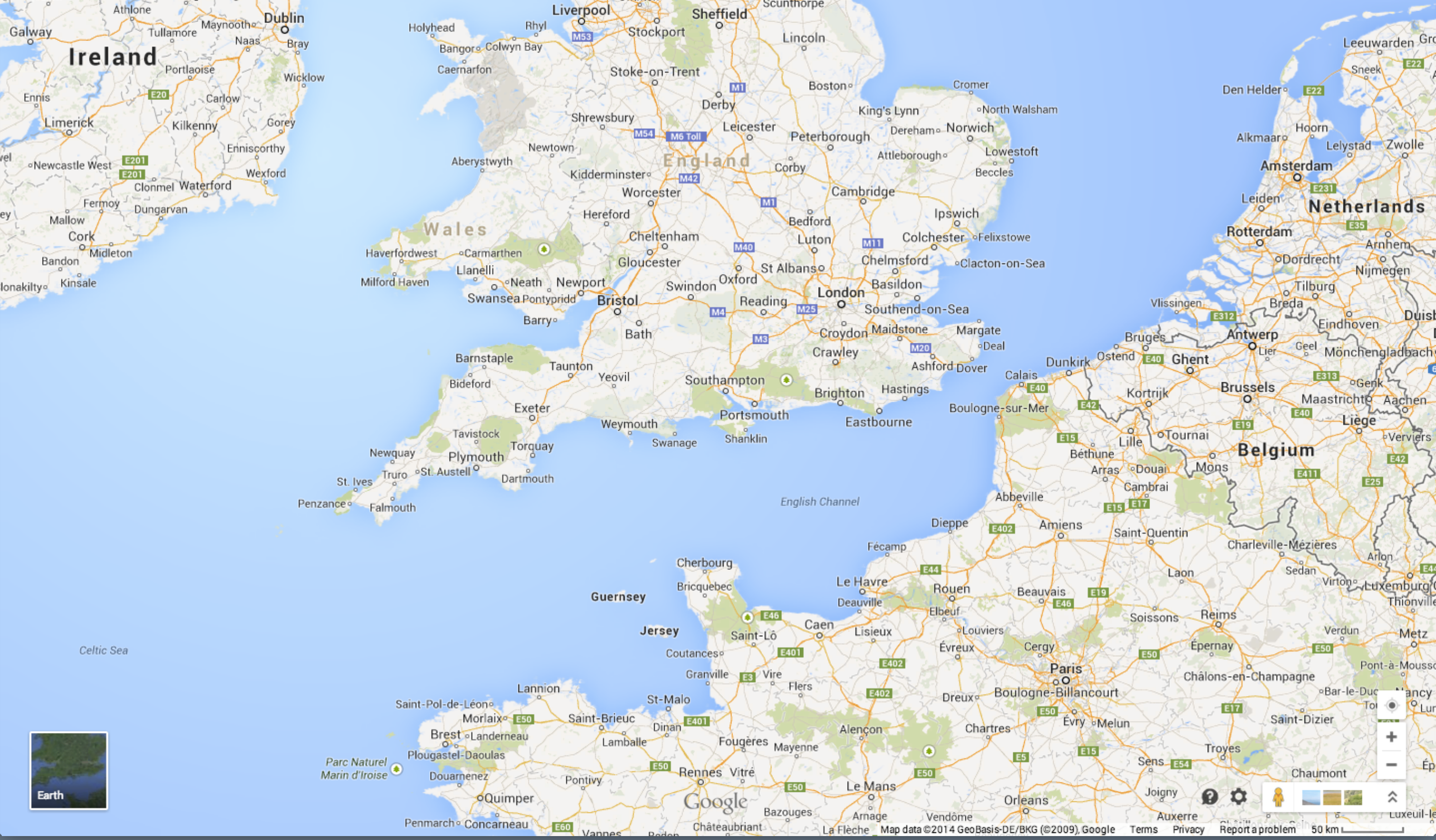
coding {the} architecture

Software Architecture vs Code

Simon Brown
@simonbrown

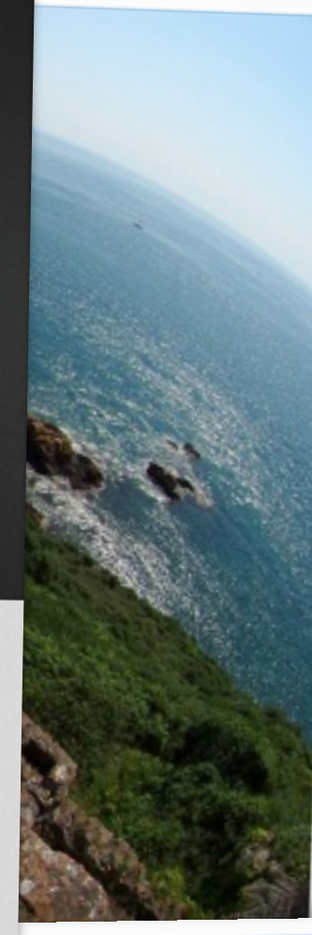


Jersey

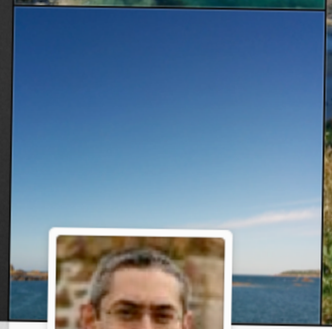
JERSEY FINANCE
VOICE OF THE INTERNATIONAL FINANCE CENTRE

DIGITAL JERSEY

Locate Jersey



simonbrowndotje

Follow

Simon Brown Software developer, author, speaker, etc from Jersey, Channel Islands
http://www.simonbrown.je

138 posts | 41 followers | 32 following

June 2014









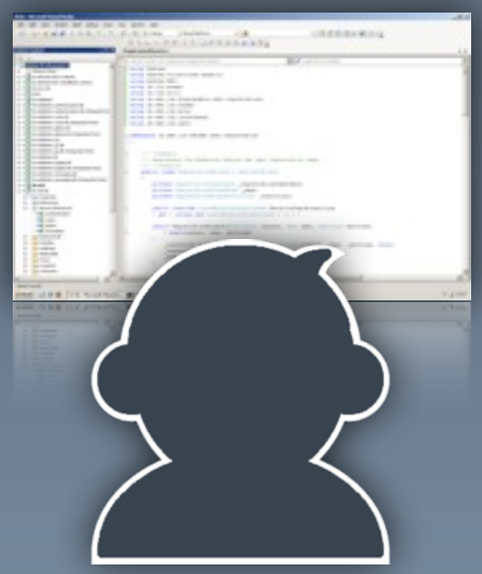

May 2014





I help software teams understand

software architecture, technical leadership and the balance with agility



I code too

⇧ ; - ⇧ 0

A developer-friendly guide to software architecture, technical leadership and the balance with agility

Leanpub

10 out of 10

"Highly recommended reading"

Junilu Lacar, JavaRanch

http://leanpub.com/software-architecture-for-developers

The intersection between

software architecture

and

code

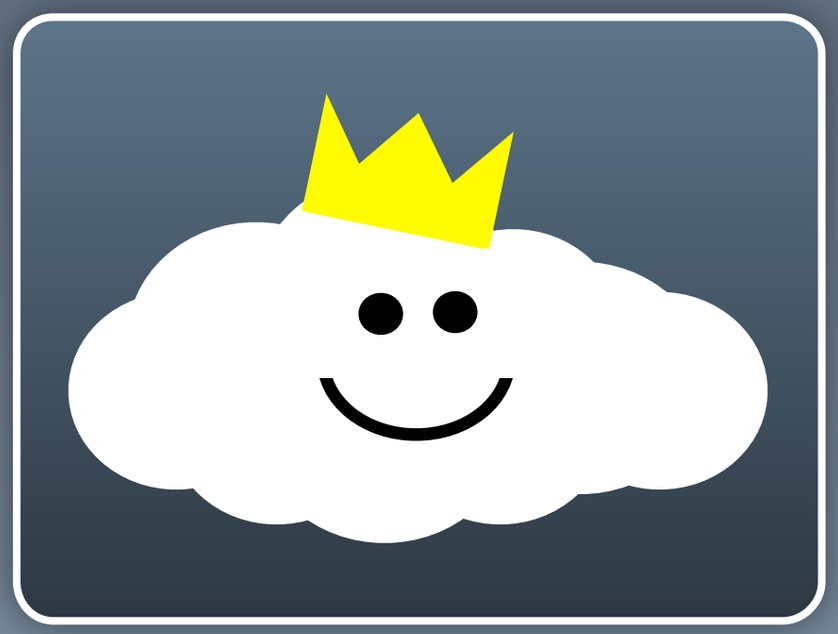
Software architecture

There’s a common misconception that software architecture should be

conceptual

and

exclude technology



As a noun...

Structure

The definition of something in terms of its components and interactions

As a verb...

Vision

The process of architecting;
making decisions based upon business goals,
requirements and constraints,
plus being able to communicate this to a team

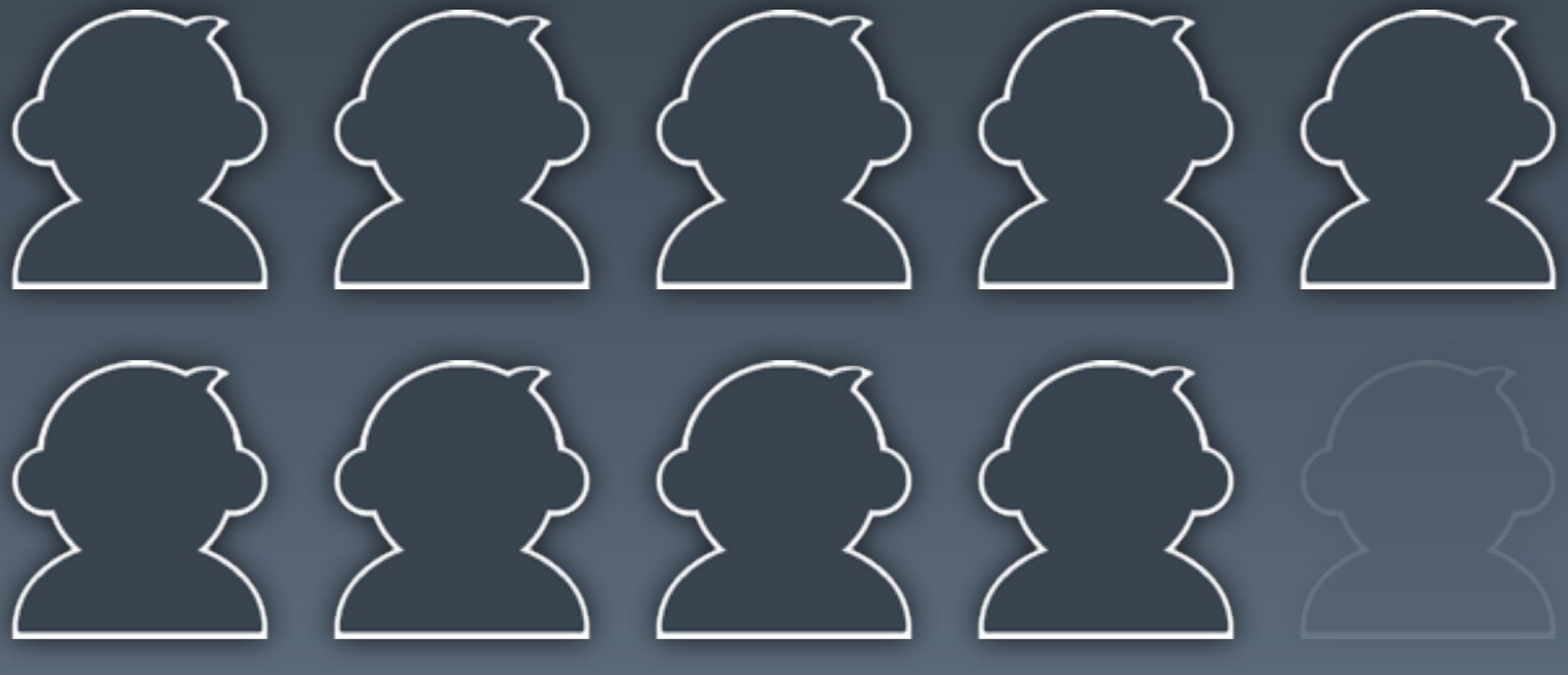
How do we

communicate

software architecture?

Who here uses UML

on a regular basis?

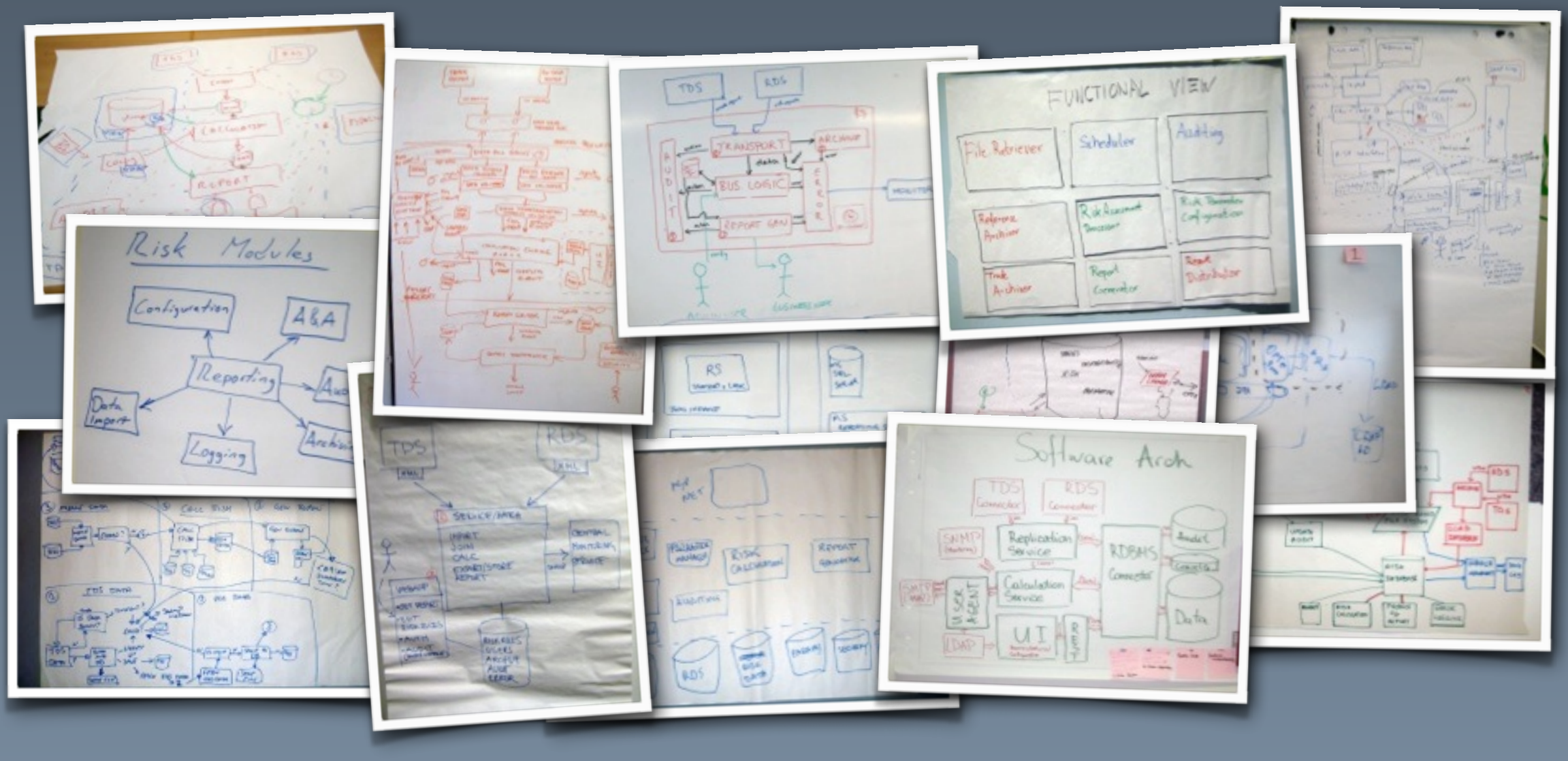


9 out of 10 people don't use UML

(in my experience)

NoUML

diagrams?



Why has UML fallen out of fashion?

UML

is too complex,
is too low-level,
nobody knows it

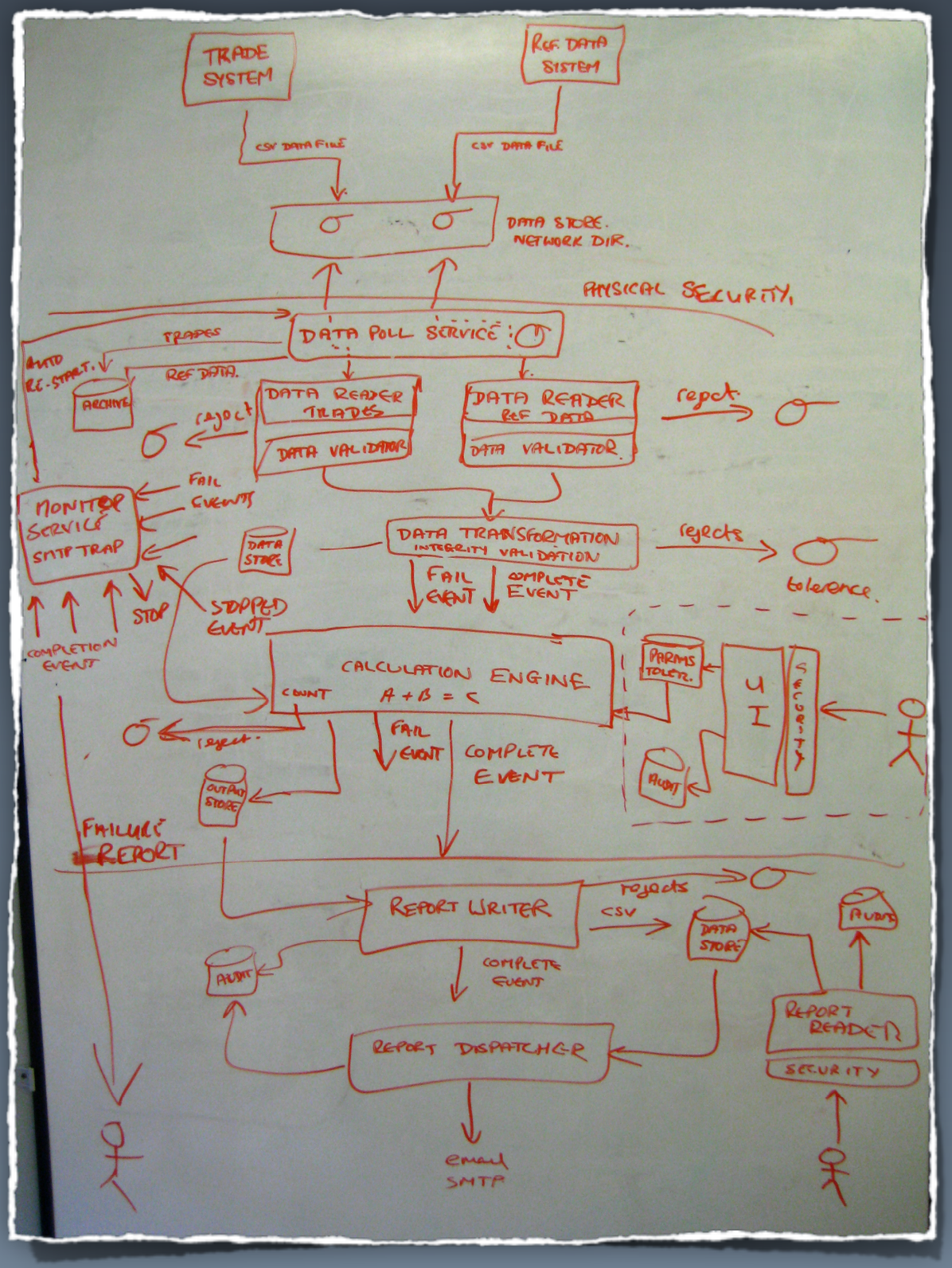
Oh, plus we're

agile

and/or do

TDD

Software architecture provides boundaries for TDD



It's usually difficult to show the entire design on a **single** diagram

Different **views** of the design can be used to manage complexity and highlight different aspects of the solution

The description of an architecture—the decisions made—can be organized around these four views, and then illustrated by a few selected *use cases*, or *scenarios* which become a fifth view. The architecture is in fact partially evolved from these scenarios as we will see later.

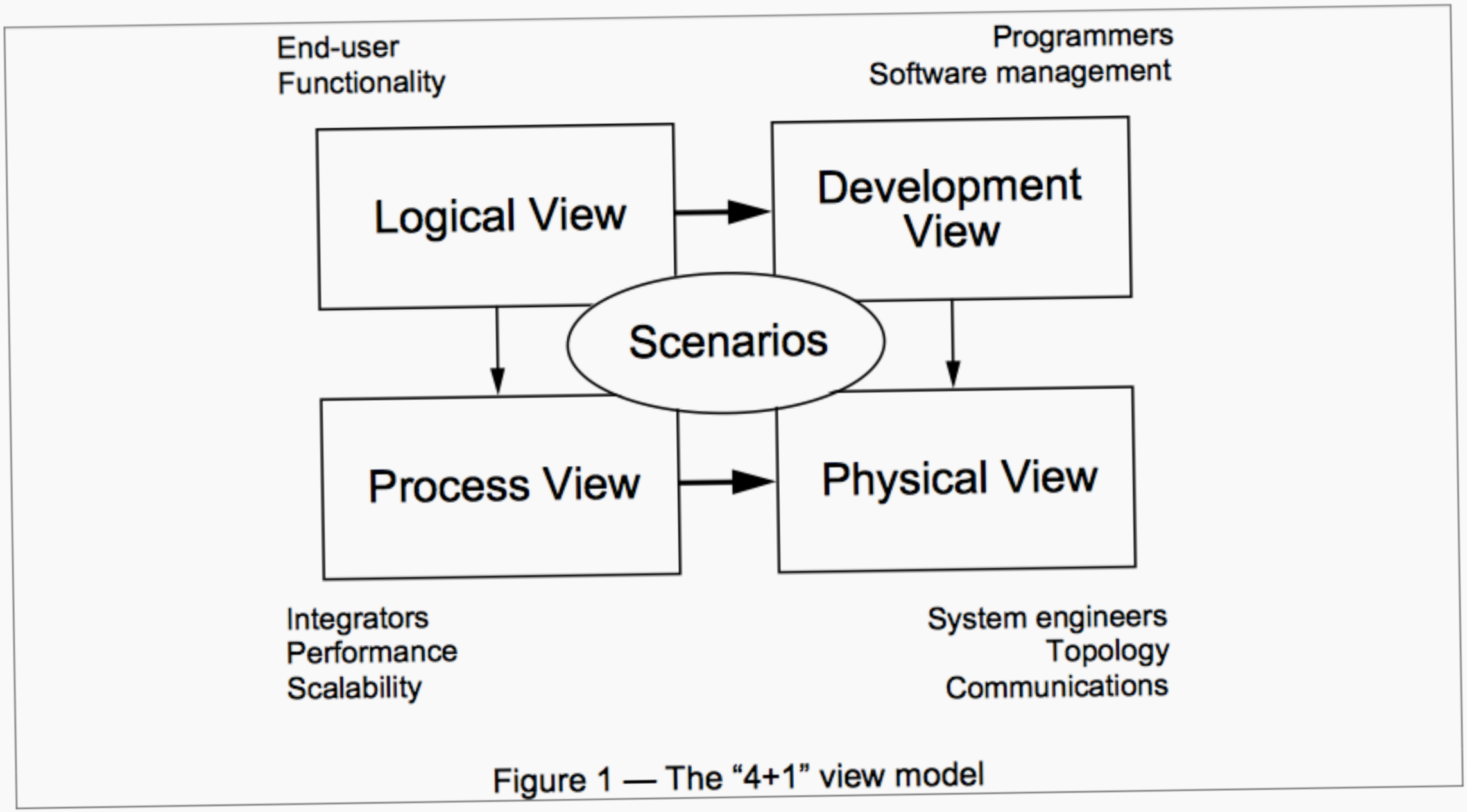


Figure 1 — The "4+1" view model

We apply Perry & Wolf's equation independently on each view, i.e., for each view we define the set of elements to use (components, containers, and connectors) , we capture the forms and patterns that work, and we capture the rationale and constraints, connecting the architecture to some of the requirements.

