Roma
20-23.03.2013

www.codemotionworld.com

Simon Brown

Software architecture for developers

Follow me on Twitter @simonbrown

I help software teams understand

software architecture, technical leadership and the balance with agility

(I code too)

Training

Book

Speaking

simon.brown@codingthearchitecture.com

@simonbrown on Twitter

Jersey

JERSEY FINANCE
VOICE OF THE INTERNATIONAL FINANCE CENTRE

DIGITAL JERSEY

Locate Jersey

What is software architecture?

What is architecture?

As a noun...

Structure

The definition of something in terms of its components and interactions

and

As a verb...

Vision

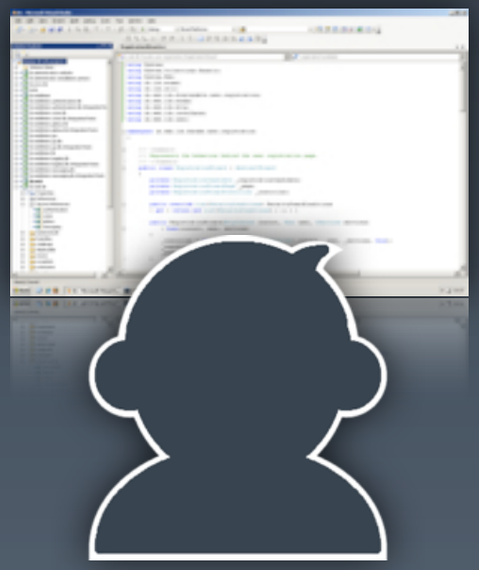
The process of architecting, making (significant) design decisions, etc

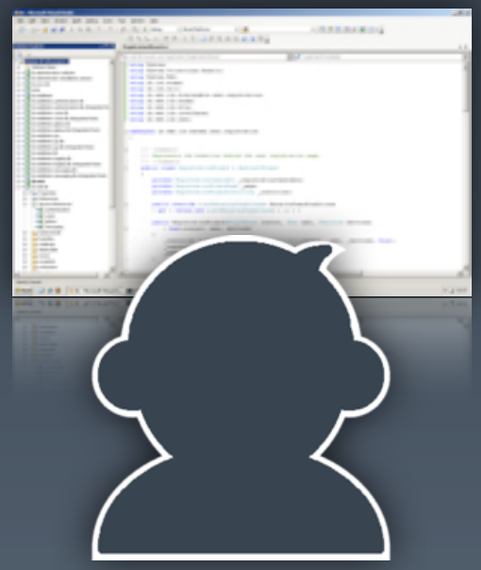
What is design?

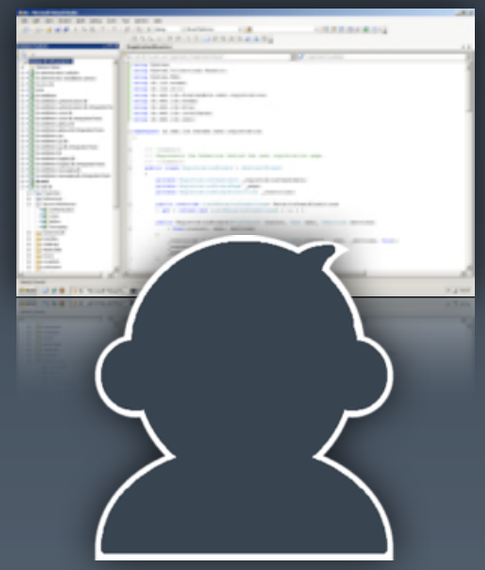
Architecture represents the
significant decisions,
where significance is measured
by **cost of change**.

Grady Booch
http://www.handbookofsoftwarearchitectu…log&part=2006

*Can you **refactor** it in an afternoon?*

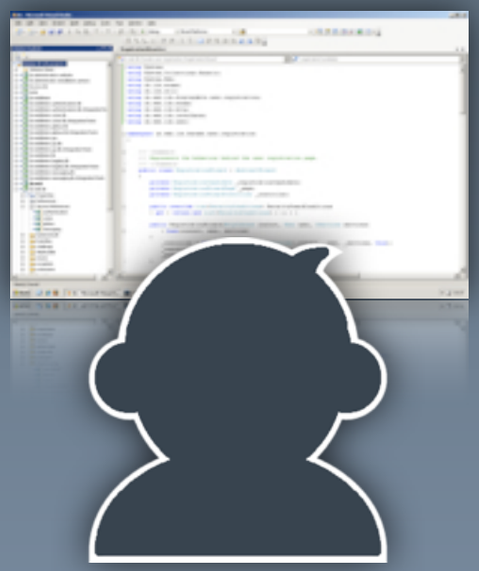


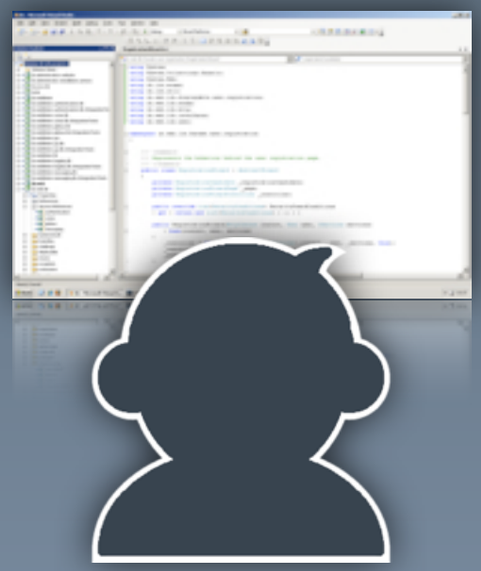


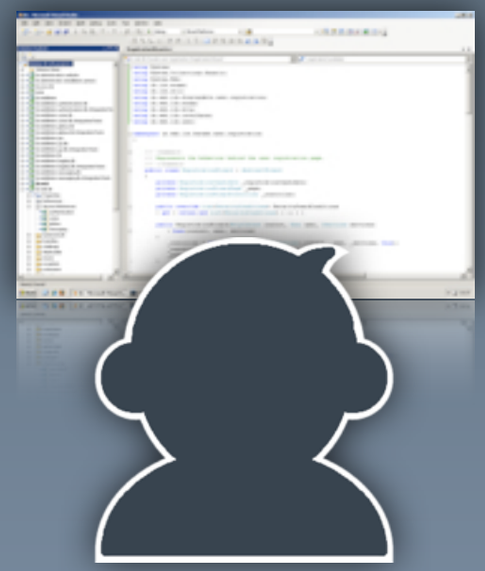


Chaos!

Does the team understand what they are building and how they are building it?



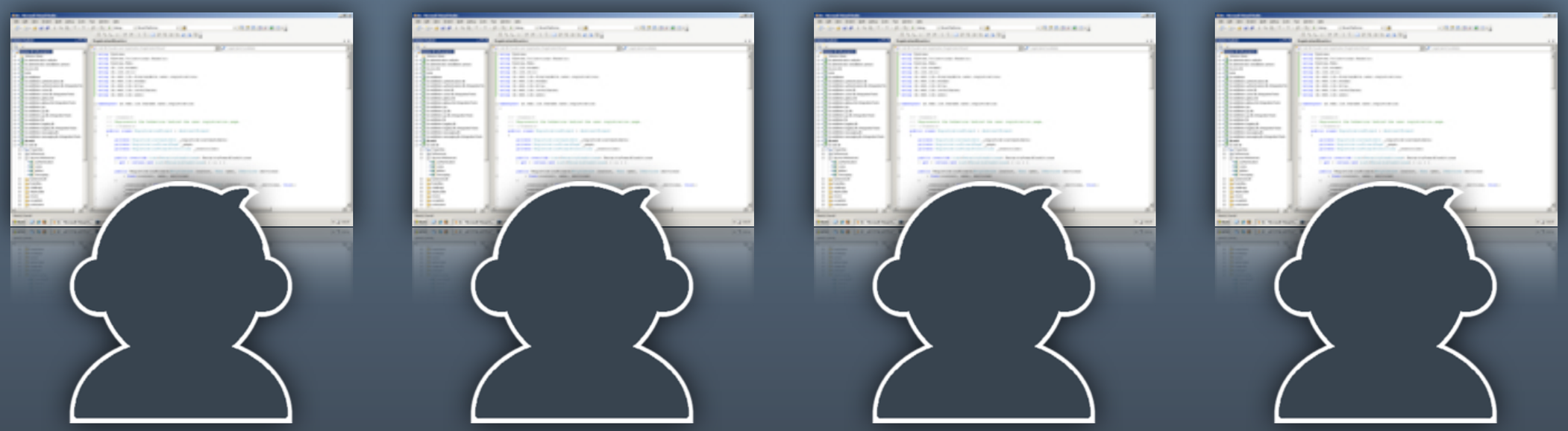


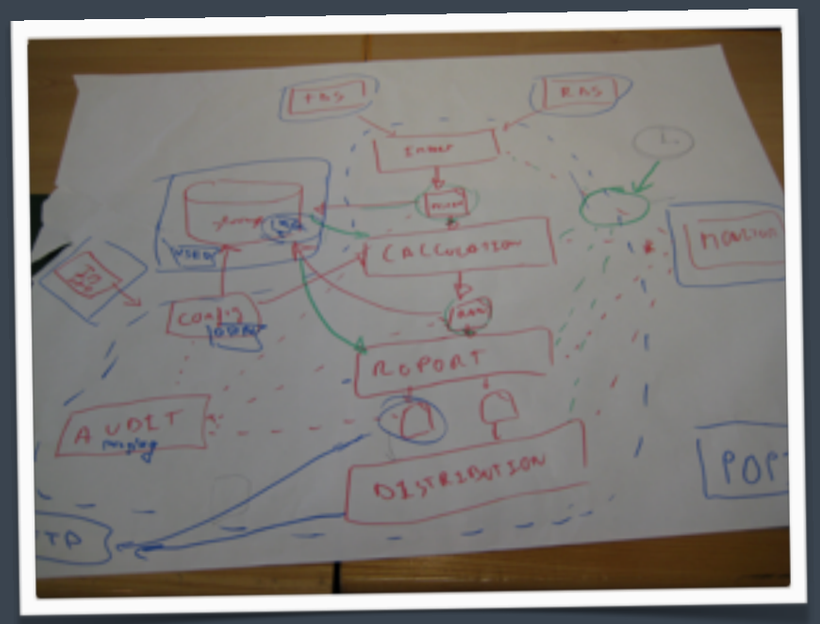



No defined structure,
inconsistent approaches,
big ball of mud,
spaghetti code, ...
STOP
Slow, insecure, unstable, unmaintainable,
hard to deploy, hard to change,
over time, over budget, ...

Shared vision of

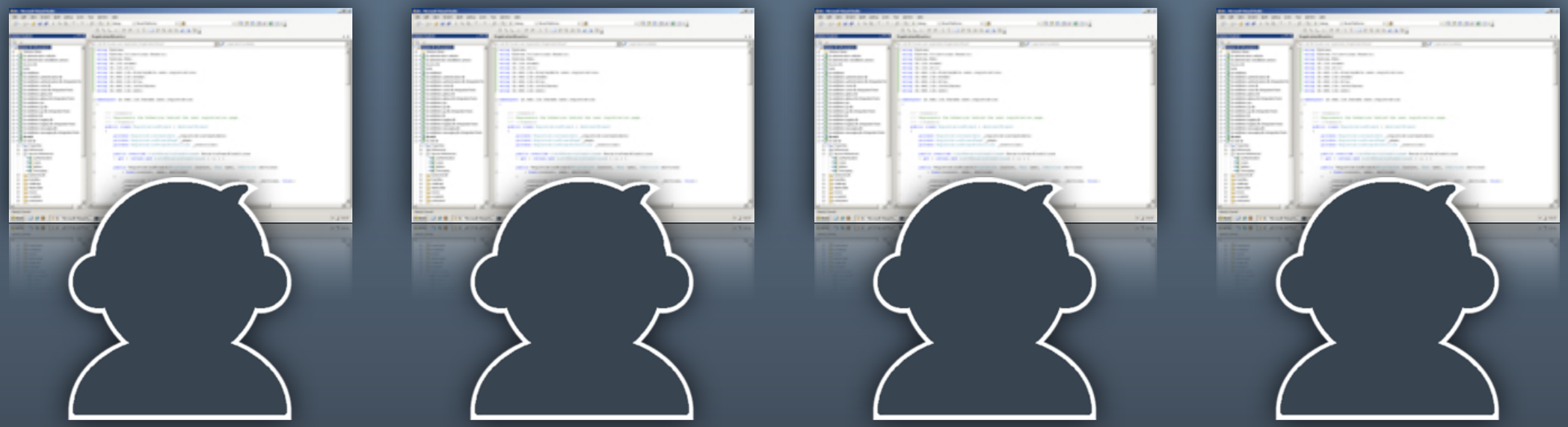
TL;DR





Shared vision of

WTF?!

