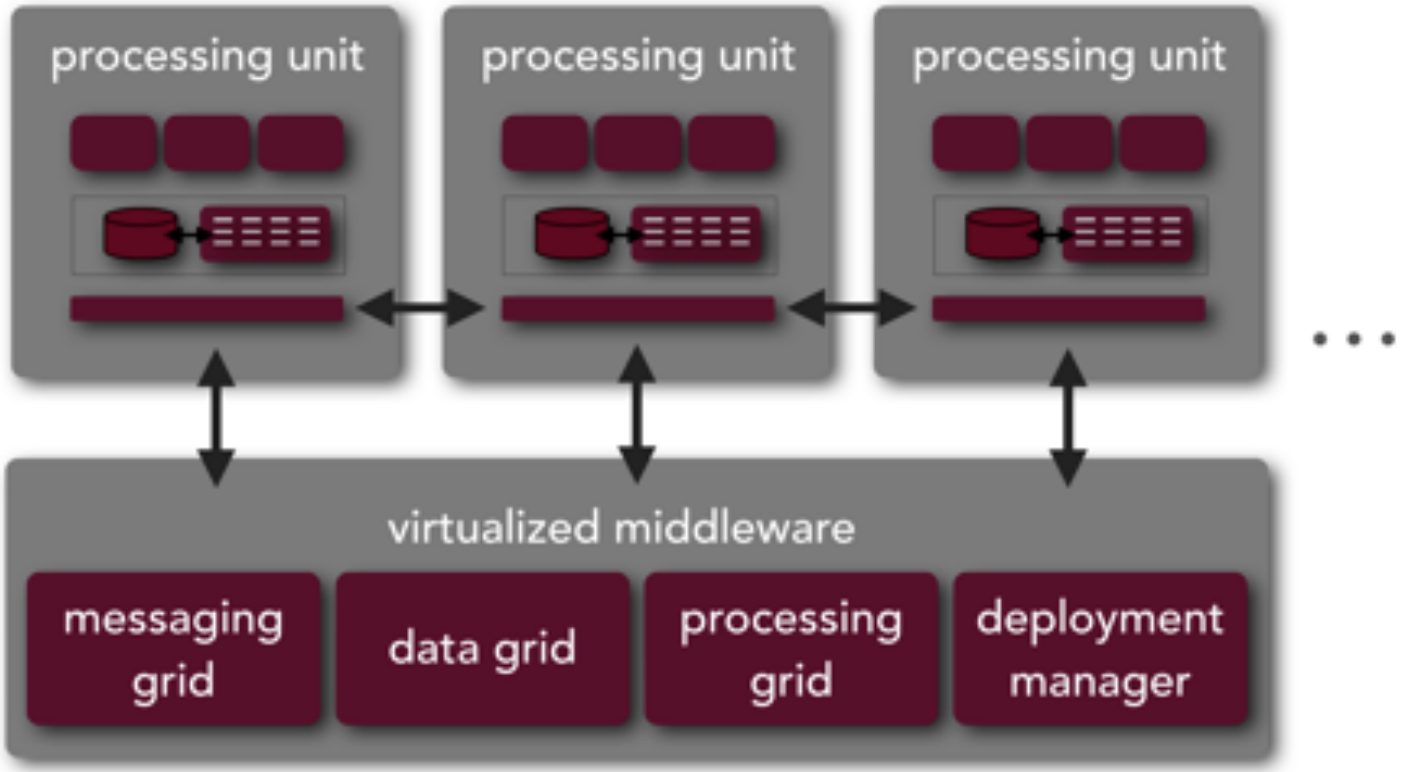


Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, Massachusetts: Addison Wesley Longman, Inc.