Do the **names**
of those views make sense?

Conceptual vs Logical

Process vs Functional

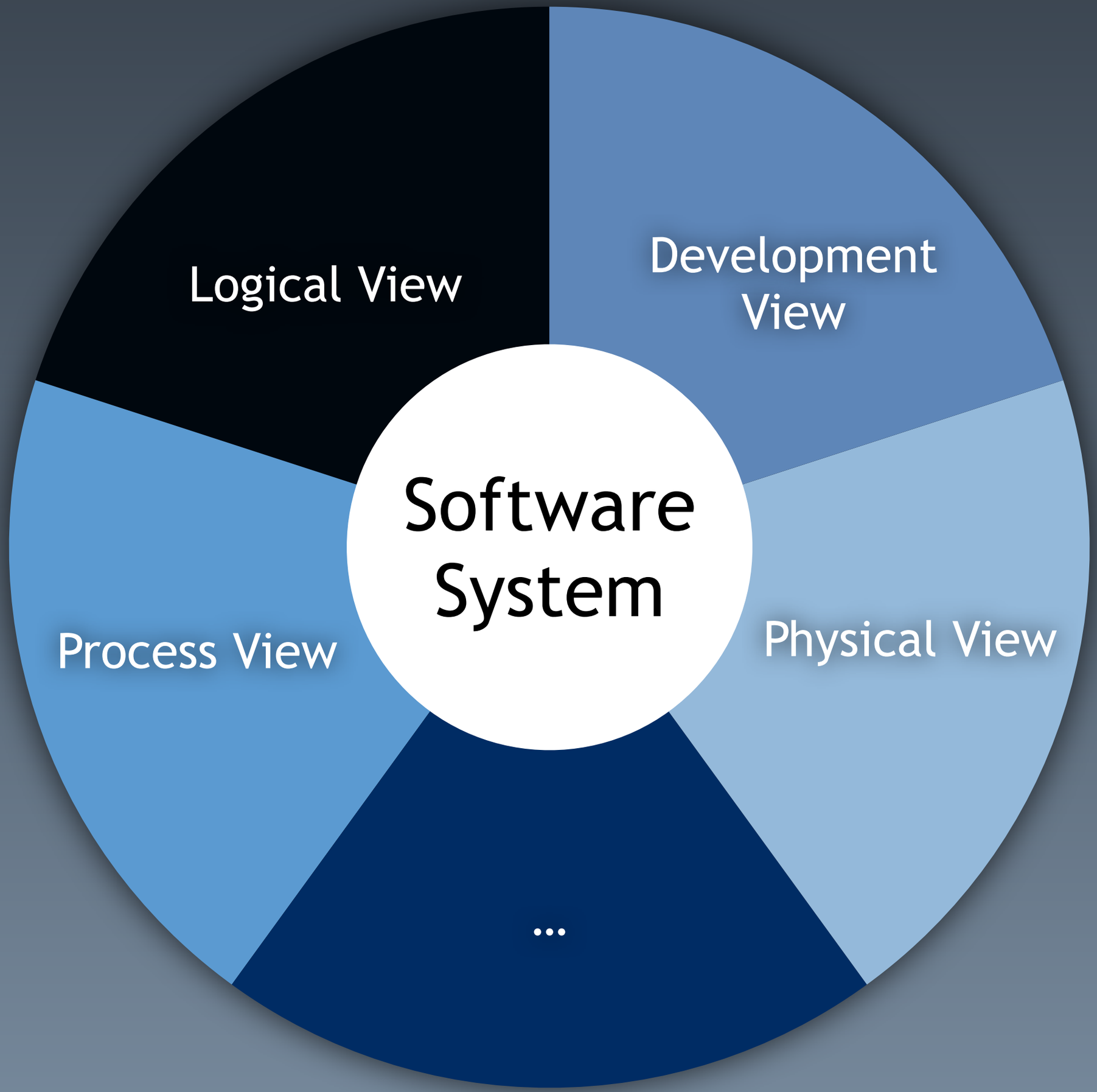
Development vs Physical

Development vs Implementation

Physical vs Implementation

Physical vs Deployment

Logical and development views are often separated

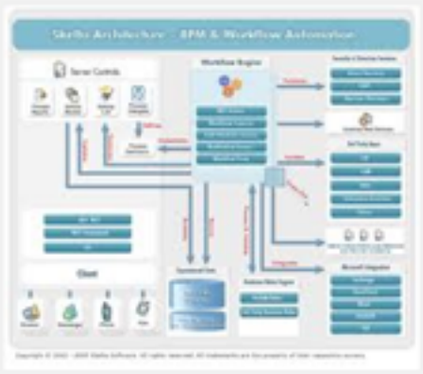

Logical View
Development View
Software System
Process View
Physical View
...

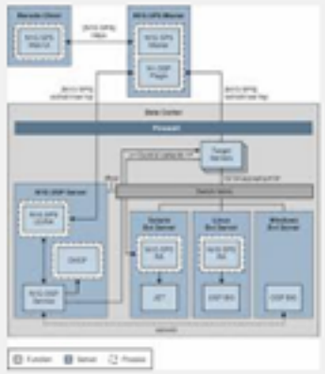
BRAIN FREEZE!

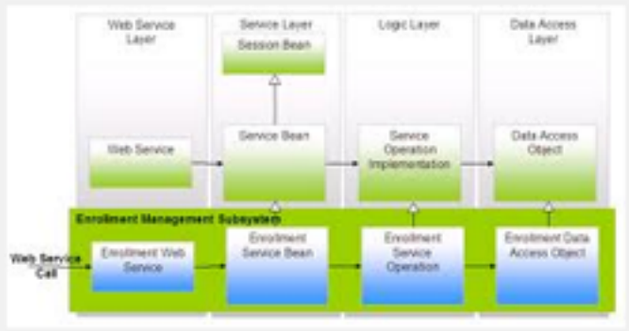
Google software architecture diagram

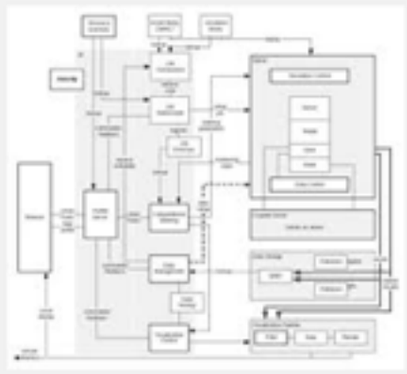
Sign in

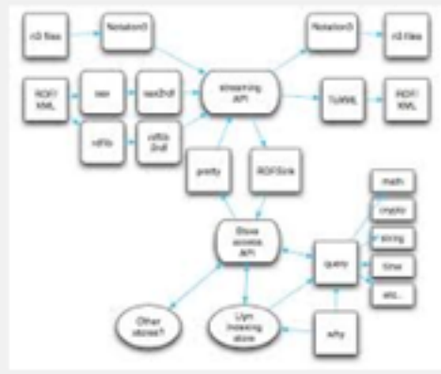
Web Images Videos Shopping News More Search tools SafeSearch

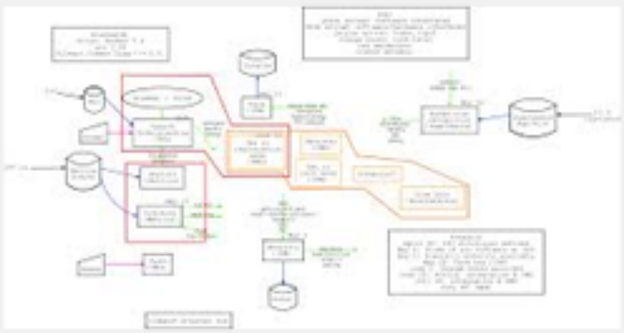


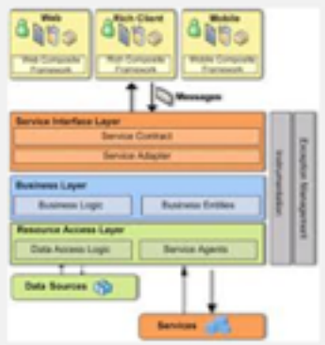


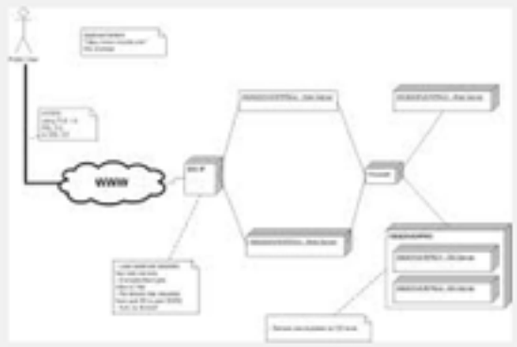


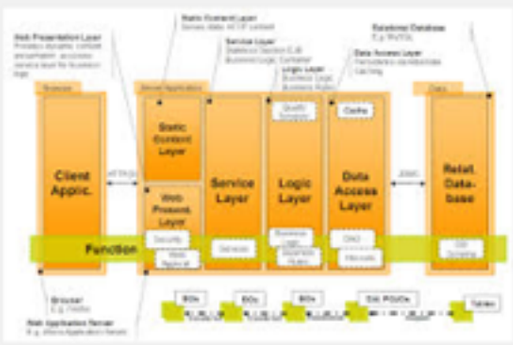


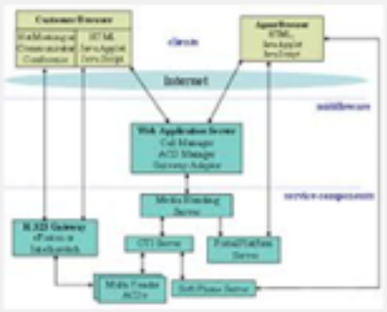


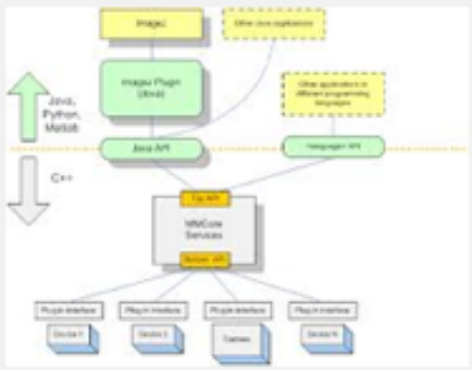


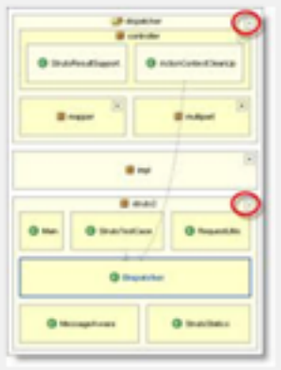


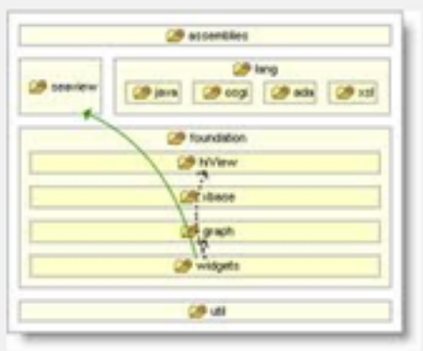


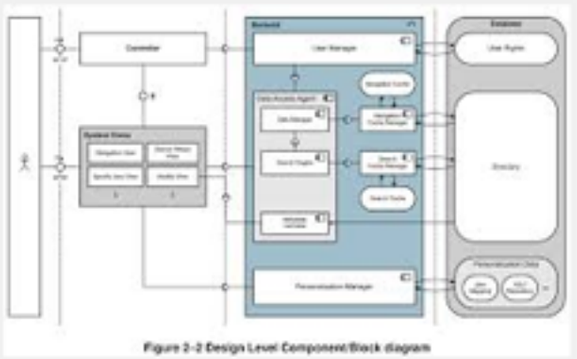


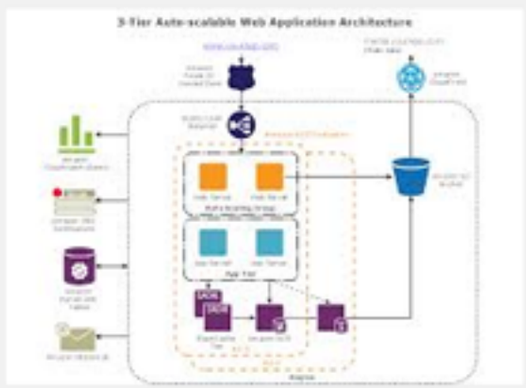








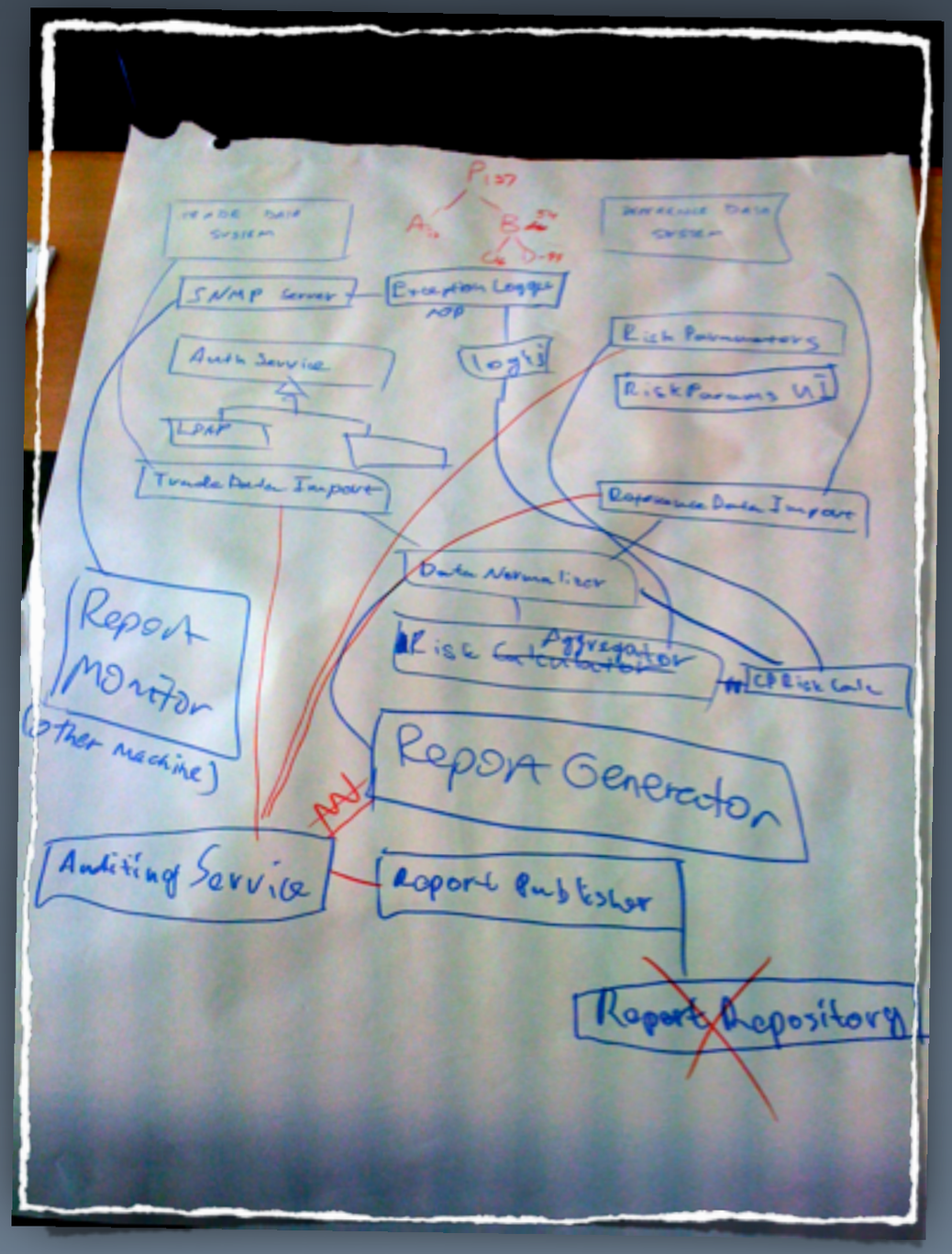


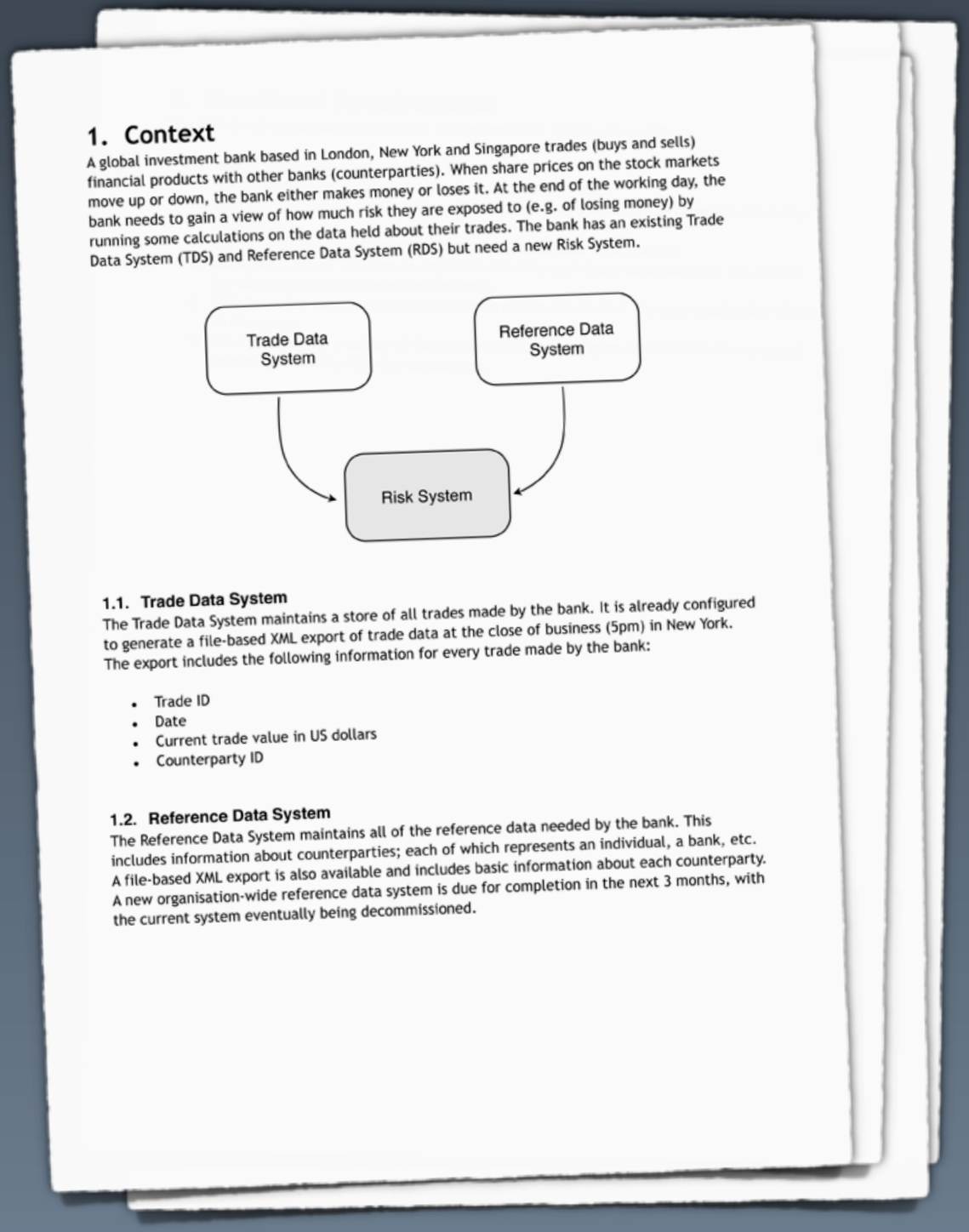


Video over IP
Capturing video frames from Camera
Format conversion and Pre-processing
H.264 Encoder
NALU Fragmentation
Audio/Video sync module
Format conversion and Post-processing
H.264 Decoder
Reconstruction of NALU
RTP/ RTCP
Voice over IP
Acoustic echo cancellation (AEC)
LEC (G.168/ G.165)
AGC - Gain Control
VAD
Speech Encoder
Audio Interface layer
Sound card Interface (PCI / USB)
FXS/TDM Interface
Telephony Interface layer - Tone detection/generation, CPT/DTMF, MF-R1/R2, and Voice Fax Modem Discrimination Unit
Administration UI
Reservation Manager UI
AutoShell
Packaging
Process Automation
Configuration Management / Compliance
Help Desk
Reporting (Jasper)
Agent Remote Deployment
Service Response Monitoring
CA IT Client Manager
CA Patch Manager
CA Process Automation
CA Configuration Automation
CA Service Desk
Chargeback
Agent Policy Configuration
Server Monitoring (Local & Remote)
Web Services and Reliable Transport (Active MQ)
Storage
Platform Management
Microsoft HyperV
Provisioning
Rapid Server Imaging
Automation Infrastructure Platform
AOM Data Service
Policy, Performance, and Resource Management
Authentication / Security (CA EEM)
State and Event Management
Scheduler (Windows)
Discovery (CA Network Discovery Gateway)
Management DB
Performance DB
Service Controller
Rich UI
Web UI
Controls
Service Interface
Activity
Business Rules
Human Workflow
Workflow
Service Agents
DAL
E-Publish
EAI
ECM
DW
OLTP
Signing
Authentication
Authorization
Monitoring
Log & Trace
Exception Management
Configuration
Presentation Sublayer
Project Management
Project Editing Service
Computational Analysis Service
e-AIRSmesh
e-AIRSview
Mesh Converter
CFD Solver
Output Converter
Remote Experiment Service
Exp. Exec.
Exp. Info.
Aerodynamic Exp.
PIV Exp.
Parametric Study
Validation
Analysis
Collaboration
Session Reservation
Resource Configuration Service
Resource Monitoring Service
P / HTTP
ASP (Scientific Application Service Provider)
Basic Execution Service
Parameter Sweeper
Global Scheduler
Resource Catalog
GRAM adapter
Plotting Service
eAIRS Plot
Conv. Graph
Data Transfer Service
Metadata Management Service
AG + VNC
Visualization Sharing
Mobile Service Component
A/V Conference
Chat
Application
ASCII File Editor
Library
ASCII Files
SCADA Develop. Environ
Export / Import
Commercial DB
Project Editor
Commercial Develop tools
Driver Toolkit
HMI
Alarm Display
Log Display
Active X Controls
3rd Party Applic.
Active X Container
Client - Server - Publish / Subscribe - TCP/IP
SCADA Server
Recipe DB
Recipe Managt
Data Process-sing
Report Gener.
Alarm
Log
Archive
Ref. DB
RT DB
SQL
Alarm DB
Log DB
Archive DB
ODBC
DDE
API DLL
Data R/W
Driver
OPC
Private Application
EXCEL
VME
PLC
PLC
Typical generic software architecture of SCADA systems
ote administration interface
Output device manager
Output devices
User manager
DB-Server
Analysis

In my experience,

software teams aren't able to effectively visualise the software architecture of their systems





1. Context

A global investment bank based in London, New York and Singapore trades (buys and sells) financial products with other banks (counterparties). When share prices on the stock markets move up or down, the bank either makes money or loses it. At the end of the working day, the bank needs to gain a view of how much risk they are exposed to (e.g. of losing money) by running some calculations on the data held about their trades. The bank has an existing Trade Data System (TDS) and Reference Data System (RDS) but need a new Risk System.