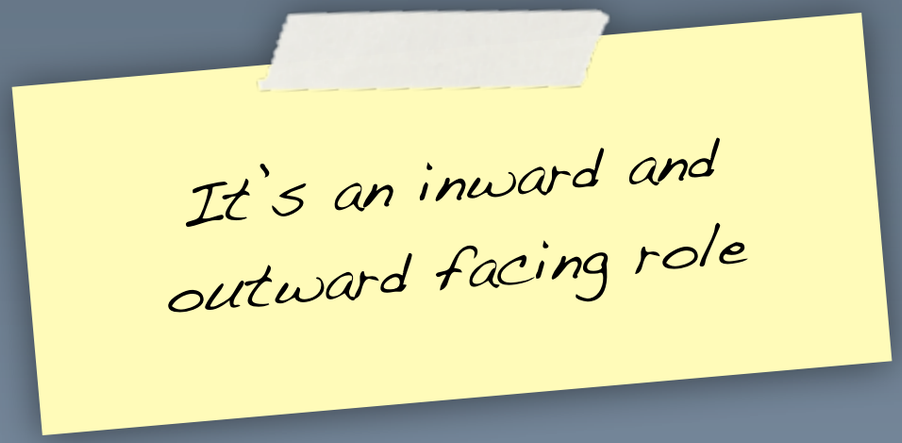


The software architecture role

The software architecture role is

different

to the lead developer role



It’s about the

big picture

Abstract

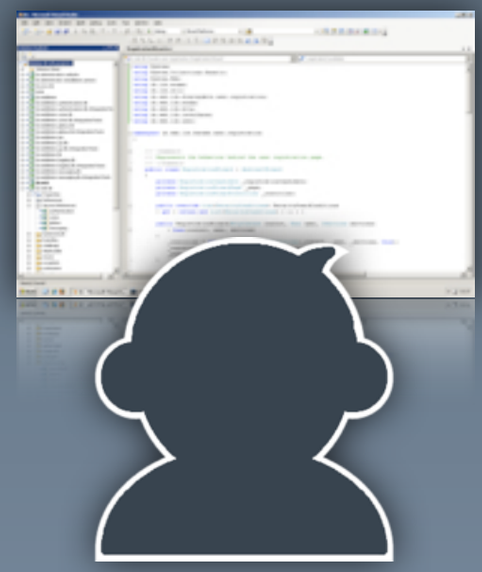
Specific

As software developers, the

code

is usually our main focus

Lines of code
Classes, functions
Design patterns
Unit tests
Refactoring



Abstract

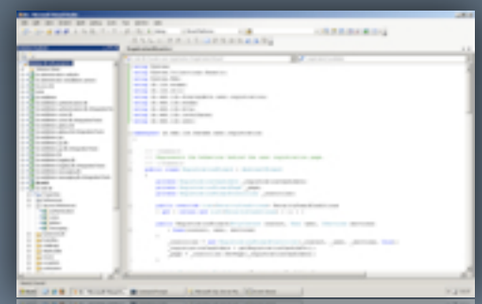
Specific

Lines of code
Classes, functions
Design patterns
Unit tests
Refactoring

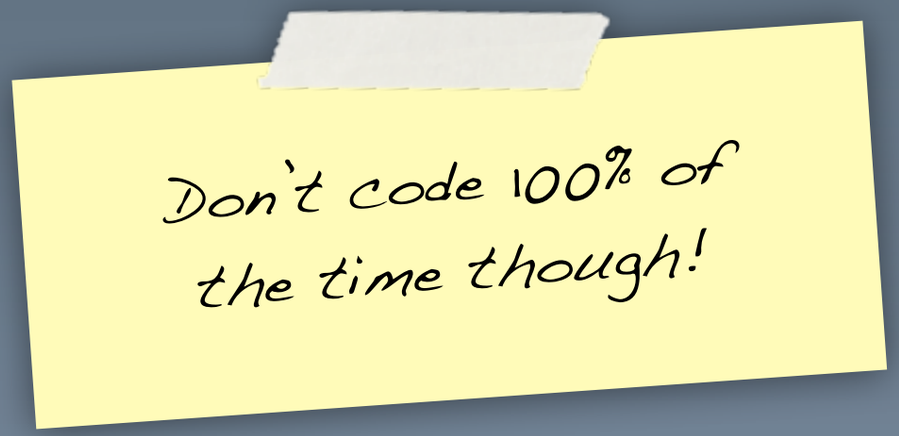
Sometimes you need to

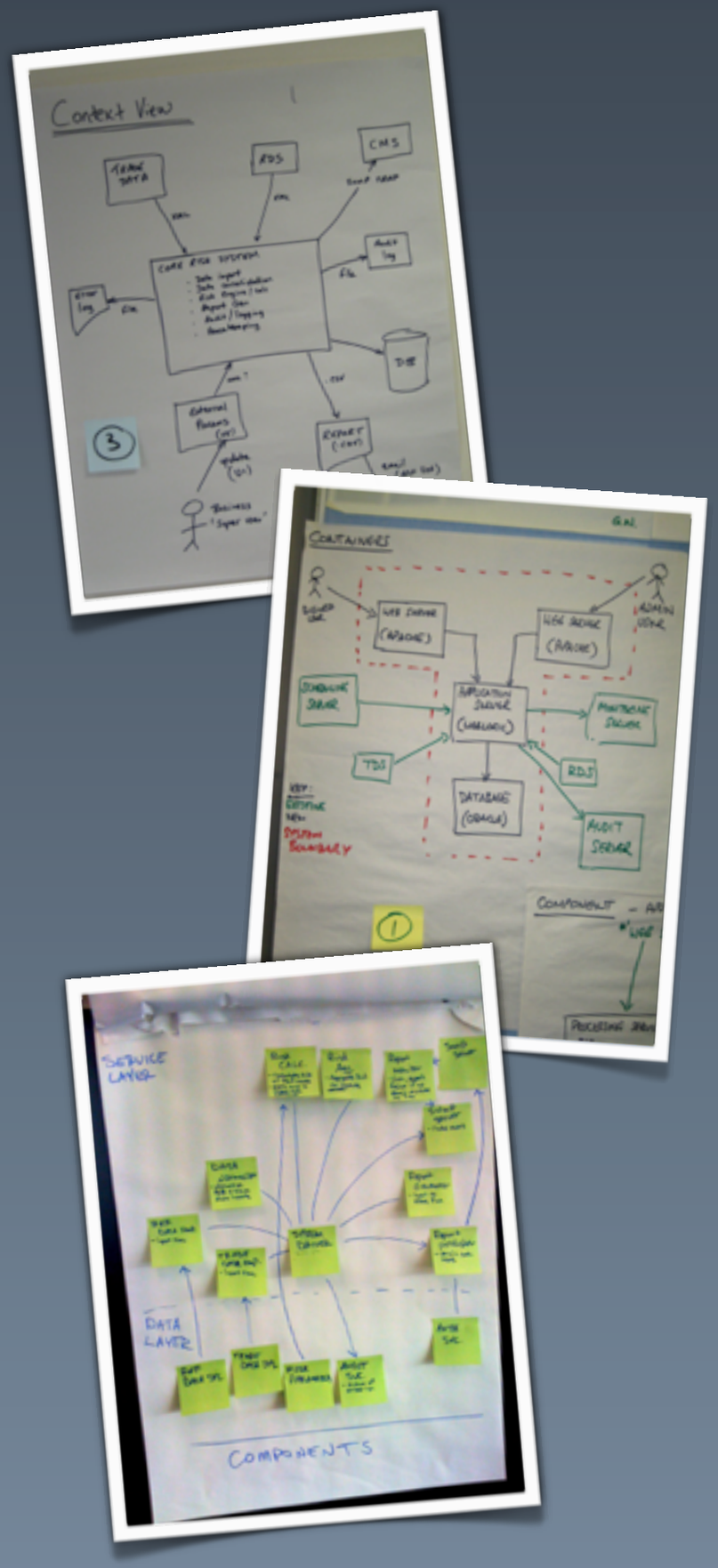
step back

from the IDE



The low-level detail is equally important





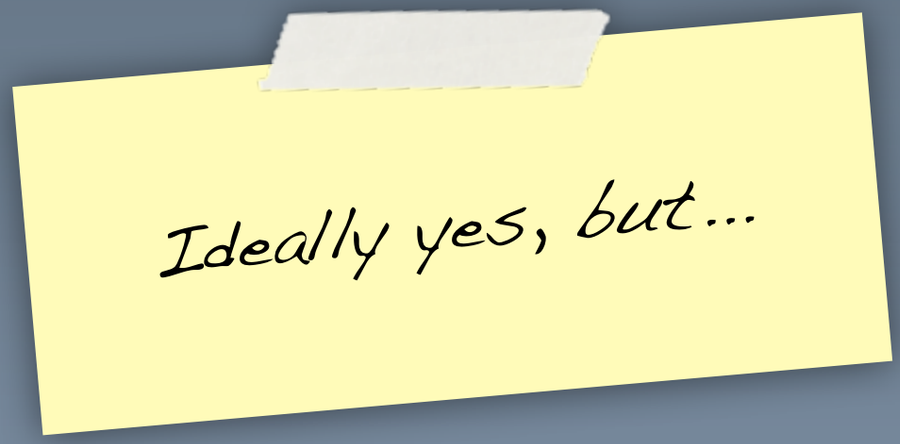
Would we code it that way?

All decisions involve a trade-off

Should software architects write

code

on software projects?



Software architects

must be

master builders

And coding is a great way
to retain this skill

Plus it reduces many of the
problems associated with
ivory tower architecture

Generalising

Specialist

Depth

Deep hands-on technology
skills and knowledge

Breadth

Broad knowledge of
patterns, designs,
approaches, technologies,
non-functional requirements,
different ways of working, etc
...
options and trade-offs

Every software development team needs a master builder

Software architecture is about

technical

and

soft skills

Software Architect

Leadership

Communication

Influencing

Negotiation

Collaboration

Coaching and Mentoring

Motivation

Facilitation

Political

The irresponsible architect

Cross-site scripting attacks possible; weak passwords allowed; HTTP sessions didn't timeout; ...

No non-functional testing (e.g. penetration testing or load testing); ...

Basic functionality errors; little or no quality assurance; rework required late in the project because of assumptions; ...

No documentation; ...

Even on "strategic platform" projects :-o

The software architecture role

Architectural Drivers
Understanding requirements and constraints

Technology Selection
Choosing and evaluating technology

Architecting
Designing software

Architecture Evaluation
Understanding that the architecture works

Coding
Involvement in the hands-on elements of software delivery

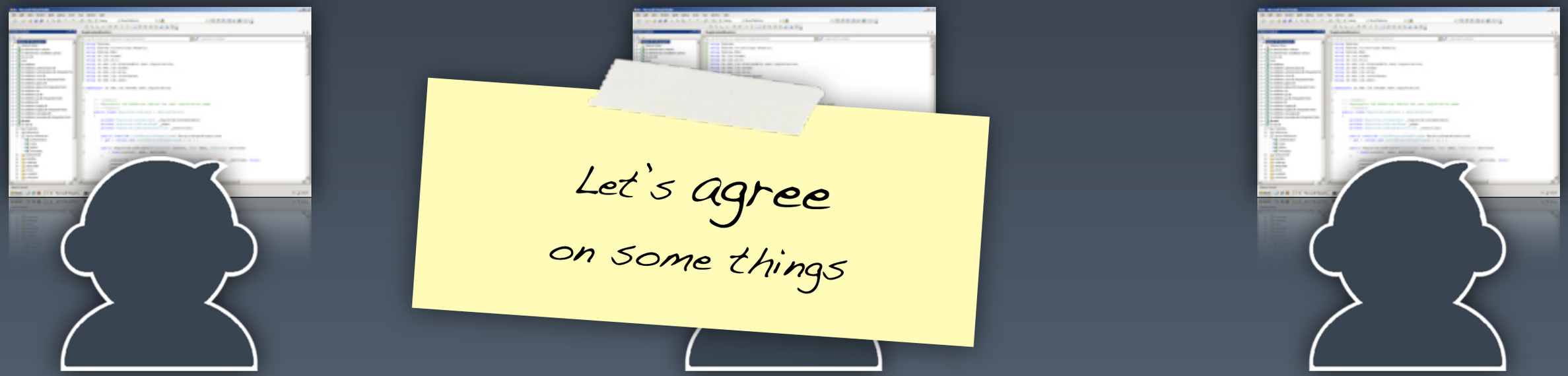
Architecture Evolution
Ownership of the architecture throughout the delivery

Quality Assurance
Introduction and adherence to standards and principles

Coaching and Mentoring
Guidance and assistance

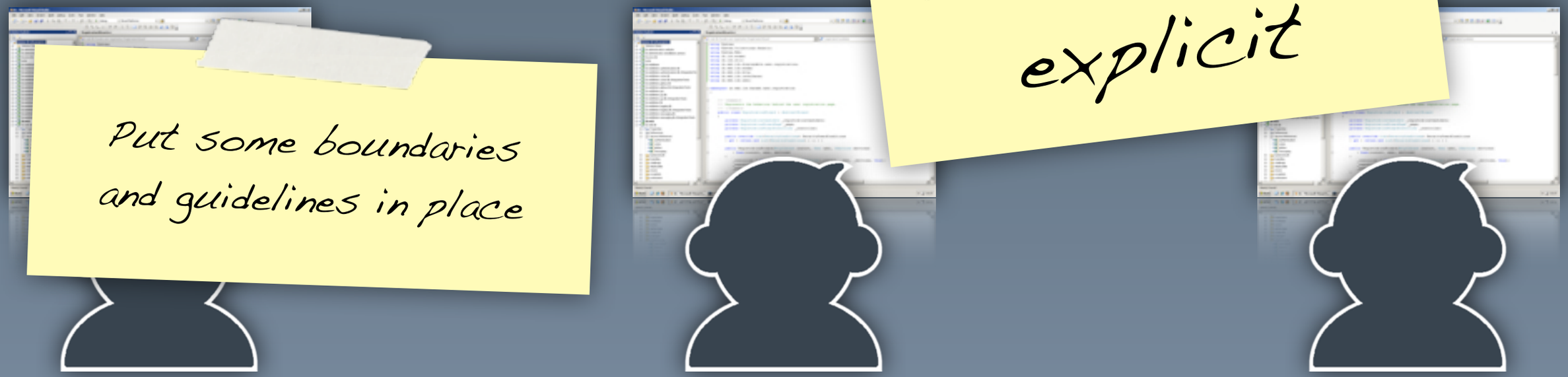
Software architecture introduces

control?



Chaos!

Does the team understand what they are buildi



Software Architect

Control

Guidelines, consistency,
discipline, rigour, boundaries,
...

Software Developers

Feedback

" I don't understand why..."
" How should we..."
" I don't like the way that..."