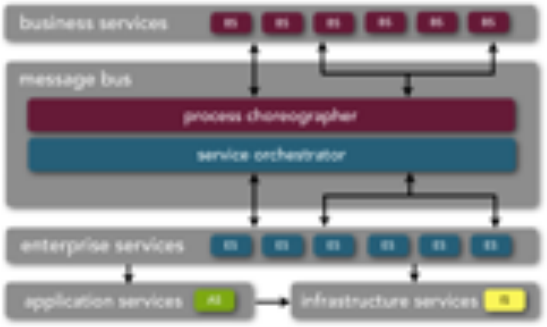


# Component Types



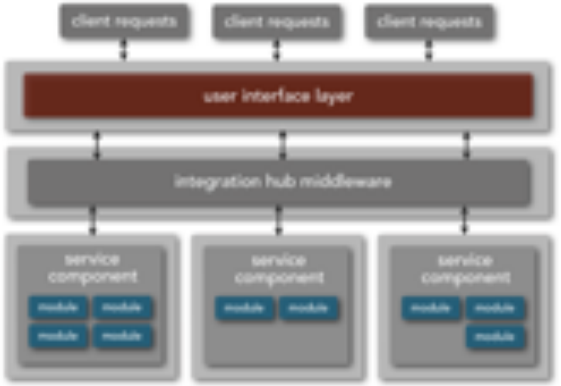



# Hybrids & Variants



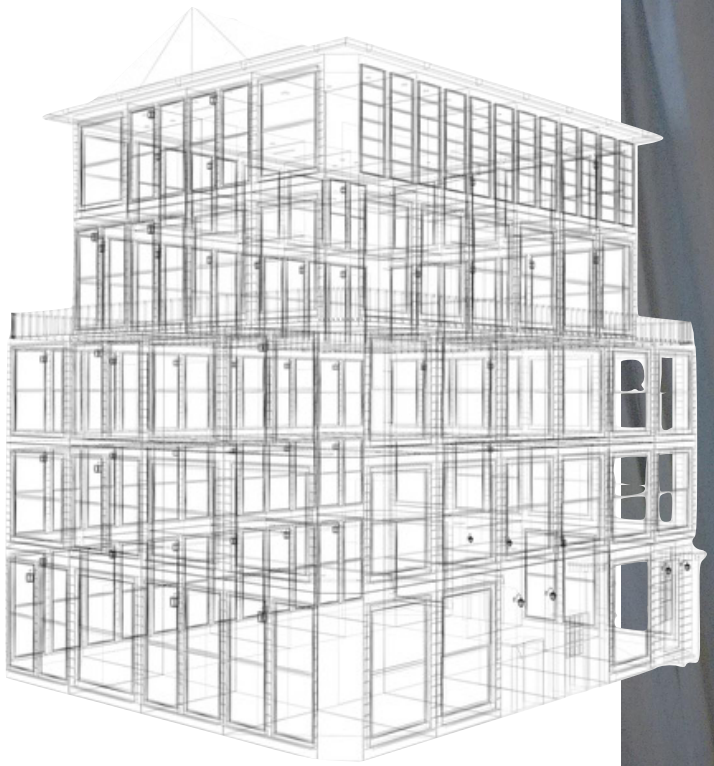

service-oriented architecture


microservices architecture

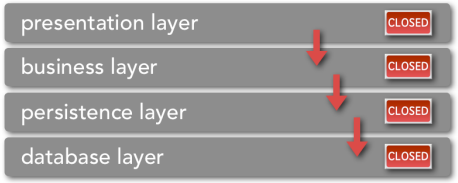



service-based architecture

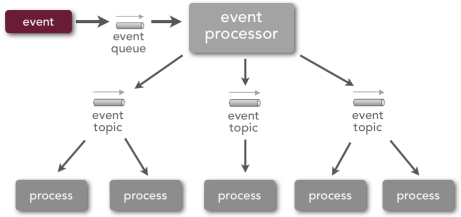
# Scaffolding vs Design



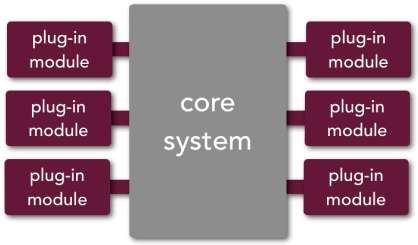
# Architecture Patterns



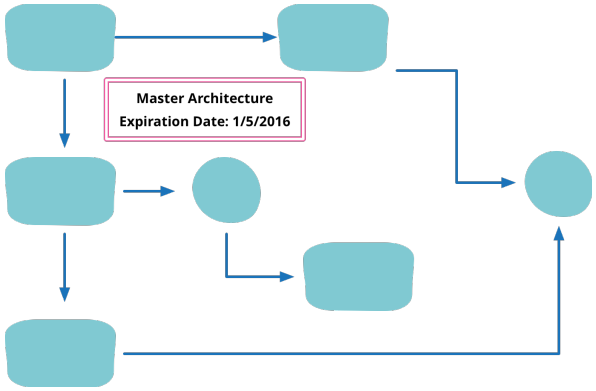

traditional layered architecture




event-driven architecture

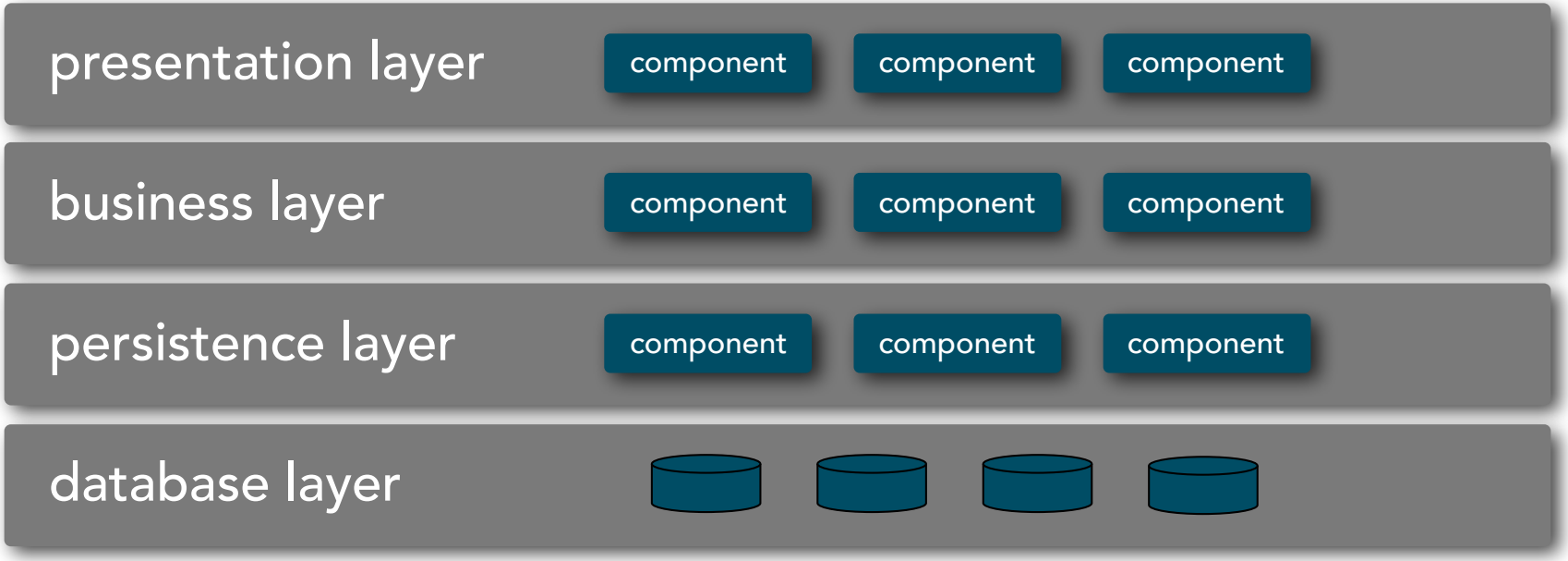



microkernel architecture

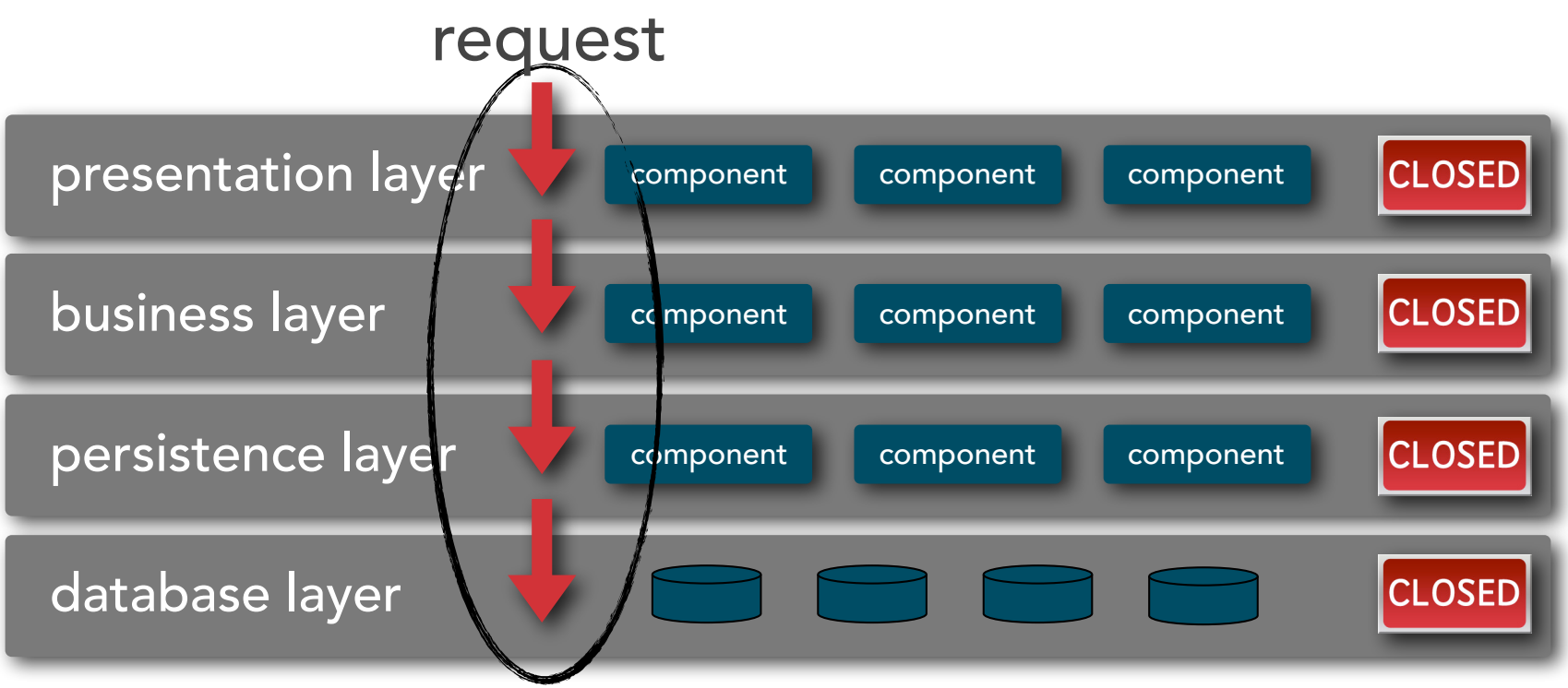



sacrificial architecture

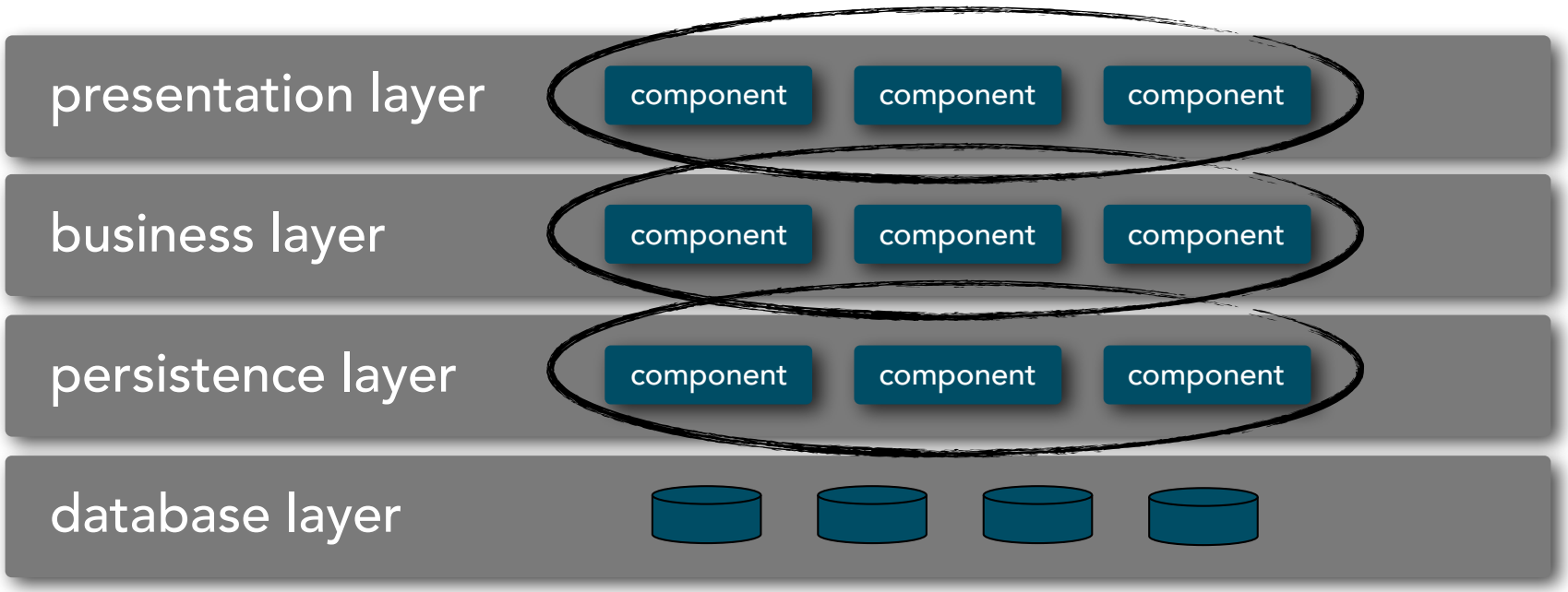
# layered architecture



# layered architecture

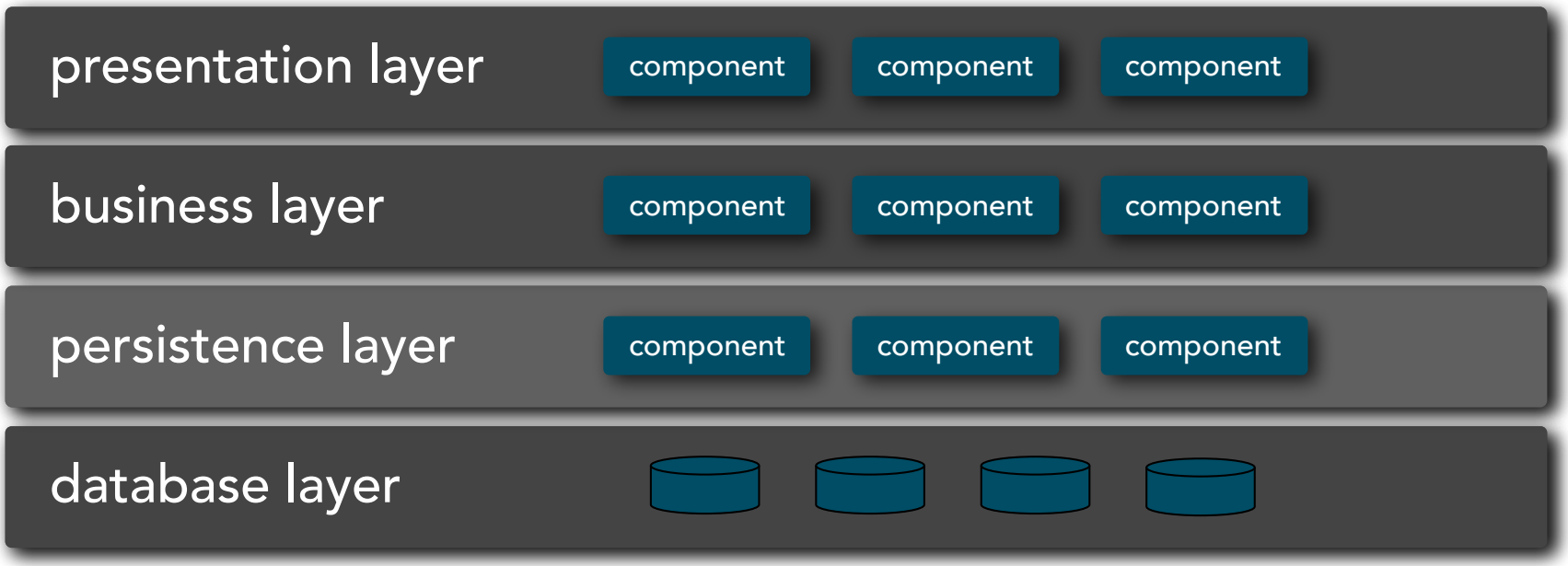


# layered architecture




separation of concerns

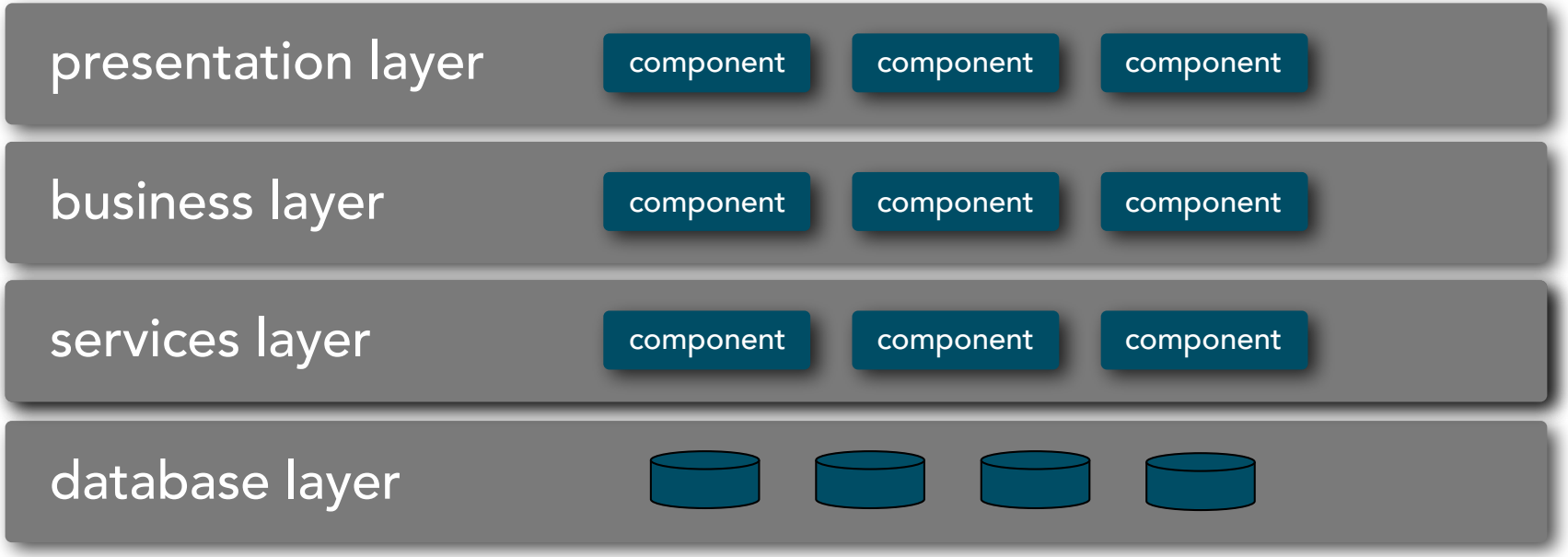
# layered architecture




layers of isolation

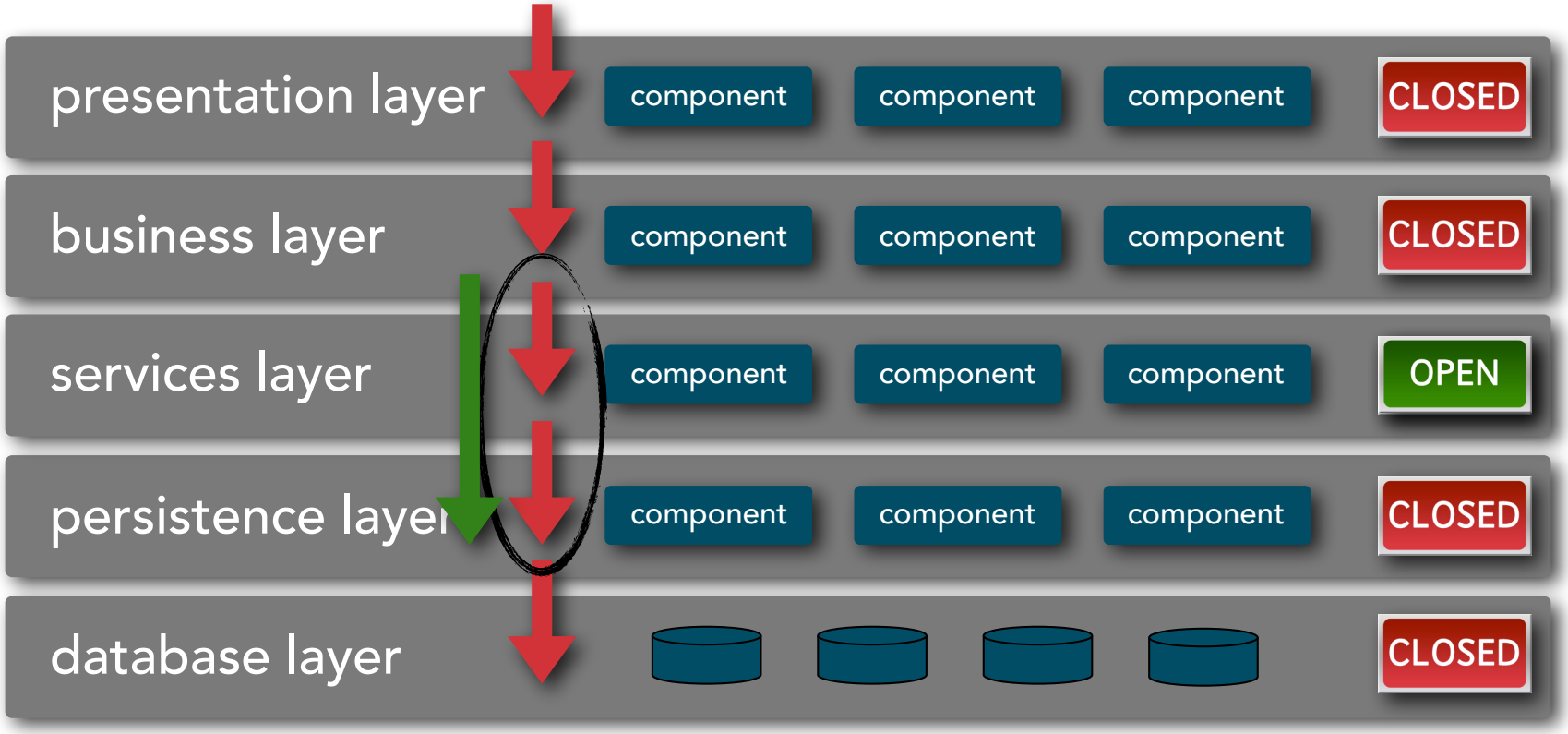
# layered architecture

## hybrids and variants



# layered architecture

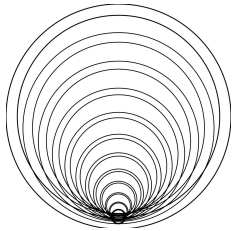
## hybrids and variants



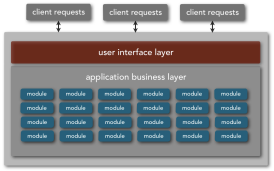
# layered architecture

## considerations


good general purpose architecture and a good starting point for most systems

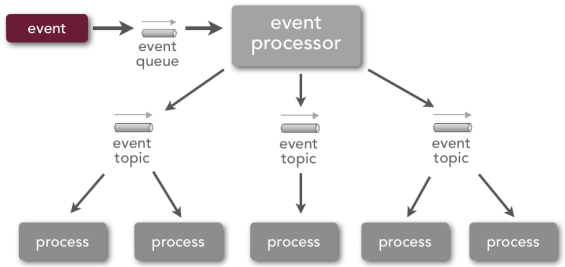


watch out for the architecture sinkhole anti-pattern

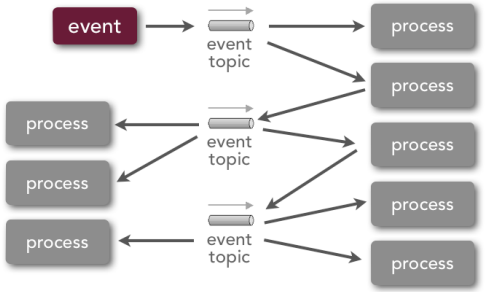



tends to lend itself towards monolithic applications

# event-driven architecture



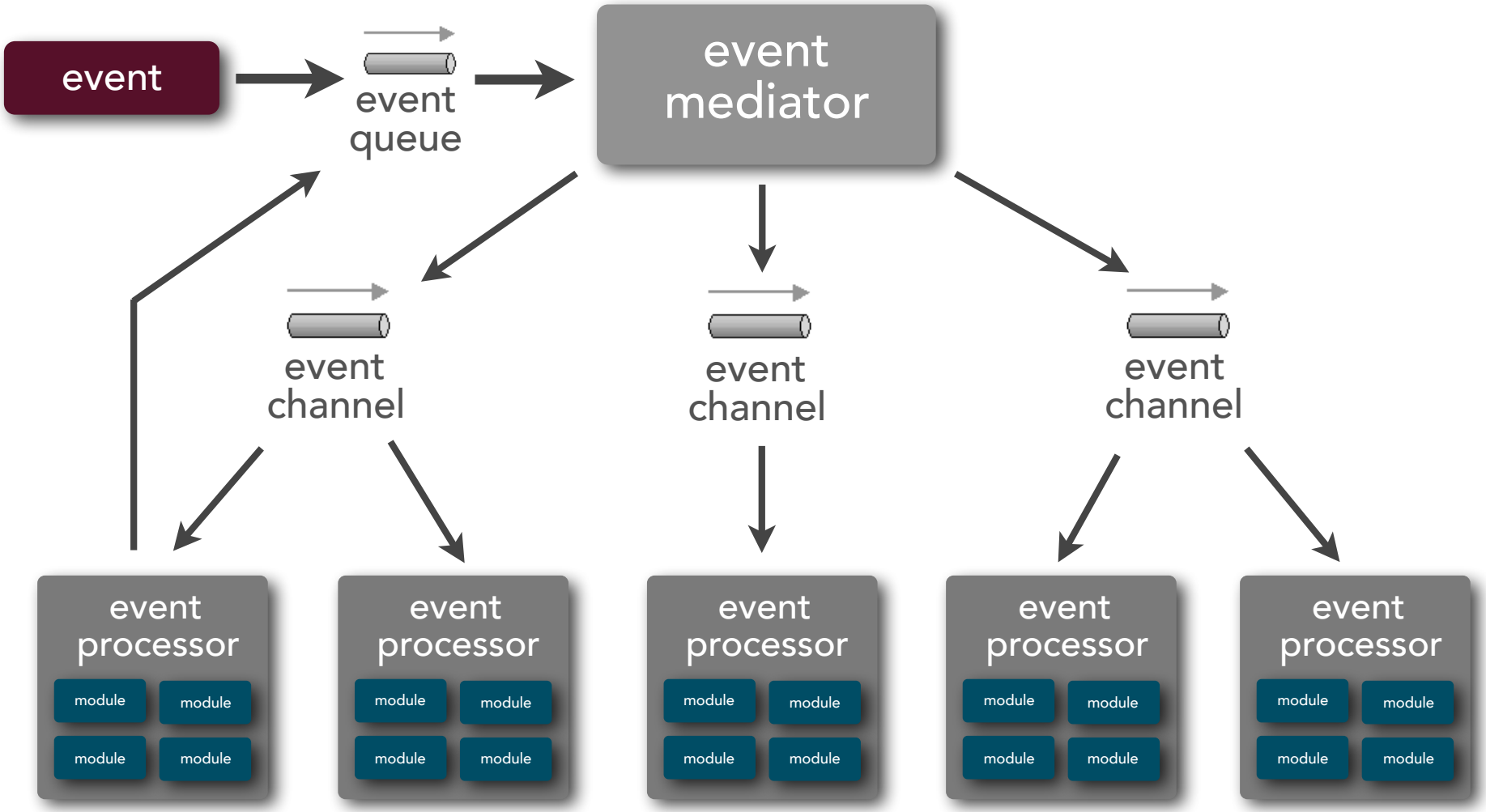

mediator topology




broker topology

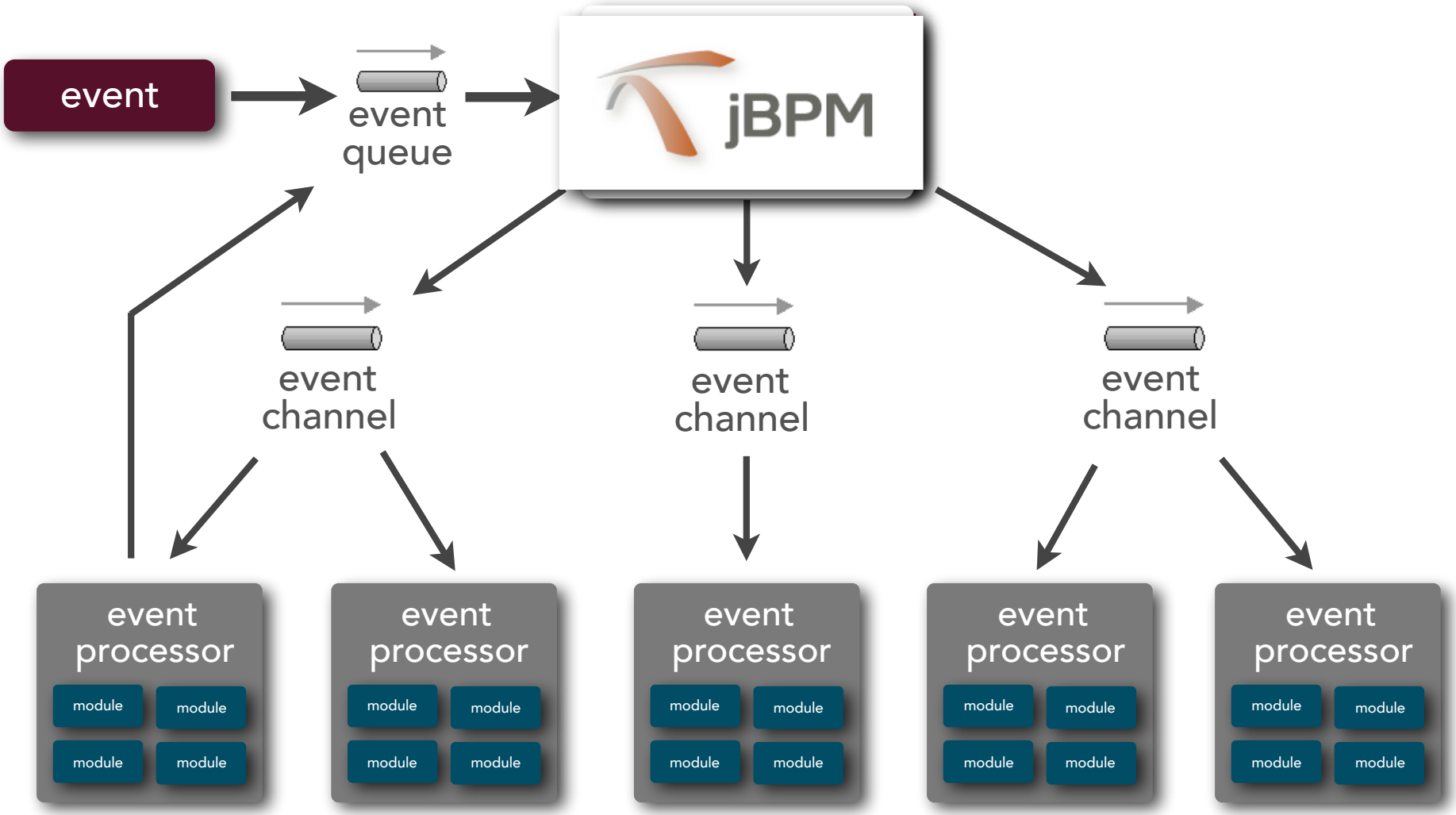
# event-driven architecture

## mediator topology



# event-driven architecture

## mediator topology



# event-driven architecture


you move...

you moved!

process engine

change address

recalc quote

update claims

adjust claims

notify insured

customer process

quote process

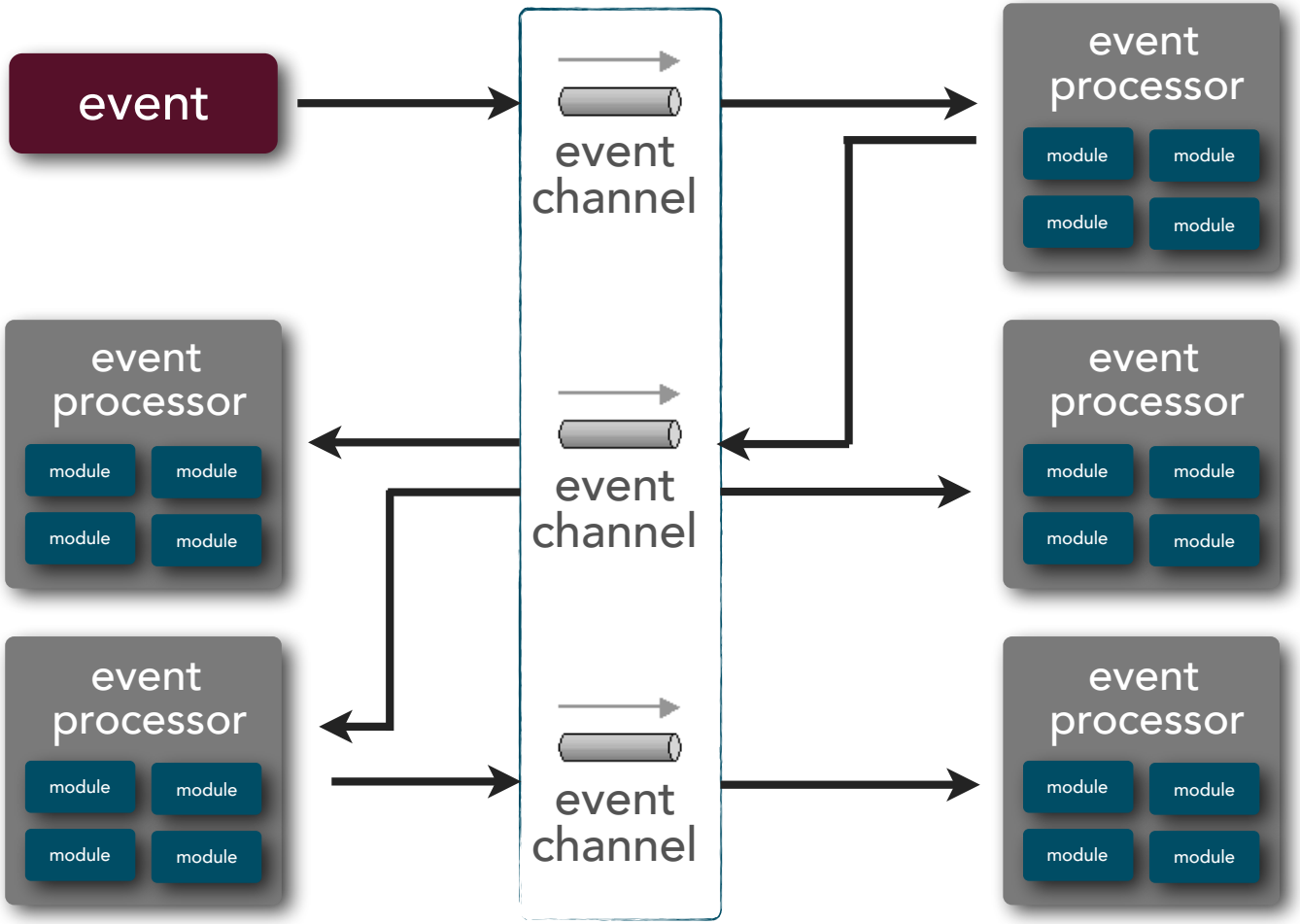
claims process

adjustment process

notification process

# event-driven architecture

## broker topology



# event-driven architecture


you move...

you moved!