1.1. Trade Data System

The Trade Data System maintains a store of all trades made by the bank. It is already configured to generate a file-based XML export of trade data at the close of business (5pm) in New York. The export includes the following information for every trade made by the bank:

- Trade ID
- Date
- Current trade value in US dollars
- Counterparty ID

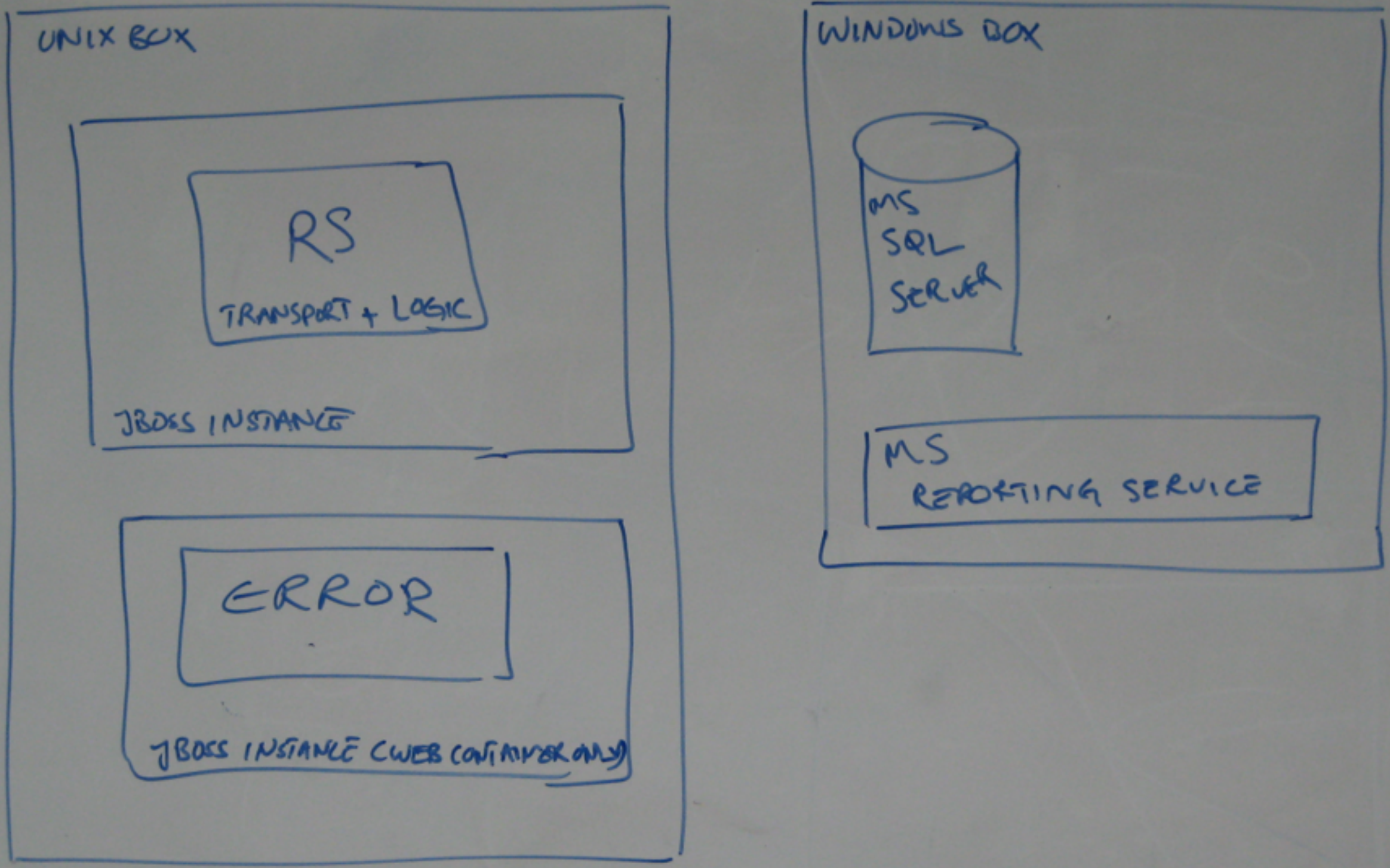
1.2. Reference Data System

The Reference Data System maintains all of the reference data needed by the bank. This includes information about counterparties; each of which represents an individual, a bank, etc. A file-based XML export is also available and includes basic information about each counterparty. A new organisation-wide reference data system is due for completion in the next 3 months, with the current system eventually being decommissioned.

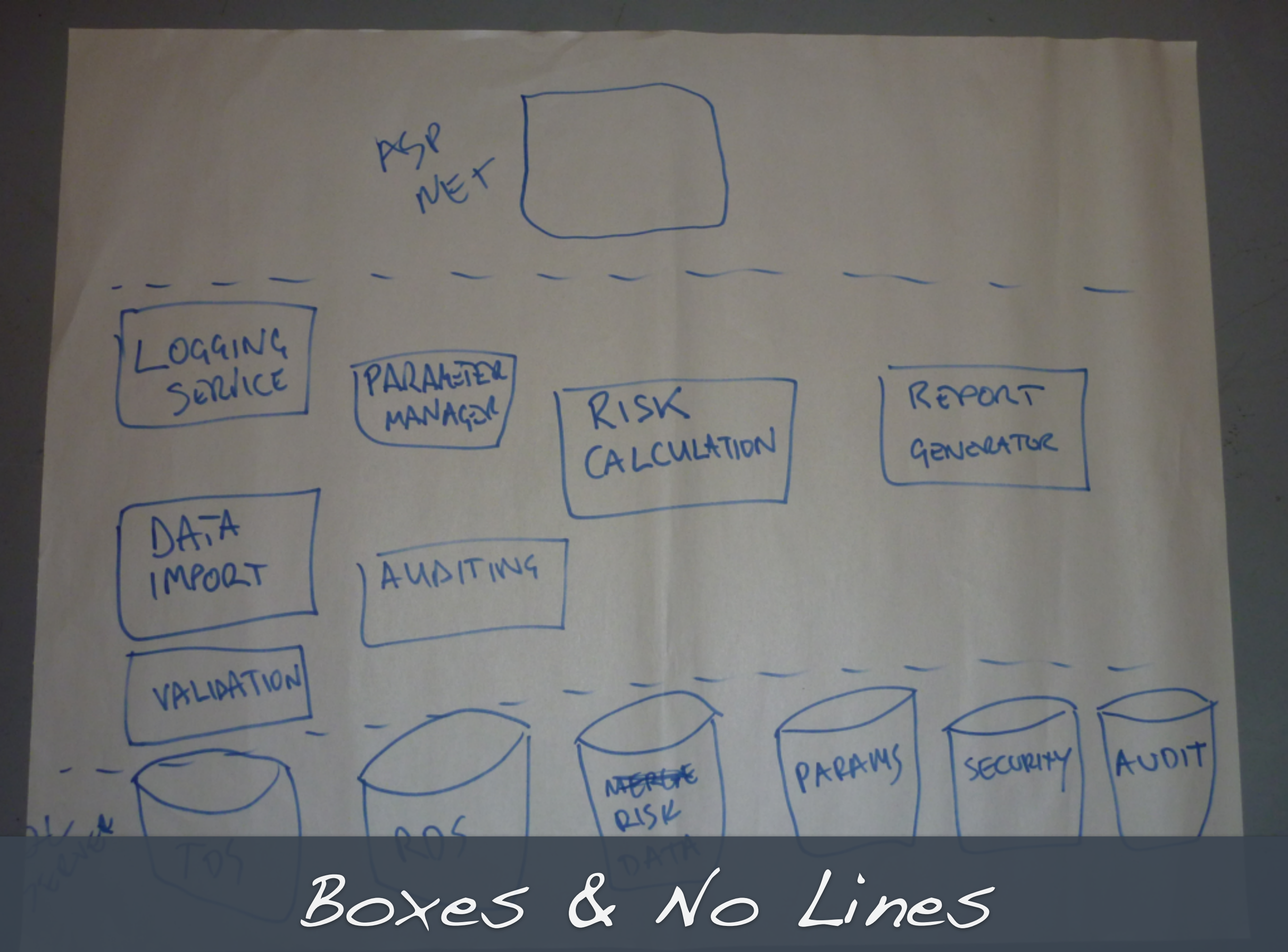
Financial risk system

1. Import data from a Trade Data System (TDS).
2. Import data from a Reference Data System (RDS).
3. Merge the data feeds, perform some risk calculations and generate a Microsoft Excel file of the risk report.
4. Allow a subset of users to modify some parameters used during the calculation process.

http://bit.ly/sa4d-risksystem



The Shopping List



Boxes & No Lines

FUNCTIONAL VIEW

File Retriever

Scheduler

Auditing

Reference Archiver

Risk Assesment Processor

Risk Parameter Configuration

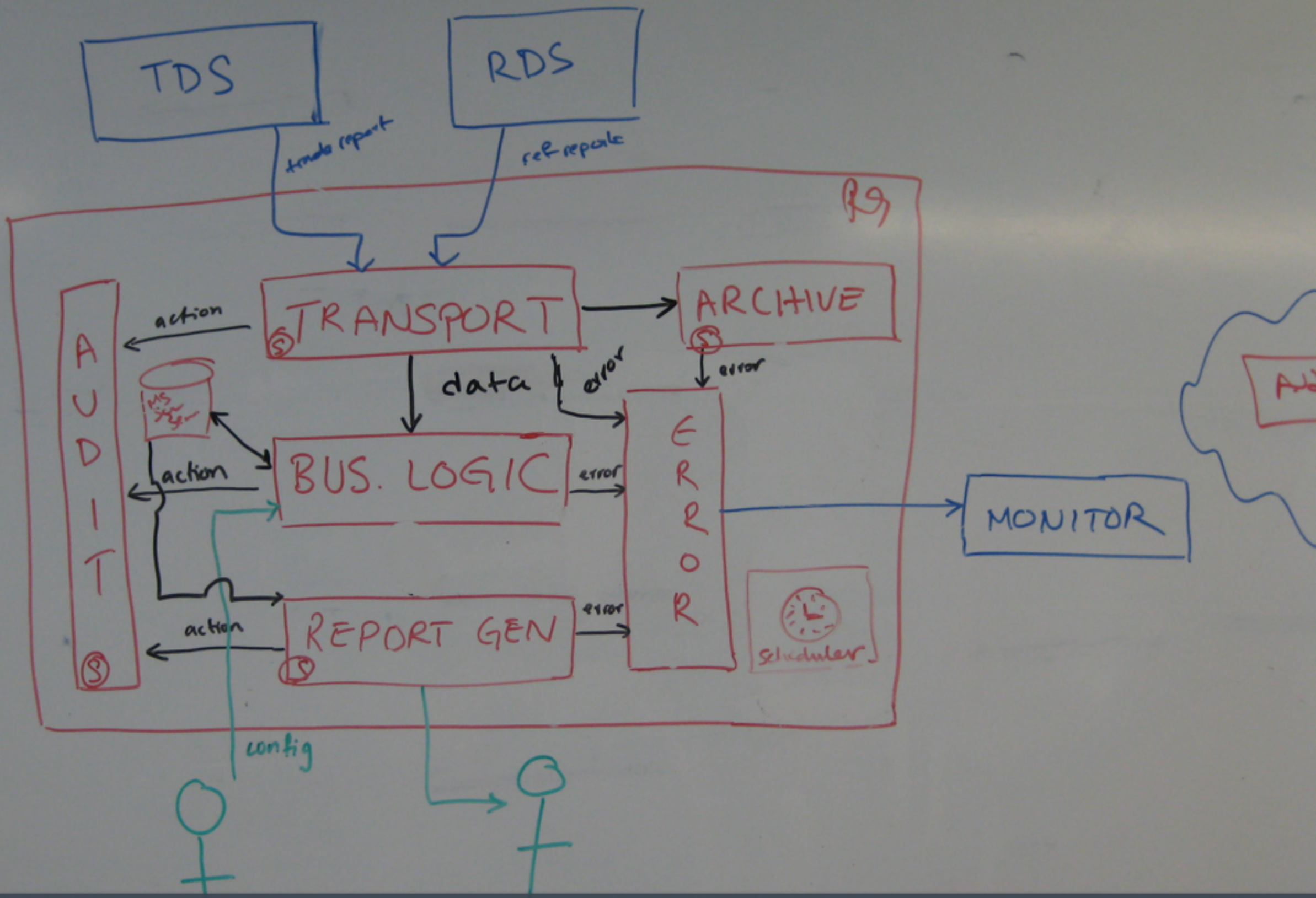
Trade Archiver

Report Generator

Report Distributor

The "functional view

The Airline Route Map



Generically True

XML

2 SERVICE/BATCH

IMPORT
JOIN
CALC

EXPORT/STORE
REPORT

CENTRAL
MONITORING
SERVICE

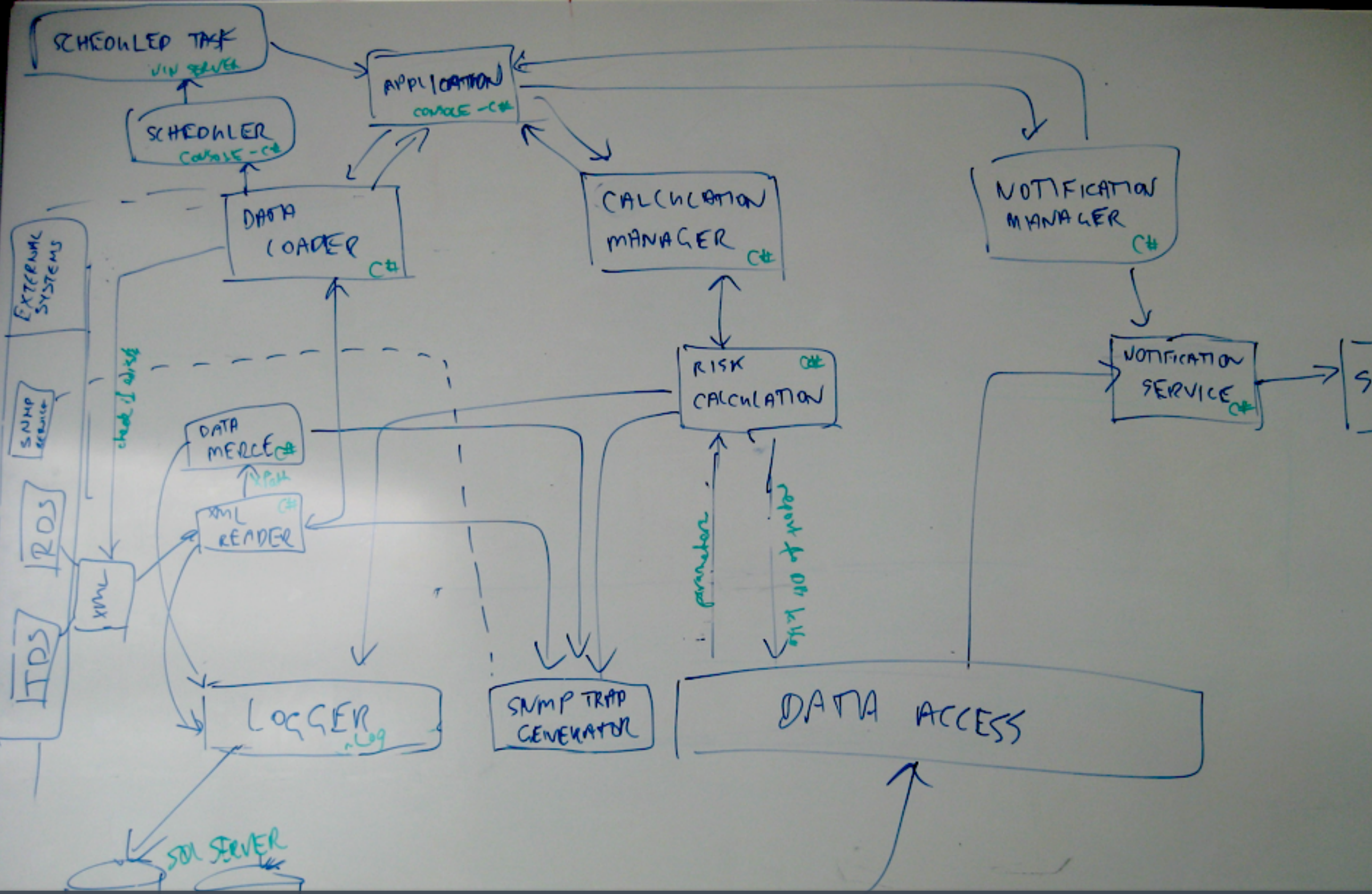
SNMP

3 WEBAPP

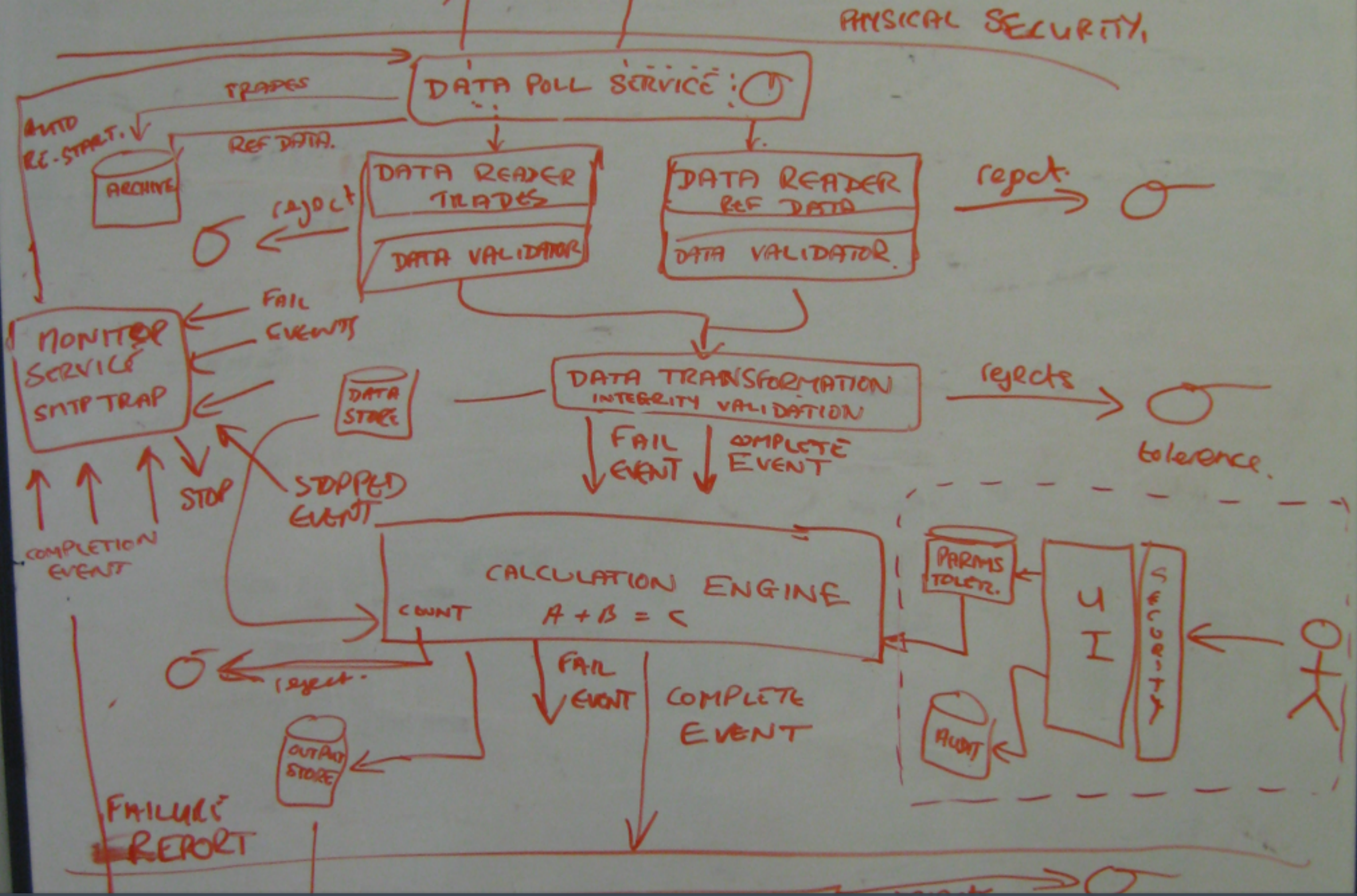
- GET REPORT
- EDIT RISK RULES
- AUTH
- AUDIT (RULES CHANGES)