Architect

Sits in an ivory tower

Gap

Focusses on the low level detail

Developer

Developer

Developer

Developer

Developer

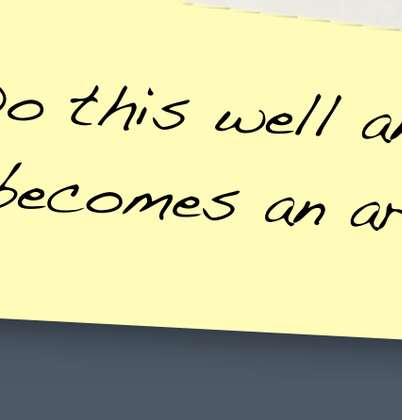
Do this well and everybody becomes an architect :-)

Software Architect

Collaborating and sharing

Reduced gap

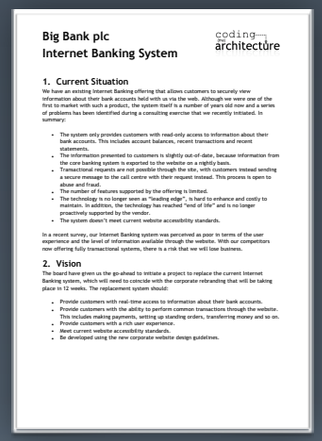
Architecturally aware



Software Developer

Avoid ivory towers by collaborating and being engaged

Designing software



Big Bank plc
Internet Banking System

coding architecture

1. Current Situation

We have an existing Internet Banking offering that allows customers to securely view information about their bank accounts held with us via the web. Although we were one of the first to market with such a product, the system itself is a number of years old now and a series of problems has been identified during a consulting exercise that we recently initiated. In summary:

- The system only provides customers with read-only access to information about their bank accounts. This includes account balances, recent transactions and recent statements.
- The information presented to customers is slightly out-of-date, because information from the core banking system is exported to the website on a nightly basis.
- Transactional requests are not possible through the site, with customers instead sending a secure message to the call centre with their request instead. This process is open to abuse and fraud.
- The number of features supported by the offering is limited.
- The technology is no longer seen as "leading edge", is hard to enhance and costly to maintain. In addition, the technology has reached "end of life" and is no longer proactively supported by the vendor.
- The system doesn't meet current website accessibility standards.

In a recent survey, our Internet Banking system was perceived as poor in terms of the user experience and the level of information available through the website. With our competitors now offering fully transactional systems, there is a risk that we will lose business.

2. Vision

The board have given us the go-ahead to initiate a project to replace the current Internet Banking system, which will need to coincide with the corporate rebranding that will be taking place in 12 weeks. The replacement system should:

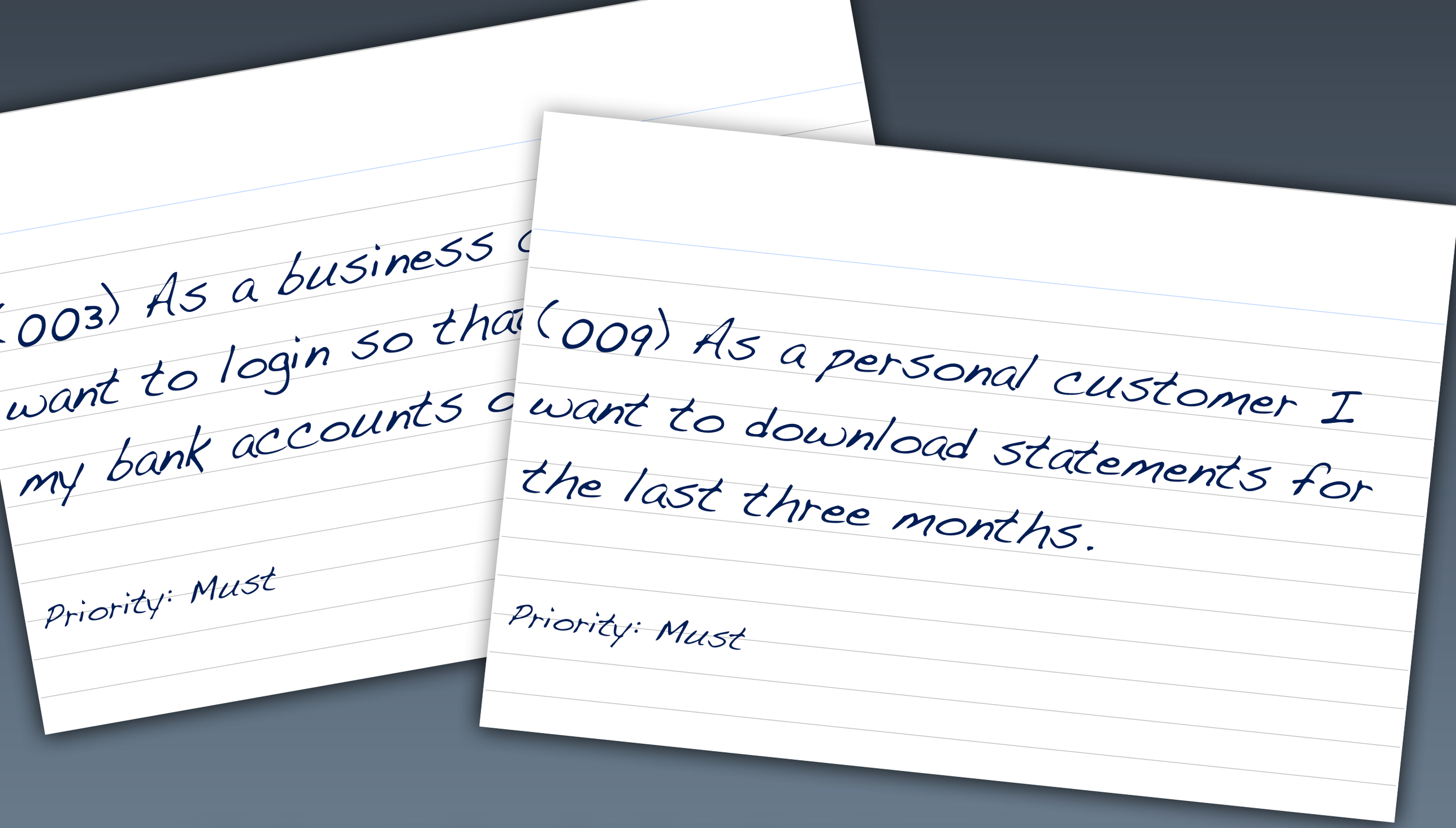
- Provide customers with real-time access to information about their bank accounts.
- Provide customers with the ability to perform common transactions through the website. This includes making payments, setting up standing orders, transferring money and so on.
- Provide customers with a rich user experience.
- Meet current website accessibility standards.
- Be developed using the new corporate website design guidelines.

Options

Functional & non-functional requirements

Constraints

Principles



Understanding the functional requirements is

obvious but forgotten

Quality Attributes

- [ ] Performance ✓
- [ ] Scalability ✓
- [ ] Availability ✓
- [ ] Security ✓
- [ ] Disaster Recovery
- [ ] Accessibility
- [ ] Monitoring ✓
- [ ] Management
- [ ] Audit ✓
- [ ] …

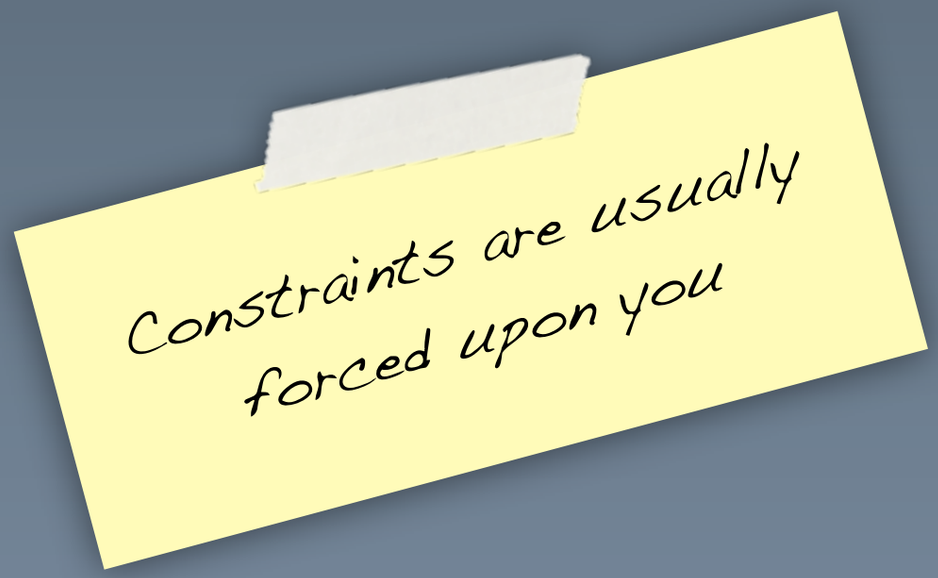
- [ ] Flexibility
- [ ] Extensibility
- [ ] Maintainability
- [ ] Interoperability ✓
- [ ] Legal
- [ ] Regulatory
- [ ] Compliance ✓
- [ ] i18n
- [ ] L10n
- [ ] …

Know which are important to YOU

Learn about and understand the (often complex) quality attributes in order to build sufficient foundations

Software lives in the real world,
and the real world has

constraints



Understand what the constraints are,
who imposed them,
why they are being imposed and
how they affect the architecture

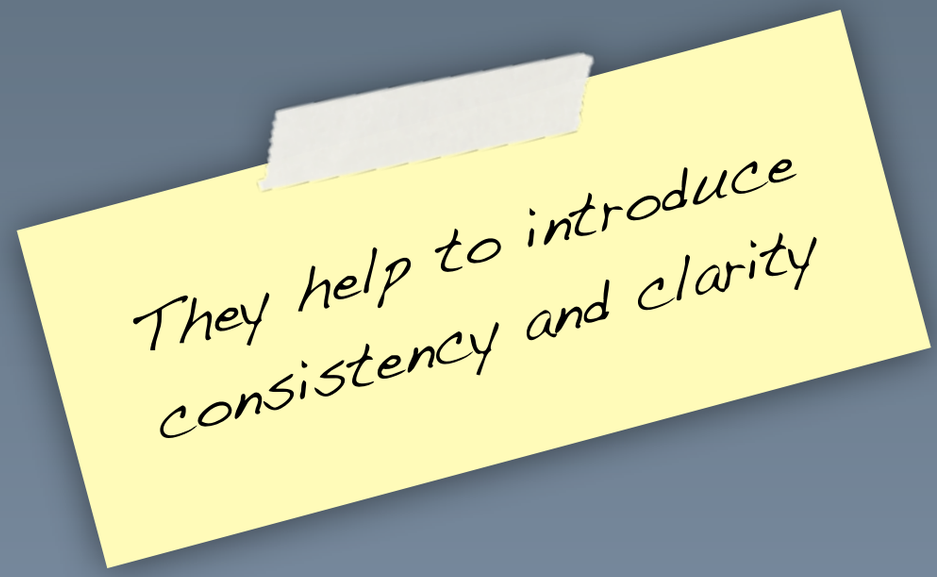
Given total freedom the work is likely to sprawl.

T.S.Eliot

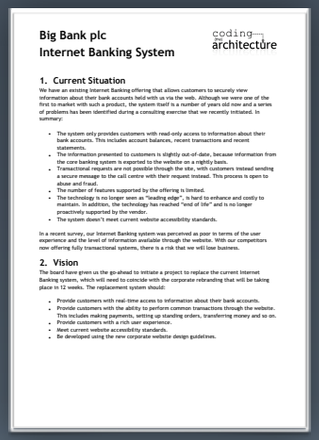
Principles

are the things

you want to adopt



Principles are good, but make sure they're realistic and don't have a negative impact



Big Bank plc
Internet Banking System

coding architecture

1. Current Situation

We have an existing Internet Banking offering that allows customers to securely view information about their bank accounts held with us via the web. Although we were one of the first to market with such a product, the system itself is a number of years old now and a series of problems has been identified during a consulting exercise that we recently initiated. In summary:

- The system only provides customers with read-only access to information about their bank accounts. This includes account balances, recent transactions and recent statements.
- The information presented to customers is slightly out-of-date, because information from the core banking system is exported to the website on a nightly basis.
- Transactional requests are not possible through the site, with customers instead sending a secure message to the call centre with their request instead. This process is open to abuse and fraud.
- The number of features supported by the offering is limited.
- The technology is no longer seen as "leading edge", is hard to enhance and costly to maintain. In addition, the technology has reached "end of life" and is no longer proactively supported by the vendor.
- The system doesn't meet current website accessibility standards.

In a recent survey, our Internet Banking system was perceived as poor in terms of the user experience and the level of information available through the website. With our competitors now offering fully transactional systems, there is a risk that we will lose business.

2. Vision

The board have given us the go-ahead to initiate a project to replace the current Internet Banking system, which will need to coincide with the corporate rebranding that will be taking place in 12 weeks. The replacement system should:

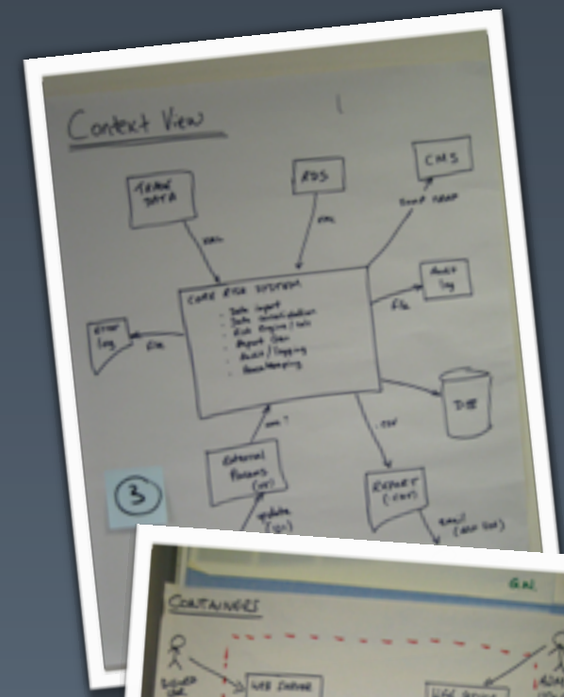
- Provide customers with real-time access to information about their bank accounts.
- Provide customers with the ability to perform common transactions through the website. This includes making payments, setting up standing orders, transferring money and so on.
- Provide customers with a rich user experience.
- Meet current website accessibility standards.
- Be developed using the new corporate website design guidelines.