customer process

change address

change address

quote process

recalc quote

claims process

update claims

update claims

notification process

adjustment process

# event-driven architecture

## considerations

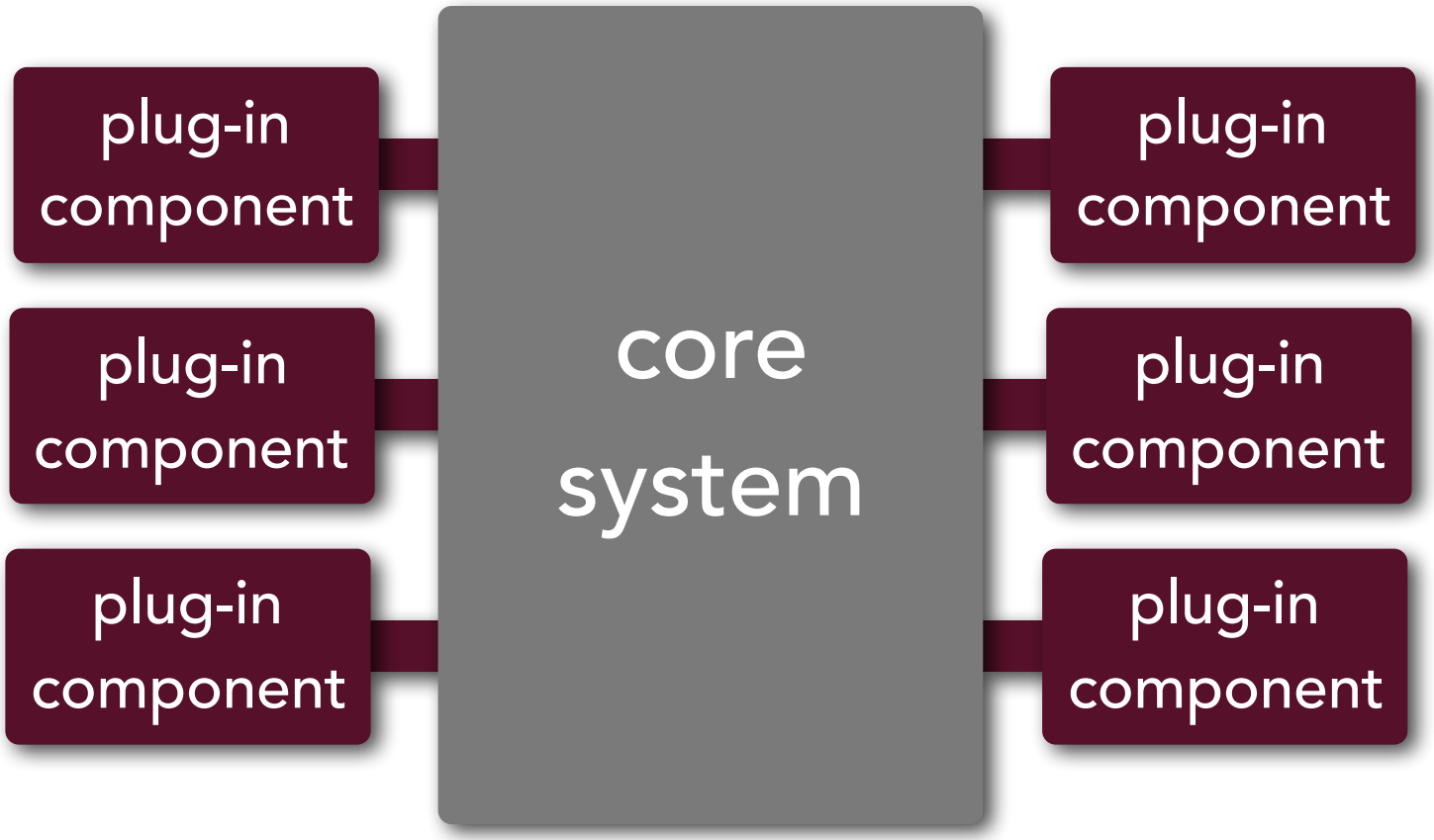

contract creation, maintenance, and versioning can be difficult


must address remote process availability or unresponsiveness


reconnection logic on server restart or failure must be addressed

# microkernel architecture

## (a.k.a. plug-in architecture pattern)



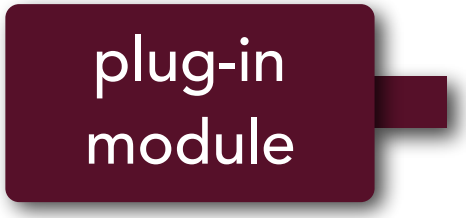
# microkernel architecture

## architectural components


minimal functionality to run system

general business rules and logic

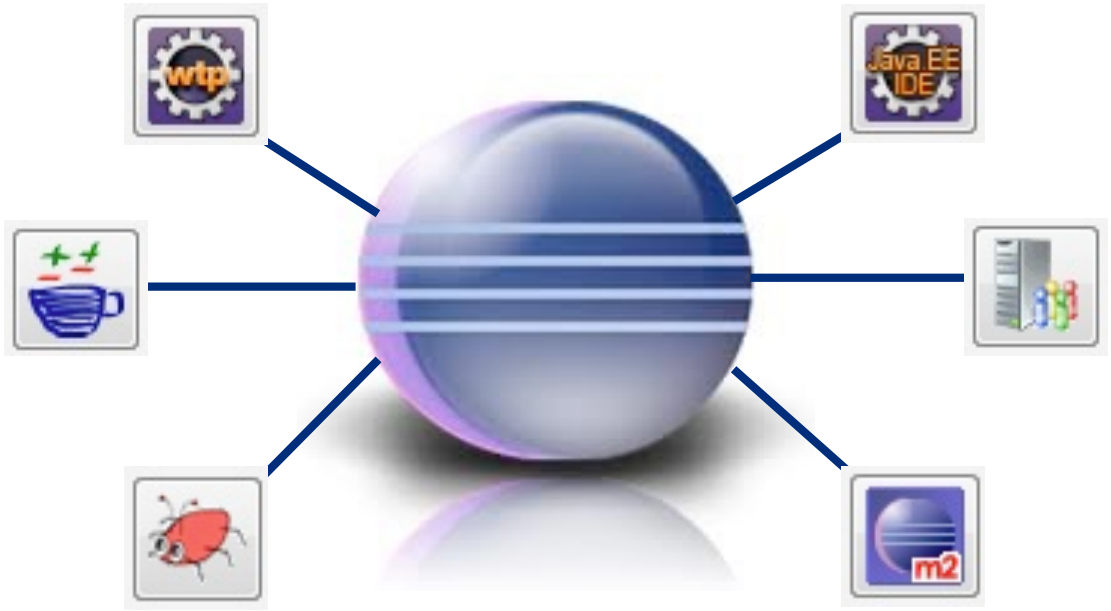
no custom processing




standalone independent module

specific additional rules or logic

# microkernel architecture



# microkernel architecture

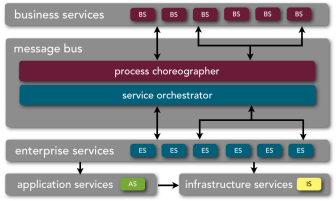
## claims processing

# microkernel architecture

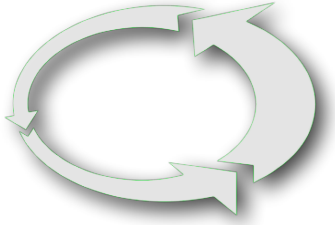
## registry

# microkernel architecture

## considerations



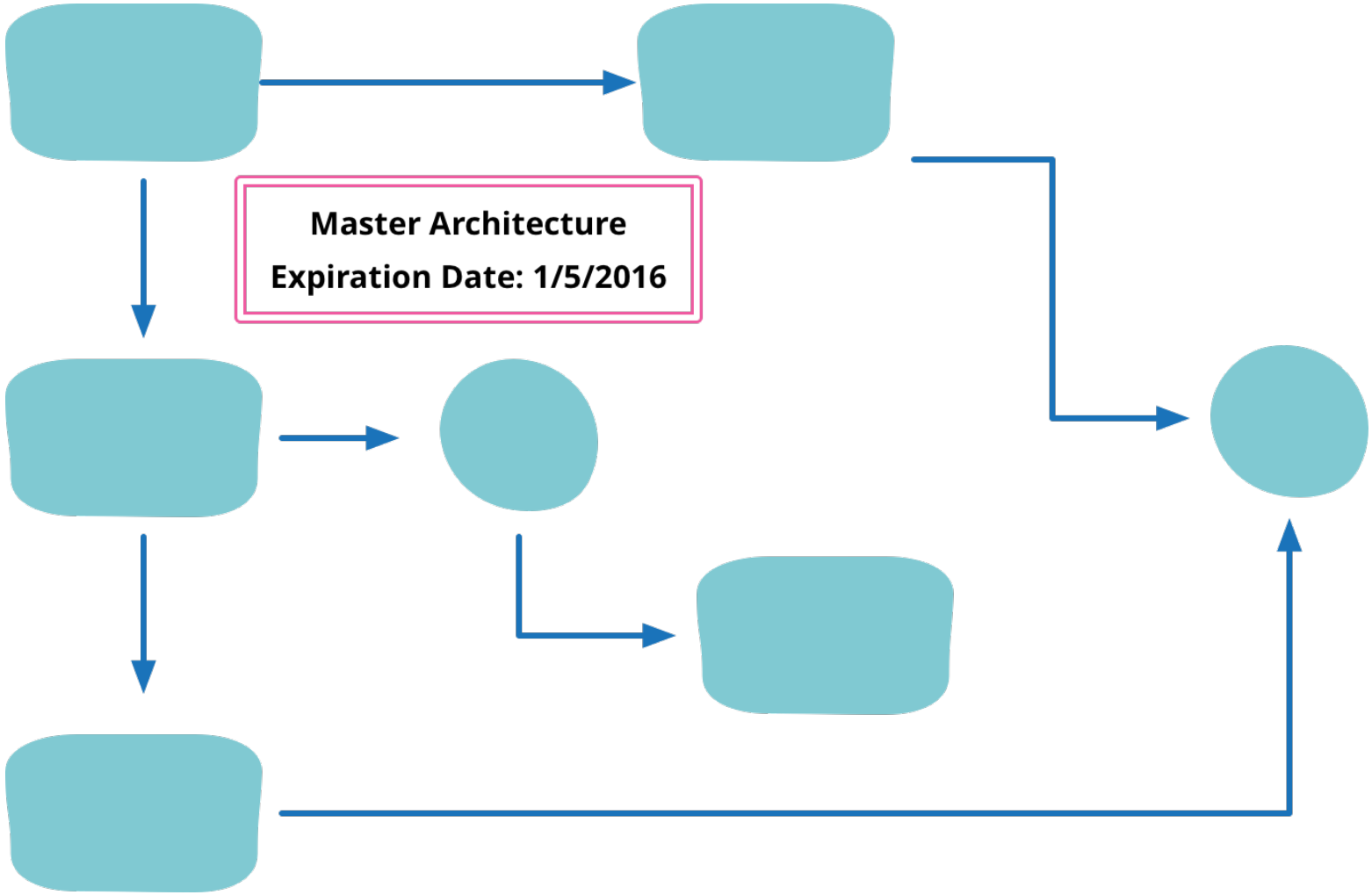

can be embedded or used as part of another pattern



great support for evolutionary design and incremental development

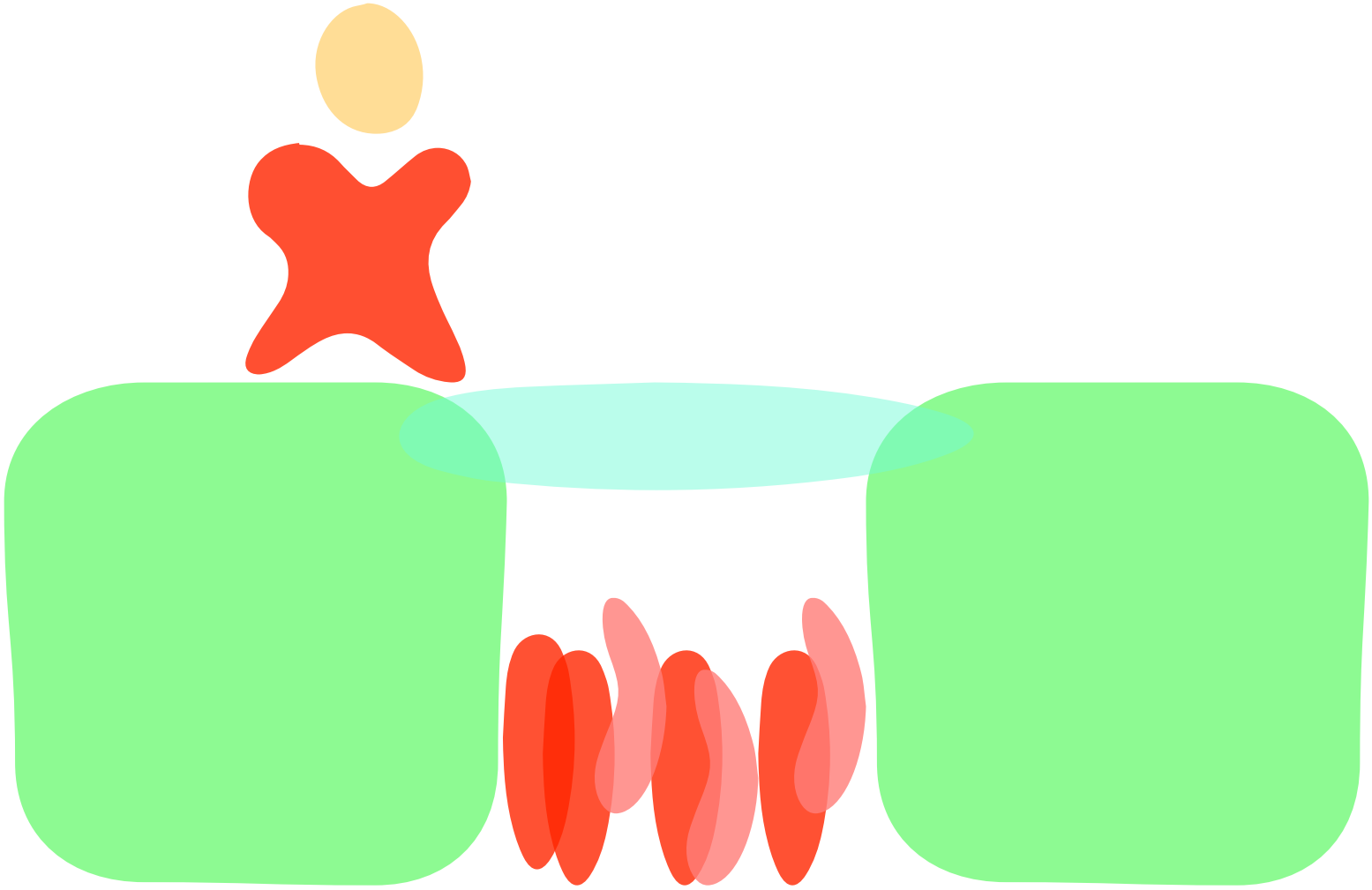
great pattern for product-based applications

# sacrificial architecture




*http://martinfowler.com/bliki/SacrificialArchitecture.html*

# architecture pitfall

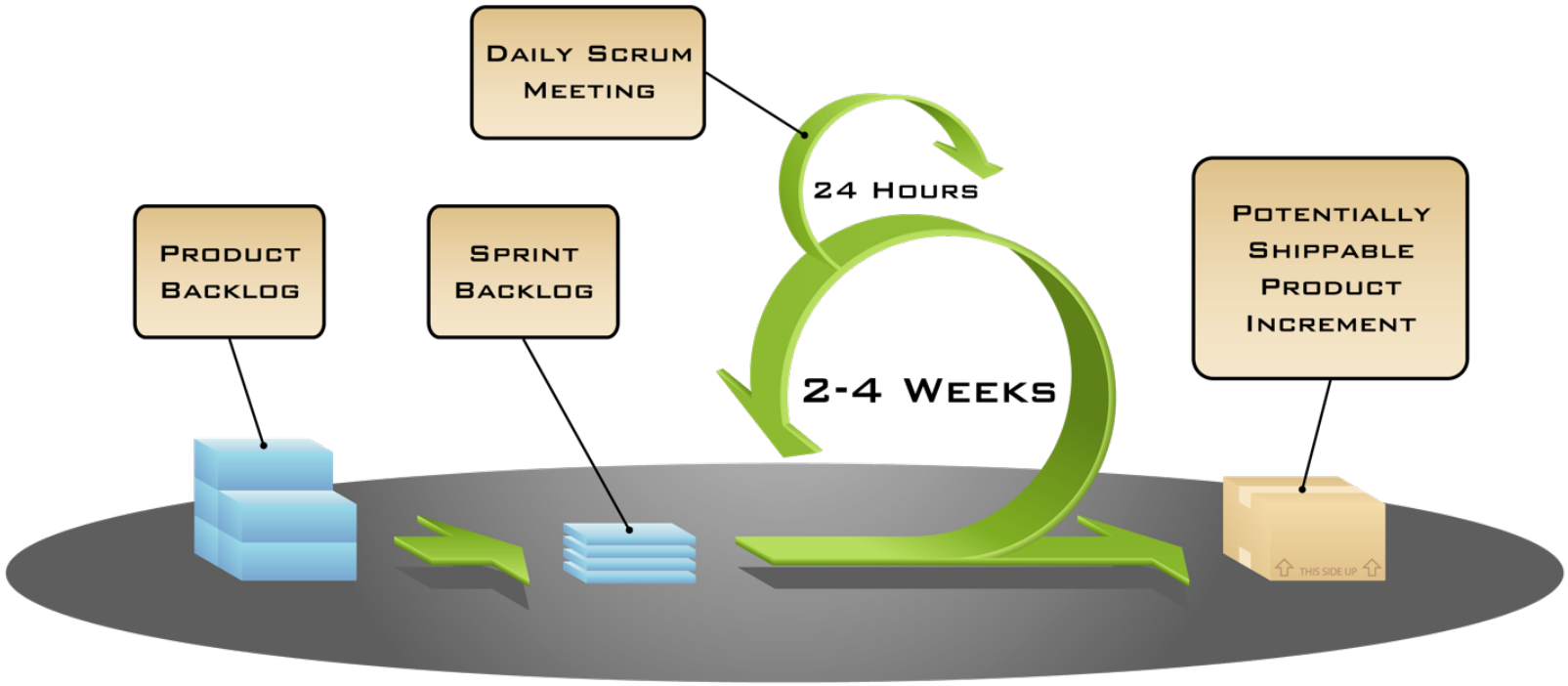


# architecture by implication

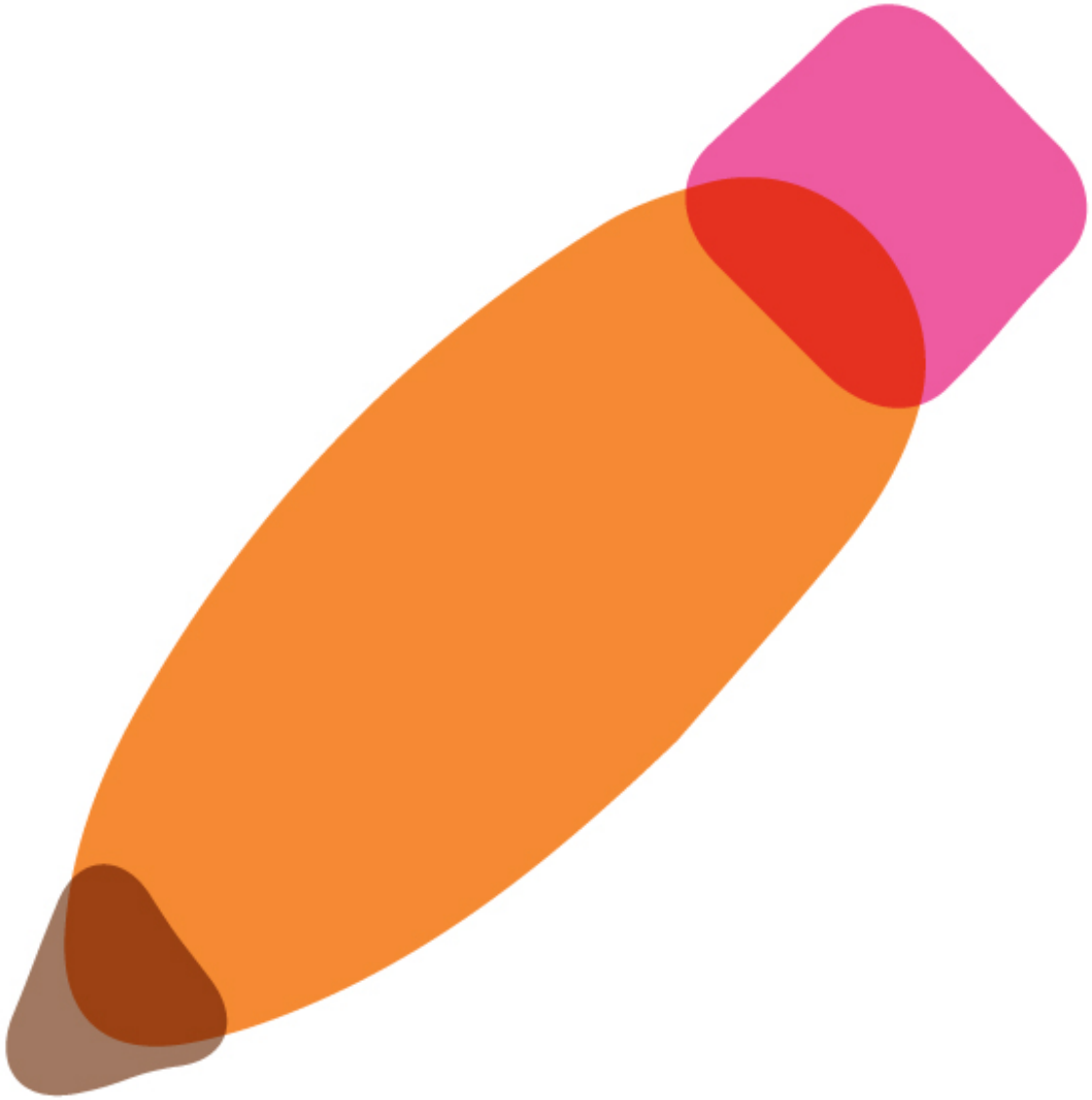
systems lacking a clear documented architecture