RISK RULES
USERS
AUDIT

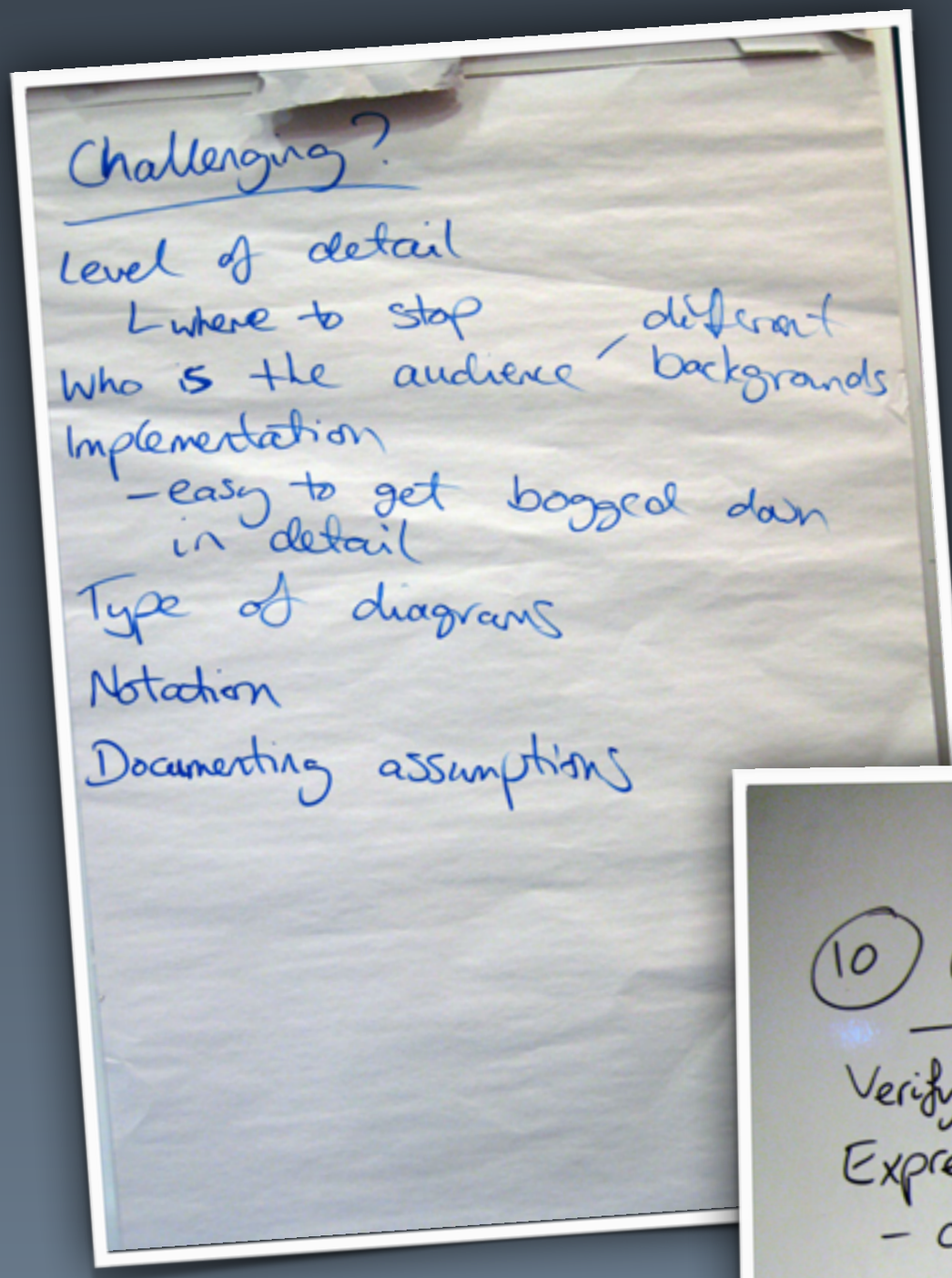
The "logical view"

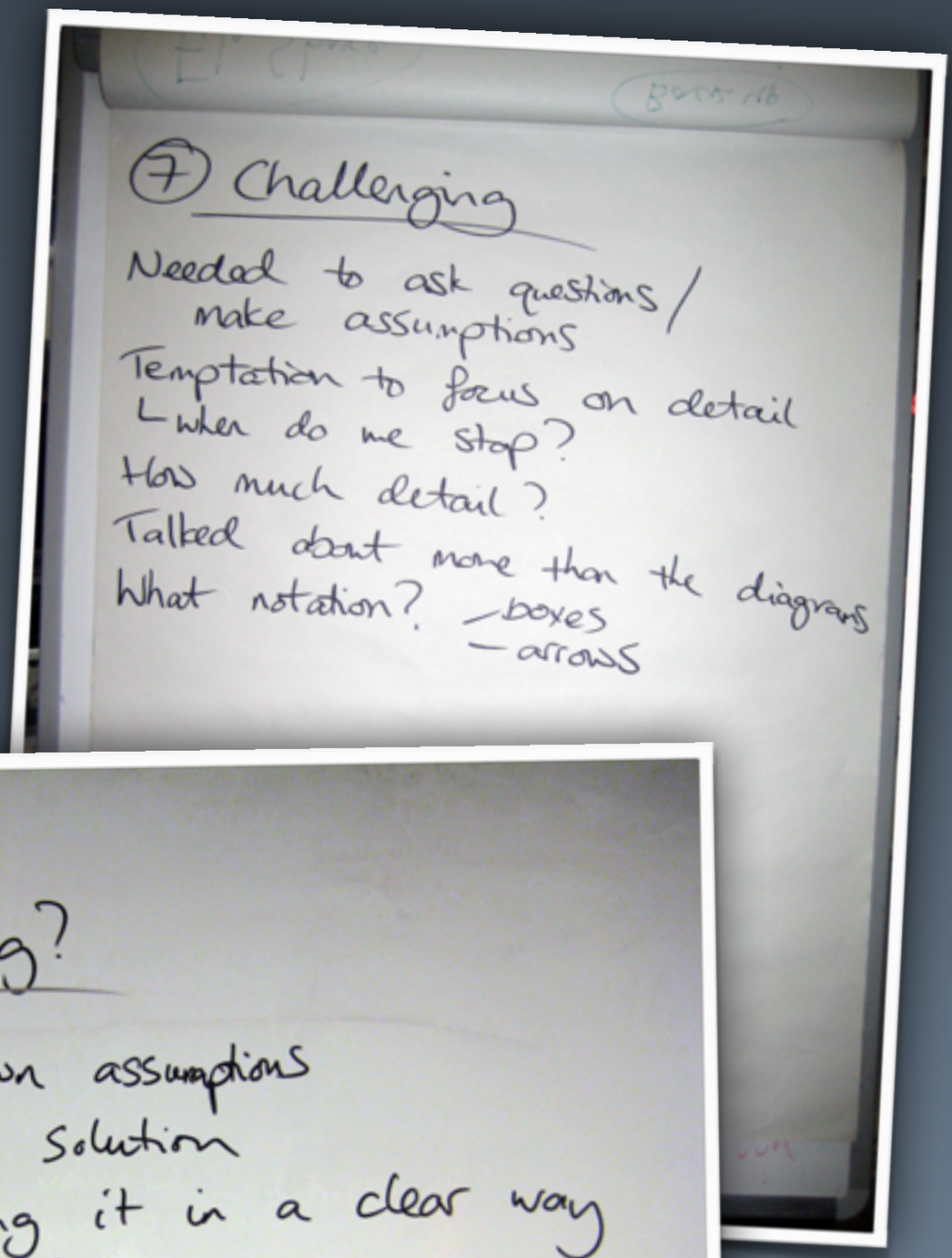


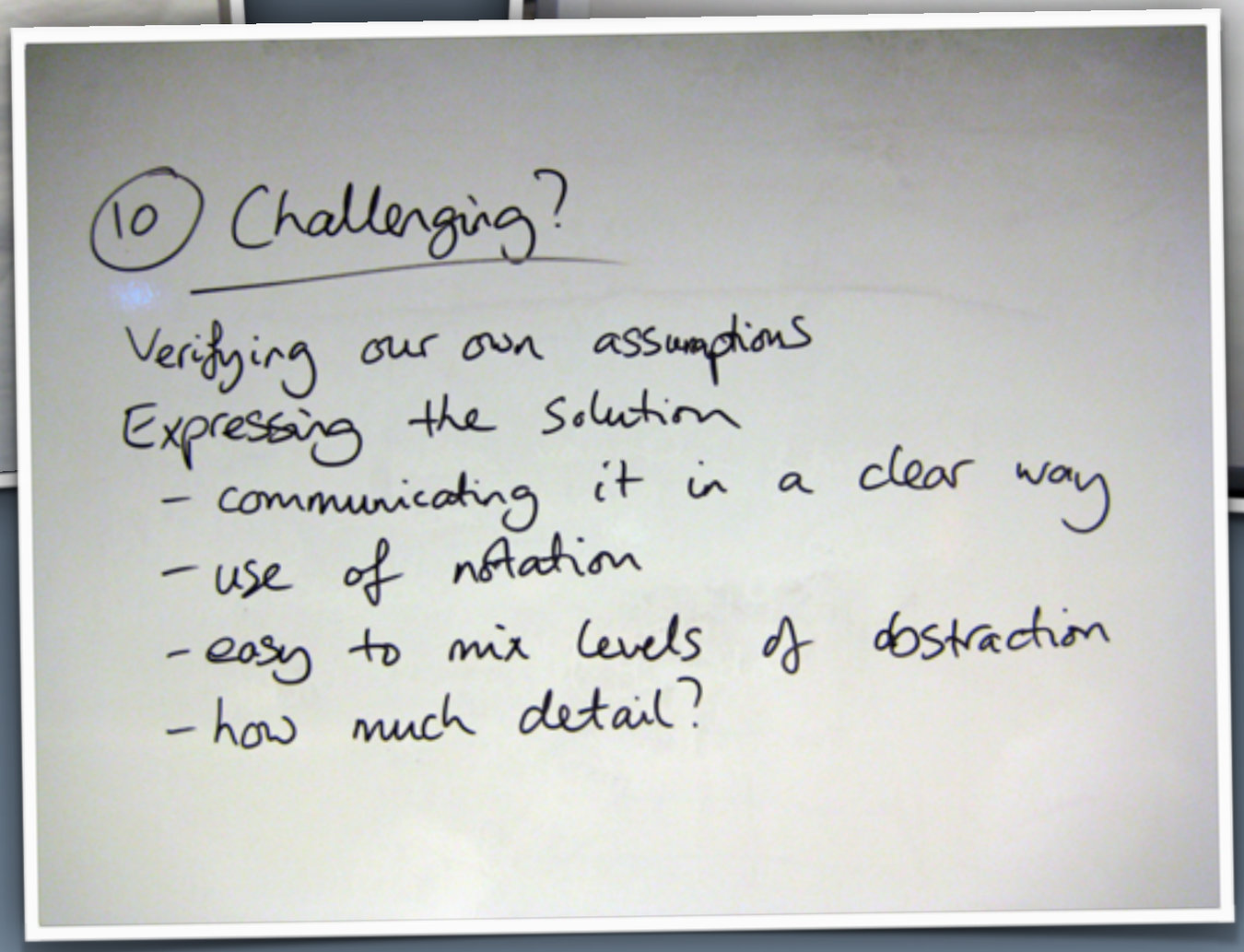
Homeless Old C# Object (HOCO)



Choose your own adventure







What's been challenging about the exercise?

Code

Would we

code

it that way?

Did we

code

it that way?

The code is the

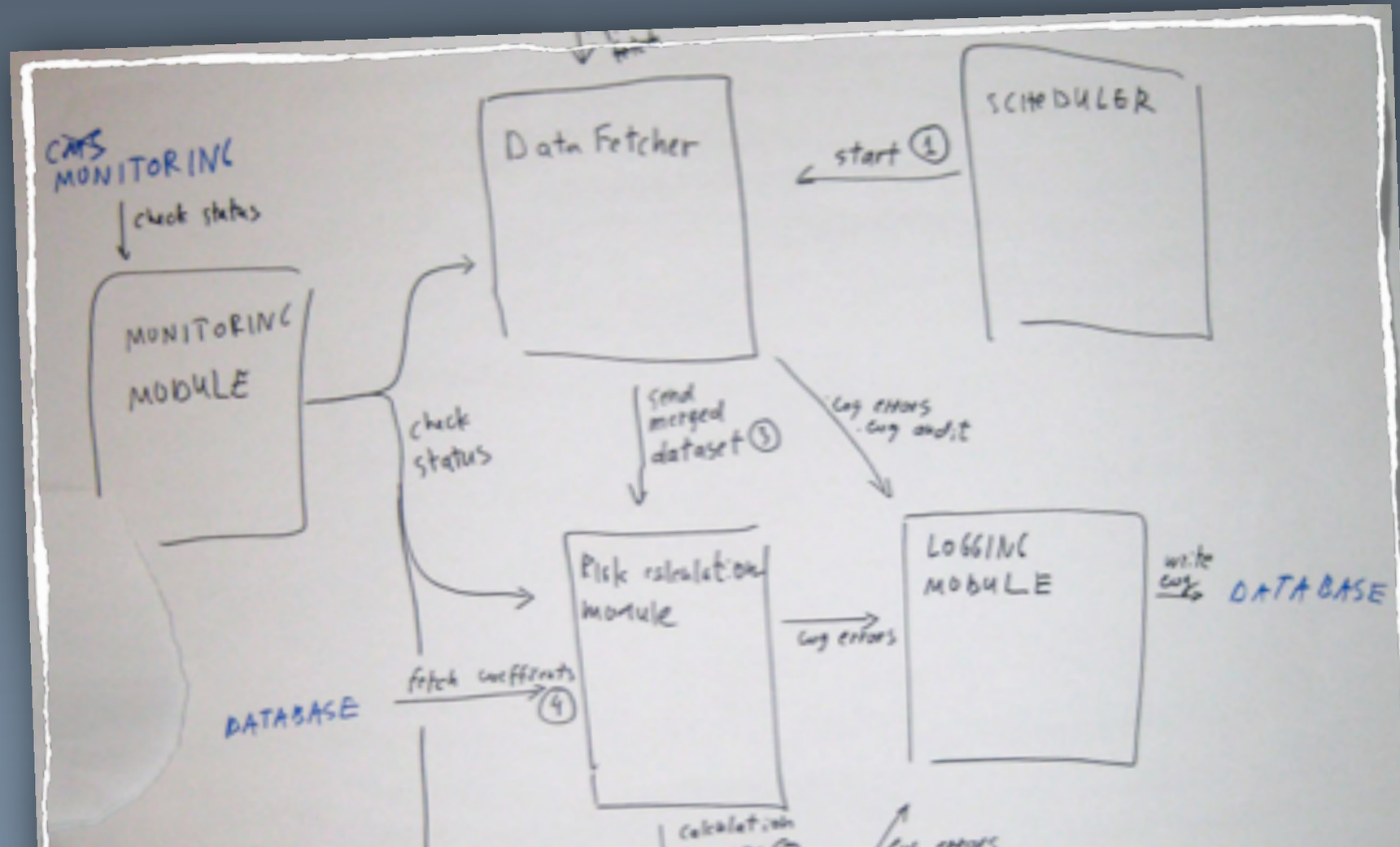
embodiment

of the architecture

Abstraction

is about reducing detail

rather than creating a different representation



Abstractions help us

reason about

a big and/or complex software system

Does your code reflect the

abstractions

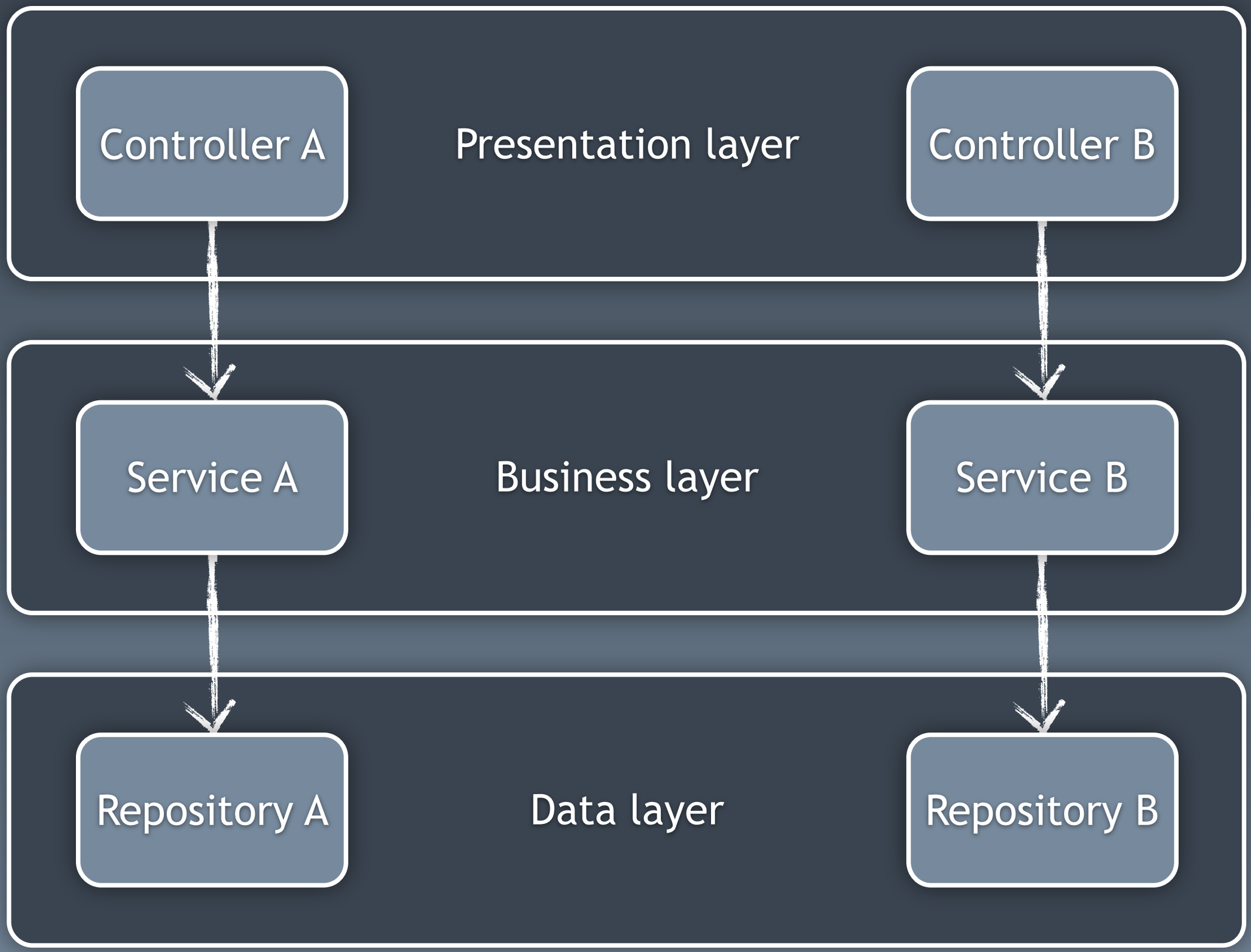
that you think about?

We often think

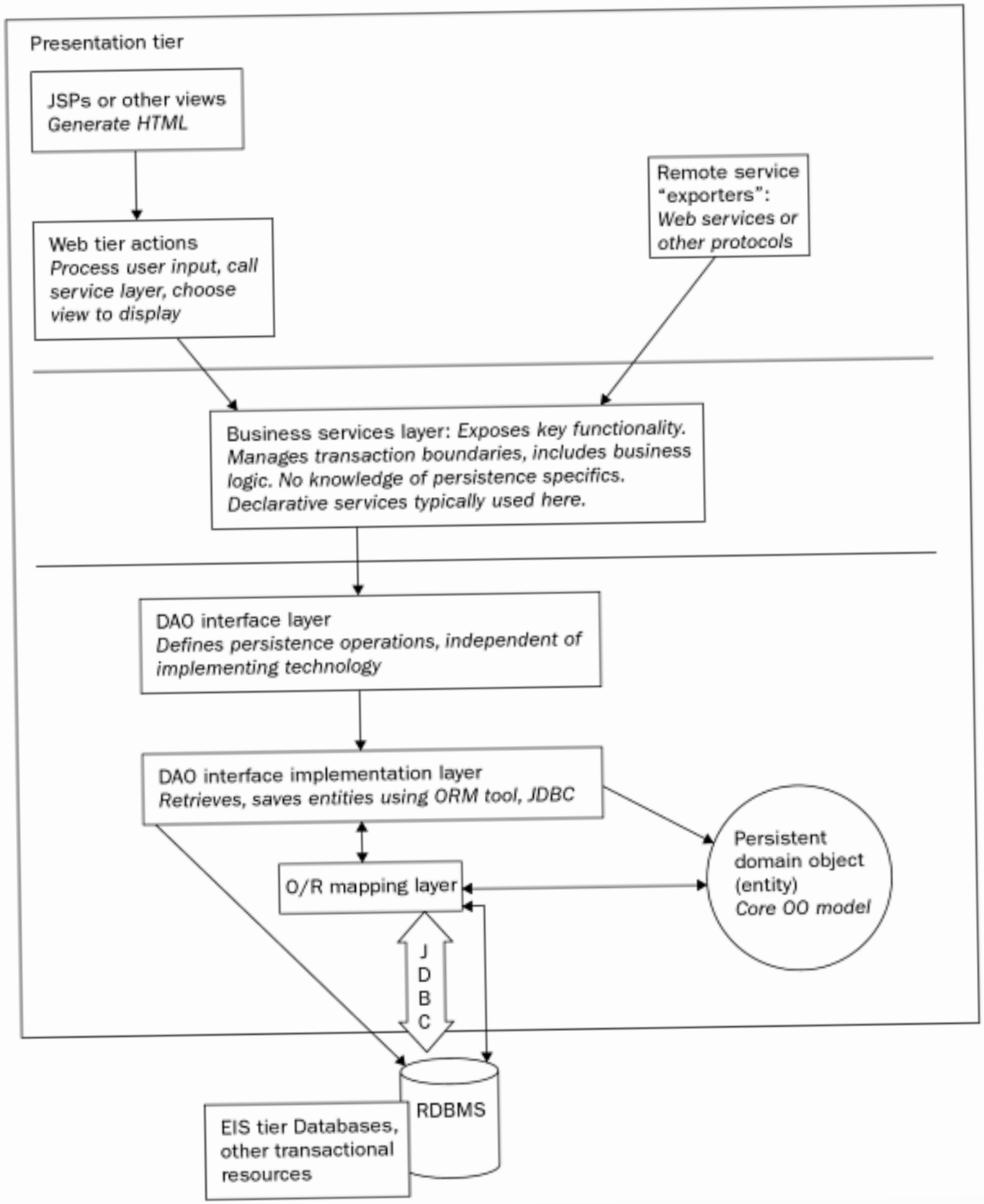
in components

but write classes

(usually in layers)



A layered architecture


Presentation tier
JSPs or other views
Generate HTML
Web tier actions
Process user input, call service layer, choose view to display
Remote service "exporters":
Web services or other protocols
Business services layer: Exposes key functionality. Manages transaction boundaries, includes business logic. No knowledge of persistence specifics. Declarative services typically used here.
DAO interface layer
Defines persistence operations, independent of implementing technology
DAO interface implementation layer
Retrieves, saves entities using ORM tool, JDBC
O/R mapping layer
Persistent domain object (entity)
Core OO model
J
D
B
C
RDBMS
EIS tier Databases, other transactional resources

Let's summarize each layer and its responsibilities, beginning closest to the database or other enterprise resources:

- **Presentation layer:** This is most likely to be a web tier. This layer should be as thin as possible. It should be possible to have alternative presentation layers — such as a web tier or remote web services facade — on a single, well-designed middle tier.
- **Business services layer:** This is responsible for transactional boundaries and providing an entry point for operations on the system as a whole. This layer should have no knowledge of presentation concerns, and should be reusable.
- **DAO interface layer:** This is a layer of interfaces *independent of any data access technology* that is used to find and persist persistent objects. This layer effectively consists of *Strategy* interfaces for the Business services layer. This layer should not contain business logic. Implementations of these interfaces will normally use an O/R mapping technology or Spring's JDBC abstraction.
- **Persistent domain objects:** These model real objects or concepts such as a bank account.
- **Databases and legacy systems:** By far the most common case is a single RDBMS. However, there may be multiple databases, or a mix of databases and other transactional or non-transactional legacy systems or other enterprise resources. The same fundamental architecture is applicable in either case. This is often referred to as the *EIS* (*Enterprise Information System*) tier.

