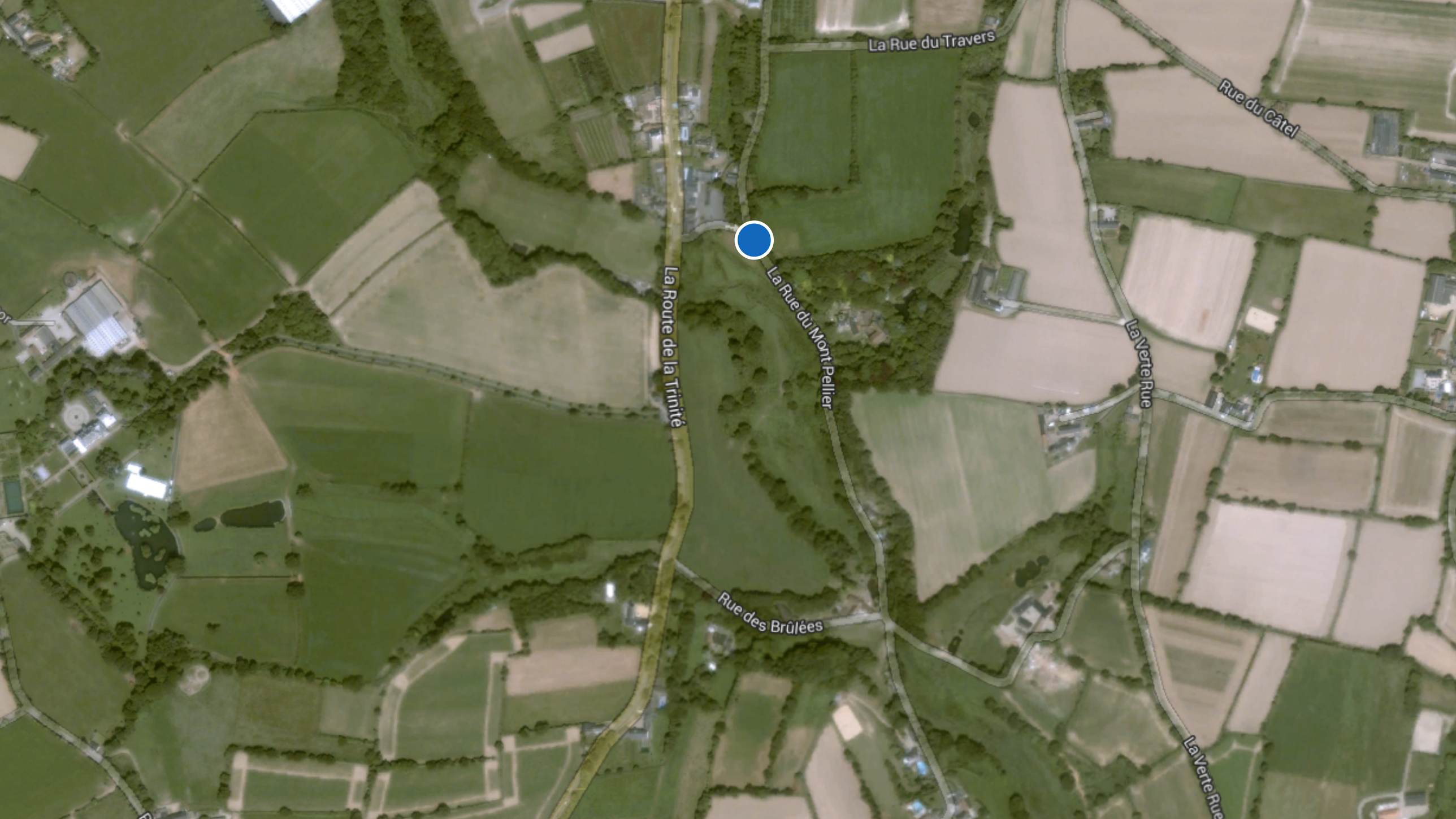
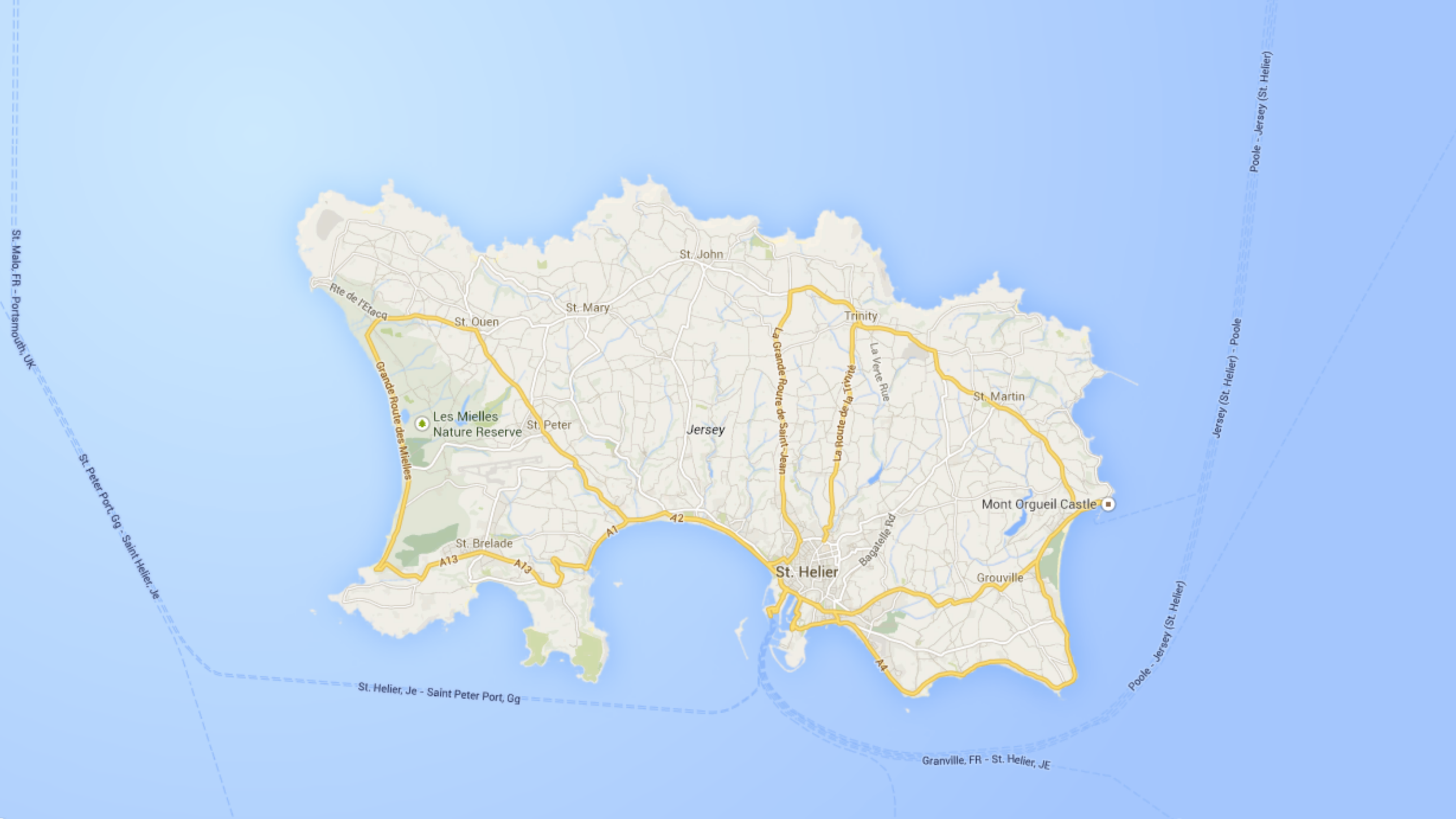
The art of visualising software architecture

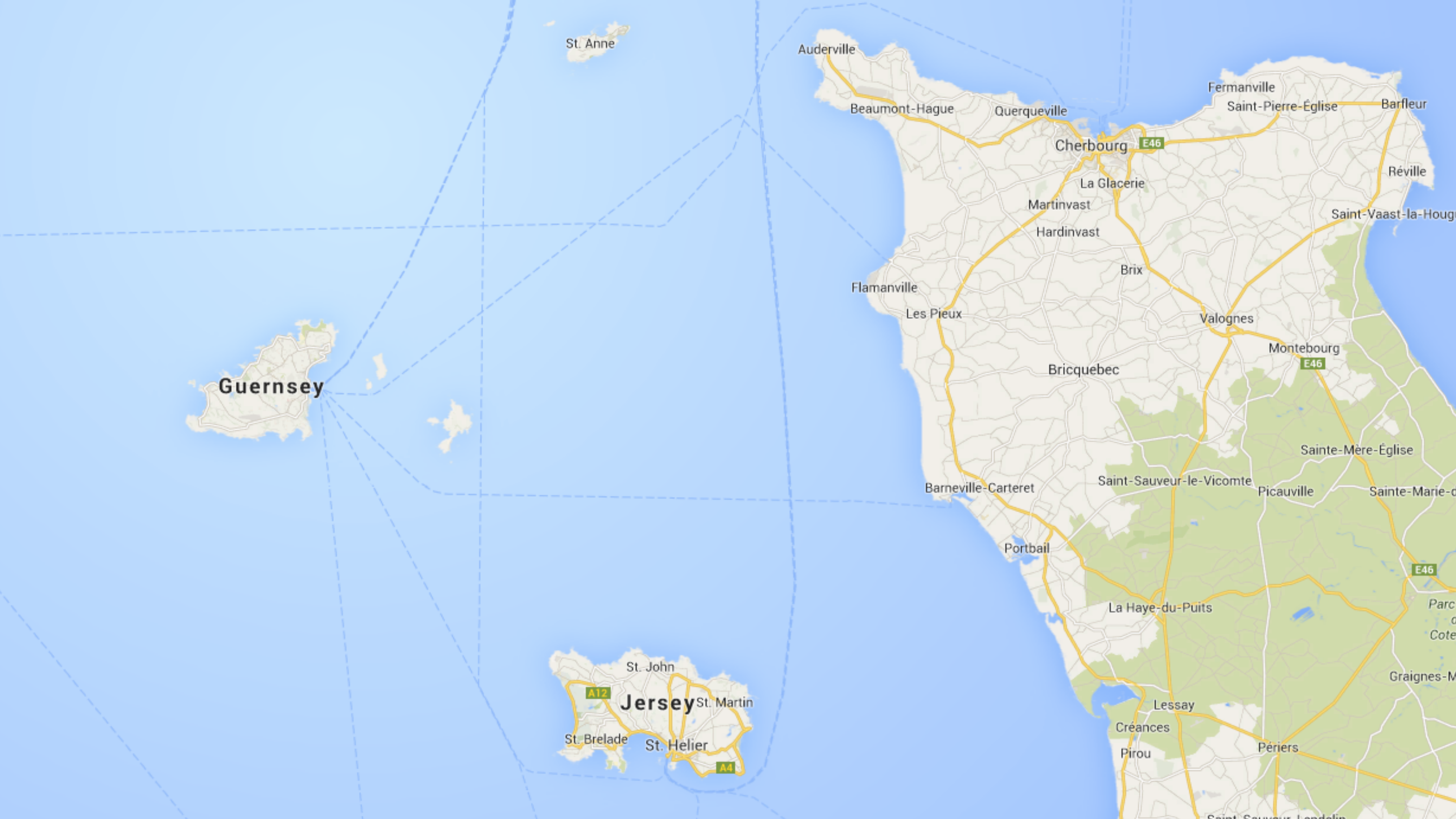
@simonbrown

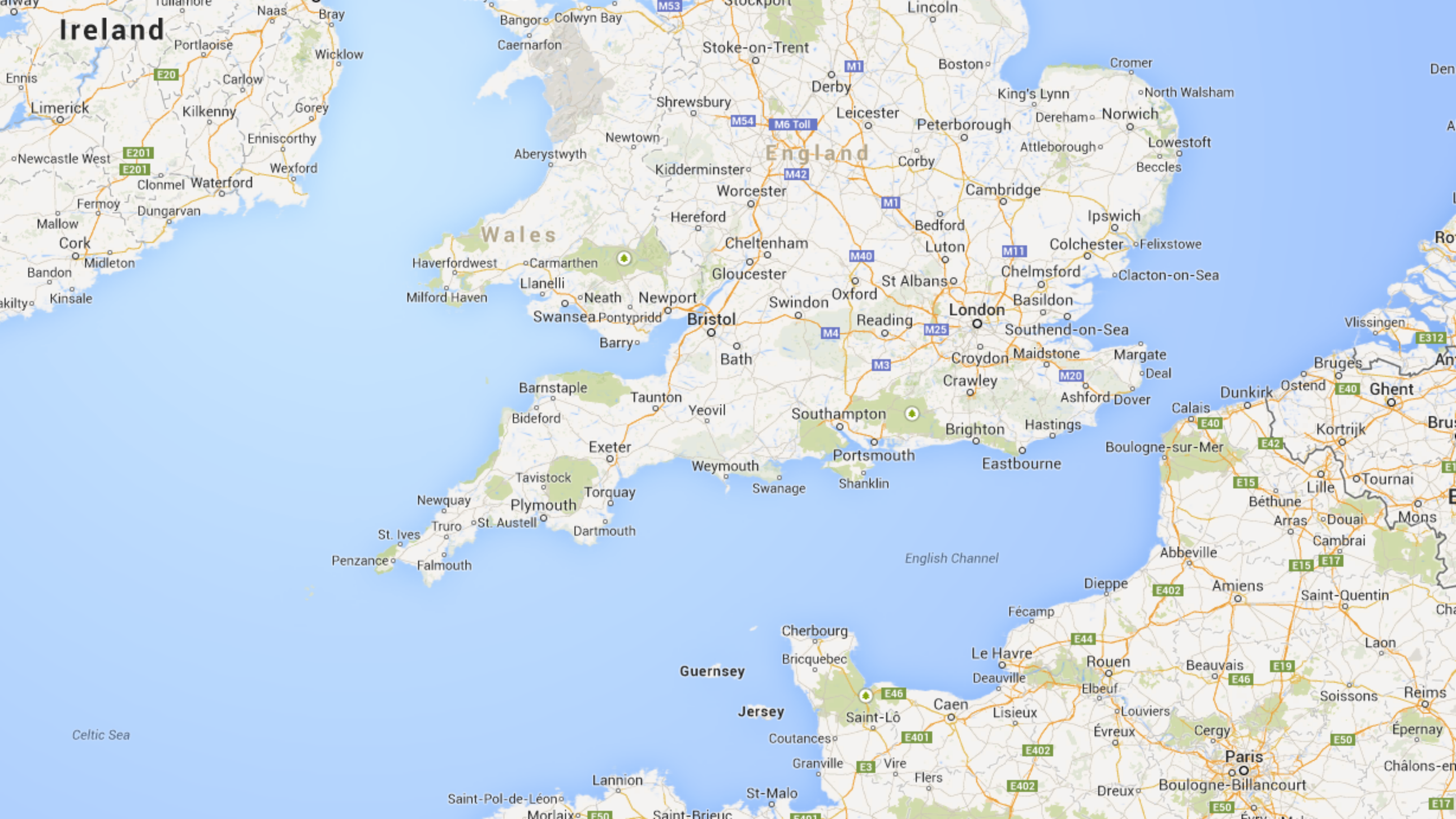

La Rue du Mont


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


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. Malo, FR - Portsmouth, UK
St. Peter Port, Gg - Saint Helier, Je
St. Helier, Je - Saint Peter Port, Gg
Granville, FR - St. Helier, JE
Poole - Jersey (St. Helier)
Jersey (St. Helier) - Poole


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

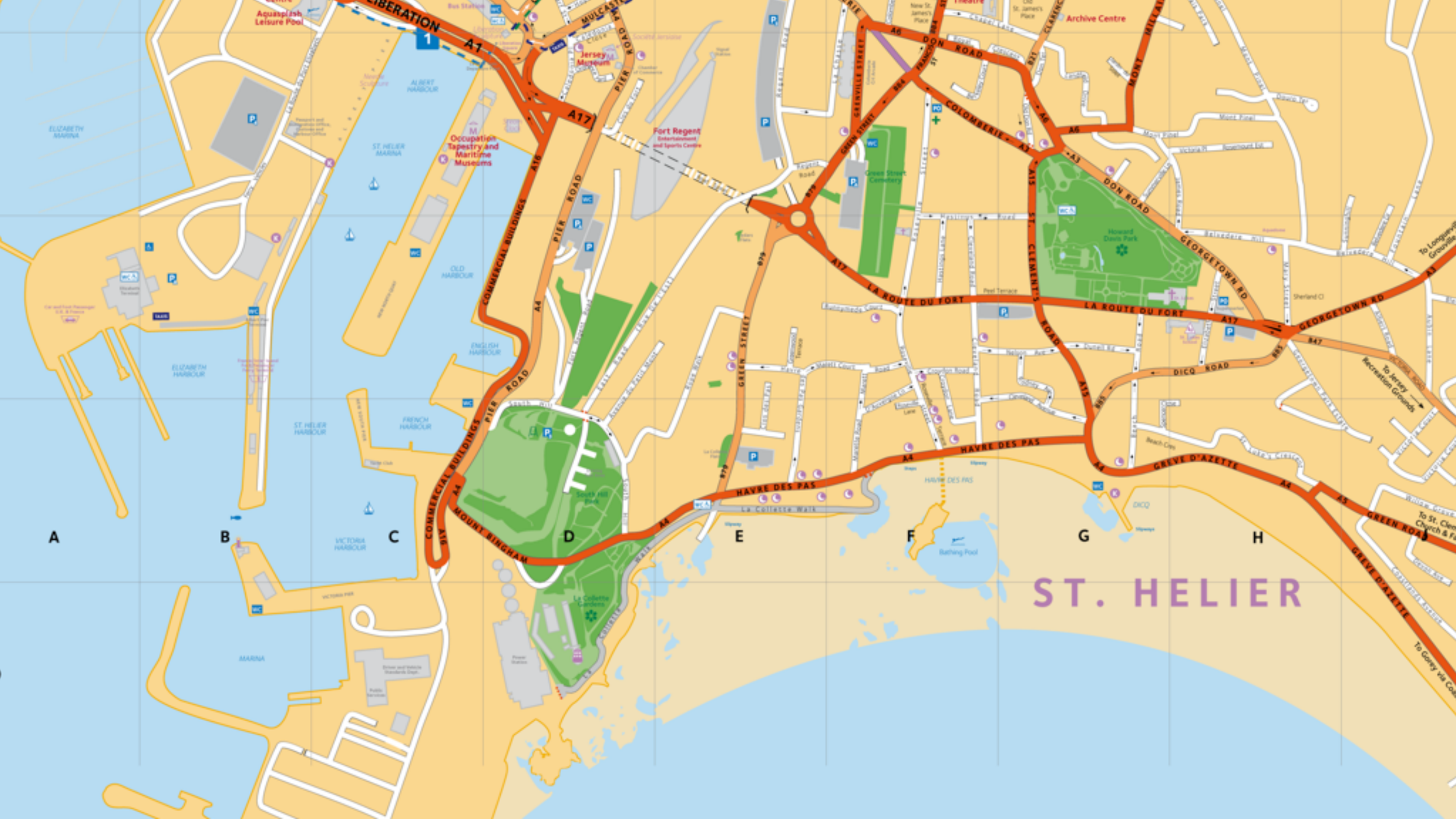

St. Anne
Auderville
Beaumont-Hague
Querqueville
Fermanville
Saint-Pierre-Église
Barfleur
Cherbourg
E46
Réville
La Glacerie
Martinvast
Saint-Vaast-la-Hougue
Hardinvast
Brix
Flamanville
Les Pieux
Valognes
Montebourg
E46
Bricquebec
Guernsey
Sainte-Mère-Église
Saint-Sauveur-le-Vicomte
Barneville-Carteret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

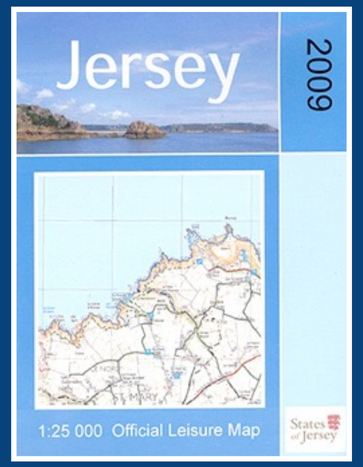

Ireland
Portlaoise
Naas
Bray
Wicklow
Ennis
Carlow
Limerick
Kilkenny
Gorey
Enniscorthy
Newcastle West
Wexford
Clonmel
Waterford
Fermoy
Dungarvan
Mallow
Cork
Midleton
Bandon
Kinsale
Bangor
Colwyn Bay
Caernarfon
Stoke-on-Trent
Lincoln
Boston
Cromer
Derby
King's Lynn
North Walsham
Shrewsbury
Leicester
Peterborough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