# architecture by implication

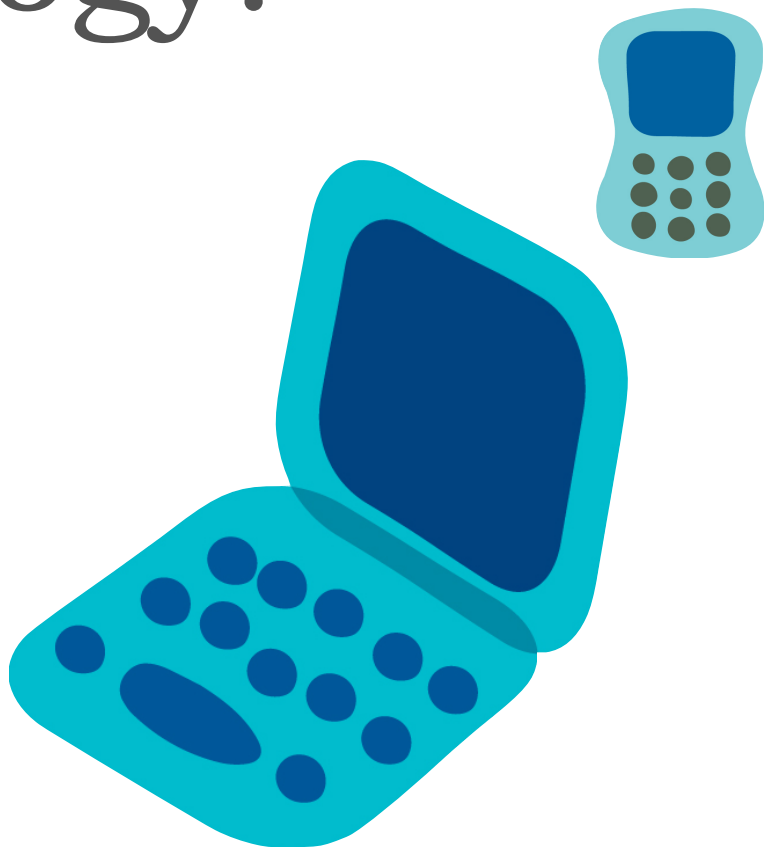
remember that agile methodologies are not a substitute for creating an architecture

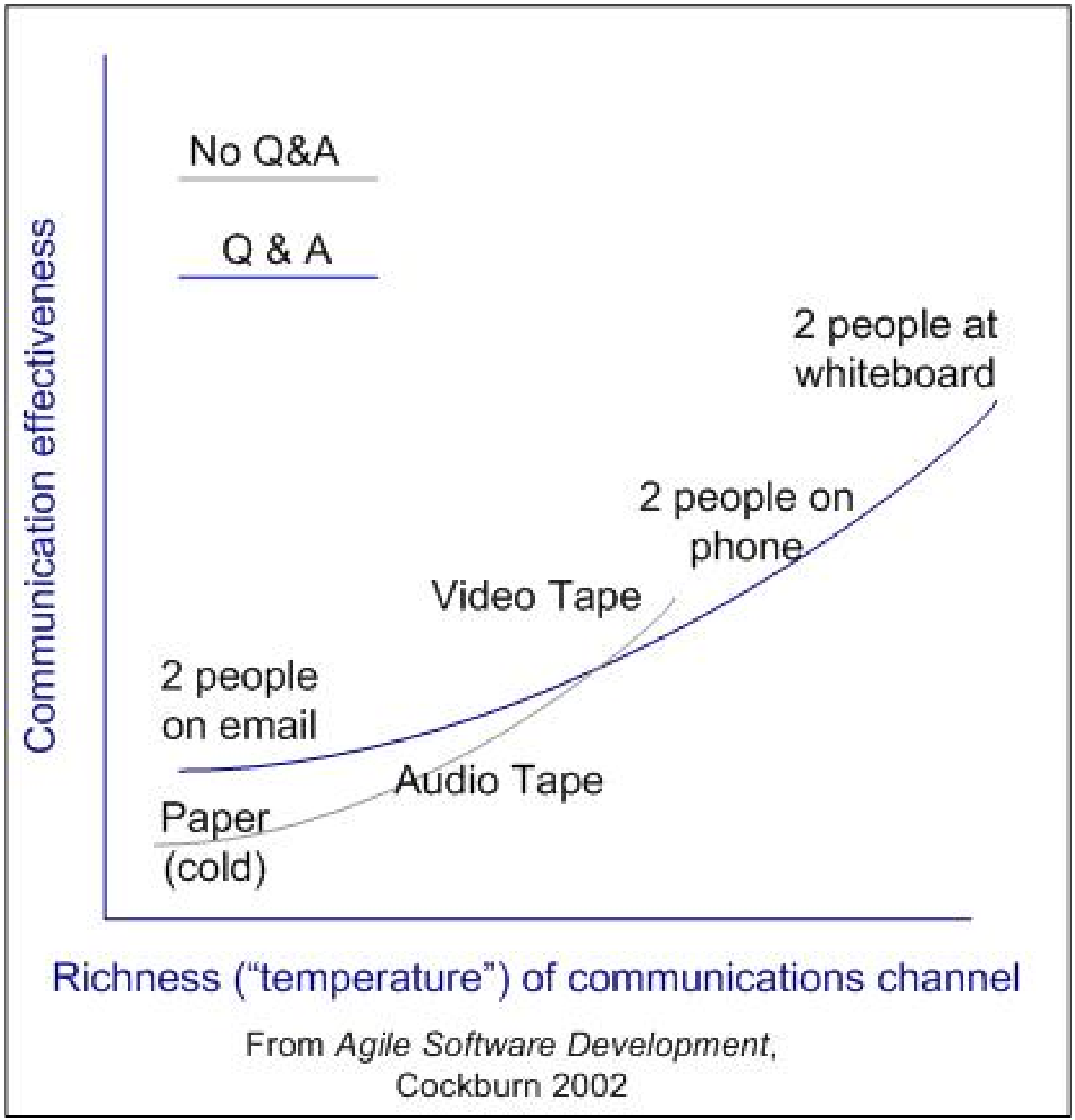


# Technical Writing Skills



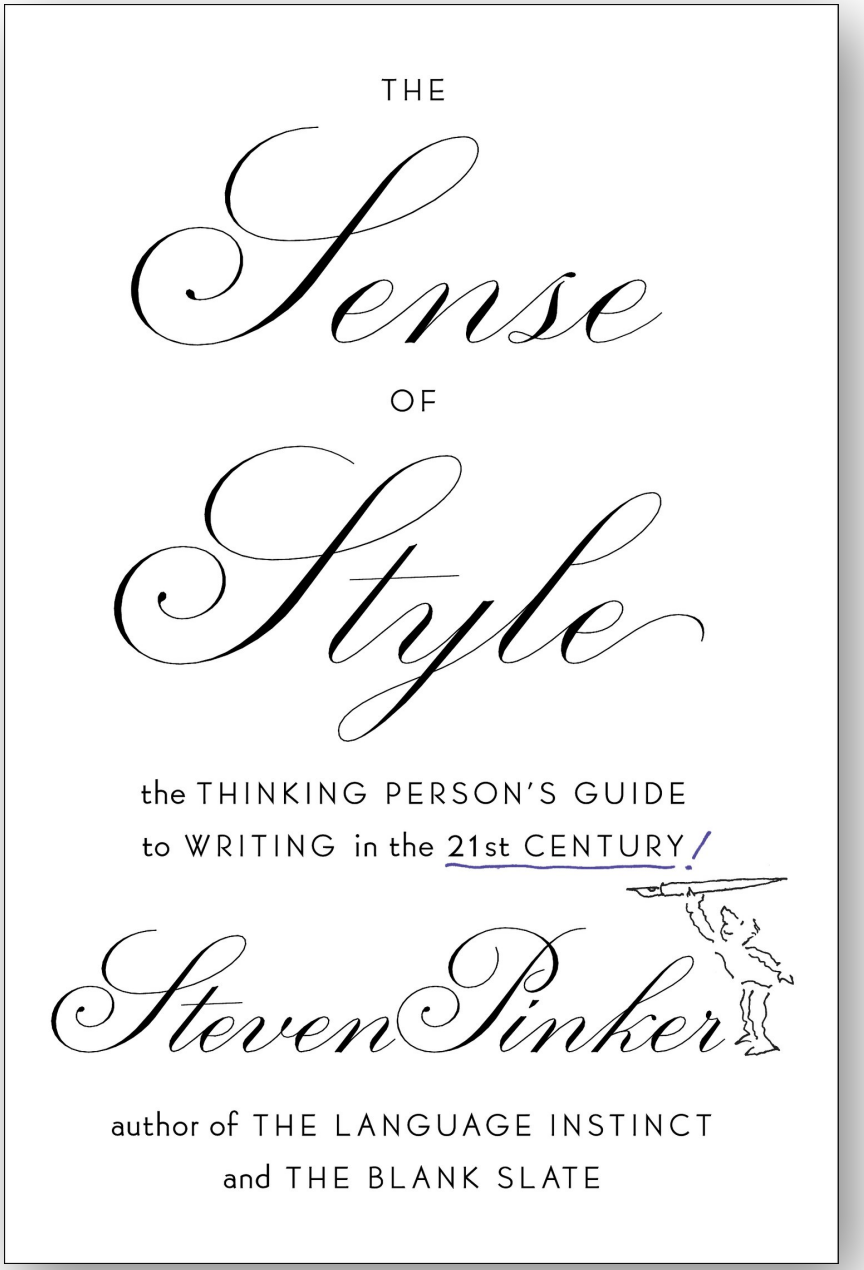
Software is more about communication than technology.






From *Agile Software Development*, Cockburn 2002

# Know Your Audience




practical
vs.
classic style

# passive voice

Passive voice occurs when you make the object of an action into the subject of a sentence.

*Why was the road crossed by the chicken?*

# examples

The metropolis has been scorched by the dragon's fiery breath.
The dragon scorched the metropolis with his fiery breath.

When her house was invaded, Penelope had to think of ways to delay her remarriage.
After suitors invaded her house, Penelope had to think of ways to delay her remarriage.

# passive voice myths

1. Use of the passive voice constitutes a grammatical error.

2. Any use of "to be" (in any form) constitutes the passive voice.

3. The passive voice always avoids the first person.

4. You should never use the passive voice.

5. I can rely on my grammar checker to catch the passive voice.

# more examples

Heart disease is considered the leading cause of death in the United States.

Research points to heart disease as the leading cause of death in the United States.

Researchers have concluded that heart disease is the leading cause of death in the United States.

The balloon is positioned in an area of blockage and is inflated.

The surgeon positions the balloon in an area of blockage and inflates it.

# “swindles & perversions”

*Mistakes were made.*

*The Exxon Company accepts that a few gallons might have been spilled.*

use of language shapes clarity and meaning

some people use the passive voice to avoid mentioning responsibility for certain actions


# it's common

Your phone will join known networks automatically.

Your phone automatically joins known networks.

If no known networks are available, you must manually select a network.

the most important rule:

# revise!

# revise!

all important documentation

proposals

emails !

all written correspondence

# technical writing

simple, declarative sentences

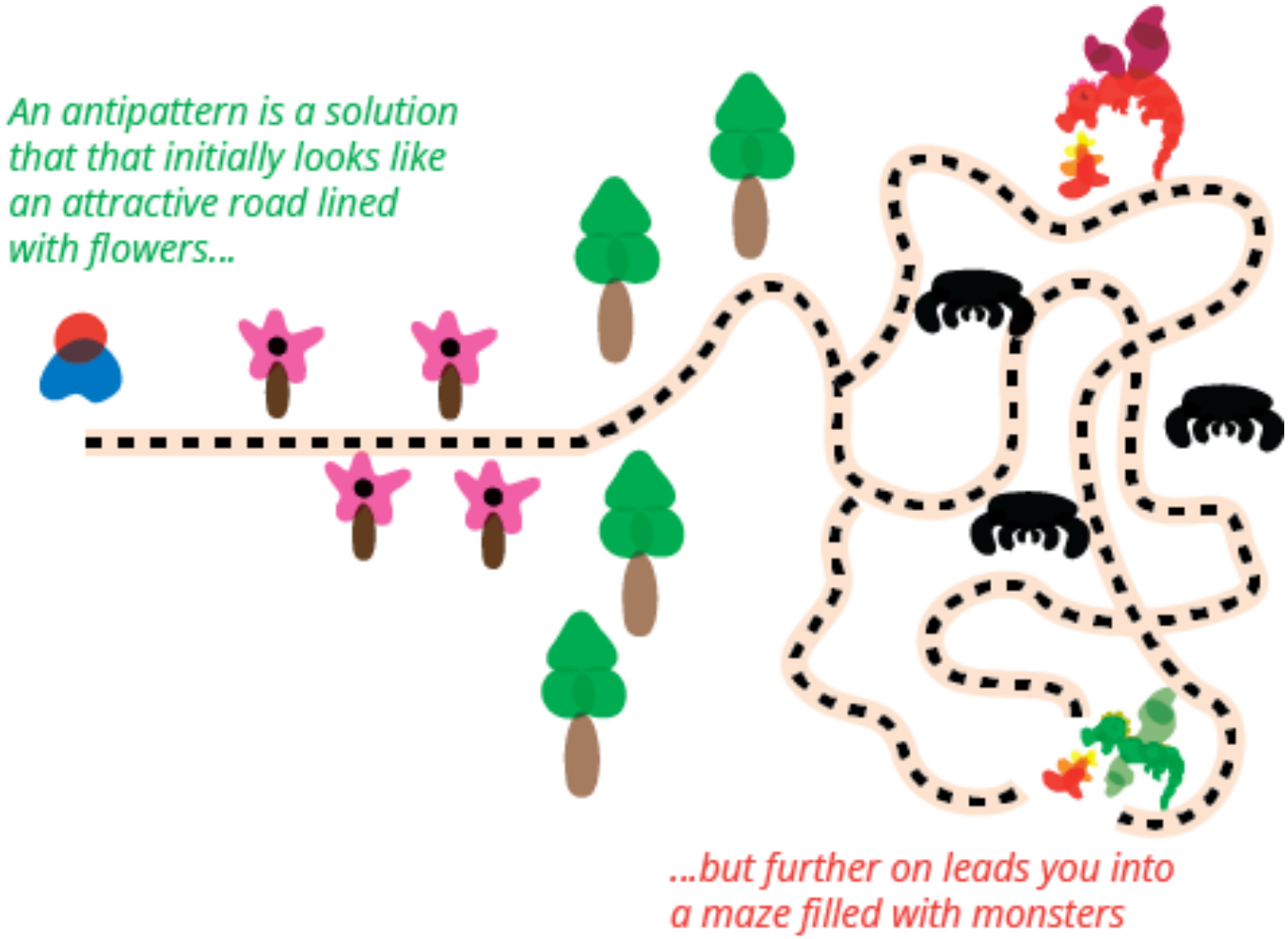
draft & rewrite

…and rewrite and rewrite and rewrite…

spell check!

have someone else read it for clarity

# Architecture Anti-pattern




big bang architecture
designing the entire architecture at the beginning of the project when you know least about the system

# big bang architecture

only architect what is absolutely necessary to get the project started and on the right track

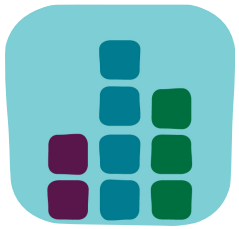
let the architecture evolve throughout the project as you discover and learn more about the system

don't forget - requirements, technology, and business needs change constantly - and so must the architecture

# Continuous Delivery

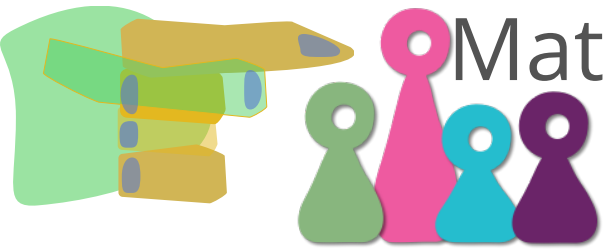
# continuous delivery ∩ architect

Architecture is abstract
until operationalized

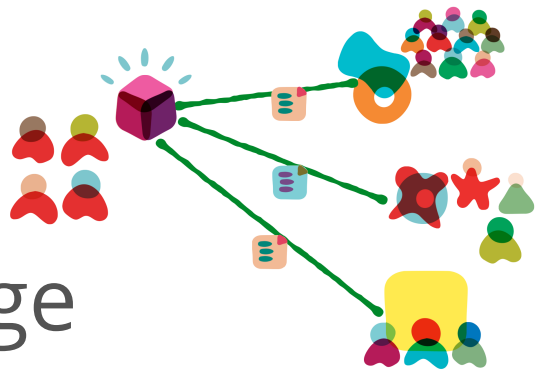


Understanding
shifting structure.

Expanding role
of architect.

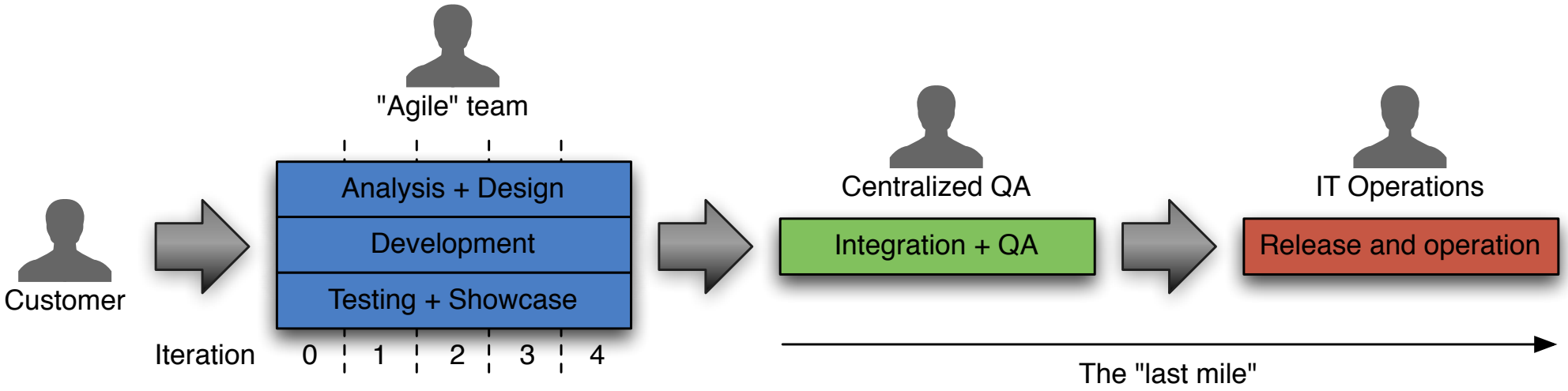


Mature engineering
practices.

