Software Architecture
for Developers

@simonbrown

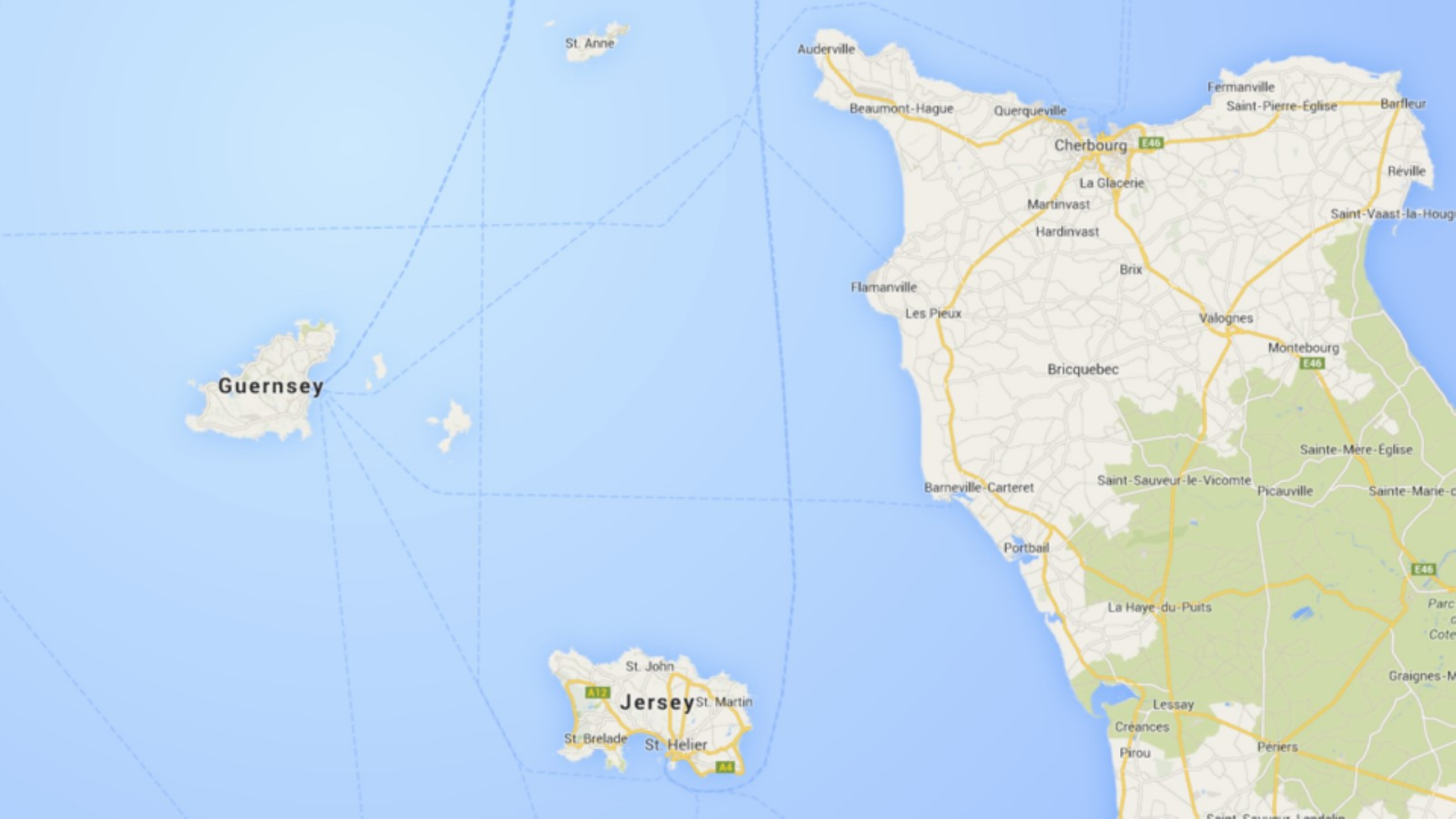
Simon Brown

Independent consultant specialising in software architecture, plus the creator of the C4 model and Structurizr

@simonbrown


United Kingdom
Northern Ireland
Ireland
Isle of Man
England
Wales
Scotland
Denmark
Germany
Netherlands
Belgium
Luxembourg
Austria
Switzerland
Glasgow
Perth
Kirkcaldy
Londonderry / Derry
Bangor
Sligo
Dundalk
Castlebar
Drogheda
Dublin
Galway
Naas
Bray
Limerick
Carlow
Kilkenny
Tralee
Clonmel
Waterford
Cork
Middlesbrough
Blackpool
Leeds
Hull
Liverpool
Sheffield
Nottingham
Stoke-on-Trent
Derby
Norwich
Great Yarmouth
Ipswich
Swansea
Bristol
Cardiff
Headington
London
Leigh-on-Sea
Reading
Barnstaple
Exeter
Southampton
Portsmouth
Bournemouth
Plymouth
Newquay
St. Ives
Truro
English Channel
Celtic Sea
Guernsey
Jersey
Amsterdam
The Hague
Den Helder
Groningen
Bremerhaven
Aurich
Hamburg
Bremen
Hanover
Berlin
Bielefeld
Brunswick
Magdeburg
Dessau
Münster
Kernstadt
Woensel-Noord
Bruges
Antwerp
Brussels
Lille
Dusseldorf
Nord
Cologne
Leipzig
Dresden
Chemnitz
Gera
Frankfurt
Wiesbaden
Würzburg
Mannheim
Nuremberg
Saarbrücken
Karlsruhe
Stuttgart
Metz
Strasbourg
Nancy
Ulm
Munich
Freiburg
Konstanz
Basel
Zurich
Innsbruck
Salzburg
Linz
Wels
Prague
Plzen
Pardubice
Ceske Budejovice
Regensburg
Ingolstadt
Landshut
Amiens
Saint-Quentin
Le Havre
Caen
Rouen
Reims
Paris
Évry
Troyes
Orleans
Le Mans
Rennes
Brest
Lannion
St-Malo
Saint-Brieuc
Quimper
Lorient
Vannes
Angers
Tours
Nantes
Saint-Nazaire
Dijon
Besançon
Aarhus
Viborg
Silkeborg
Horsens
Esbjerg
Odense
Copenhagen
Malmö
Helsingborg
Kristianstad
Karlskrona
Flensburg
Kiel
Heide
Rostock
Stralsund
Swinoujscie
Anklam
Gryfice
Schwerin
Neubrandenburg
Bergedorf
Szczecin
Stargard Szczecinski
Gorzów Wielkopolski
Zielona Gora
Cottbus
Görlitz
Liberec
Ústí nad Labem-město
Oslo
Gothenburg
Stockholm
Helsinki
Tallinn
Riga
Vilnius
Minsk
Warsaw
Kraków
Katowice
Ostrava
Brno
Bratislava
Vienna
Budapest
Győr
Graz
Maribor
Ljubljana
Zagreb
Rijeka
Split
Sarajevo
Belgrade
Novi Sad
Subotica
Osijek
Timişoara
Arad
Cluj-Napoca
Oradea
Debrecen
Miskolc
Košice
Lviv
Kyiv
Chernihiv
Sumy
Kharkiv
Dnipro
Zaporizhzhia
Mariupol
Donetsk
Luhansk
Simferopol
Sevastopol
Yalta
Kerch
Crimea
Rostov-on-Don
Volgograd
Saratov
Samara
Ufa
Chelyabinsk
Yekaterinburg
Perm
Kazan
Nizhny Novgorod
Moscow
Tula
Ryazan
Voronezh
Belgorod
Kursk
Orel
Bryansk
Smolensk
Tver
Yaroslavl
Vologda
Arkhangelsk
Murmansk
Petrozavodsk
Saint Petersburg
Pskov
Novgorod
Kaliningrad
Gdańsk
Bydgoszcz
Poznań
Łódź
Wrocław
Opole
Gliwice
Rybnik
Ostrawa
Olomouc
Pardubice
Hradec Králové
Plzeň
Karlovy Vary
Ústí nad Labem
Liberec
Jablonec nad Nisou
Trutnov
Jelenia Góra
Legnica
Lubin
Zielona Góra
Gorzów Wielkopolski
Stargard Szczecinski
Szczecin
Gryfice
Swinoujscie
Anklam
Neubrandenburg
Bergedorf
Schwerin
Rostock
Stralsund
Heide
Kiel
Flensburg
Esbjerg
Odense
Copenhagen
Malmö
Helsingborg
Kristianstad
Karlskrona
Horsens
Silkeborg
Viborg
Aarhus
Kirkcaldy
Perth
Glasgow
Kilmarnock
Ayr
Middlesbrough
Blackpool
Leeds
Hull
Liverpool
Sheffield
Nottingham
Stoke-on-Trent
Derby
Norwich
Great Yarmouth
Ipswich
Swansea
Bristol
Cardiff
Headington
London
Leigh-on-Sea
Reading
Barnstaple
Exeter
Southampton
Portsmouth
Bournemouth
Plymouth
Newquay
St. Ives
Truro
English Channel
Celtic Sea
Guernsey
Jersey
Amsterdam
The Hague
Den Helder
Groningen
Bremerhaven
Aurich
Hamburg
Bremen
Hanover
Berlin
Bielefeld
Brunswick
Magdeburg
Dessau
Münster
Kernstadt
Woensel-Noord
Bruges
Antwerp
Brussels
Lille
Dusseldorf
Nord
Cologne
Leipzig
Dresden
Chemnitz
Gera
Frankfurt
Wiesbaden
Würzburg
Mannheim
Nuremberg
Saarbrücken
Karlsruhe
Stuttgart
Metz
Strasbourg
Nancy
Ulm
Munich
Freiburg
Konstanz
Basel
Zurich
Innsbruck
Salzburg
Linz
Wels
Prague
Plzen
Pardubice
Ceske Budejovice
Regensburg
Ingolstadt
Landshut
Amiens
Saint-Quentin
Le Havre
Caen
Rouen
Reims
Paris
Évry
Troyes
Orleans
Le Mans
Rennes
Brest
Lannion
St-Malo
Saint-Brieuc
Quimper
Lorient
Vannes
Angers
Tours
Nantes
Saint-Nazaire
Dijon
Besançon
North Sea


Ireland
Dublin
Belfast
Cork
Limerick
Galway
Waterford
Kilkenny
Wexford
Bray
Naas
Portlaoise
Wicklow
Ennis
Carlow
Enniscorthy
Gorey
Newcastle West
Clonmel
Fermoy
Dungarvan
Mallow
Midleton
Bandon
Kinsale
Tullamore
Bangor
Colwyn Bay
Caernarfon
Wales
Cardiff
Swansea
Newport
Pontypridd
Barry
Haverfordwest
Carmarthen
Llanelli
Neath
Milford Haven
Aberystwyth
Newtown
Shrewsbury
Hereford
Worcester
Kidderminster
Cheltenham
Gloucester
Bristol
Bath
Swindon
Oxford
Reading
London
Basildon
Southend-on-Sea
Croydon
Maidstone
Margate
Deal
Dover
Ashford
Crawley
Brighton
Hastings
Eastbourne
Southampton
Portsmouth
Isle of Wight
Newport
Yeovil
Taunton
Barnstaple
Bideford
Exeter
Tavistock
Torquay
Plymouth
Newquay
Truro
St Austell
St Ives
Penzance
Falmouth
Dartmouth
Weymouth
Swanage
Shanklin
Stoke-on-Trent
Derby
Leicester
Peterborough
Boston
Lincoln
Cromer
King's Lynn
North Walsham
Dereham
Norwich
Lowestoft
Beccles
Attleborough
Corby
Cambridge
Bedford
Luton
St Albans
Chelmsford
Colchester
Felixstowe
Clacton-on-Sea
Ipswich
England
Scotland
Northern Ireland
English Channel
Celtic Sea
Guernsey
Jersey
Cherbourg
Bricquebec
Le Havre
Deauville
Caen
Saint-Lô
Lisieux
Coutances
Granville
Vire
Flers
Saint-Malo
Dreux
Rouen
Elbeuf
Louviers
Évreux
Beauvais
Amiens
Dieppe
Fécamp
Abbeville
Saint-Quentin
Cergy
Paris
Boulogne-Billancourt
Laon
Soissons
Reims
Épernay
Châlons-en
Lannion
Saint-Pol-de-Léon
Morlaix
Dunkirk
Calais
Boulogne-sur-Mer
Ostend
Bruges
Ghent
Kortrijk
Lille
Tournai
Béthune
Arras
Douai
Cambrai
Mons
Vlissingen
M53
M1
M54
M6 Toll
M42
M40
M4
M3
M25
M11
M20
E20
E201
E30
E40
E42
E15
E17
E402
E44
E46
E19
E401
E3
E50
E312


St. Anne
Auderville
Beaumont-Hague
Querqueville
Fermanville
Saint-Pierre-Église
Barfleur
Cherbourg
E46
Réville
La Glacerie
Martinvast
Hardinvast
Saint-Vaast-la-Hougue
Brix
Flamanville
Les Pieux
Valognes
Montebourg
E46
Bricquebec
Guernsey
Sainte-Mère-Église
Saint-Sauveur-le-Vicomte
Barneville-Carteret
Picauville
Portbail
E46
La Haye-du-Puits
St. John
A12
Jersey
St. Martin
St. Brelade
St. Helier
A4
Lessay
Créances
Pirou
Périers


St. John
St. Mary
St. Ouen
Rte de l'Etacq
Grande Route des Mielles
Les Mielles
Nature Reserve
St. Peter
Jersey
Trinity
La Grande Route de Saint Jean
La Route de la Trinité
La Verte Rue
St. Martin
Mont Orgueil Castle
St. Brelade
A13
A13
A1
A2
St. Helier
Bagatelle Rd
Grouville
A4
St. Malo, FR - Portsmouth, UK
St. Peter Port, Gg - Saint Helier, Je
St. Helier, Je - Saint Peter Port, Gg
Poole - Jersey (St. Helier)
Jersey (St. Helier) - Poole
Poole - Jersey (St. Helier)
Granville, FR - St. Helier, JE

I’ve visited and spoken at events/organisations in 30+ countries

What is software architecture?

Structure

The definition of software in terms
of its building blocks and their interactions

Vision

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

Application Architecture

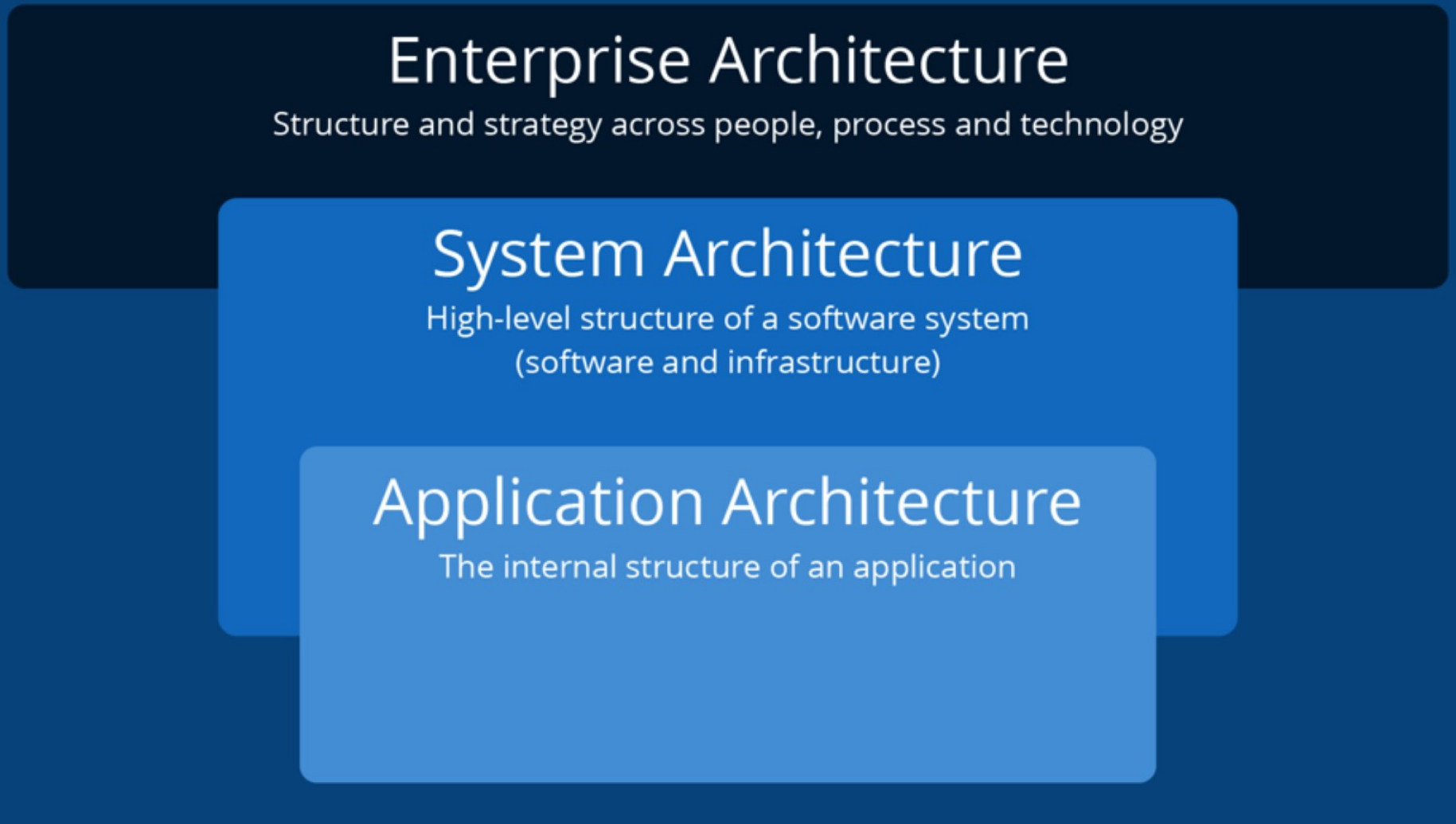
The internal structure of an application

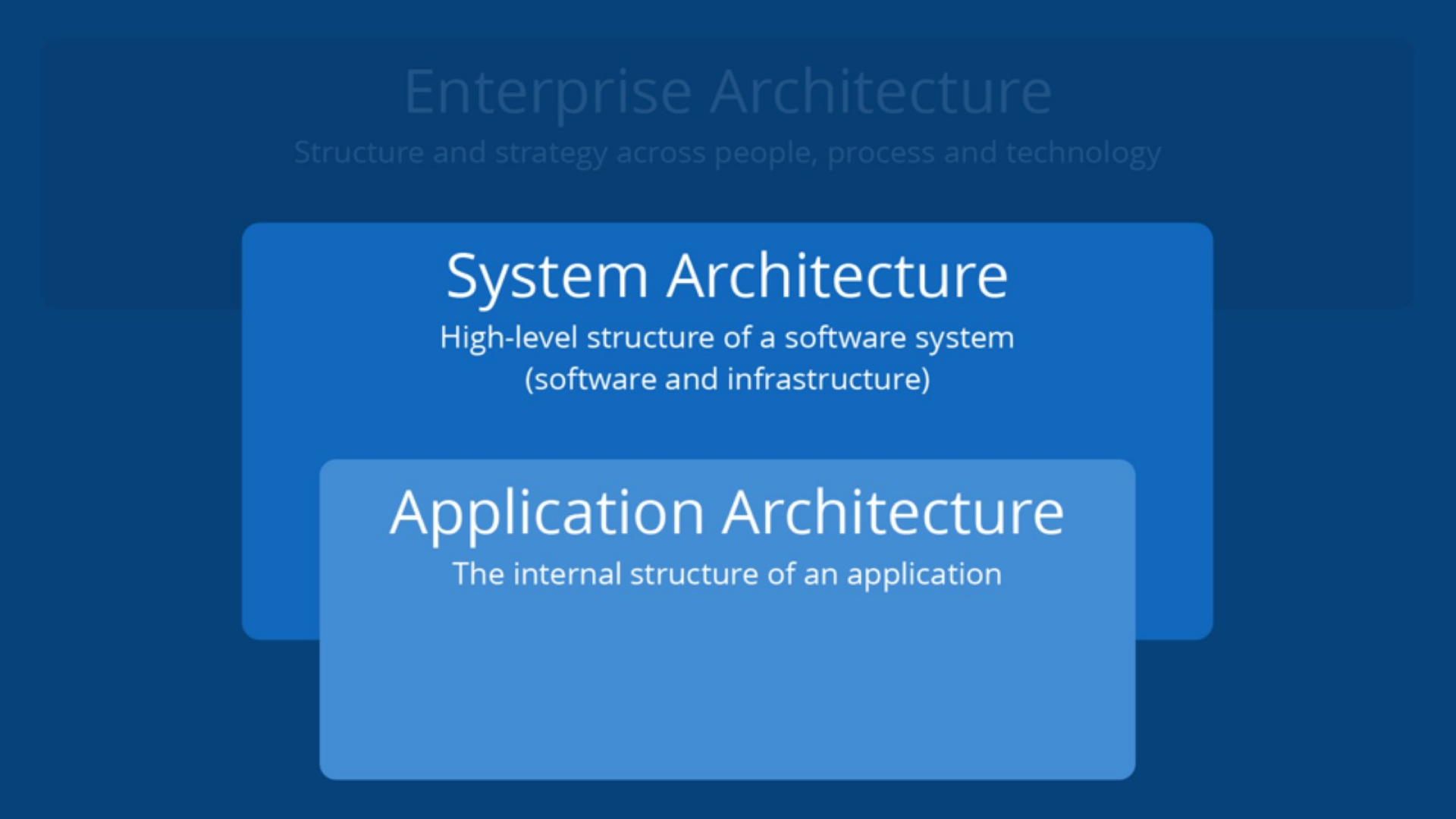
System Architecture

High-level structure of a software system
(software and infrastructure)

Application Architecture

The internal structure of an application


Enterprise Architecture
Structure and strategy across people, process and technology
System Architecture
High-level structure of a software system
(software and infrastructure)
Application Architecture
The internal structure of an application


Enterprise Architecture
Structure and strategy across people, process and technology
System Architecture
High-level structure of a software system
(software and infrastructure)
Application Architecture
The internal structure of an application

As a noun, design is the named structure or behaviour of a system whose presence resolves ... a force on that system. A design thus represents one point in a potential decision space.

Grady Booch

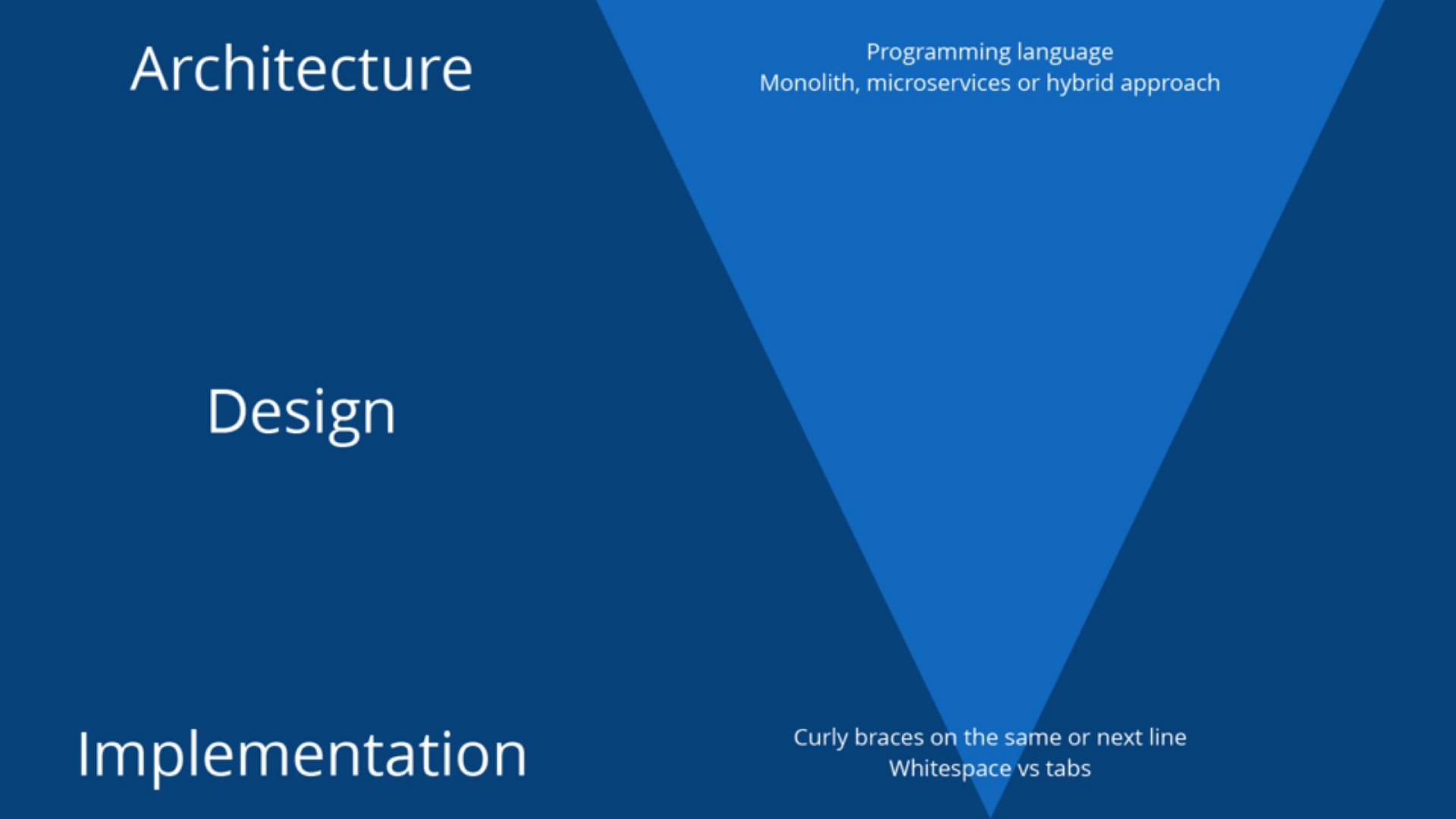
All architecture is design, but
not all design is architecture.

Grady Booch

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

Grady Booch

As architects, we define
the **significant decisions**


Architecture
Programming language
Monolith, microservices or hybrid approach
Design
Implementation
Curly braces on the same or next line
Whitespace vs tabs

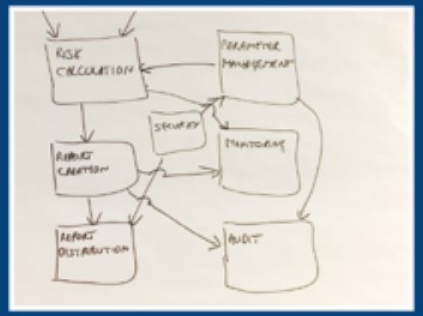
What happens if a software development team **doesn't think about architecture**?

Chaos

Big ball of mud, spaghetti code, inconsistent approaches to solving the same problems, quality attributes are ignored, deployment problems, maintenance issues, etc

Big design up front

vs



No design up front

Big design up front is dumb.
Doing no design up front
is even dumber.