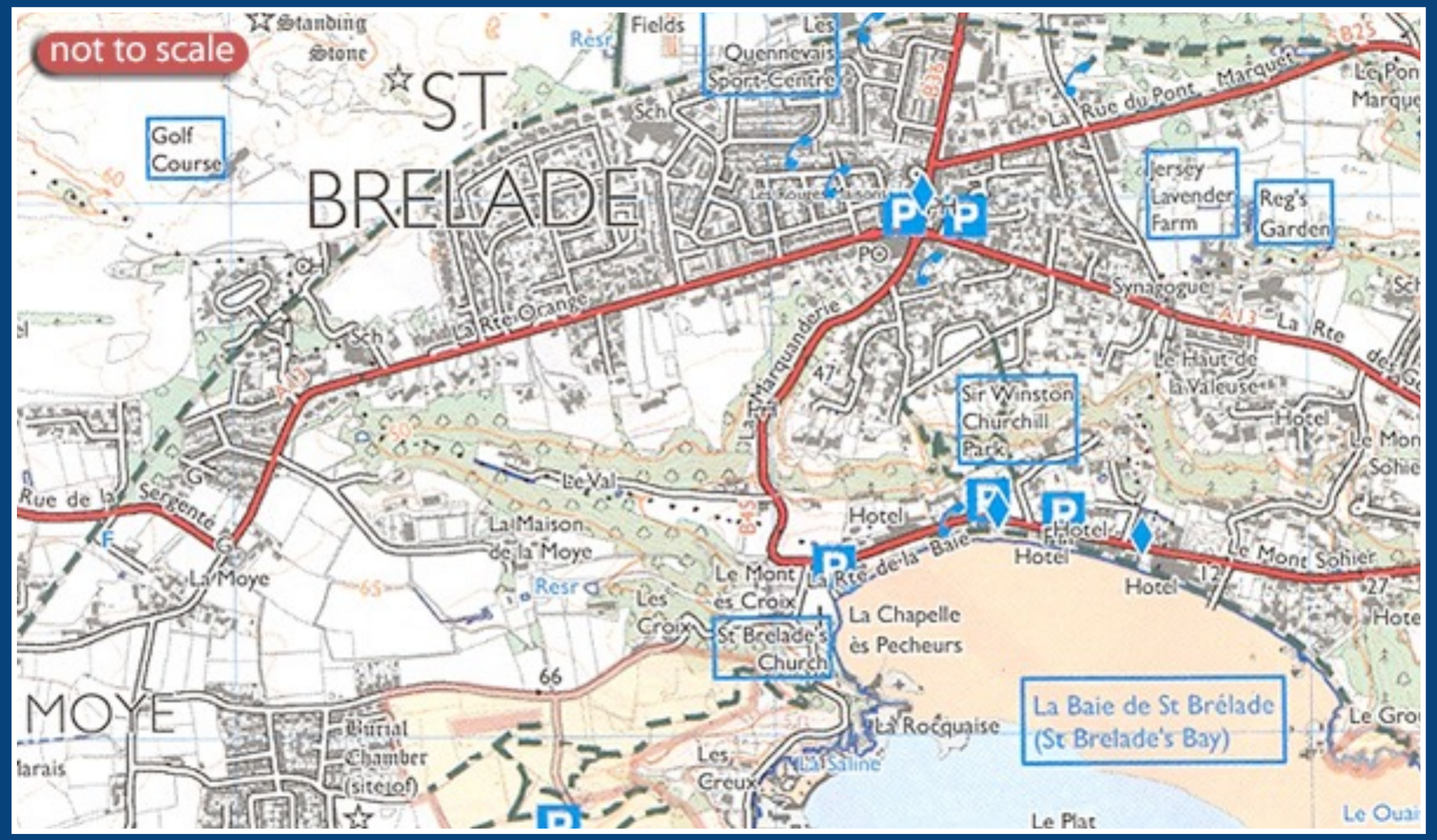

North Sea
Perth
Kirkcaldy
Glasgow
Kilmarnock
Ayr
United Kingdom
Londonderry / Derry
Northern Ireland
Bangor
Sligo
Isle of Man
Dundalk
Castlebar
Drogheda
Dublin
Galway
Ireland
Naas
Bray
Limerick
Carlow
Kilkenny
Tralee
Clonmel
Cork
Waterford
Middlesbrough
Blackpool
Leeds
Hull
Liverpool
Sheffield
Stoke-on-Trent
Nottingham
Derby
Norwich
Great Yarmouth
England
Ipswich
Wales
Headington
Swansea
Bristol
London
Leigh-on-Sea
Cardiff
Reading
Barnstaple
Southampton
Exeter
Portsmouth
Bournemouth
Plymouth
Newquay
St. Ives
Truro
English Channel
Celtic Sea
Guernsey
Jersey
Le Havre
Caen
Rouen
Amiens
Saint-Quentin
Reims
Paris
Évry
Lannion
St-Malo
Brest
Saint-Brieuc
Rennes
Quimper
Le Mans
Orleans
Lorient
Vannes
Angers
Tours
Nantes
Saint-Nazaire
Troyes
Dijon
Besançon
Viborg
Aarhus
Silkeborg
Horsens
Esbjerg
Denmark
Odense
Helsingborg
Kristianstad
Karlskrona
Copenhagen
Malmö
Flensburg
Kiel
Heide
Stralsund
Rostock
Swinoujscie
Anklam
Gryfice
Kosza
Aurich
Bremerhaven
Hamburg
Schwerin
Neubrandenburg
Szczecin
Bergedorf
Groningen
Bremen
Stargard Szczecinski
Den Helder
Gorzów Wielkopolski
Berlin
Amsterdam
Hanover
The Hague
Netherlands
Bielefeld
Brunswick
Magdeburg
Dessau
Zielona Gora
Münster
Kernstadt
Cottbus
Woensel-Noord
Leipzig
Bruges
Antwerp
Dusseldorf
Nord
Dresden
Görlitz
Brussels
Germany
Chemnitz
Liberec
Lille
Cologne
Gera
Ústí nad Labem-město
Belgium
Frankfurt
Prague
Wiesbaden
Würzburg
Plzen
Luxembourg
Mannheim
Nuremberg
Saarbrücken
Regensburg
Karlsruhe
Ceske Budejovice
Metz
Stuttgart
Ingolstadt
Strasbourg
Landshut
Nancy
Ulm
Linz
Munich
Freiburg
Salzburg
Wels
Konstanz
Basel
Zurich
Innsbruck
Austria

Enough detail to start **exploring**


01
02
03
Grosnez
La Grève au Lançon (Plémont)
Grosnez Castle (ruins)
Plémont
Creux Gros
Creux Gabourel
Douet de la Mer
Race Course
Les Landes
Portinfer
Ville la Bas
Les Pallières
Rue de la Pointe
Route de Vinchelez
Le Pinacle
Model Aircraft Field
Le Pulec
La Route de Millais
La Route de Trodez
Sea Wall
La Villaise
L'Étacq
Treasures of the Earth
Battle of Flowers Museum
Les Prés
Jersey Pearl
Channel Islands Military Museum
La Route des Laveurs
La Grande Route des Mielles
Barge Aground
La Mielle de Morville
Dolmen du Mont Grantez
La Crabière
Le Canné
Route du Marais
ST. OUEN
Ville au Bas
St Ouen's Parish Church
La Ville de L'Église
St Ouen's Manor
St Ouen's Pond
Les Mielles
Le Mont Rossignol
Val de la Mare Reservoir
Mont du Jubilé
St Peter's Parish Church & Hall
Le Moulin de Quétivel
Ville Emphrie
GREVE DE LECQ
Grève de Lecq Barracks
Creux de Lasse
Rifle Range
Crabbé
Les Reuses
Devil's Hole
Bleu Soleil Campsite
Léoville
Judith Quérée Garden
Bouchet Agateware Pottery
Catherine Best Jewellery
La Grande Route de St. Ouen
La Grande Route de St. Pierre
aMaizin! Maze Adventure Park & Craft Centre
Living Legend
ST. MARY
La Mare Wine Estate
St Mary's Parish Hall
St Mary's Parish Church
La Rue du Pont
La Rue de la Vallée
The Elms
Jersey Goldsmiths & Lion Park
Carrefour
Sorel
La Houle
Ronez
Le Fosse Vicq
Go Kart Racing
Moto Cross
Mourier Bay
ST. JOHN'S BAY
Wolf's Caves
Frémont
ST. JOHN
St John's Parish Church
Parish Hall
La Rue de L'Étocquet
La Route de St. Jean
La Grande Route de St. Laurent
Mont Mado
Route des Issues
Le Claire Stables
Bon Air
Handois Reservoir
BSJA Showground
1
4
6
6a

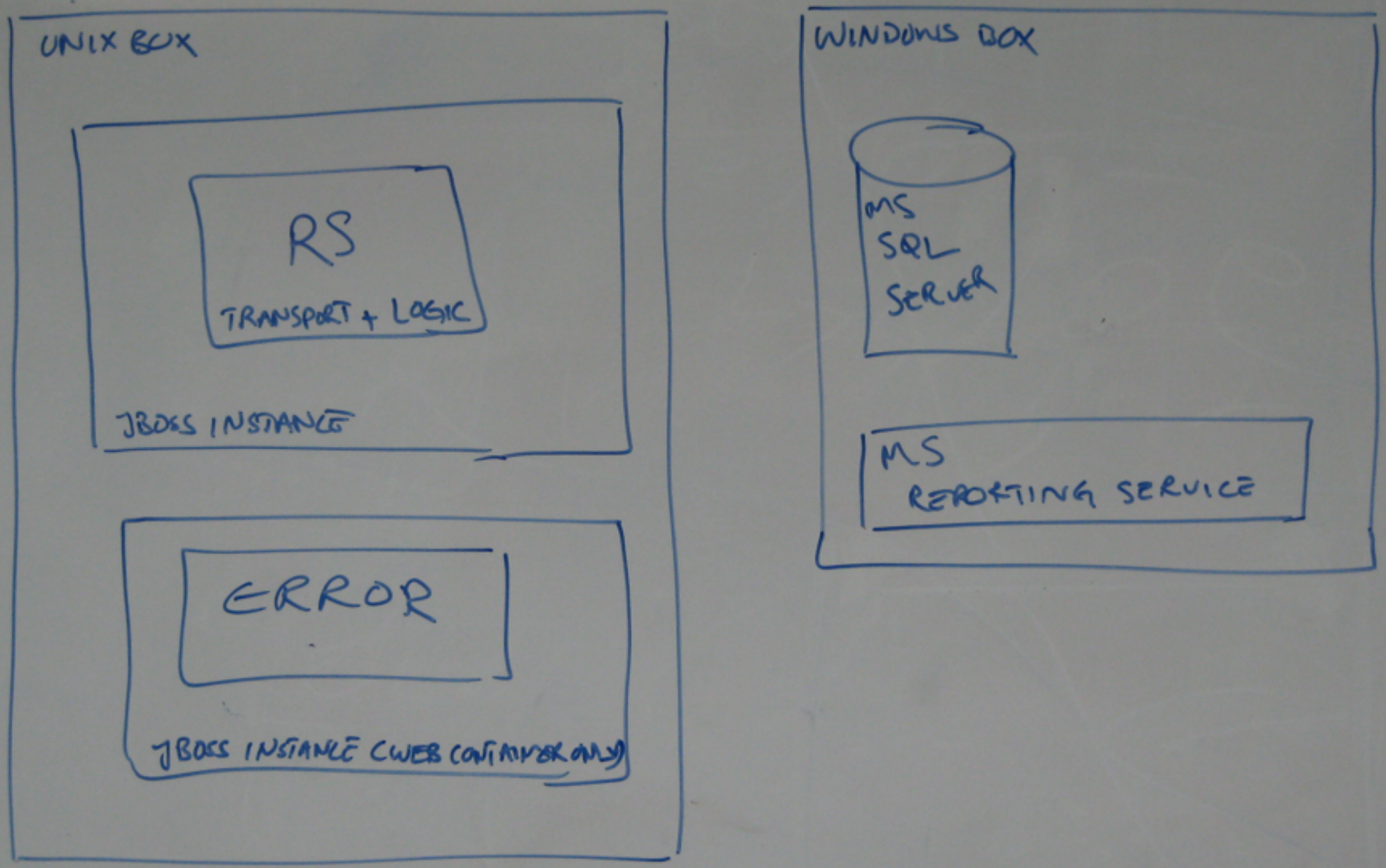

ST. HELIER
A
B
C
D
E
F
G
H
Aquasplash Leisure Pool
Jersey Museum
Occupation Tapestry and Maritime Museums
Fort Regent
Entertainment and Sports Centre
Archive Centre
Howard Davis Park
Green Street Cemetery
South Hill Park
La Collette Gardens
Bathing Pool
ELIZABETH MARINA
ST. HELIER MARINA
ALBERT HARBOUR
OLD HARBOUR
ENGLISH HARBOUR
ELIZABETH HARBOUR
ST. HELIER HARBOUR
FRENCH HARBOUR
VICTORIA HARBOUR
MARINA
LIBERATION
A1
A17
A16
A4
A6
A3
A15
A13
B79
B84
B47
B85
PIER ROAD
COMMERCIAL BUILDINGS
MULCASTER
MOUNT BINGHAM
GREEN STREET
GRENVILLE STREET
COLOMBERIE
DON ROAD
MONT MILLAIS
ST. CLEMENT'S ROAD
LA ROUTE DU FORT
GEORGETOWN RD
DICQ ROAD
HAVRE DES PAS
GREVE D'AZETTE
GREEN ROAD
La Collette Walk
To Longueville Grounds
To Jersey Recreation Grounds
To St. Clement Church & Fa
To Gorey via Co

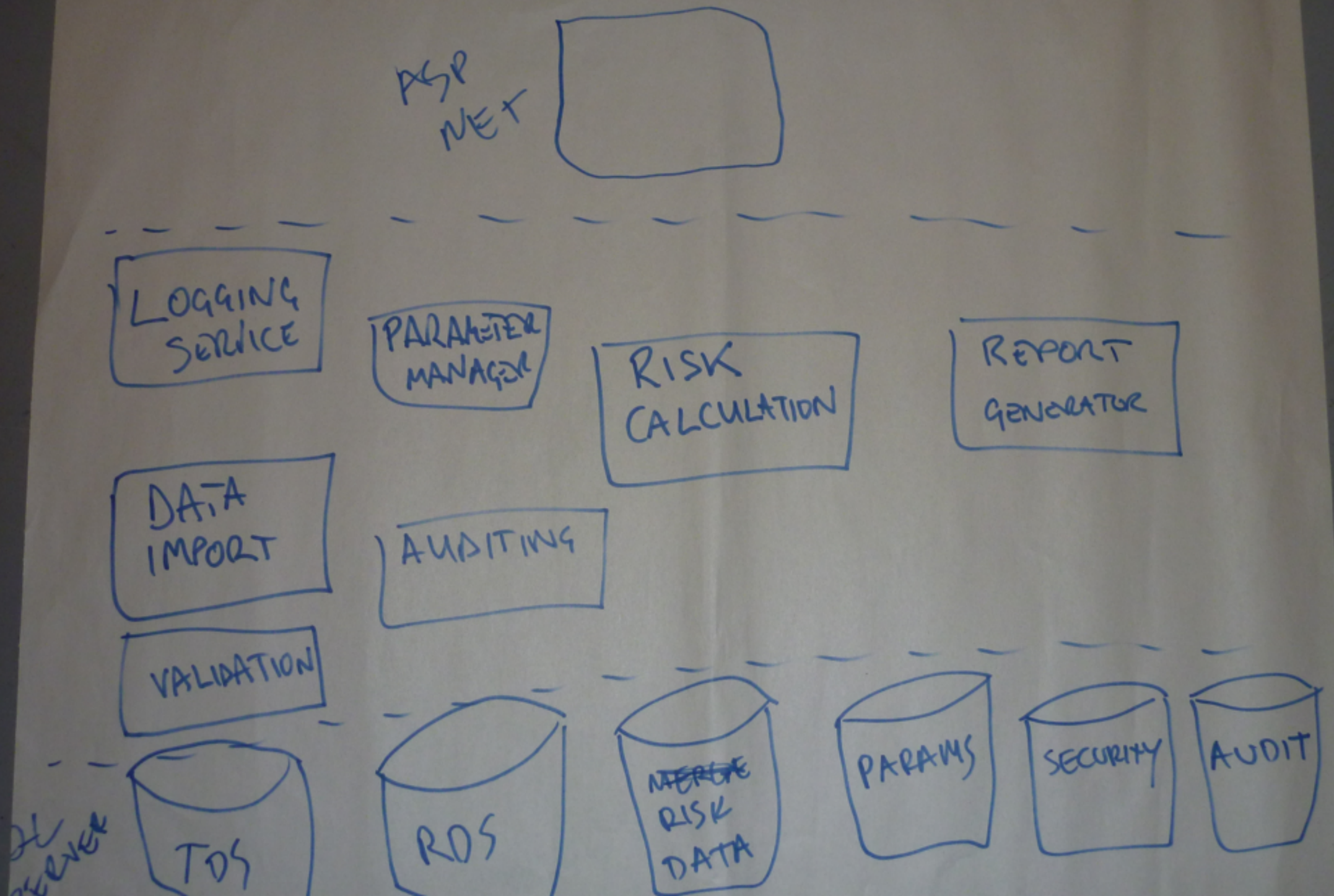




Very detailed and precise

(terrain, buildings, etc)


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
~~MERGE~~ RISK DATA
PARAMS
SECURITY
AUDIT