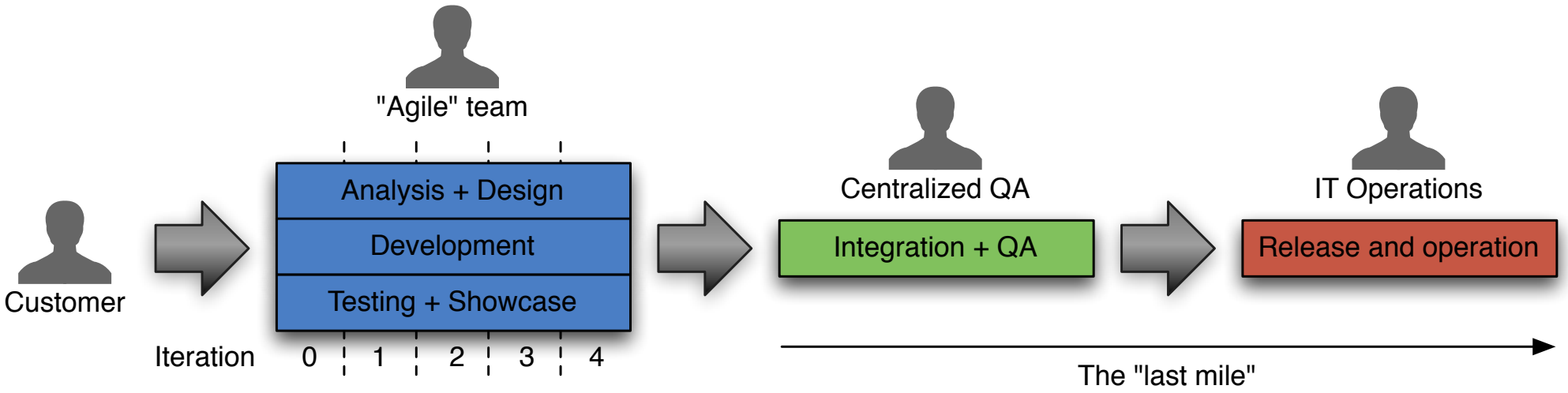


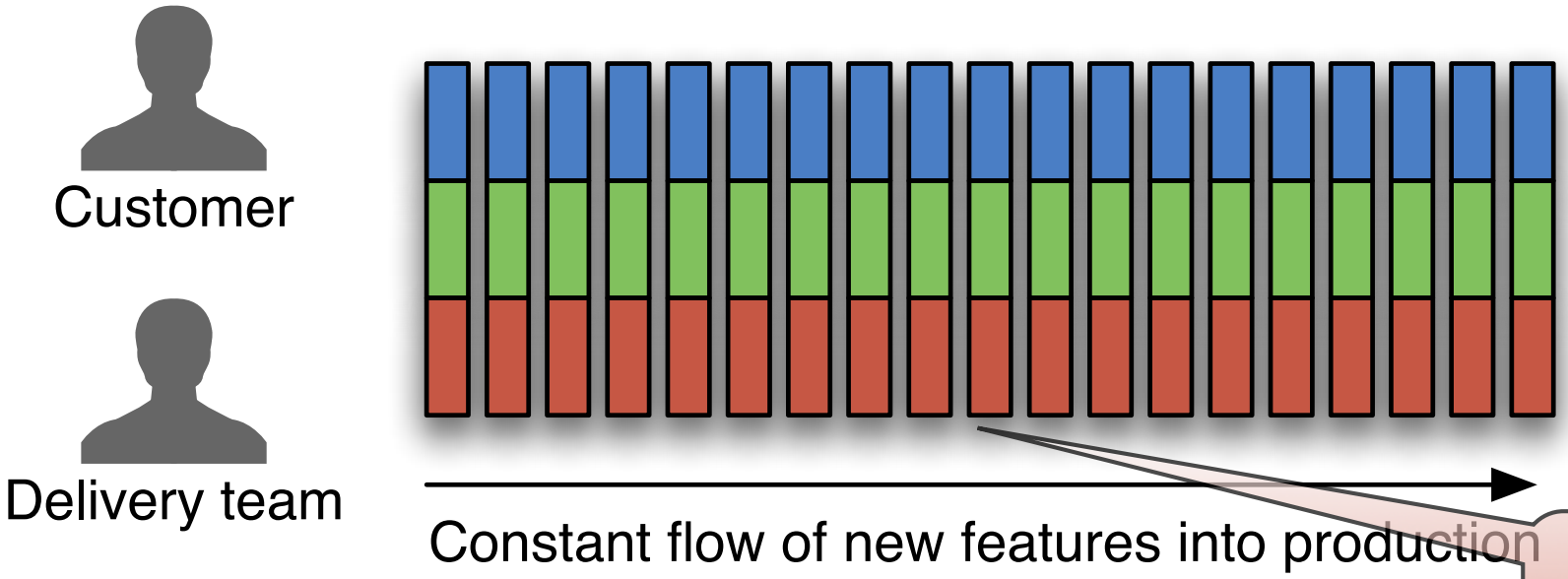
Manage
coupling intelligently.

# agile 101


"Agile" team
Analysis + Design
Development
Testing + Showcase
Customer
Iteration
0
1
2
3
4
Centralized QA
Integration + QA
IT Operations
Release and operation
The "last mile"

# continuous delivery




business needs > operational constraints

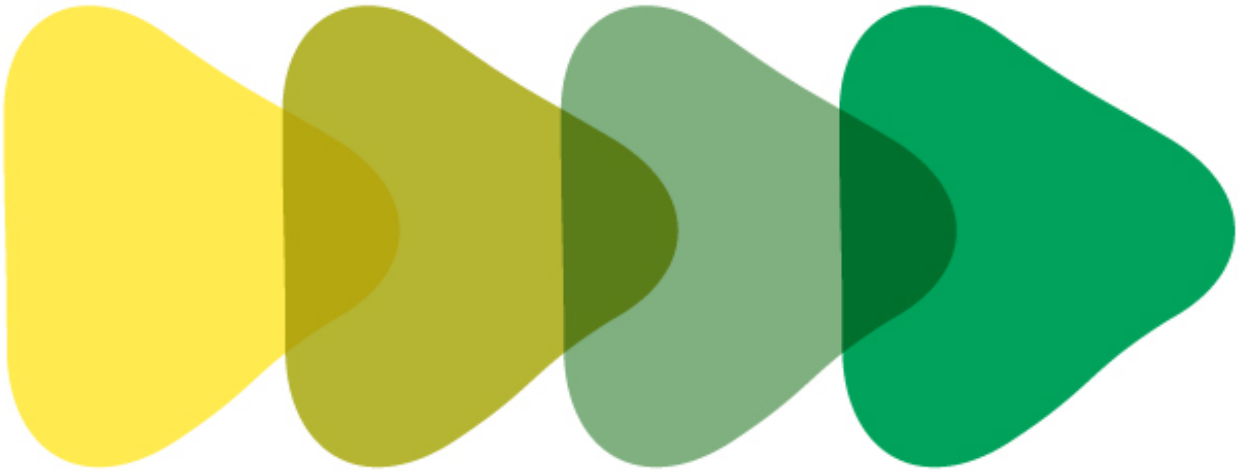
# Continuous Integration

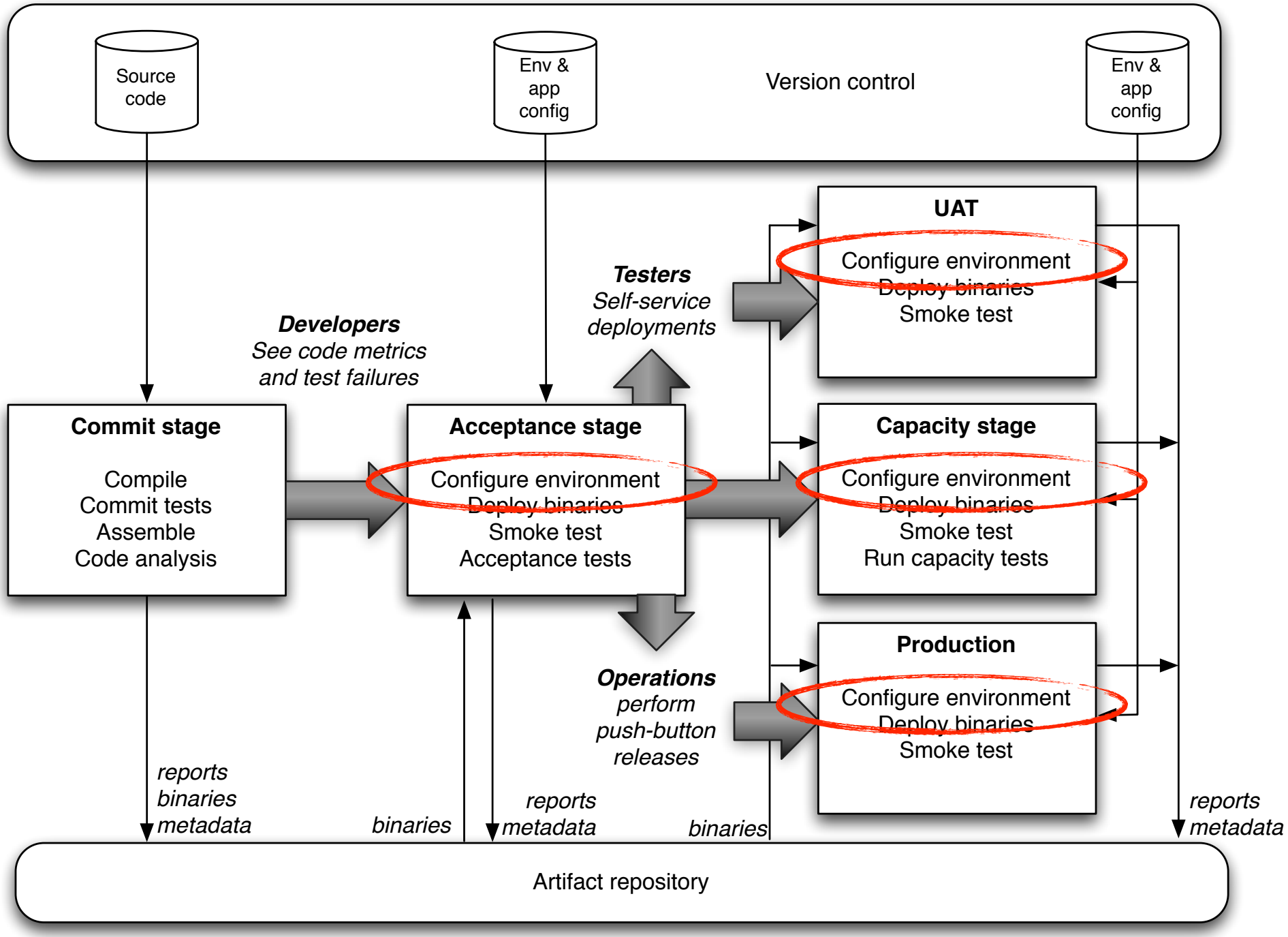
Fast, automated feedback on the correctness of your application every time there is a change to code

# Deployment Pipeline

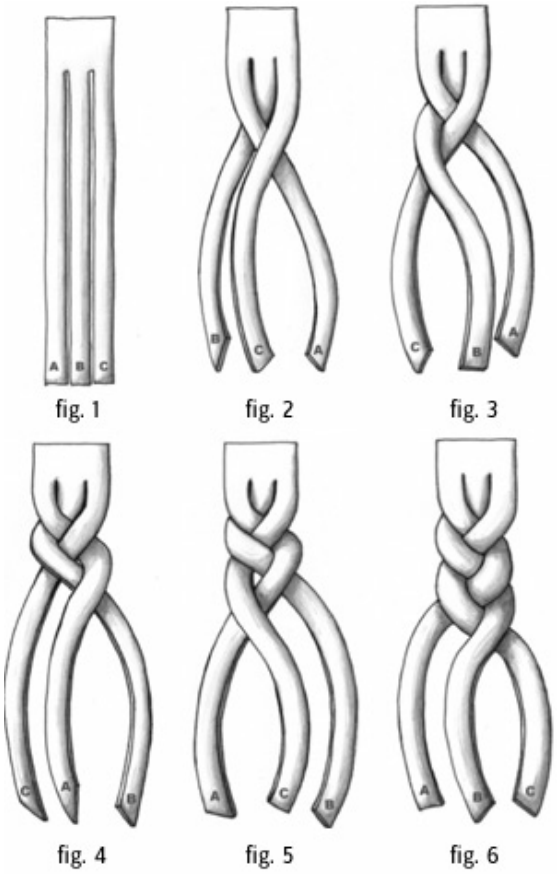
Fast, automated feedback on the **production readiness** of your application every time there is a change — to **code**, **infrastructure**, or **configuration**

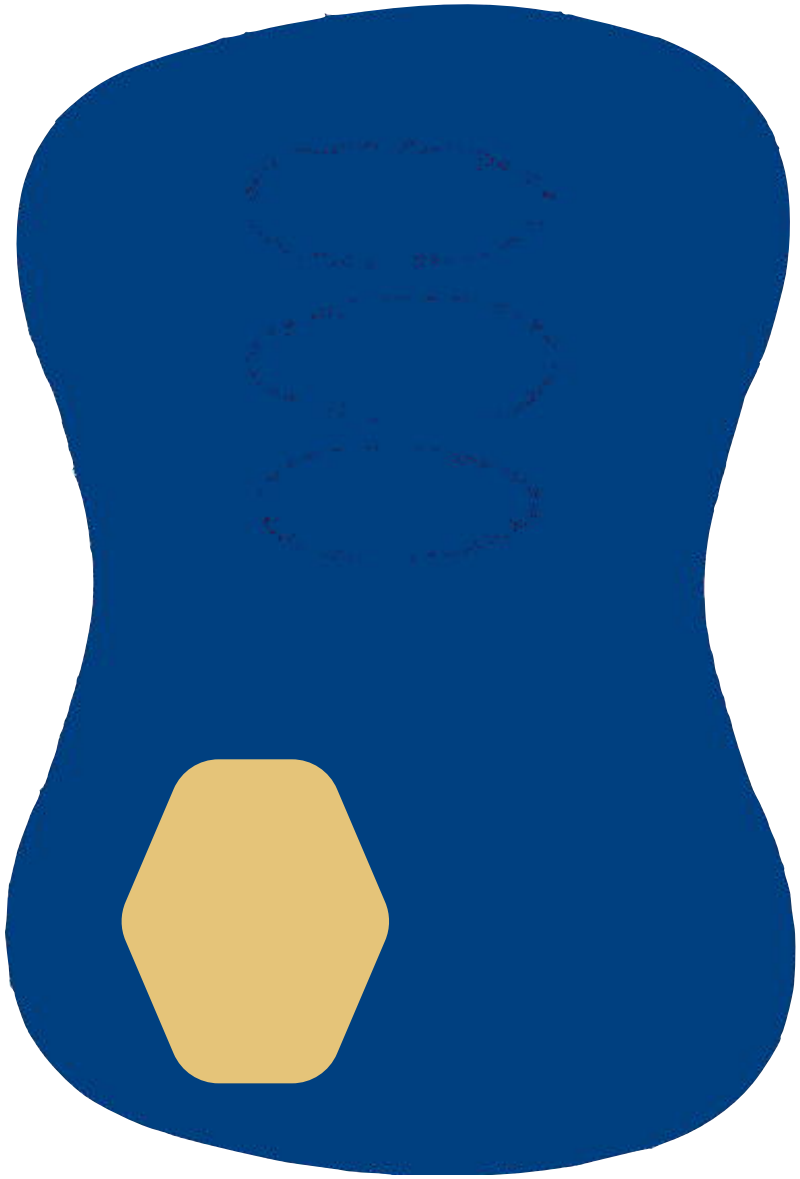
# Deployment Pipelines




Source code
Env & app config
Version control
Env & app config
UAT
Configure environment
Deploy binaries
Smoke test
Testers
Self-service deployments
Developers
See code metrics and test failures
Commit stage
Compile
Commit tests
Assemble
Code analysis
Acceptance stage
Configure environment
Deploy binaries
Smoke test
Acceptance tests
Capacity stage
Configure environment
Deploy binaries
Smoke test
Run capacity tests
Production
Configure environment
Deploy binaries
Smoke test
Operations
perform push-button releases
reports
binaries
metadata
binaries
reports
metadata
binaries
reports
metadata
Artifact repository

# Complected Deployments



**complect**, *transitive verb*:
intertwine, embrace, especially
to plait together

production

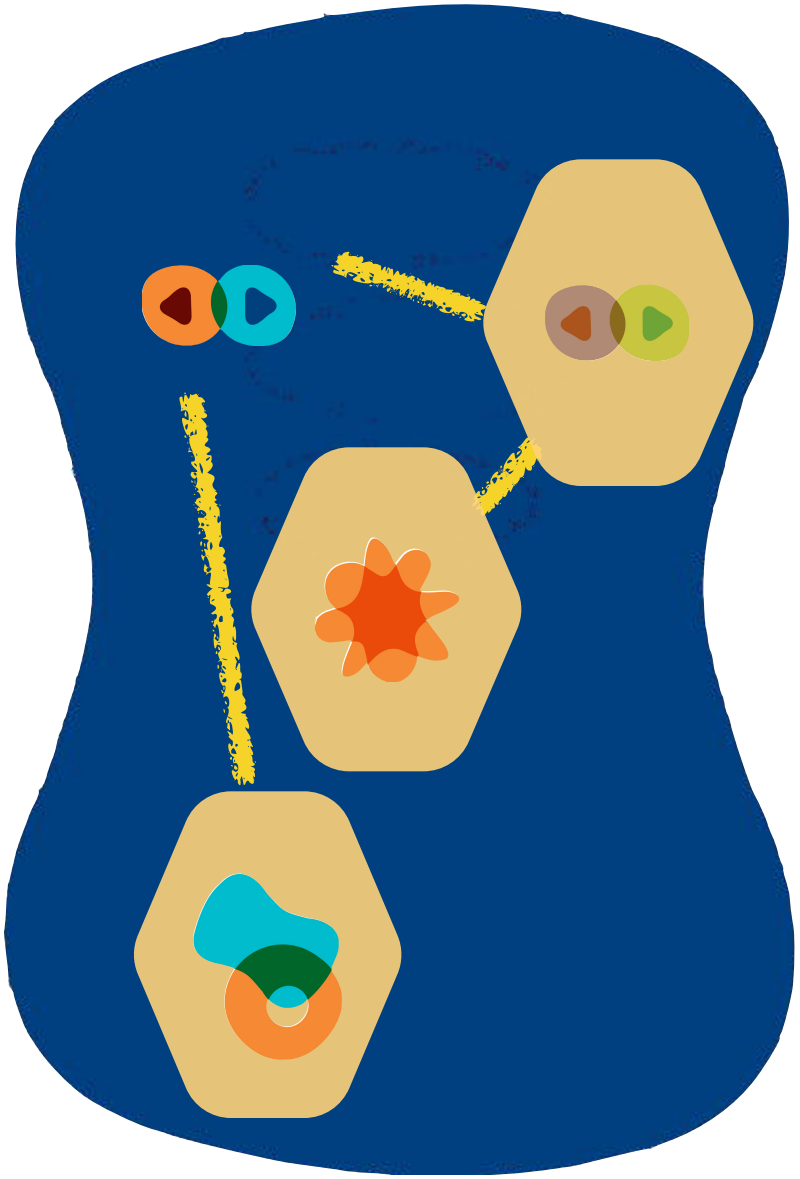
# Evolutionary Architecture

Components are *deployed*.



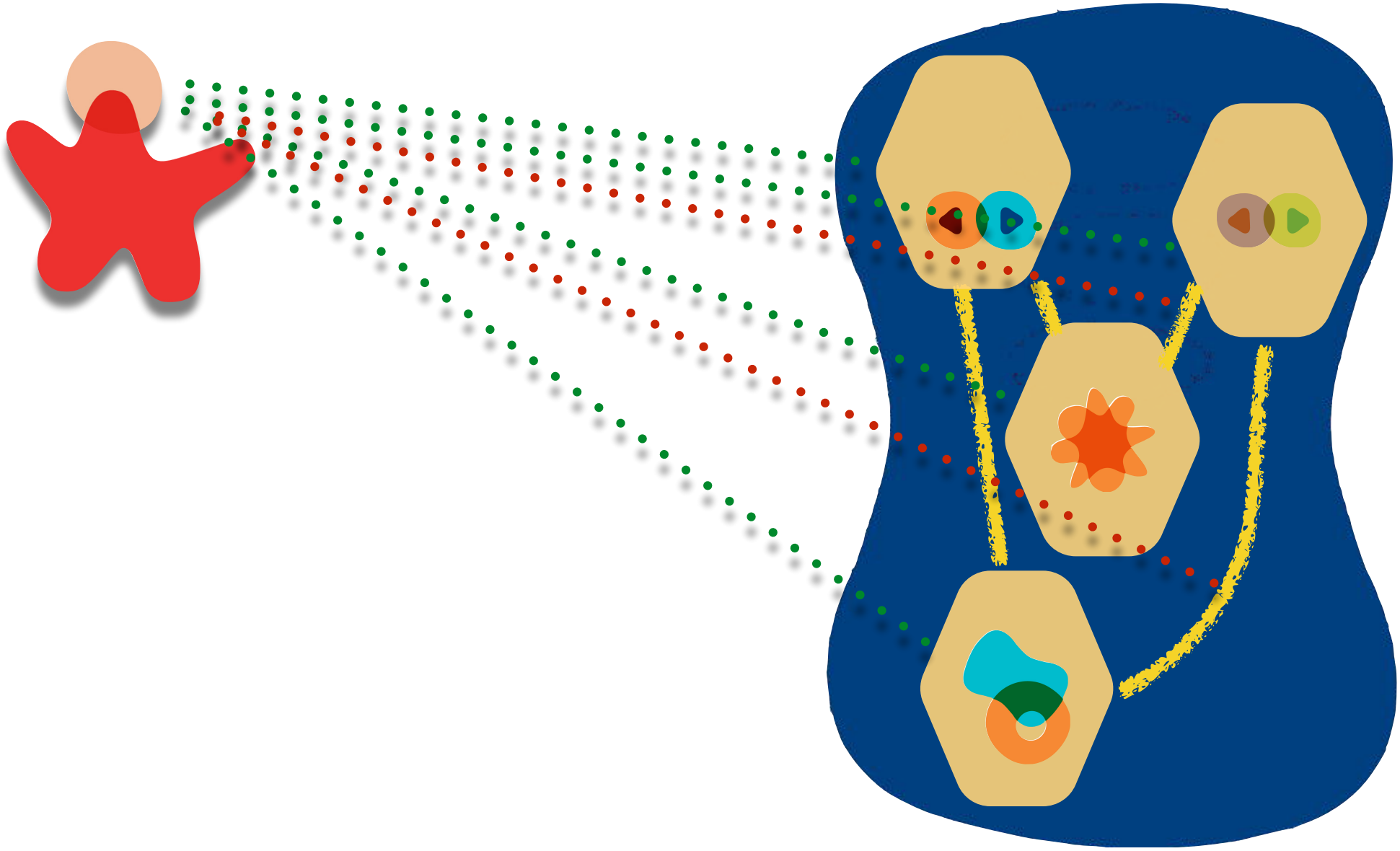
Features are *released*.