Dave Thomas

Software architecture helps us avoid chaos

Architectural drivers

Requirements
drive architecture

(use cases, user stories, features, etc)

Quality attributes

(also known as non-functional requirements,
cross-cutting concerns, service-level agreements, etc)

- Performance
- Scalability
- Availability
- Security
- Disaster Recovery
- Accessibility
- Monitoring
- Management
- Audit
- Flexibility
- Extensibility
- Maintainability
- Interoperability
- Legal
- Regulatory
- Compliance
- i18n
- L10n

Create a **checklist** of quality attributes you regularly encounter

Understand how to **capture**, **refine** and **challenge** quality attributes

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

Typical constraints include time and budget, technology, people and skills, politics, etc

Constraints can **sometimes** be prioritised

Principles

are selected by the team

Development principles include coding conventions, naming guidelines, testing approaches, review practices, etc

Architecture and design principles typically relate to modularity or crosscutting concerns

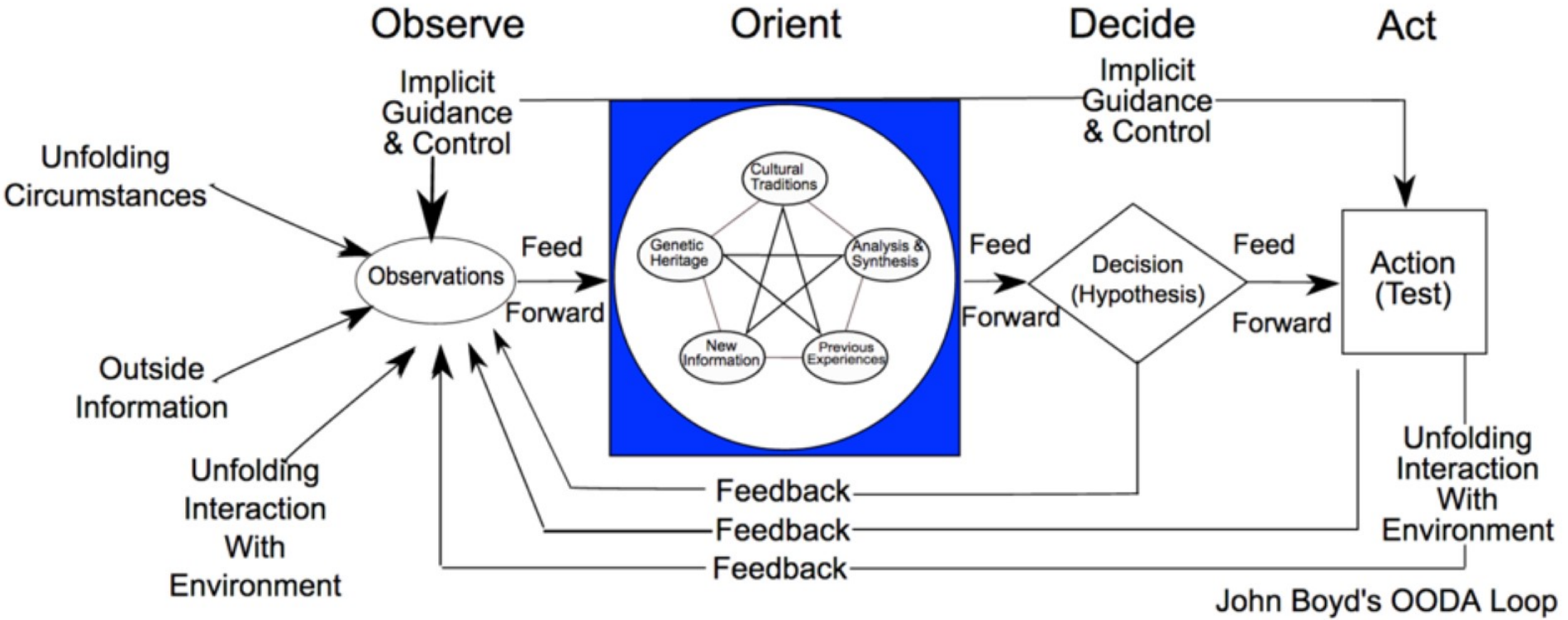
(architectural layering, separation of concerns, stateless vs stateful, rich vs anaemic domain, security, error handling, logging, etc)

Ensure you have a good understanding of the requirements, quality attributes, constraints and principles to create **sufficient foundations**

What about agile, and agility?

Agile is about moving fast, embracing change, releasing often, getting feedback, ...

Agile is about a mindset of
continuous improvement


Observe
Orient
Decide
Act
Implicit Guidance & Control
Implicit Guidance & Control
Unfolding Circumstances
Observations
Feed Forward
Cultural Traditions
Genetic Heritage
Analysis & Synthesis
New Information
Previous Experiences
Feed Forward
Decision (Hypothesis)
Feed Forward
Action (Test)
Outside Information
Unfolding Interaction With Environment
Feedback
Feedback
Feedback
Unfolding Interaction With Environment
John Boyd's OODA Loop

Inspect and **adapt**

Agility is relative and time-based

What Lessons Can the Agile Community Learn from A Maverick Fighter Pilot?

Steve Adolph
University of British Columbia
Vancouver, B.C.
Canada
steve@wsaconsulting.com

Abstract

For the agile software development community, agility is defined by the values expressed in the agile manifesto. But in concrete terms, what does it mean for

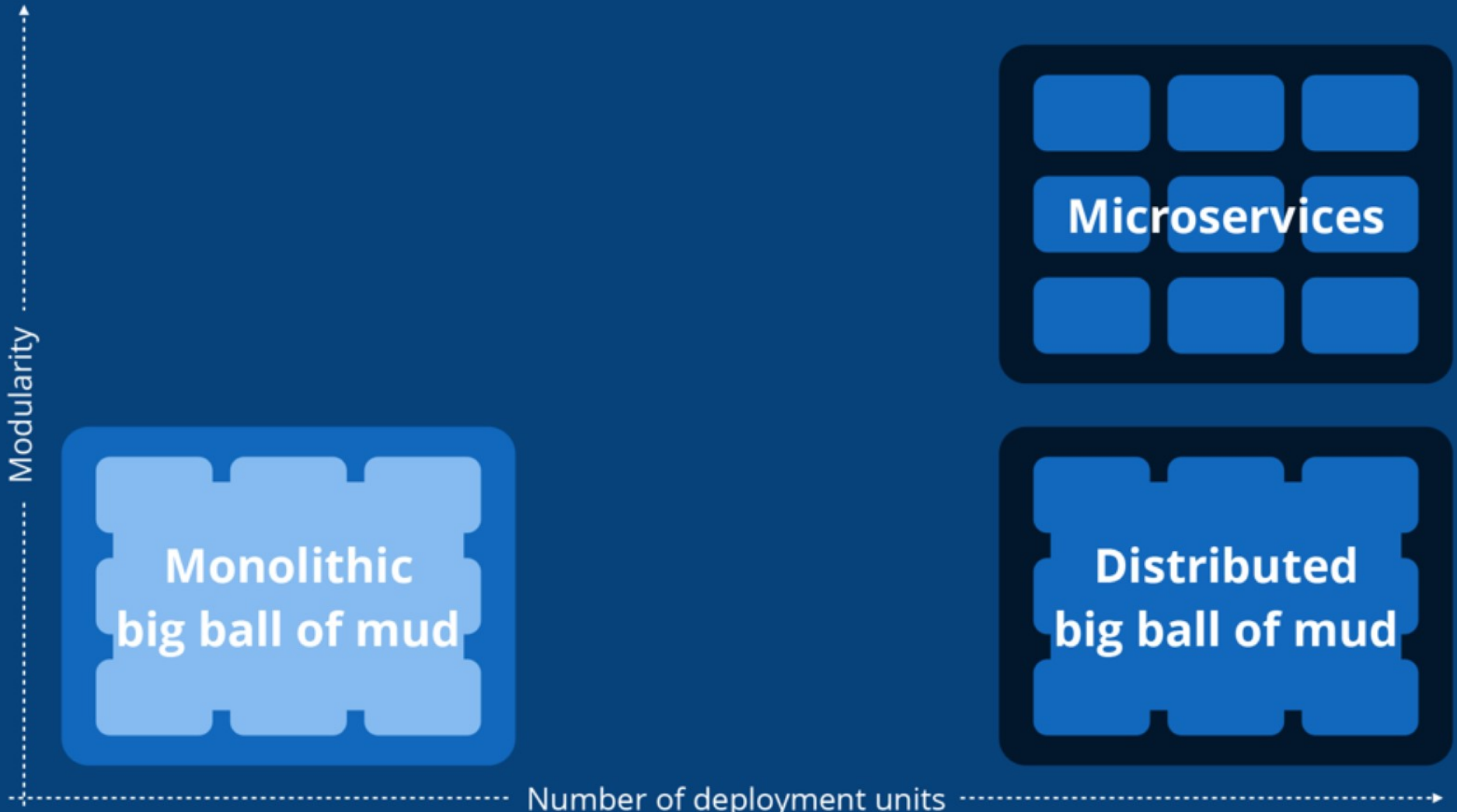
based methodologies is it by definition not agile? Many RUP and UP experts claim their process is agile [5, 6, 7] and many practitioners (including this author) will enthusiastically claim that even though they were not following an agile methodology, their projects were agile. So what does it really mean for a software

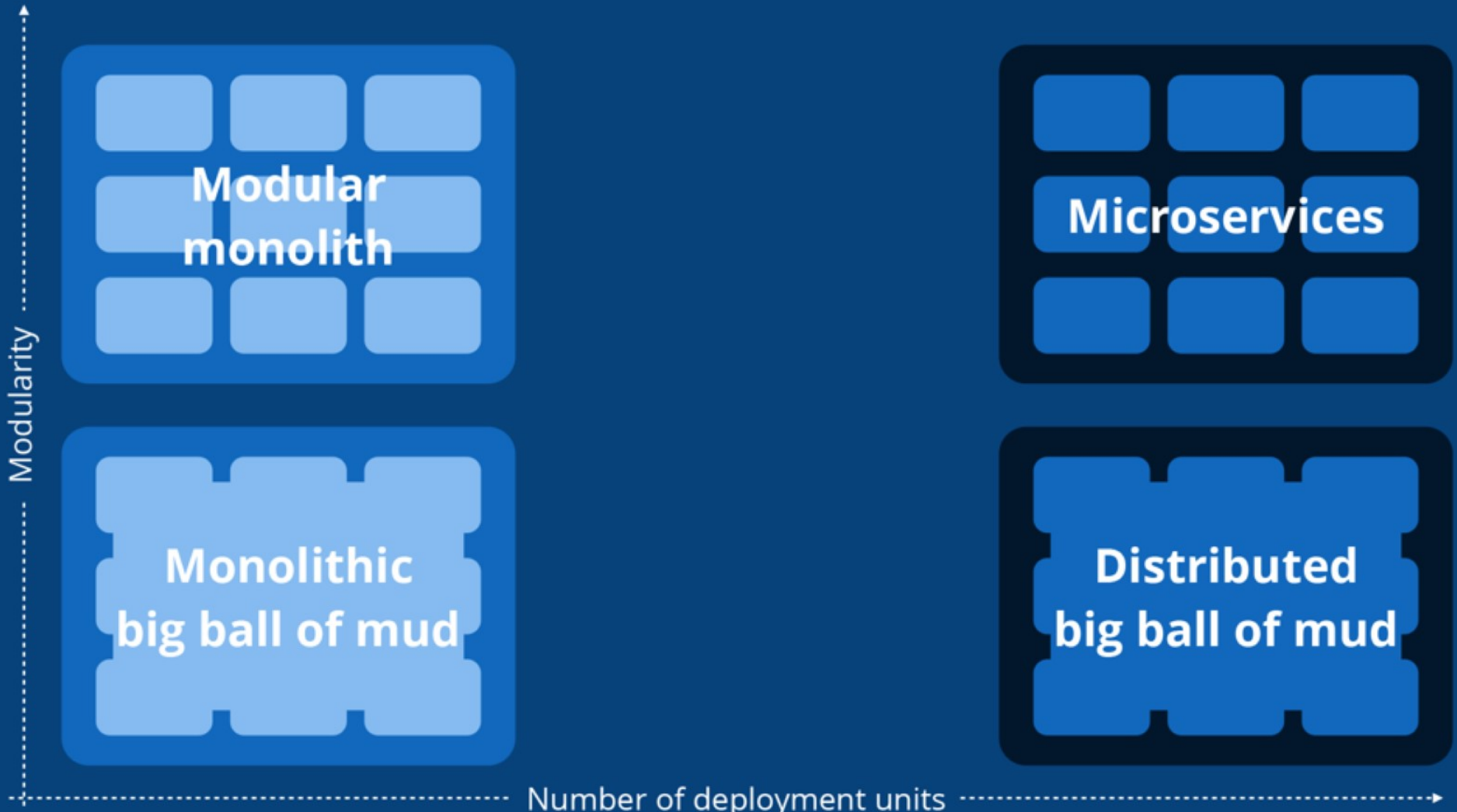
Continuous attention to technical excellence and good design enhances agility.

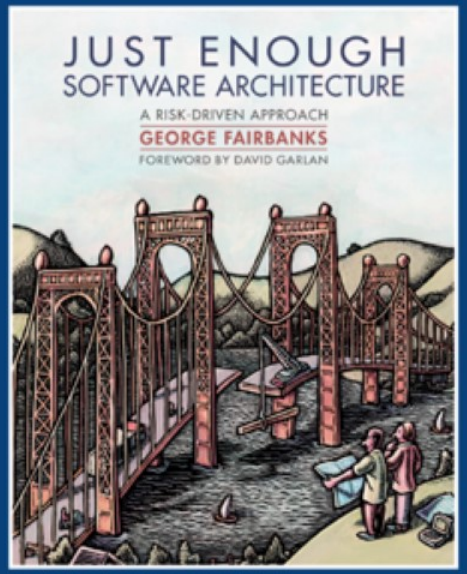
Principle 9 of the Manifesto for Agile Software Development

A good architecture enables agility


Modularity
Microservices
Monolithic
big ball of mud
Number of deployment units


Modularity
Microservices
Monolithic
big ball of mud
Distributed
big ball of mud
Number of deployment units


Modularity
Modular monolith
Microservices
Monolithic big ball of mud
Distributed big ball of mud
Number of deployment units



A good architecture rarely happens through **architecture-indifferent design**

Agility is a

quality attribute

Questions and break

The software architecture role

The software architecture role
is about the “**big picture**”
and, sometimes, this means
stepping away from the code

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

Architectural drivers

Understanding the goals; capturing, refining and challenging the requirements and constraints.

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

Architectural drivers

Understanding the goals; capturing, refining and challenging the requirements and constraints.

Designing software

Creating the technical strategy, vision and roadmap.

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

Architectural drivers

Understanding the goals; capturing, refining and challenging the requirements and constraints.

Designing software

Creating the technical strategy, vision and roadmap.

Technical risks

Identifying, mitigating and owning the technical risks to ensure that the architecture “works”.

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

Architectural drivers

Understanding the goals; capturing, refining and challenging the requirements and constraints.

Designing software

Creating the technical strategy, vision and roadmap.

Technical risks

Identifying, mitigating and owning the technical risks to ensure that the architecture "works".

Technical leadership

Continuous technical leadership and ownership of the architecture throughout the software delivery.

The software architecture role

(technical leadership, and responsible for the technical success of the project/product)

Architectural drivers

Understanding the goals; capturing, refining and challenging the requirements and constraints.

Designing software

Creating the technical strategy, vision and roadmap.

Technical risks

Identifying, mitigating and owning the technical risks to ensure that the architecture "works".

Technical leadership

Continuous technical leadership and ownership of the architecture throughout the software delivery.

Quality assurance

Introduction and adherence to standards, guidelines, principles, etc.

I am a senior developer. Recently, I was promoted to the position as architect. Could anyone please let me know which tools/software an architect should master/be familiar with. Thank you

Experience is important ...

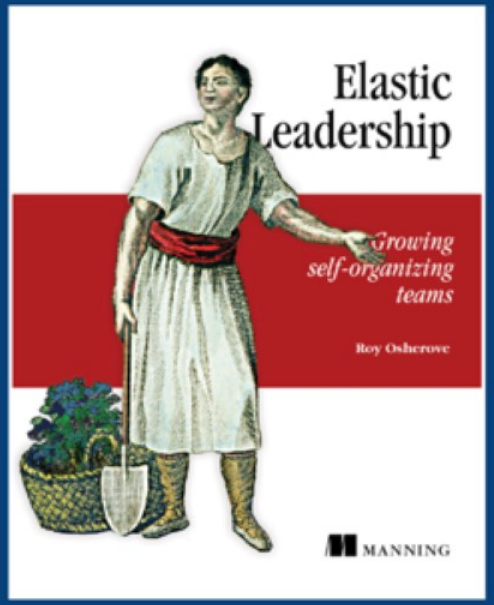
software architecture is not a rank!

Do you have your own definition of the software architecture role?

Every team needs technical leadership

Introducing control?
Avoiding chaos?

How much control do you need?



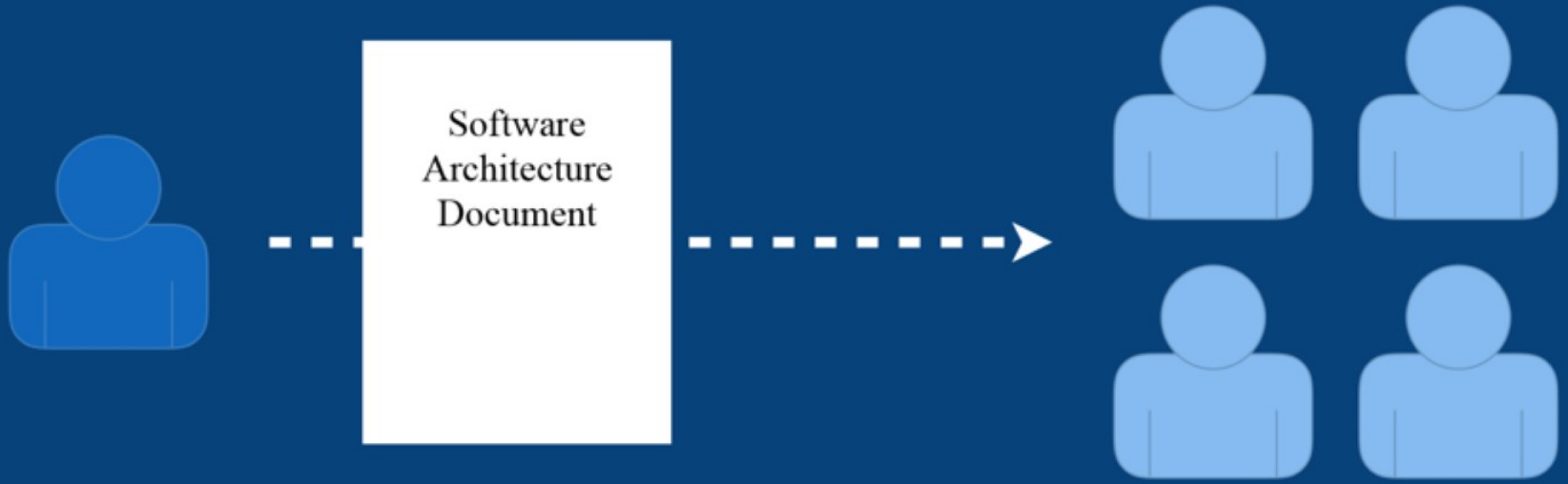
Different types of teams need different leadership styles

Pair architecting

Collaborative technical leadership

is not easy

Software development is not a relay sport



AaaS

Architecture as a Service

Continuous technical leadership

(somebody needs to continuously steer the ship)

Should software architects write **code**?

Poll

Should software architects write code?

Yes

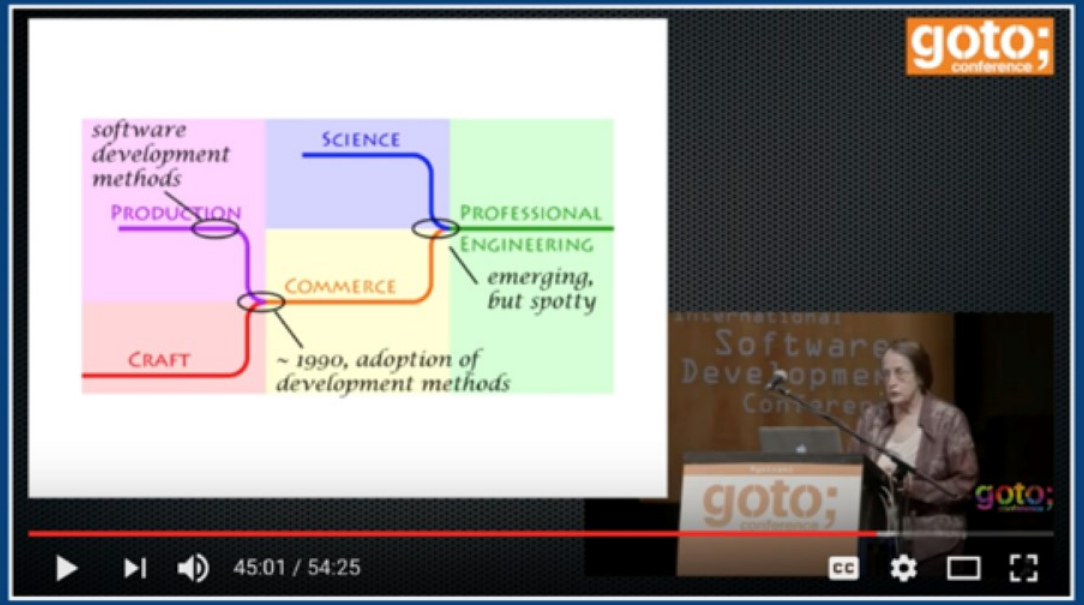
No

Production code, prototypes, frameworks, foundations, code reviews, experimenting, etc

Don’t code all of the time!

There is often a tension between being “senior” and writing code...

Software architects
should be
master builders



Progress Toward an Engineering Discipline of Software

Mary Shaw

Software architecture is not a “post-technical” career option!

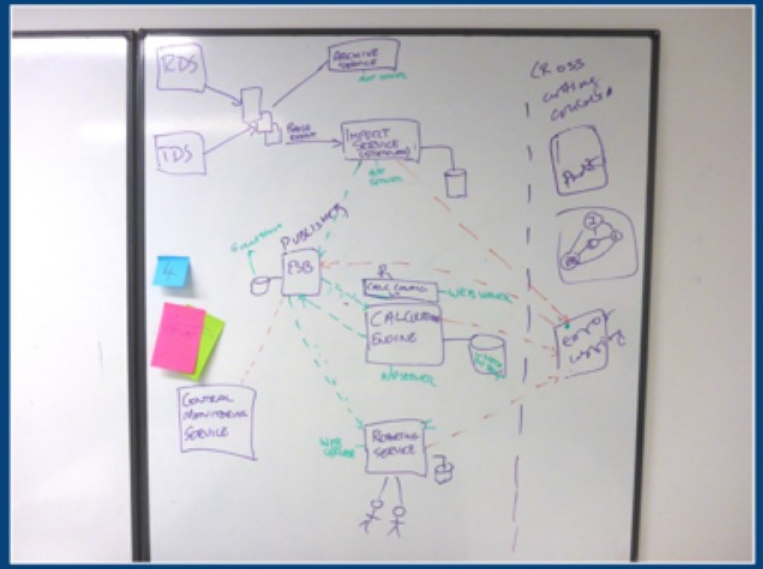
Technology skills

(depth and breadth)

Good software architects are good software developers

The people designing software must understand technology ...
all decisions involve trade-offs

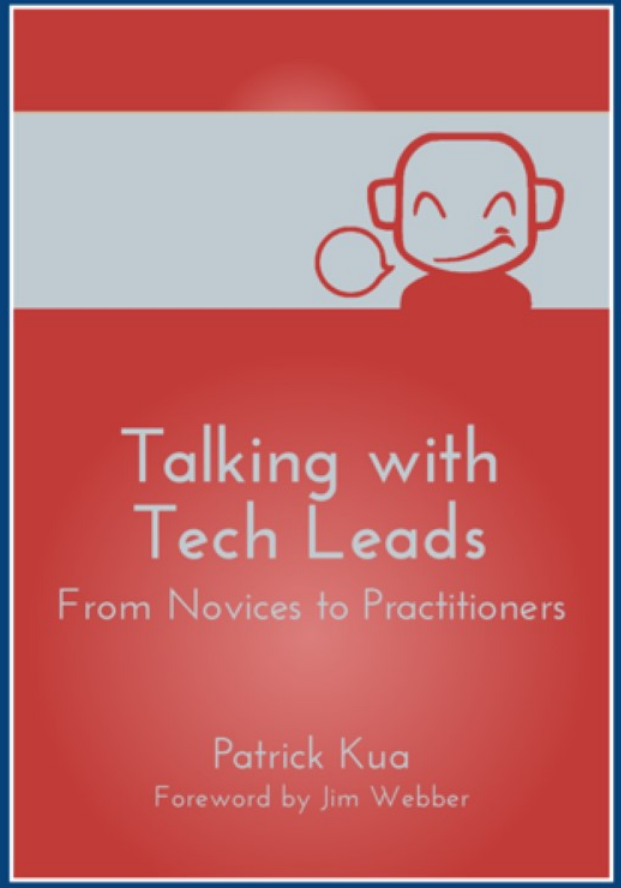
1. Is that what we're going to **build**?



2. Is it going to **work**?

Soft skills

(leadership, communication, presentation, influencing, negotiation, collaboration, coaching and mentoring, motivation, facilitation, political, etc)


Talking with
Tech Leads
From Novices to Practitioners
Patrick Kua
Foreword by Jim Webber

Collaborate or fail

Domain knowledge

(or the ability to learn quickly)

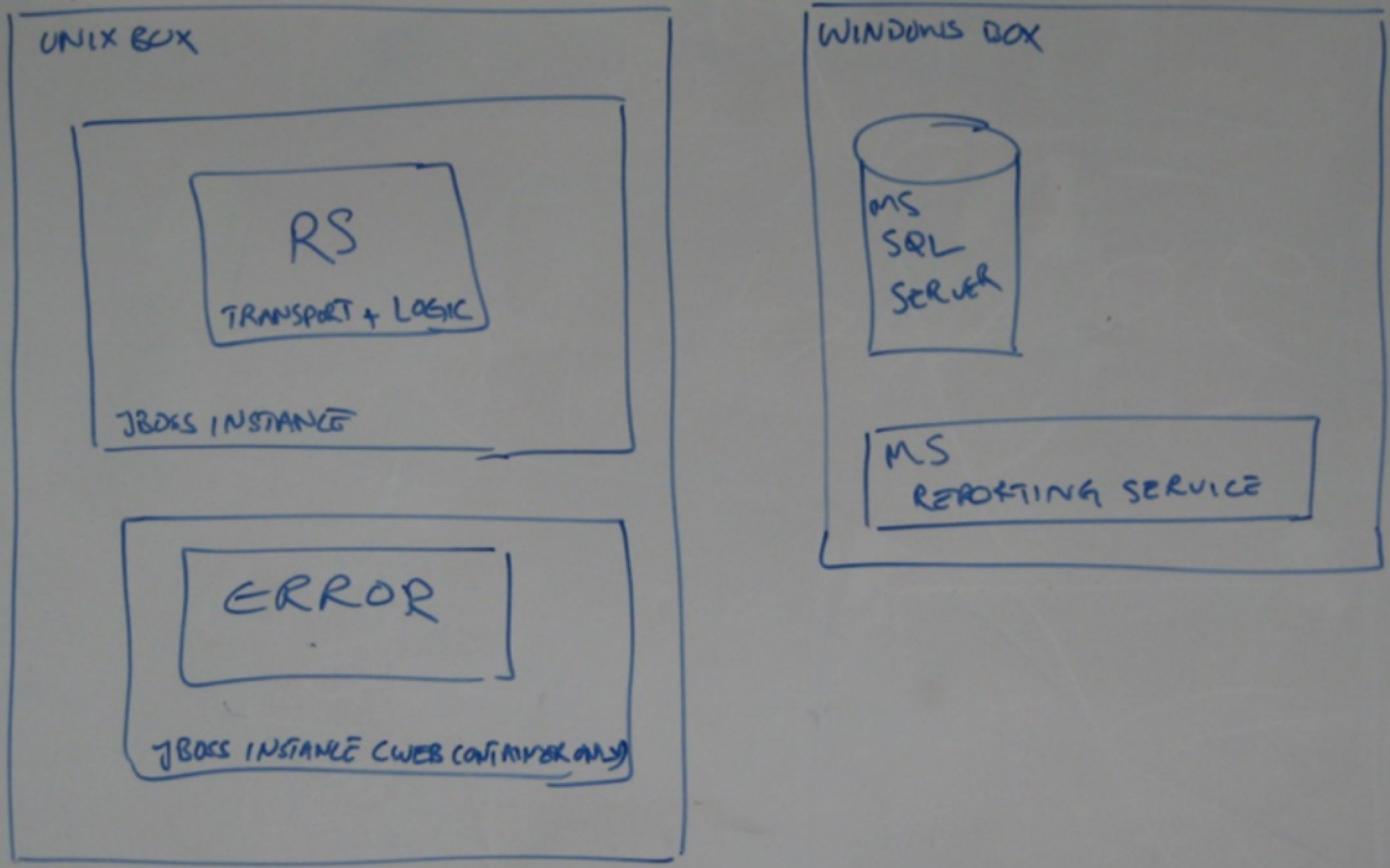
The software architecture role is **multi-faceted**