FUNCTIONAL VIEW

File Retriever

Scheduler

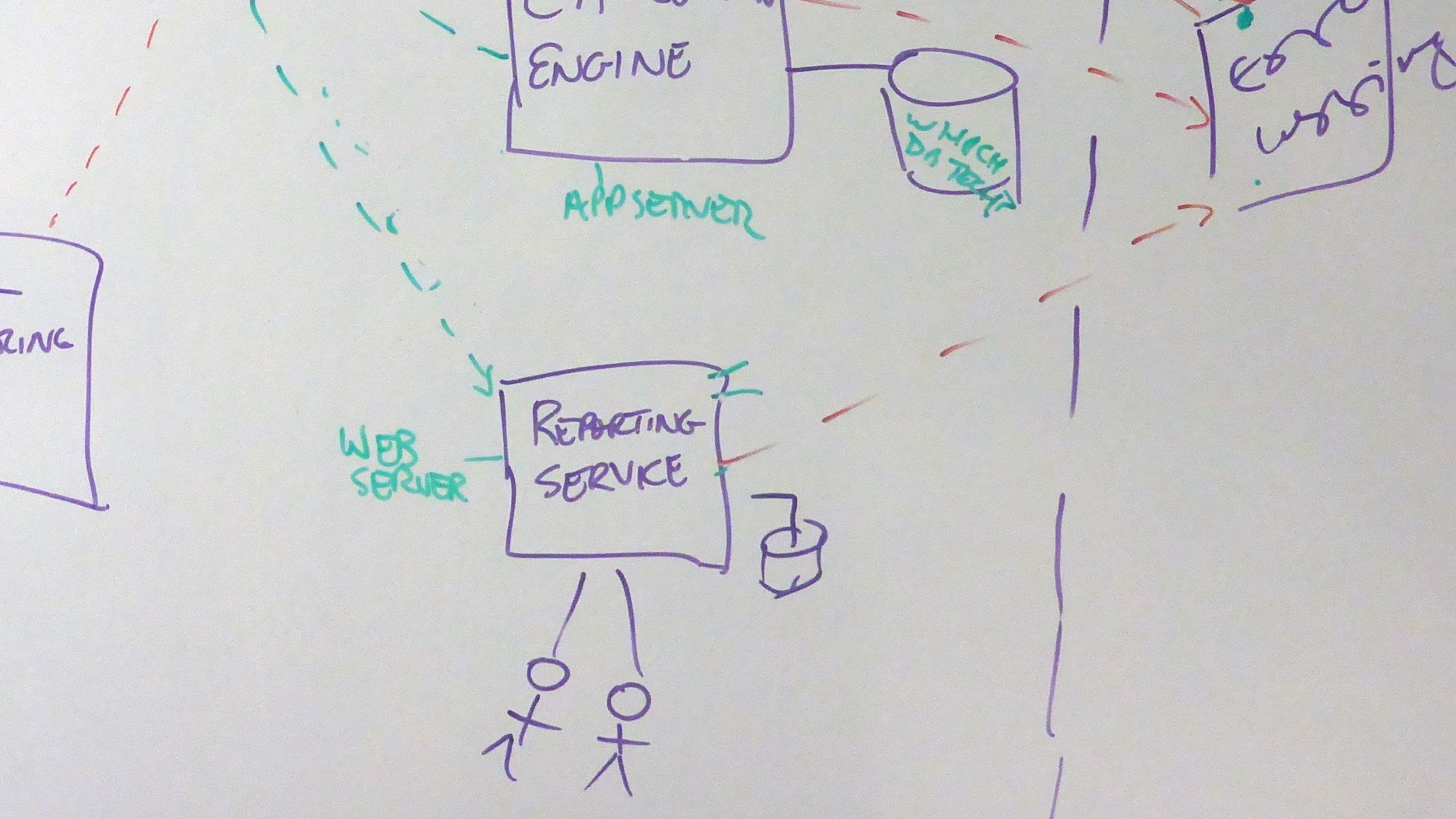
Auditing

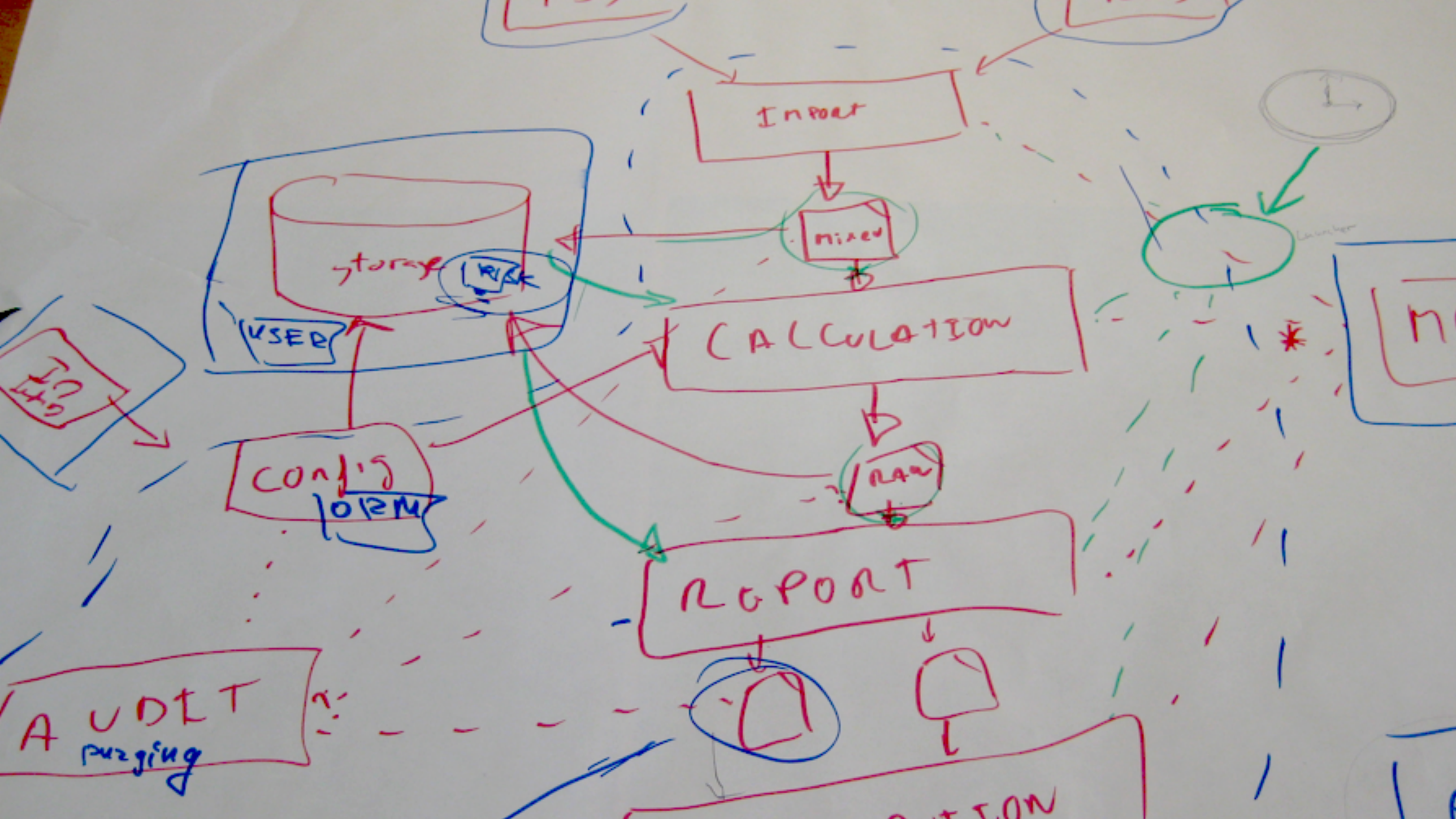
Reference Archiver

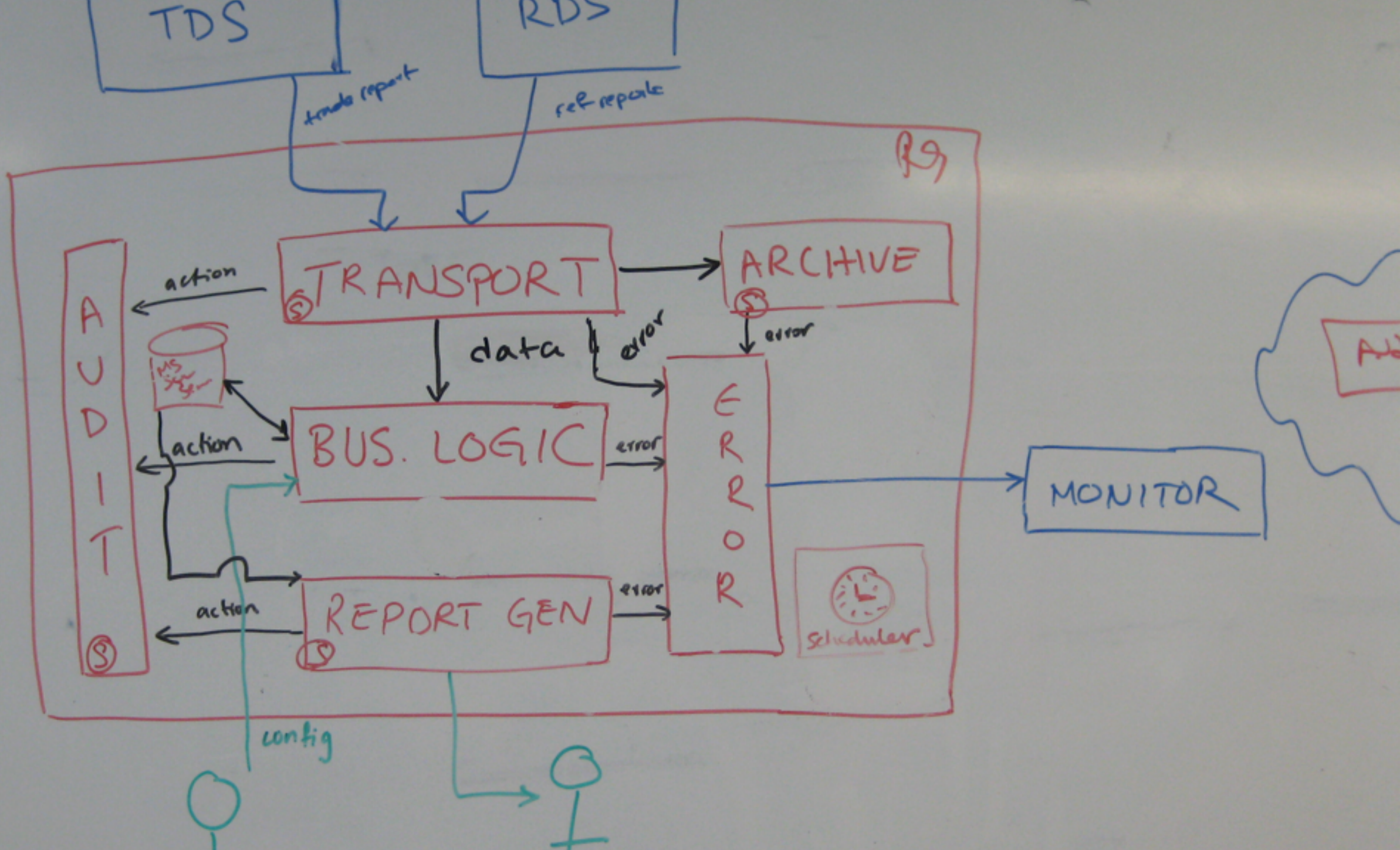
Risk Assesment Processor

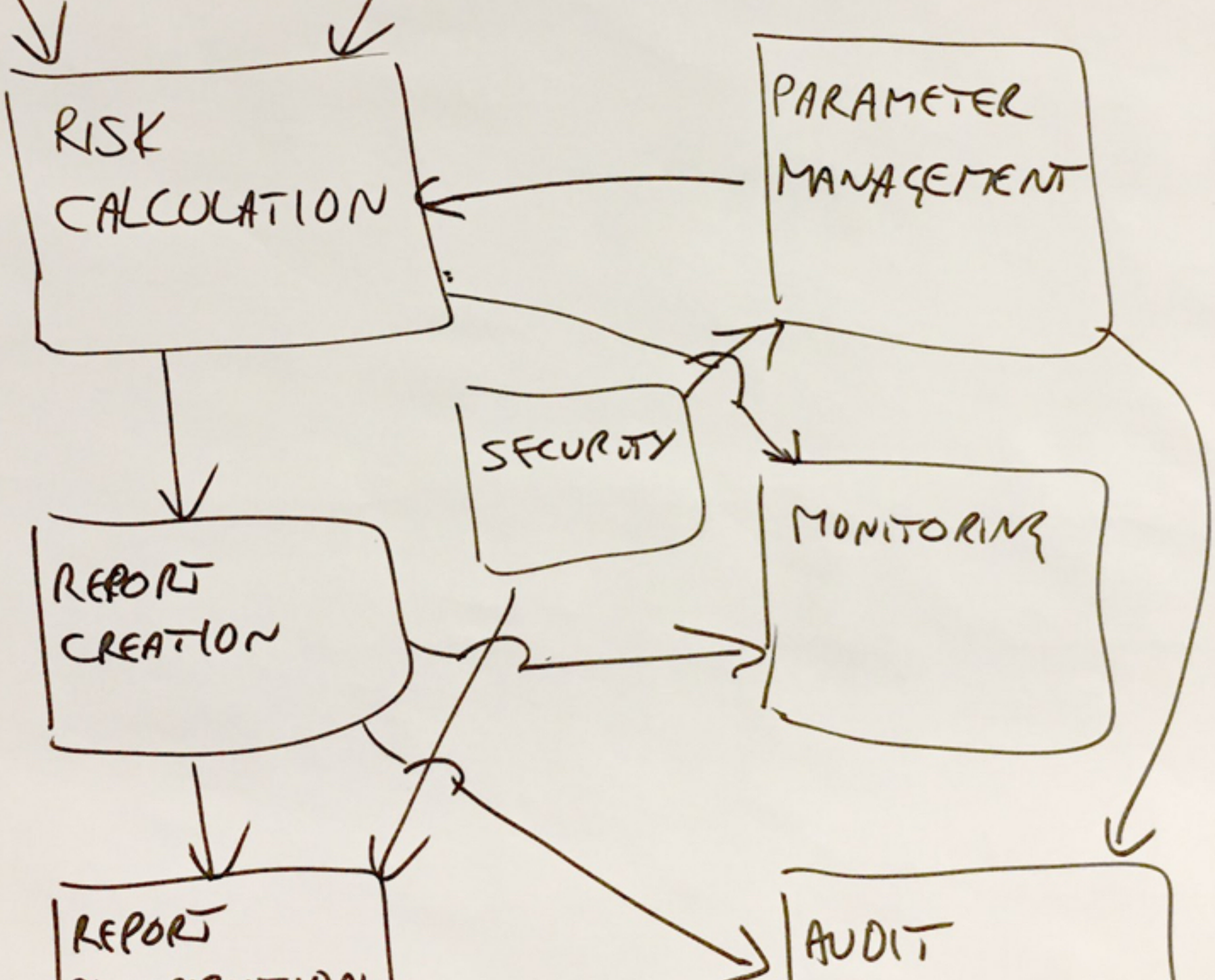
Risk Parameter Configuration

Report

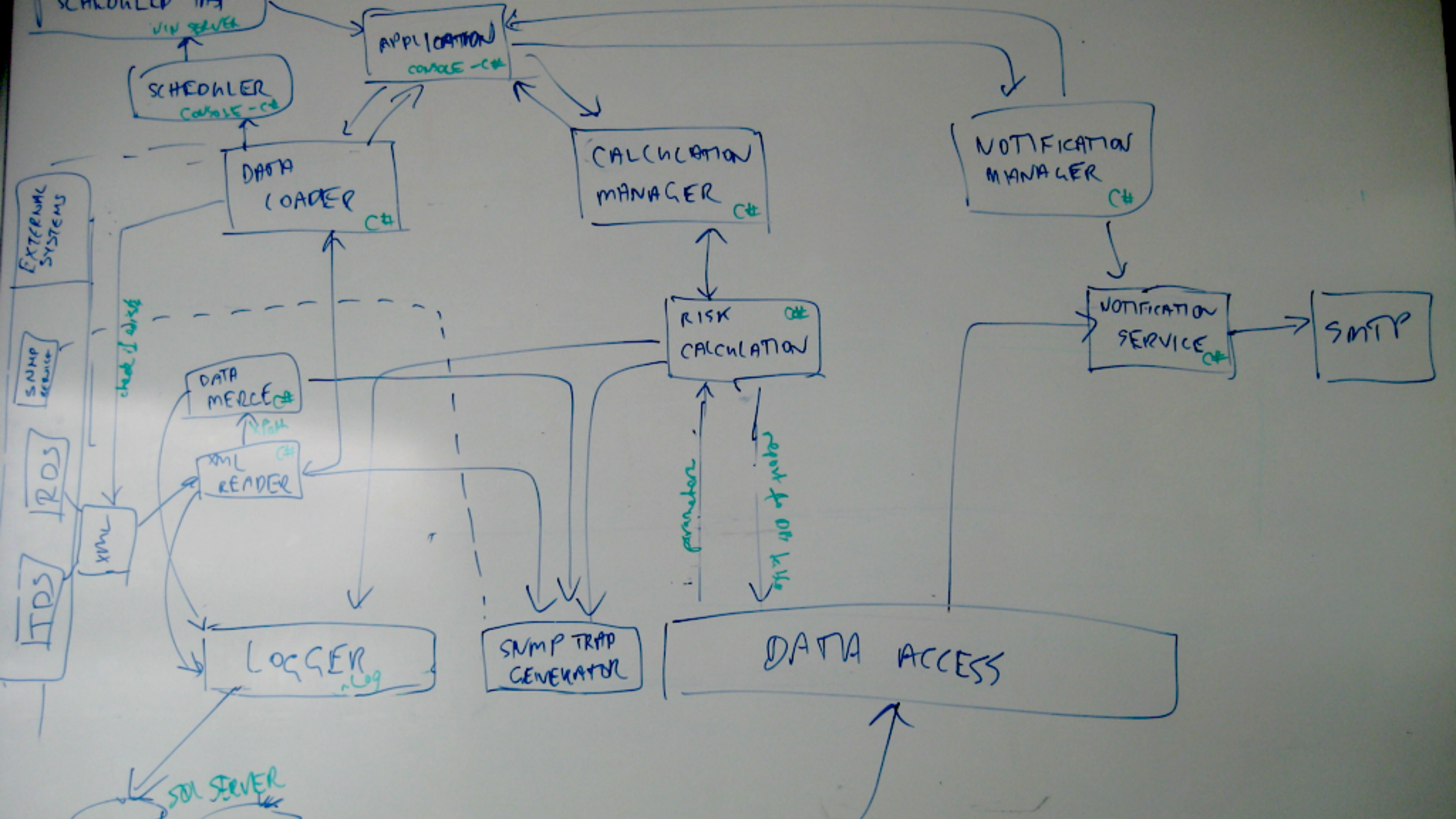

ENGINE
APP SERVER
WEB SERVER
REPORTING SERVICE

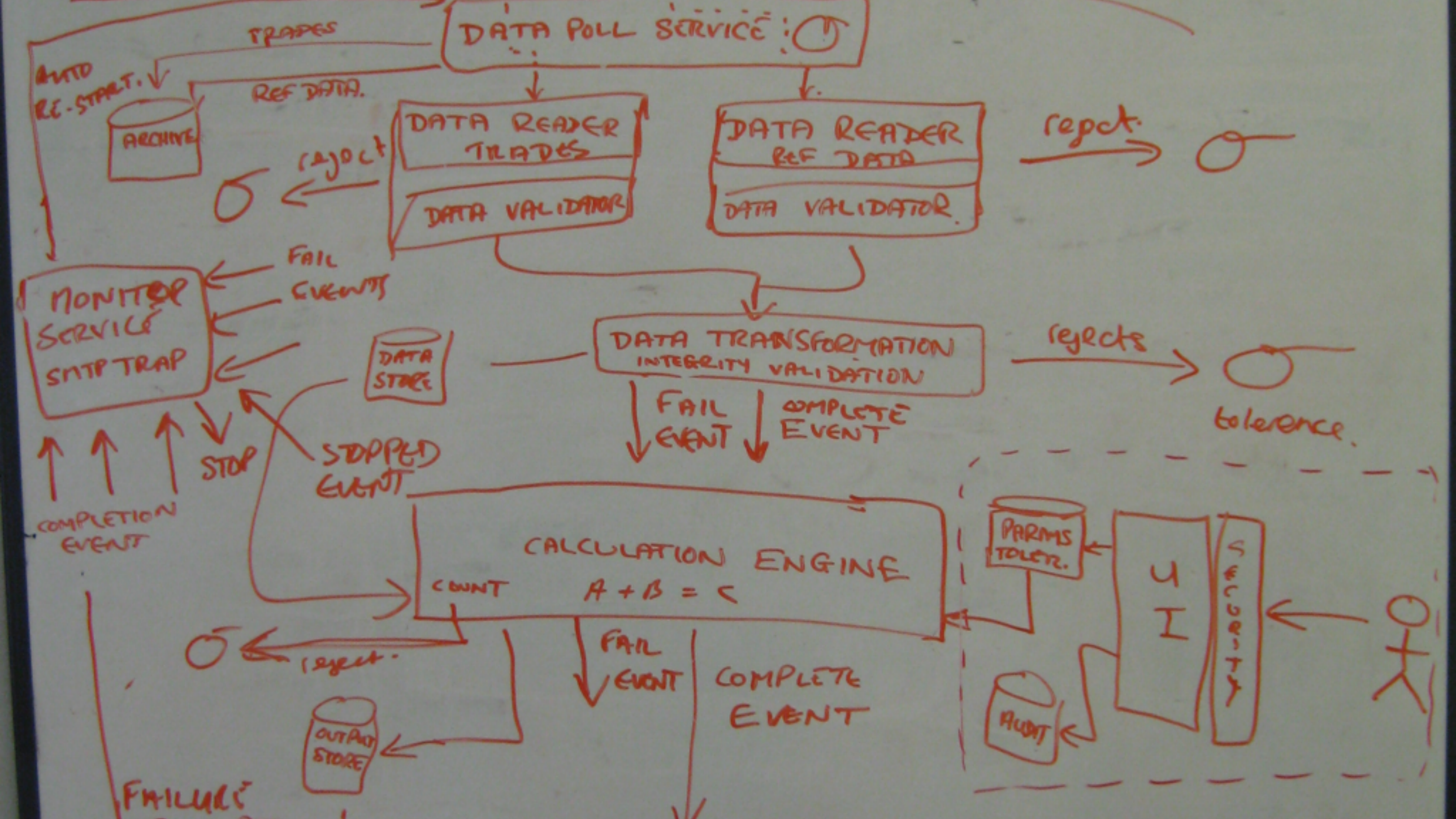

Import
mixer
storage
RISK
USER
CALCULATION
GTI
config
ORM
RAW
REPORT
AUDIT
purging


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


RISK CALCULATION
PARAMETER MANAGEMENT
SECURITY
MONITORING
REPORT CREATION
REPORT
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
VIN SERVER
SCHEDULER
CONSOLE - C#
APPLICATION
CONSOLE - C#
DATA LOADER
C#
CALCULATION MANAGER
C#
NOTIFICATION MANAGER
C#
EXTERNAL SYSTEMS
SNMP
ROS
TDS
XML
check if alive
DATA MERGE
C#
XPath
XML READER
C#
RISK CALCULATION
C#
parameters
output of the risk
NOTIFICATION SERVICE
C#
SMTP
LOGGER
.Log
SNMP TRAP GENERATOR
DATA ACCESS
SQL SERVER

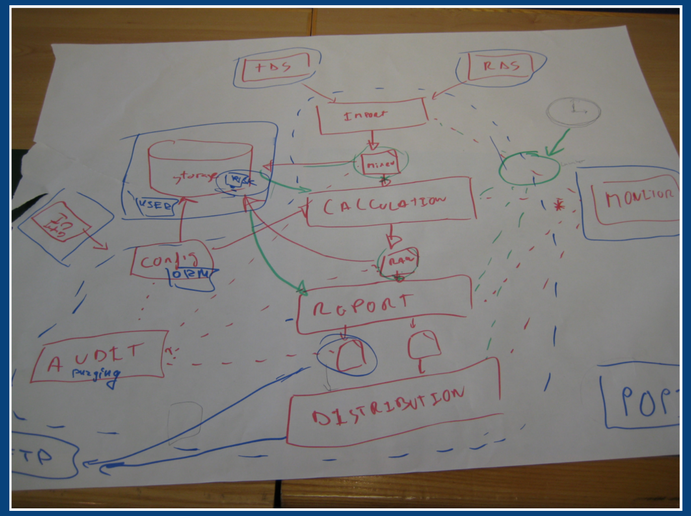

DATA POLL SERVICE
TRADES
AUTO RE-START.
REF DATA.
ARCHIVE
DATA READER TRADES
DATA VALIDATOR
reject
DATA READER REF DATA
DATA VALIDATOR
reject
MONITOR SERVICE SMTP TRAP
FAIL EVENTS
DATA STORE
DATA TRANSFORMATION INTEGRITY VALIDATION
rejects
tolerence.
FAIL EVENT
COMPLETE EVENT
STOP
STOPPED EVENT
COMPLETION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


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

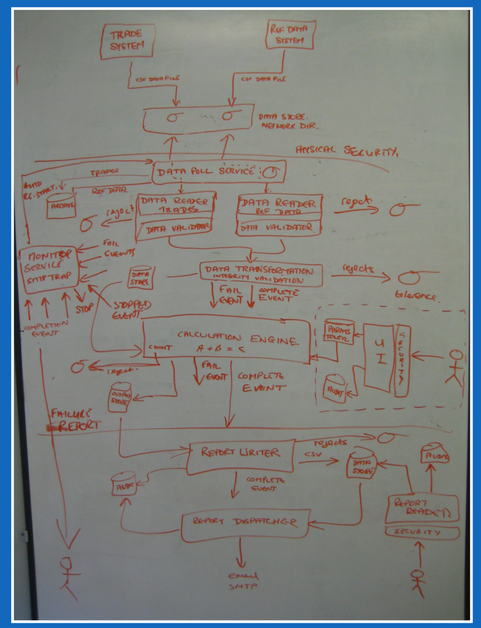

Params
Calcs
EH?
Floor
App Log
Date
- Risk Cell
- calcs
- Par
Batch
Params_Distrib
B_id:
Date id

“This doesn’t make sense, but we’ll explain it.”

1 1 1


TDS
RDS
Import
Mixer
storage
MQ
VSEB
CALCULATION
Raw
Config
OPN
REPORT
AUDIT
Parsing
DISTRIBUTION

2 2 2


TRADE SYSTEM
REF DATA SYSTEM
DATA SERVE NETWORK DIR.
PHYSICAL SECURITY
DATA PULL SERVICE
DATA READER TRADES
DATA READER REF DATA
DATA VALIDATOR
reports
MONITOR SERVICE SNMP TRAP
DATA TRANSFORMATION INTEGRITY VALIDATION
rejects
balance
STOPPED EVENT
STOP
CALCULATION ENGINE
A + B = C
FAIL EVENT
COMPLETE EVENT
UI
FAILURE REPORT
REPORT WRITER
REPORT DISPATCHER
REPORT BROKER
SECURITY
email SMTP

What does colour mean?

NO ANNOTATION ON FLOWS

SHOULD USE MORE COLORS

POST ITS CAN FALL OFF

Objects vs actions

—

MIXES DIFFERENT LEVELS OF DETAIL

NOT SURE OF TRANSITION BETWEEN DIFFERENT DIAGRAMS -

—

CONFLICTING LEVELS OF DETAIL IN PRESENTATION

What about the different arrows?

What shapes mean

WHY ARE SOME LINES PINK?

WHAT DO THE SHAPE MEAN?

UML IS GOOD, BUT NOT EVERYONE KNOWS IT

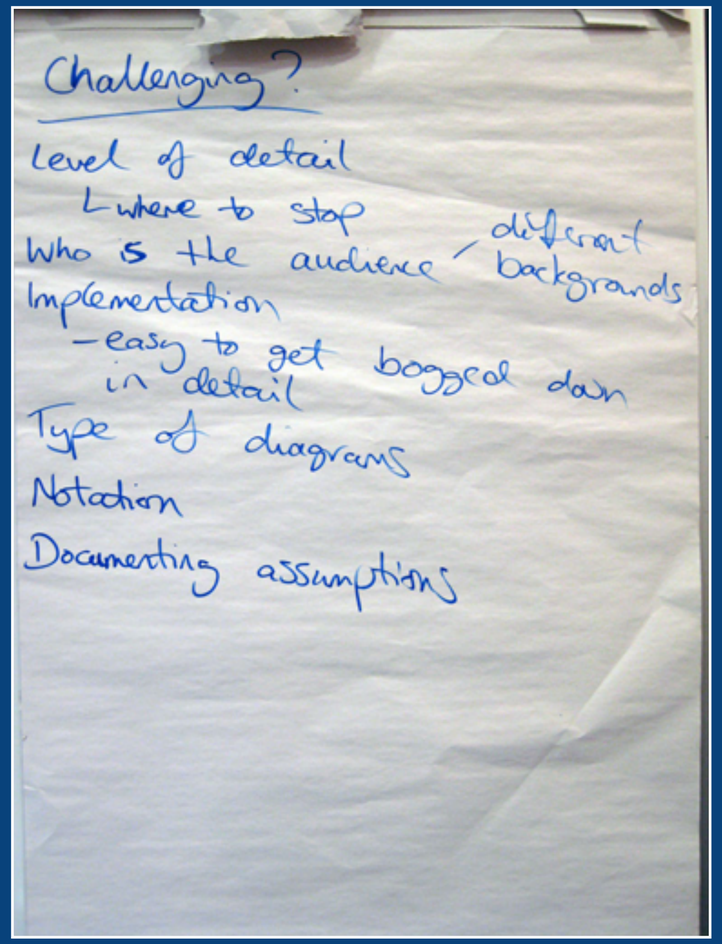
WHAT DO LINES REPRESENT? (DATA? CONTROL? DEP.?)