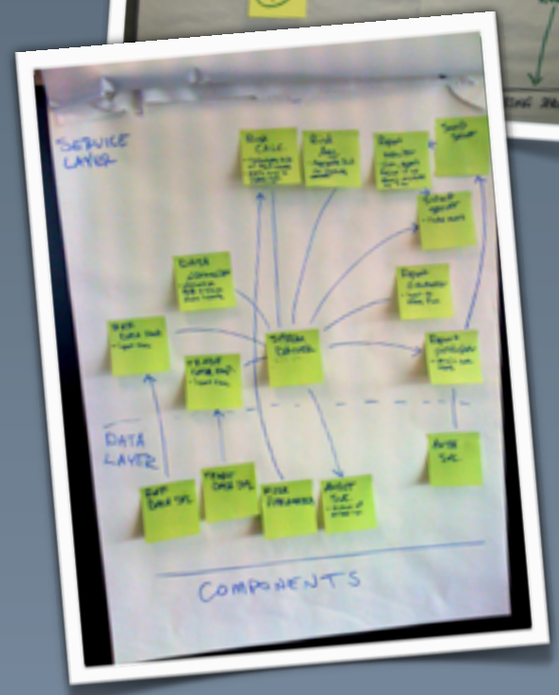
Options

Functional & non-functional requirements

Constraints

Principles

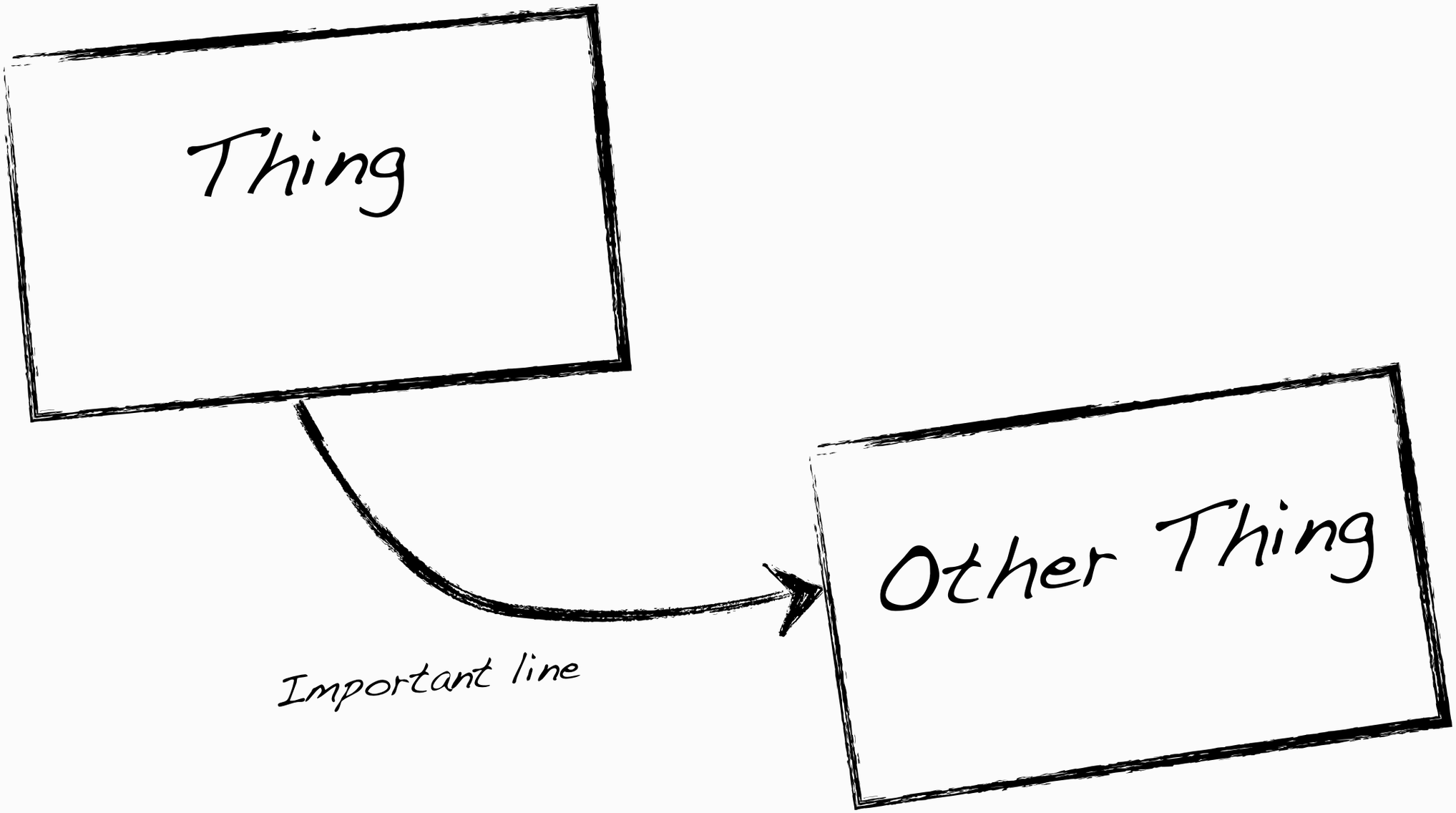




Start analysing or start coding?

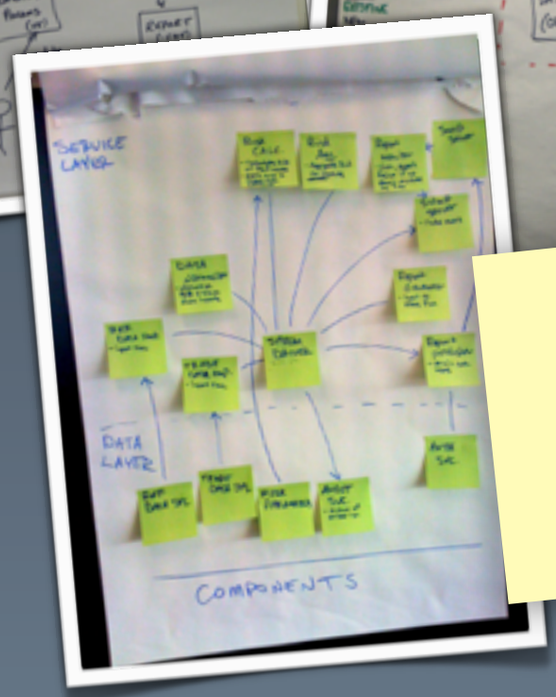
"Analysis paralysis" &
"refactor distractor"
are both bad

Boxes & lines

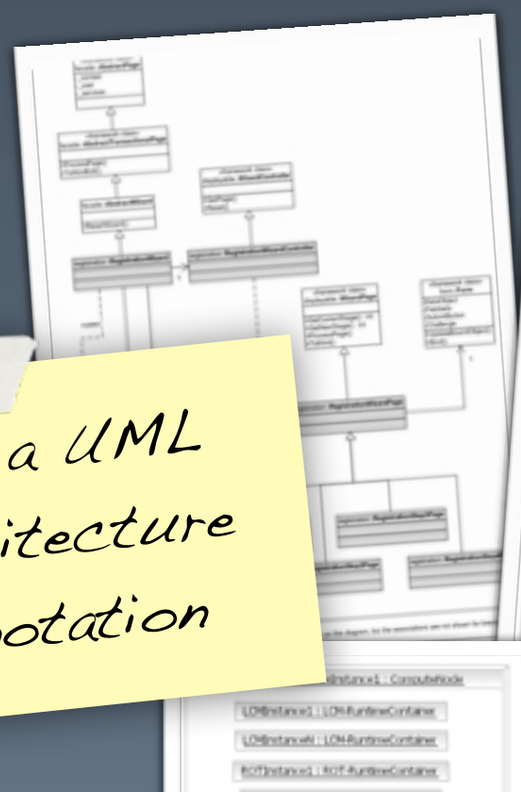


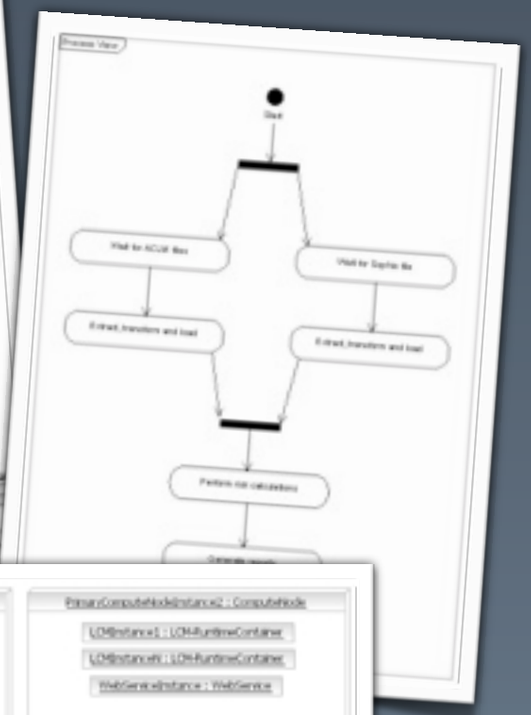
Visualising software

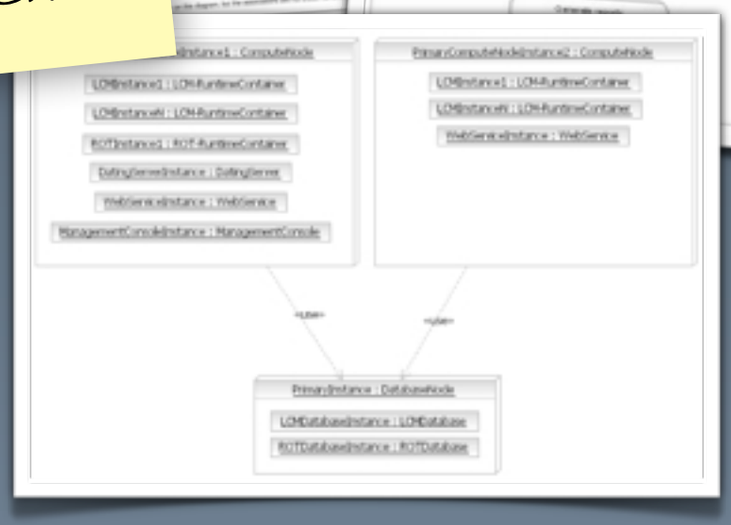
UML tool?



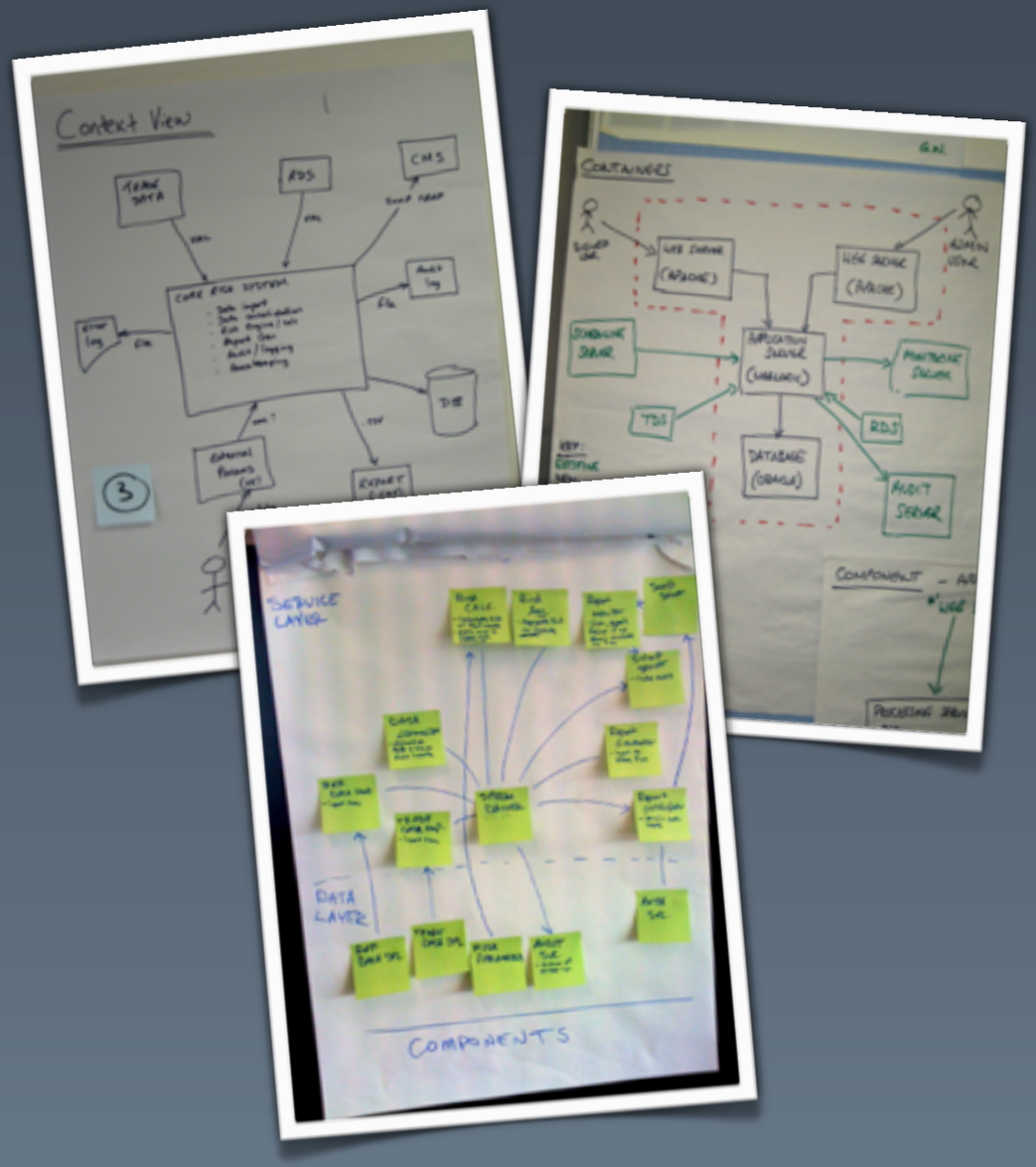
You don't need a UML tool to do architecture but agree on notation







Whiteboard or flip chart?