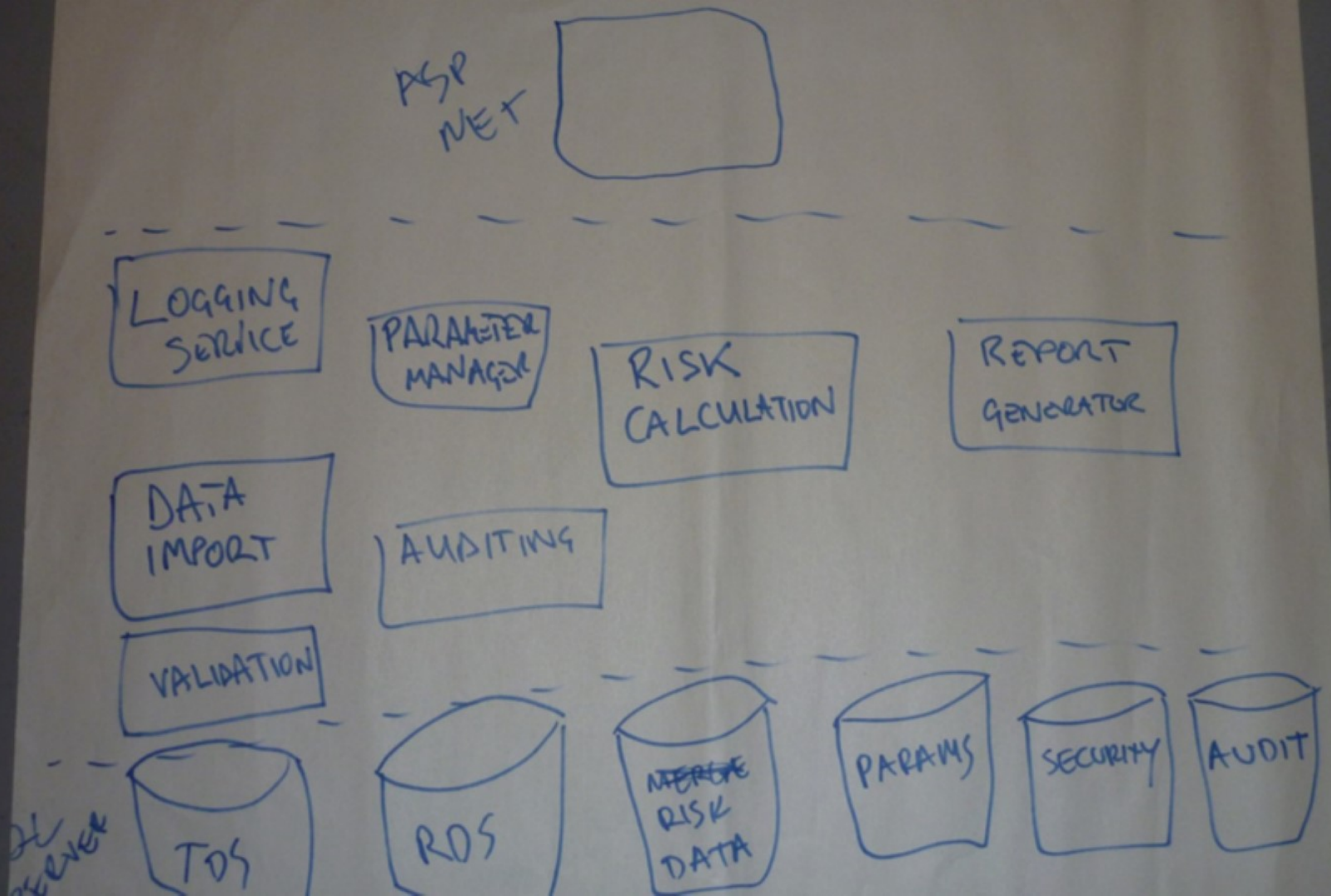
(technology, soft skills, domain knowledge)

Questions and break

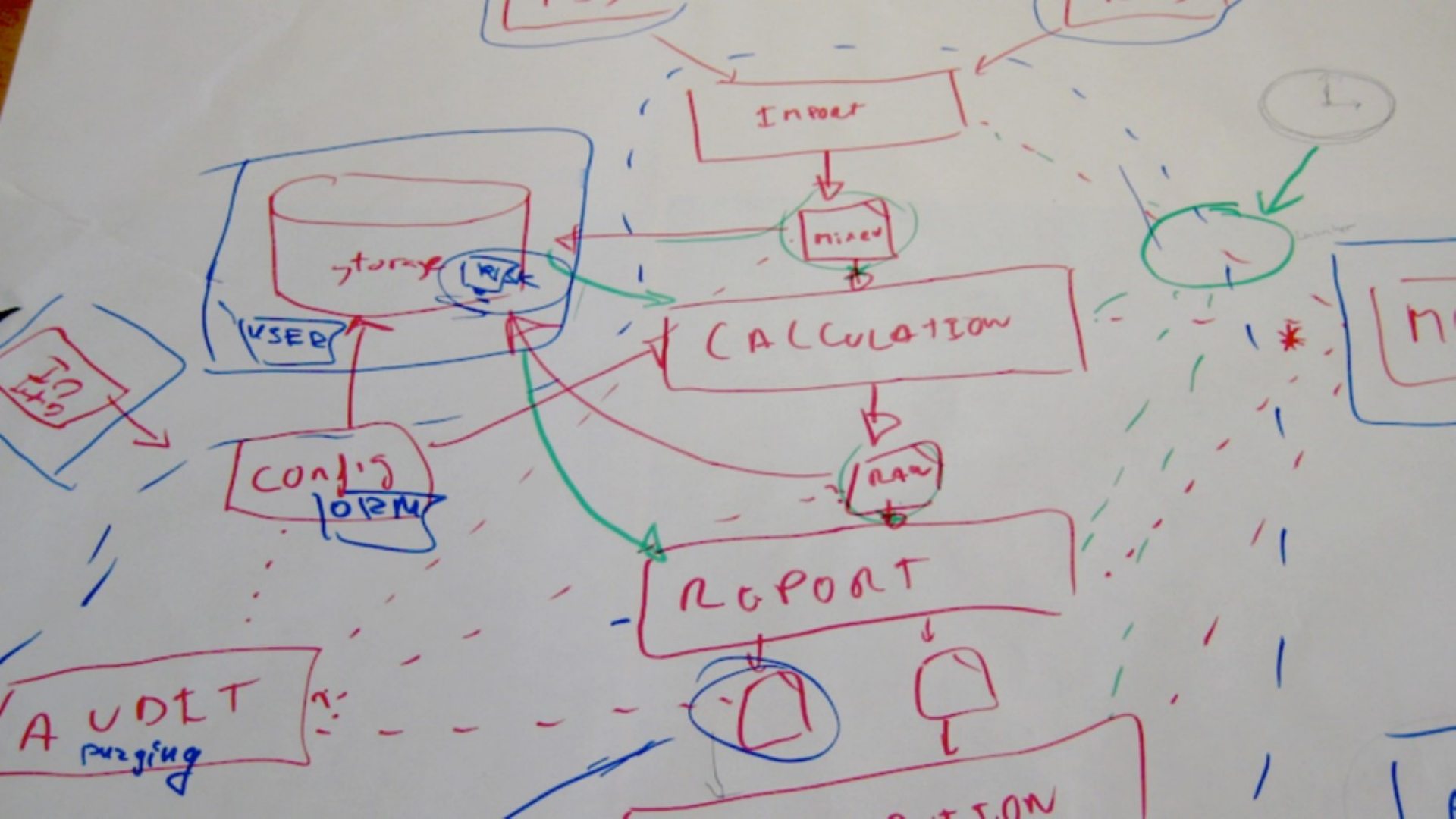
Visualising software architecture

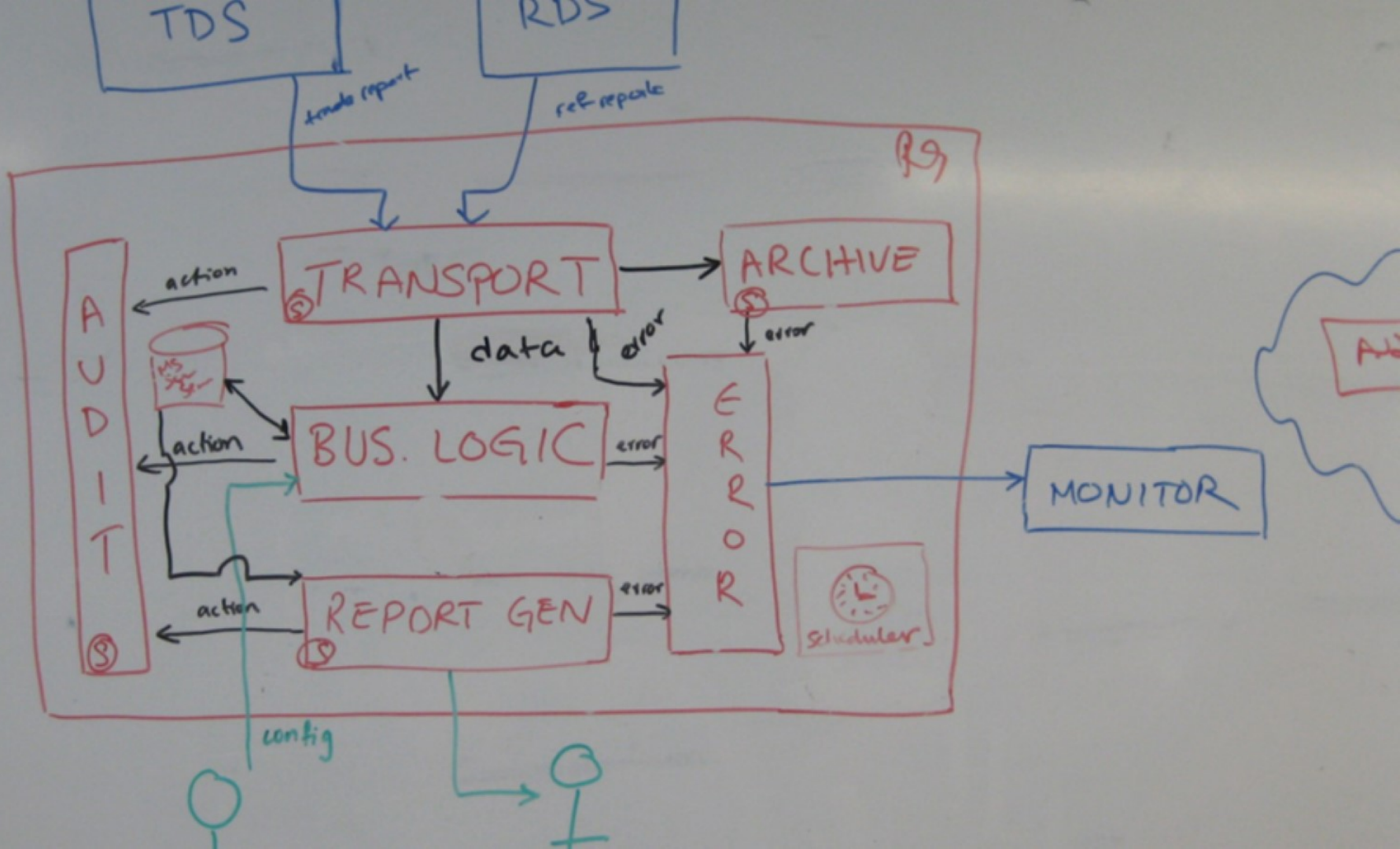
I work with teams around the world,
helping them to draw
software architecture diagrams

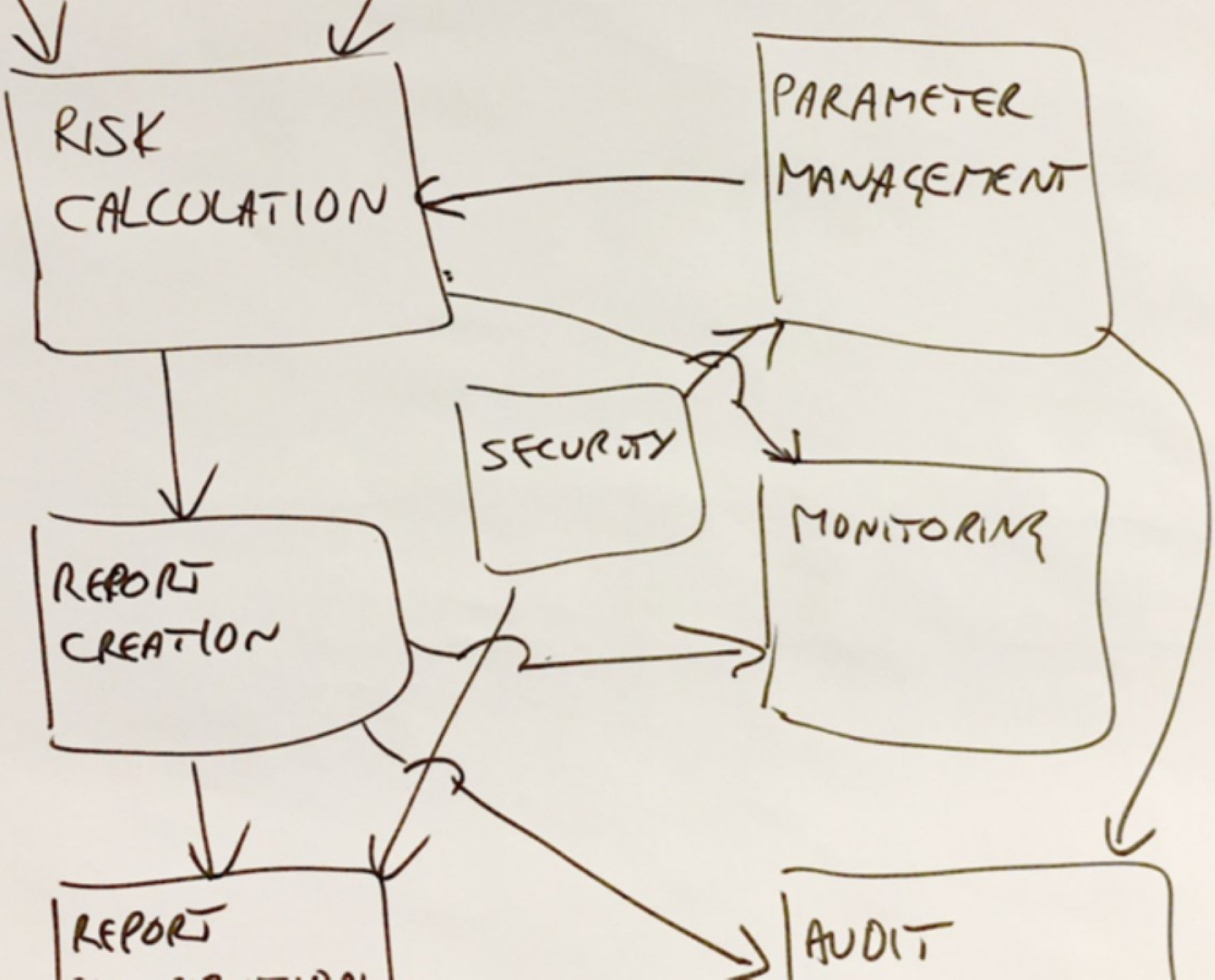

UNIX BOX
RS
TRANSPORT + LOGIC
JBOSS INSTANCE
ERROR
JBOSS INSTANCE (WEB CONTAINER ONLY)
WINDOWS BOX
MS
SQL
SERVER
MS
REPORTING SERVICE


ASP NET
LOGGING SERVICE
PARAMETER MANAGER
RISK CALCULATION
REPORT GENERATOR
DATA IMPORT
AUDITING
VALIDATION
TDS
RDS
RISK DATA
PARAMS
SECURITY
AUDIT

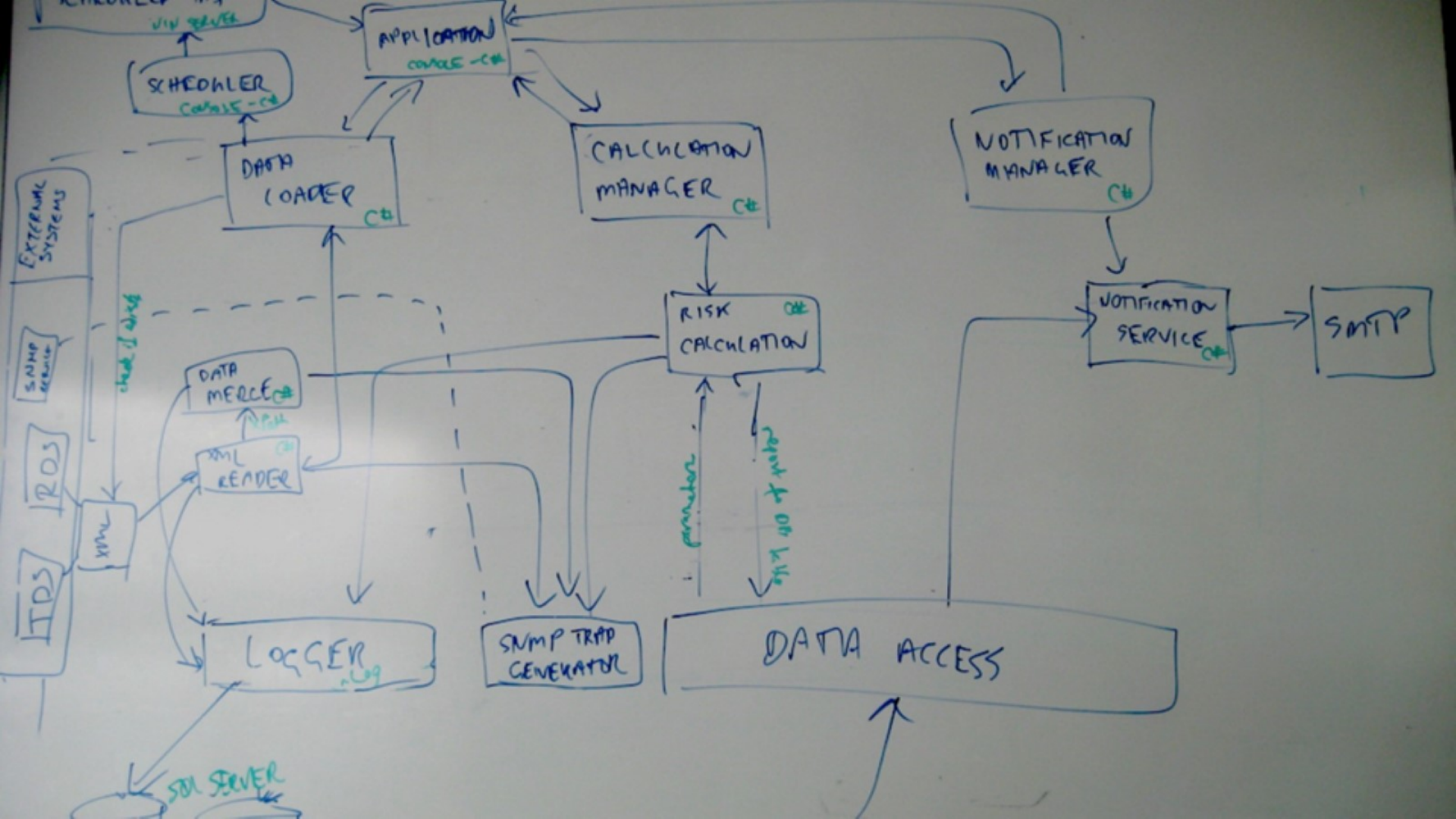
FUNCTIONAL VIEW

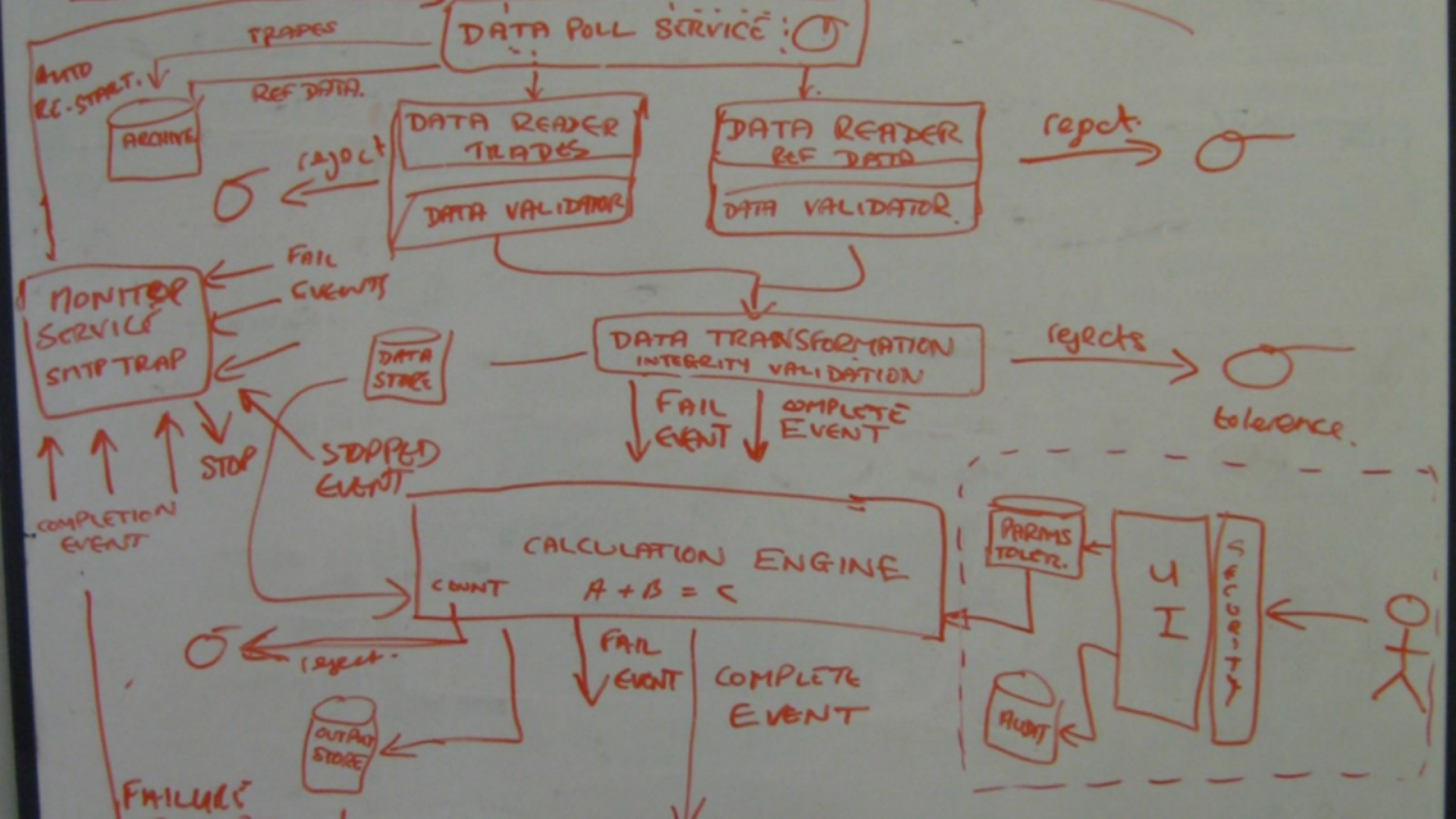

storage
CALCULATION
REPORT
AUDIT
purging

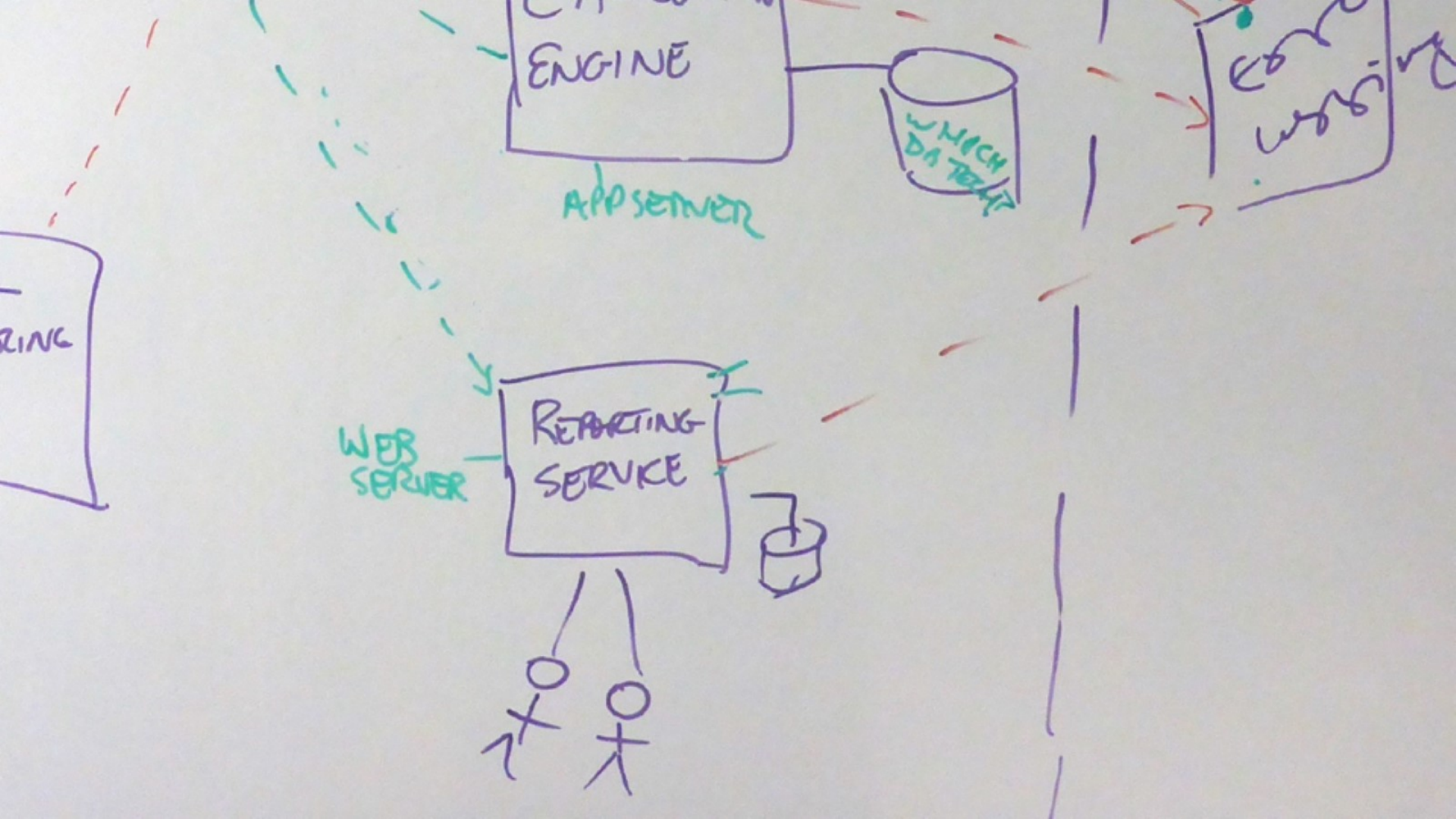

TDS
RDS
trade report
ref repos
TRANSPORT
ARCHIVE
action
data
error
error
AUDIT
action
BUS. LOGIC
error
ERROR
MONITOR
action
REPORT GEN
error
Scheduler
config