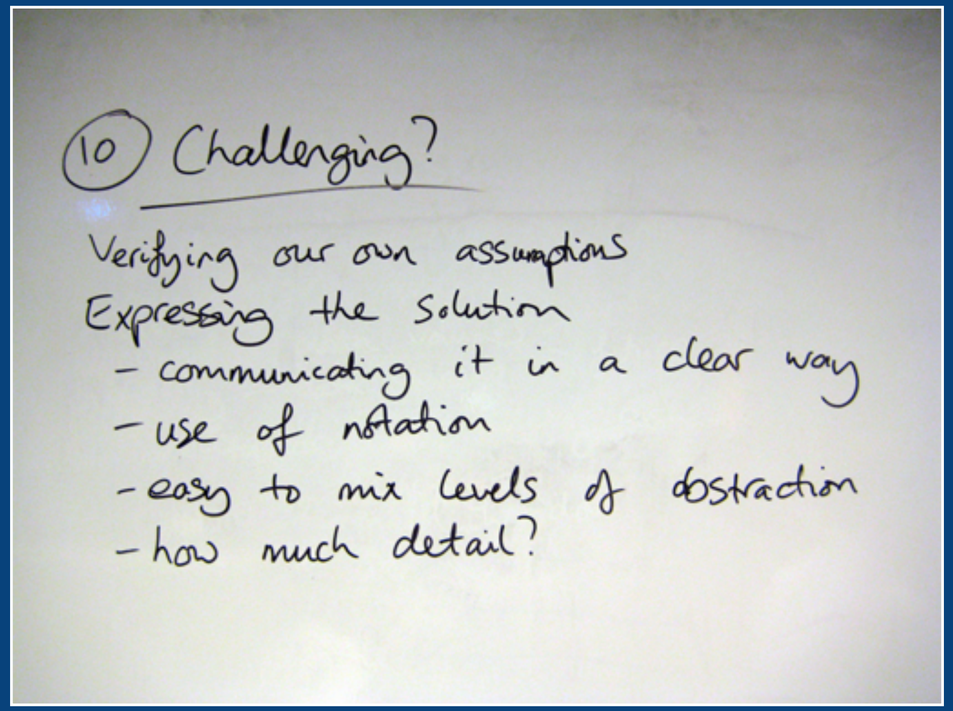
What's the DB-like icon?

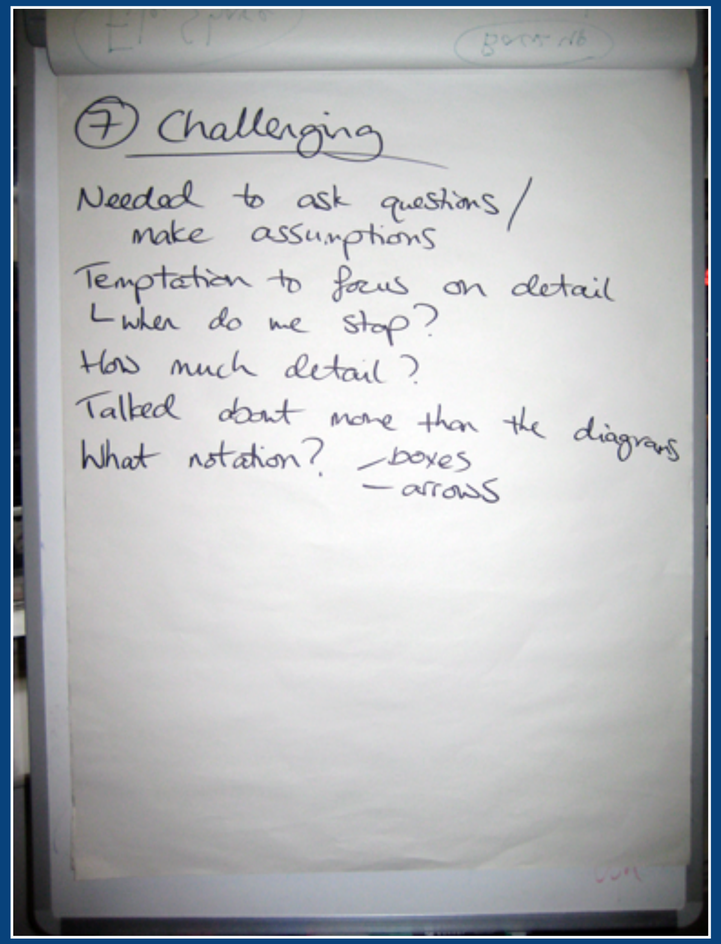
Not sure what this is

DIFFERENT LEVELS IN SAME DIAGRAM

ARE THE ARROWS THE RIGHT WAY ROUND?







Did you find this exercise challenging?

I've run this workshop
in **25+ countries**
for **10,000+ people**

Software architects

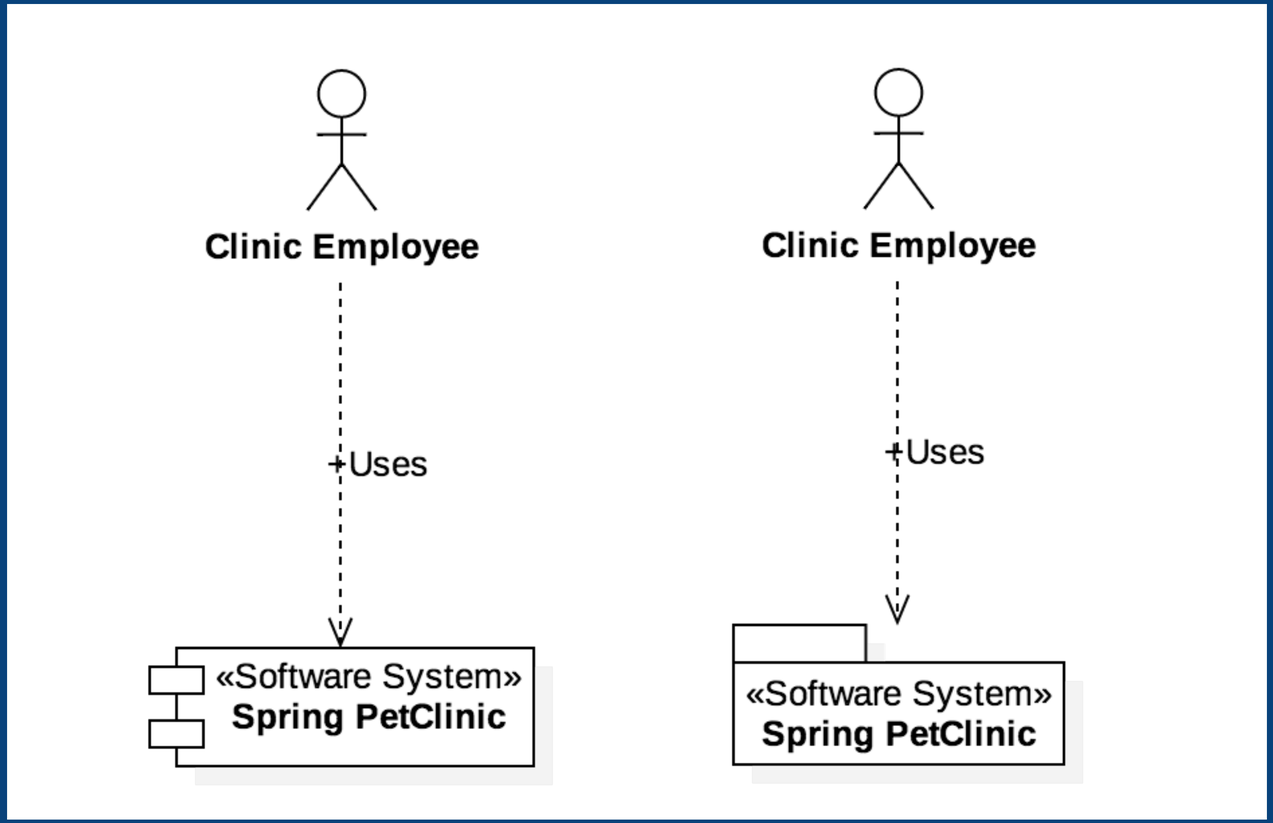
struggle to communicate

software architecture

Do **you** use UML?

In my experience, optimistically,

1 out of 10 people use UML



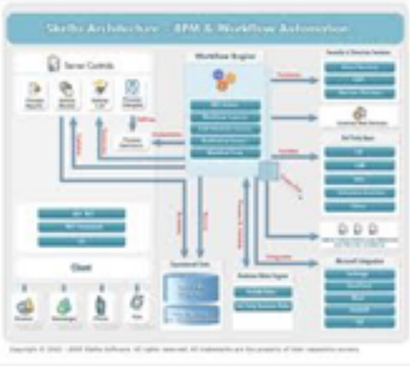
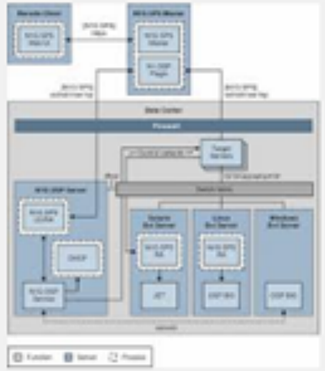
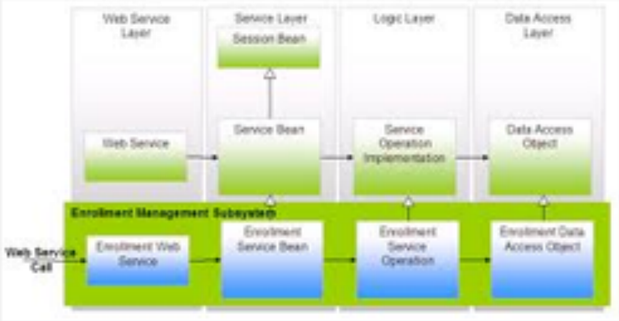
I do use UML, but not for software architecture

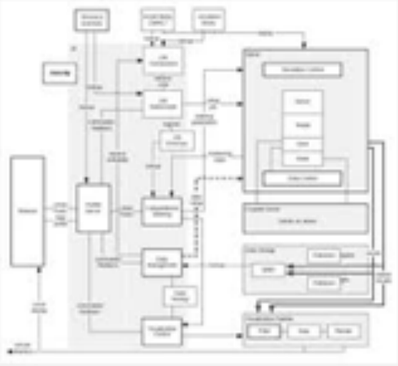
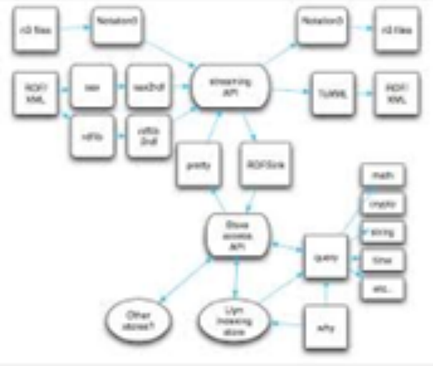
Google software architecture diagram

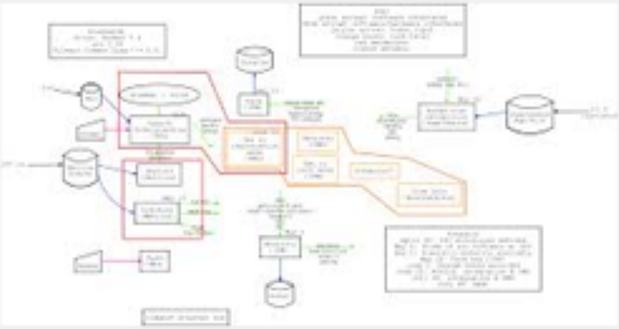
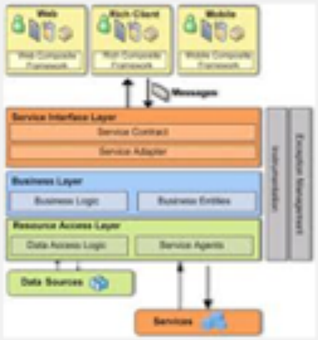
Sign in

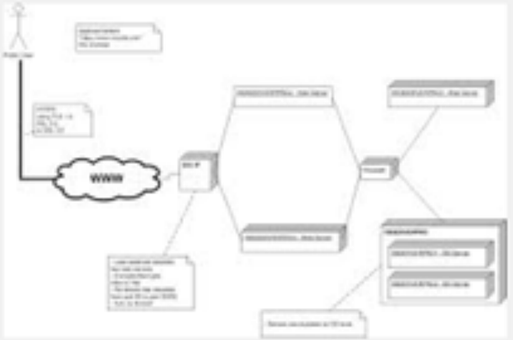
Web Images Videos Shopping News More Search tools

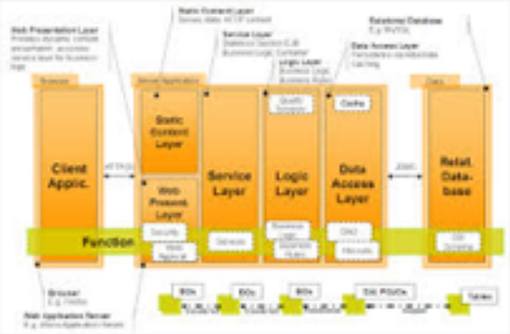
SafeSearch

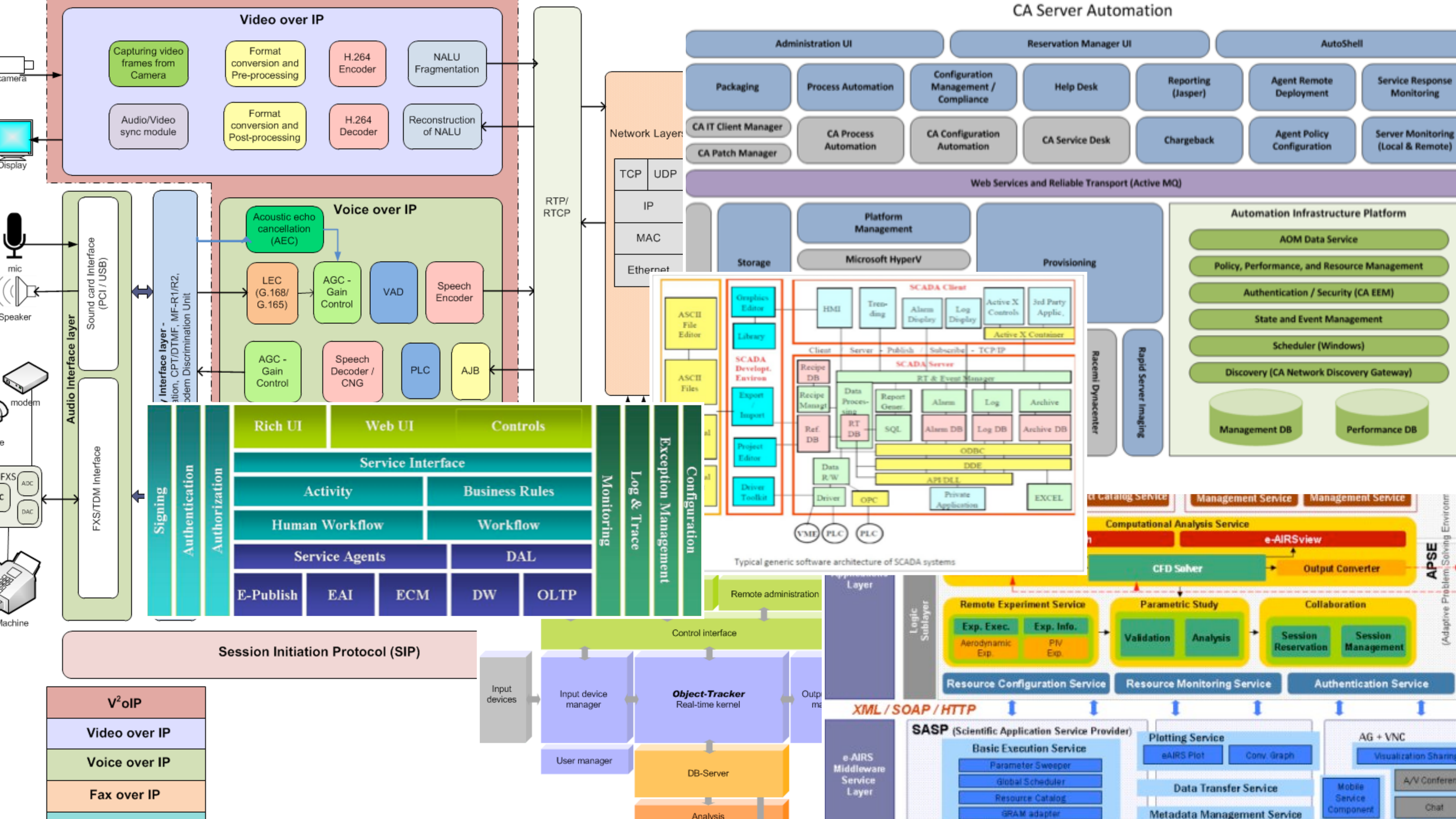










Video over IP
Capturing video frames from Camera
Format conversion and Pre-processing
H.264 Encoder
NALU Fragmentation
Audio/Video sync module
Format conversion and Post-processing
H.264 Decoder
Reconstruction of NALU
camera
Display
Voice over IP
Acoustic echo cancellation (AEC)
LEC (G.168/ G.165)
AGC - Gain Control
VAD
Speech Encoder
AGC - Gain Control
Speech Decoder / CNG
PLC
AJB
mic
Speaker
modem
Audio Interface layer
Sound card Interface (PCI / USB)
FXS/TDM Interface
Interface layer - tion, CPT/DTMF, MF-R1/R2, odem Discrimination Unit
RTP/ RTCP
Network Layers
TCP
UDP
IP
MAC
Ethernet
Signing
Authentication
Authorization
Rich UI
Web UI
Controls
Service Interface
Activity
Business Rules
Human Workflow
Workflow
Service Agents
DAL
E-Publish
EAI
ECM
DW
OLTP
Monitoring
Log & Trace
Exception Management
Configuration
Session Initiation Protocol (SIP)
V²oIP
Video over IP
Voice over IP
Fax over IP
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 Process Automation
CA Configuration Automation
CA Service Desk
Chargeback
Agent Policy Configuration
Server Monitoring (Local & Remote)
CA Patch Manager
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
Racemi Dynacenter
Rapid Server Imaging
ASCII File Editor
ASCII Files
Graphics Editor
Library
SCADA Developt. Environ
Export / Import
Project Editor
Driver Toolkit
SCADA Client
HMI
Trending
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
Ref. DB
Data Process-ing
RT & Event Manager
RT DB
Report Gener.
SQL
Alarm
Log
Archive
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
Remote administration
Control interface
Input devices
Input device manager
Object-Tracker
Real-time kernel
User manager
DB-Server
Analysis
Management Service
Management Service
Computational Analysis Service
e-AIRSview
CFD Solver
Output Converter
APSE
(Adaptive Problem Solving Environm
Remote Experiment Service
Exp. Exec.
Exp. Info.
Aerodynamic Exp.
PIV Exp.
Parametric Study
Validation
Analysis
Collaboration
Session Reservation
Session Management
Resource Configuration Service
Resource Monitoring Service
Authentication Service
Logic Sublayer
XML / SOAP / HTTP
e-AIRS Middleware Service Layer
SASP (Scientific Application Service Provider)
Basic Execution Service
Parameter Sweeper
Global Scheduler
Resource Catalog
SRAM adapter
Plotting Service
eAIRS Plot
Conv. Graph
Data Transfer Service
Metadata Management Service
AG + VNC
Visualization Sharing
Mobile Service Component
A/V Conferen
Chat

Moving fast in the same direction
as a team requires

good communication

Titles

Short and meaningful, numbered if diagram order is important

Lines

Favour unidirectional arrows, add descriptive text to provide additional information

Layout

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

Labels

Be wary of using acronyms, especially those related to the business/domain that you work in

Colour

Ensure that colour coding is made explicit; watch out for colour-blindness and black/white printers