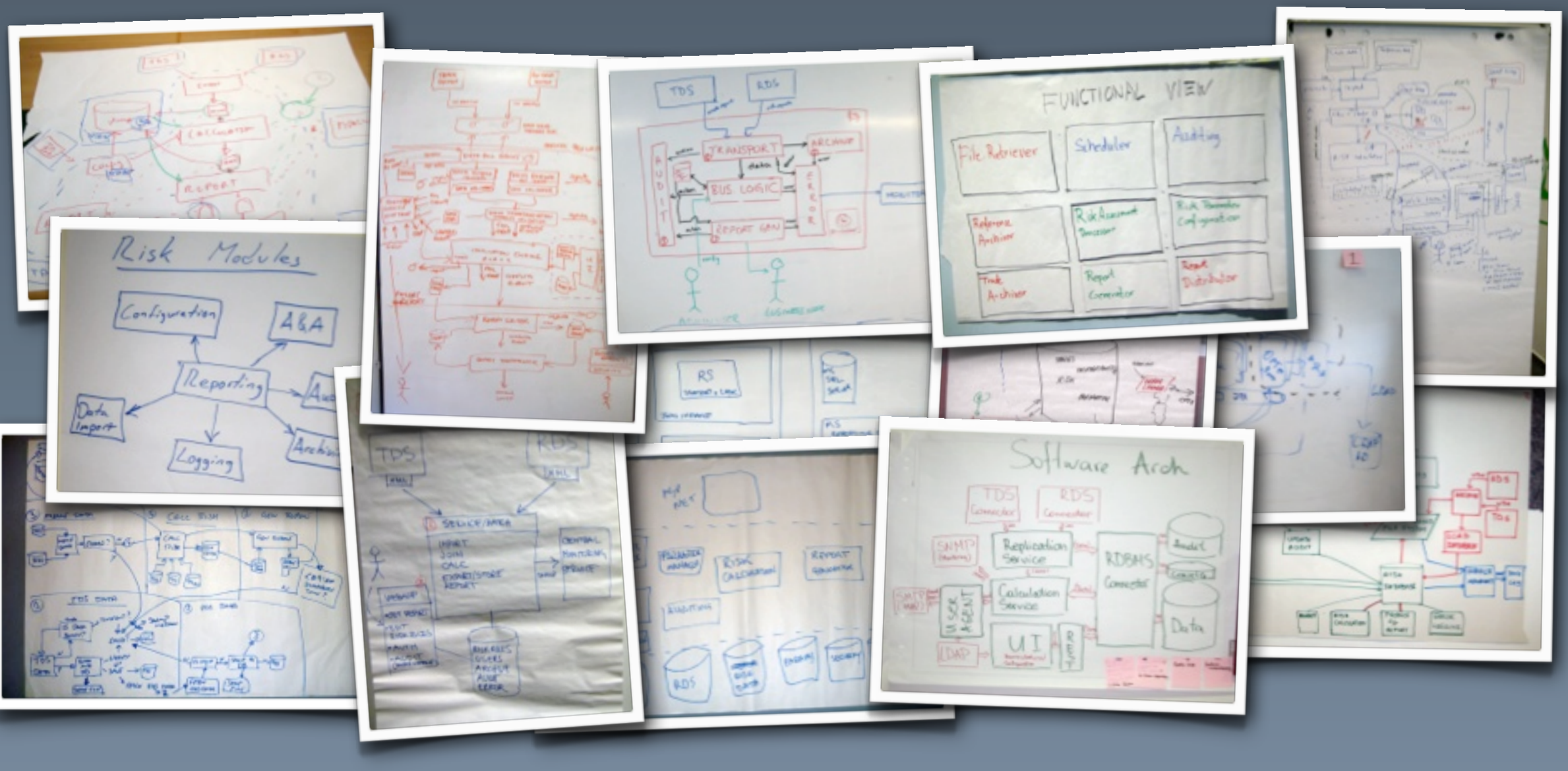


Collaborative design

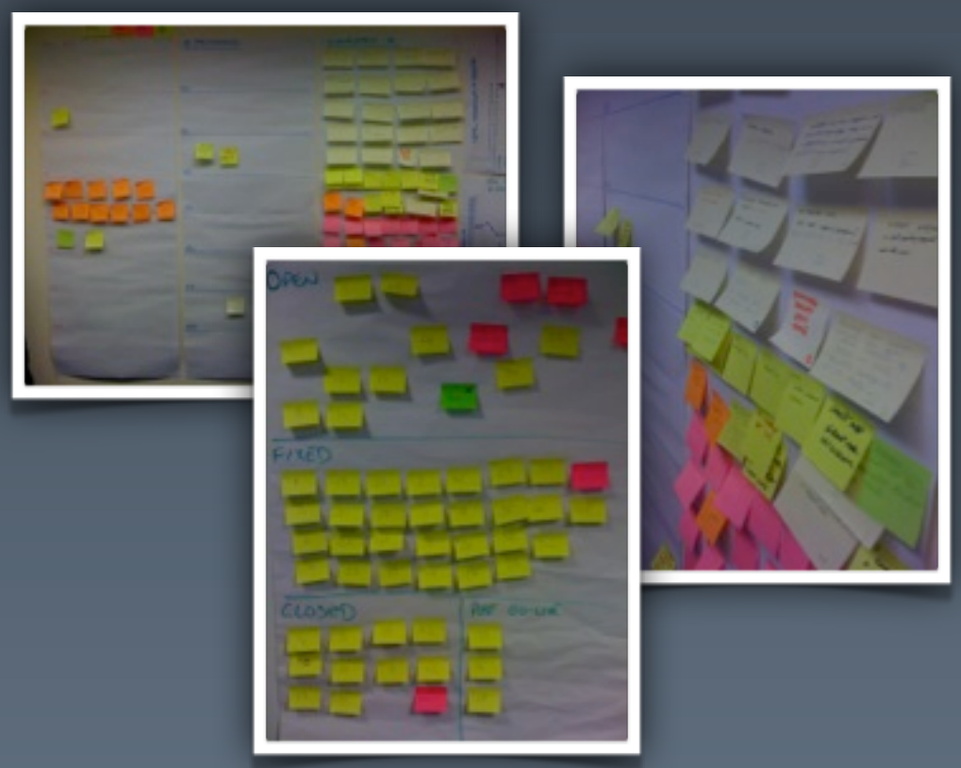
(e.g. pair architecting)

NoUML

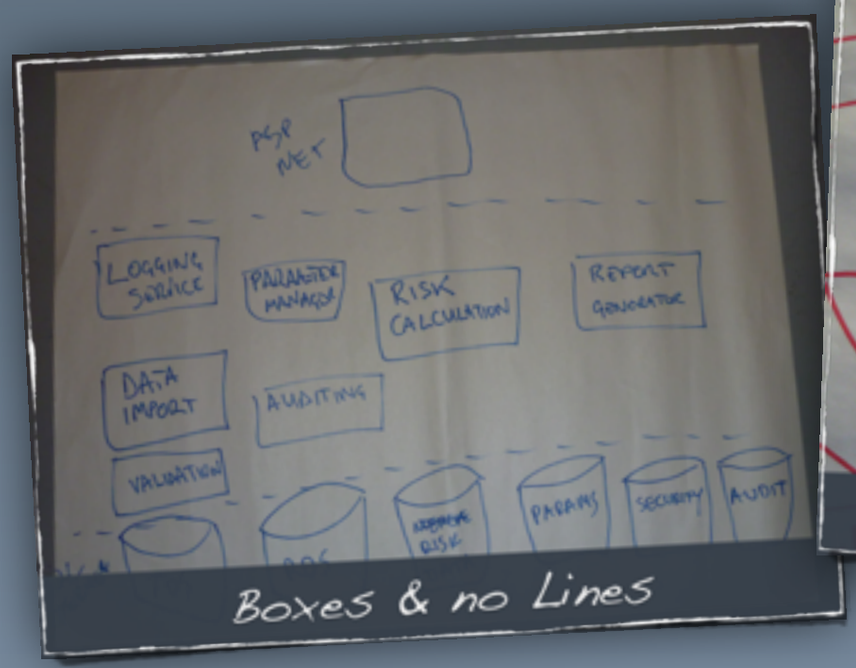
diagrams?



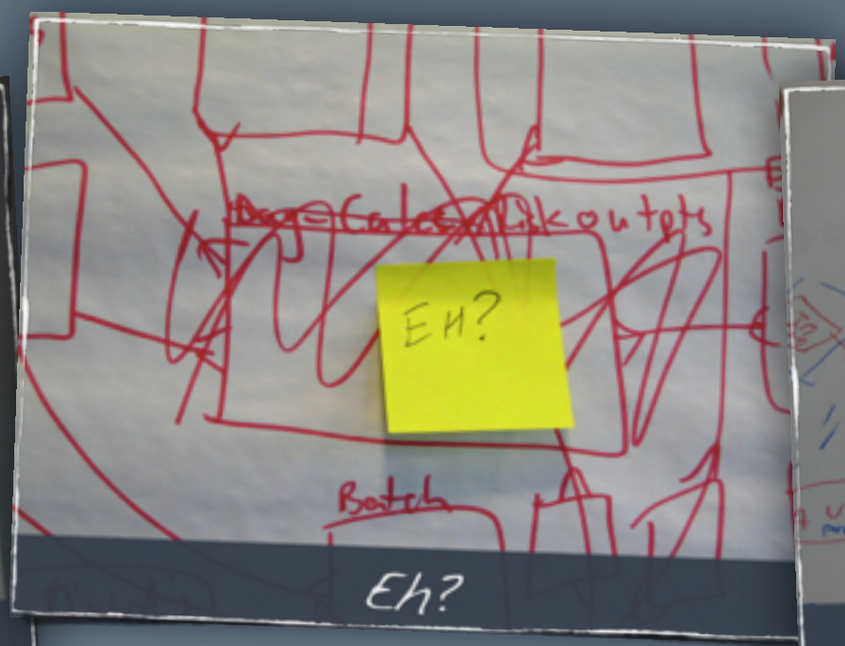
We can visualise our process…



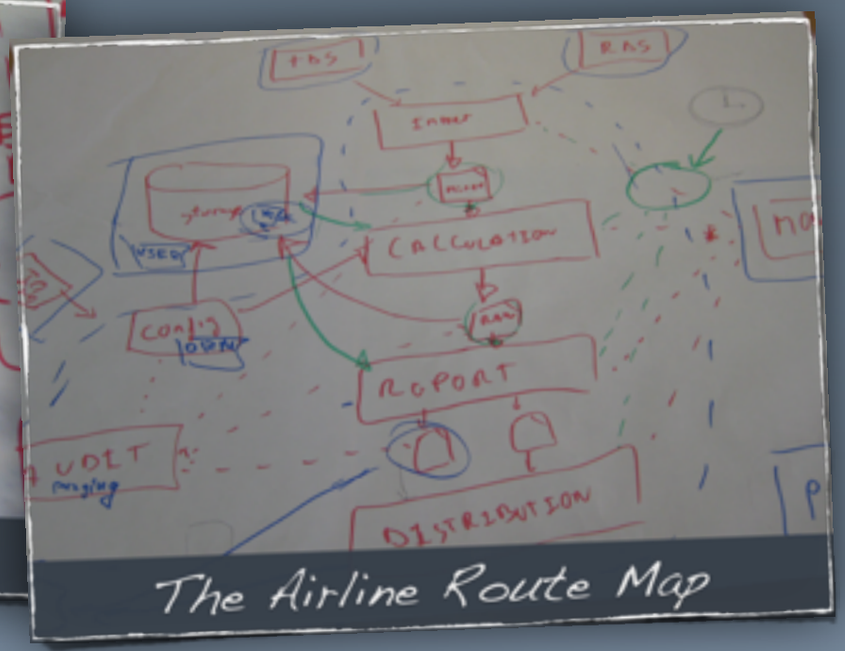
…but not our software!



Boxes & no Lines



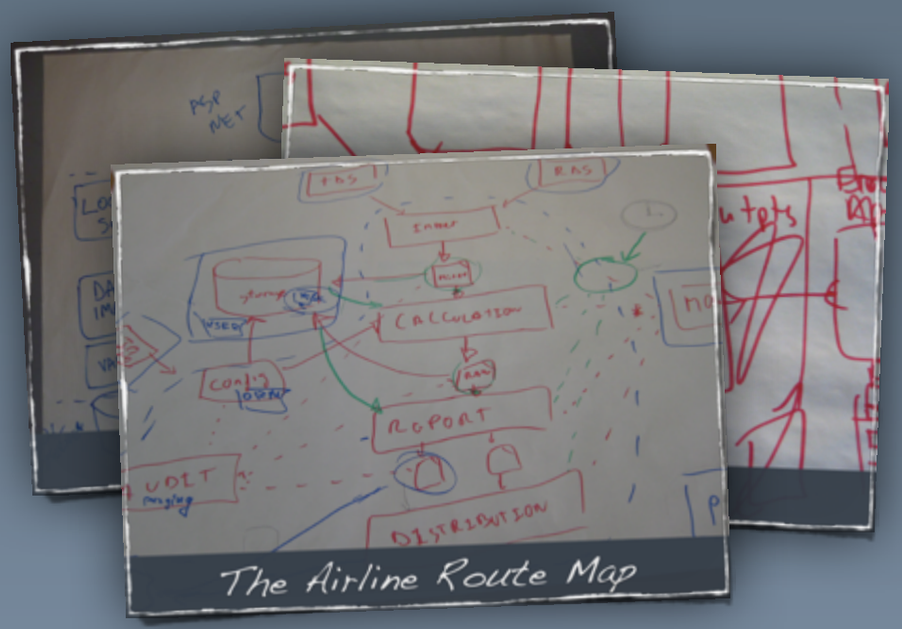
Eh?