RISK
CALCULATION
PARAMETER
MANAGEMENT
SECURITY
MONITORING
REPORT
CREATION
AUDIT

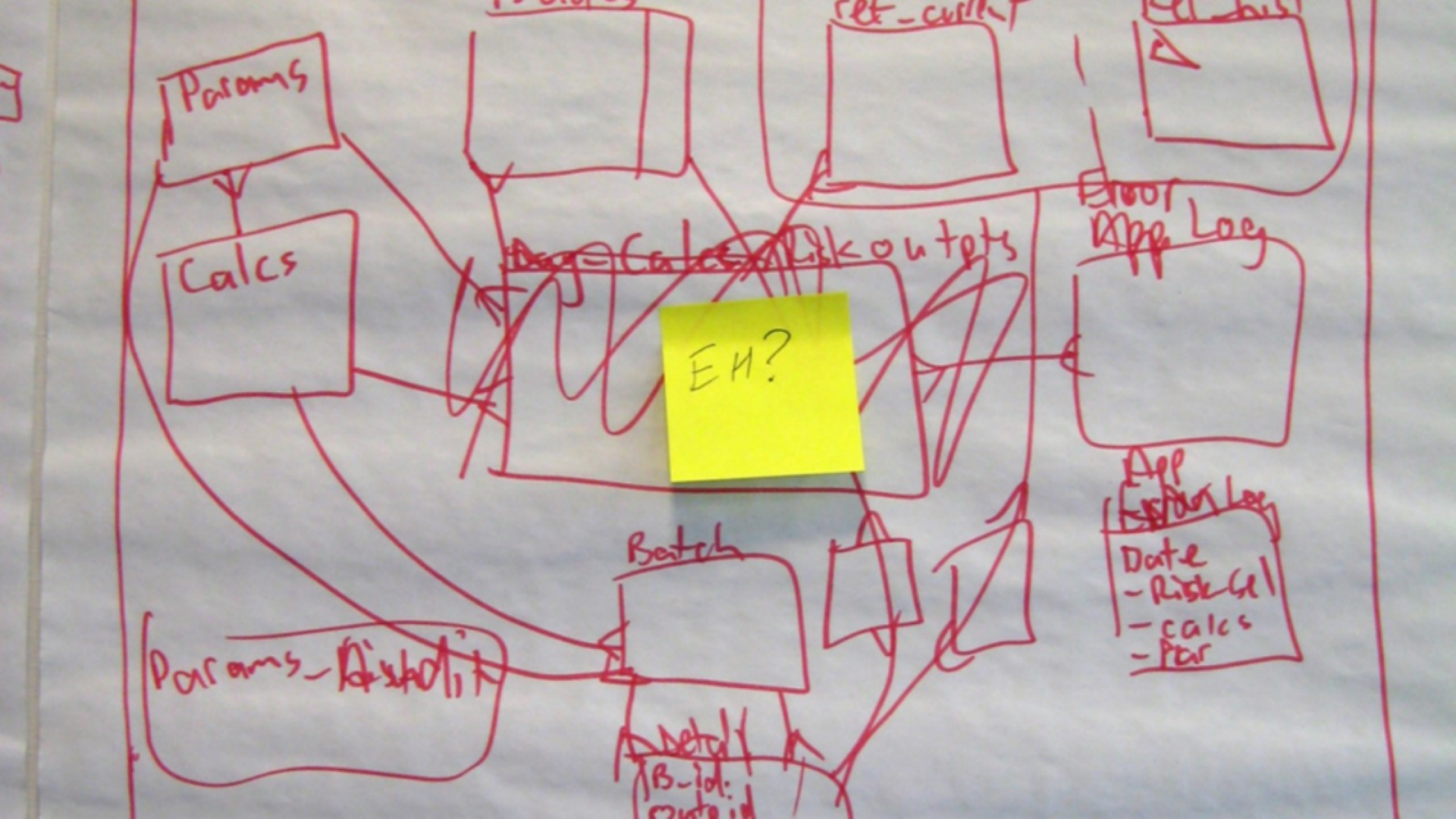

2
SERVICE/BATCH
IMPORT
JOIN
CALC
EXPORT/STORE
REPORT
SNMP
CENTRAL
MONITORING
SERVICE
3
WEBAPP
•GET REPORT
•EDIT
RISK RULES
•AUTH
•AUDIT
RISK RULES


SCHEDULER
Console - C#
APPLICATION
Console - C#
DATA LOADER
C#
CALCULATION MANAGER
C#
NOTIFICATION MANAGER
C#
EXTERNAL SYSTEMS
SNMP
RDS
TDS
RISK CALCULATION
C#
NOTIFICATION SERVICE
C#
SMTP
DATA MERGE
C#
XML READER
C#
XML
LOGGER
SNMP TRAP GENERATOR
DATA ACCESS
SQL SERVER


DATA POLL SERVICE
TRADES
REF DATA.
AUTO RE-START.
ARCHIVE
DATA READER TRADES
DATA VALIDATOR
DATA READER REF DATA
DATA VALIDATOR
reject
reject
MONITOR SERVICE SMTP TRAP
FAIL EVENTS
DATA STORE
DATA TRANSFORMATION INTEGRITY VALIDATION
reject
tolerance.
FAIL EVENT
COMPLETE EVENT
STOP
STOPPED EVENT
COMPLETION EVENT
CALCULATION ENGINE
COUNT
A + B = C
PARMS TOLER.
UI
SECURITY
reject
FAIL EVENT
COMPLETE EVENT
OUTPUT STORE
AUDIT
FAILURE


ENGINE
APP SERVER
REPORTING
SERVICE
WEB
SERVER

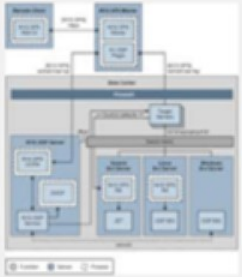
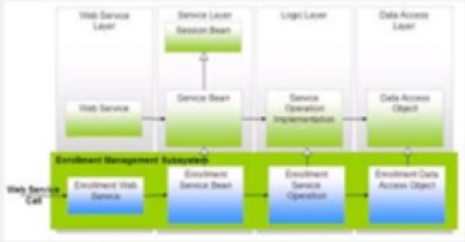

Data Normalizer
Report Monitor
(other machine)
Risk Calculator
Aggregator
CP Risk Calc
Report Generator
Auditing Service
Report Publisher
Report Repository

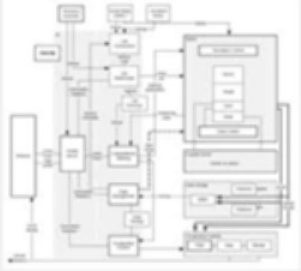
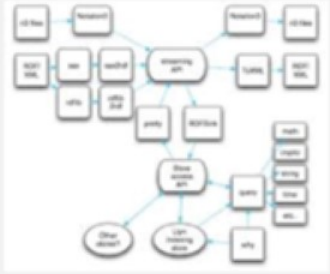

Params
Calcs
EH?
Batch
Date
- calcs

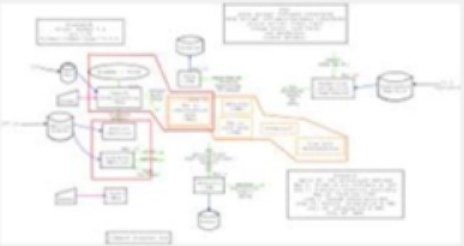
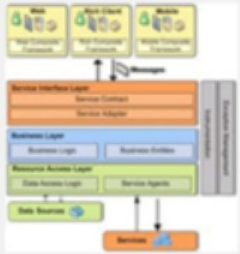
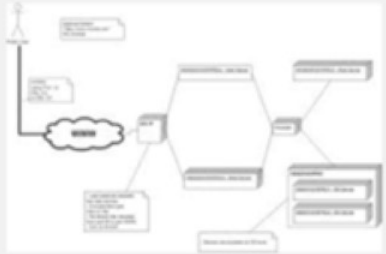
Google software architecture diagram

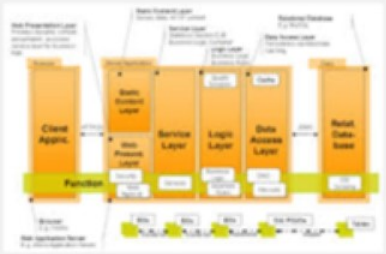
Sign in

Web Images Videos Shopping News More Search tools SafeSearch

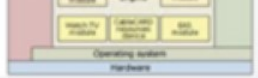



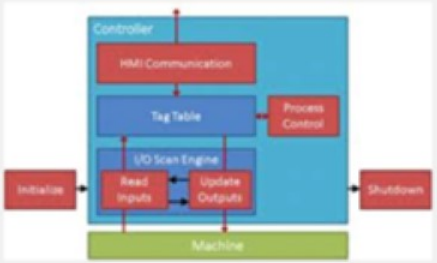




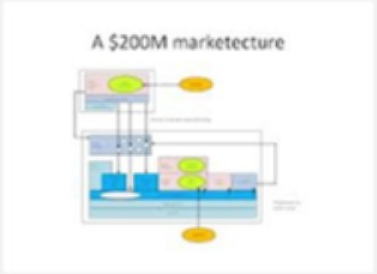


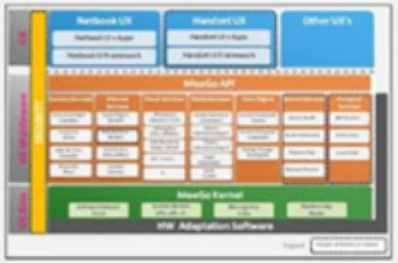




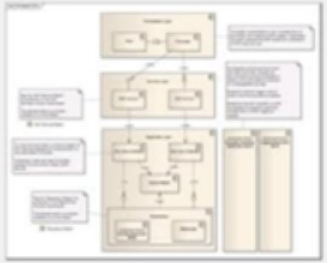


Operating system
Hardware

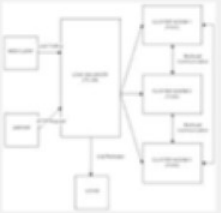

Controller
HMI Communication
Tag Table
Process Control
I/O Scan Engine
Initialize
Read Inputs
Update Outputs
Shutdown
Machine

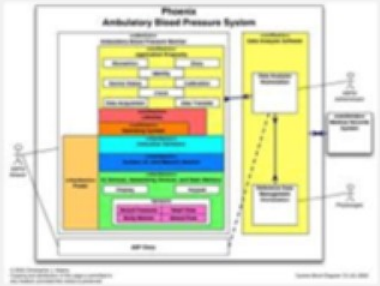

A $200M marketecture

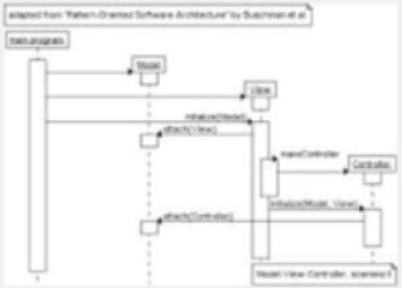

NetBook UX
Handset UX
Other UX's
MeeGo API
MeeGo Kernel
HW Adaptation Software

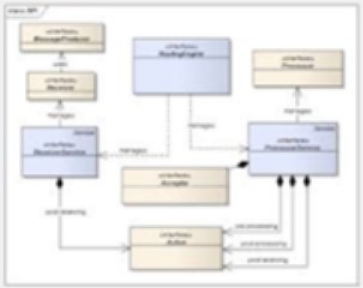

Customer Site
Service Organization






Phoenix
Ambulatory Blood Pressure System

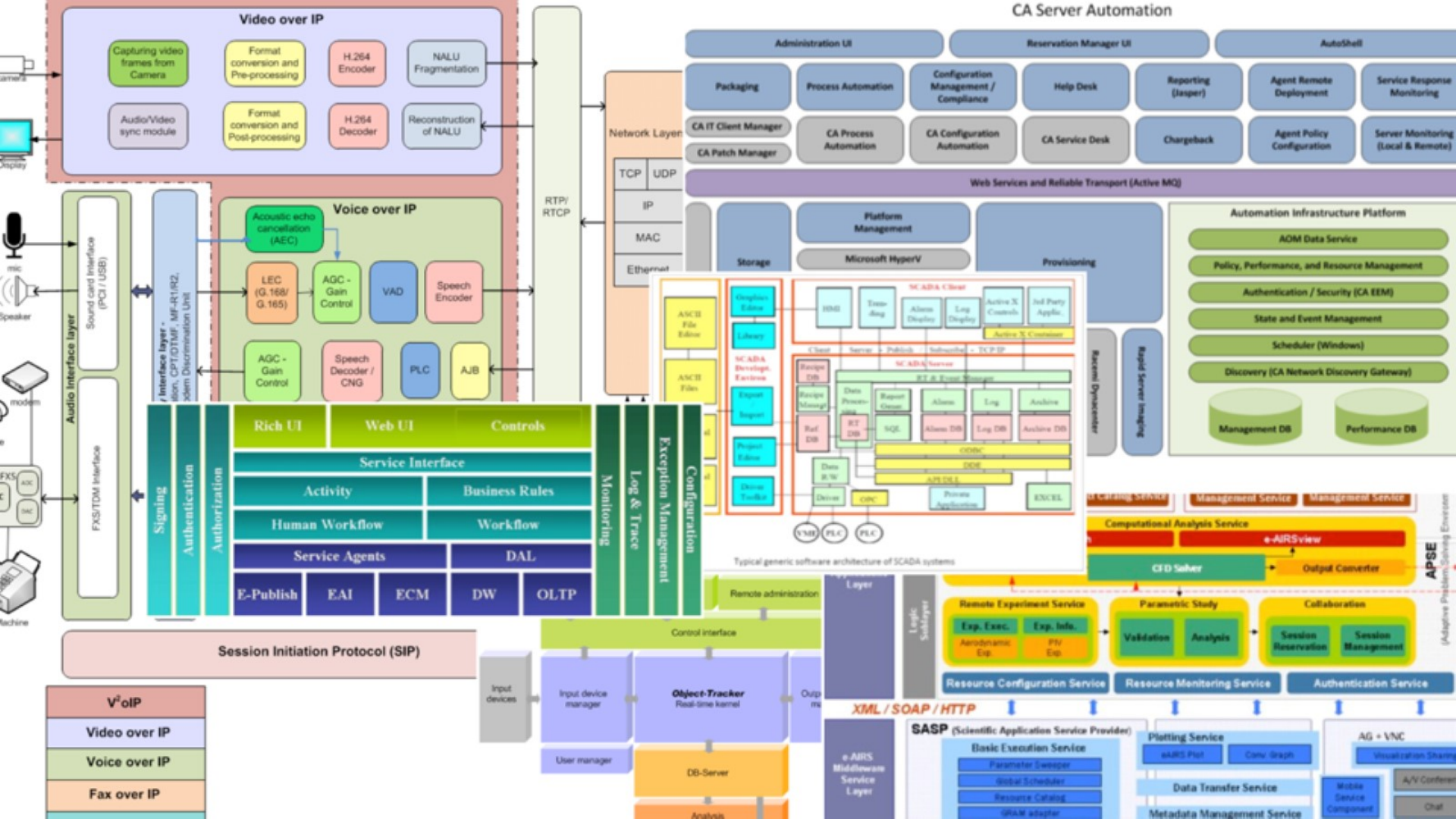





Transient Objects
Application
Persistent Objects
SessionFactory
Session
Transaction


Presentation
ASPX, JSP, HTML
Servlet, Controllers
WinForm, WPF, Swing


The Software Architecture of Global Climate Models


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
Voice over IP
Acoustic echo cancellation (AEC)
LEC (G.168/ G.165)
AGC - Gain Control
VAD
Speech Encoder
AGC - Gain Control
Speech Decoder / CNG
PLC
AJB
Audio Interface layer
Sound card Interface (PCI / USB)
FXS/TDM Interface
RTP/ RTCP
Network Layer
TCP
UDP
IP
MAC
Ethernet
CA Server Automation
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
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
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
Typical generic software architecture of SCADA systems
Computational Analysis Service
e-AIRSview
CFD Solver
Output Converter
APSE
Remote Experiment Service
Exp. Exec.
Exp. Info.
Aerodynamic Exp.
FV Exp.
Parametric Study
Validation
Analysis
Collaboration
Session Reservation
Session Management
Resource Configuration Service
Resource Monitoring Service
Authentication Service
XML / SOAP / HTTP
SASP (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 Device Component
A/V Conferen
Chat
Session Initiation Protocol (SIP)
Remote administration
Control interface
Input devices
Input device manager
Object-Tracker
Real-time kernel
User manager
DB-Server
Analysis
e-AIRS Middleware Service Layer
Logic Sublayer
V²oIP
Video over IP
Voice over IP
Fax over IP

Moving fast in the same direction
as a team requires
good communication

In my experience, software architects struggle to communicate software architecture

Do **you** use UML?

Poll

Do you use UML?

Yes

No

In my experience, optimistically,

1 out of 10 people use UML

Who are the **stakeholders** that you need to communicate software architecture to; what **information** do they need?

There are many **different audiences** for diagrams and documentation, all with **different interests**

(software architects, software developers, operations and support staff, testers, Product Owners, project managers, Scrum Masters, users, management, business sponsors, potential customers, potential investors, ...)

The primary use for diagrams and documentation is **communication** and **learning**

Software architecture diagrams,
when connected to the code,
are also a fantastic tool for
architectural improvement

To describe a software architecture, we use a model composed of multiple views or perspectives.

Architectural Blueprints - The “4+1” View Model of Software Architecture
Philippe Kruchten

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.

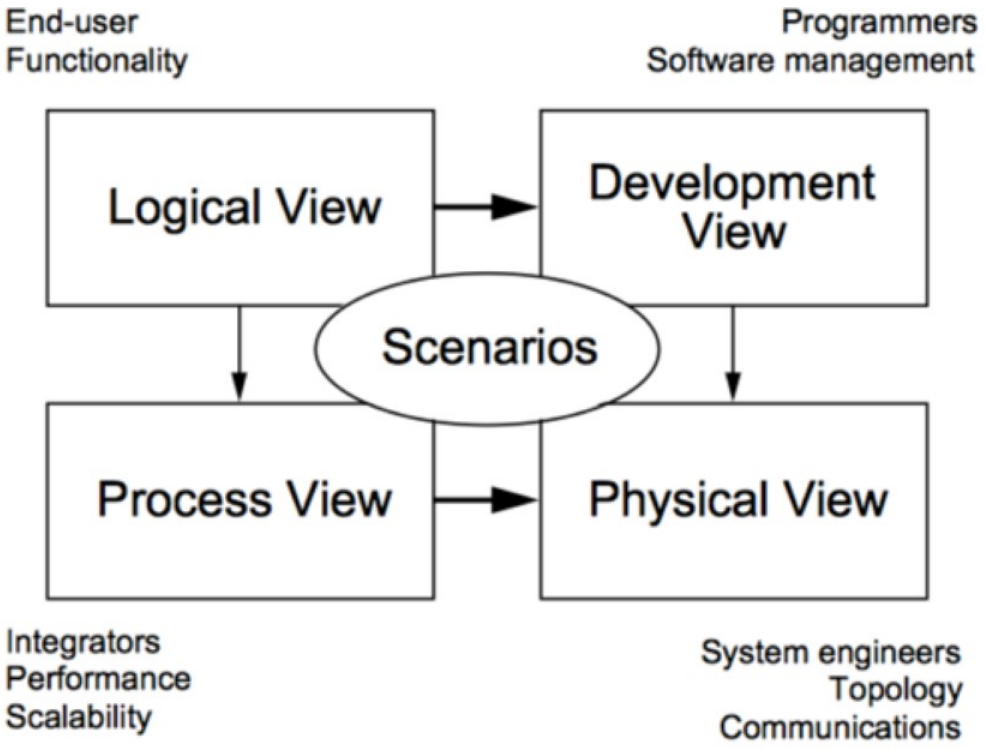
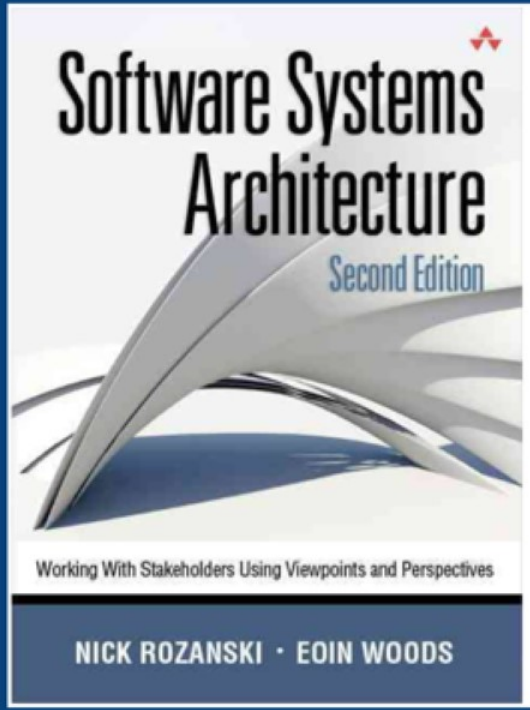
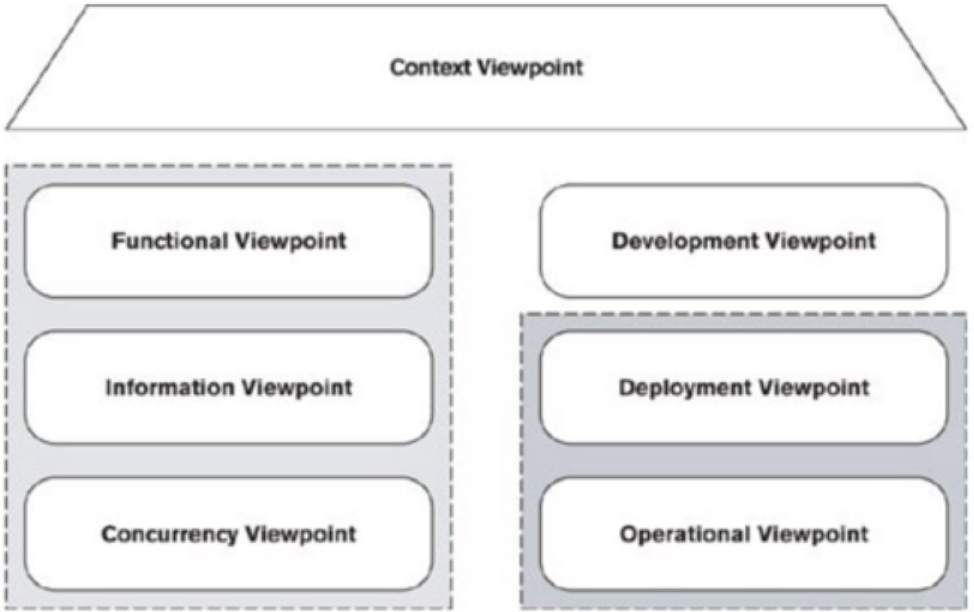


Figure 1 — The "4+1" view model

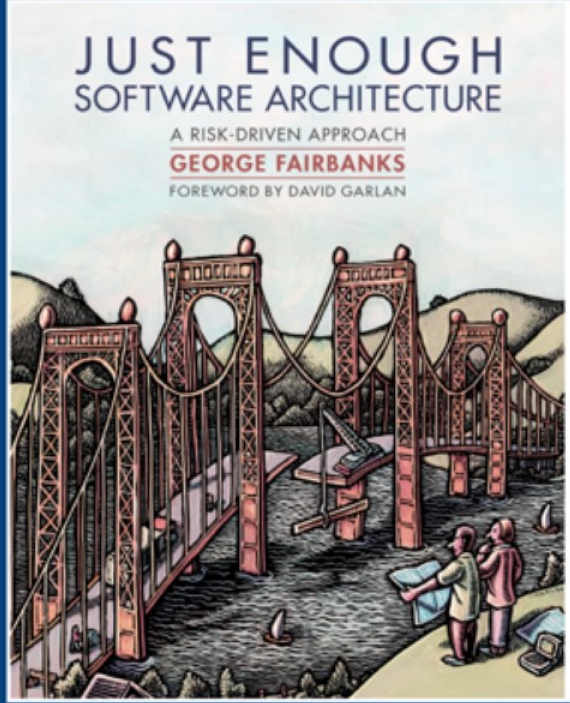




“Viewpoints and Perspectives”

Why is there a separation between the **logical** and **development** views?

Our architecture diagrams don’t match the code.



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.

“model-code gap”

Software Reflexion Models:
Bridging the Gap between Source and High-Level Models*

Gail C. Murphy and David Notkin

Dept. of Computer Science & Engineering
University of Washington
Box 352350
Seattle WA, USA 98195-2350
{gmurphy, notkin}@cs.washington.edu

Kevin Sullivan

Dept. of Computer Science
University of Virginia
Charlottesville VA, USA 22903
sullivan@cs.virginia.edu

Abstract

Software engineers often use high-level models (for instance, box and arrow sketches) to reason and communicate about an existing software system. One problem with high-level models is that they are almost always inaccurate with respect to the system's source code. We have developed an approach that helps an engineer use a high-level model of the structure of an existing software system as a lens through which to see a model of that system's source code. In particular, an engineer defines a high-level model and specifies how the model maps to the source. A tool then computes a software reflexion model that shows where the engineer's high-level model agrees with and where it differs from a model of the source.

The paper provides a formal characterization of reflexion models, discusses practical aspects of the approach, and relates experiences of applying the approach and tools to a number of different systems. The illustrative example used in the paper describes the application of reflexion models to NetBSD, an implementation of Unix comprised of 250,000 lines of C code. In only a few hours, an engineer computed several reflexion models that provided him with a useful, global overview of the structure of the NetBSD virtual memory subsystem. The approach has also been applied to aid in the understanding and experimental reengineering of the Microsoft Excel spreadsheet product.

*This research was funded in part by the NSF grant CCR-8858804 and a Canadian NSERC post-graduate scholarship.

SIGSOFT '95 Washington, D.C., USA
©1995 ACM 0-89791-716-3/95/0010...$3.50

1 Introduction

Software engineers often think about an existing software system in terms of high-level models. Box and arrow sketches of a system, for instance, are often found on engineers' whiteboards. Although these models are commonly used, reasoning about the system in terms of such models can be dangerous because the models are almost always inaccurate with respect to the system's source.

Current reverse engineering systems derive high-level models from the source code. These derived models are useful because they are, by their very nature, accurate representations of the source. Although accurate, the models created by these reverse engineering systems may differ from the models sketched by engineers; an example of this is reported by Wong et al. [WTMS95].

We have developed an approach, illustrated in Figure 1, that enables an engineer to produce sufficiently accurate high-level models in a different way. The engineer defines a high-level model of interest, extracts a source model (such as a call graph or an inheritance hierarchy) from the source code, and defines a declarative mapping between the two models. A *software reflexion model* is then computed to determine where the engineer's high-level model does and does not agree with the source model.[1] An engineer interprets the reflexion model and, as necessary, modifies the input to iteratively compute additional reflexion models.

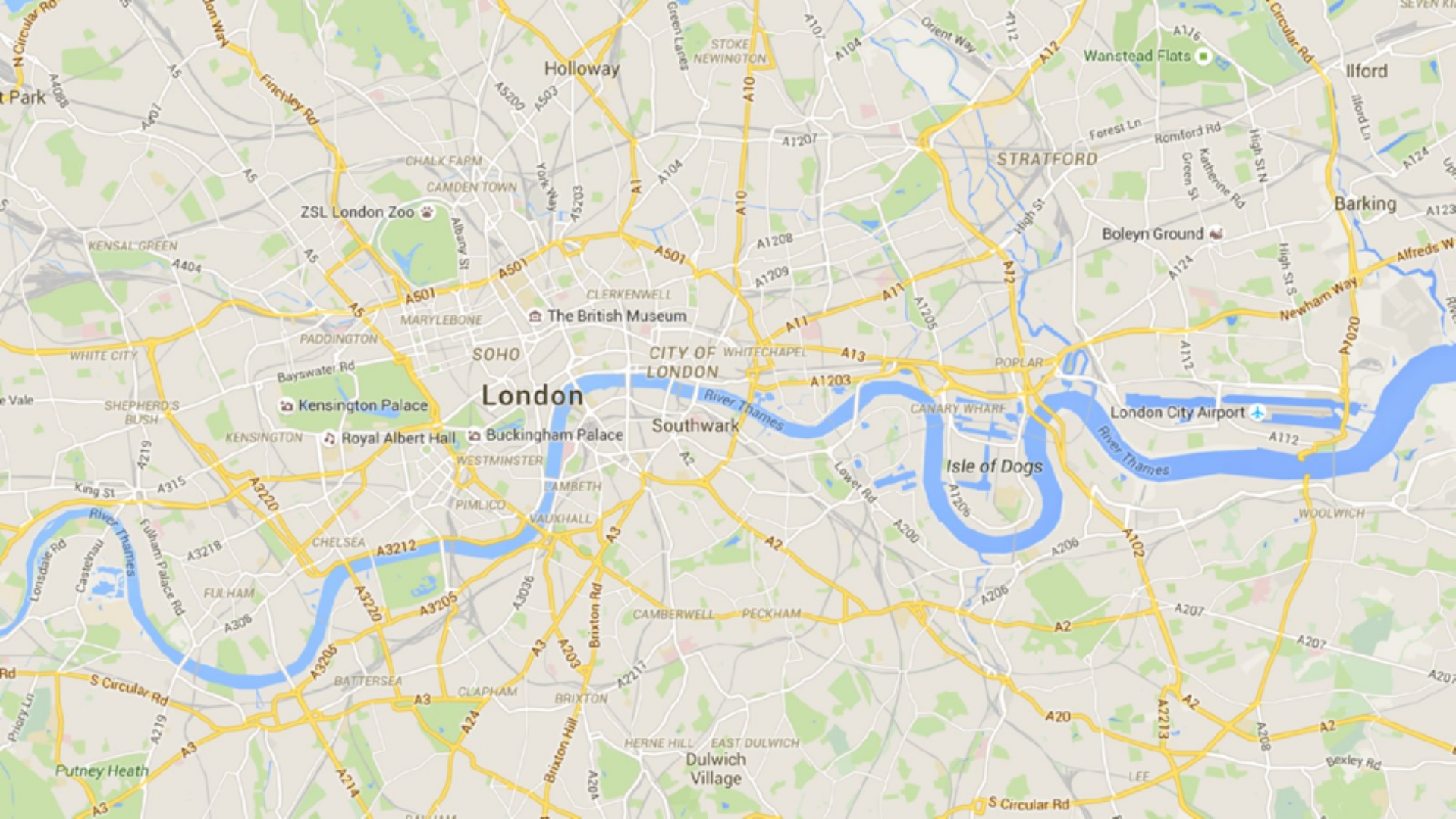
[1] The old English spelling differentiates our use of "reflexion" from the field of reflective computing [Smi84].