In a J2EE application, all layers except the EIS tier will run in the application server or web container. Domain objects will typically be passed up to the presentation layer, which will display data they contain, *but not modify them*, which will occur only within the transactional boundaries defined by the business services layer. Thus there is no need for distinct Transfer Objects, as used in traditional J2EE architecture.

In the following sections we'll discuss each of these layers in turn, beginning closest to the database.

Spring aims to decouple architectural layers, so that each layer can be modified as far as possible without impacting other layers. No layer is aware of the concerns of the layer above; as far as possible, dependency is purely on the layer immediately below. Dependency between layers is normally in the form of interfaces, ensuring that coupling is as loose as possible.

Java - Eclipse Platform - /Users/Shared/Projects

Java | Java EE

Package Explorer | JUnit

- springapp
  - src
    - springapp.domain
      - Product.java
    - springapp.repository
      - JdbcProductDao.java
      - ProductDao.java
    - springapp.service
      - PriceIncrease.java
      - PriceIncreaseValidator.java
      - ProductManager.java
      - SimpleProductManager.java
    - springapp.web
  - test
  - JRE System Library [JVM 1.5.0 (MacOS X Default)]
  - Referenced Libraries
  - db
  - war
    - WEB-INF
      - classes
        - springapp
        - jdbc.properties
        - messages.properties
      - jsp
        - hello.jsp
        - include.jsp

1 Answer

active oldest votes

8

If you are using maven, it's best to follow the standard maven project layout. You can get maven to generate this structure for you by doing,

```
mvn archetype:generate
```

and select spring-mvc-jpa-archetype from the list of choices

This will give you a package structure like,

```
├── pom.xml
└── src
    ├── main
    │   ├── java
    │   │   └── mygroup
    │   │       ├── controller
    │   │       │   ├── HomeController.java
    │   │       │   └── PersonController.java
    │   │       ├── dao
    │   │       │   └── PersonDao.java
    │   │       └── model
    │   │           └── Person.java
    │   ├── resources
    │   │   ├── db.properties
    │   │   ├── log4j.xml
    │   │   └── META-INF
    │   │       └── persistence.xml
    │   └── webapp
    │       ├── index.html
    │       ├── META-INF
    │       │   ├── context.xml
    │       │   └── MANIFEST.MF
    │       ├── resources
    │       │   └── css
    │       │       └── screen.css
    │       └── WEB-INF
    │           ├── spring
    │           │   ├── app
    │           │   │   ├── controllers.xml
    │           │   │   └── servlet-context.xml
    │           │   ├── db.xml
    │           │   └── root-context.xml
    │           ├── views
```


Related

2 How do I use a custom authentication mechanism for a Java web application with Spring Security?

0 Is there a sample web project of integrating Spring-MVC and Spring-Data-JPA?

1 Reusing DAOs in another web application

0 Java Web Application Warming

3 Is a parallel Spring-MVC application possible with a non-spring web app?

2 Adding a web interface (Spring MVC) to existing Java application

0 package strucutre for Spring MVC project with multiple sub projects using maven

0 How to add a setup

MvcMovie - Microsoft Visual Studio Expres...

Quick Launch (Ctrl+Q)

FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST WINDOW HELP

Internet Explorer | Debug

HomeController.cs

MvcMovie.Controllers.HomeCont | Index()

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace MvcMovie.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            ViewBag.Message = "Modify this templ

            return View();
        }

        public ActionResult About()
        {
            ViewBag.Message = "Your app descript