Orientation

Most important thing in the middle; be consistent across diagrams

Shapes

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

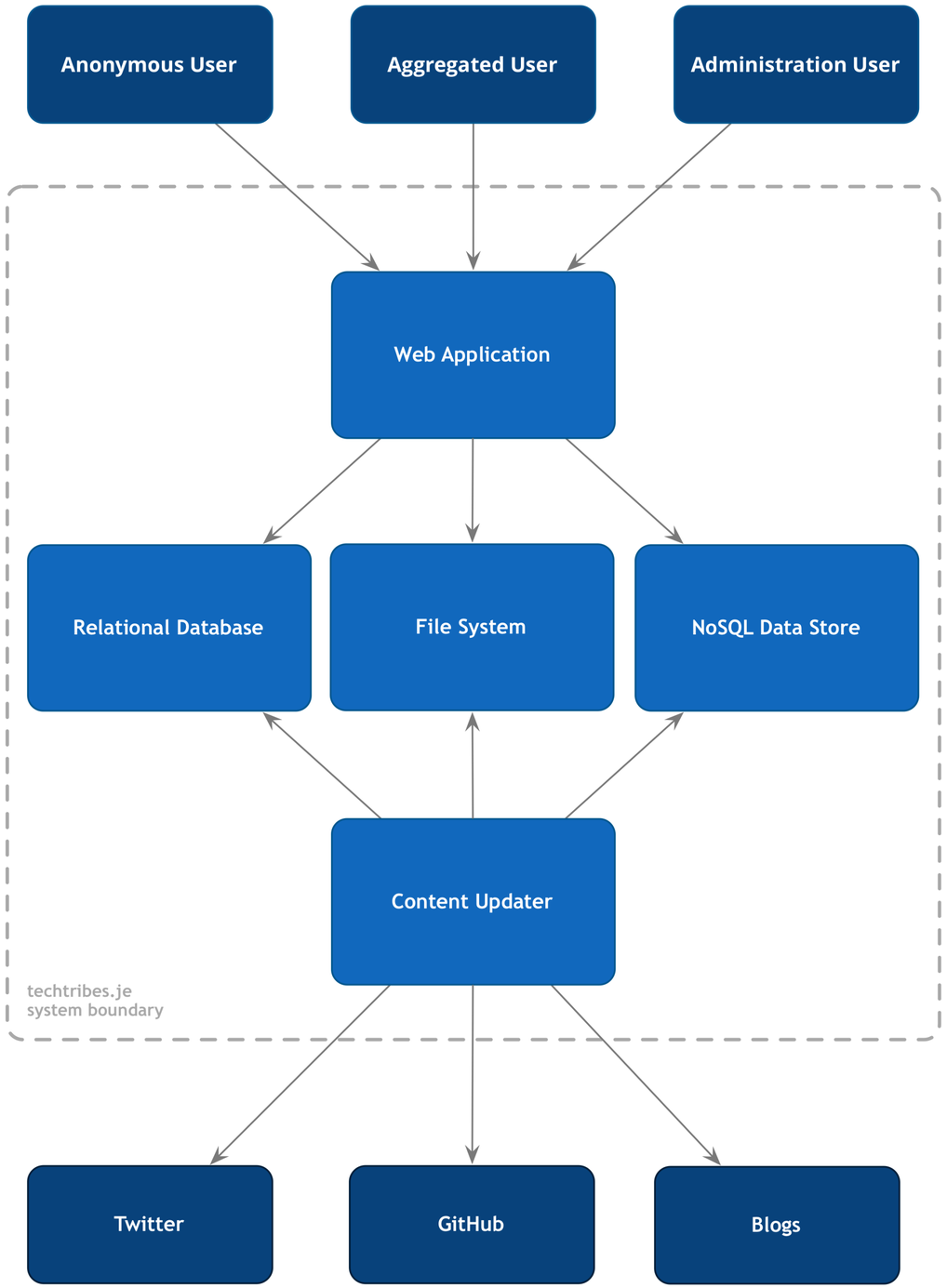
Keys

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

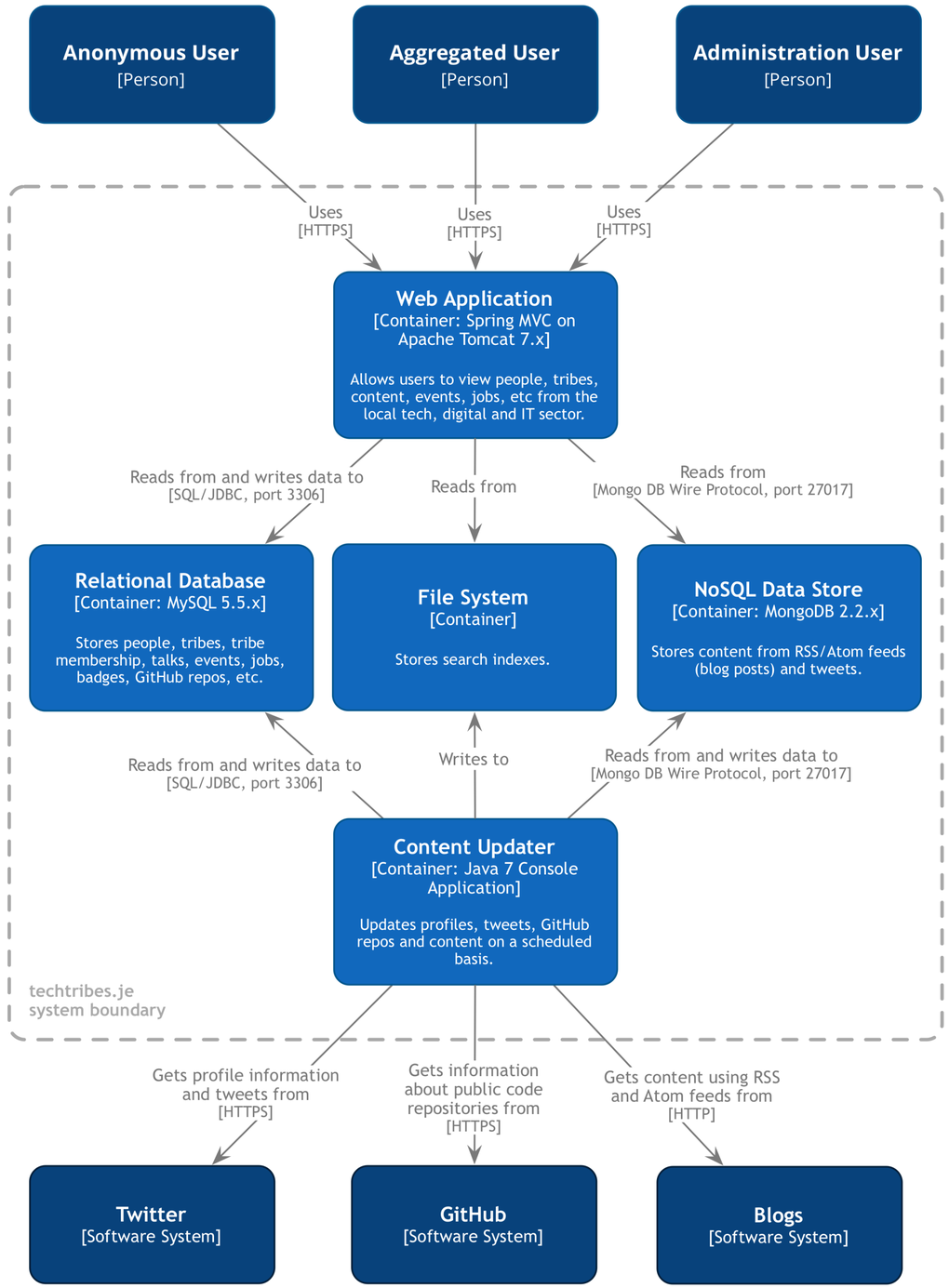
Responsibilities

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

Think about notation



[Containers] techtribes.je



[Containers] techtribes.je

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.

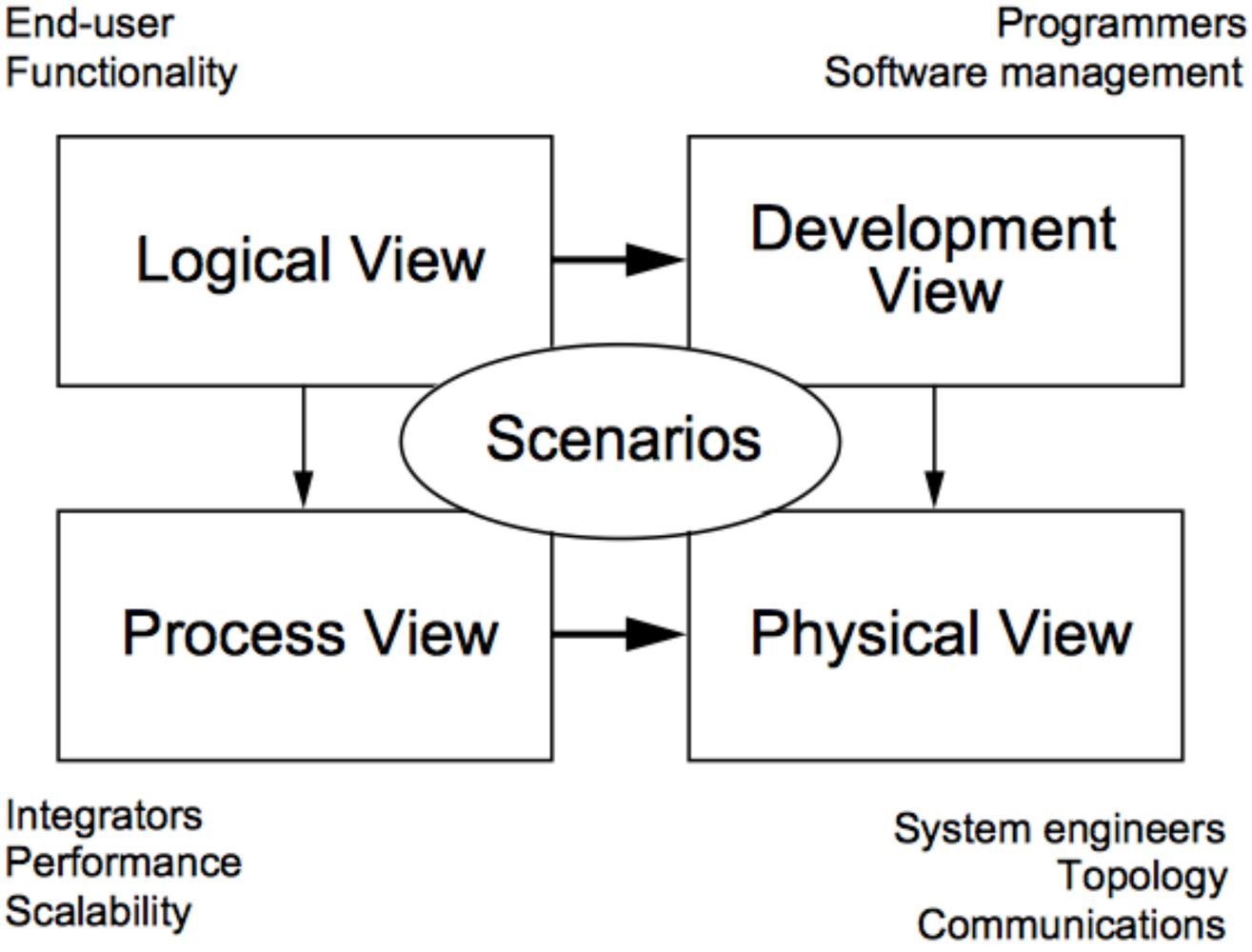
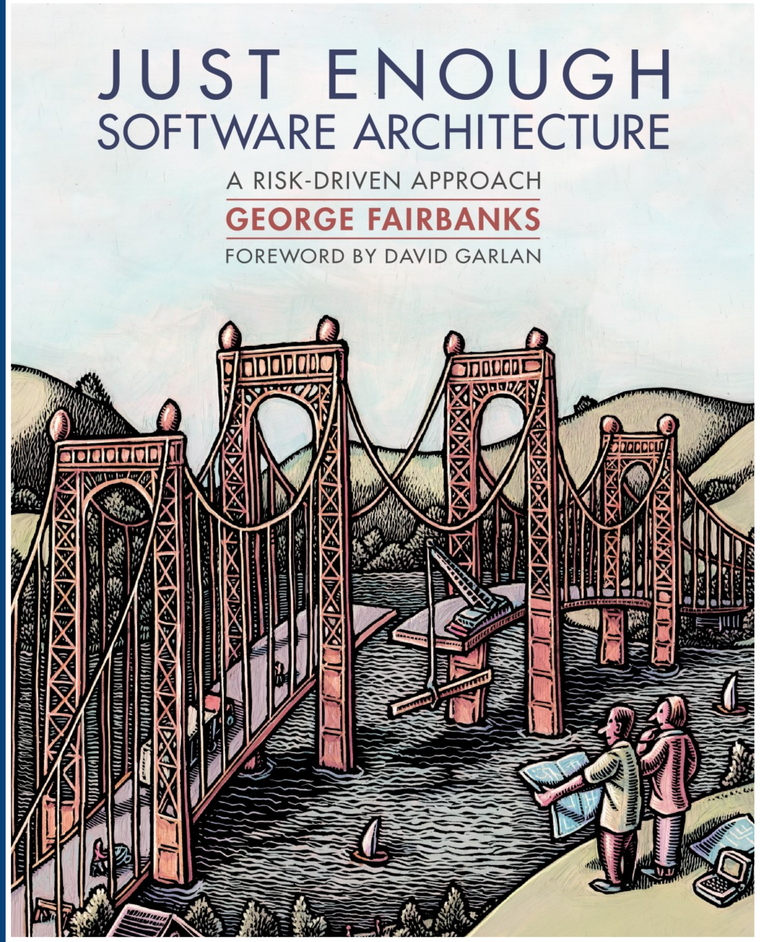


Figure 1 — The "4+1" view model

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

“Our architecture diagrams don’t match the code.”



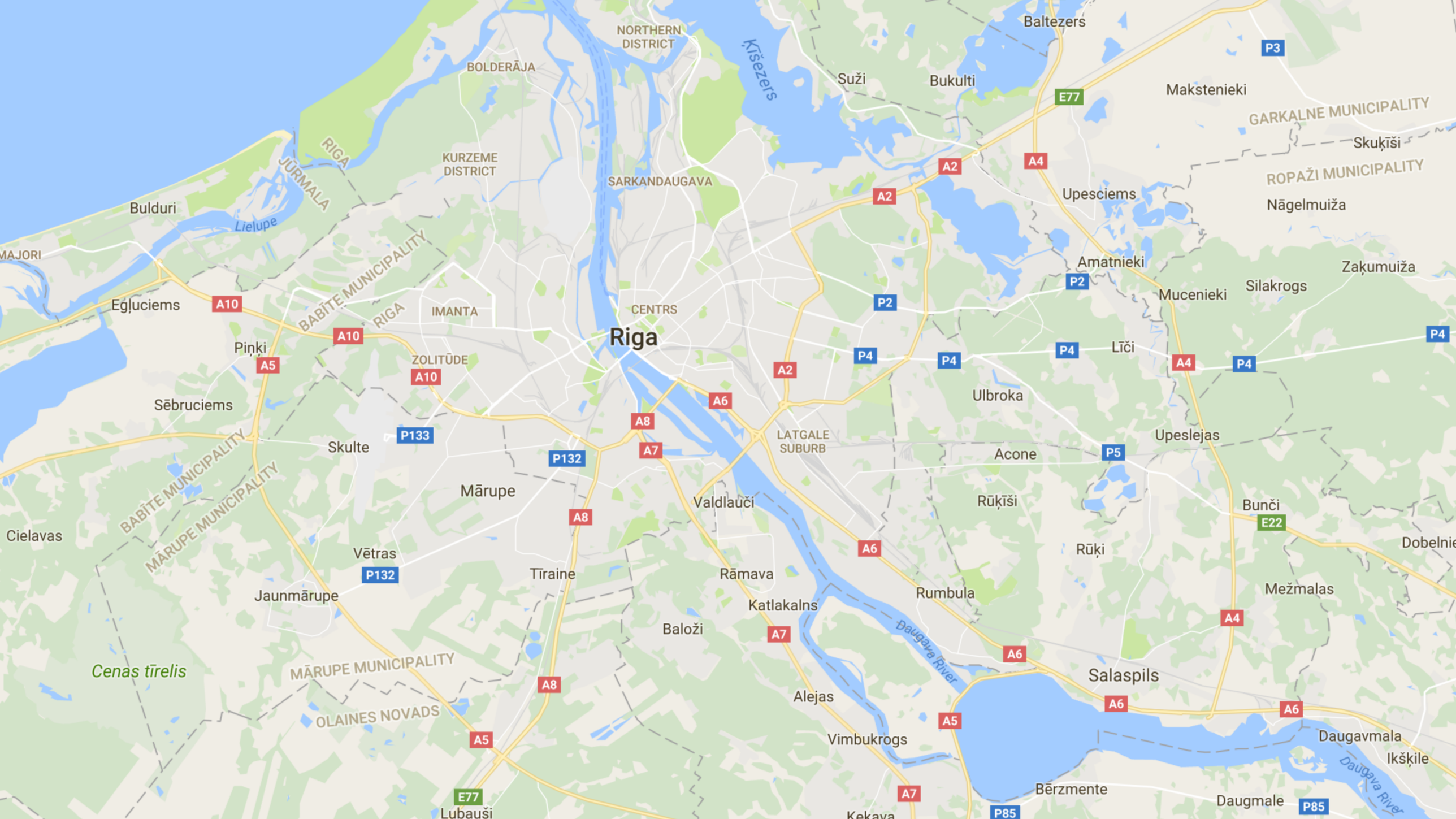
Model-code gap. Your architecture models and your source code will not show the same things. The difference between them is the *model-code gap*. Your architecture models include some abstract concepts, like components, that your programming language does not, but could. Beyond that, architecture models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

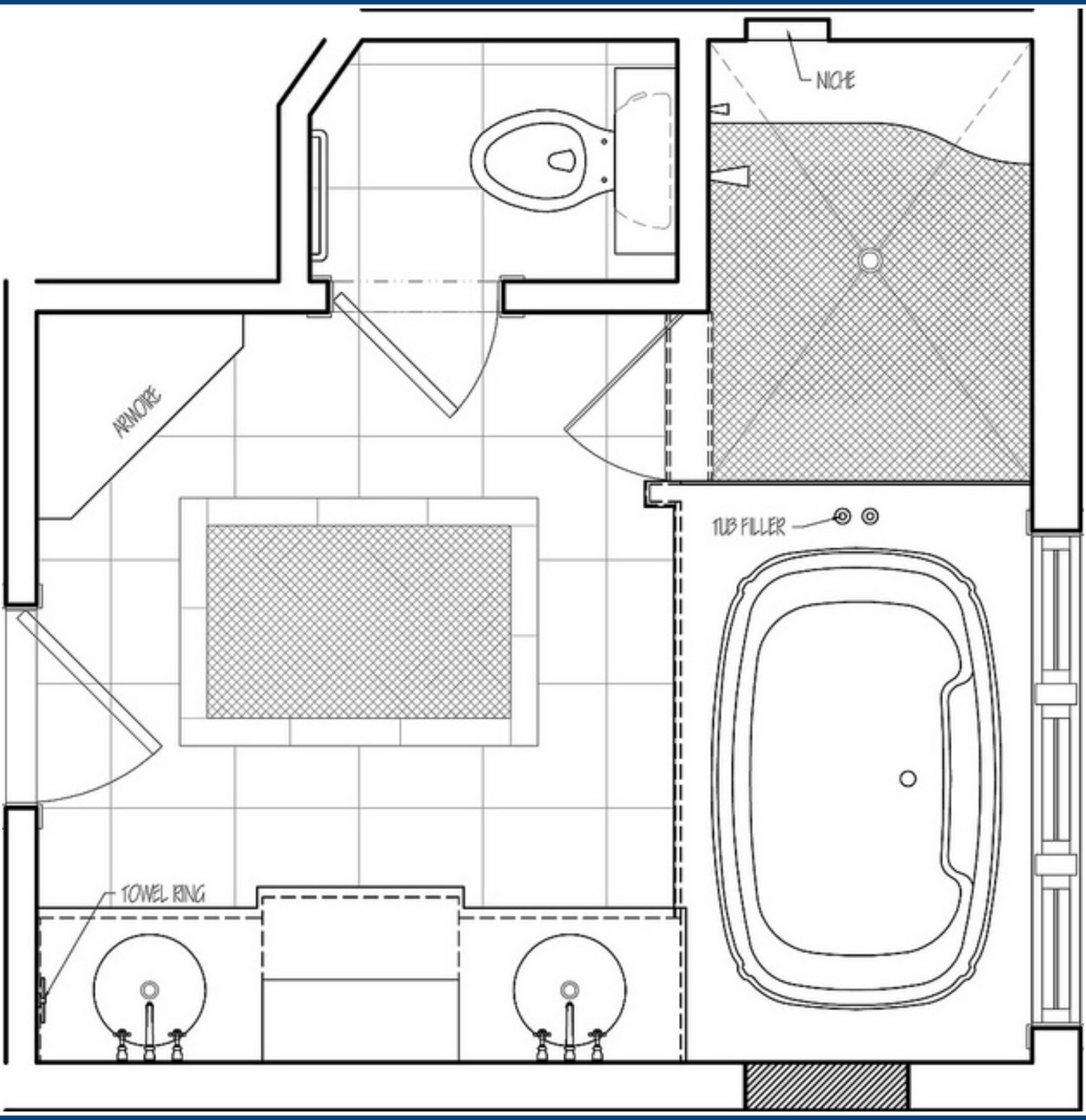
Consequently, the relationship between the architecture model and source code is complicated. It is mostly a refinement relationship, where the extensional elements in the architecture model are refined into extensional elements in source code. This is shown in Figure 10.3. However, intensional elements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

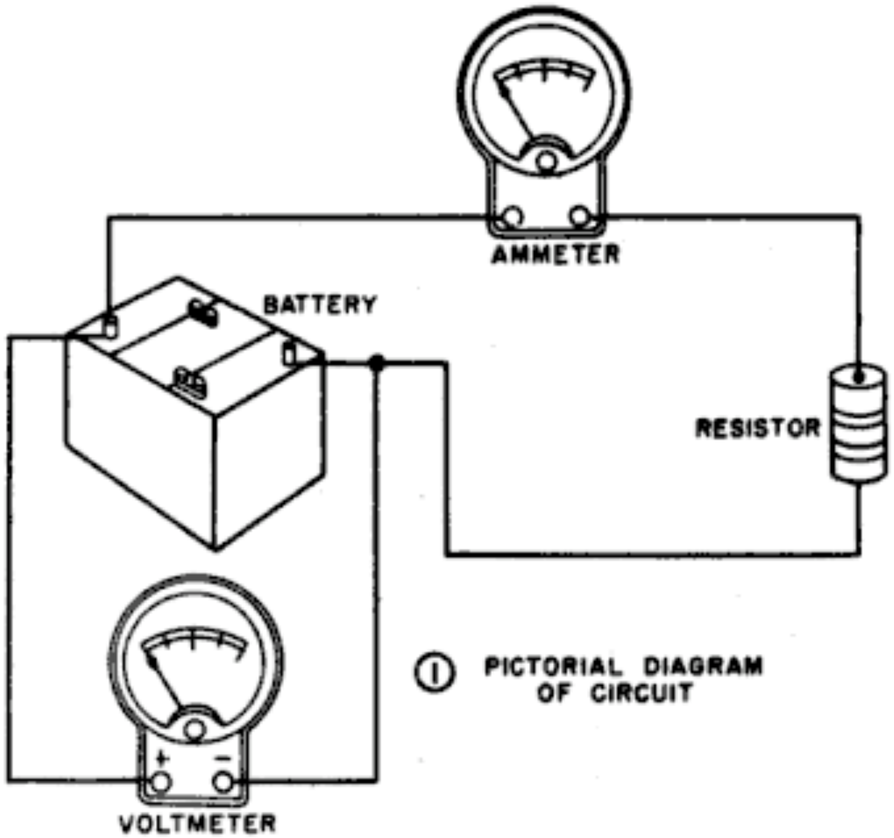
Upon learning about the model-code gap, your first instinct may be to avoid it. But reflecting on the origins of the gap gives little hope of a general solution in the short term: architecture models help you reason about complexity and scale because they are abstract and intensional; source code executes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

“model-code gap”

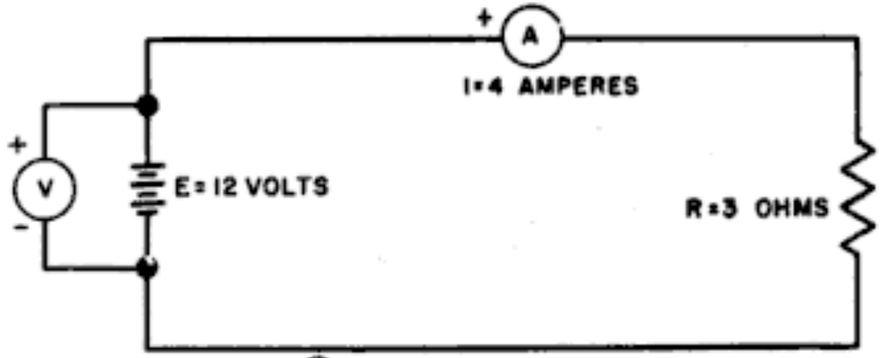
We lack a **common vocabulary** to describe software architecture