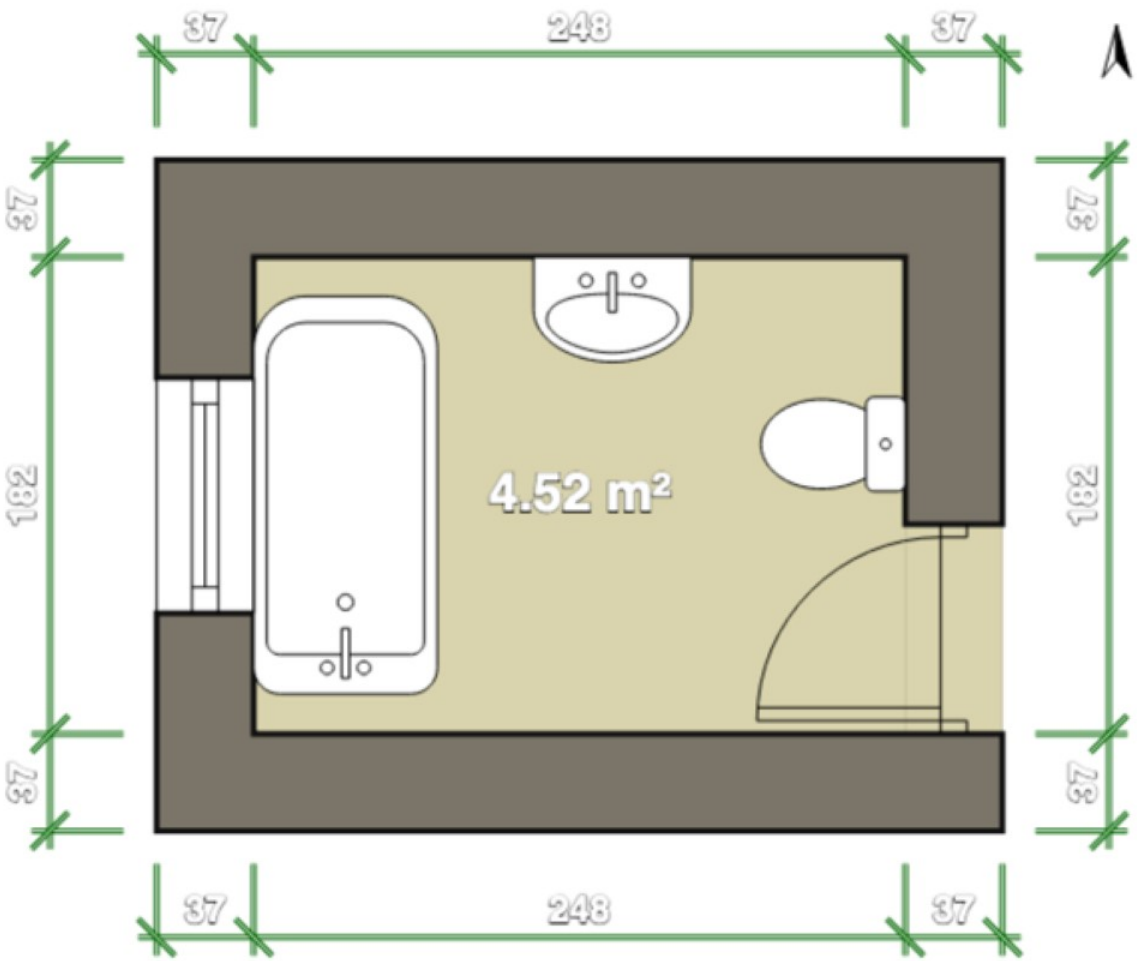
1 Introduction

Software engineers often think about an existing software system in terms of high-level models. Box and arrow sketches of a system, for instance, are often found on engineers' whiteboards. Although these models are commonly used, reasoning about the system in terms of such models can be dangerous because the models are almost always inaccurate with respect to the system's source.

Current reverse engineering systems derive high-level models from the source code. These derived models are useful because they are, by their very nature, accurate representations of the source. Although accurate, the models created by these reverse engineering systems may differ from the models sketched by engineers; an example of this is reported by Wong et al. [WTMS95].

We lack a **common vocabulary**
to describe software architecture


Holloway
Wanstead Flats
Ilford
Park
STRATFORD
Barking
ZSL London Zoo
Boleyn Ground
The British Museum
SOHO
CITY OF LONDON
London
Kensington Palace
Royal Albert Hall
Buckingham Palace
Southwark
London City Airport
Isle of Dogs
River Thames
Dulwich Village
Putney Heath
CHALK FARM
CAMDEN TOWN
KENSAL GREEN
CLERKENWELL
MARYLEBONE
PADDINGTON
WHITE CITY
SHEPHERD'S BUSH
KENSINGTON
WESTMINSTER
LAMBETH
PIMLICO
VAUXHALL
CHELSEA
FULHAM
BATTERSEA
CLAPHAM
BRIXTON
CAMBERWELL
PECKHAM
HERNE HILL
EAST DULWICH
WHITECHAPEL
POPLAR
CANARY WHARF
WOOLWICH
LEE
A1
A2
A3
A4
A5
A10
A11
A12
A13
A20
A40
A102
A104
A107
A112
A116
A124
A200
A201
A202
A203
A204
A205
A206
A207
A208
A213
A214
A219
A220
A315
A400
A401
A404
A406
A501
A503
A1020
A1200
A1203
A1205
A1206
A1207
A1208
A1209
A3036
A3205
A3212
A3218
A3220
A4088
A5203
A5209
Finchley Rd
Green Lanes
York Way
Albany St
Bayswater Rd
Forest Ln
Romford Rd
Katherine Rd
Green St
High St
High St N
High St S
Ilford Ln
Alfreds W
Newham Way
Orient Way
Lower Rd
Brixton Rd
Brixton Hill
Lonsdale Rd
Castelnau
Fulham Palace Rd
King St
Priory Ln
Bexley Rd
N Circular Rd
S Circular Rd


37
248
37
43
281
43
4.52 m²
37
248
37

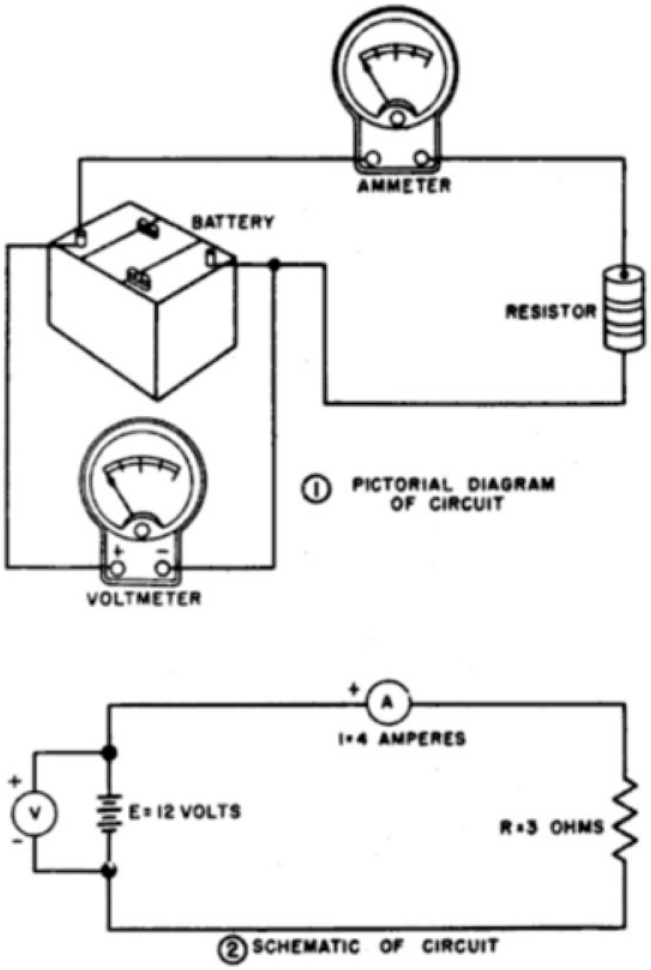
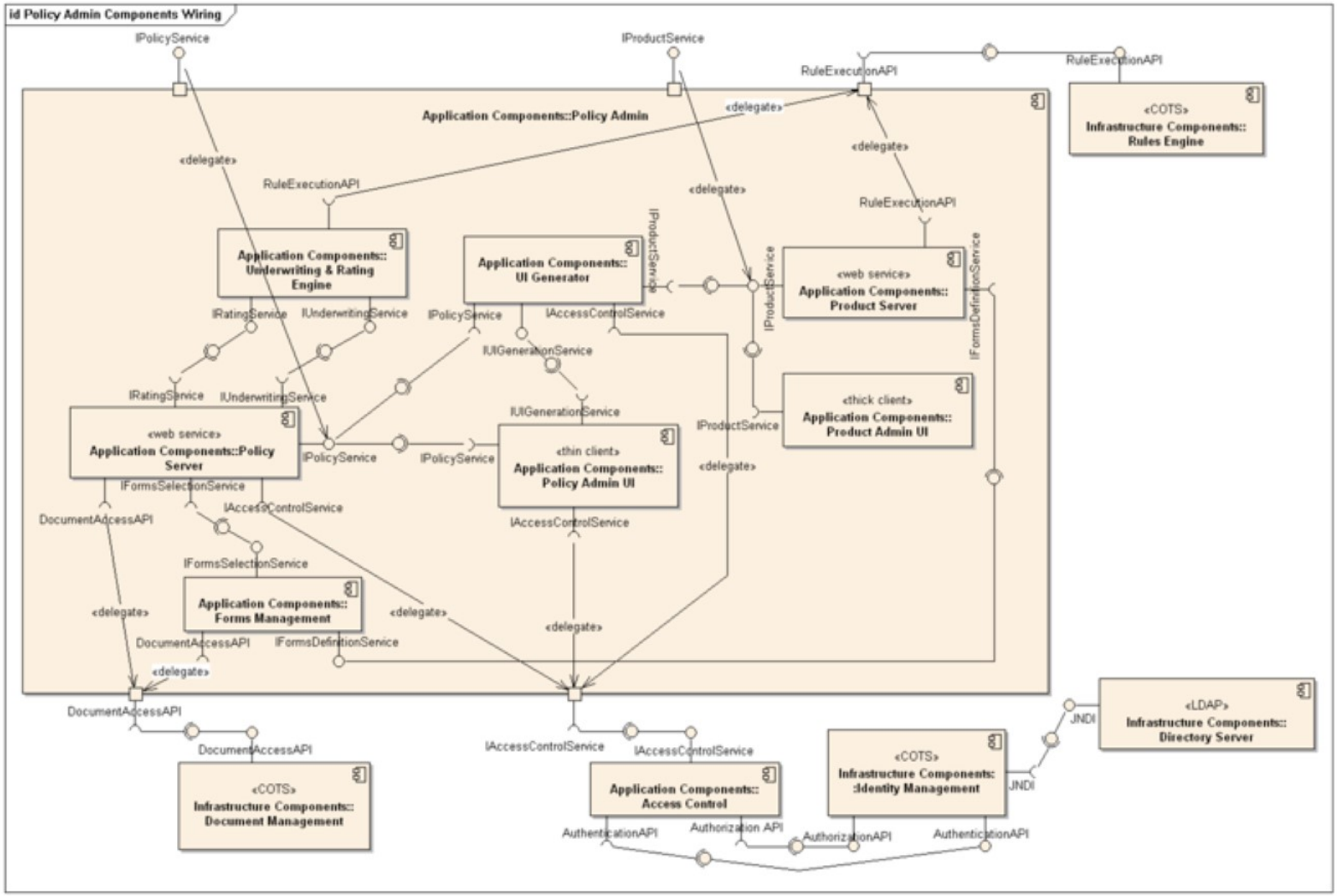
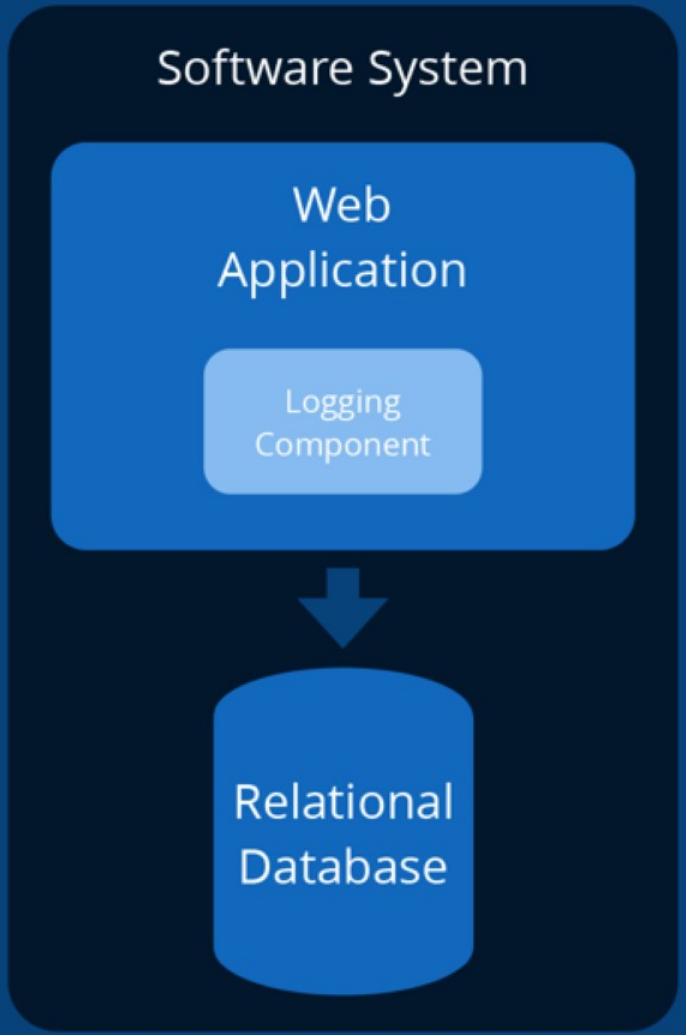


Figure 48. Diagram of a basic circuit.

https://en.wikipedia.org/wiki/Circuit_diagram



https://en.wikipedia.org/wiki/Component_diagram



[1] component

noun | com·po·nent | \kəm-ˈpō-nənt, ˈkäm-ˌ, käm-ˈ\

Simple Definition of COMPONENT

Popularity: Top 30% of words

: one of the parts of something (such as a system or mixture) : an important piece of something

Source: Merriam-Webster's Learner's Dictionary

Ubiquitous language

Abstractions

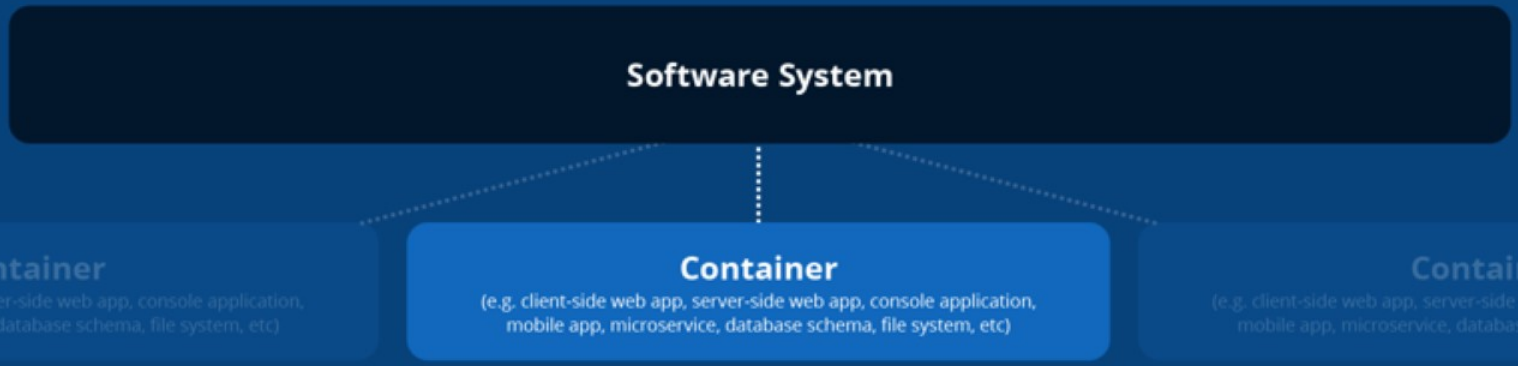
Would you code it that way?

(ensure that your diagrams reflect
your implementation intent)

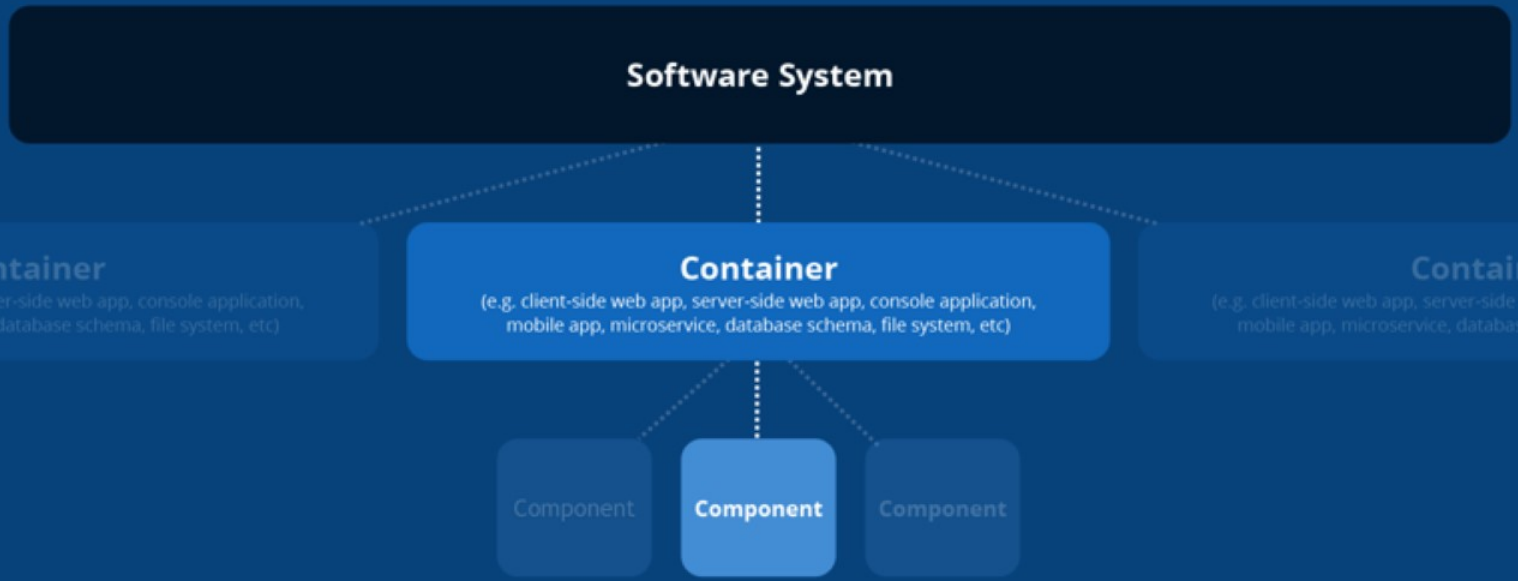
When drawing software architecture diagrams, think like a software developer

A **common set of abstractions**
is more important
than a common notation

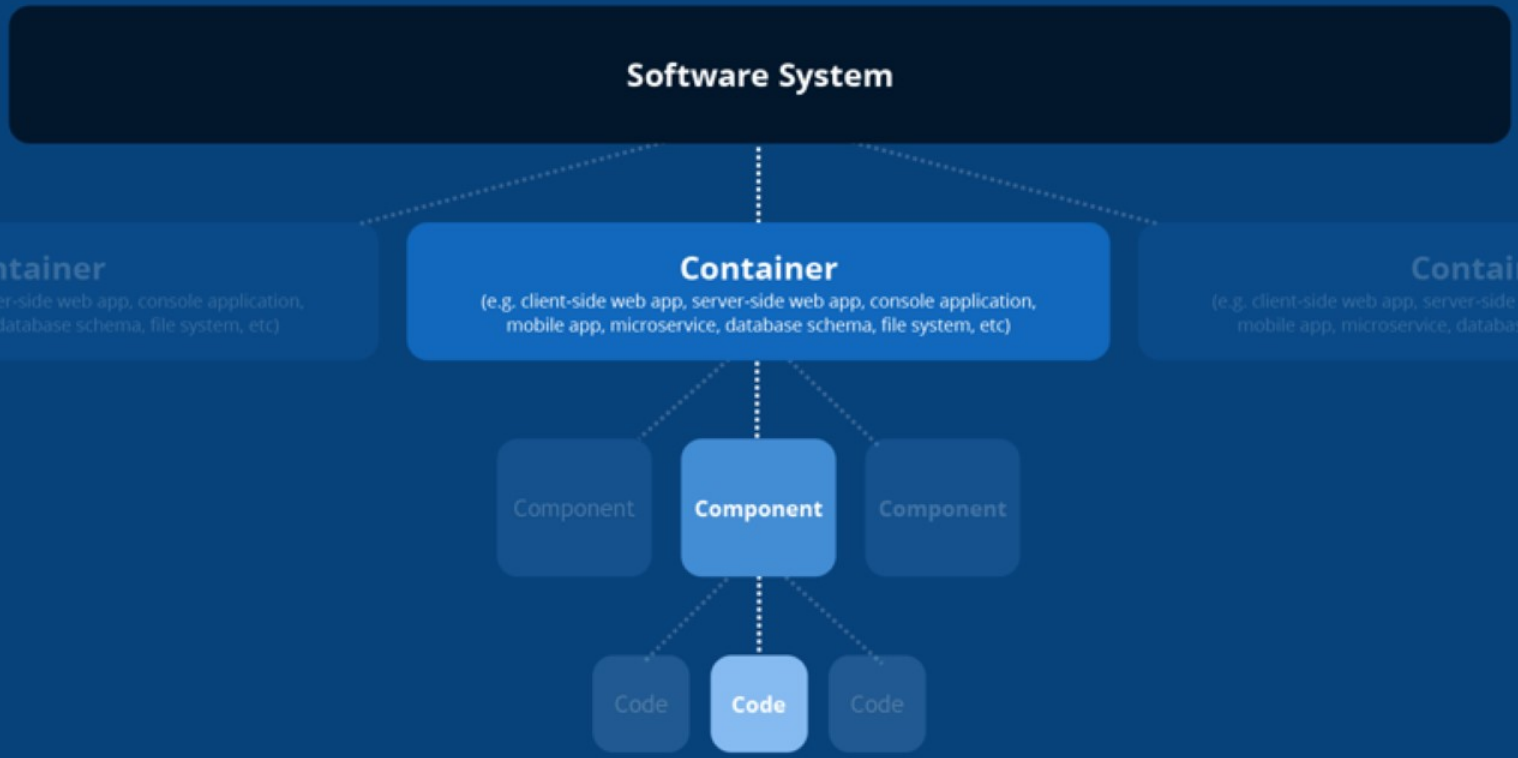
A **software system** is made up of one or more **containers**,
each of which contains one or more **components**,
which in turn are implemented by one or more **code elements**.



A **software system** is made up of one or more **containers**,
each of which contains one or more **components**,
which in turn are implemented by one or more **code elements**.



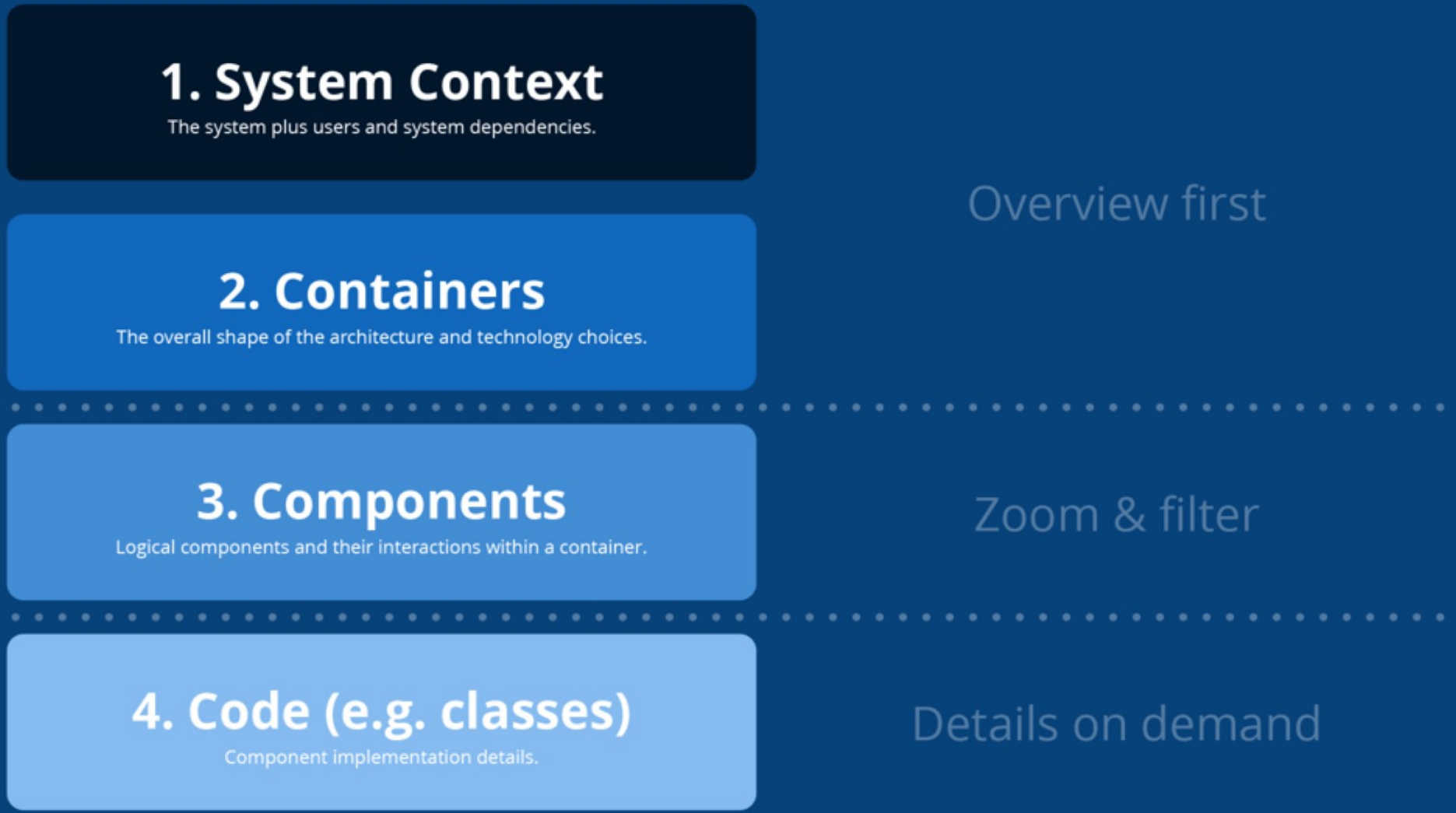
A **software system** is made up of one or more **containers**,
each of which contains one or more **components**,
which in turn are implemented by one or more **code elements**.