            return View();
        }
```

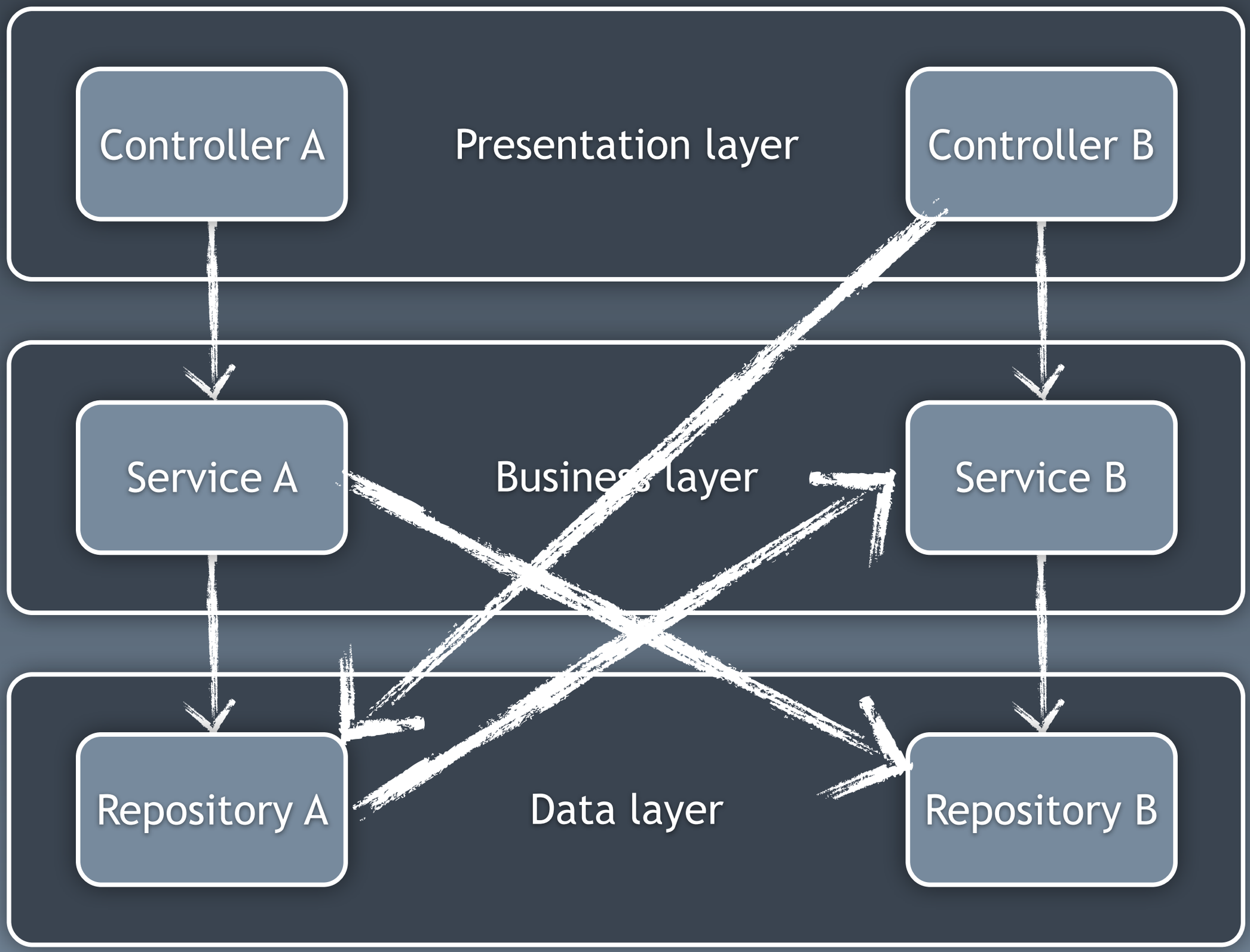
100 %

Solution Explorer

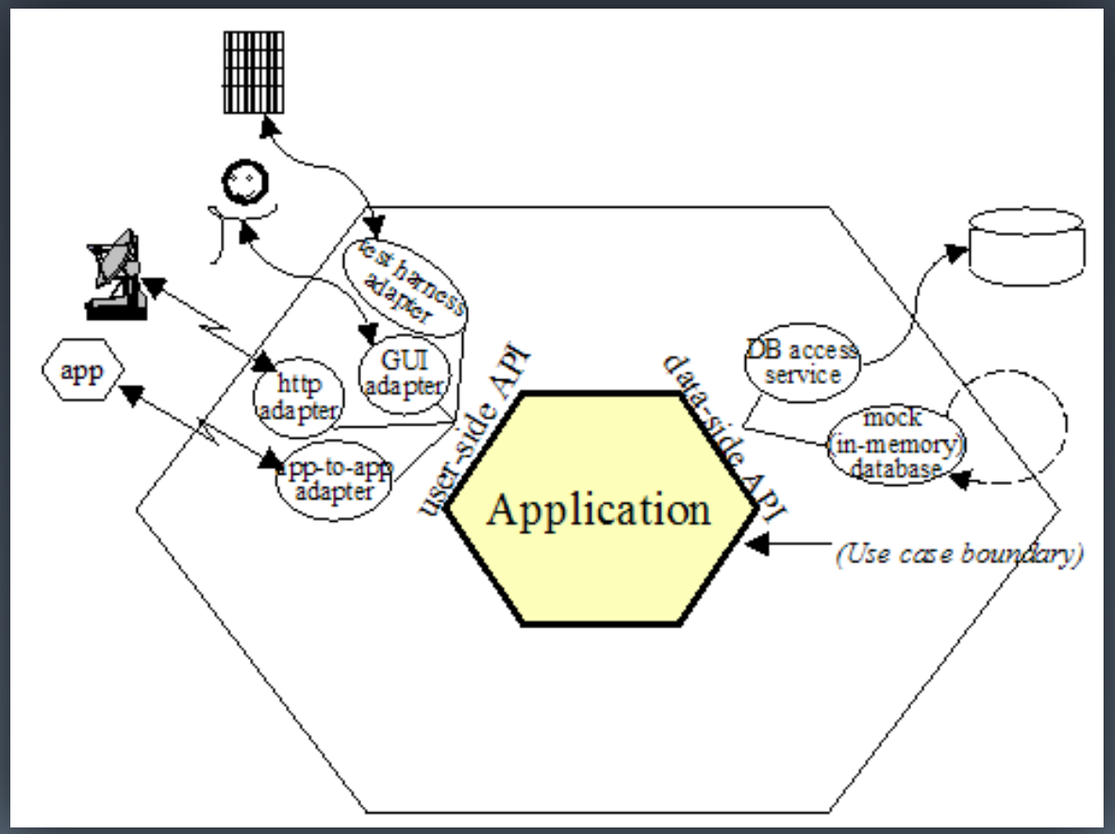
Search Solution Explorer (Ctrl+;)

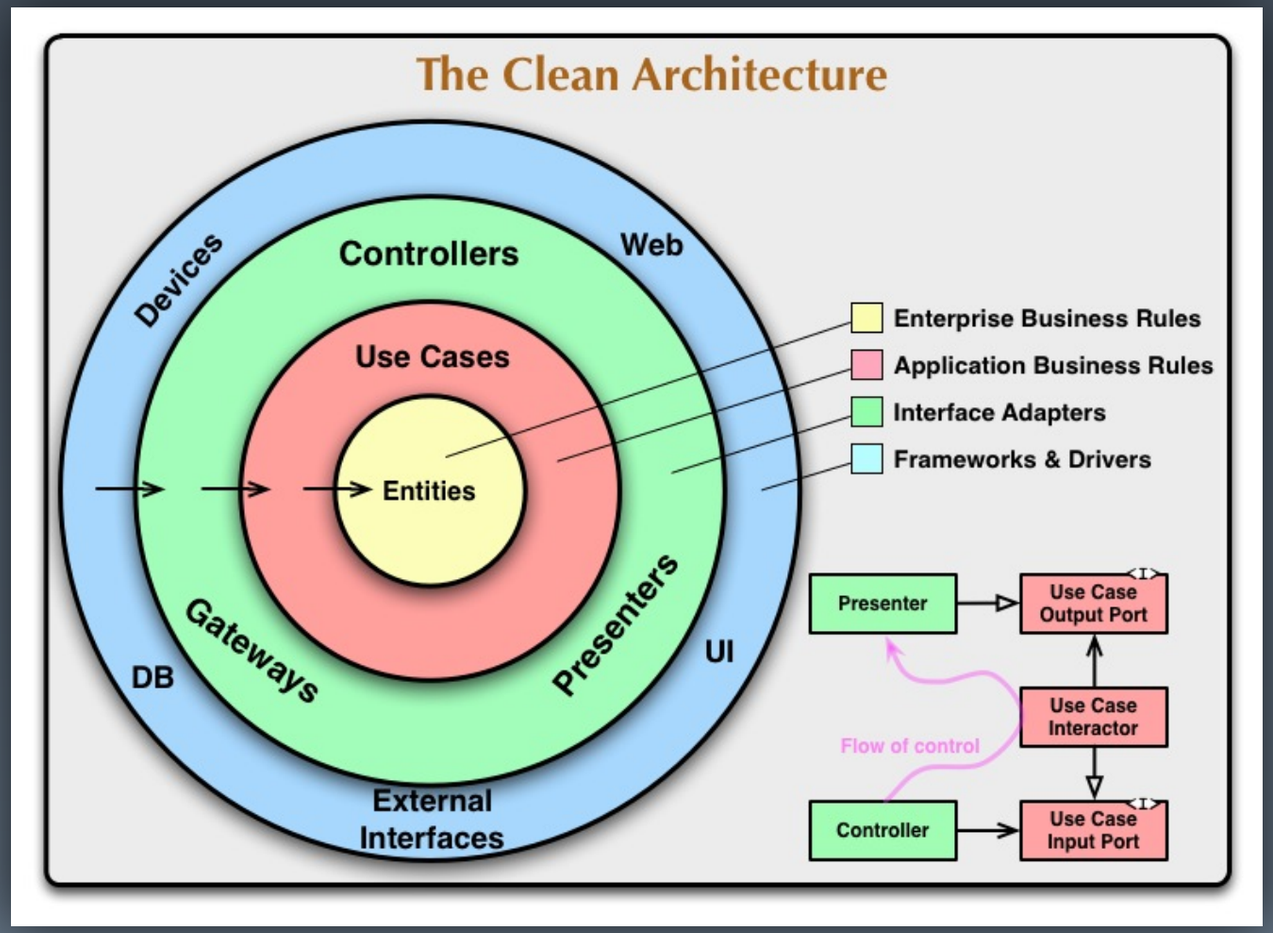
- Solution 'MvcMovie' (1 project)
  - MvcMovie
    - Properties
    - References
    - App_Data
    - App_Start
    - Content
    - Controllers
    - Filters
    - Images
    - Models
    - Scripts
    - Views
    - favicon.ico
    - Global.asax
    - packages.config
    - Web.config

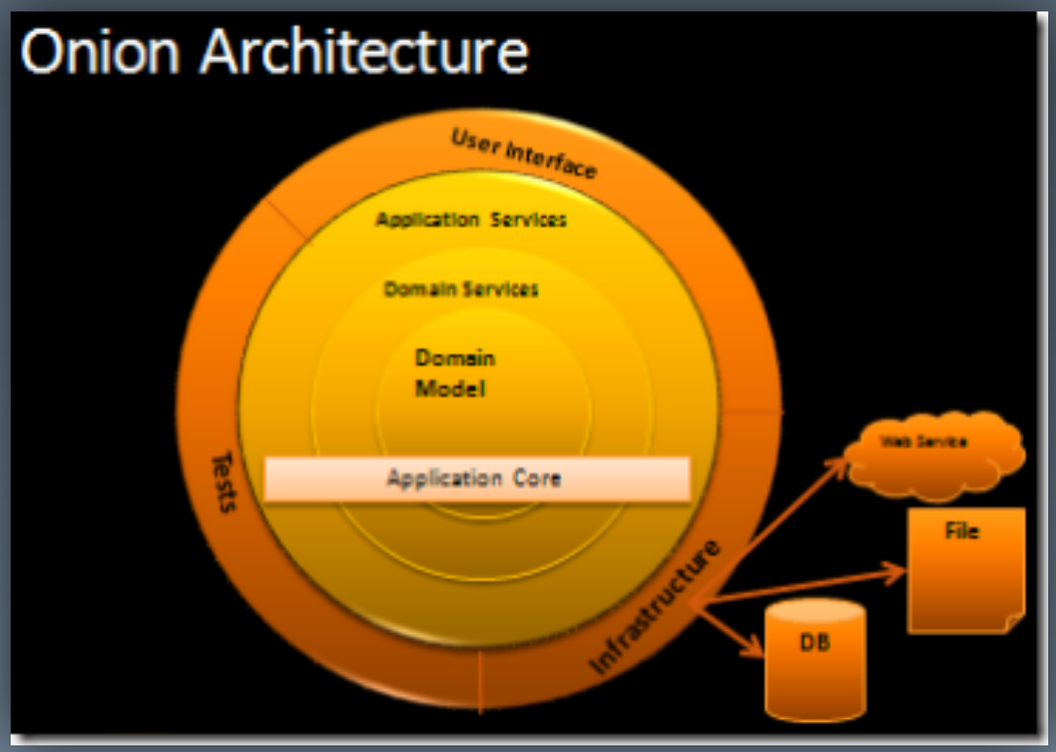
Ready | Ln 1 | Col 1 | Ch 1 | INS



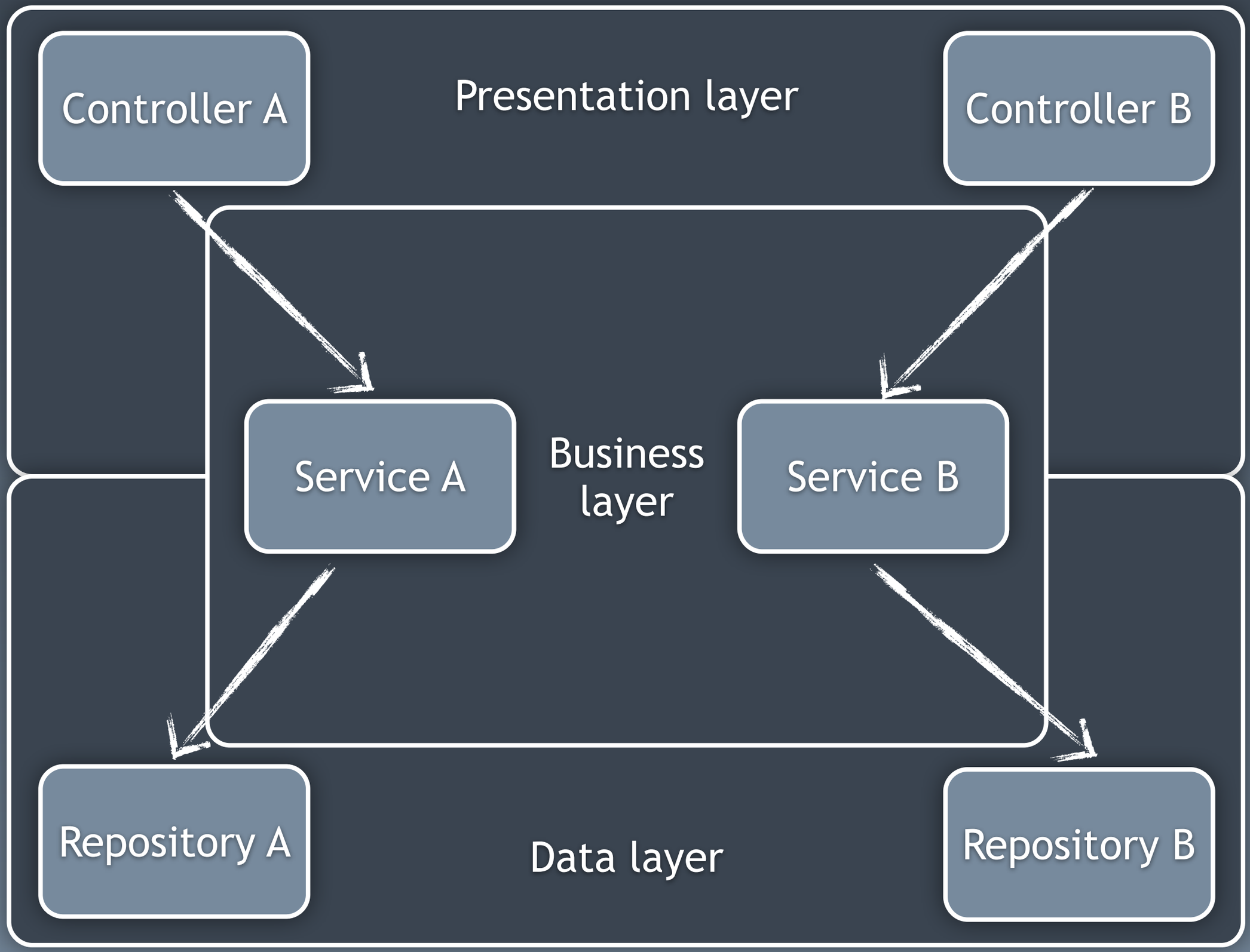
A typical layered architecture :-)







Hexagons and onions



This is still a layered architecture

Should layers be

considered

harmful?

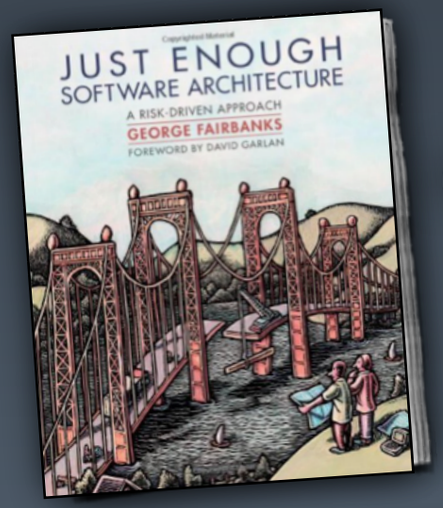
Are layers significant

structural elements

or just an

implementation detail?

Organisation of code vs the architectural views



“the model-code gap”

Model-code gap. Your architecture models and your source code will not show the same things. The difference between them is the *model-code gap*. Your architecture models include some abstract concepts, like components, that your programming language does not, but could. Beyond that, architecture models include intensional elements, like design decisions and constraints, that cannot be expressed in procedural source code at all.

Consequently, the relationship between the architecture model and source code is complicated. It is mostly a refinement relationship, where the extensional elements in the architecture model are refined into extensional elements in source code. This is shown in Figure 10.3. However, intensional elements are not refined into corresponding elements in source code.

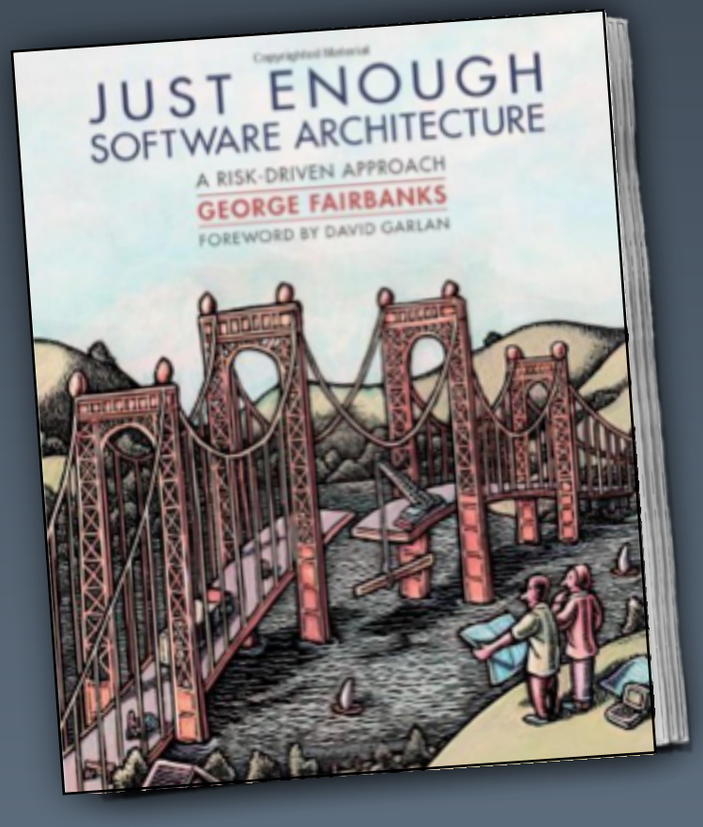
Upon learning about the model-code gap, your first instinct may be to avoid it. But reflecting on the origins of the gap gives little hope of a general solution in the short term: architecture models help you reason about complexity and scale because they are abstract and intensional; source code executes on machines because it is concrete and extensional.

Software architecture vs code

	Business Model	Domain Model	Design Model		Code Model
			Boundary Model	Internals Model	
Bosch			System context	Component design	Code
Cheesman & Daniels		Business concept	Type specs	Component architecture	Code
D'Souza (MAp)	Business architec-ture	Domain	Blackbox	Whitebox	Code
Software Engineering Institute (SEI)			Requirements	Architecture	Code
Jackson		Domain	Domain + machine	Machine	
RUP	Business modeling	Business modeling	Requirements	Analysis & design	Code

Merge

the code
and the model?



“architecturally-evident coding style”

(subclassing, naming conventions, module dependencies, package structure, …)

It’s 2014, why can’t we auto-generate a useful set of software architecture diagrams from code?!

Diagramming tools see

packages

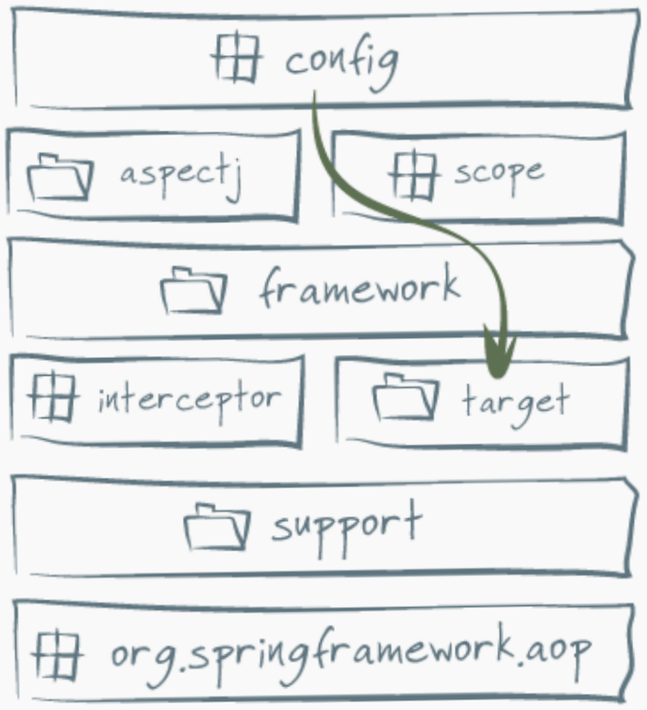
and classes

rather than components

structure101

Products Resources Download Pricing About Blog

Architecture



Diagrams & rules

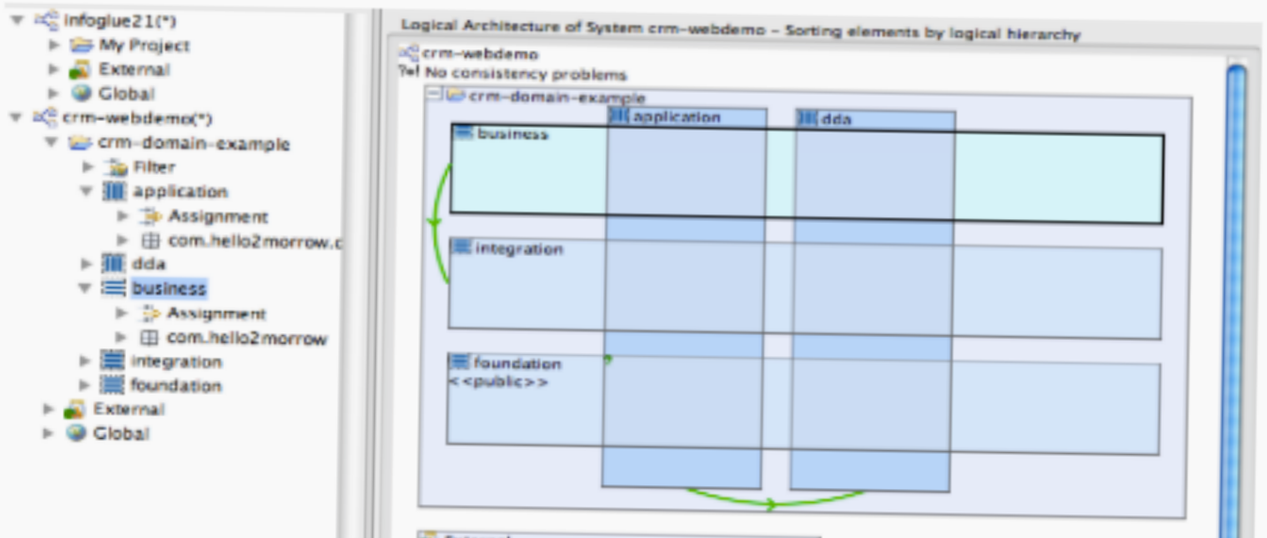
Define your desired architecture with Structure101's semantically rich, intuitive, layered block architecture diagrams, present it to your developers through our IDE plugin and enforce it by integrating into your build.

Design

Define and Enforce Architecture

Sonargraph-Architect is designed to simplify software architecture and dependency management. You begin by defining the intended logical architecture of your system and map it to your code. A logical architecture is a set of rules designating allowed and forbidden dependencies in your code. Typical rules would be "the UI layer is not allowed to directly access the DAO layer" or "the UI layer cannot use JDBC directly".

Once defined, our IDE plugins (for Eclipse or IntelliJ IDEA) check every code change for rule compliance. Violating code lines will be marked with error markers so that developers are able to fix rule violations even before they commit their changes to the version control system. If you are not using Eclipse or IntelliJ, you can always use Sonargraph-Architect standalone for immediate rule checking. Moreover Sonargraph-Architect integrates with ANT and Maven so that architecture rules can also be enforced in your build process.



Structural over-complexity

100

% Tangled

Unstructured

Levelized Structure Map (LSM)

icefaces-1_8_1

com

icesoft

net

messaging

faces

async

facelets

renderkit

webapp

context

component

util

application

el

env

jasper

util

org.apache.tools

ant.taskdefs

ant

load..

input

listener

help

filter

util

types

*

UTILITIES

$root

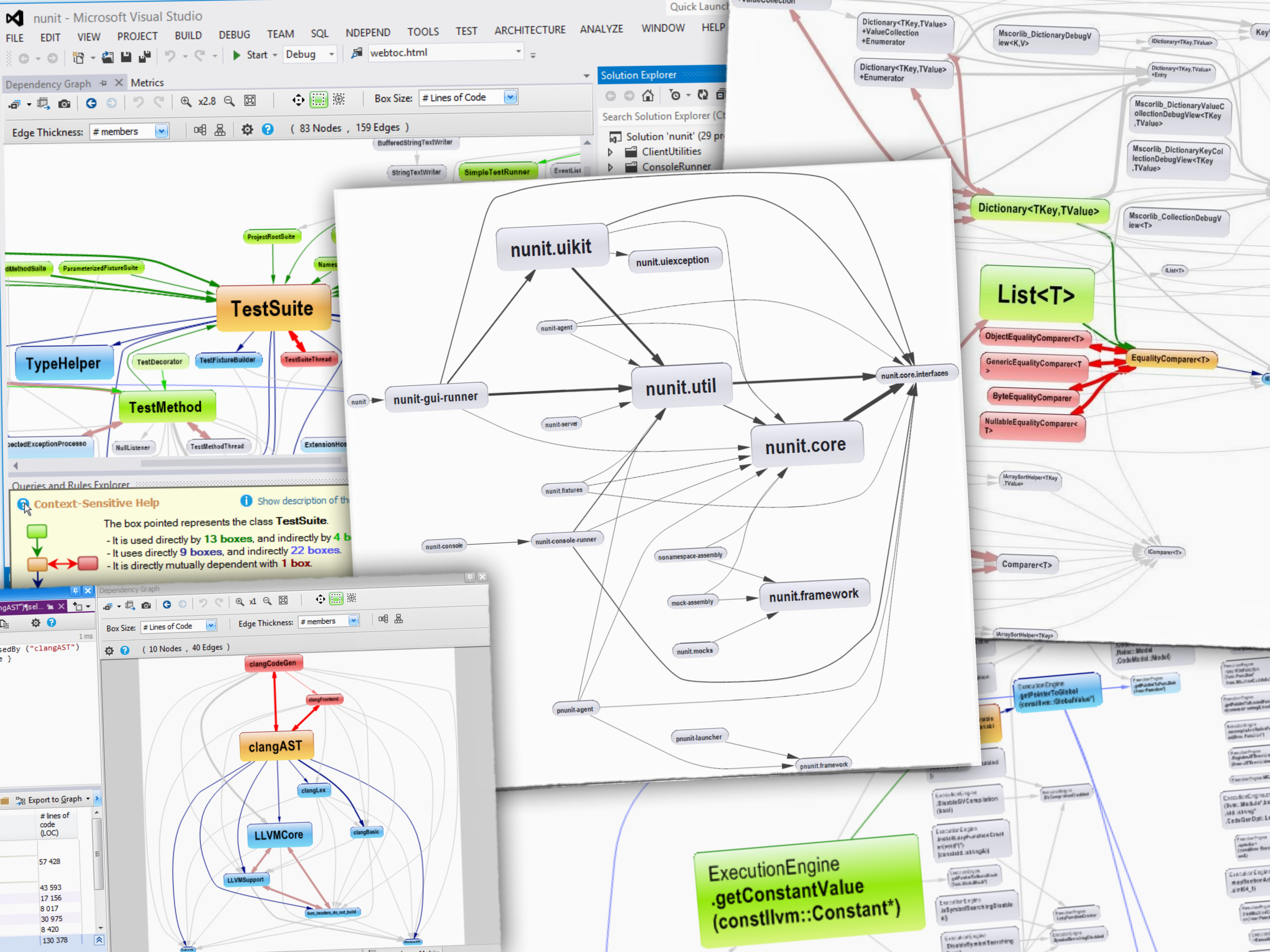
org.apache.tools

ant.taskde...

ant

cvslib	1	
compilers	2	
rmic	3	
condition	4	
email	5	
*	6	
loader	7	
listener	8	
helper	9	
input	10	
filters	11	
util	12	
types	13	
*	14	
UTILITIES	15	

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	.					4									
2		.				10									
3			.			47									
4				.		3		23				6	1	2	
5	50	61	41	6	.		9							33	
6						.	.								
7						4		.				1	2		
8						7			.				10		
9						25				.	16	48			
10	3	13	5	3	1	284	3	2	18	2	33	.	52	64	
11	5	297	67	4	8	1378					88	96	.	49	
12	66	79	59	69	75	2800	5	18	300	18	60	175	585	.	
13					14	123		10					10		
14															
15															