Riga
CENTRS
NORTHERN DISTRICT
BOLDERĀJA
KURZEME DISTRICT
SARKANDAUGAVA
Ķīšezers
Baltezers
Suži
Bukulti
Makstenieki
GARKALNE MUNICIPALITY
Skuķīši
ROPAŽI MUNICIPALITY
Upesciems
Nāgelmuiža
Amatnieki
Zaķumuiža
Mucenieki
Silakrogs
Līči
Ulbroka
Upeslejas
Acone
Rūķīši
Bunči
Rūķi
Dobelnie
Mežmalas
Salaspils
Daugavmala
Ikšķile
Daugmale
Bērzmente
Kekava
Lubauši
Vimbukrogs
Alejas
Rumbula
Katlakalns
Rāmava
Baloži
Tīraine
Valdlauči
LATGALE SUBURB
Daugava River
Māmuve
Skulte
Vētras
Jaunmārupe
Cielavas
Sēbruciems
Piņķi
ZOLITŪDE
IMANTA
Egļuciems
Bulduri
Lielupe
MAJORI
JŪRMALA
RIGA
BABĪTE MUNICIPALITY
MĀRUPE MUNICIPALITY
OLAINES NOVADS
Cenas tīrelis
A2
A4
A5
A6
A7
A8
A10
E77
E22
P2
P3
P4
P5
P85
P132
P133


NICHE
ARMOIRE
TUB FILLER
TOWEL RING

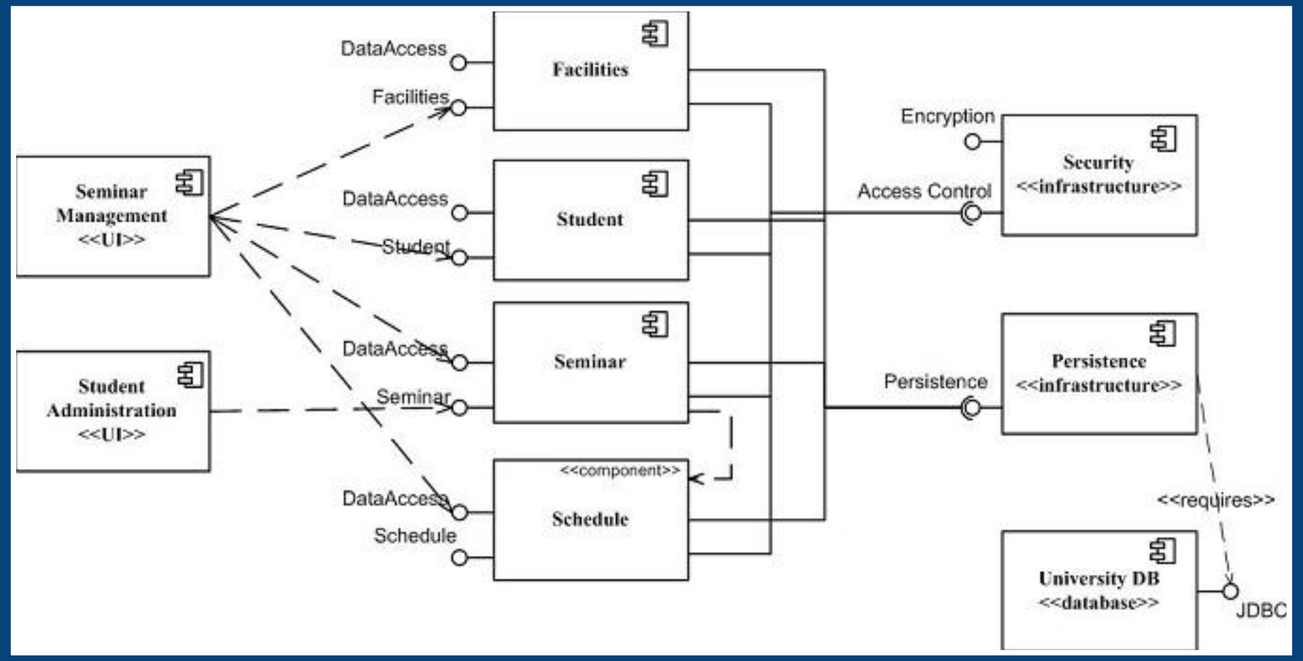


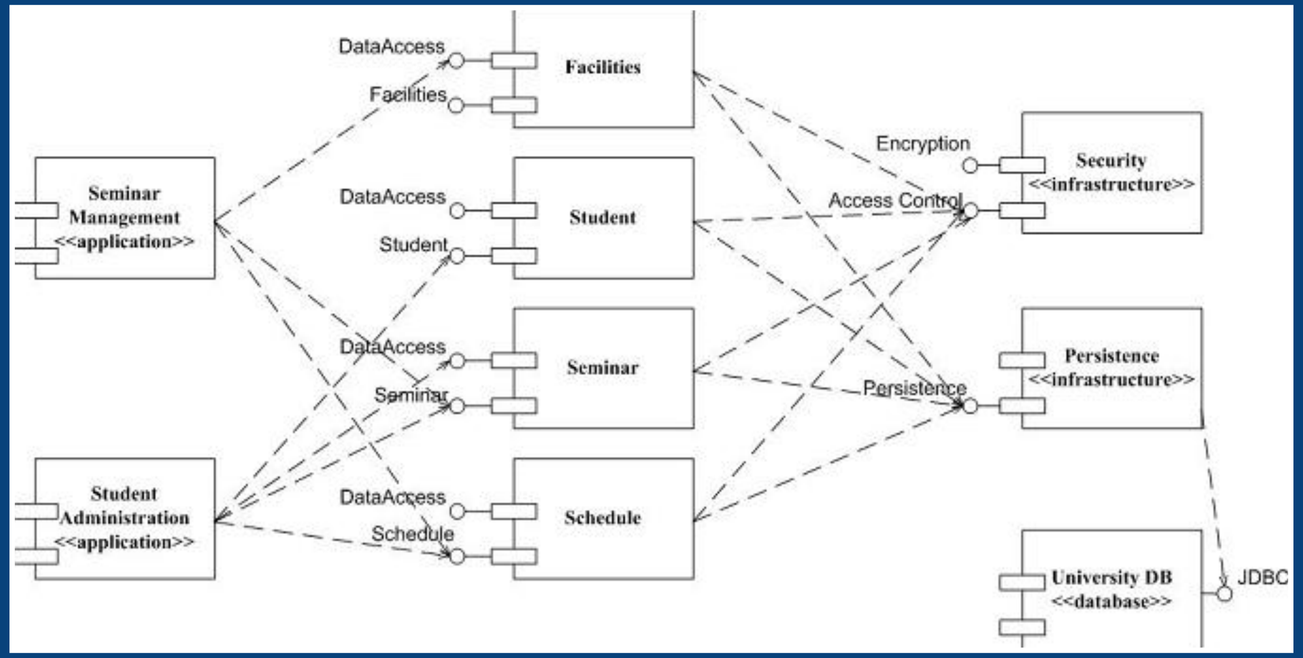
① PICTORIAL DIAGRAM OF CIRCUIT

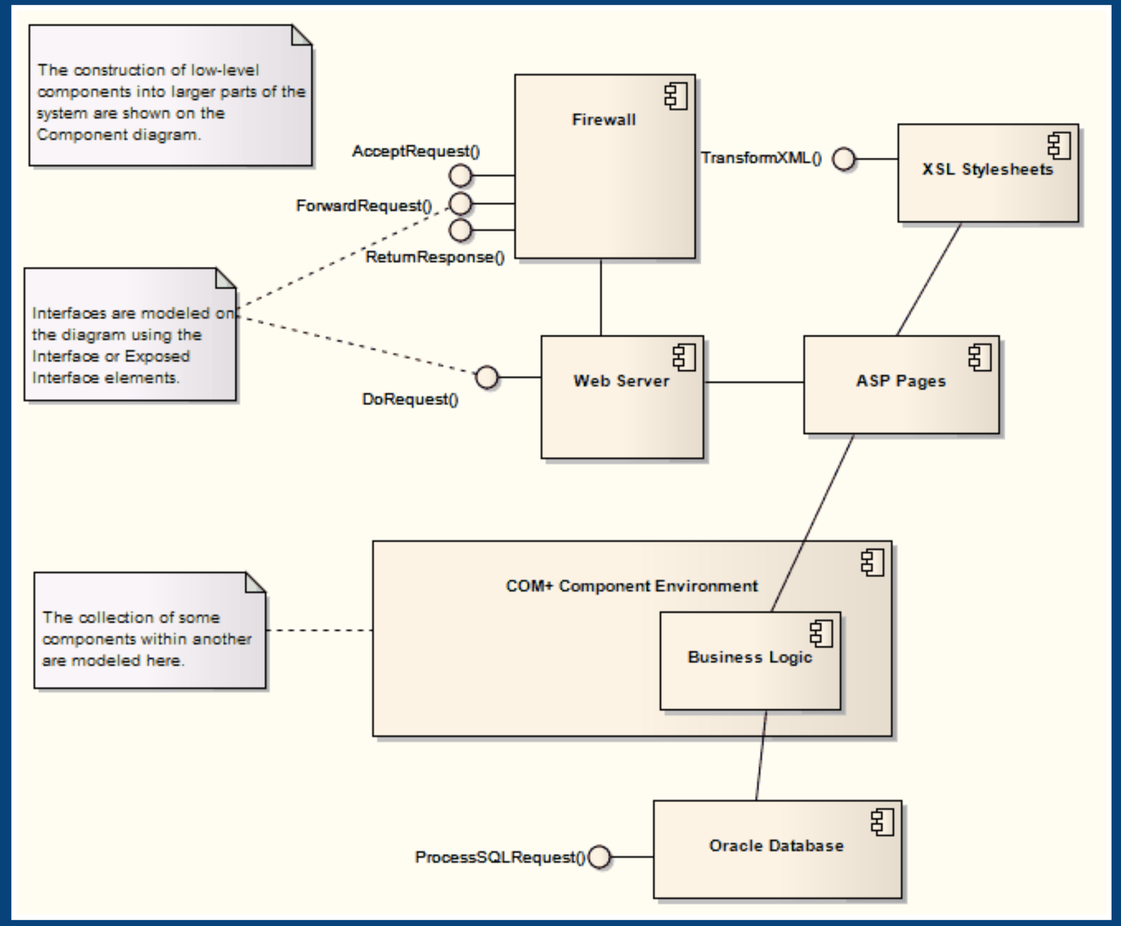


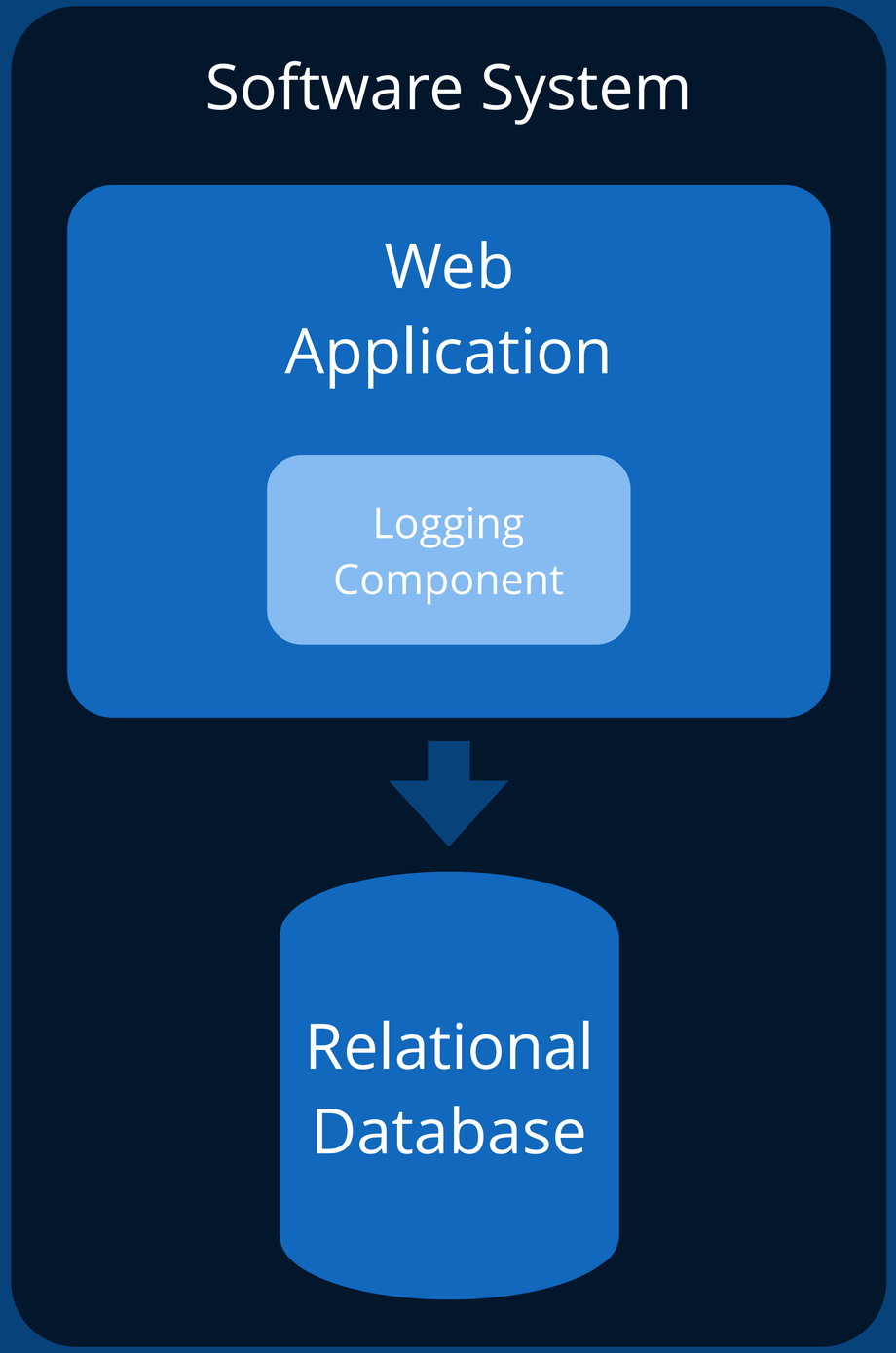
② SCHEMATIC OF CIRCUIT

Figure 48. Diagram of a basic circuit.


DataAccess
Facilities
Facilities
Seminar Management <<UI>>
DataAccess
Student
Student
Student Administration <<UI>>
DataAccess
Seminar
Seminar
DataAccess
Schedule
Schedule
<<component>>
Encryption
Access Control
Security <<infrastructure>>
Persistence
Persistence <<infrastructure>>
<<requires>>
University DB <<database>>
JDBC


DataAccess
Facilities
Facilities
Seminar Management <<application>>
DataAccess
Student
Student
DataAccess
Seminar
Seminar
Student Administration <<application>>
DataAccess
Schedule
Schedule
Encryption
Access Control
Security <<infrastructure>>
Persistence
Persistence <<infrastructure>>
University DB <<database>>
JDBC


The construction of low-level components into larger parts of the system are shown on the Component diagram.
Firewall
AcceptRequest()
ForwardRequest()
ReturnResponse()
TransformXML()
XSL Stylesheets
Interfaces are modeled on the diagram using the Interface or Exposed Interface elements.
DoRequest()
Web Server
ASP Pages
The collection of some components within another are modeled here.
COM+ Component Environment
Business Logic
ProcessSQLRequest()
Oracle Database



1 component

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

Simple Definition of COMPONENT

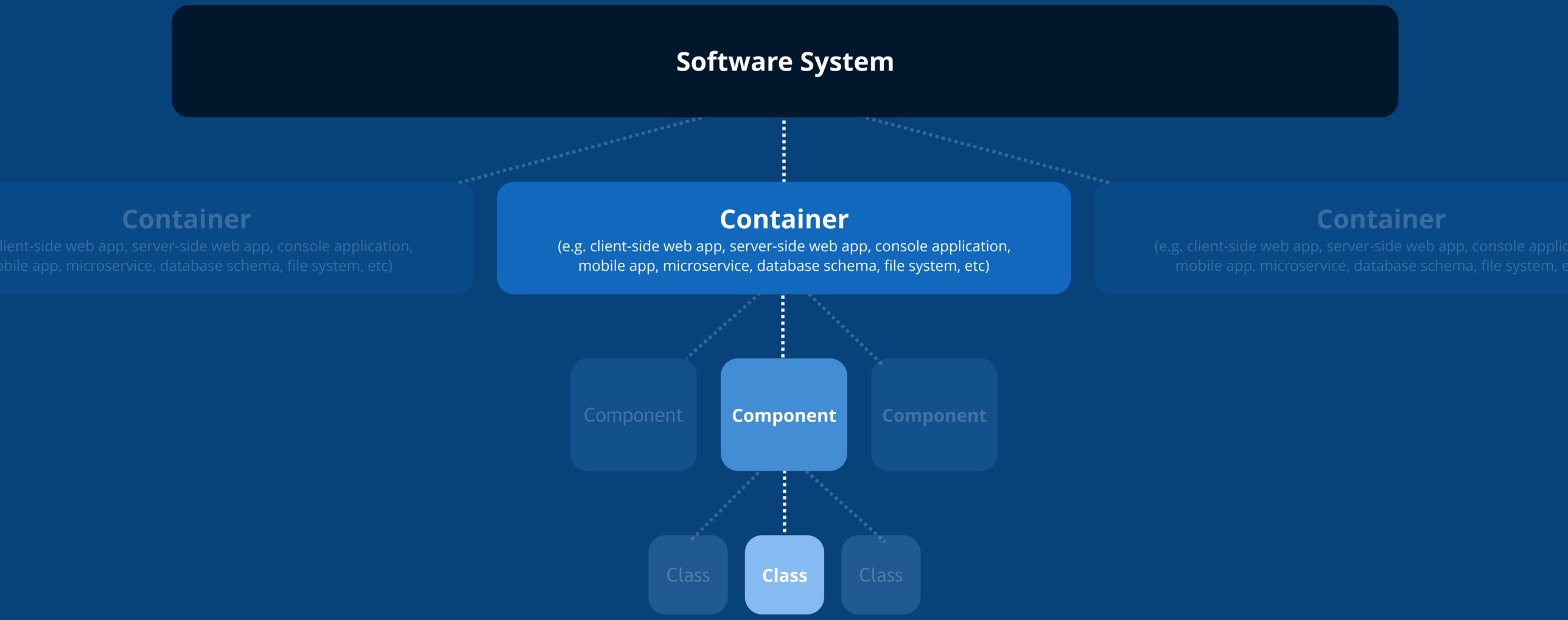
Popularity: Top 30% of words

: 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

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 **classes** (or **code**).

C4

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. Classes (or Code)

Component implementation details.

Overview first

Zoom & filter

Details on demand

techtribes.je

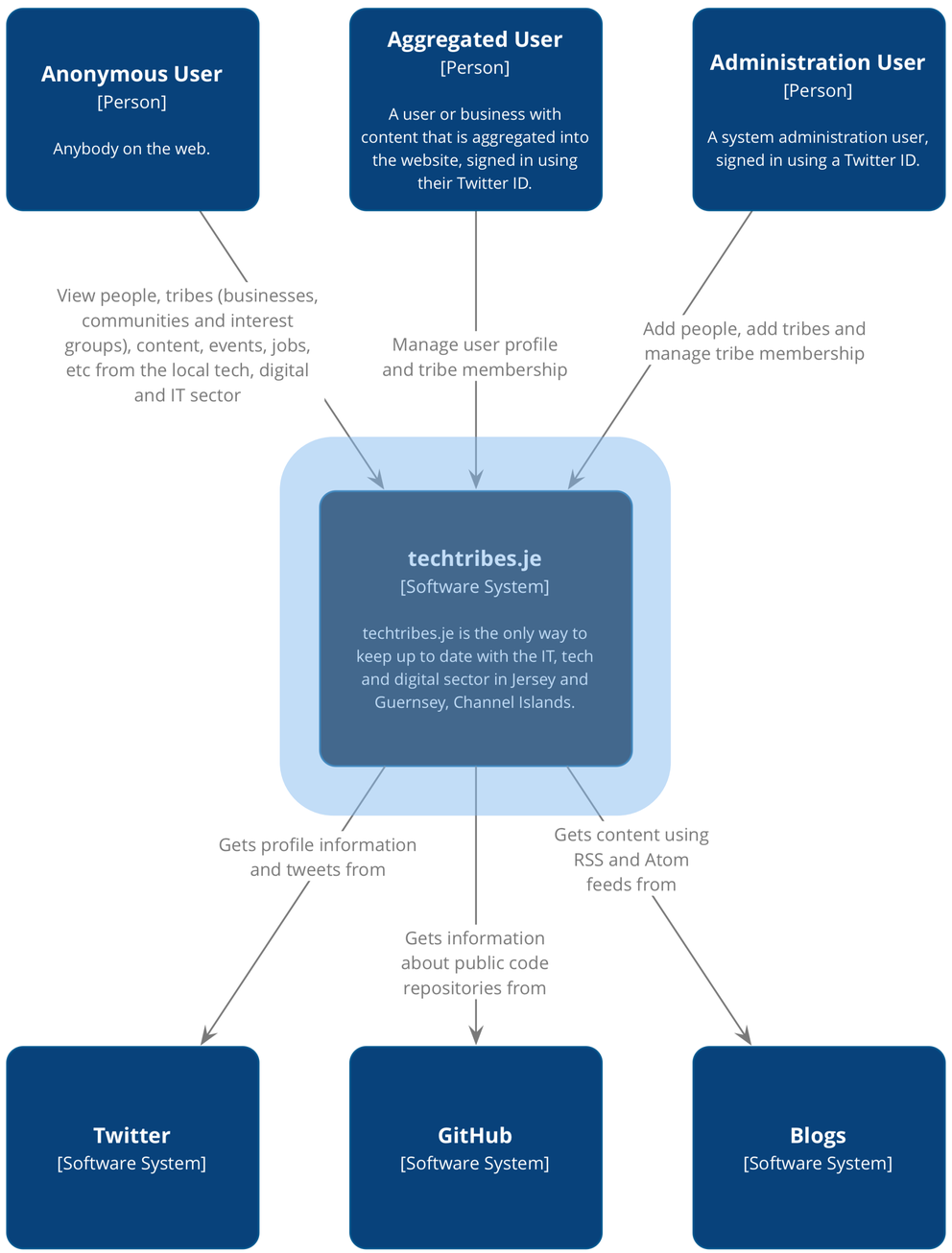
A simple content aggregator for the local tech and digital industry