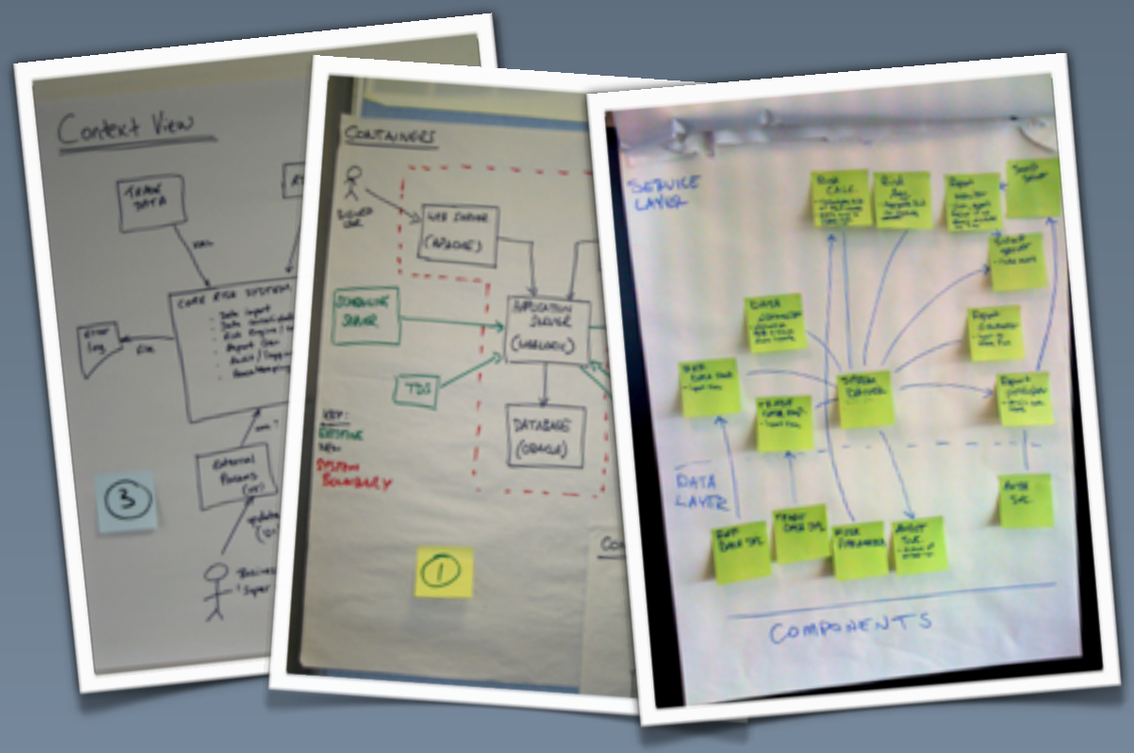


The Airline Route Map

Moving fast (agility) requires

good communication





System

Container

Container

Component

Class

Class

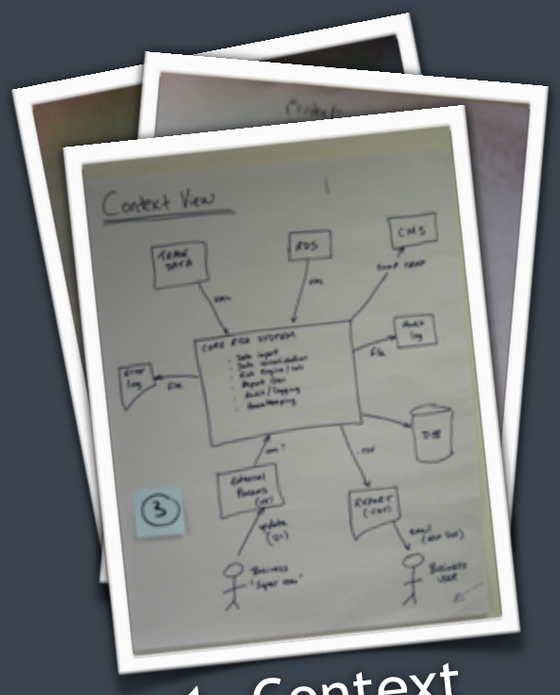
Class

Class

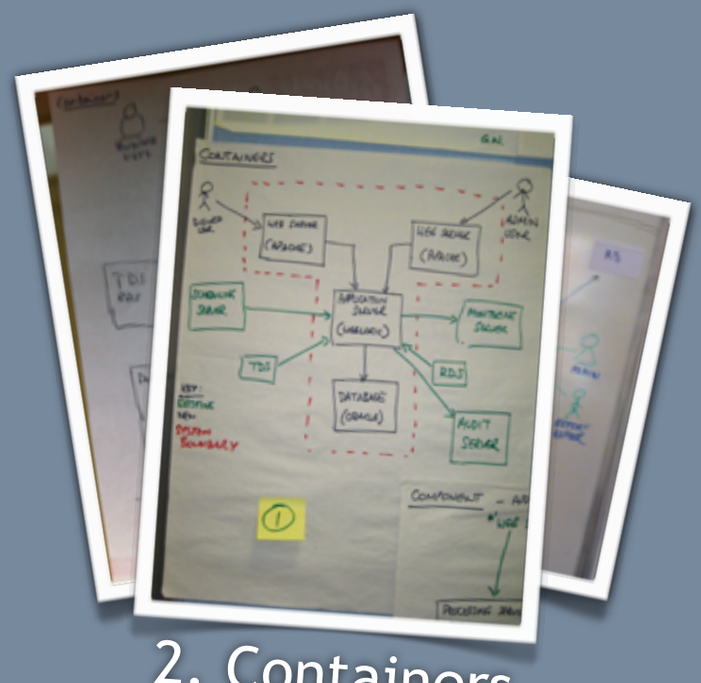
Component

Component

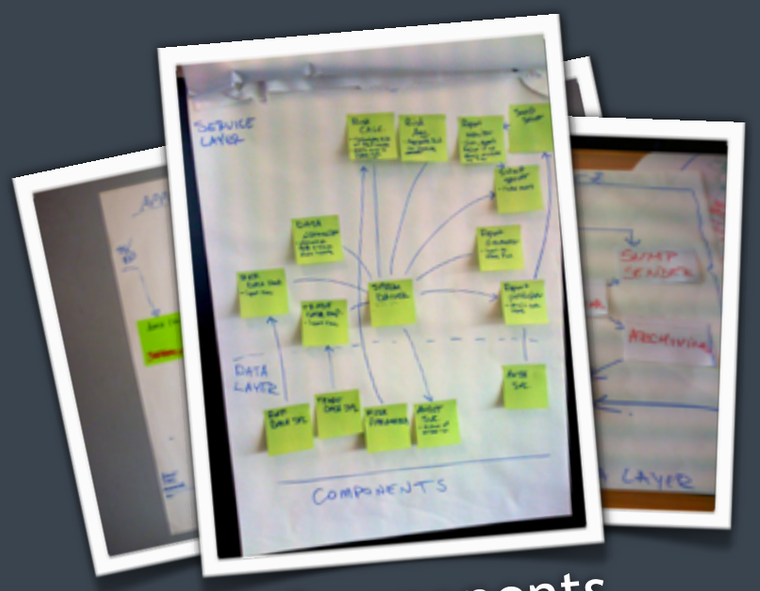
Container



1. Context



2. Containers



3. Components

C4

- Context
- Containers
- Components
- Classes

This only covers
the static structure
(runtime, infrastructure,
deployment, etc are also important)

… and, optionally,
4. Classes

Thinking inside the box

This isn't about creating a standard

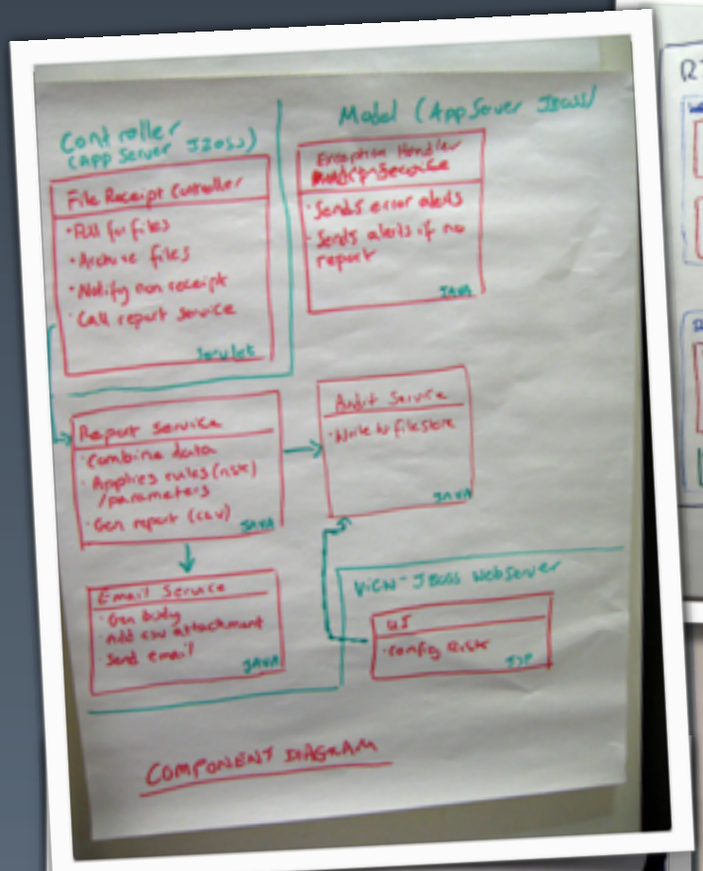
It's about providing you some organisational ideas

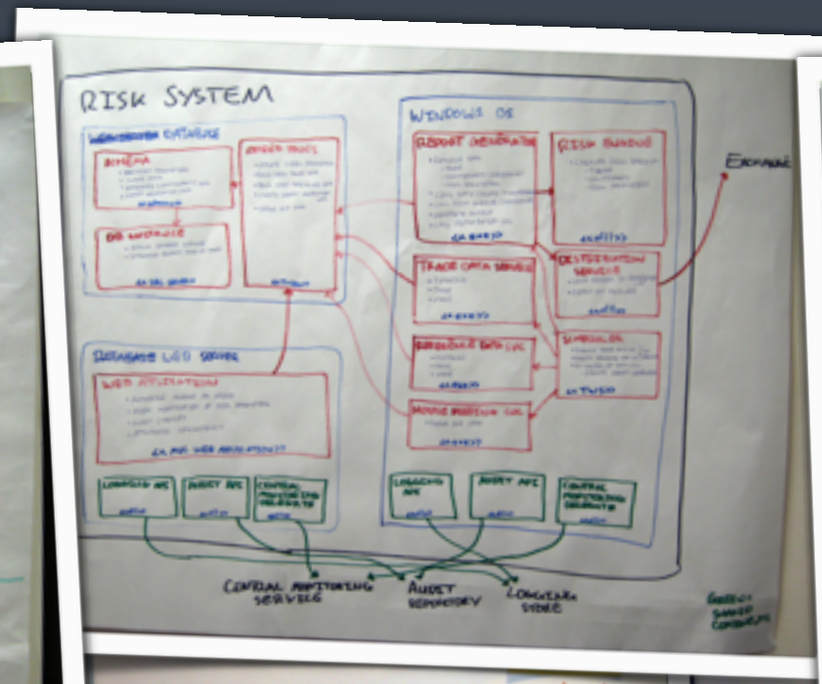
Context

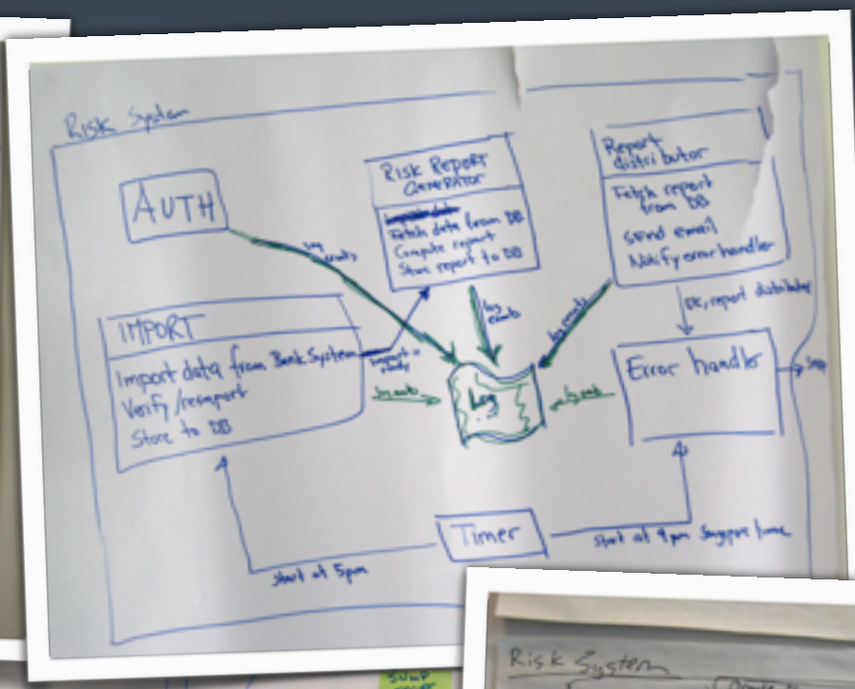
- What are we building?
- Who is using it? (users, actors, roles, personas, etc)
- How does it fit into the existing IT environment?