nunit - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL NDEPEND TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Start Debug webtoc.html
Dependency Graph Metrics
Box Size: # Lines of Code
Edge Thickness: # members
(83 Nodes , 159 Edges)
Solution Explorer
Search Solution Explorer
Solution 'nunit'
ClientUtilities
ConsoleRunner
TestSuite
TypeHelper
TestMethod
TestDecorator
TestFixtureBuilder
TestSuiteThread
ProjectRootSuite
ParameterizedFixtureSuite
StringTextWriter
SimpleTestRunner
NullListener
TestMethodThread
Queries and Rules Explorer
Context-Sensitive Help
The box pointed represents the class TestSuite.
- It is used directly by 13 boxes, and indirectly by 4
- It uses directly 9 boxes, and indirectly 22 boxes.
- It is directly mutually dependent with 1 box.
nunit.uikit
nunit.uiexception
nunit-agent
nunit
nunit-gui-runner
nunit.util
nunit.core.interfaces
nunit-server
nunit.core
nunit.fixtures
nunit-console
nunit-console-runner
nonamespace-assembly
mock-assembly
nunit.framework
nunit.mocks
pnunit-agent
pnunit-launcher
pnunit.framework
Dictionary<TKey,TValue>
List<T>
ObjectEqualityComparer<T>
GenericEqualityComparer<T>
ByteEqualityComparer
NullableEqualityComparer<T>
EqualityComparer<T>
Comparer<T>
Dependency Graph
Box Size: # Lines of Code
Edge Thickness: # members
(10 Nodes , 40 Edges)
clangCodeGen
clangFrontend
clangAST
clangLex
LLVMCore
clangBasic
LLVMSupport
Export to Graph
lines of code (LOC)
57 428
43 593
17 156
8 017
30 975
8 420
130 378
ExecutionEngine
.getConstantValue
(constllvm::Constant*)

Do OO programming languages provide the wrong abstractions for building modern software systems?

The intersection of
software architecture
and code

Abstractions

on diagrams

should reflect the

code

A common set of abstractions

is more important than
a common notation

Software System

iner
tion server, standalone
e, file system, etc)

Container
(e.g. web server, application server, standalone application, browser, database, file system, etc)

Cont
(e.g. web server, applic
application, databa

Component

Component

Component

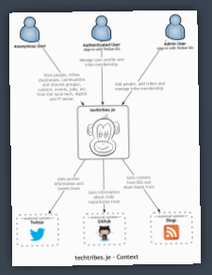
Class

Class

Class

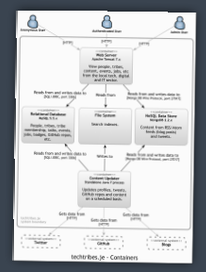
Agree on a simple set of abstractions that the whole team can use to communicate

The C4 model



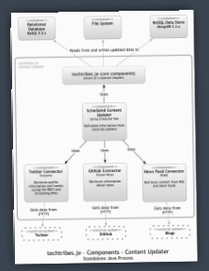
System Context

The system plus users and system dependencies



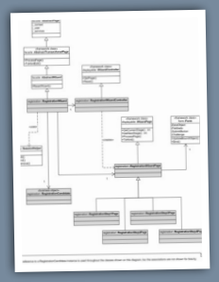
Containers

The overall shape of the architecture and technology choices



Components

Logical components and their interactions within a container

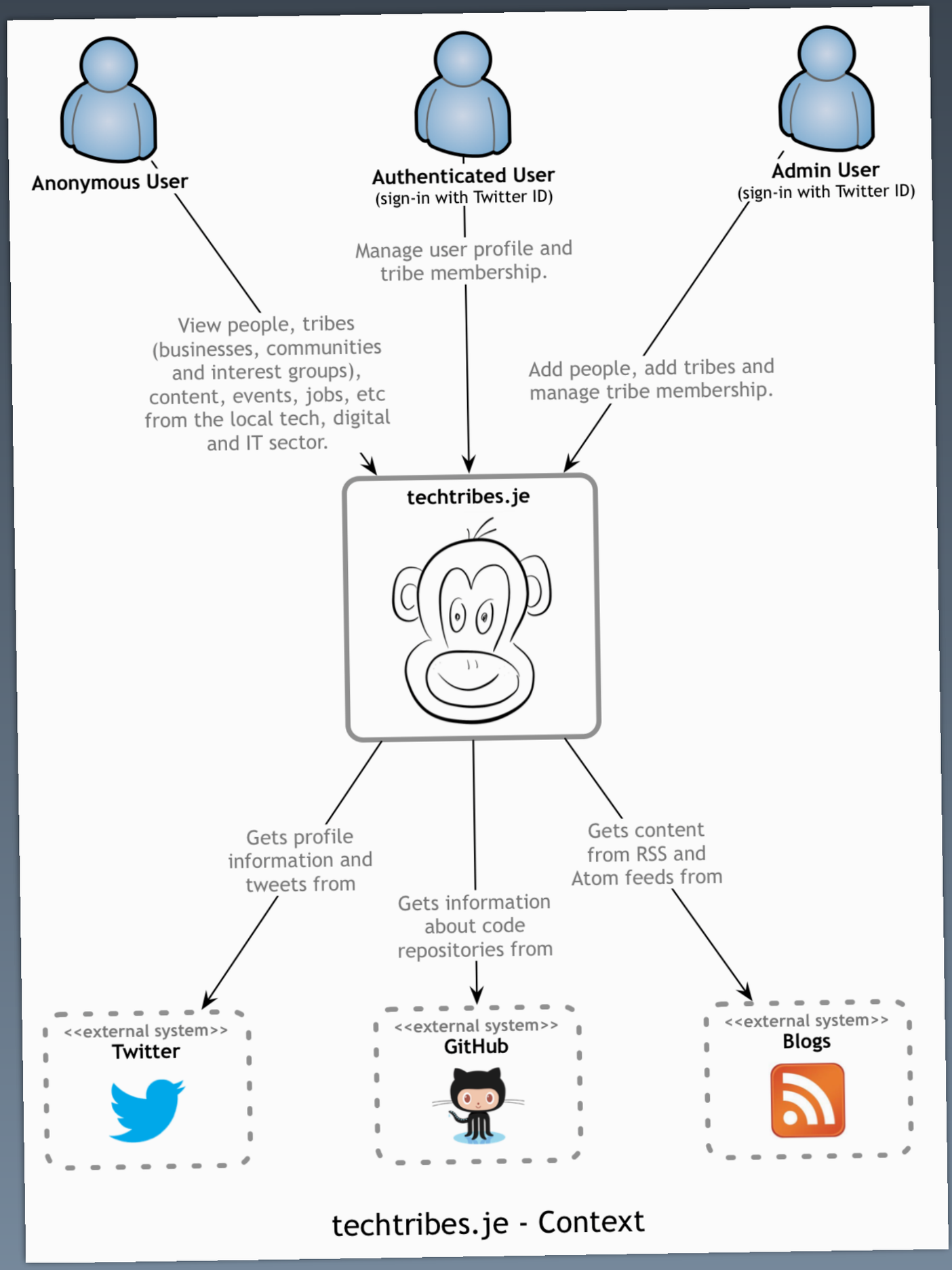


Classes

Component or pattern implementation details

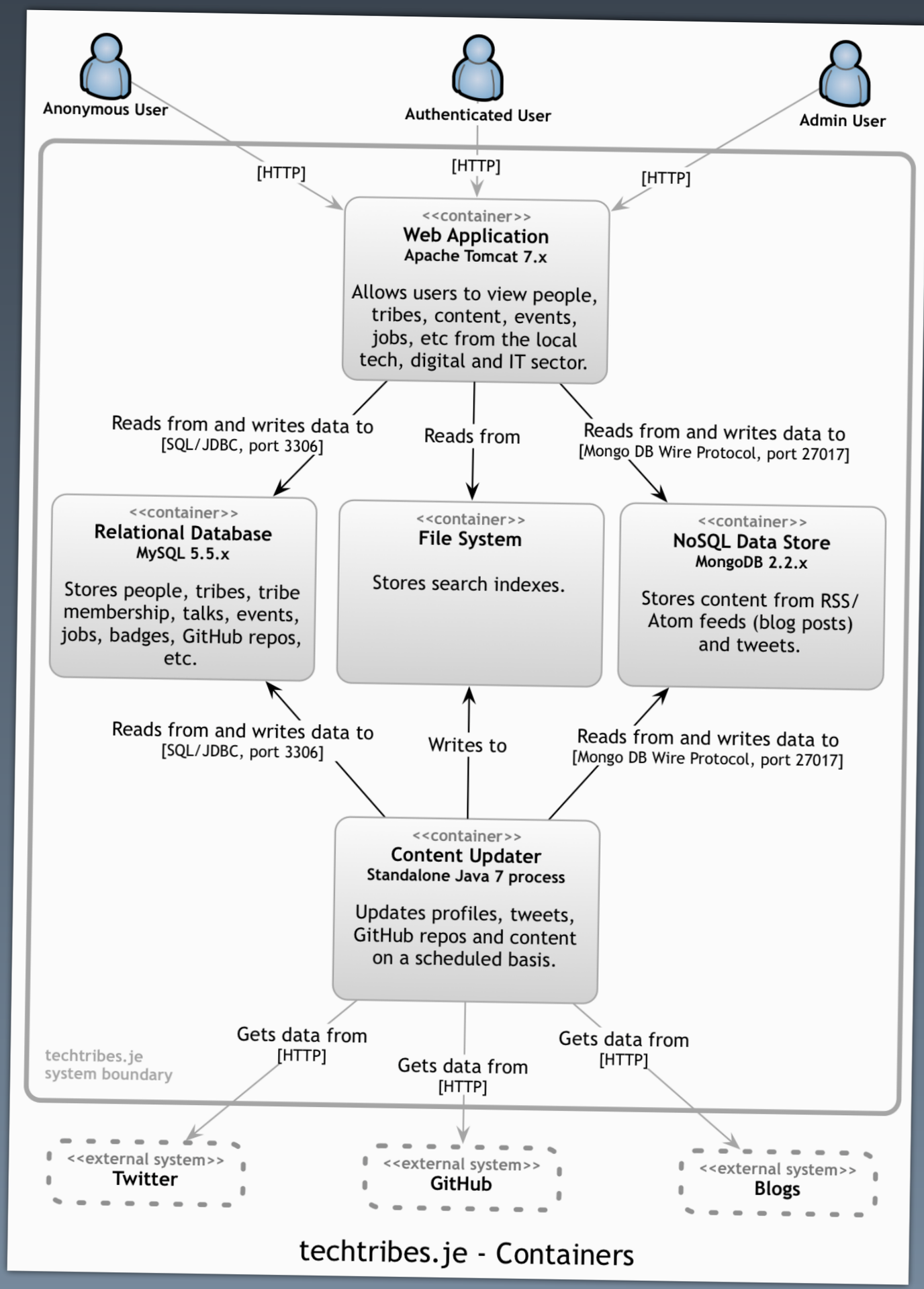
Context

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



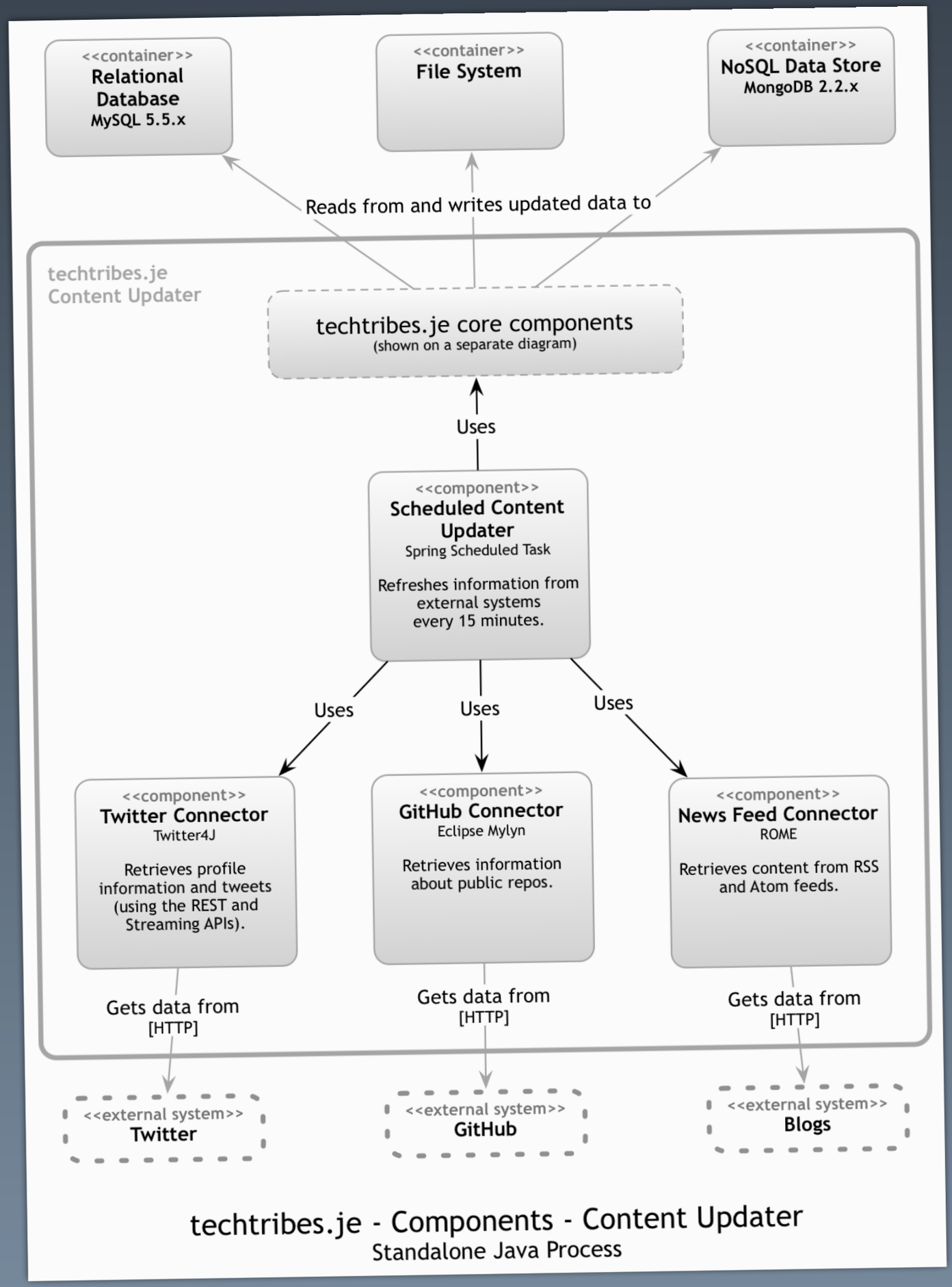
Containers

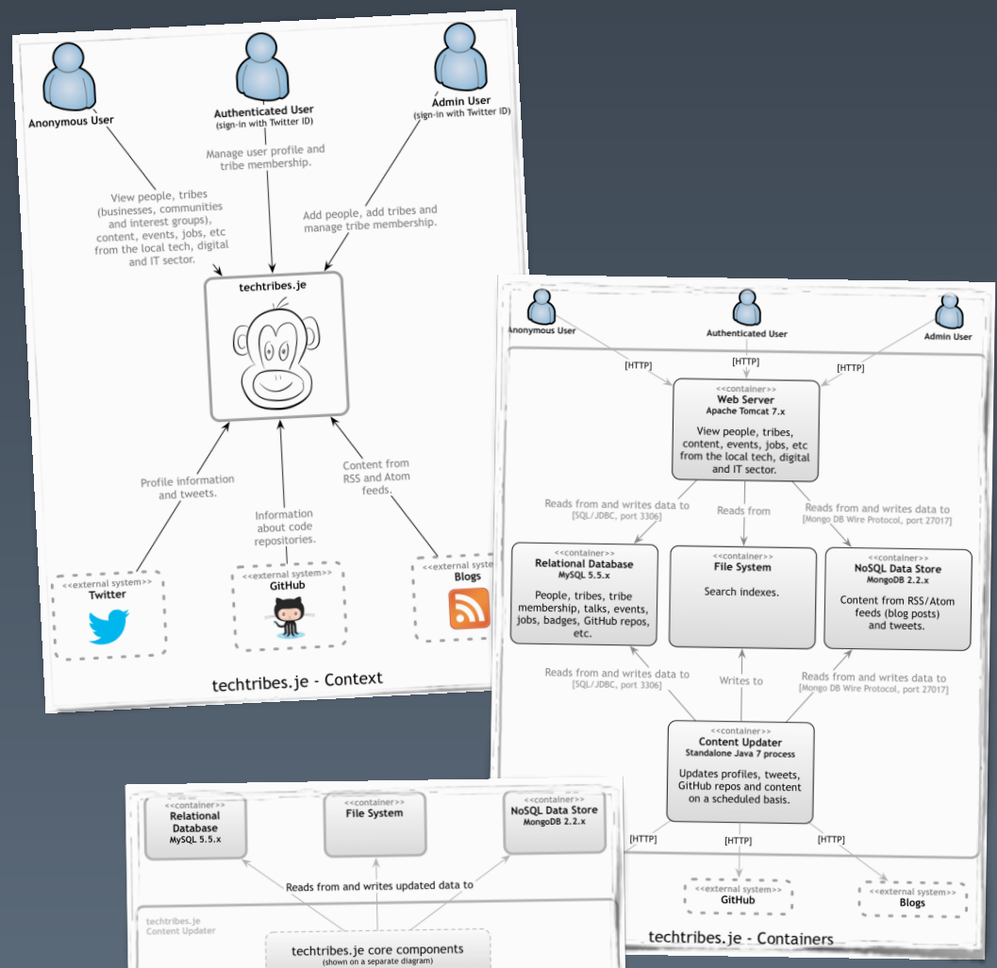
- What are the high-level technology decisions? (including responsibilities)
- How do containers communicate with one another?
- As a developer, where do I need to write code?

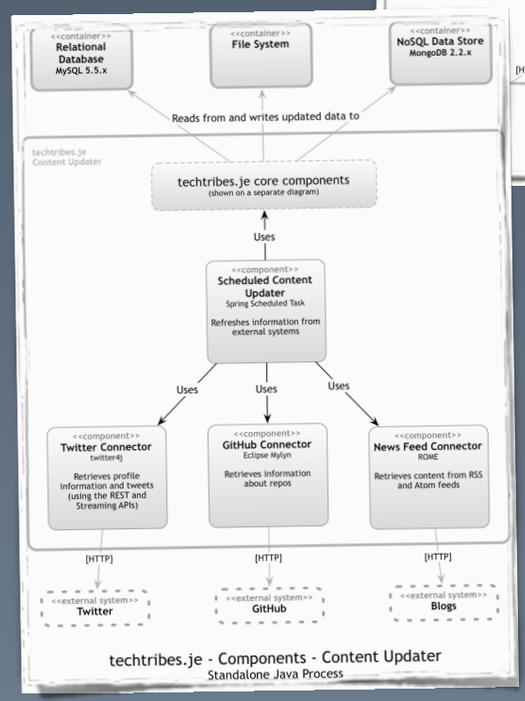


Components

- What components/ services is the container made up of?
- Are the technology choices and responsibilities clear?



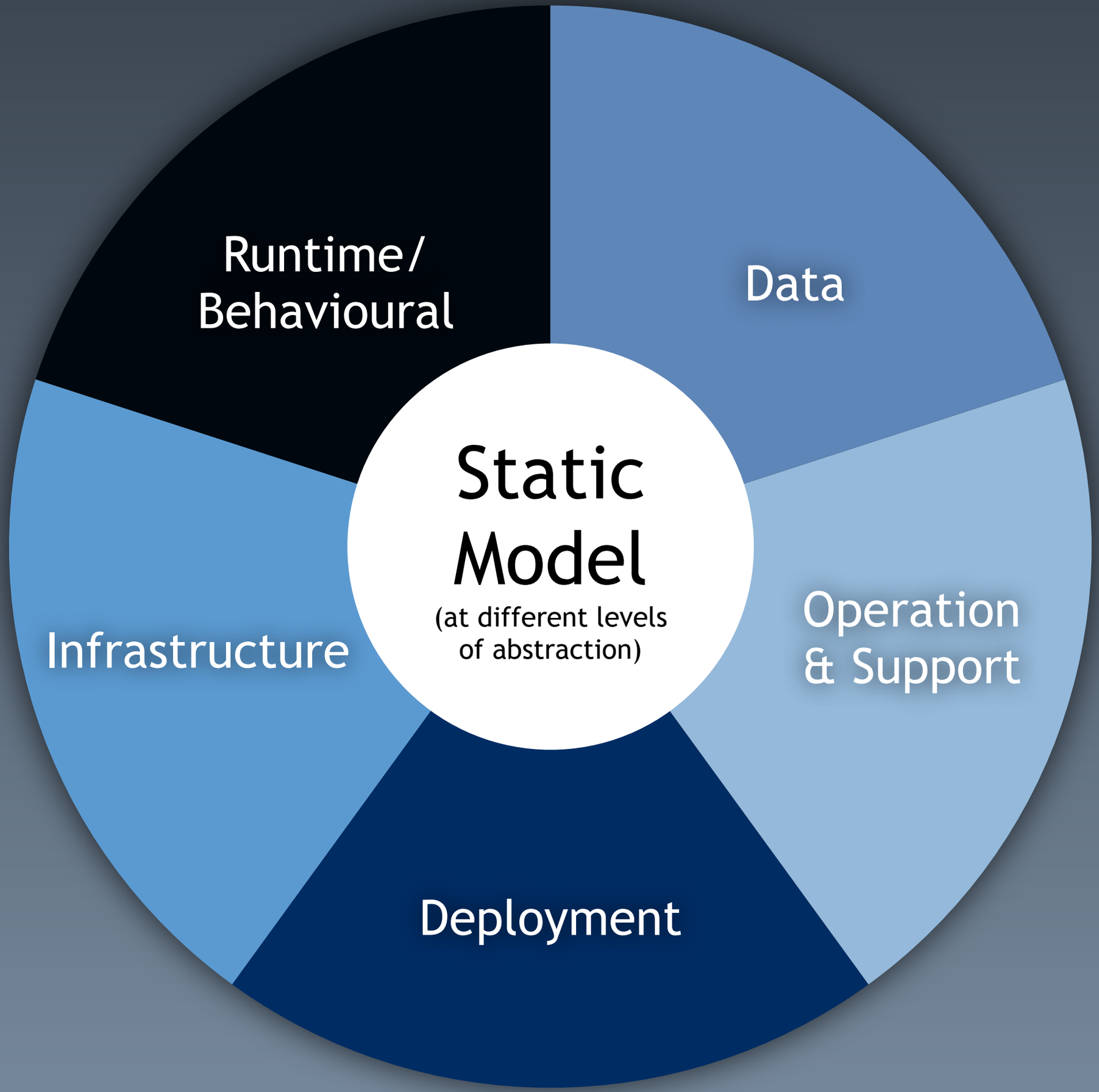




Sketches are maps

that help a team navigate a complex codebase

This isn't about creating a standard


Runtime/
Behavioural
Data
Static
Model
(at different levels
of abstraction)
Infrastructure
Operation
& Support
Deployment

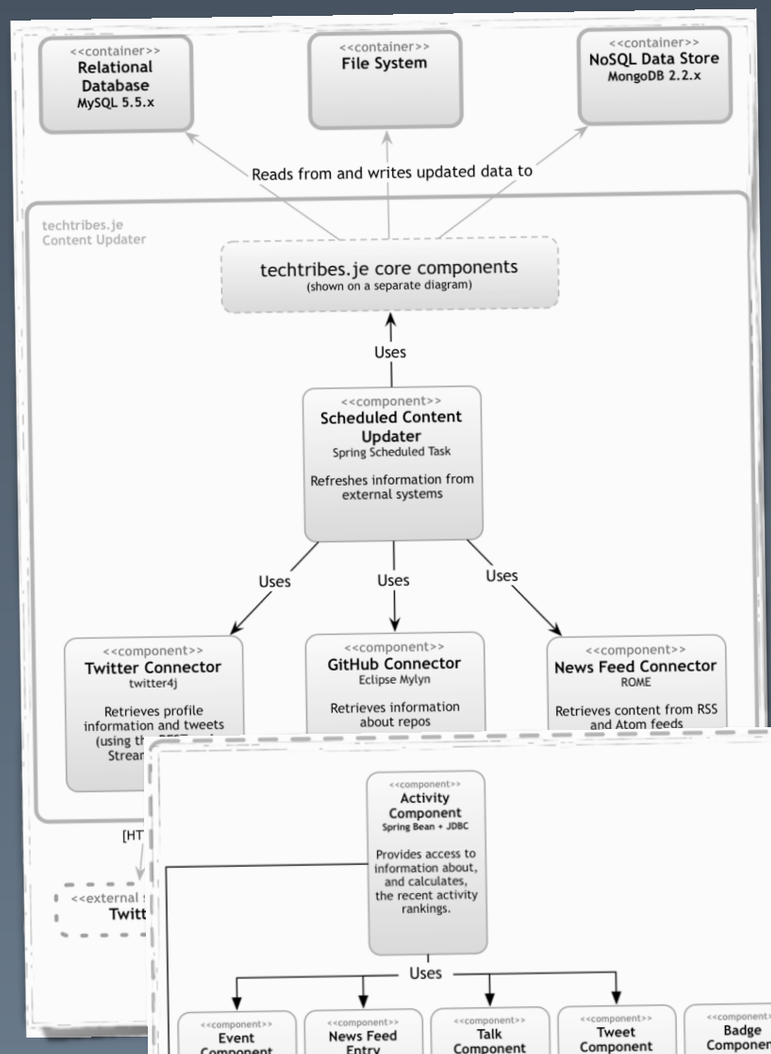
C4 is about the static structure of software, which is ultimately about code

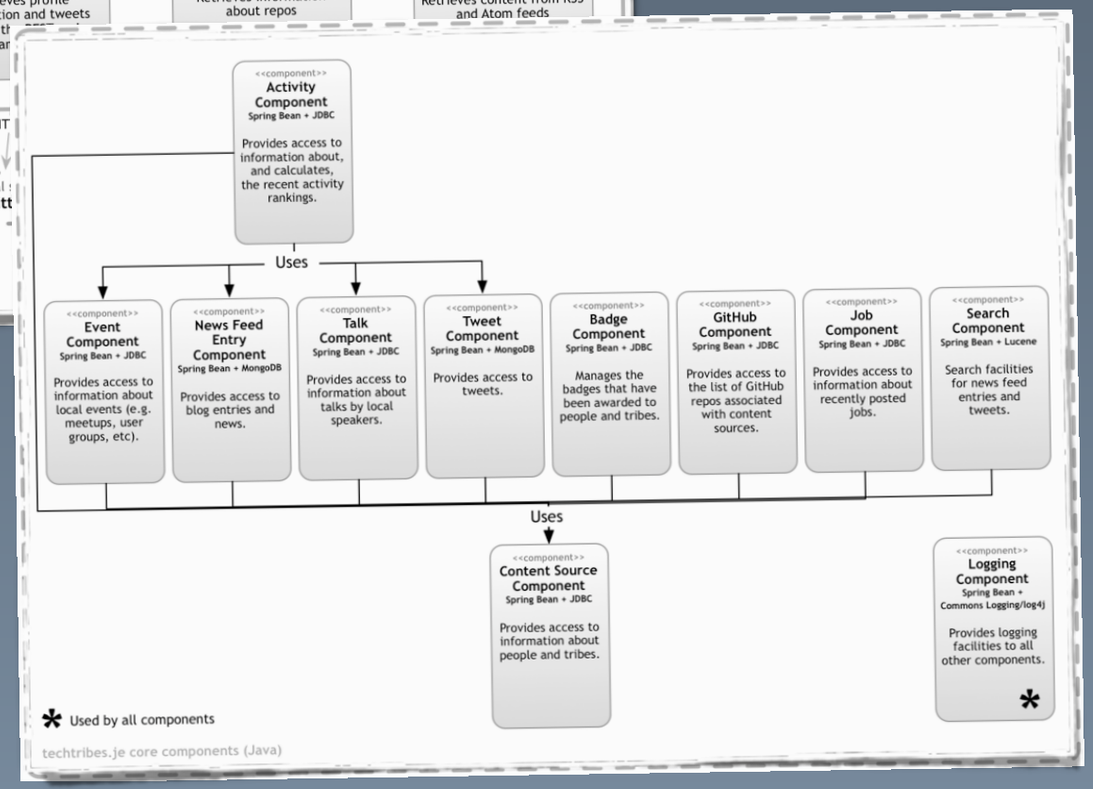
Software developers are the most important

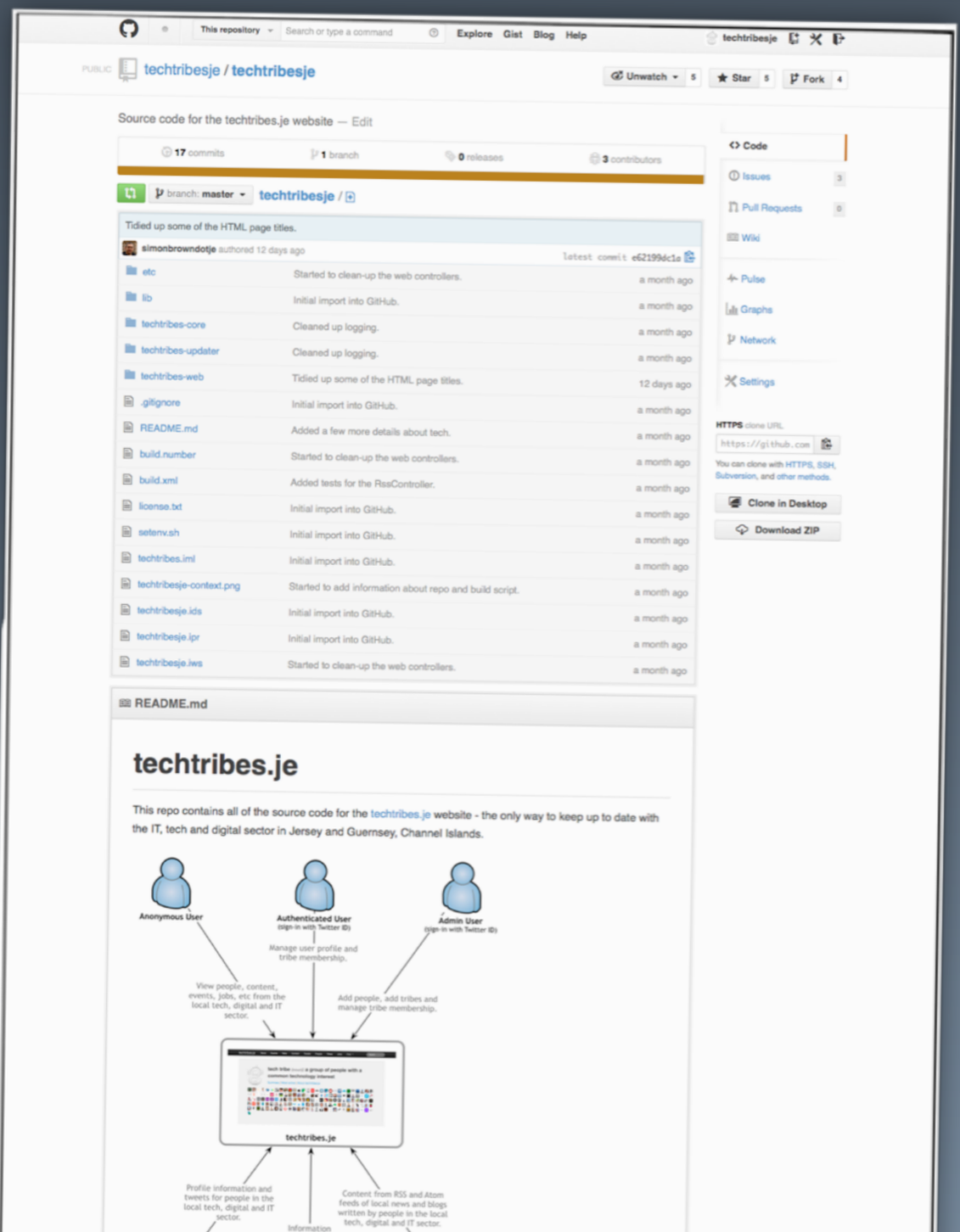
stakeholders

of software architecture

Does your code reflect the abstractions that appear on your software architecture diagrams?



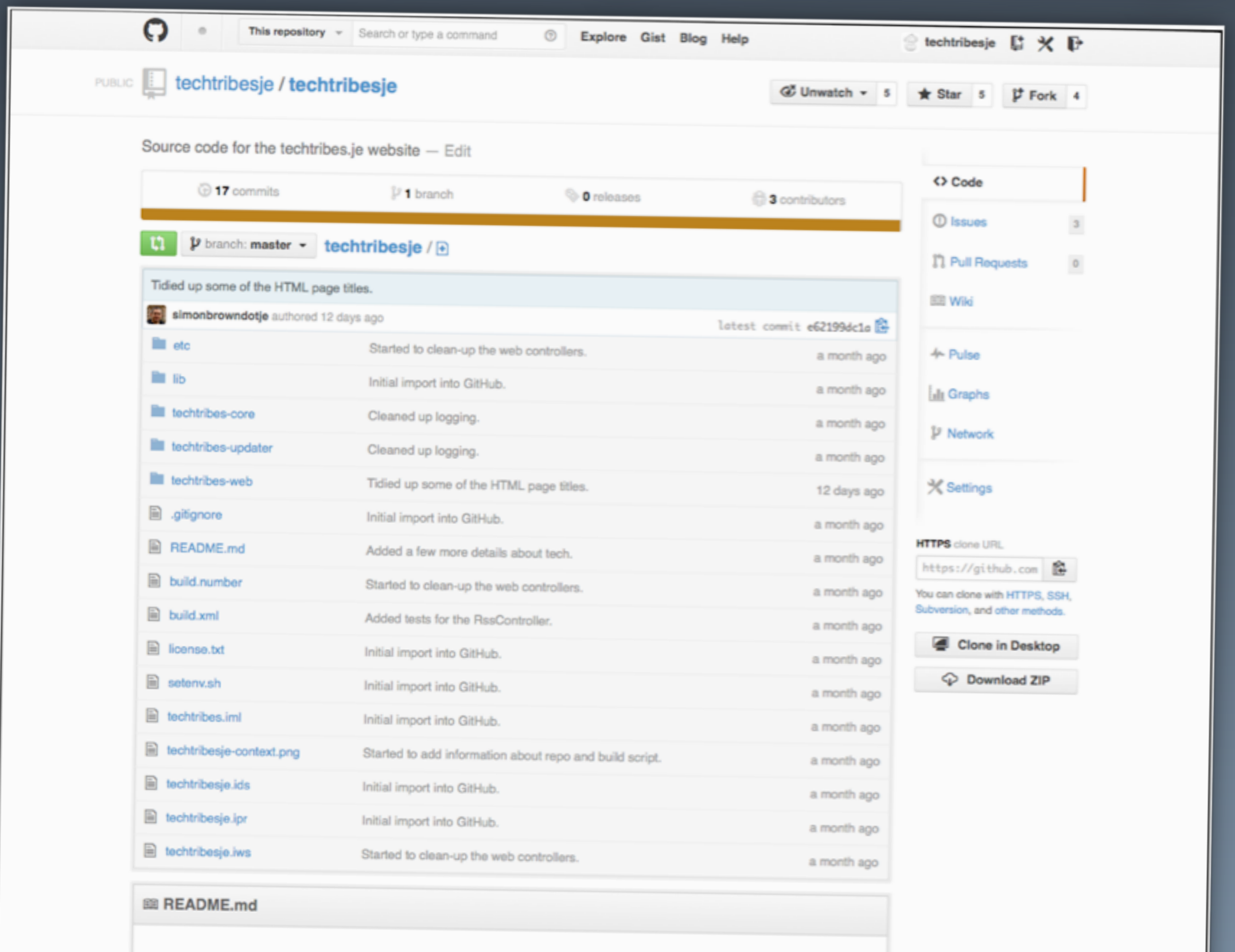




If the answer is “no” … are the diagrams actually useful?

Does the `code` for techtribes.je reflect the abstractions on the software architecture diagrams?

https://github.com/techtribesje/techtribesje



TweetComponent.java – [techtribes-core] – techtribesje – [~/Documents/sandbox/techtribesje]

Tomcat 7.0

techtribesje › techtribes-core › src › je › techtribes › component › tweet

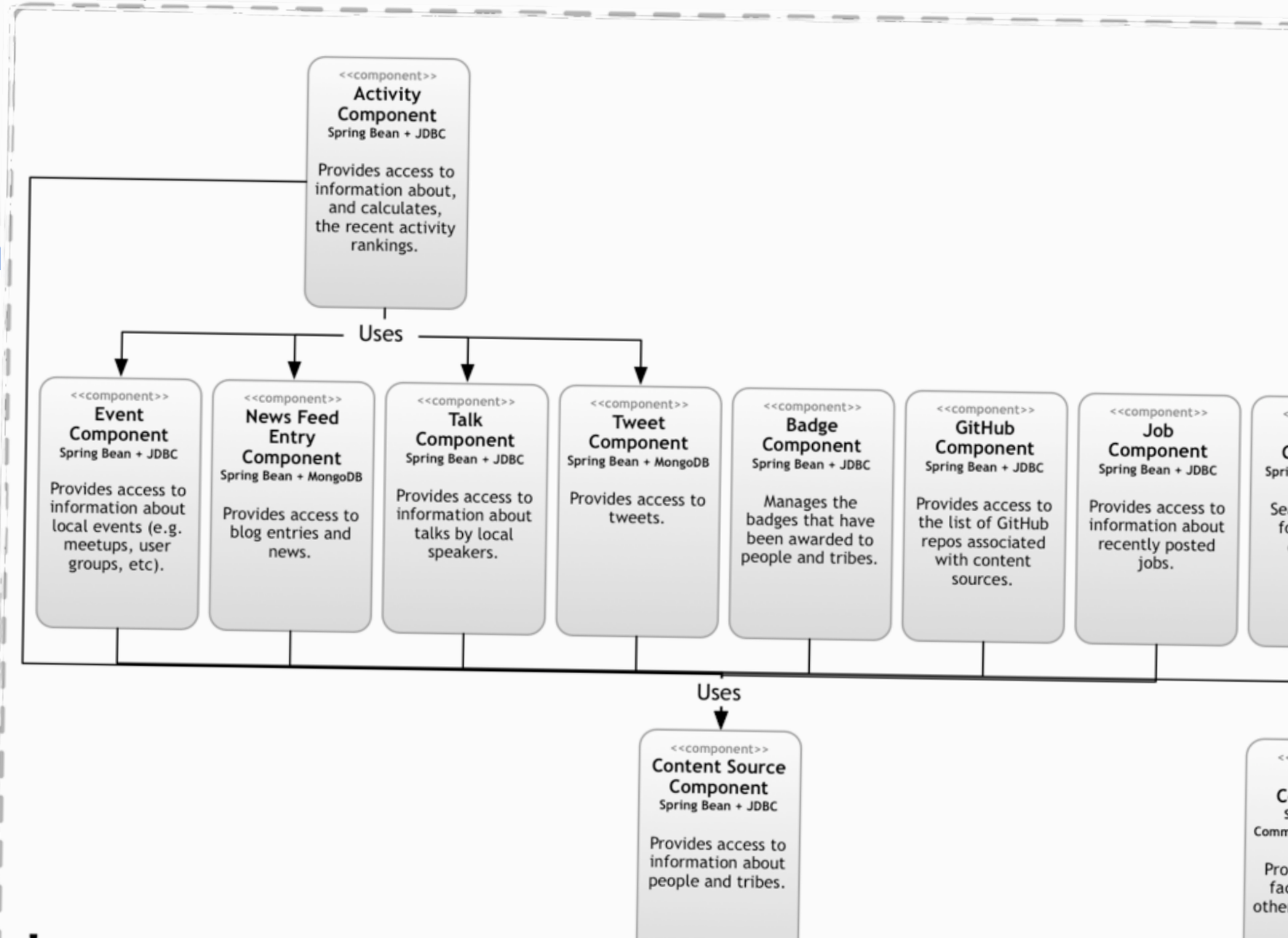
Project

- techtribesje [techtribes] (~/Documents/sandbox/techtribesje)
  - docs
  - lib
  - techtribes-core
    - sql
    - src
      - je.techtribes
        - component
          - activity
          - badge
          - contentsource
          - event
          - github
          - job
          - log
          - newsfeedentry
          - search
          - talk
          - tweet
            - component.xml
            - MongoDbTweetDao
            - TweetComponent
            - TweetComponentImpl
            - TweetException
        - domain
        - util
        - config.xml
    - test
    - techtribes-core.iml
  - techtribes-updater
  - techtribes-web
  - .gitignore
  - build.number
  - build.xml
  - license.txt
  - README.md
  - setenv.sh
  - software-architecture-for-developers.png
  - techtribes.iml

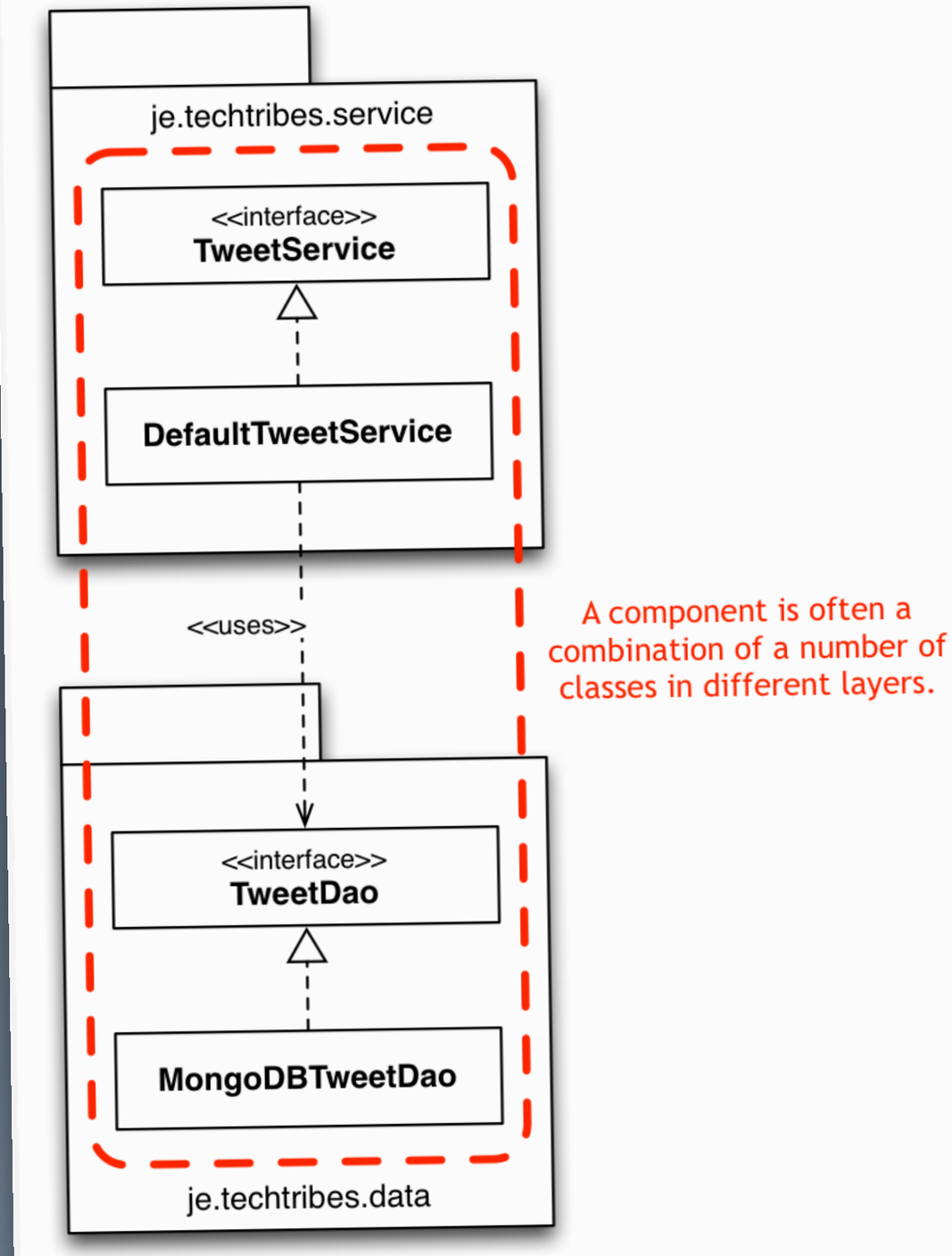
6: TODO 9: Changes Application Servers

TweetComponent.java | TweetComponentImpl.java | MongoDbTweetDao.java

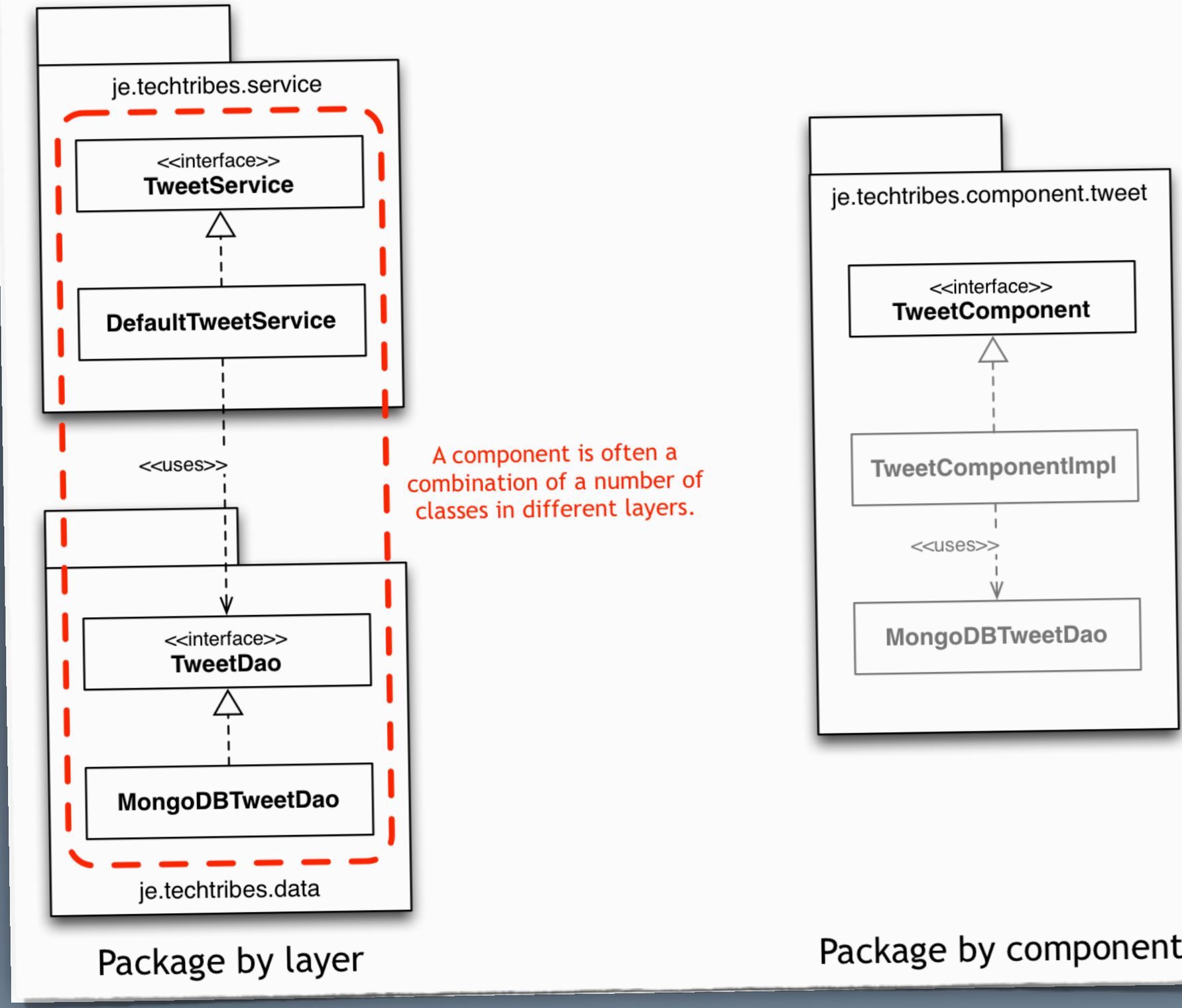
```java
package je.techtribes.component.tweet;