1. System Context diagram

2. Container diagram

3. Component diagram

4. Class diagram



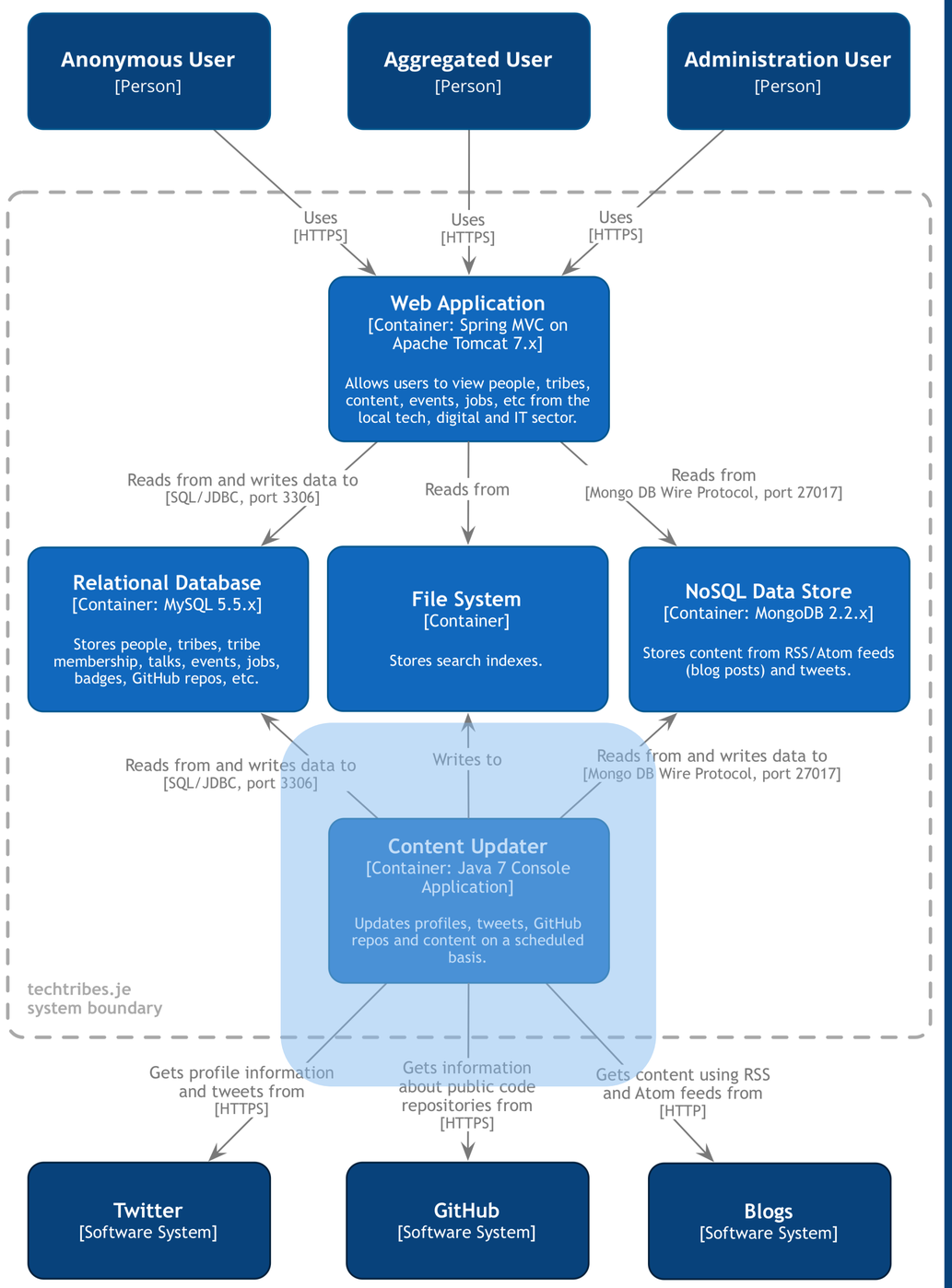
[System Context] techtribes.je

1. System Context diagram

2. Container diagram

3. Component diagram

4. Class diagram



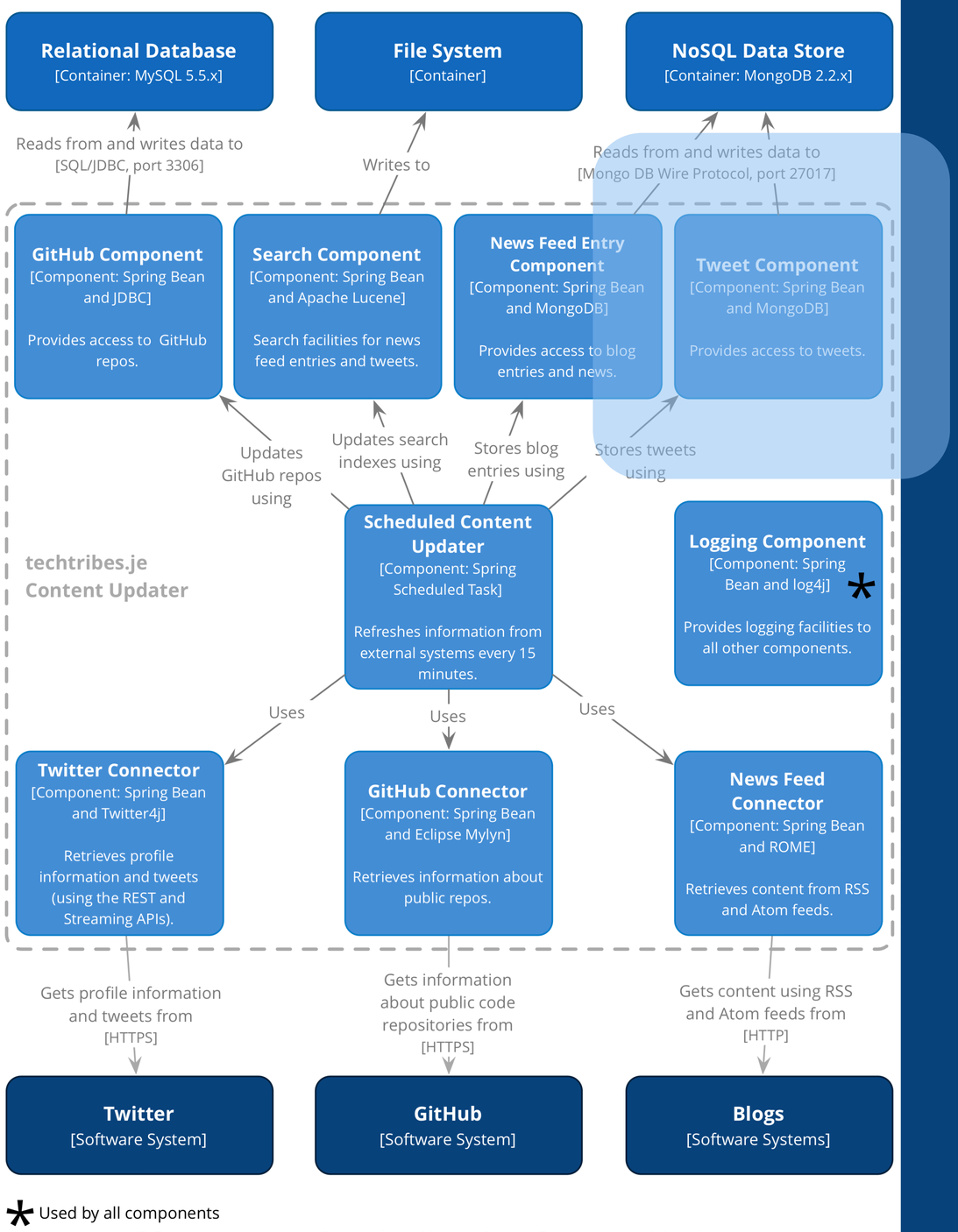
[Containers] techtribes.je

1. System Context diagram

2. Container diagram

3. Component diagram

4. Class diagram

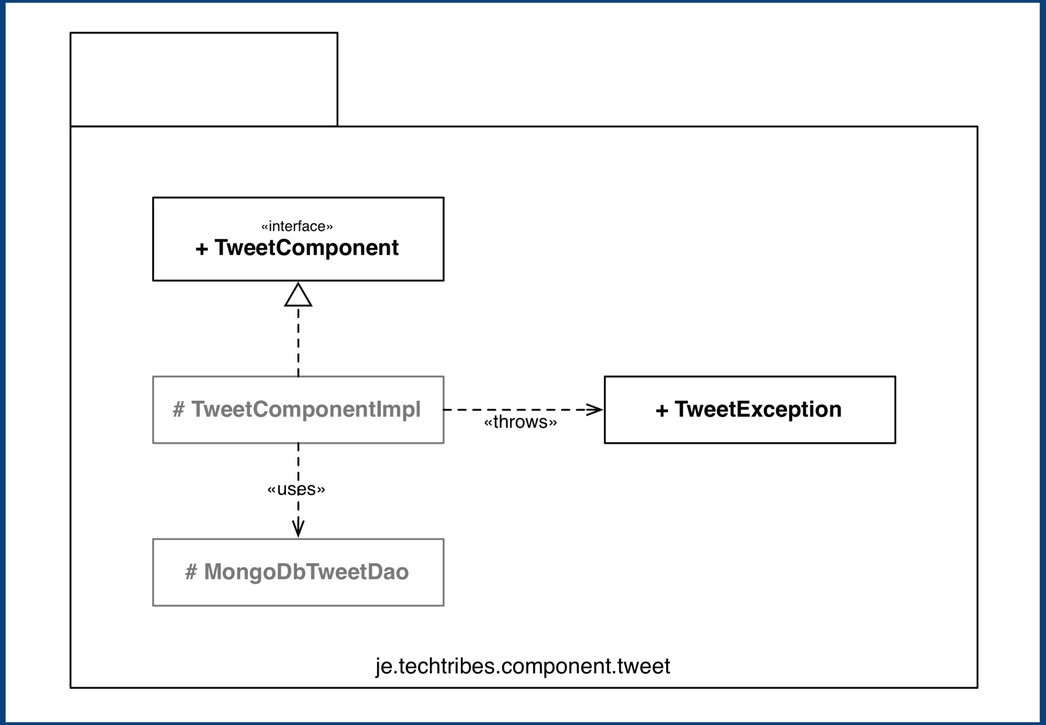


[Components] techtribes.je - Content Updater

1. System Context diagram

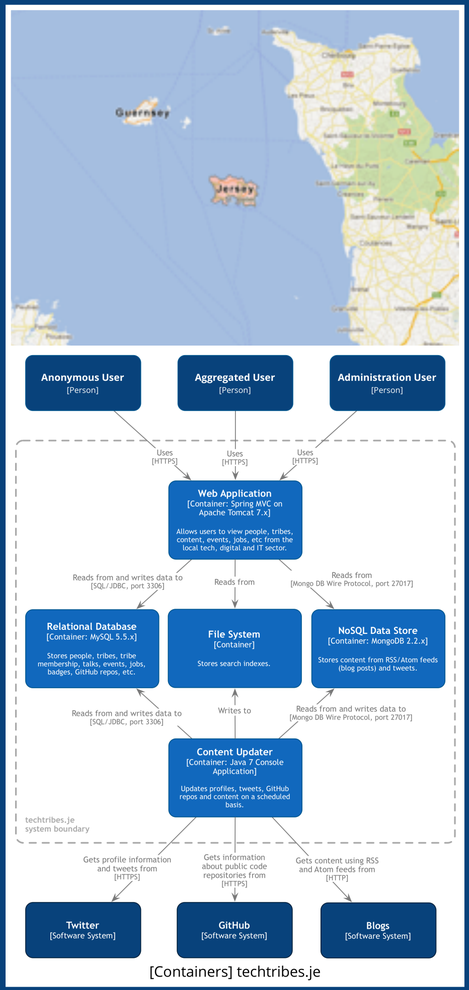
2. Container diagram

3. Component diagram

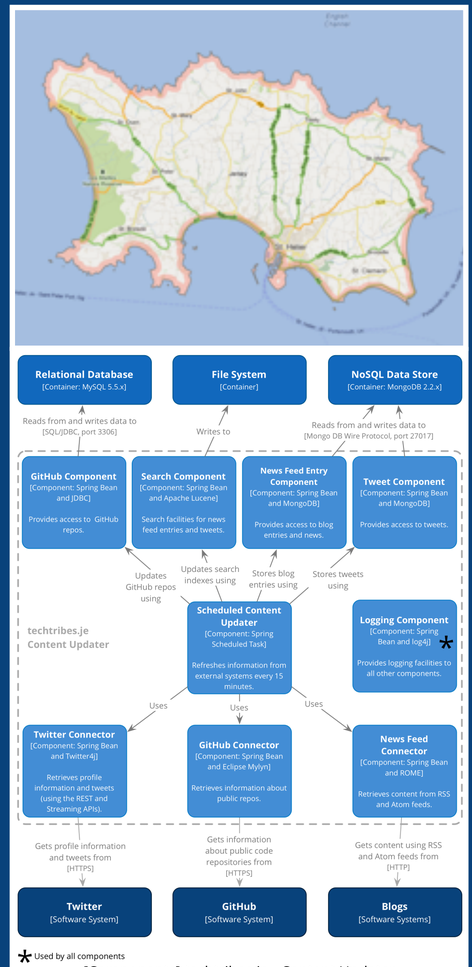


4. Class diagram

[System Context] techtribes.je



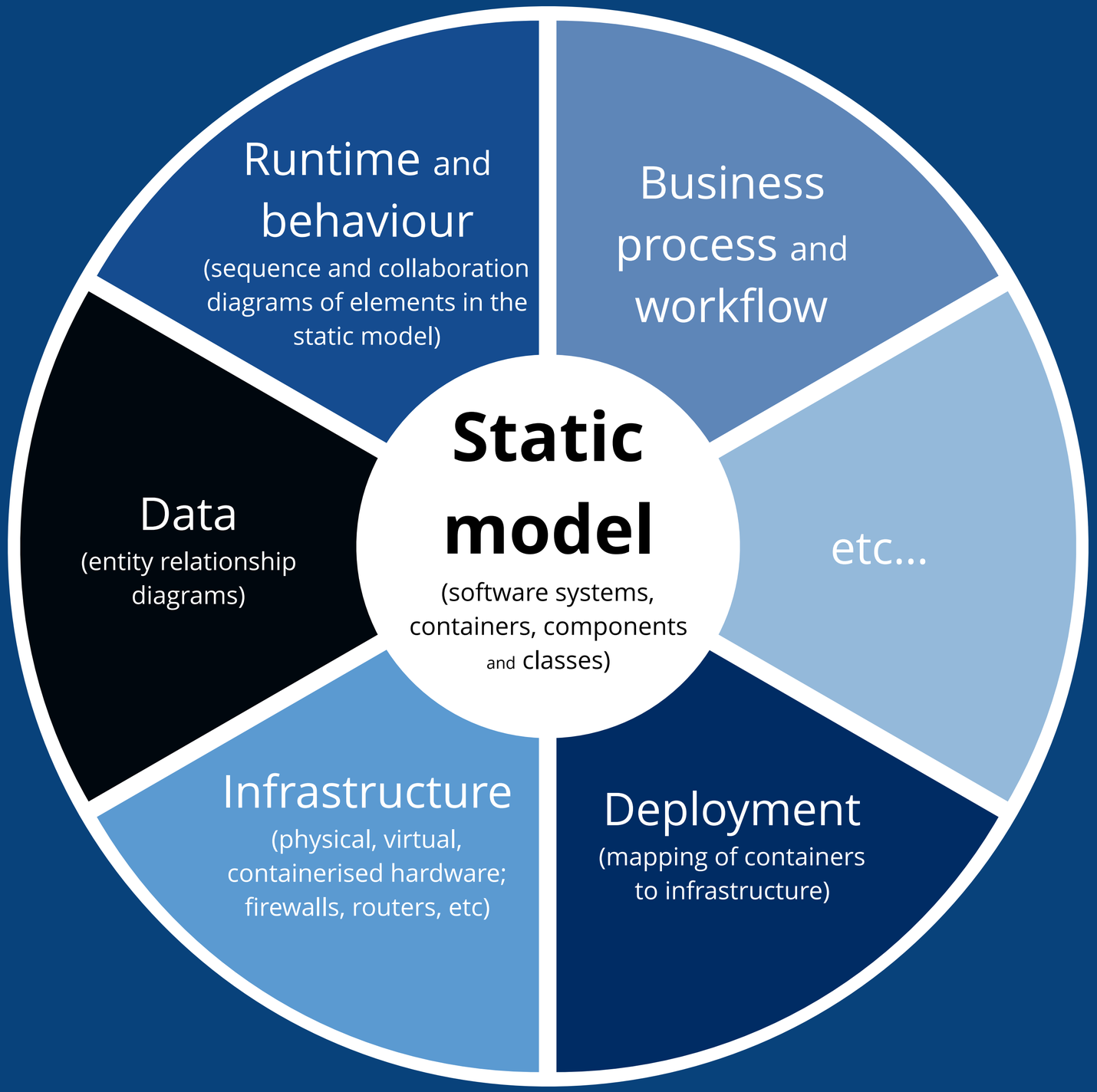
[Containers] techtribes.je



[Components] techtribes.je - Content Updater

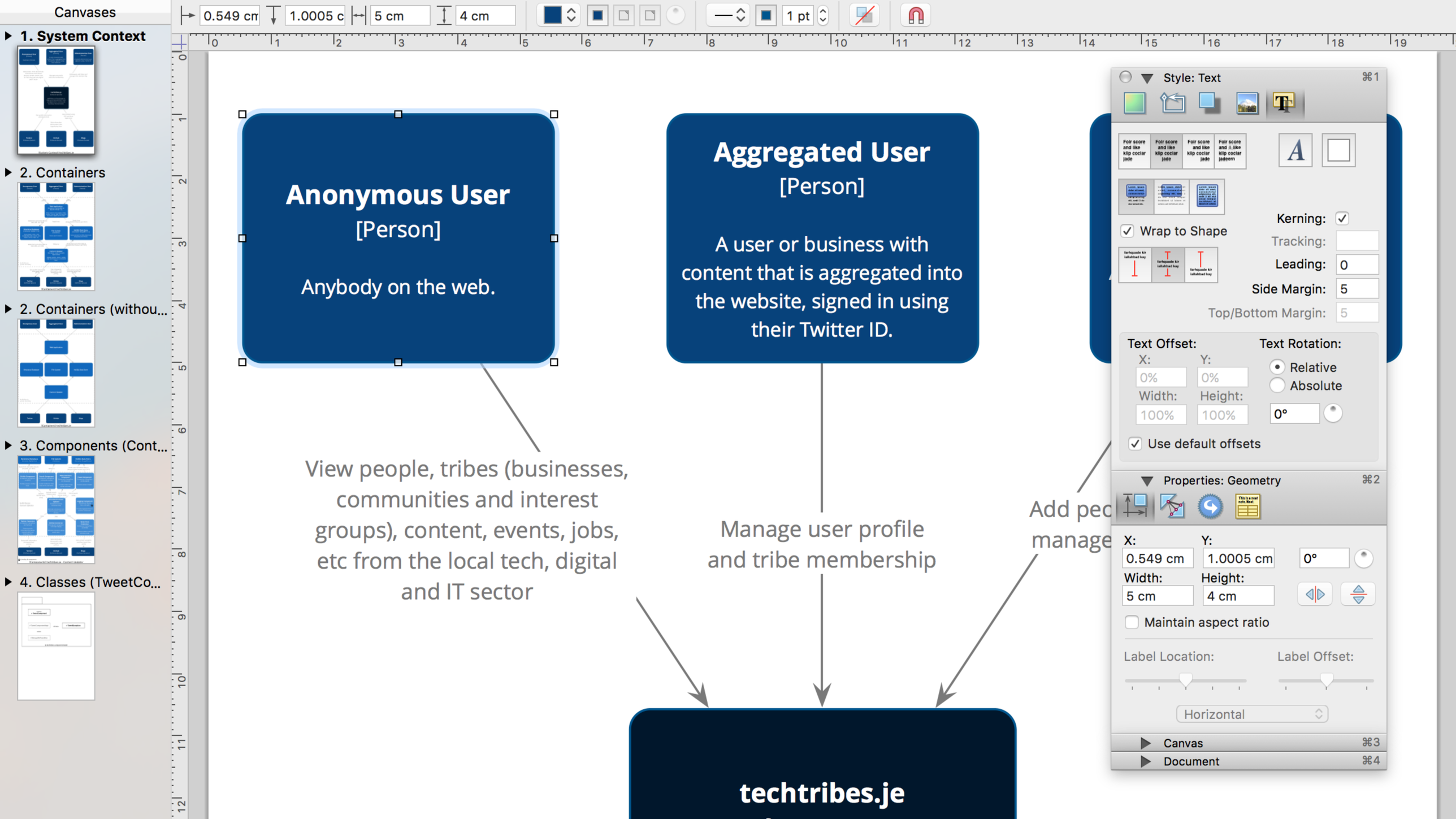
Diagrams are maps

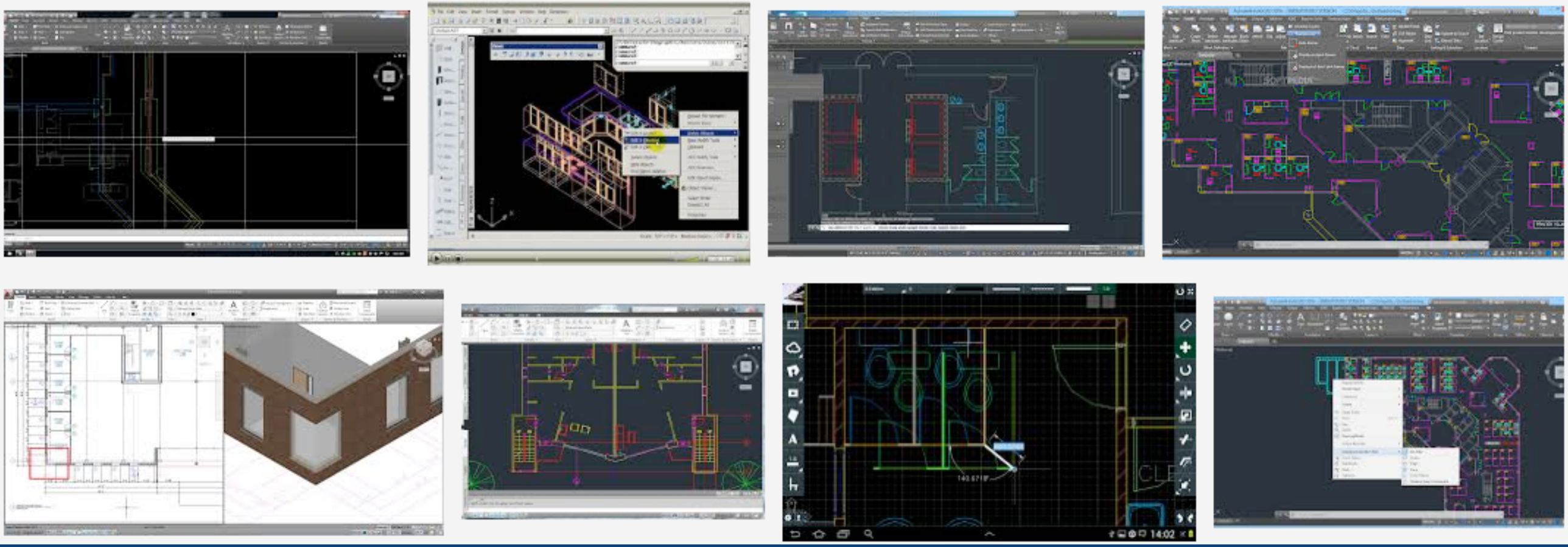
that help software developers navigate a large and/or complex codebase



A model of the **static structure** forms the basis for other views

What **tools** do you recommend?