Applications consist of *routing*.



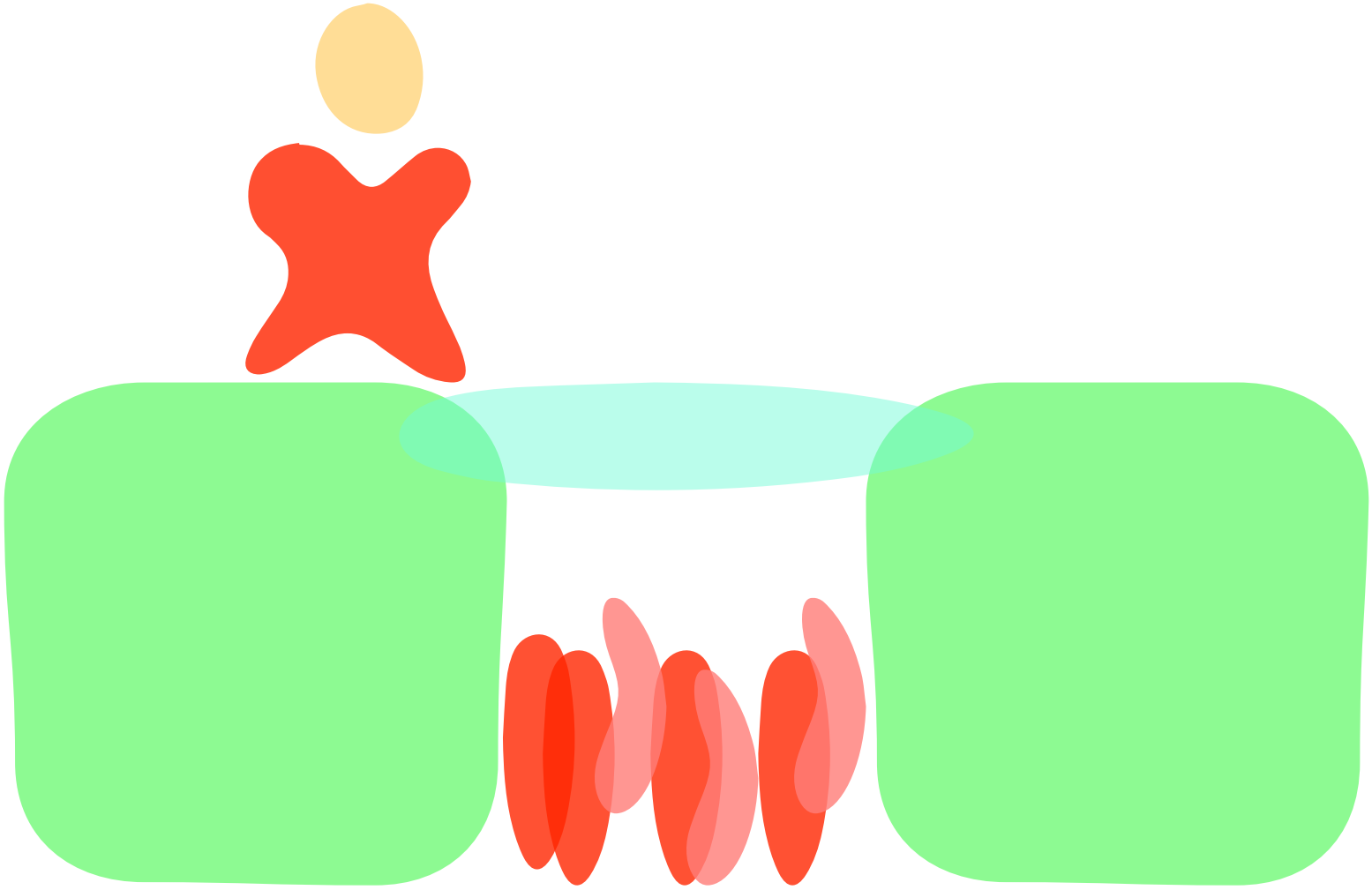
production

# Evolutionary Architecture

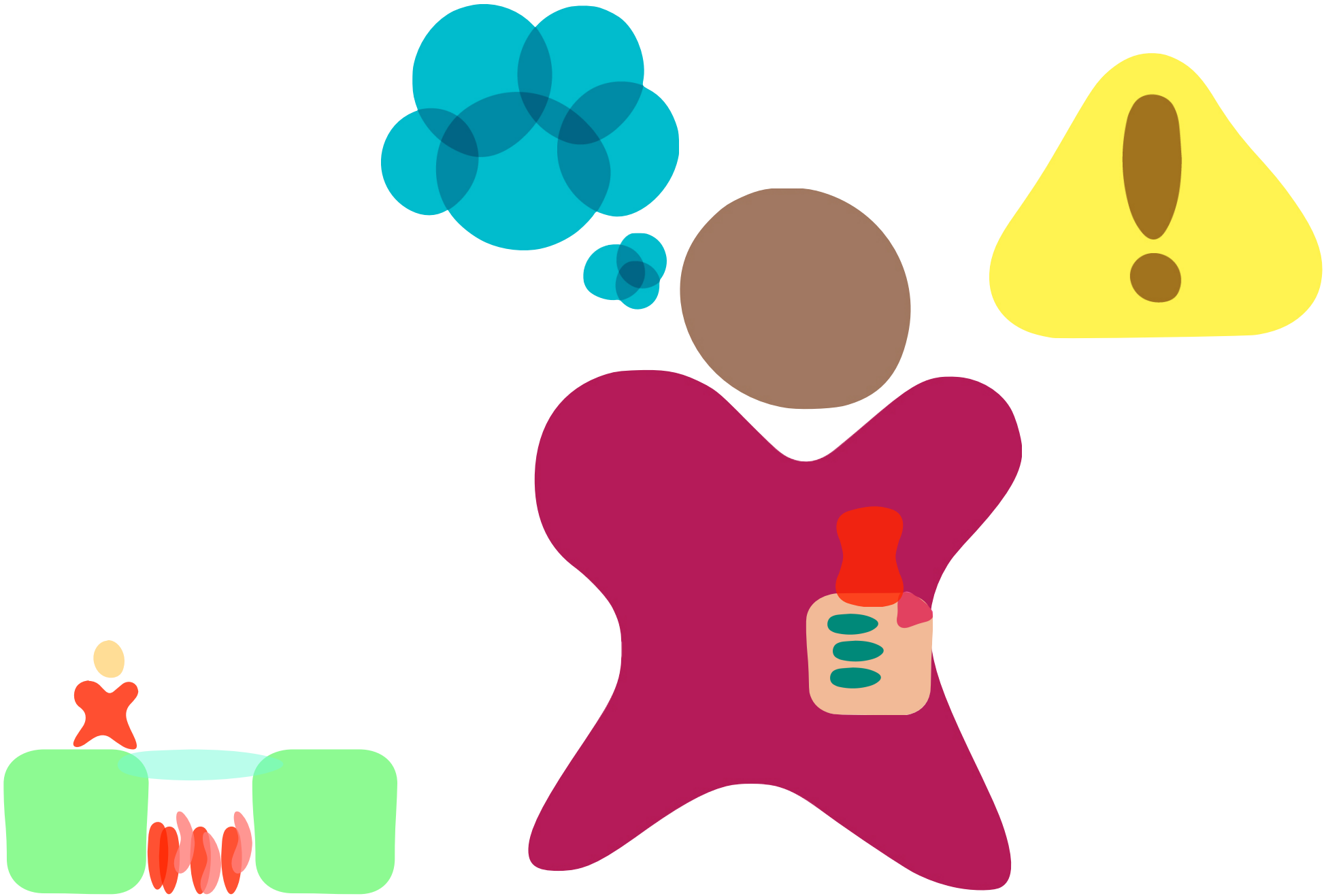


production

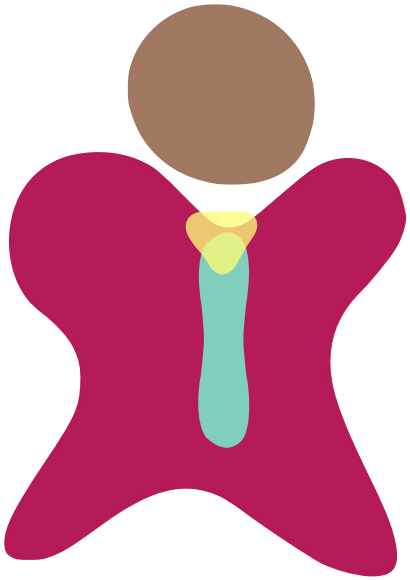
# architecture pitfall



# “accidental” architect



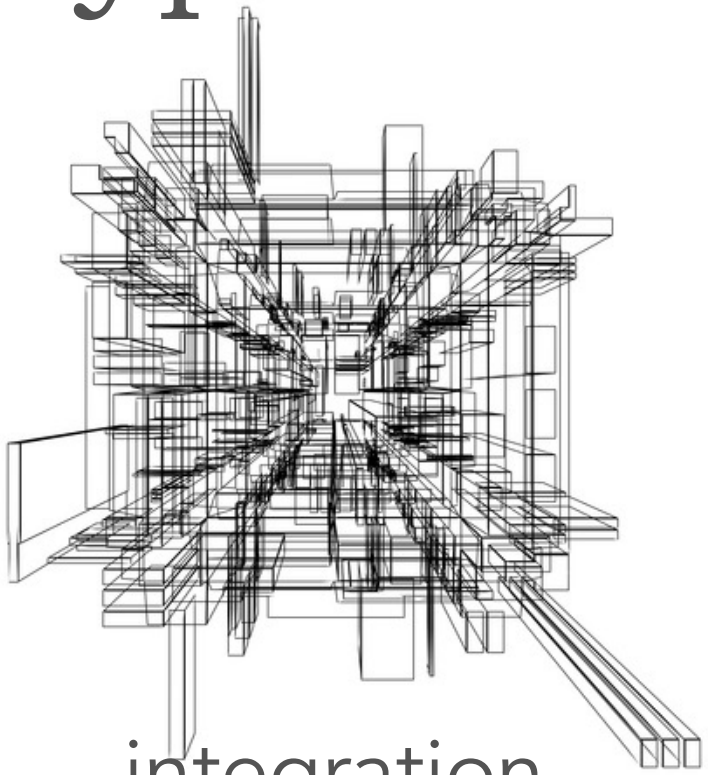
# role versus title



# architecture types

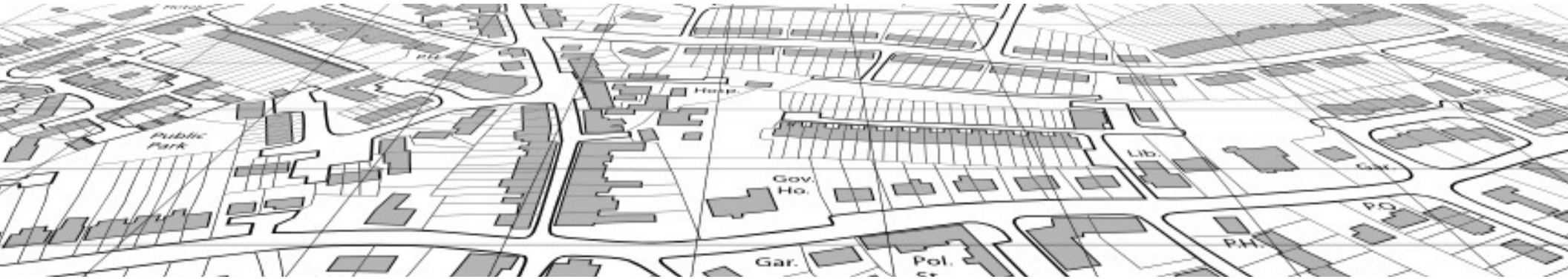
# architecture types

application

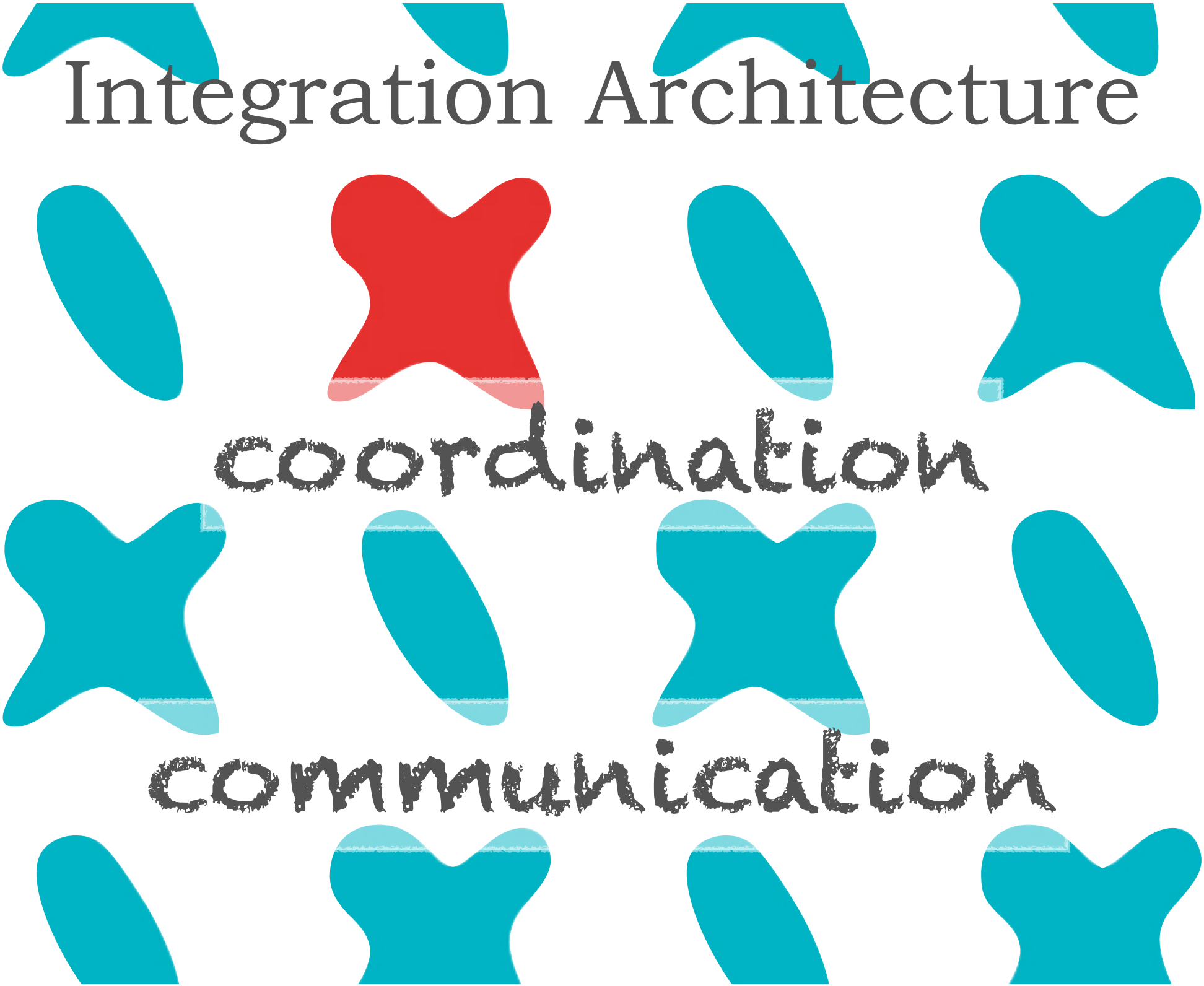


integration

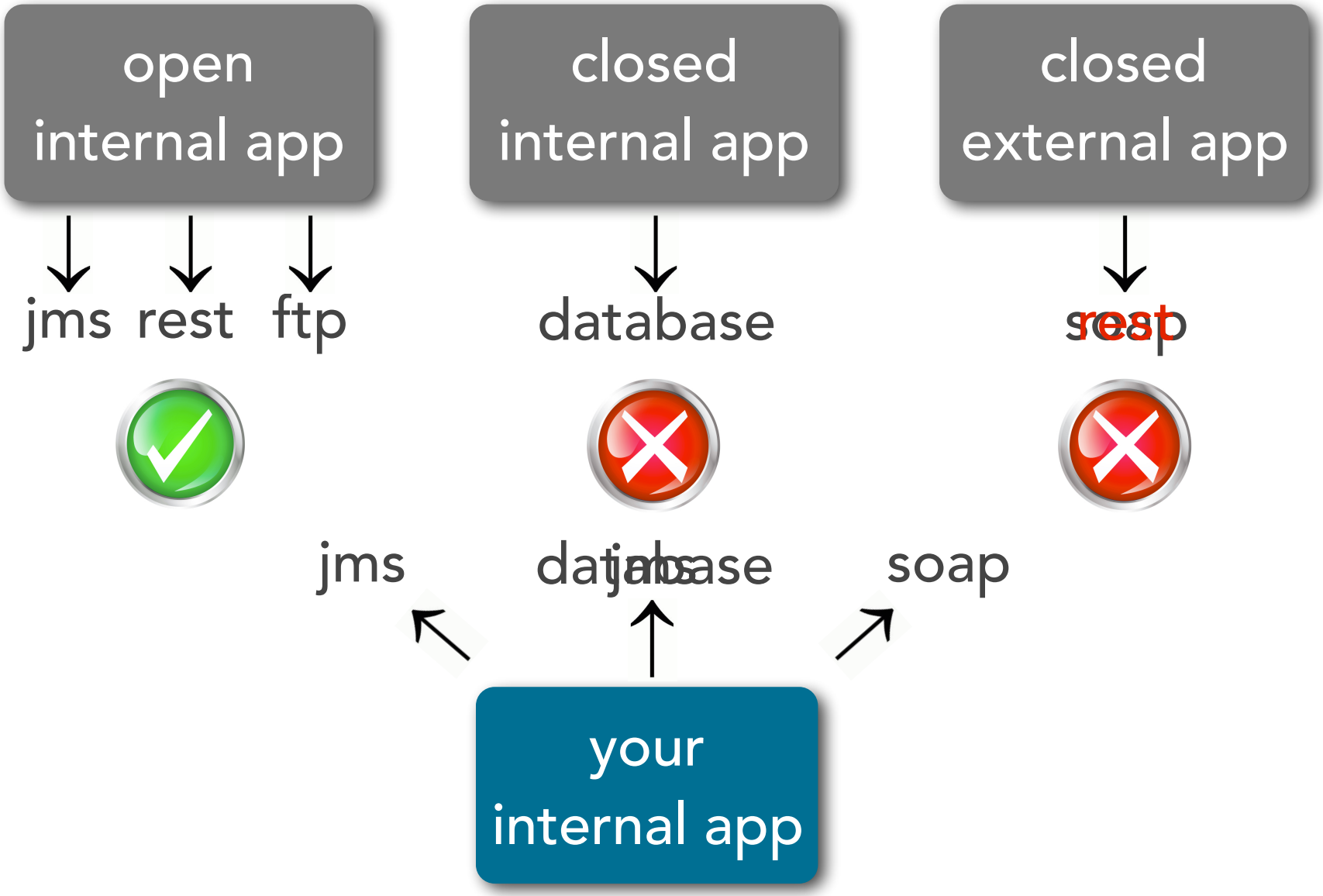
enterprise

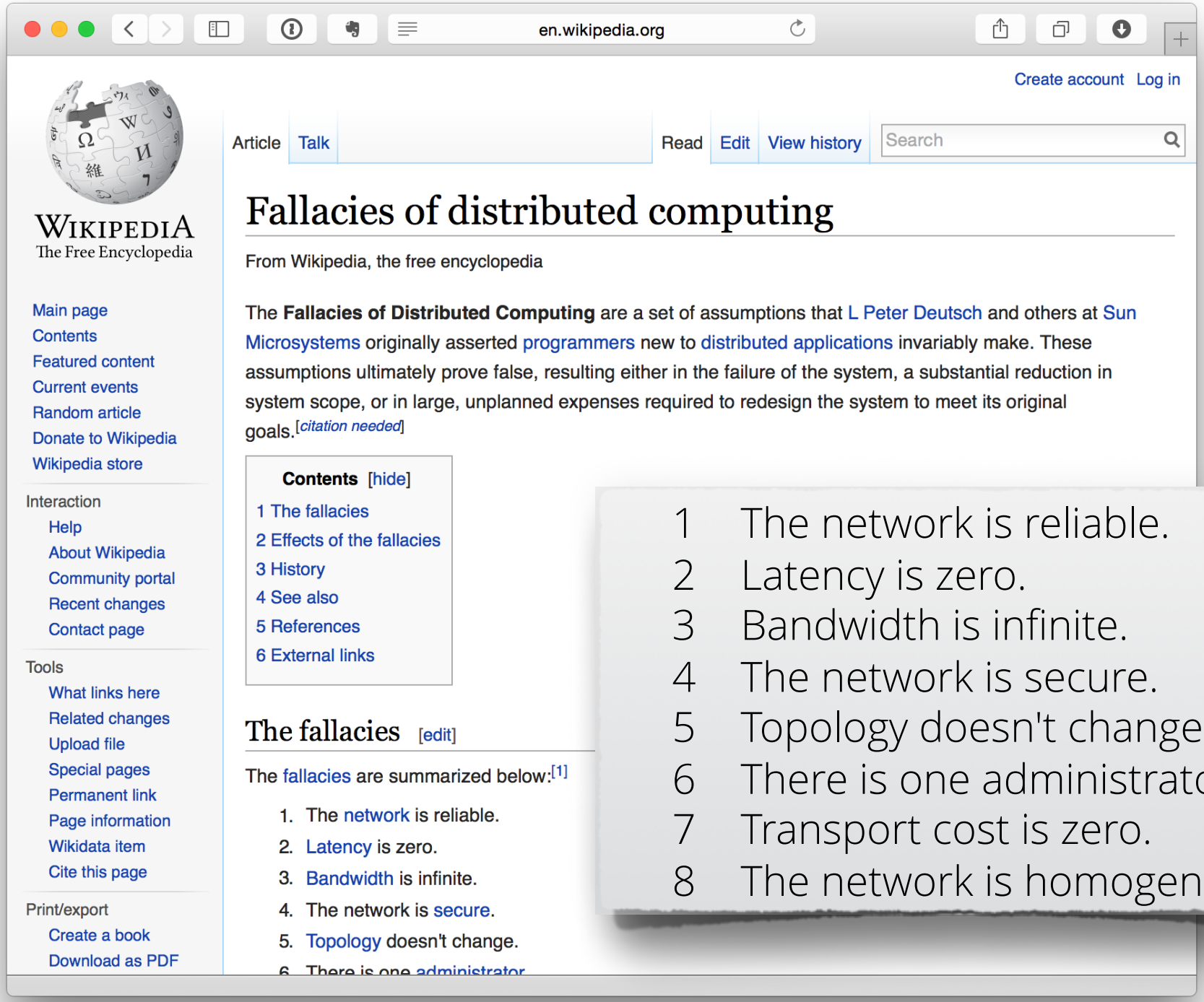


# Integration Architecture



# challenges

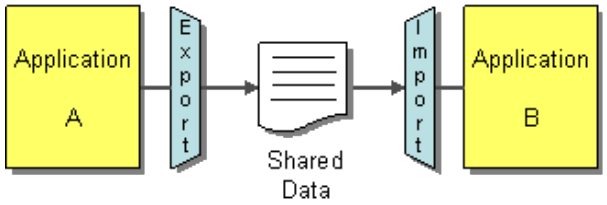

open
internal app
closed
internal app
closed
external app
jms
rest
ftp
database
soap
jms
database
soap
your
internal app



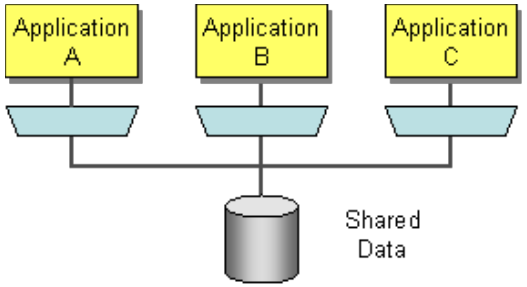

en.wikipedia.org/wiki/Fallacies_of_distributed_computing

# integration styles

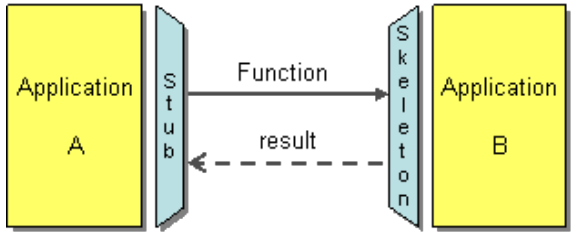
*From Enterprise Integration Patterns by Hohpe and Woolf*



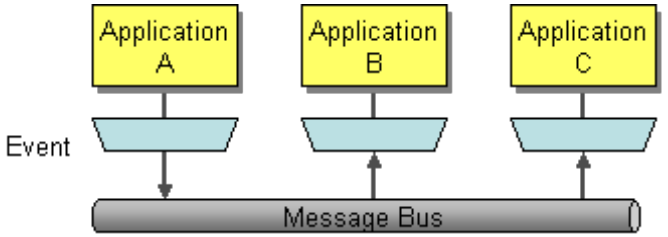

file transfer




shared database

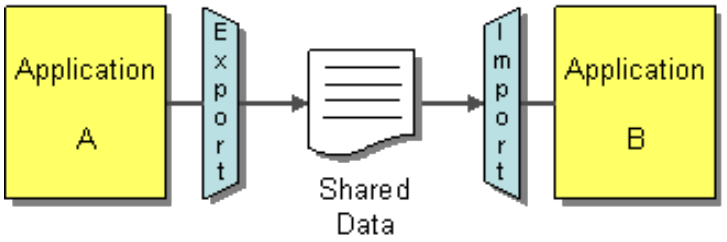



remote procedure invocation




messaging

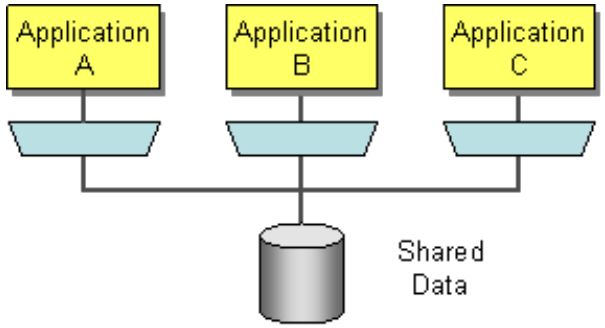
# file transfer




universal integration style, integration simplicity, system decoupling and system abstraction

file-based processing is expensive, error processing, timeliness of data synchronization, data-only transfer

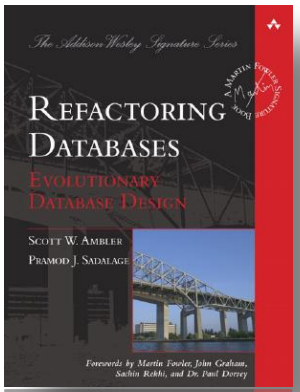
# shared database

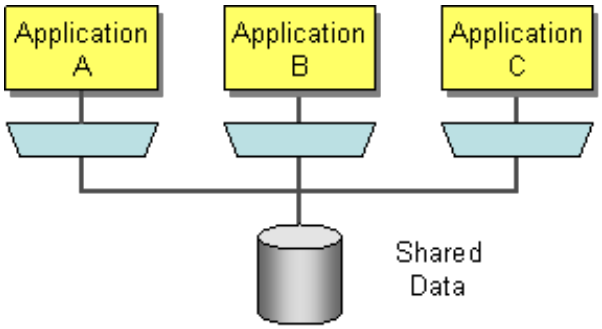