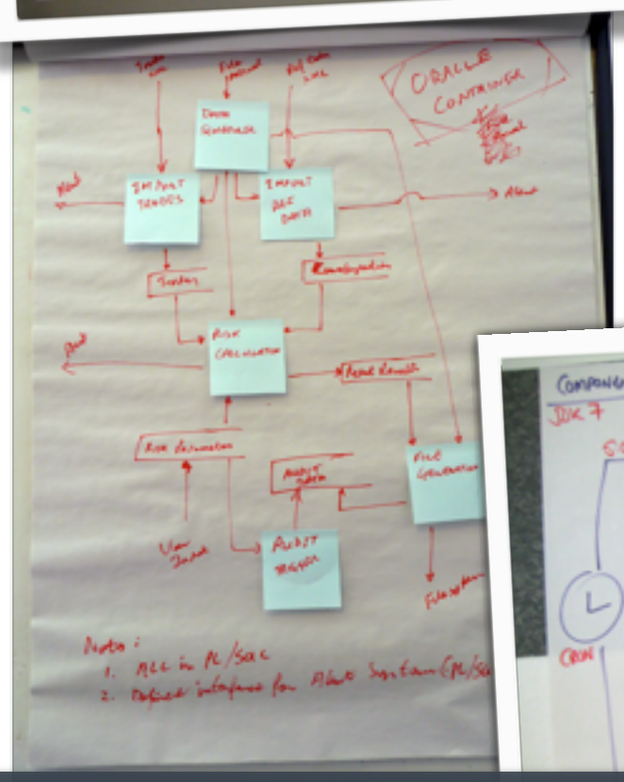
Containers

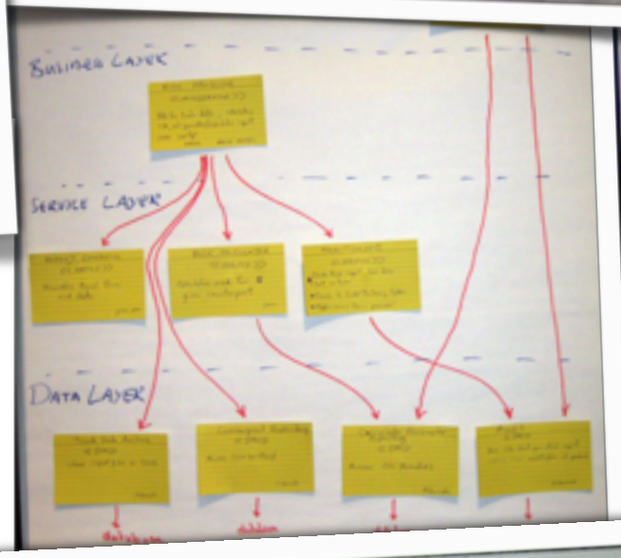
- What are the high-level technology decisions?
- How do containers communicate with one another?
- As a developer, where do I need to write code?

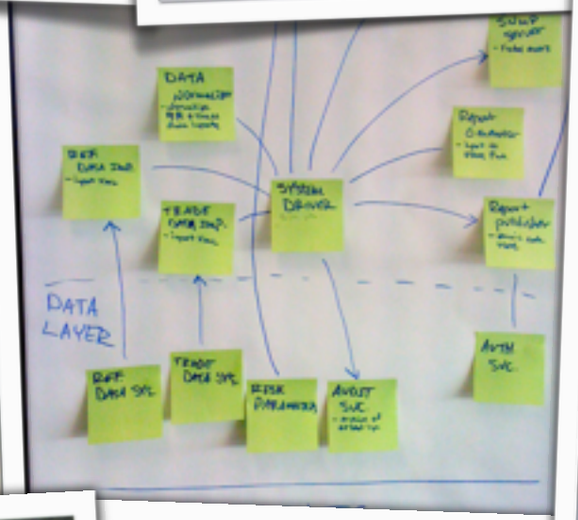


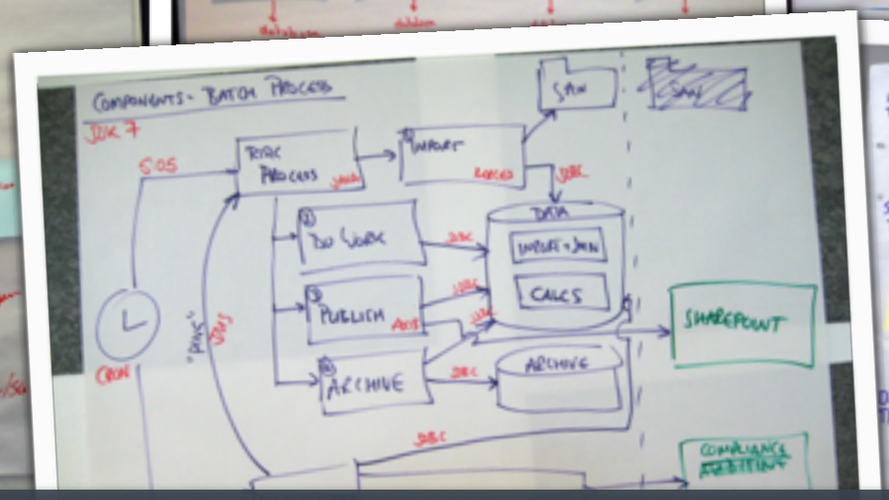


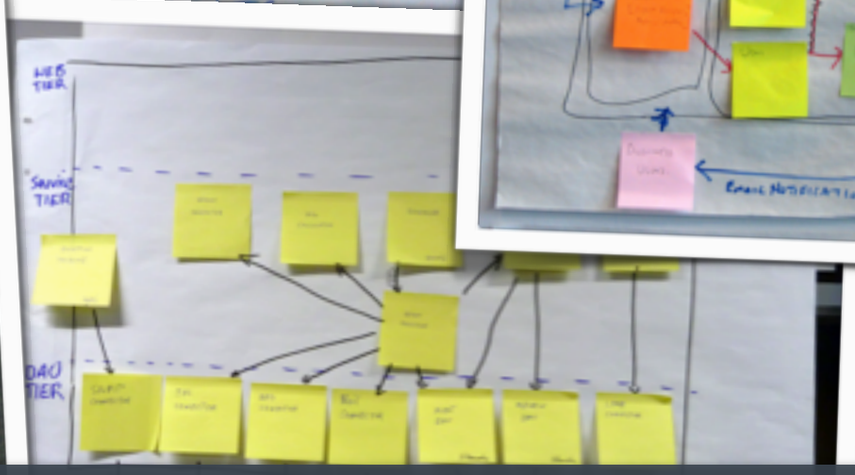












Components

- What components/services is the system made up of?
- Is it clear how the system works at a high-level?
- Do all components have a home (a container)?

Titles

Short and meaningful, numbered if diagram order is important

Lines

Make line style and arrows explicit, add annotations to lines to provide additional information

Layout

Sticky notes and index cards make a great substitute for drawn boxes, especially early on

Labels

Be wary of using acronyms

Colour

Ensure that colour coding is made explicit

Orientation

Users at the top and database at the bottom? Or perhaps “upside-down”?

Shapes

Don’t assume that people will understand what different shapes are being used for

Borders

Use borders to provide emphasis or group related items, but ensure people know why

Keys

Explain shapes, lines, colours, borders, acronyms, etc

Responsibilities

Adding responsibilities to boxes can provide a nice “at a glance” view (Miller’s Law; 7±2)

Some tips for *effective sketches*

Effective sketches

are an excellent way to

communicate

software architecture

During the design process
and retrospectively

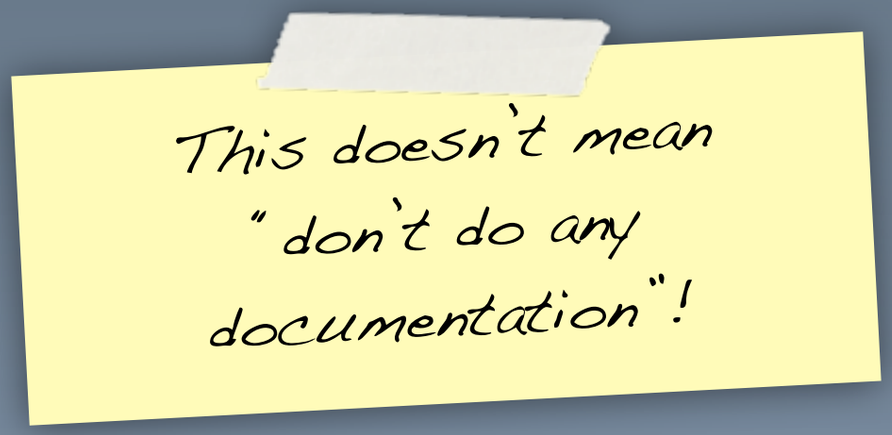
Documenting software

Working software

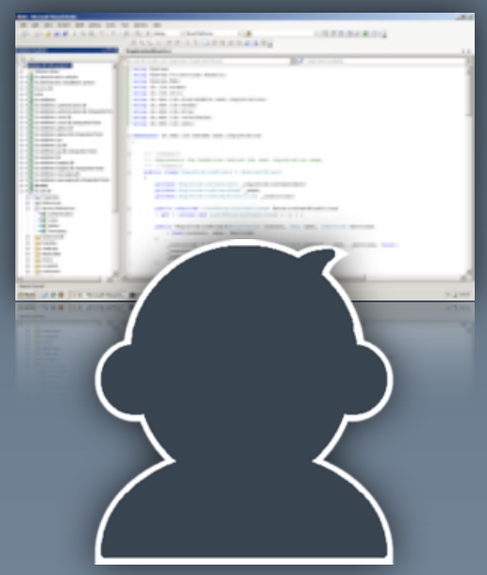
over

comprehensive documentation

Manifesto for Agile Software Development, 2001



The code doesn't tell the *whole* story, but it does *a* story



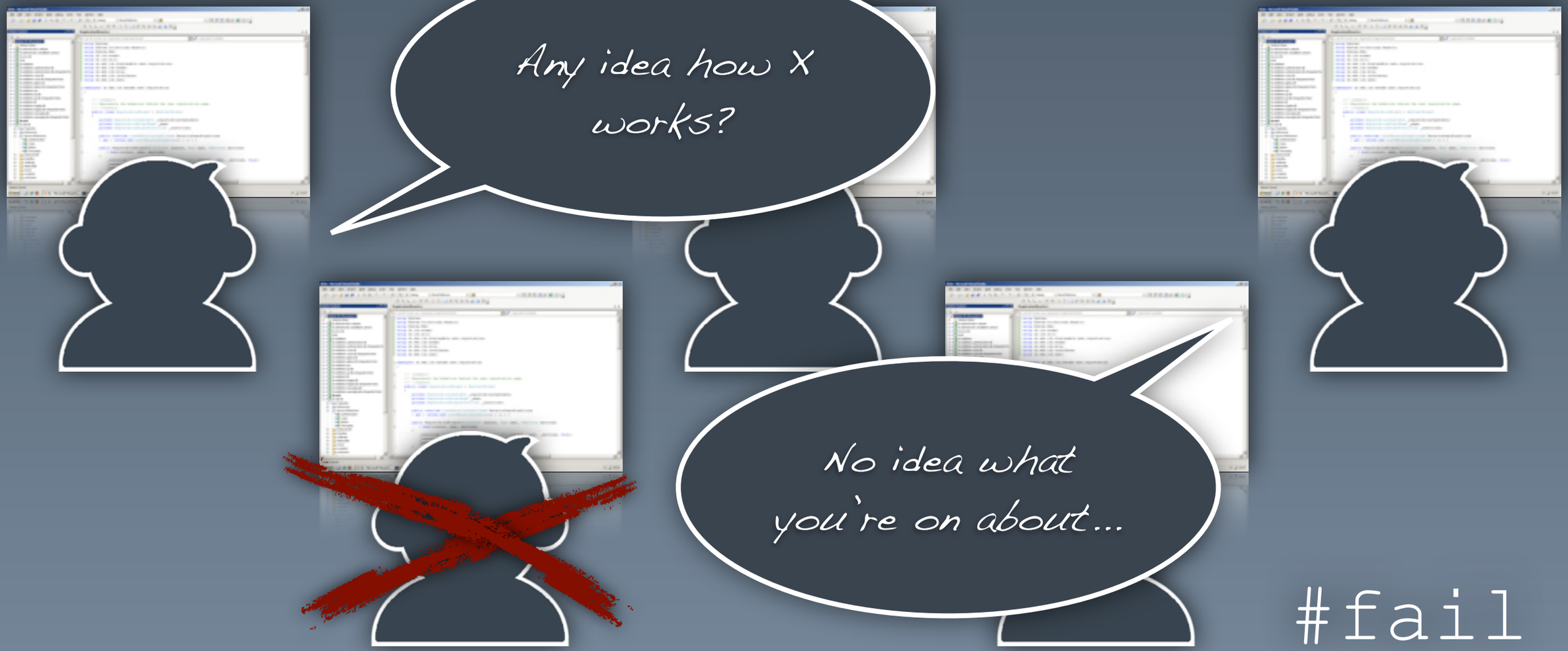
Tribal knowledge

"just talk!"

"diagrams and documents
are just props
for conversations"

The bus factor

(it's not just about buses though!)



#fail

Current Development Team

Business Sponsors

Future Development Team

Database Administrators

Your system

Operations/Support Staff

Other Teams

Security Team

Software architecture is a platform for conversation ... be social!

Compliance and Audit

Software ~~Architecture~~ ~~Document~~ Guidebook

Maps

Sights and itineraries

History and culture

Practical Information

Context
What is this all about?

Functional Overview
What does the system do?

Quality Attributes
Are there any significant non-functional requirements?

Constraints
Are there any significant constraints?

Principles
What design and development principles have been adopted?

Software Architecture
What does the big picture look like and how is the system structured?

External Interfaces
What are the external system interfaces?

Code
Are there any implementation details you need to explain?

Data
What does the data model look like and where is it being stored?

Infrastructure Architecture
What does the target deployment environment look like?