A **software system** is made up of one or more **containers**,
each of which contains one or more **components**,
which in turn are implemented by one or more **code elements**.

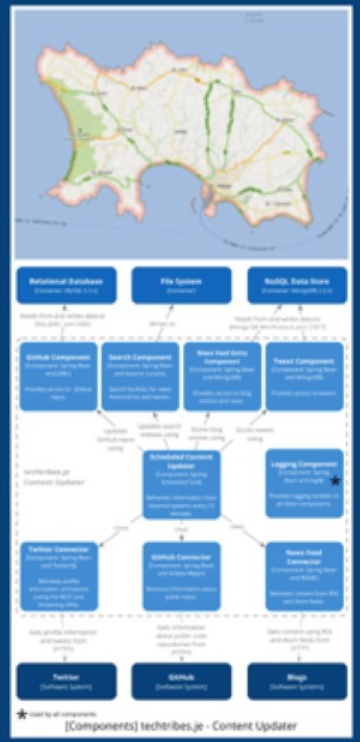
C4

c4model.com

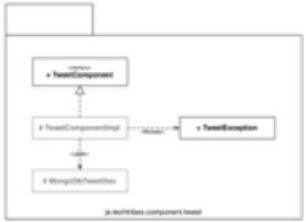

1. System Context
The system plus users and system dependencies.
2. Containers
The overall shape of the architecture and technology choices.
3. Components
Logical components and their interactions within a container.
4. Code (e.g. classes)
Component implementation details.
Overview first
Zoom & filter
Details on demand

[System Context] techtribes.je

[Containers] techtribes.je



[Components] techtribes.je - Content Updater



Diagrams are maps

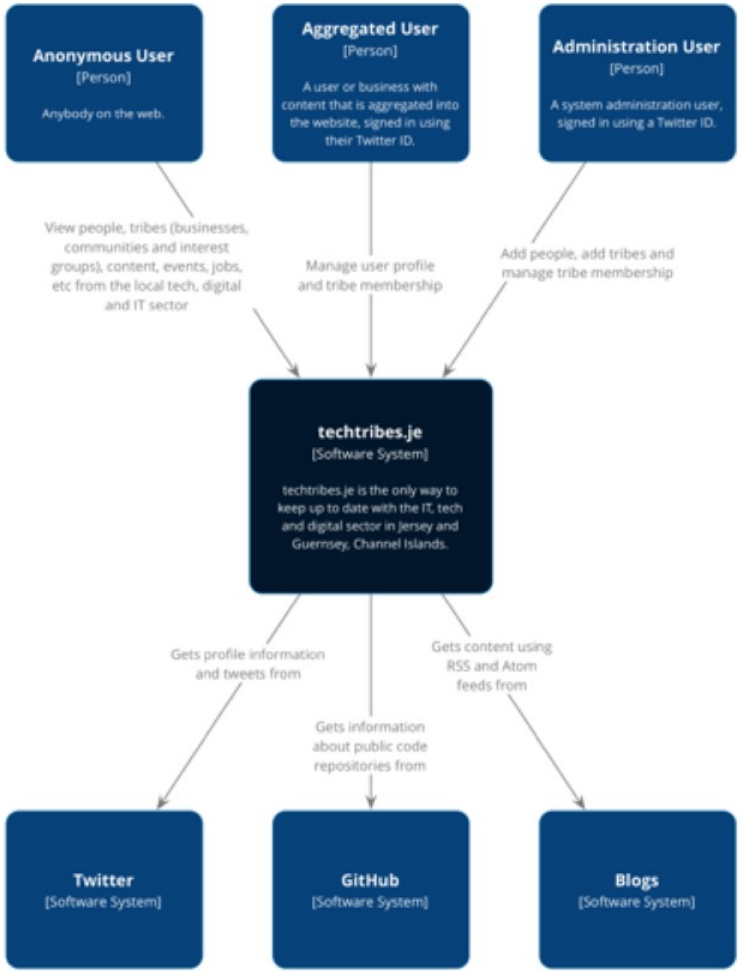
that help software developers navigate a large and/or complex codebase

Example

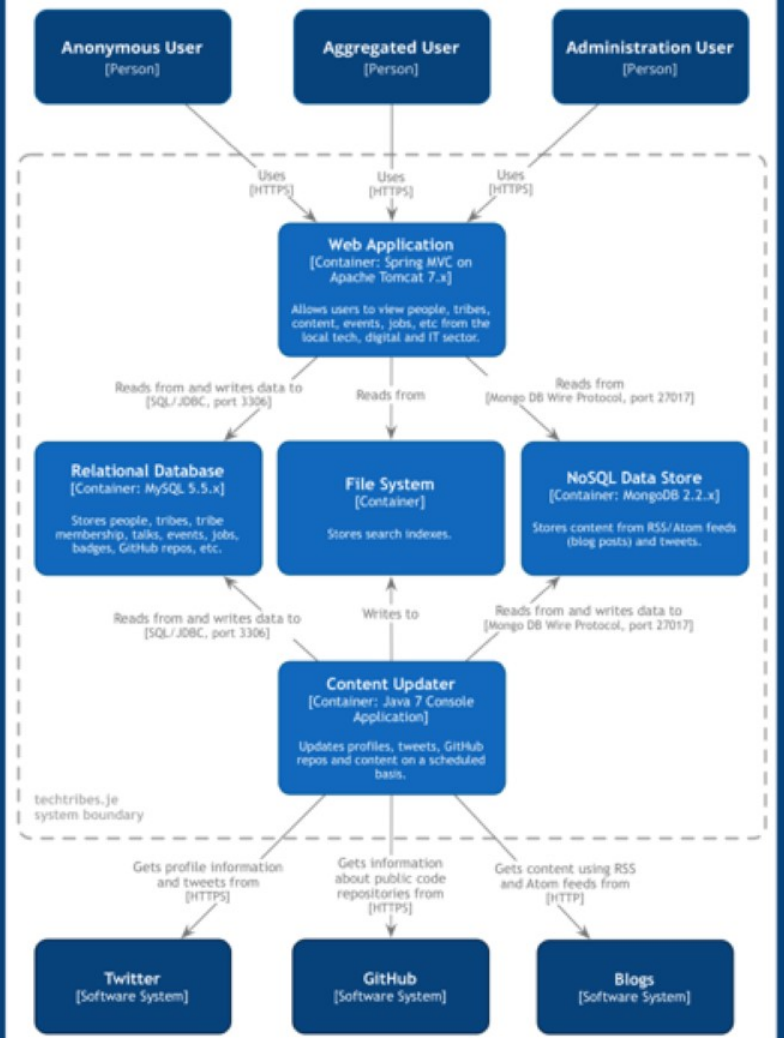
(techtribes.je)

techtribes.je

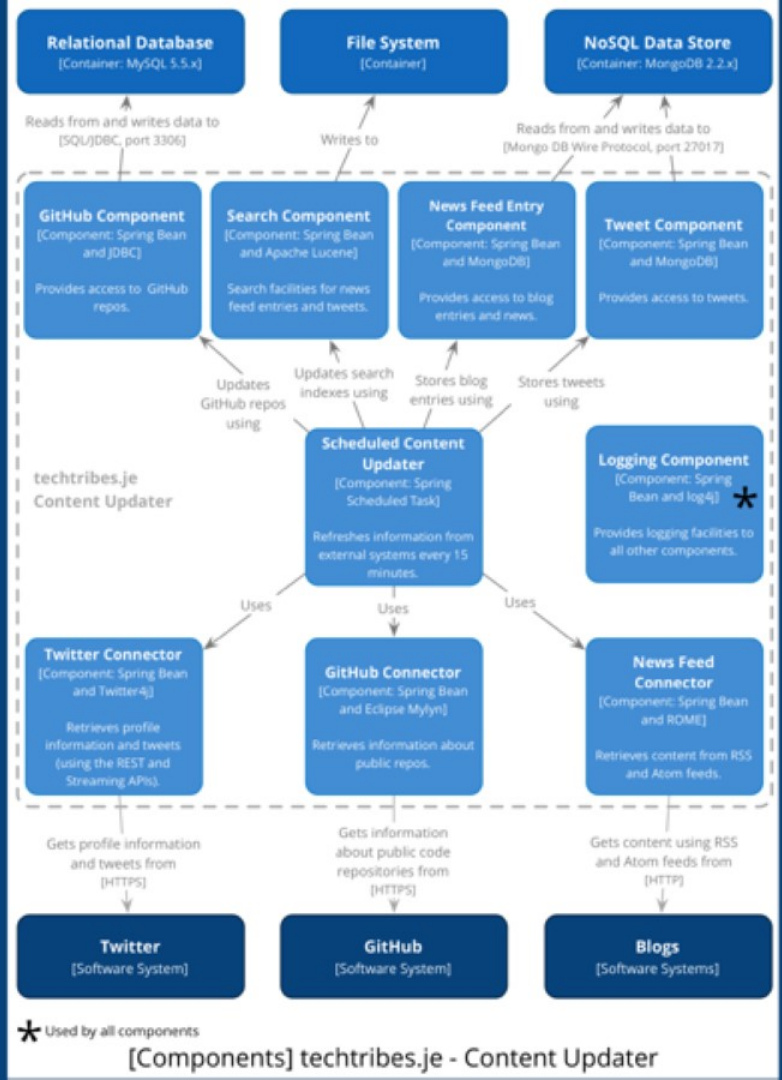
A simple content aggregator for the local tech and digital industry



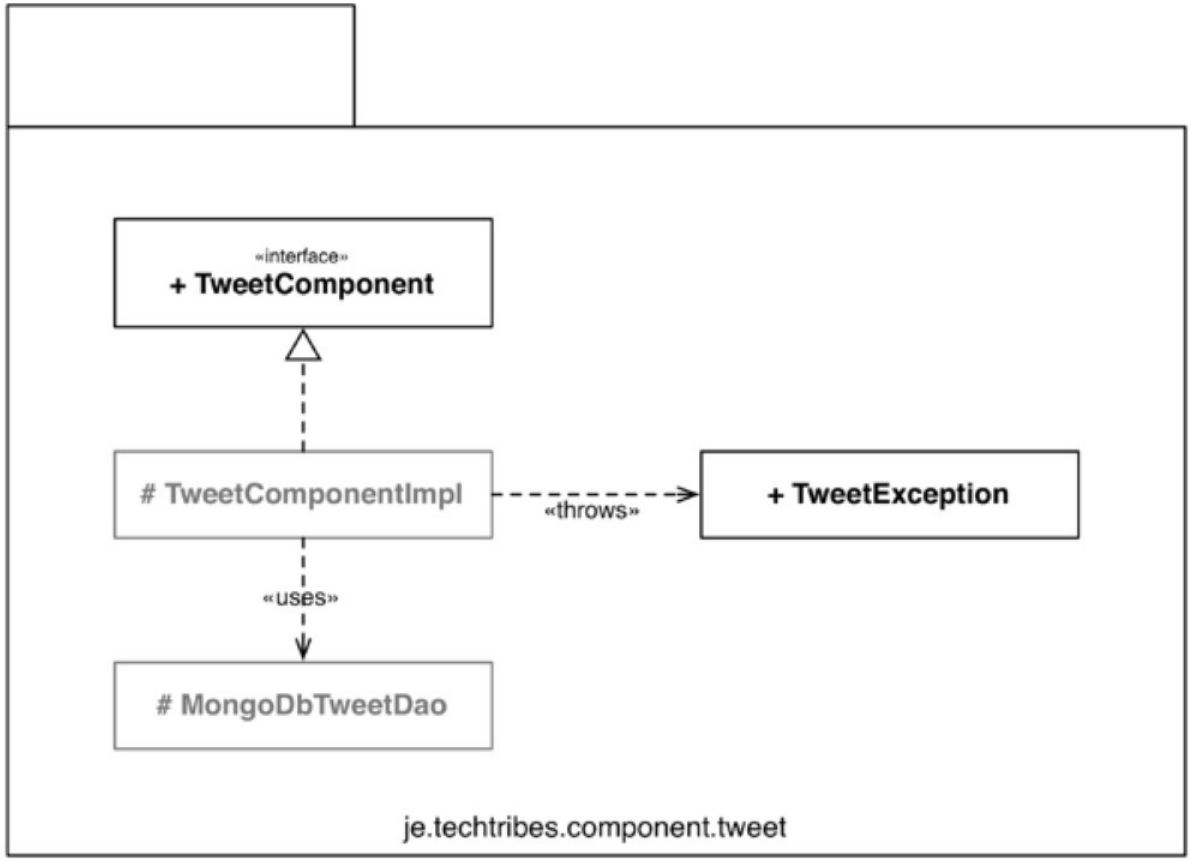
[System Context] techtribes.je



[Containers] techtribes.je



[Components] techtribes.je - Content Updater


«interface»
+ TweetComponent
# TweetComponentImpl
«throws»
+ TweetException
«uses»
MongoDbTweetDao
je.techtribes.component.tweet

Notation

Elements

Start with simple boxes containing the element name, type, technology (if appropriate) and a description/responsibilities

Anonymous User

[Person]

Anybody on the web.

techtribes.je

[Software System]

techtribes.je is the only way to keep up to date with the IT, tech and digital sector in Jersey and Guernsey, Channel Islands.

Web Application

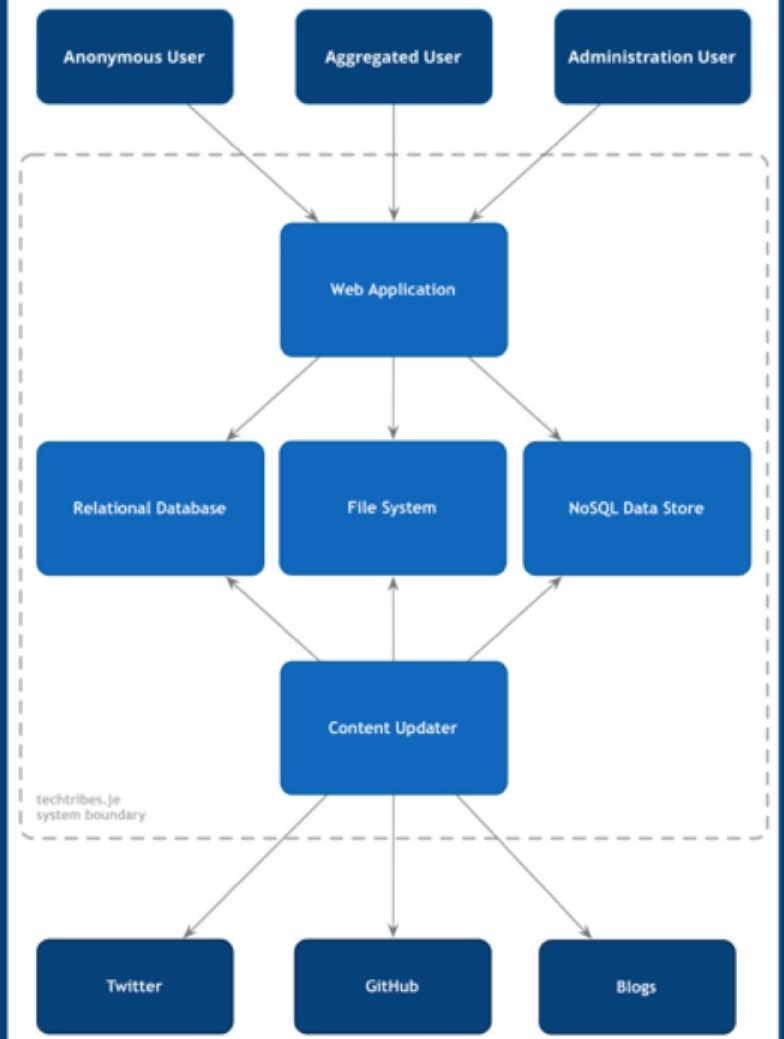
[Container: Java + Spring MVC]

Allows users to view people, tribes, content, events, jobs, etc from the local tech, digital and IT sector.

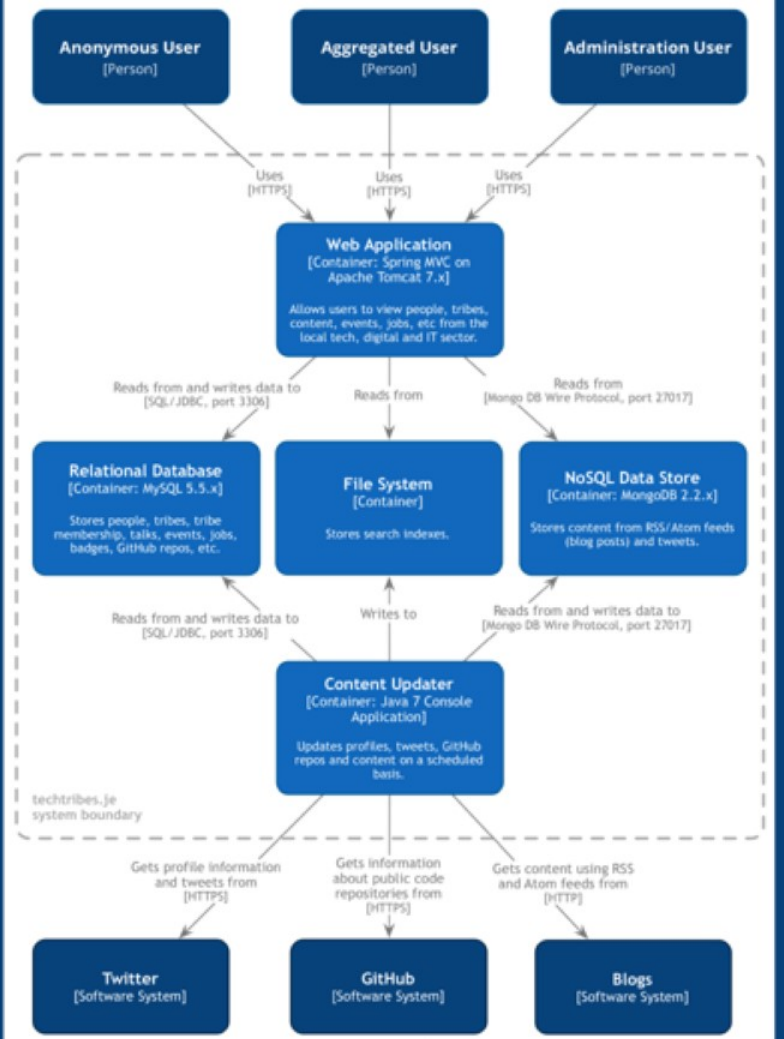
Twitter Connector

[Component: Spring Bean + Twitter4j]

Retrieves profile information and tweets (using the REST and Streaming APIs).



[Containers] techtribes.je



[Containers] techtribes.je

Lines

Favour uni-directional lines showing the most important dependencies or data flow, with an annotation to be explicit about the purpose of the line and direction

Dependency vs data flow

Key/legend

Explain shapes, line styles, colours, borders, acronyms, etc

... even if your notation seems obvious!

Notation, notation, notation

A software architecture diagram review checklist

General

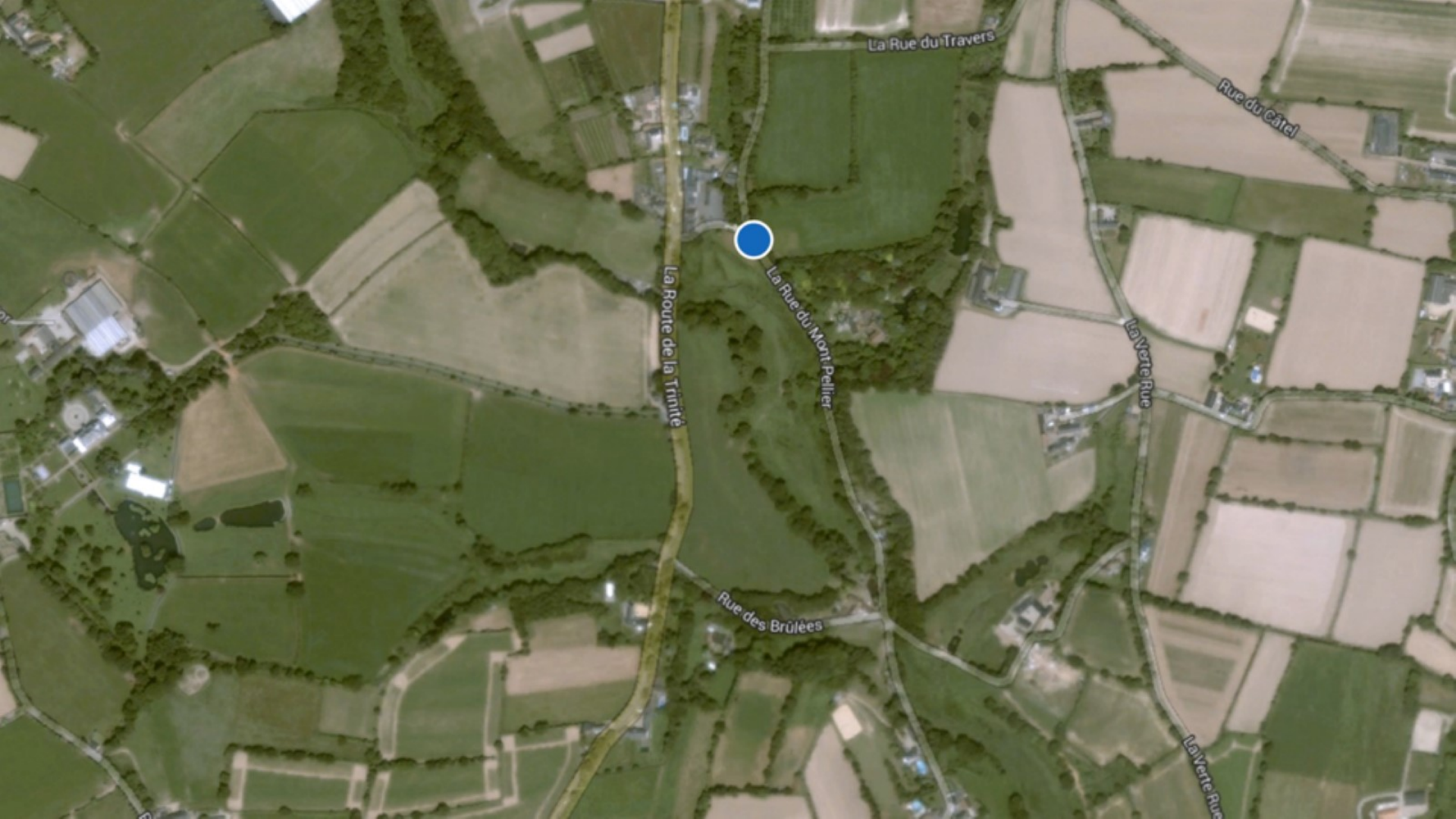
Does the diagram have a title?	Yes	No
Do you understand what the diagram type is?	Yes	No
Do you understand what the diagram scope is?	Yes	No
Does the diagram have a key/legend?	Yes	No

Elements

Does every element have a name?	Yes	No
Do you understand the type of every element? (i.e. the level of abstraction; e.g. software system, container, etc)	Yes	No
Do you understand what every element does?	Yes	No
Where applicable, do you understand the technology choices associated with every element?	Yes	No
Do you understand the meaning of all acronyms and abbreviations used?	Yes	No
Do you understand the meaning of all colours used?	Yes	No

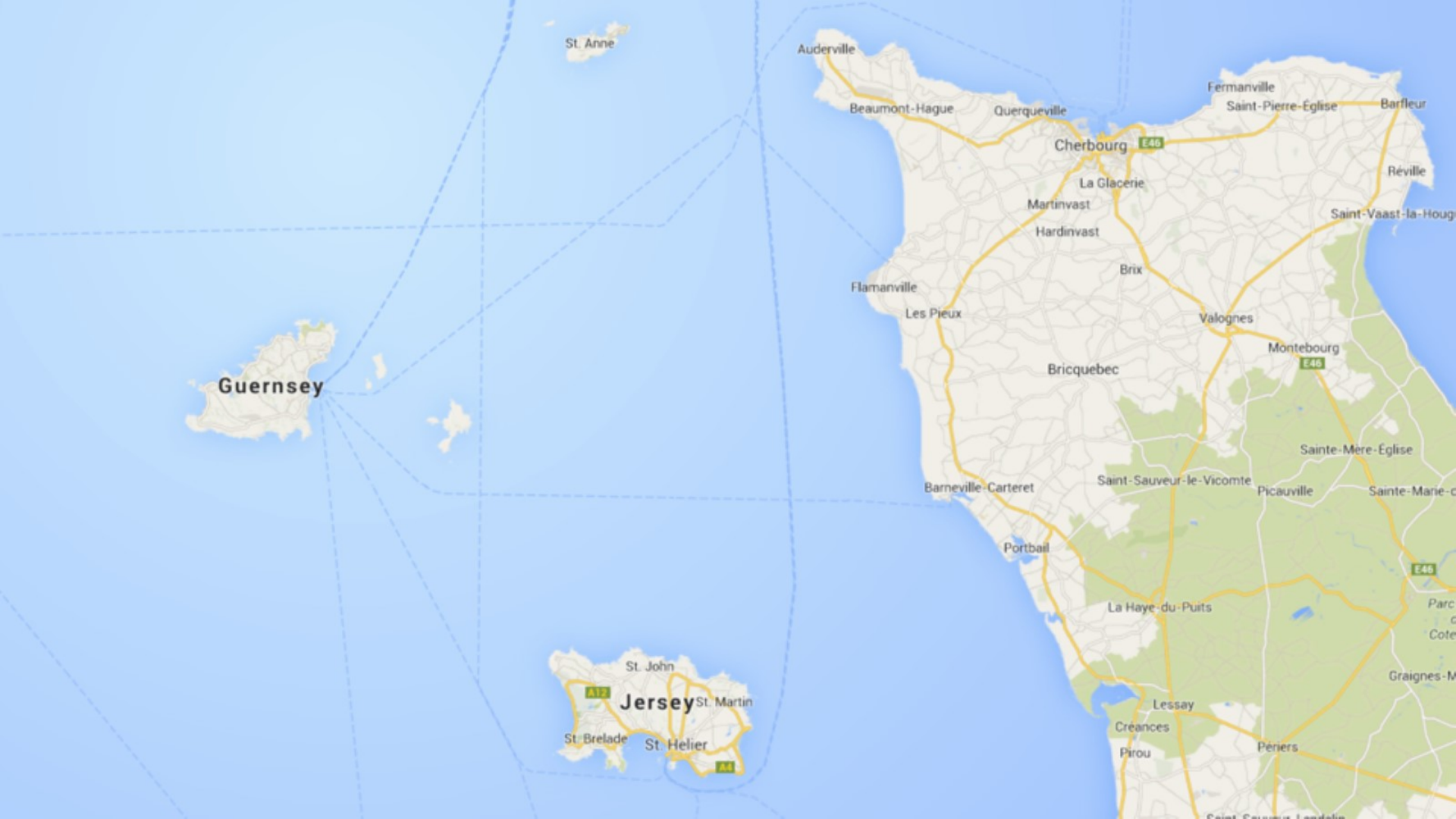
Documenting software architecture


La Rue du


La Rue du Travers
Rue du Câtel
La Route de la Trinité
La Rue du Mont Pellier
La Verte Rue
Rue des Brûlées


St. John
St. Mary
St. Ouen
Trinity
St. Martin
St. Peter
Jersey
Les Mielles
Nature Reserve
Mont Orgueil Castle
St. Brelade
St. Helier
Grouville
St. Helier, Je - Saint Peter Port, Gg
Granville, FR - St. Helier, JE
St. Malo, FR - Portsmouth, UK
St. Peter Port, Gg - Saint Helier, Je
Poole - Jersey (St. Helier)
Jersey (St. Helier) - Poole


St. John
St. Mary
St. Ouen
Rte de l'Etacq
Grande Route des Mielles
Les Mielles
Nature Reserve
St. Peter
Jersey
Trinity
La Grande Route de Saint Jean
La Route de la Trinité
Verte Rue
St. Martin
Mont Orgueil Castle
St. Brelade
A13
A13
A1
A2
St. Helier
Bagatelle Rd
Grouville
A4
St. Malo, FR - Portsmouth, UK
St. Peter Port, Gg - Saint Helier, Je
St. Helier, Je - Saint Peter Port, Gg
Poole - Jersey (St. Helier)
Jersey (St. Helier) - Poole
Poole - Jersey (St. Helier)
Granville, FR - St. Helier, JE