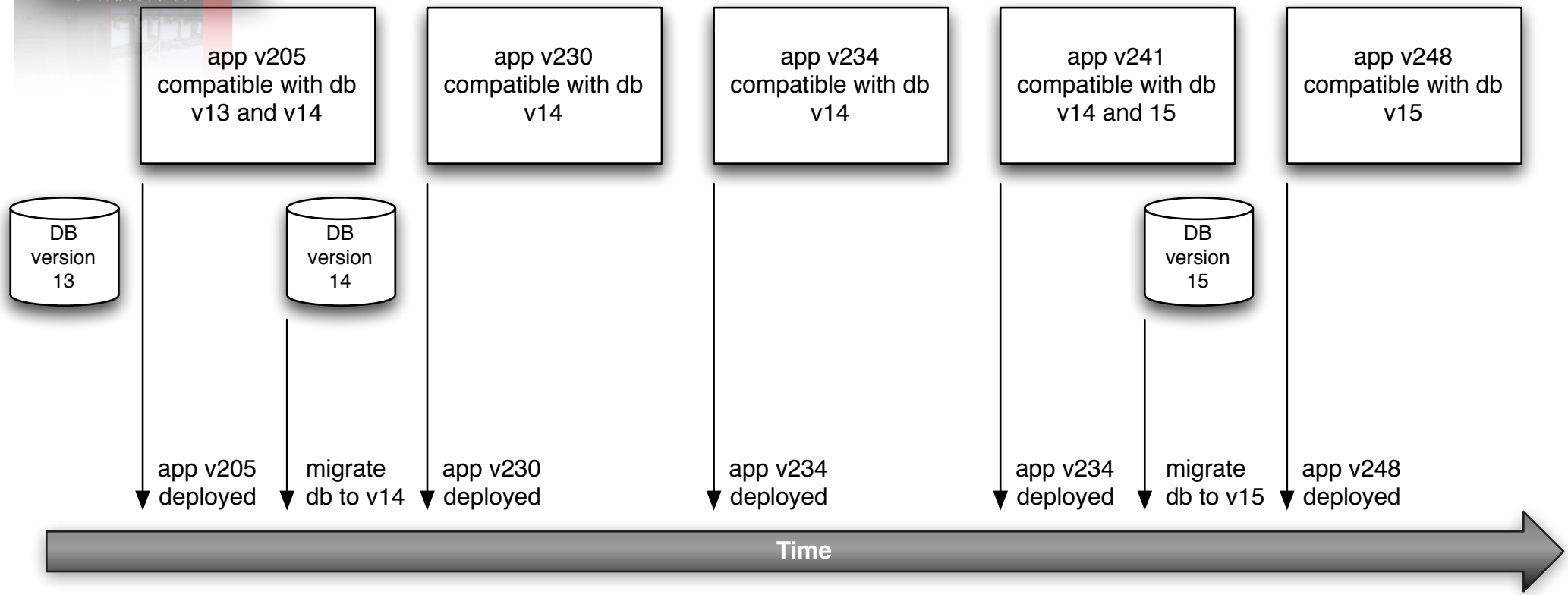

near-universal integration via SQL, system abstraction, system decoupling

cannot use persistence caching (ORM), performance bottleneck issues, schema change issues, data ownership issues

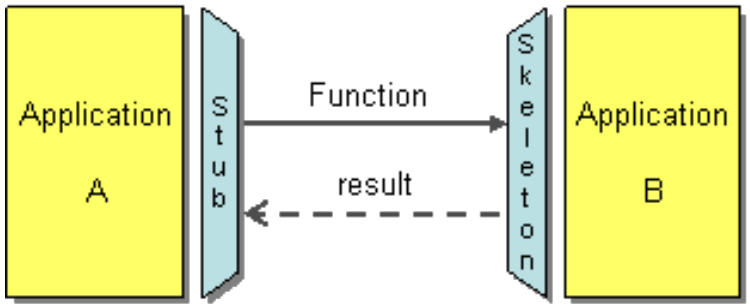
# expand/contract pattern



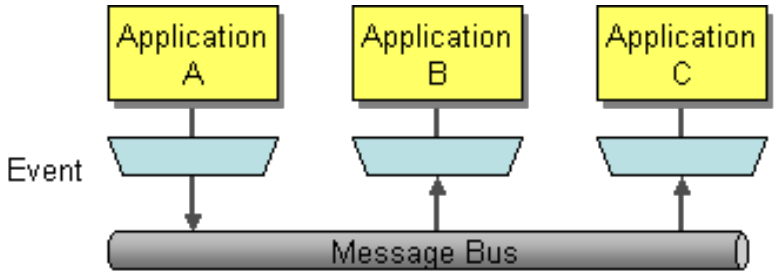
# remote procedure




data encapsulation and ownership, external systems integration via web services, mature frameworks and tools

tight system coupling due to dependency on service availability and location knowledge, poor asynchronous communications

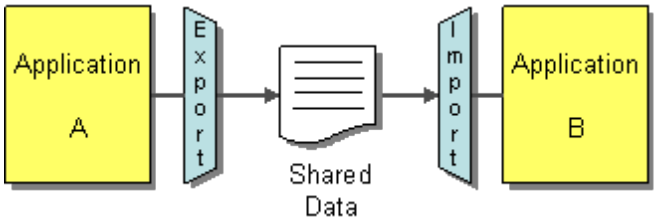
# messaging



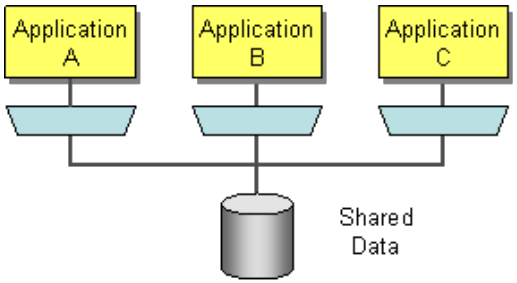

asynchronous and reliable messaging, highly decoupled systems, excellent scalability capabilities, monitoring

external integration beyond firewall, implementation and testing complexity, cross platform standards still evolving

# which is the best integration style?

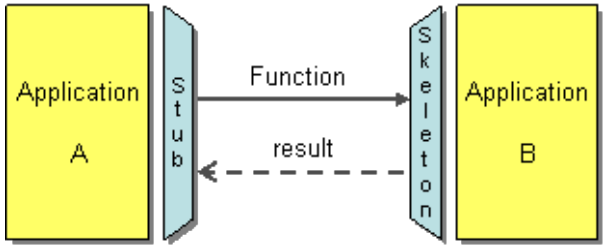



file transfer

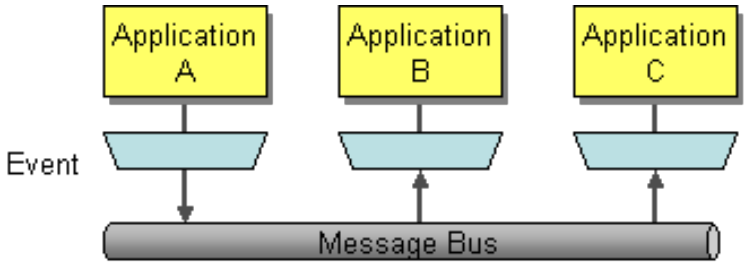



shared database

?

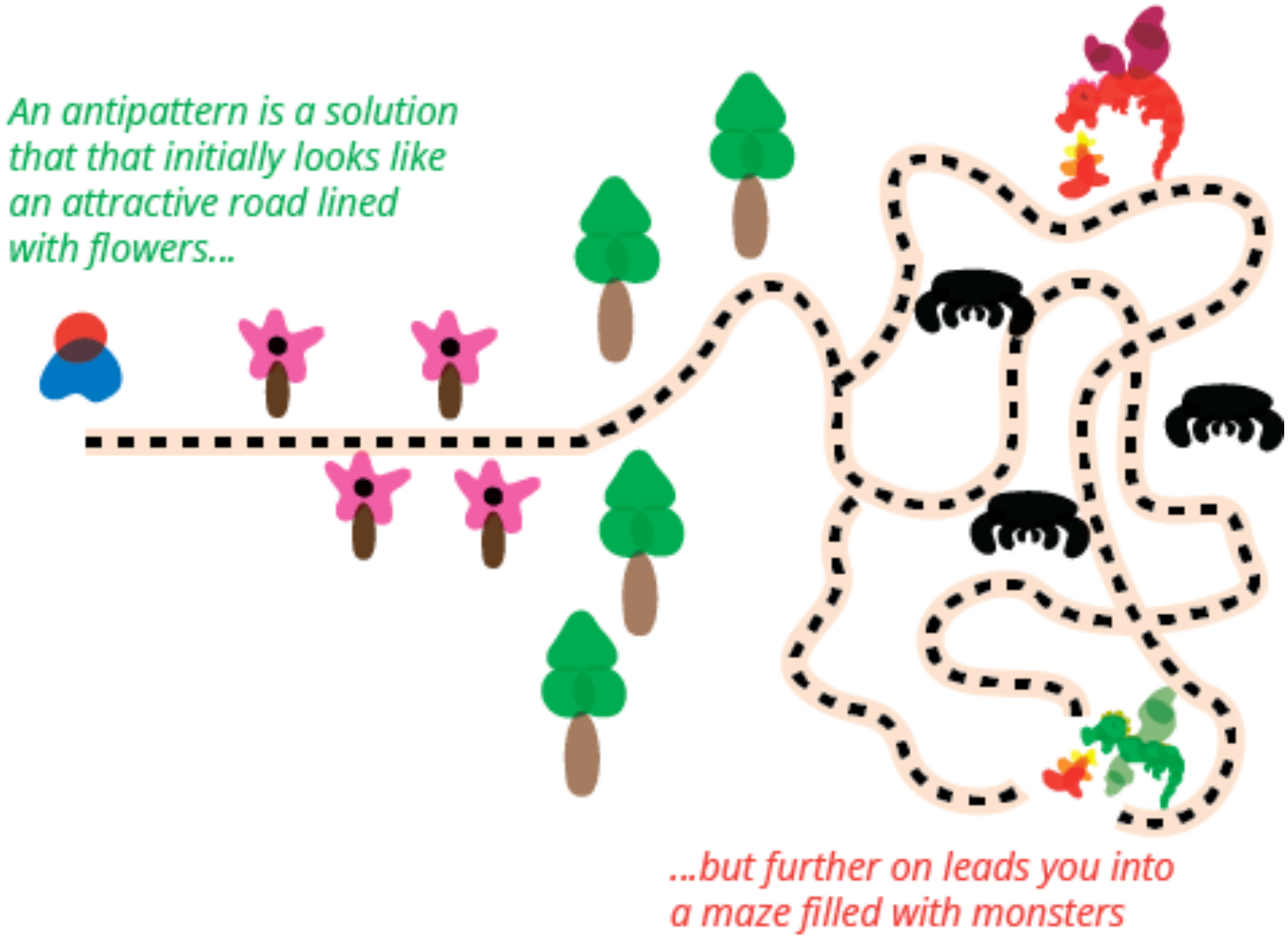



remote procedure invocation




messaging

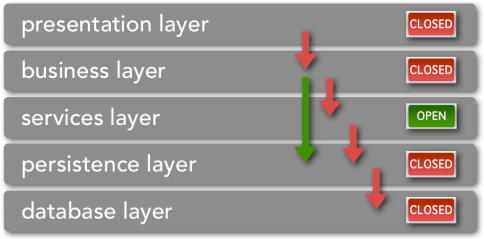
# Architecture Anti-pattern



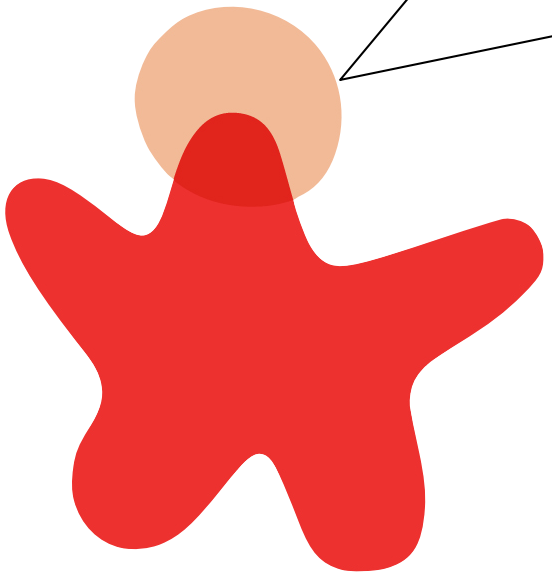
# cover your assets

continuing to document and present alternatives
without ever making an architecture decision

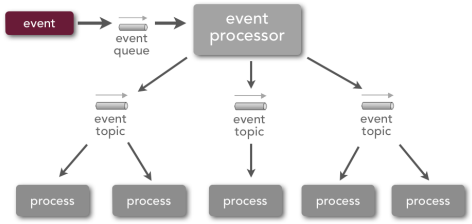
# cover your assets



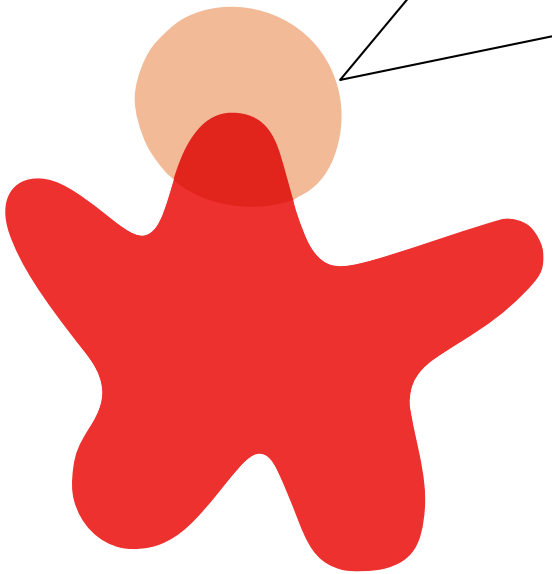

"the layered architecture approach would work here..."