Deployment
What is the mapping between software and infrastructure?

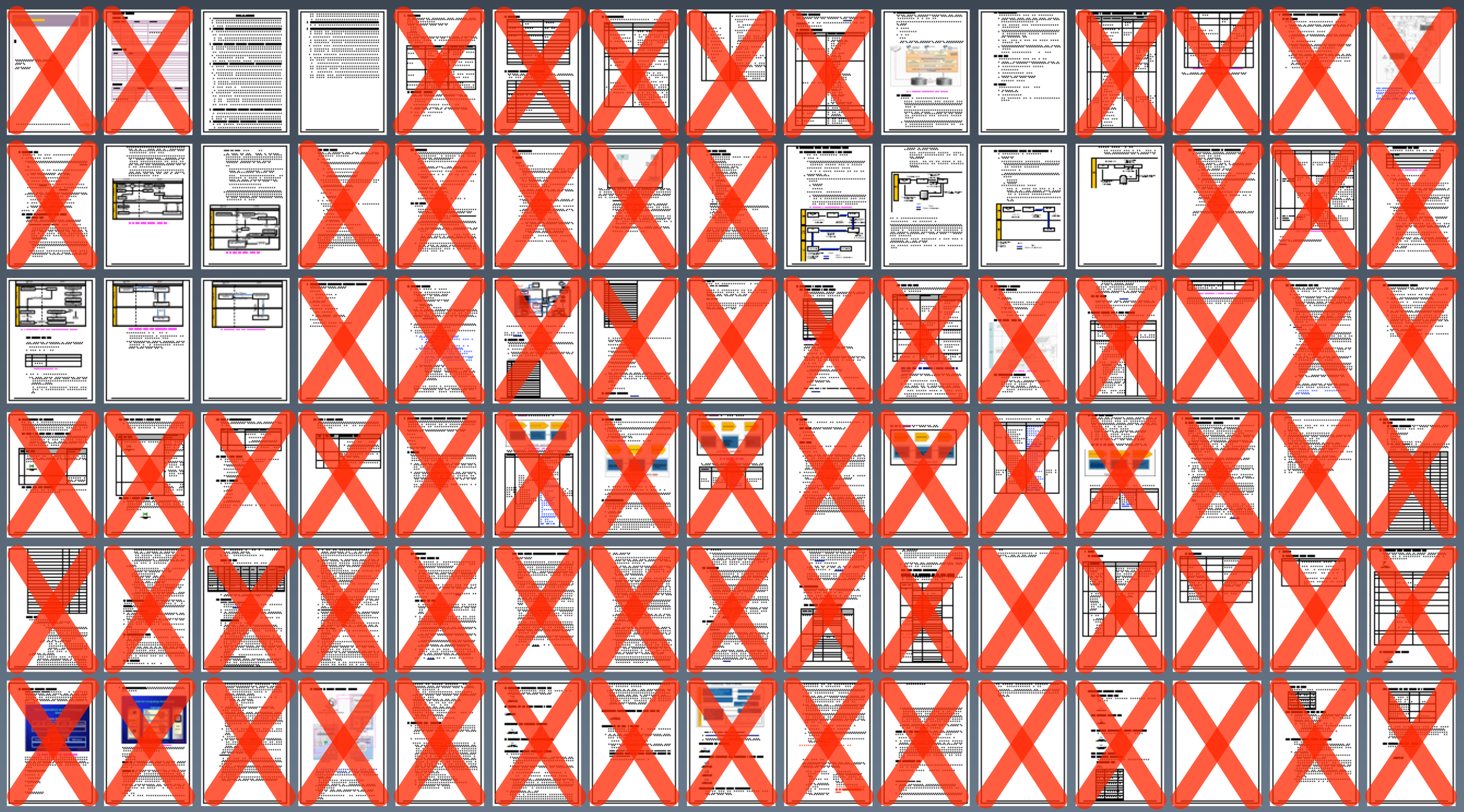
Operation & Support
How will people operate and support the system?

Documentation should

describe what the code doesn't

Reduce waste, add value

Use it to explain intent and act as a guide to navigate the source code



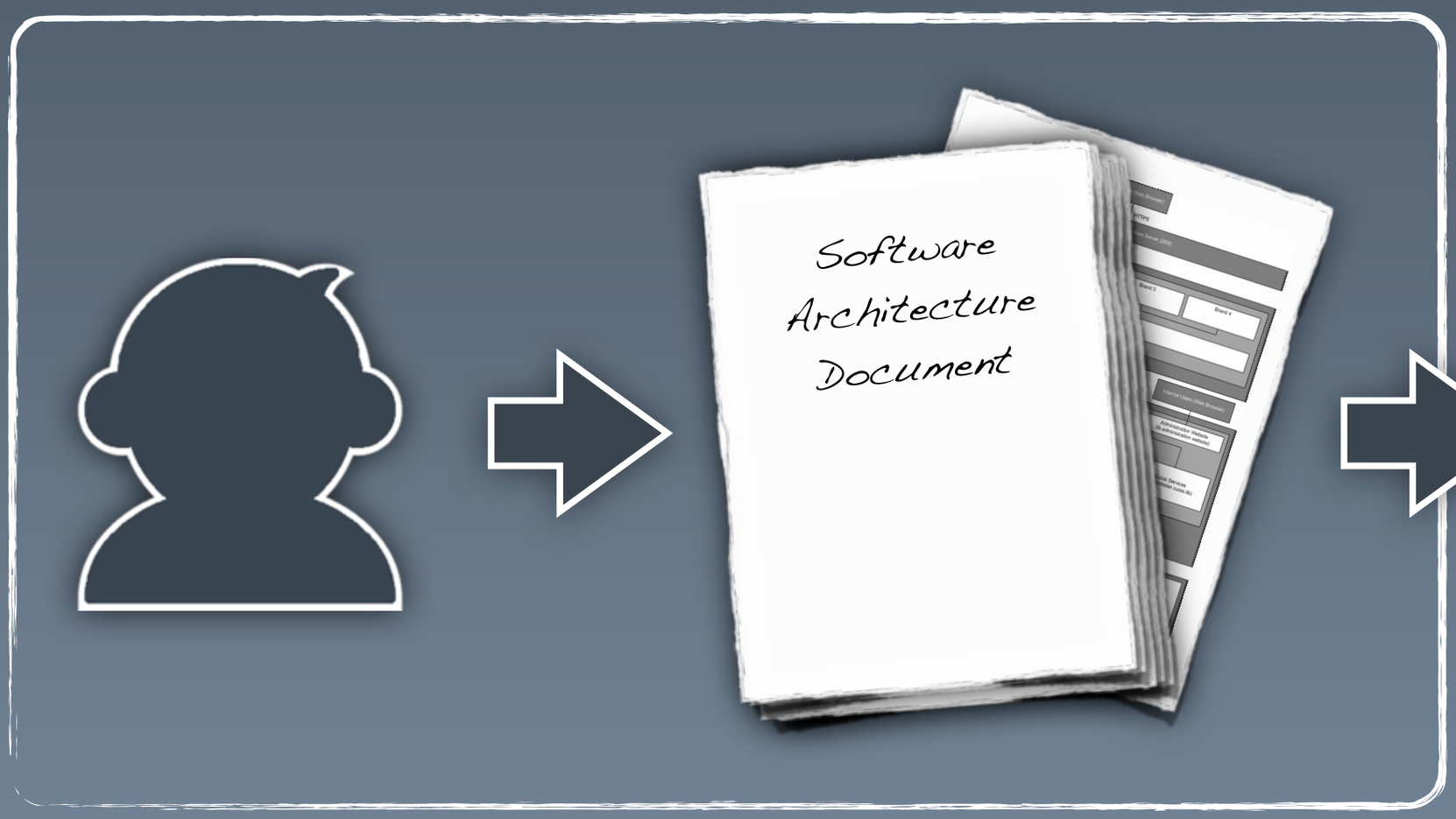
How much of the document is

up to date and relevant?

Software architecture in the development lifecycle

Software development is not a

relay sport



AaaS ... architecture as a service

The software architecture role should be engaged throughout

(not just analysis and design)

How much up front design should you do?

Big design up front?

Emergent design?
(or none, depending on your viewpoint!)

Waterfall

RUP®
Rational Unified Process®

Extreme Programming

DSDM Atern

ScrumAlliance®
transforming the world of work

Something in between?

You should do

“just enough”

Base your architecture on requirements, travel light and prove your architecture with concrete experiments.

Scott Ambler
http://www.agilemodeling.com/essays/agileArchitecture.htm

What is architecturally **significant**?

Costly to change
(can you refactor it in an afternoon?)

Complex

New

You need to

identify and mitigate your highest priority risks

Things that will cause
your project to fail
or you to be fired!

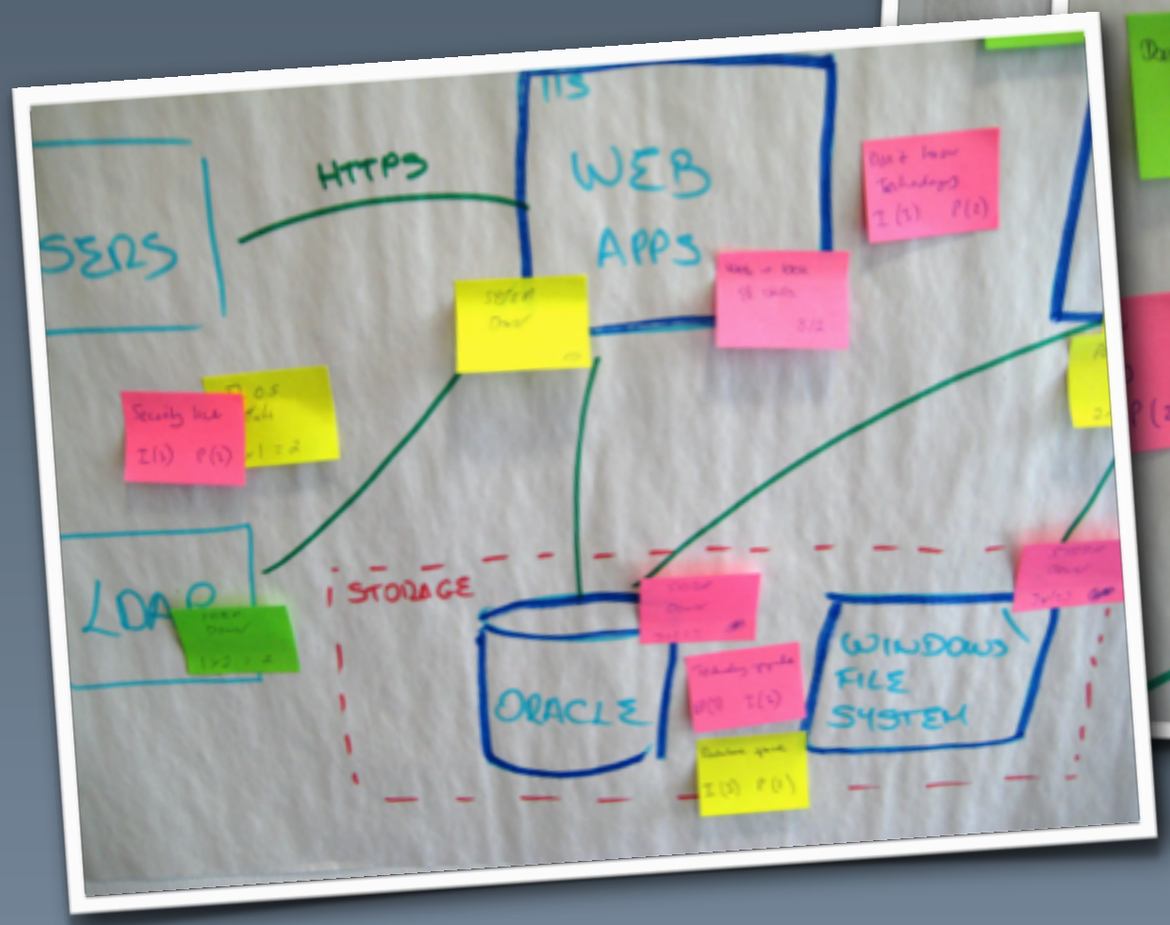
Probability

Impact	Low (1)	Medium (2)	High (3)
Low (1)	1	2	3
Medium (2)	2	4	6
High (3)	3	6	9

Who looks after the risks on most software projects?

A (usually) non-technical project manager!

Risk-storming



A collaborative and visual technique for identifying risk

You still need to deal with the risks

(mitigation strategies include hiring people,
undertaking proof of concept
and changing your architecture)

How much up front design should you do?

“Just enough”

Understand how the significant elements fit together

Identify and mitigate the key risks

Provide firm foundations and a vision to move forward

The software architecture **role** and the **process** of software architecting are different

The software architecture role

Dedicated
software architect

Single point of responsibility for the technical aspects of the software project

From chaos to self-organising

Elastic Leadership (Roy Osherove)
Survival (command and control),
learning (coaching),
self-organising (facilitation)

Everybody is a
software architect

Joint responsibility for the technical aspects of the software project

The process of software architecting

From big design up front to evolutionary

```
/// <summary>
/// Represents the behaviour behind the ...
/// </summary>
public class SomeWizard : AbstractWizard
{
    private DomainObject _object;
    private WizardPage _page;
    private WizardController _controller;

    public SomeWizard()
    {
    }

    ...

}
```

Big up front design

Requirements capture, analysis and design complete before coding starts

Evolutionary architecture

The architecture evolves secondary to the value created by early regular releases of working software

The role

21st century software architecture

"just enough"

Understand how the significant elements fit together

Identify and mitigate the key risks

Provide firm foundations and a vision to move forward

The process

```
/// <summary>
/// Represents the behaviour behind the ...
/// </summary>
public class SomeWizard : AbstractWizard
{
    private DomainObject _object;
    private WizardPage _page;
    private WizardController _controller;

    public SomeWizard()
    {
    }

    ...

}
```

Do whatever works for

you