St. Anne
Auderville
Beaumont-Hague
Querqueville
Fermanville
Saint-Pierre-Église
Barfleur
Cherbourg
E46
Réville
La Glacerie
Martinvast
Hardinvast
Saint-Vaast-la-Hougue
Brix
Flamanville
Les Pieux
Valognes
Montebourg
E46
Bricquebec
Guernsey
Sainte-Mère-Église
Saint-Sauveur-le-Vicomte
Barneville-Carteret
Picauville
Portbail
E46
La Haye-du-Puits
St. John
A12
Jersey
St. Martin
St. Brelade
St. Helier
A4
Lessay
Créances
Pirou
Périers

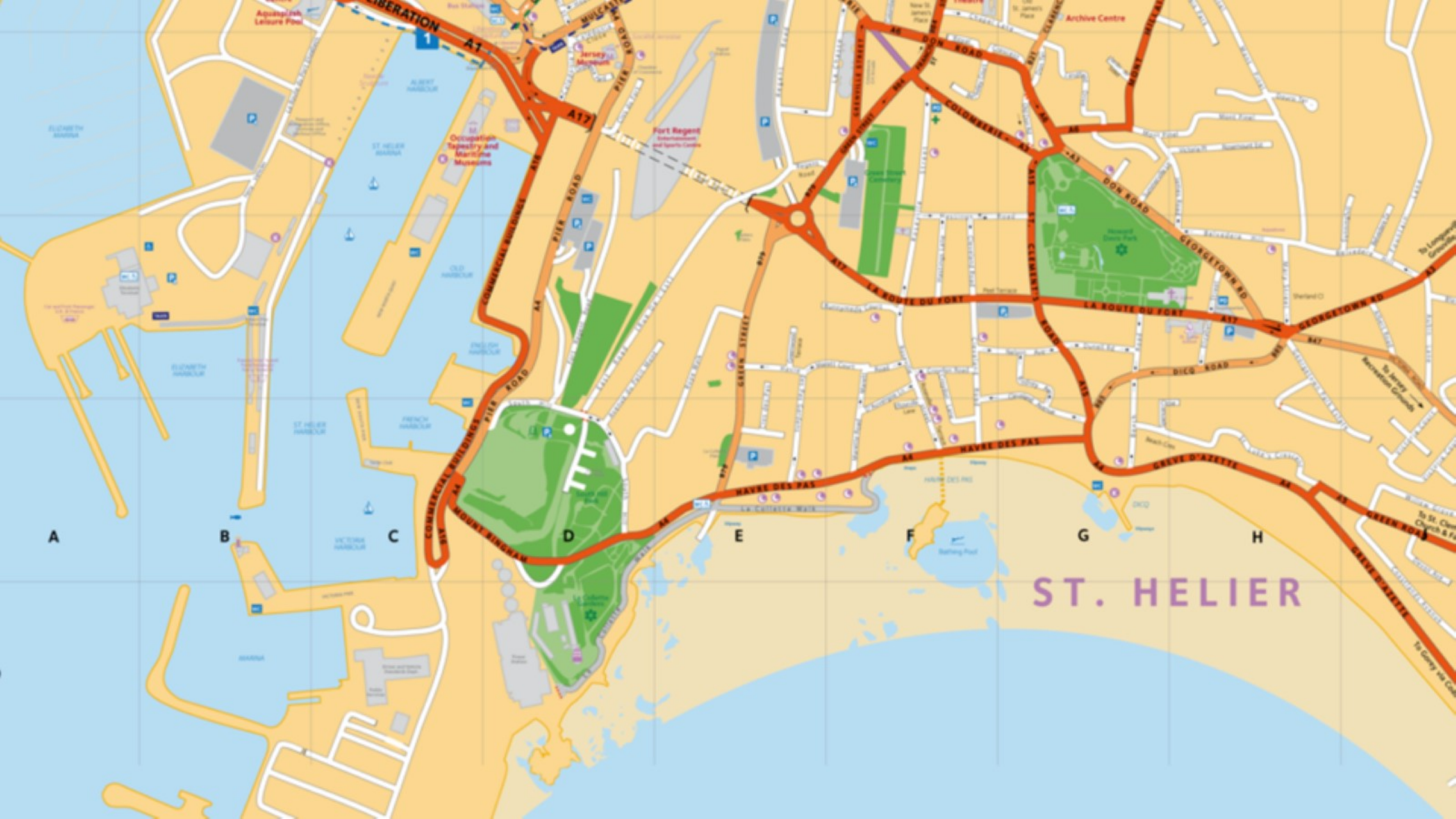

Ireland
Dublin
Belfast
Cork
Limerick
Galway
Waterford
Kilkenny
Wexford
Bray
Naas
Portlaoise
Wicklow
Ennis
Carlow
Enniscorthy
Gorey
Newcastle West
Clonmel
Fermoy
Dungarvan
Mallow
Midleton
Bandon
Kinsale
Tullamore
Bangor
Colwyn Bay
Caernarfon
Wales
Cardiff
Swansea
Newport
Pontypridd
Barry
Haverfordwest
Carmarthen
Llanelli
Neath
Milford Haven
Aberystwyth
Newtown
Shrewsbury
Hereford
Worcester
Kidderminster
Cheltenham
Gloucester
Bristol
Bath
Swindon
Oxford
Reading
London
Basildon
Southend-on-Sea
Croydon
Maidstone
Margate
Deal
Dover
Ashford
Crawley
Brighton
Hastings
Eastbourne
Southampton
Portsmouth
Isle of Wight
Newport
Yeovil
Taunton
Barnstaple
Bideford
Exeter
Tavistock
Torquay
Plymouth
Newquay
Truro
St Austell
St Ives
Penzance
Falmouth
Dartmouth
Weymouth
Swanage
Shanklin
Stoke-on-Trent
Derby
Leicester
Peterborough
Boston
Lincoln
Cromer
King's Lynn
North Walsham
Dereham
Norwich
Lowestoft
Beccles
Attleborough
Corby
Cambridge
Bedford
Luton
St Albans
Chelmsford
Colchester
Felixstowe
Clacton-on-Sea
Ipswich
England
Scotland
Northern Ireland
English Channel
Celtic Sea
Guernsey
Jersey
Cherbourg
Bricquebec
Le Havre
Deauville
Caen
Saint-Lô
Lisieux
Coutances
Granville
Vire
Flers
Saint-Malo
Dreux
Rouen
Elbeuf
Louviers
Évreux
Beauvais
Amiens
Dieppe
Fécamp
Abbeville
Saint-Quentin
Cergy
Paris
Boulogne-Billancourt
Laon
Soissons
Reims
Épernay
Châlons-en
Lannion
Saint-Pol-de-Léon
Morlaix
Dunkirk
Calais
Boulogne-sur-Mer
Ostend
Bruges
Ghent
Kortrijk
Lille
Tournai
Béthune
Arras
Douai
Cambrai
Mons
Vlissingen
M53
M1
M54
M6 Toll
M42
M40
M4
M3
M25
M11
M20
E20
E201
E30
E40
E42
E15
E17
E402
E44
E46
E19
E401
E3
E50
E312


United Kingdom
Northern Ireland
Ireland
Isle of Man
England
Wales
Scotland
Denmark
Germany
Netherlands
Belgium
Luxembourg
Austria
Switzerland
Glasgow
Perth
Kirkcaldy
Londonderry / Derry
Bangor
Sligo
Dundalk
Castlebar
Drogheda
Dublin
Galway
Naas
Bray
Limerick
Carlow
Kilkenny
Tralee
Clonmel
Waterford
Cork
Middlesbrough
Blackpool
Leeds
Hull
Liverpool
Sheffield
Nottingham
Stoke-on-Trent
Derby
Norwich
Great Yarmouth
Ipswich
Swansea
Bristol
Cardiff
Headington
London
Leigh-on-Sea
Reading
Barnstaple
Exeter
Southampton
Portsmouth
Bournemouth
Plymouth
Newquay
St. Ives
Truro
English Channel
Celtic Sea
Guernsey
Jersey
Amsterdam
The Hague
Den Helder
Groningen
Bremerhaven
Aurich
Hamburg
Bremen
Hanover
Berlin
Bielefeld
Brunswick
Magdeburg
Dessau
Münster
Kernstadt
Woensel-Noord
Bruges
Antwerp
Brussels
Lille
Dusseldorf
Nord
Cologne
Leipzig
Dresden
Chemnitz
Gera
Frankfurt
Wiesbaden
Würzburg
Mannheim
Nuremberg
Saarbrücken
Karlsruhe
Stuttgart
Metz
Strasbourg
Nancy
Ulm
Munich
Freiburg
Konstanz
Basel
Zurich
Innsbruck
Salzburg
Linz
Wels
Prague
Plzen
Pardubice
Ceske Budejovice
Regensburg
Ingolstadt
Landshut
Amiens
Saint-Quentin
Le Havre
Caen
Rouen
Reims
Paris
Évry
Troyes
Orleans
Le Mans
Rennes
Brest
Lannion
St-Malo
Saint-Brieuc
Quimper
Lorient
Vannes
Angers
Tours
Nantes
Saint-Nazaire
Dijon
Besançon
Aarhus
Viborg
Silkeborg
Horsens
Esbjerg
Odense
Copenhagen
Malmö
Helsingborg
Kristianstad
Karlskrona
Flensburg
Kiel
Heide
Rostock
Stralsund
Swinoujscie
Anklam
Gryfice
Schwerin
Neubrandenburg
Bergedorf
Szczecin
Stargard Szczecinski
Gorzów Wielkopolski
Zielona Gora
Cottbus
Görlitz
Liberec
Ústí nad Labem-město
Oslo
Gothenburg
Stockholm
Helsinki
Tallinn
Riga
Vilnius
Minsk
Warsaw
Kraków
Katowice
Ostrava
Brno
Bratislava
Vienna
Budapest
Győr
Graz
Maribor
Ljubljana
Zagreb
Rijeka
Split
Sarajevo
Belgrade
Novi Sad
Subotica
Osijek
Timişoara
Arad
Cluj-Napoca
Oradea
Debrecen
Miskolc
Košice
Lviv
Kyiv
Chernihiv
Sumy
Kharkiv
Dnipro
Zaporizhzhia
Mariupol
Donetsk
Luhansk
Simferopol
Sevastopol
Yalta
Kerch
Crimea
Rostov-on-Don
Volgograd
Saratov
Samara
Ufa
Chelyabinsk
Yekaterinburg
Perm
Kazan
Nizhny Novgorod
Moscow
Tula
Ryazan
Voronezh
Belgorod
Kursk
Orel
Bryansk
Smolensk
Tver
Yaroslavl
Vologda
Arkhangelsk
Murmansk
Petrozavodsk
Saint Petersburg
Pskov
Novgorod
Kaliningrad
Gdańsk
Bydgoszcz
Poznań
Łódź
Wrocław
Opole
Gliwice
Rybnik
Ostrawa
Olomouc
Pardubice
Hradec Králové
Plzeň
Karlovy Vary
Ústí nad Labem
Liberec
Jablonec nad Nisou
Trutnov
Jelenia Góra
Legnica
Lubin
Zielona Góra
Gorzów Wielkopolski
Stargard Szczecinski
Szczecin
Gryfice
Swinoujscie
Anklam
Neubrandenburg
Bergedorf
Schwerin
Rostock
Stralsund
Heide
Kiel
Flensburg
Esbjerg
Odense
Copenhagen
Malmö
Helsingborg
Kristianstad
Karlskrona
Horsens
Silkeborg
Viborg
Aarhus
Kirkcaldy
Perth
Glasgow
Kilmarnock
Ayr
Middlesbrough
Blackpool
Leeds
Hull
Liverpool
Sheffield
Nottingham
Stoke-on-Trent
Derby
Norwich
Great Yarmouth
Ipswich
Swansea
Bristol
Cardiff
Headington
London
Leigh-on-Sea
Reading
Barnstaple
Exeter
Southampton
Portsmouth
Bournemouth
Plymouth
Newquay
St. Ives
Truro
English Channel
Celtic Sea
Guernsey
Jersey
Amsterdam
The Hague
Den Helder
Groningen
Bremerhaven
Aurich
Hamburg
Bremen
Hanover
Berlin
Bielefeld
Brunswick
Magdeburg
Dessau
Münster
Kernstadt
Woensel-Noord
Bruges
Antwerp
Brussels
Lille
Dusseldorf
Nord
Cologne
Leipzig
Dresden
Chemnitz
Gera
Frankfurt
Wiesbaden
Würzburg
Mannheim
Nuremberg
Saarbrücken
Karlsruhe
Stuttgart
Metz
Strasbourg
Nancy
Ulm
Munich
Freiburg
Konstanz
Basel
Zurich
Innsbruck
Salzburg
Linz
Wels
Prague
Plzen
Pardubice
Ceske Budejovice
Regensburg
Ingolstadt
Landshut
Amiens
Saint-Quentin
Le Havre
Caen
Rouen
Reims
Paris
Évry
Troyes
Orleans
Le Mans
Rennes
Brest
Lannion
St-Malo
Saint-Brieuc
Quimper
Lorient
Vannes
Angers
Tours
Nantes
Saint-Nazaire
Dijon
Besançon
North Sea

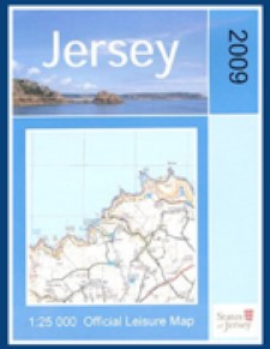

Iceland
Sweden
Finland
Norway
Russia
United Kingdom
Ireland
Denmark
Belarus
Poland
Germany
Ukraine
Austria
France
Romania
Kazakhstan
Mongolia
Italy
Spain
Portugal
Greece
Turkey
Uzbekistan
Kyrgyzstan
Turkmenistan
Sea of Okhotsk
Sea of Japan
China
South Korea
Japan
Syria
Iraq
Iran
Afghanistan
Pakistan
Tunisia
Morocco
Algeria
Libya
Egypt
Nepal
India
Western Sahara
Saudi Arabia
Oman
Myanmar (Burma)
Philippine Sea
East China Sea
Mauritania
Mali
Niger
Chad
Sudan
Yemen
Thailand
Bay of Bengal
Arabian Sea
Gulf of Aden
Vietnam
Philippines
Burkina Faso
Guinea
Nigeria
Ethiopia
Gulf of Thailand
Ghana
South Sudan
Laccadive Sea
Malaysia
Gulf of Guinea
Somalia
Kenya
Gabon
DR Congo
Indonesia
Tanzania
Banda Sea
Papua New Guinea
Arafura Sea

Enough detail to start **exploring**


01
02
03
Grosnez
La Grève au Lançon (Plémont)
Plémont
Creux Gros
Creux Gobourel
Douet de la Mer
Grosnez Castle (ruin)
Race Course
Les Landes
Portinfer
Rue de la Pointe
Ville la Bas
Les Pallières
Le Pinacle
Model Aircraft Field
Le Pulec
Route de Vinchelez
La Route de Millais
Bleu Soleil Campsite
Léoville
La Villaise
Sea Wall
L'Étacq
Treasures of the Earth
Battle of Flowers Museum
Jersey Pearl
Channel Islands Military Museum
Les Frès
La Route de Trodez
Le Carré
Dolmen du Mont Grantez
La Crabière
Ville au Bas
La Mielle de Morville
Barge Aground
La Grande Route des Mielles
St. Ouen's Pond
Les Mielles
Le Mont Rossignol
Fol de la Mare
La Ville de L'Église
Mont du Jubile
ST. OUEN'S
ST. OUEN
Grève de Lecq
GREVE DE LECQ
Creux de Lasse
Les Reuses
Grève de Lecq Barracks
Rifle Range
Crabbé
Devil's Hole
Sorel
La Houle
Ronez
Le Fosse Vicq
Go Kart Racing
Moto Cross
Mourier Bay
ST. JOHN'S BAY
Wolf's Coves
Fremont
BONNE NUIT
Les Landes
ST. JOHN
La Mare Wine Estate
ST. MARY
Mont Mado
Le Claire Stables
Handois Reservoir
BSJA Showground
Judith Quérée Garden
Bouchet Agateware Pottery
Catherine Best Jewellery
Jersey Goldsmiths & Lion Park
Bon Air
Carrefour
The Elms
aMaizin! Maze Adventure Park & Craft Centre
Living Legend
de Quétivel
Ville Emphrie
C99
C100
C101
C103
C104
C105
C106
C112
C114
C115
C117
C122
B31
B32
B33
B34
B35
B40
B50
B52
B53
B55
B56
B63
B64
B65
B67
B69
B82
B83
B84
B85
A10
A12
A9
1
4
6
6a



Aquasplash Leisure Pool
LIBERATION
A1
A17
Jersey Museum
PIER ROAD
Fort Regent
Entertainment and Sports Centre
Occupation Tapestry and Maritime Museums
Archive Centre
ELIZABETH MARINA
ALBERT HARBOUR
ST. HELIER MARINA
OLD HARBOUR
ENGLISH HARBOUR
ELIZABETH HARBOUR
ST. HELIER HARBOUR
FRENCH HARBOUR
VICTORIA HARBOUR
MARINA
COMMERCIAL BUILDINGS
MOUNT BINGHAM
A4
DON ROAD
A6
COLOMBERIE
A3
GREEN STREET
LA ROUTE DU FORT
ST. CLEMENTS ROAD
GEORGETOWN RD
Howard Davis Park
Green Street Cemetery
Peel Terrace
DICQ ROAD
B47
To Jersey Recreation Grounds
To Longueville
To St. Clement Church
HAVRE DES PAS
La Collette Walk
La Collette Gardens
Bathing Pool
DICQ
GREVE D'AZETTE
GREEN ROAD
To Gorey via
A
B
C
D
E
F
G
H
ST. HELIER

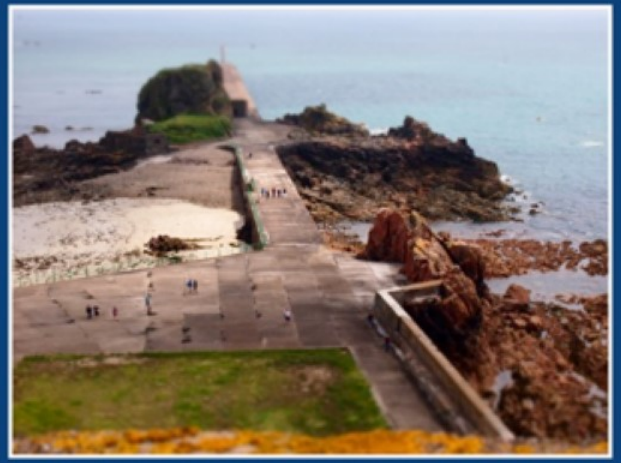


Very detailed and precise

(terrain, buildings, etc)

Points of interest

Elizabeth Castle

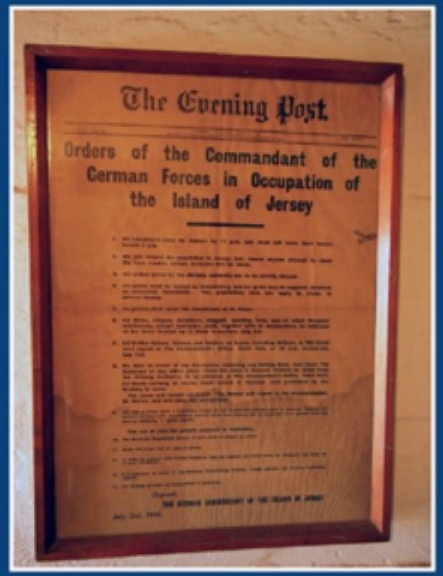




Granite and concrete?!


Magazine
1594
The powder and munitions store for the original castle. The entrance doorway was built c1680. In 1617 the Royal Commissioners' inventory of the castle recorded 41 barrels of powder stored here as well as quantities of saltpetre
for making
Dépôt des munitions.
En 1617 l'inventaire
du château fait mention de
41 barils de poudre


OCCUPATION
During the refortification of the castle many older features were remodelled or demolished. The most distinctive change was the addition of the large tower on top of the Mount.


The Evening Post
Orders of the Commandant of the German Forces in Occupation of the Island of Jersey

History


OCCUPATION

Travel Guidebook

(maps, points of interest, sights, itineraries, history, culture, practical information, etc)

Working software

over

comprehensive documentation

Manifesto for Agile Software Development

The code doesn’t tell the whole story

Software Architecture Document

Useful information spread across hundreds of pages; rarely read or updated

Software Guidebook

(maps, points of interest, sights, itineraries, history, culture, practical information, etc)

The scope is a single **software system**

Describe what you

can’t get from the code

Documentation should be **constantly evolving**

Context

A system context diagram, plus some narrative text to "set the scene".

Functional Overview

An overview of the software system; perhaps including wireframes, UI mockups, screenshots, workflow diagrams, business process diagrams, etc.

Quality Attributes

A list of the quality attributes (non-functional requirements; e.g. performance, scalability, security, etc).

Constraints

A list of the environmental constraints (e.g. timescales, budget, technology, team size/skills, etc).

Principles

A list of the development and architecture principles (e.g. coding conventions, separation of concerns, patterns, etc).

Software Architecture

A description of the software architecture, including static structure (e.g. containers and components) and dynamic/ runtime behaviour.

Code

A description of important or complicated component implementation details, patterns, frameworks, etc.

Data

Data models, entity relationship diagrams, security, data volumes, archiving strategies, backup strategies, etc.

This is a **starting point**; add and remove sections as necessary.

Infrastructure Architecture

A description of the infrastructure available to run the software system.

Deployment

The mapping of software (e.g. containers) to infrastructure.

Development Environment

A description of how a new developer gets started.

Operation and Support

An overview of how the software system is operated, supported, monitored, etc.

Decision Log

A log of the major decisions made; e.g. as free format text or a collection of "Architecture Decision Records".

Questions and break

Software architecture in the delivery process

Structured Programming

(Software Craftsmanship v1.0)

Historically there's been
a tendency towards
big design up front

“Waterfall”

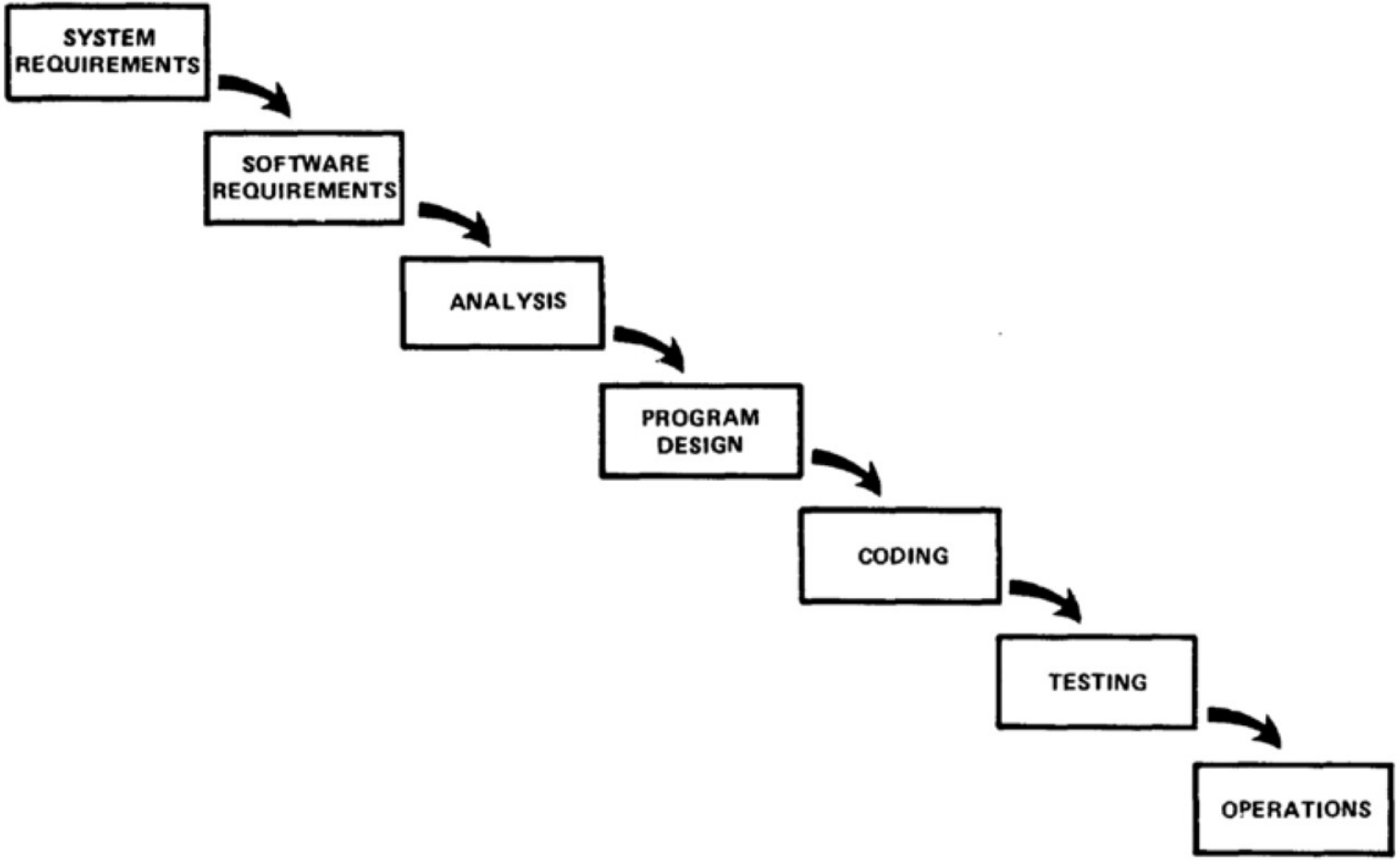


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

time spent early in the software production cycle can reduce costs at later stages

Wikipedia

A “structured approach”
with an emphasis on
comprehensive documentation

SSADM

Structured Systems Analysis and Design Method

> a systems approach to the analysis and design of information systems

Wikipedia

I believe in this concept, but the implementation described above is risky and invites failure.