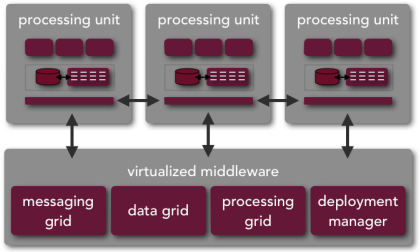
# cover your assets



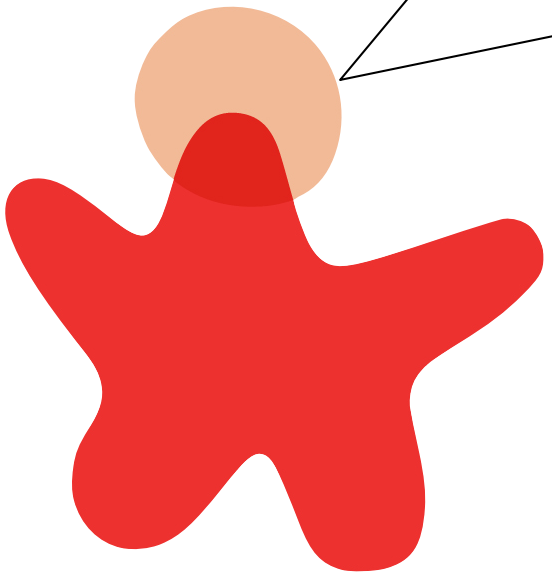

"but of course, there's always EDA, which would also be a fit..."



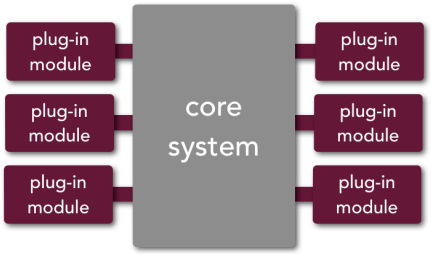
# cover your assets



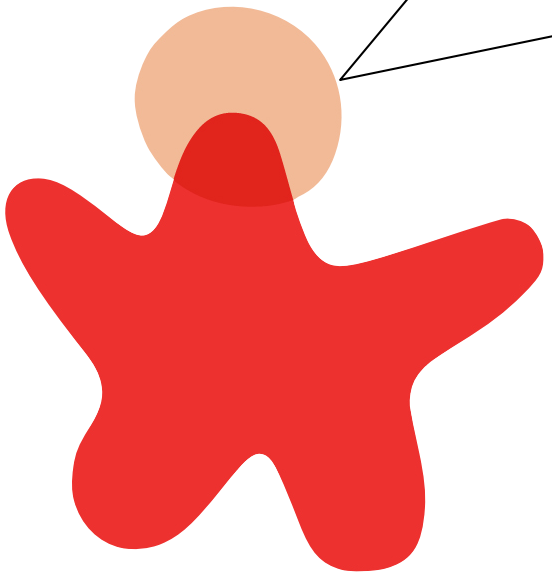

"space-based architecture has always been a safe choice in these situations..."



# cover your assets




"but then again, the microkernel pattern has some real selling points here...



# cover your assets

it's your job as an architect to present alternatives, clearly articulate the pros and cons of each, and **recommend the best solution for the situation**

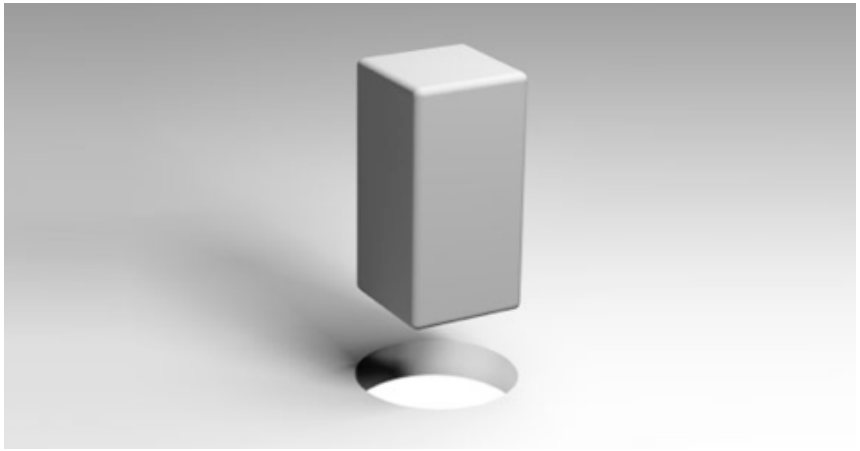
# Enterprise Architecture

Enterprise Architecture

# enterprise architecture context

Business Strategy and Operating Model

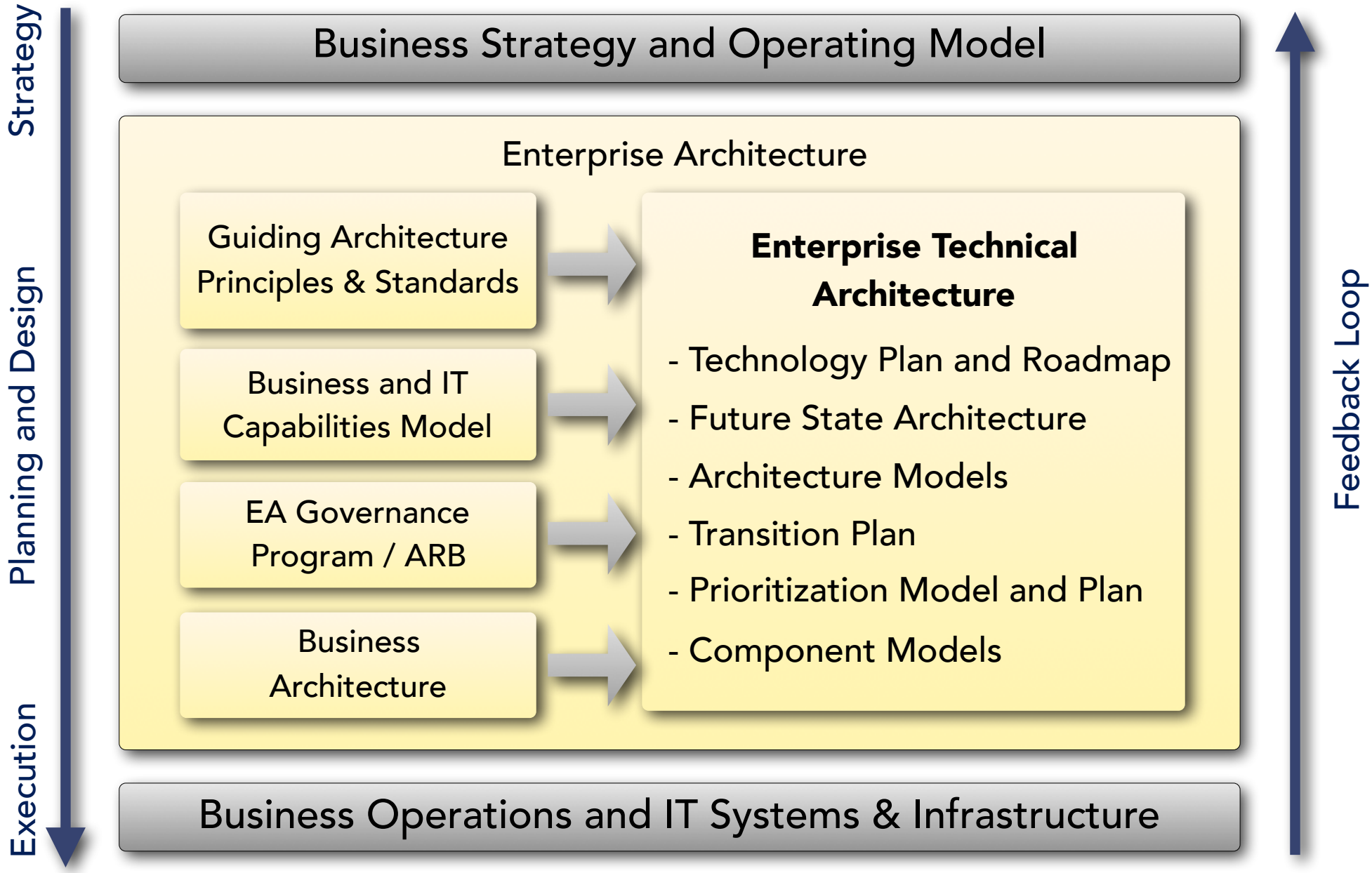
Business Needs



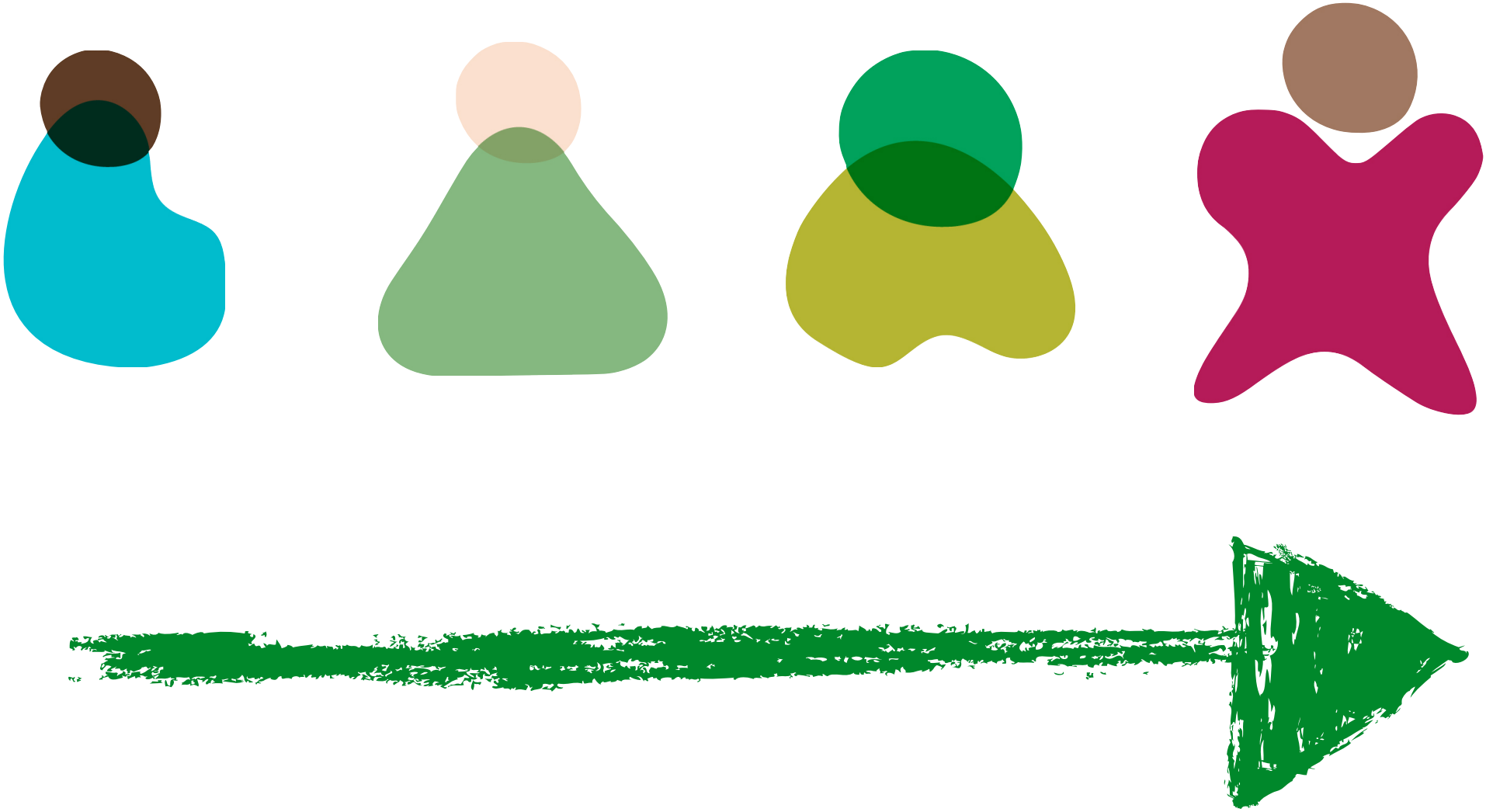
IT Capabilities

Business Operations and IT Systems & Infrastructure

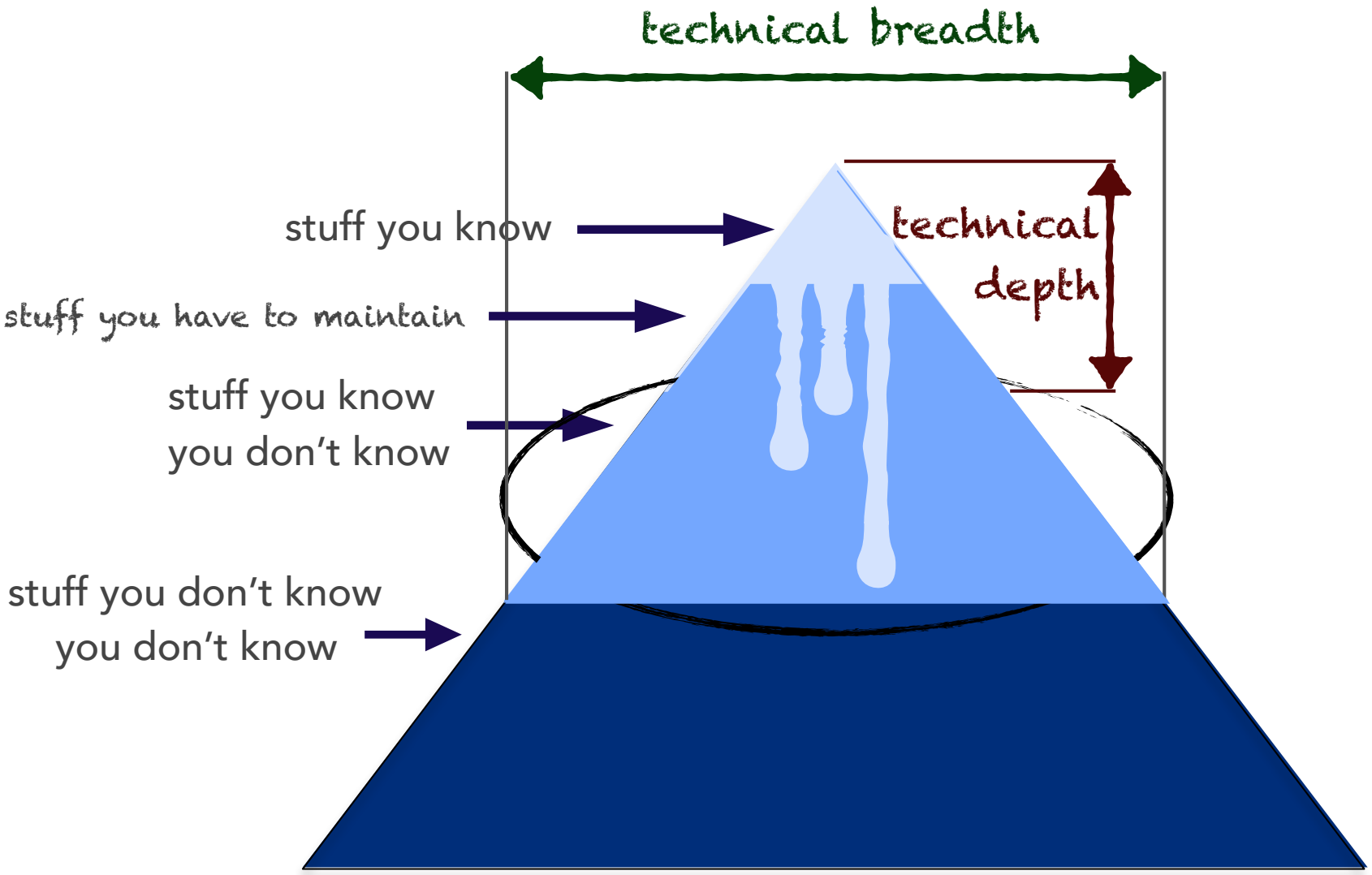
# enterprise architecture context

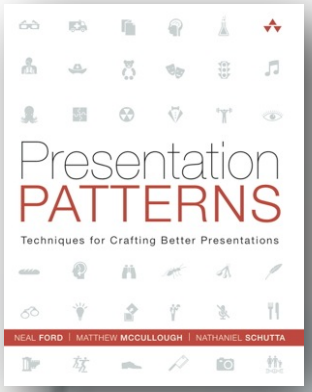


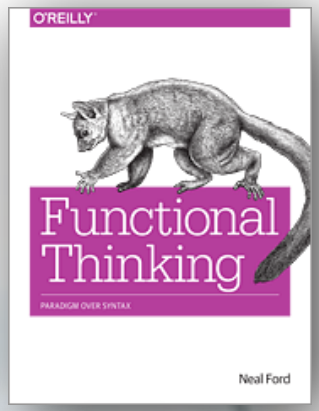
# from developer to architect



# the knowledge triangle





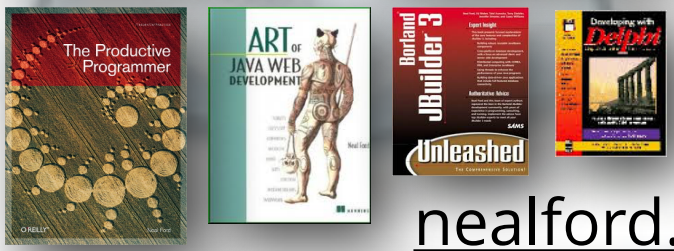

nealford.com

@neal4d

ThoughtWorks®

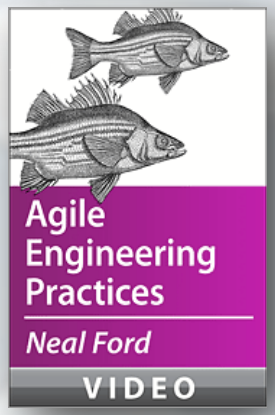
**NEAL FORD**

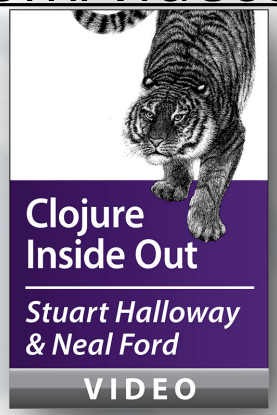
*Director / Software Architect / Meme Wrangler*

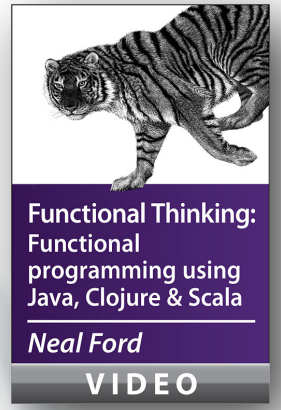



nealford.com/books

nealford.com/videos




O'REILLY®

**SOFTWARE ARCHITECTURE SERIES**

www.oreilly.com/software-architecture-video-training-series.html