import ...

@Component(responsibility = "Provides access to tweets.")
@Collaboration(target = ContentSourceComponent.class, responsibility = "Enriches with")
public interface TweetComponent {

    /**
     * Gets the most recent tweets by page number.
     */
    List<Tweet> getRecentTweets(int page, int pageSize);
```



Did it start out that way?



What's a "component"?



What's a "component"?

Don’t do unit testing!

"In the early days of computing when computers were slow, unit tests gave the developer more immediate feedback about whether a change broke the code instead of waiting for system tests to run. Today, with cheaper and more powerful computers, that argument is less persuasive."

RBCS

TIME TESTED.
TESTING IMPROVED.

www.RBCS-US.com

Why Most Unit Testing is Waste

By James O Coplien

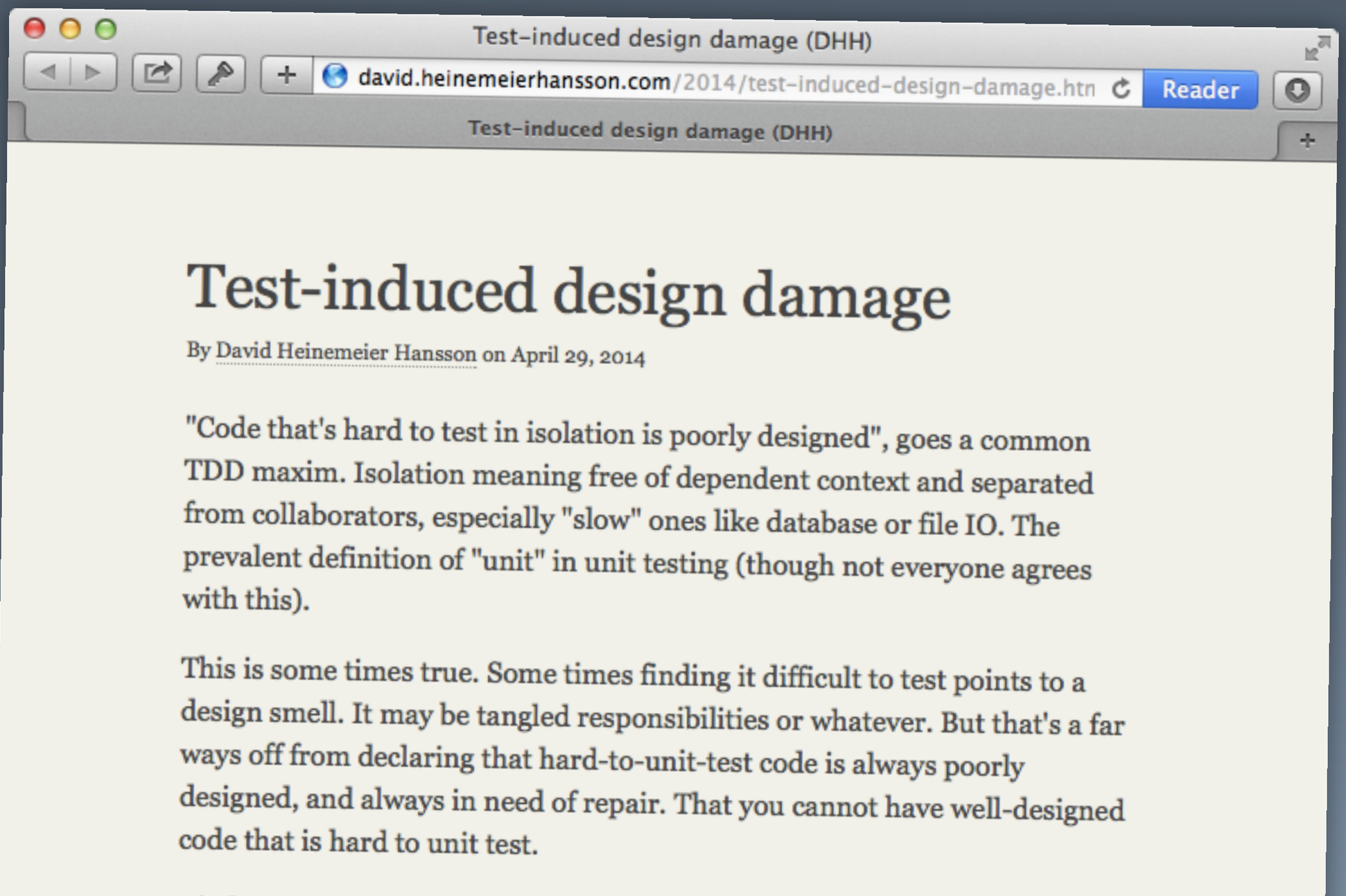
1.1 Into Modern Times

Unit testing was a staple of the FORTRAN days, when a function was a function and was sometimes worthy of functional testing. Computers computed, and functions and procedures

“If your coders have more lines of unit tests than of code, it probably means one of several things.
… Or the problem may be at the other end:

developers don’t have adequately refined design skills, or the process doesn't encourage architectural thinking and conscientious design.”

“do not let your tests drive your design”



Instead of unit testing everything, what about testing your significant structural elements as black boxes?

The point of this?

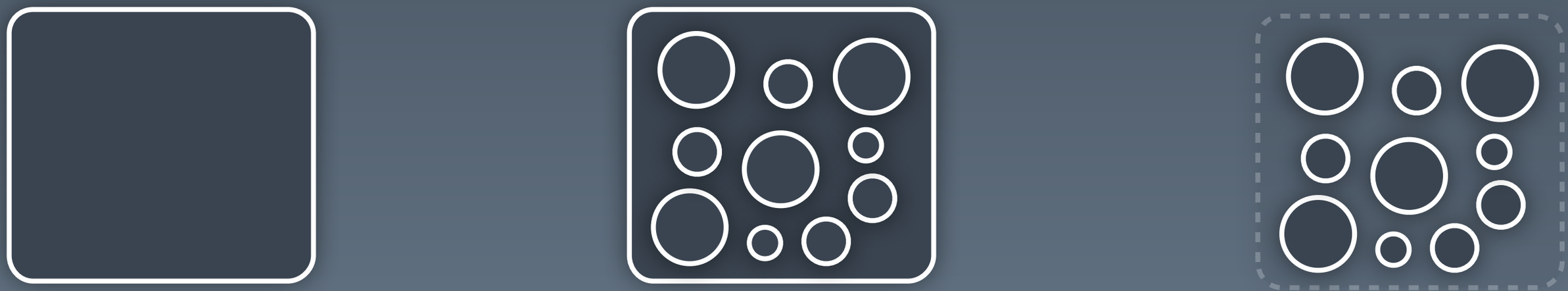
A good architecture enables

agility

Monolithic architecture

Service-based architecture

(SOA, micro-services, etc)



Something in between

(components)

The structure of your software and the **decomposition** strategy you use to get there are important

Micro-services …
also known as a
distributed
big ball of mud

Inspect

and

Adapt

Think about how to align the

software architecture

and the

code

A simple and explicit mapping assists with understanding, refactoring, etc

Until we find the one true solution that works in all contexts,

you'll need to think

If your software system is hard to work with,

change it!

simon.brown@codingthearchitecture.com
@simonbrown on Twitter