Managing the development of large software systems

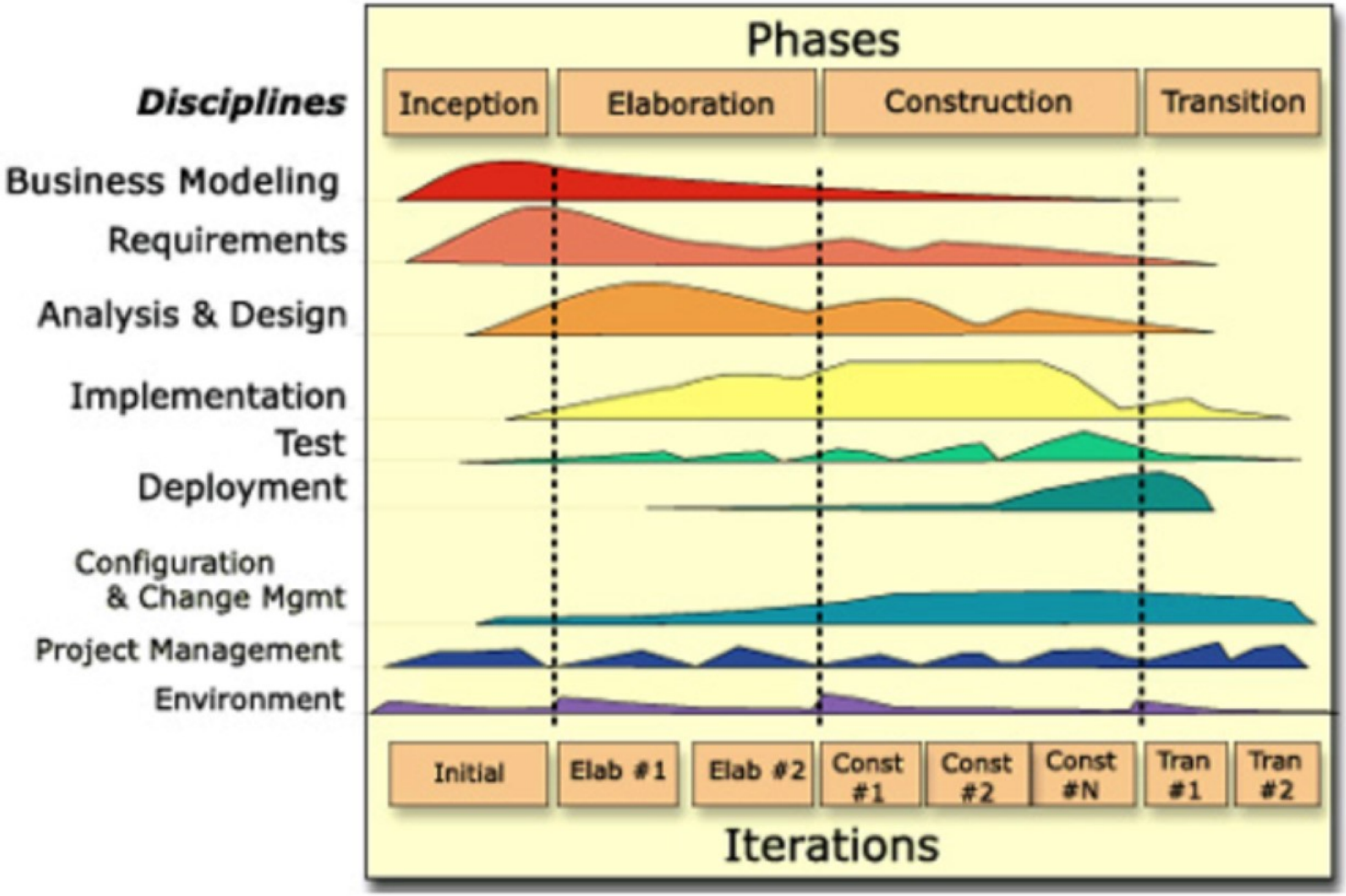
Dr Winston W. Royce

Long feedback loops
often resulted in the
wrong thing being delivered

Iterative and incremental development

RUP

Rational Unified Process


Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration
& Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const #1
Const #2
Const #N
Tran #1
Tran #2
Iterations

Roles and artifacts

Rapid Application Development

(“prototype-driven design”)

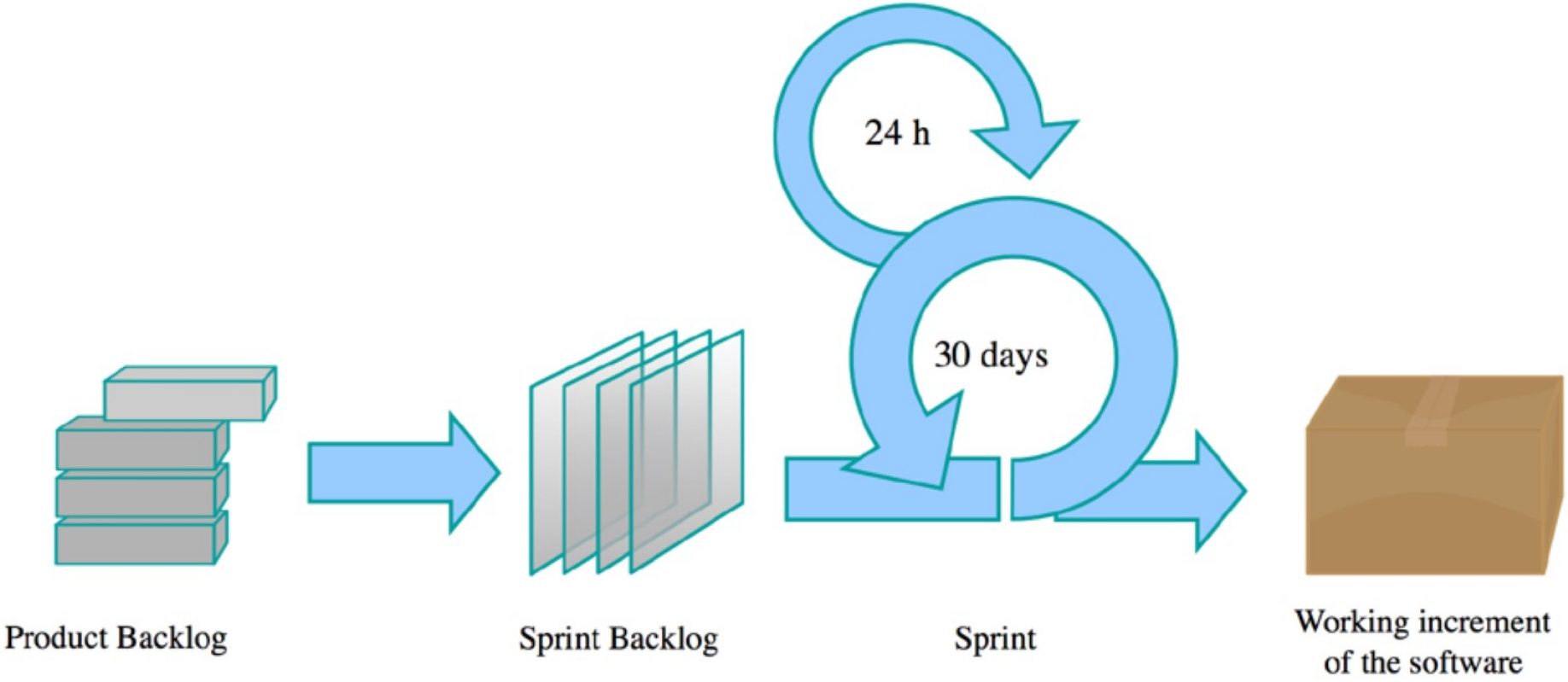
Agile

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.


24 h
30 days
Product Backlog
Sprint Backlog
Sprint
Working increment
of the software

We do Scrum

(as described in the book)

Is there a conflict between agile and architecture?

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

Principle 11 of the Manifesto for Agile Software Development

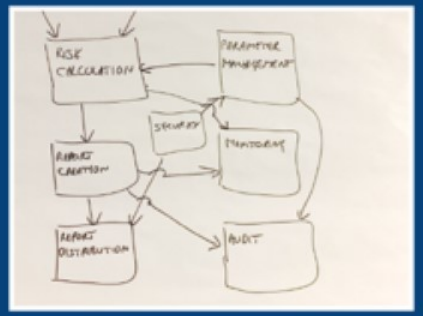
Architects? We don't need no stinkin’ architects!

Small, self-organising teams
of generalising specialists,
often where everybody
is jointly responsible

Responding to change
over
following a plan

Big design up front

vs



No design up front

Evolutionary architecture

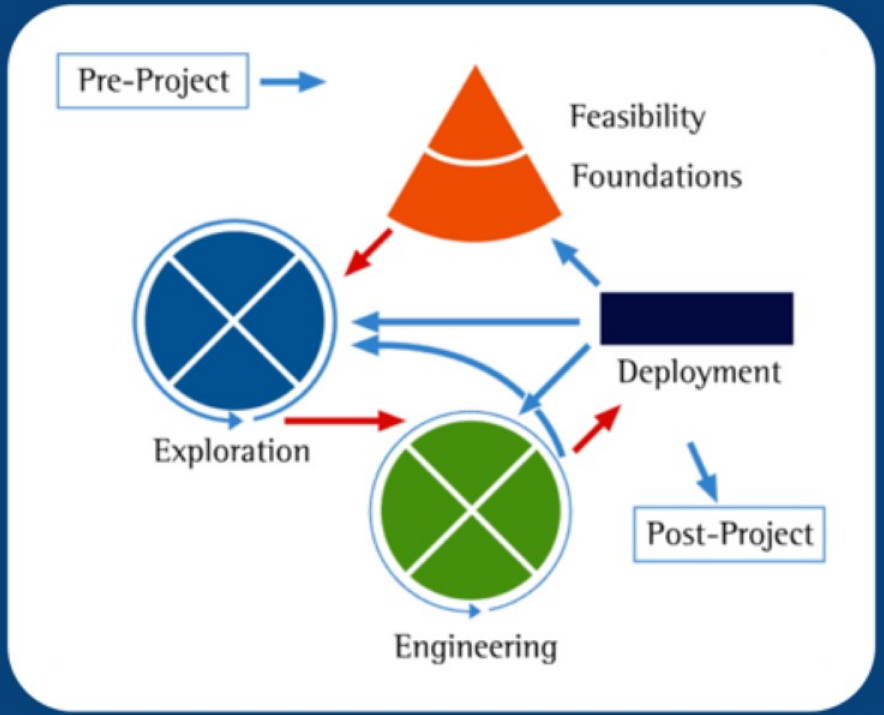
Are we allowed to do
up front design if we're agile?

Beyond Scrum

DSDM Atern

Disciplined Agile Delivery (DAD)

Scaled Agile Framework (SAFe)



“The Atern lifecycle process”

https://www.dsdm.org/content/lifecycle

The Foundations phase
is aimed at establishing
firm and enduring foundations
for the project.

It’s not about creating a perfect end-state or complete architecture

You need a starting point

Detailed blueprints

vs

setting a direction

Continuous technical leadership

Add value,
eliminate waste

How much **up front design** should you do?

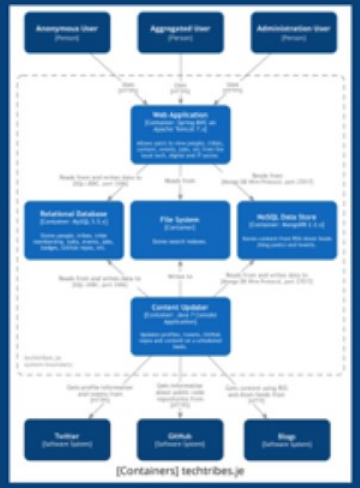
0%

100%

just enough

Not too much,
not too little!

[System Context] techtribes.je

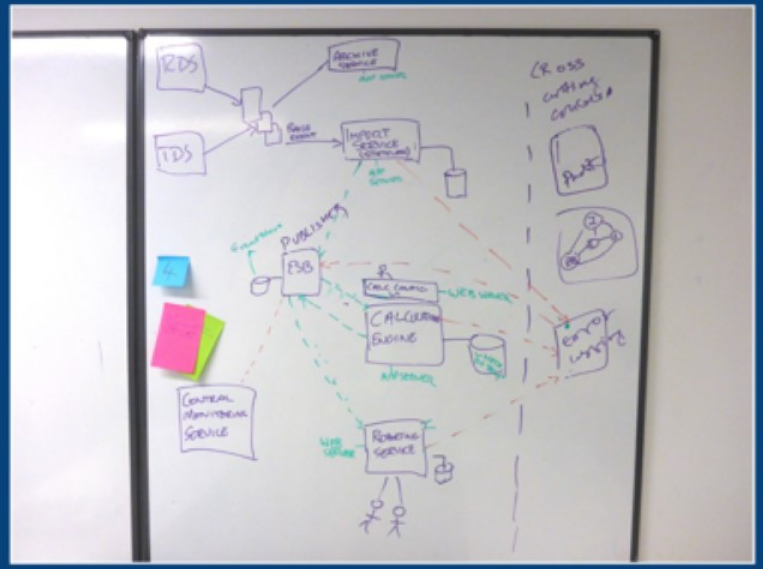


[Containers] techtribes.je

[Components] techtribes.je - Content Updater

Understand the structure and create a shared vision

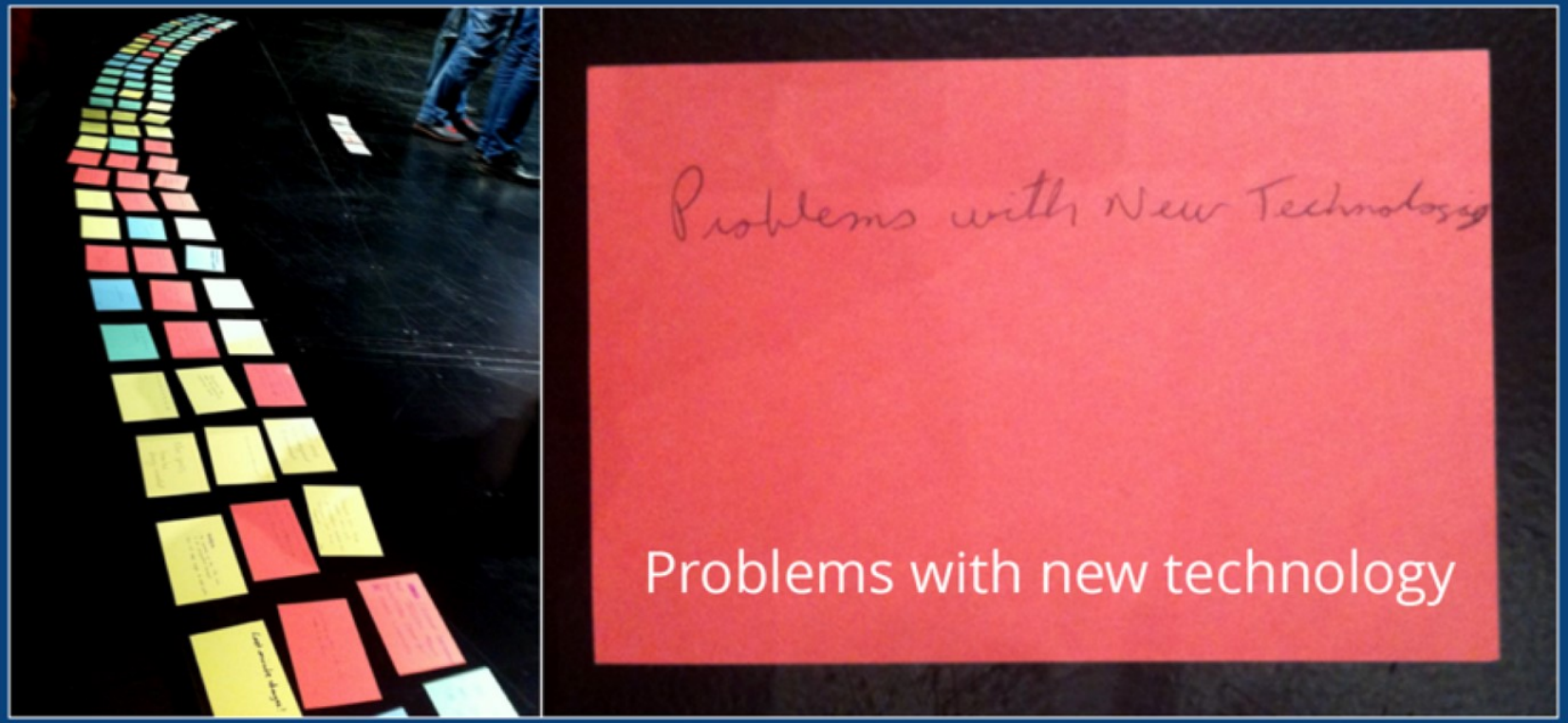
1. Is that what we're going to **build**?



2. Is it going to **work**?

Teams need to explicitly manage technical risk

An example timeline from “Beyond Retrospectives”
Linda Rising, GOTO Aarhus 2011



An example timeline from “Beyond Retrospectives”
Linda Rising, GOTO Aarhus 2011

Identify and **mitigate**
your **highest priority risks**

Probability

Impact

	Low 1	Medium 2	High 3
Low 1	1	2	3
Medium 2	2	4	6
High 3	3	6	9

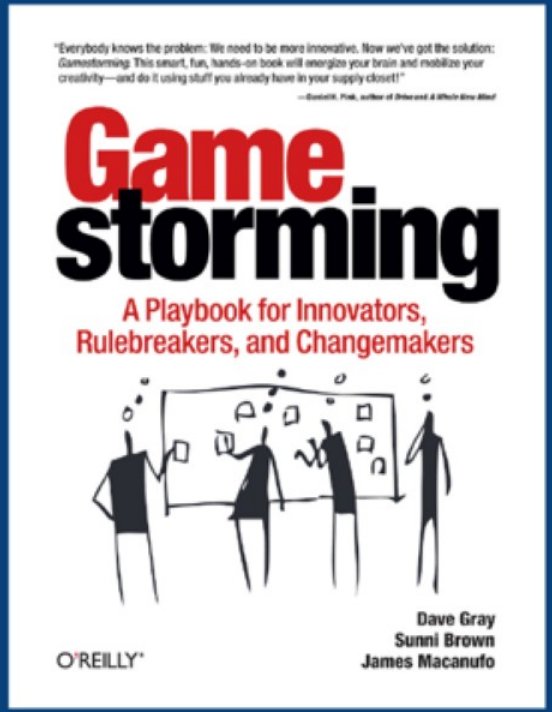
The software architecture role should own the technical risks

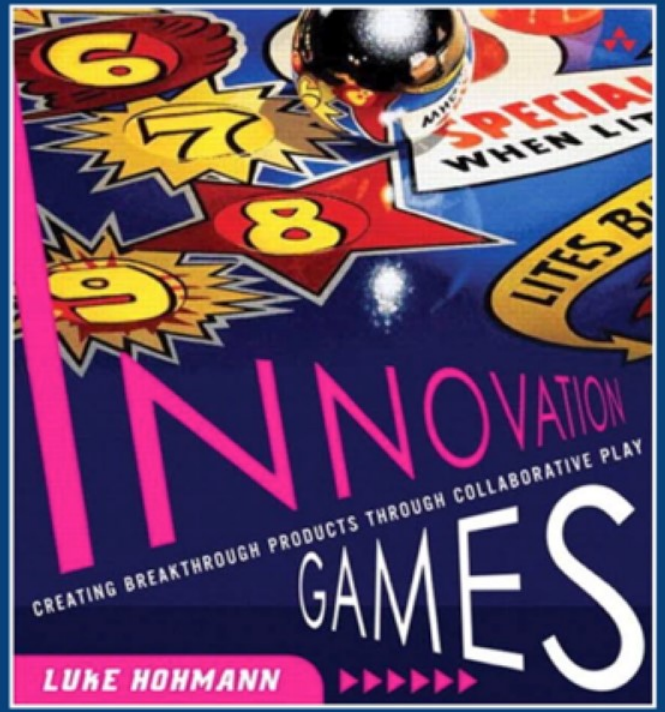
Architecturally significant?

Costly to change, complex or new

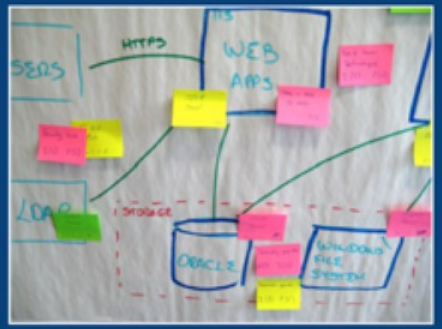
Like estimates,

risks are subjective





Visual and collaborative “games”




Risk-storming

A visual and collaborative technique for identifying risk

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

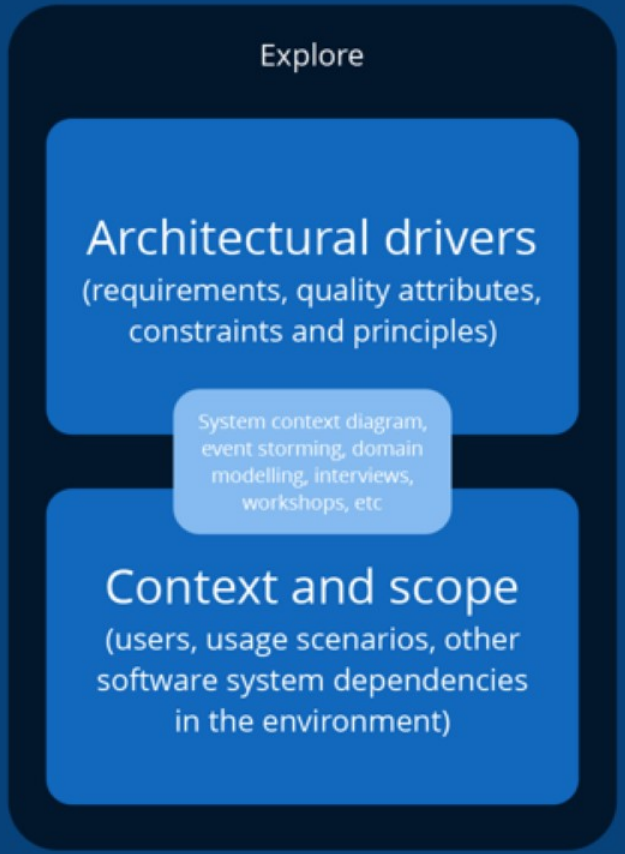
Agile Architecture: Strategies for Scaling Agile Development

Scott Ambler

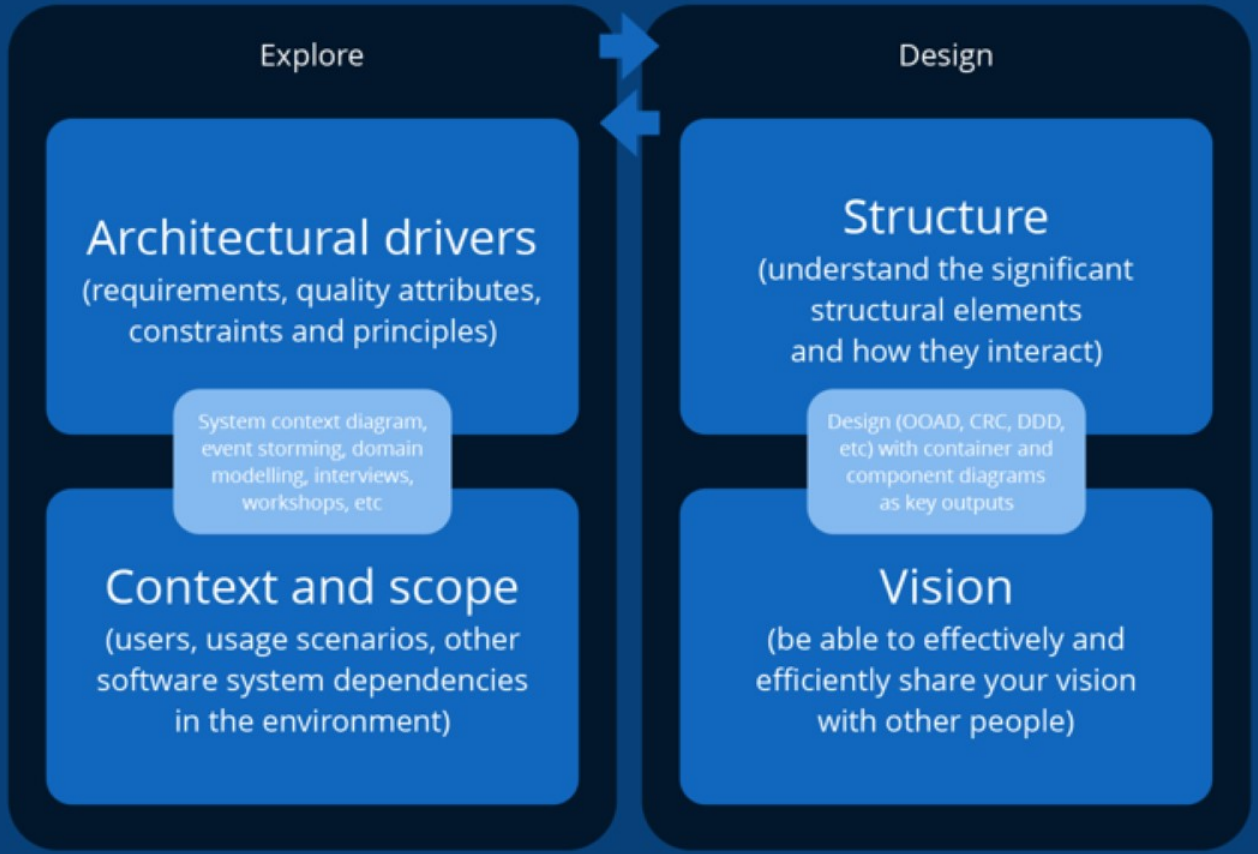
Concrete experiment

Proof of concept, prototype, spike, tracer, vertical slice, walking skeleton, executable reference architecture, ...

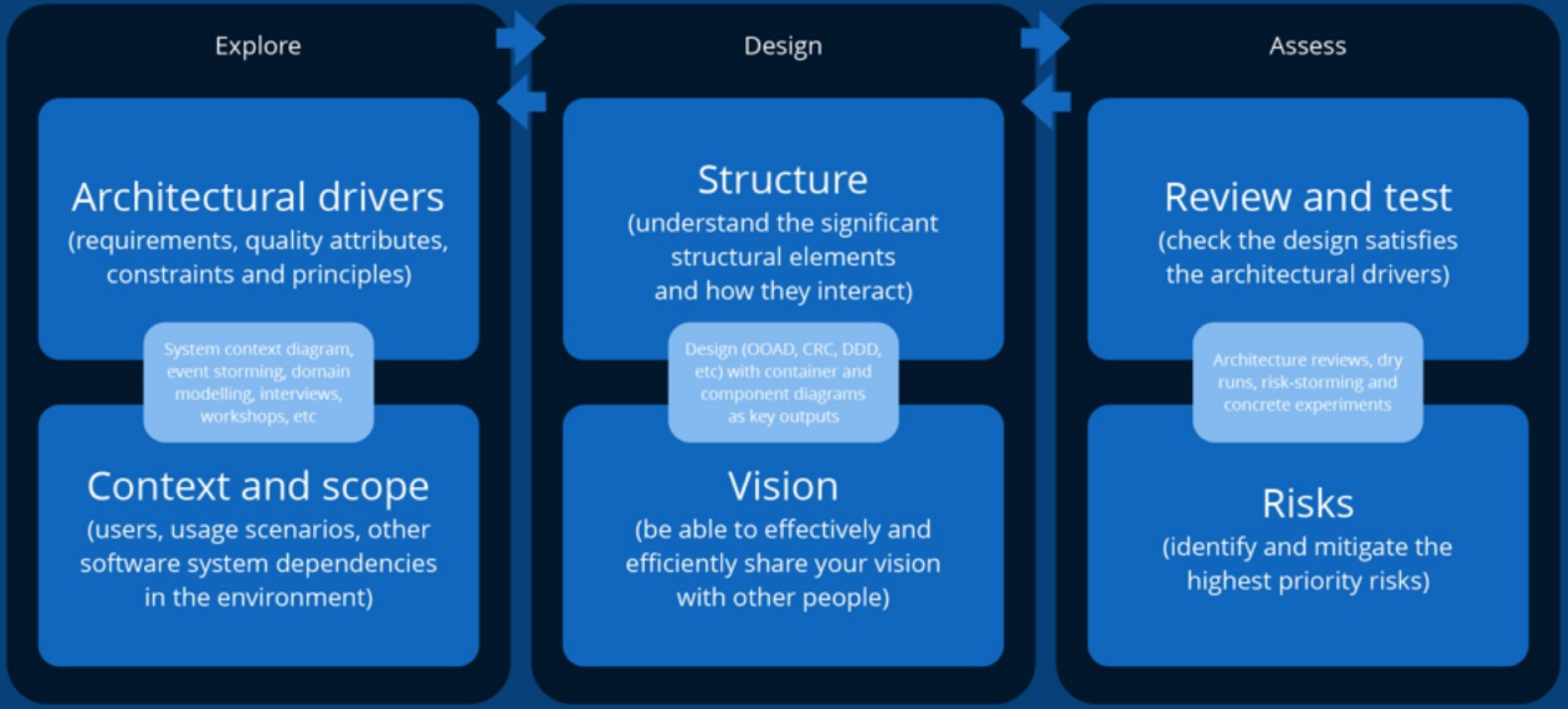
Just enough up front design to create
firm and sufficient foundations



The goal is a quick (hours or days rather than months or years), visual and collaborative approach to up front design to provide a starting point and set a direction for the team



The goal is a quick (hours or days rather than months or years), visual and collaborative approach to up front design to provide a starting point and set a direction for the team



The goal is a quick (hours or days rather than months or years), visual and collaborative approach to up front design to provide a starting point and set a direction for the team

Up front design is an iterative and incremental process; stop when:

You understand the context and scope of what you're designing.

You understand the significant design decisions (i.e. technology, modularity, etc).

You are confident that your design satisfies the architectural drivers.

You have identified and are comfortable with the risks associated with building the software.

You have a way to communicate your vision to other people.

How long?

Hours, days or weeks ... not months or years

Estimates?

Features vs components

we used to do stuff like this,
it worked but we stopped doing
it when we became agile

We all came away impressed with the practical, agile approach to software architecture. I'm going to push that we adopt most of these practices as a baseline...

Adopt an agile mindset

Choose a starting point and continuously improve to discover what works for you

Questions

Thank you!

simon.brown@codingthearchitecture.com
@simonbrown