Canvases
1. System Context
2. Containers
2. Containers (withou...
3. Components (Cont...
4. Classes (TweetCo...
0.549 cm
1.0005 c
5 cm
4 cm
1 pt
Anonymous User
[Person]
Anybody on the web.
Aggregated User
[Person]
A user or business with content that is aggregated into the website, signed in using their Twitter ID.
View people, tribes (businesses, communities and interest groups), content, events, jobs, etc from the local tech, digital and IT sector
Manage user profile and tribe membership
Add pec
manage
techtribes.je
Style: Text
Wrap to Shape
Kerning:
Tracking:
Leading: 0
Side Margin: 5
Top/Bottom Margin: 5
Text Offset:
X: 0%
Y: 0%
Width: 100%
Height: 100%
Text Rotation:
Relative
Absolute
0°
Use default offsets
Properties: Geometry
X: 0.549 cm
Y: 1.0005 cm
0°
Width: 5 cm
Height: 4 cm
Maintain aspect ratio
Label Location:
Label Offset:
Horizontal
Canvas
Document



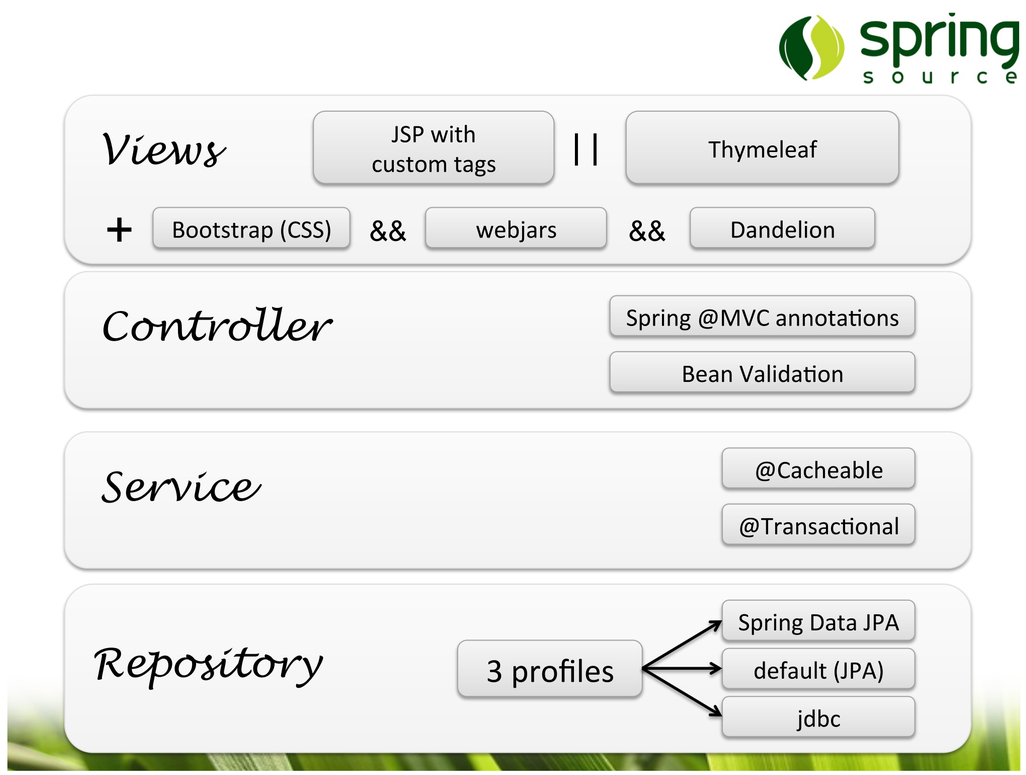
Do structural engineers and building architects use general purpose drawing tools?

Reverse-engineer
code to diagrams?

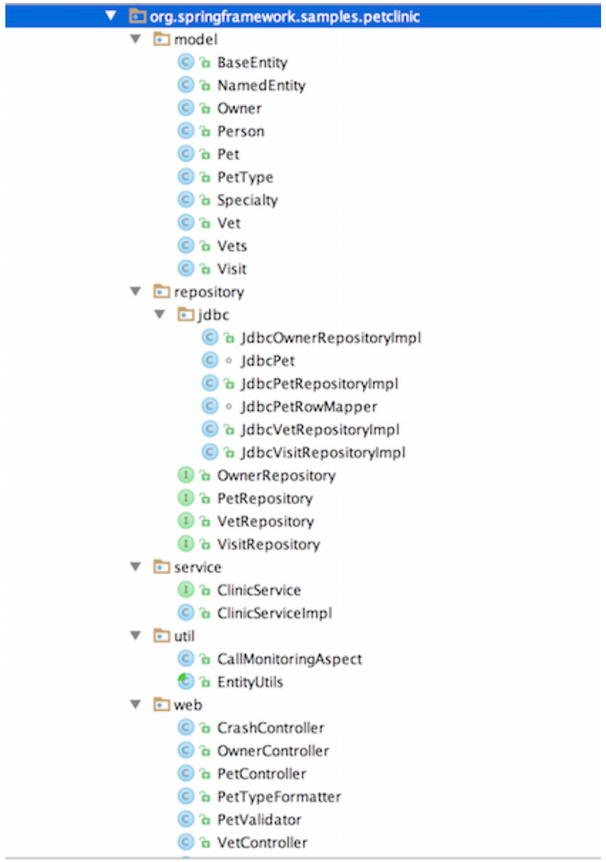
Spring PetClinic

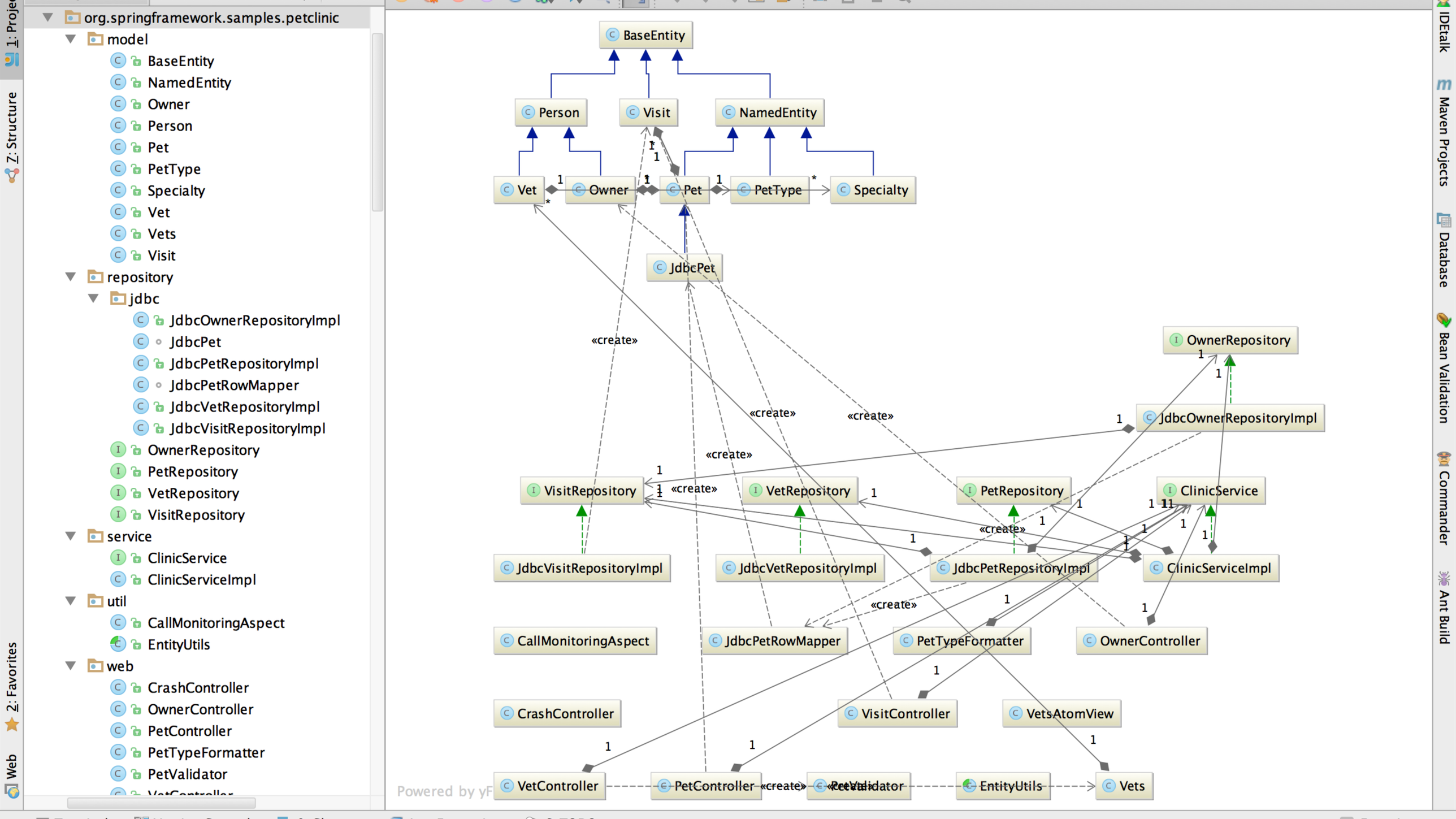
A sample application that illustrates how to build Java web applications using the Spring MVC framework

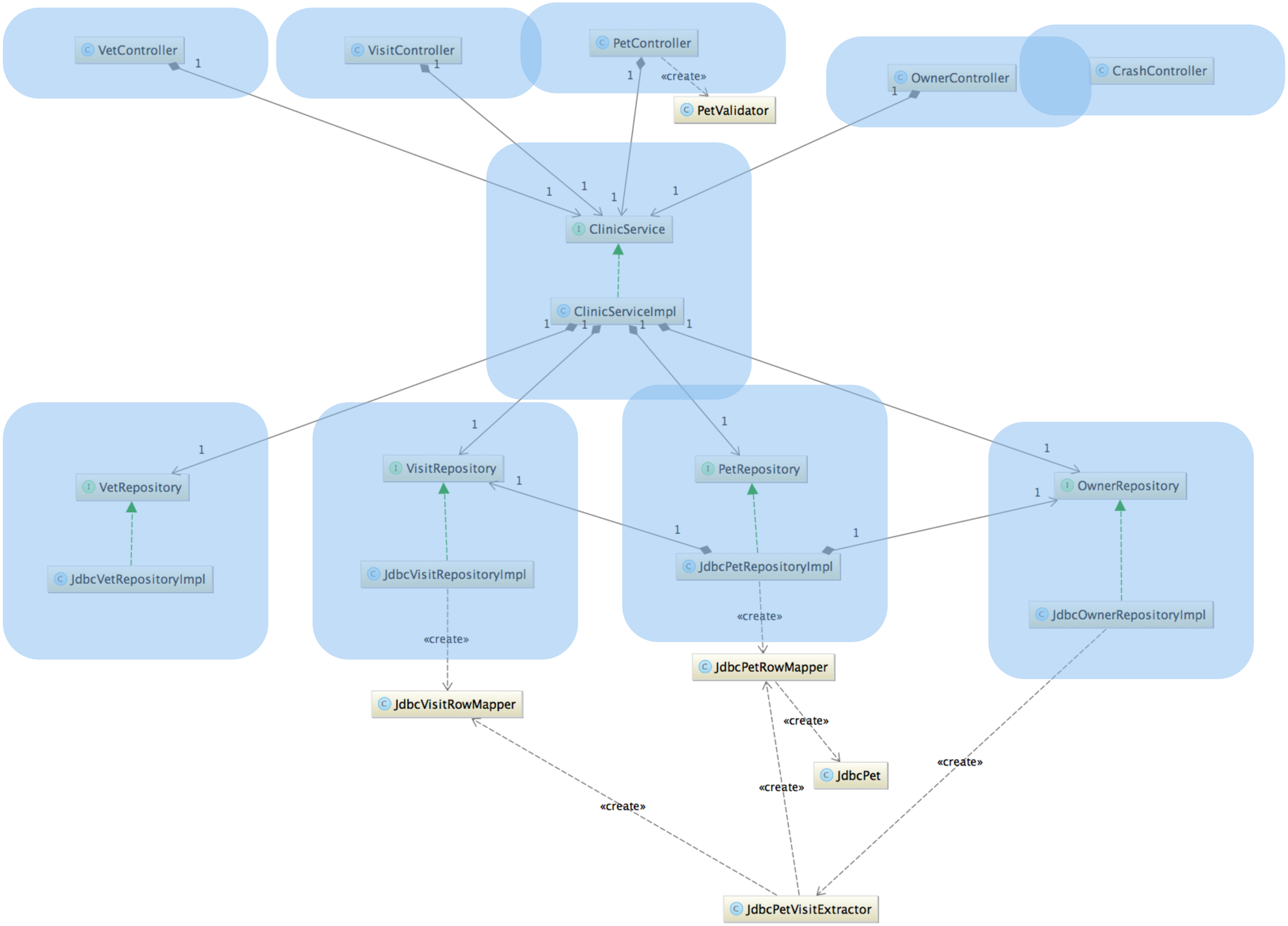
https://github.com/spring-projects/spring-petclinic/



https://speakerdeck.com/michaelisvy/spring-petclinic-sample-application




org.springframework.samples.petclinic
model
BaseEntity
NamedEntity
Owner
Person
Pet
PetType
Specialty
Vet
Vets
Visit
repository
jdbc
JdbcOwnerRepositoryImpl
JdbcPet
JdbcPetRepositoryImpl
JdbcPetRowMapper
JdbcVetRepositoryImpl
JdbcVisitRepositoryImpl
OwnerRepository
PetRepository
VetRepository
VisitRepository
service
ClinicService
ClinicServiceImpl
util
CallMonitoringAspect
EntityUtils
web
CrashController
OwnerController
PetController
PetTypeFormatter
PetValidator
BaseEntity
Person
Visit
NamedEntity
Vet
Owner
Pet
PetType
Specialty
JdbcPet
«create»
OwnerRepository
JdbcOwnerRepositoryImpl
VisitRepository
VetRepository
PetRepository
ClinicService
JdbcVisitRepositoryImpl
JdbcVetRepositoryImpl
JdbcPetRepositoryImpl
ClinicServiceImpl
CallMonitoringAspect
JdbcPetRowMapper
PetTypeFormatter
OwnerController
CrashController
VisitController
VetsAtomView
VetController
PetController
EntityUtils
Vets
Powered by yF


VetController
VisitController
PetController
«create»
PetValidator
OwnerController
CrashController
ClinicService
ClinicServiceImpl
VetRepository
JdbcVetRepositoryImpl
VisitRepository
JdbcVisitRepositoryImpl
PetRepository
JdbcPetRepositoryImpl
OwnerRepository
JdbcOwnerRepositoryImpl
JdbcPetRowMapper
JdbcVisitRowMapper
JdbcPet
JdbcPetVisitExtractor

Most tools see code,
not **components**

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

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

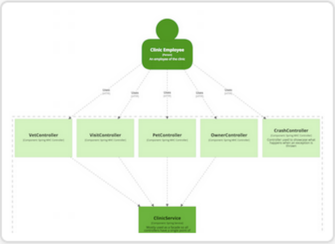
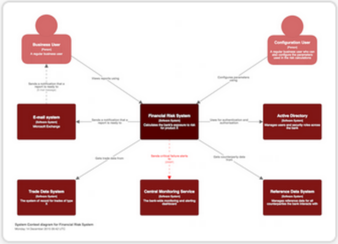
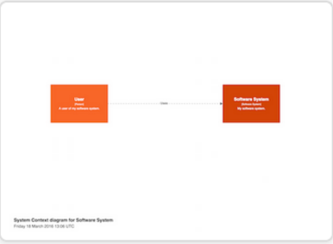
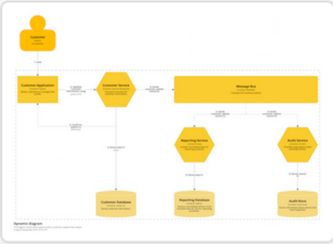
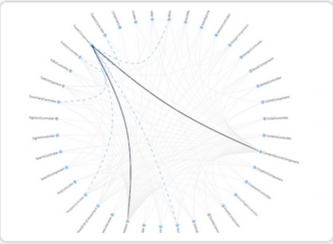
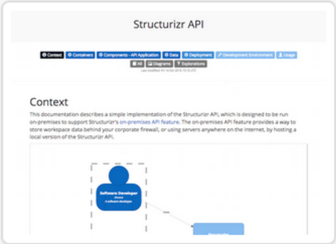
Software engineers often think about an existing software system in terms of high-level models. Box and arrow sketches of a system, for instance, are often found on engineers' whiteboards. Although these models are commonly used, reasoning about the system in terms of such models can be dangerous because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

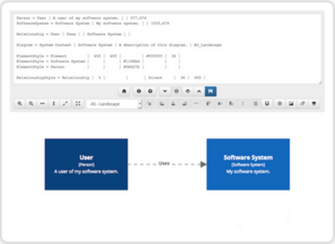
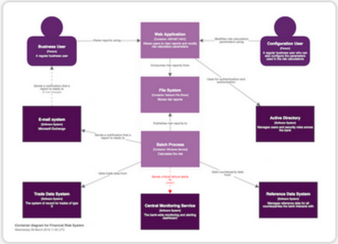
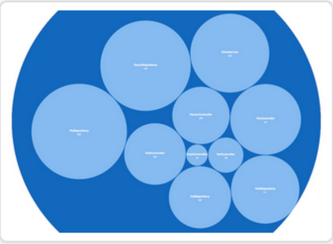
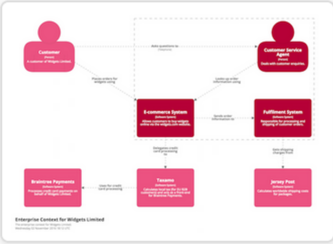
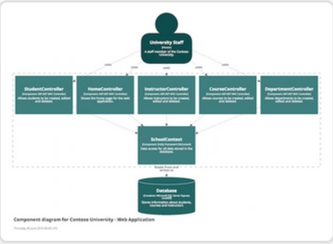
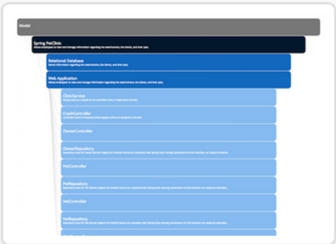
Current reverse engineering systems derive high-level models from the source code. These derived models are useful because they are, by their very nature, accurate representations of the source. Although accurate, the models created by these reverse engineering systems may differ from the models sketched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

Information about software architecture **doesn’t exist in the code**

Structurizr

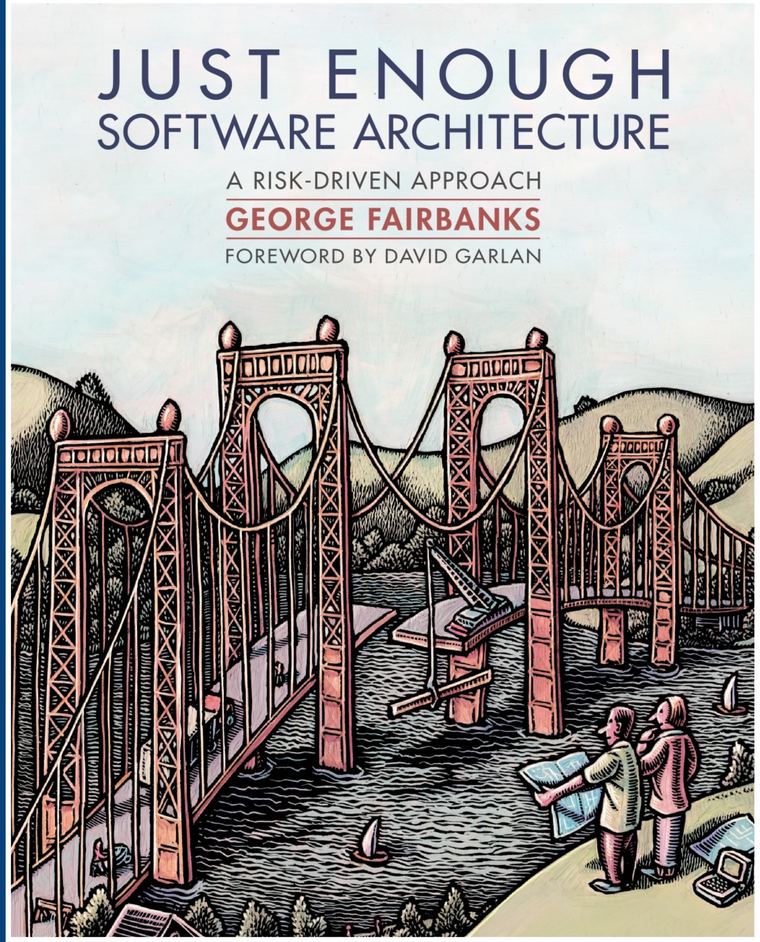
Visualise, document and explore your software architecture

```
Person = User | A user of my software system. | | 277,674
SoftwareSystem = Software System | My software system. | | 1552,674

Relationship = User | Uses | | Software System | |

Diagram = System Context | Software System | A description of this diagram. | A5_Landscape

ElementStyle = Element          |  650 |  400 |         | #ffffff |  36 |
ElementStyle = Software System |      |      | #1168bd |         |     |
ElementStyle = Person          |      |      | #08427b |         |     |
```

A5 - Landscape



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.

“architecturally-evident coding style”

Examples of architecturally-evident coding styles

Annotations/attributes (@Component, [Component], etc)

Naming conventions (*Controller, *Service, etc)

Namespacing/packaging (com.mycompany.system.components.*)

Maven & Gradle modules, OSGi & Java 9 modules

JavaScript module patterns, ECMAScript 6 modules, microservices, etc

Executable architecture description language

Structurizr for Java and .NET

```
public static void main(String[] args) throws Exception {
    Workspace workspace = new Workspace(
        "Spring PetClinic",
        "This is a C4 representation of the Spring PetClinic sample app
         (https://github.com/spring-projects/spring-petclinic/)");

    Model model = workspace.getModel();





}
```

```
// software systems and people
SoftwareSystem springPetClinic = model.addSoftwareSystem(
    "Spring PetClinic",
    "Allows employees to view and manage information regarding the
     veterinarians, the clients, and their pets.");

Person clinicEmployee = model.addPerson(
    "Clinic Employee", "An employee of the clinic");

clinicEmployee.uses(springPetClinic, "Uses");
```

```
// containers
Container webApplication = springPetClinic.addContainer(
    "Web Application",
    "Allows employees to view and manage information regarding the
     veterinarians, the clients, and their pets.",
    "Apache Tomcat 7.x");

Container relationalDatabase = springPetClinic.addContainer(
    "Relational Database",
    "Stores information regarding the veterinarians, the clients,
     and their pets.", "HSQLDB");

clinicEmployee.uses(webApplication,
    "Uses", “HTTP");

webApplication.uses(relationalDatabase,
    "Reads from and writes to", "JDBC, port 9001");
```

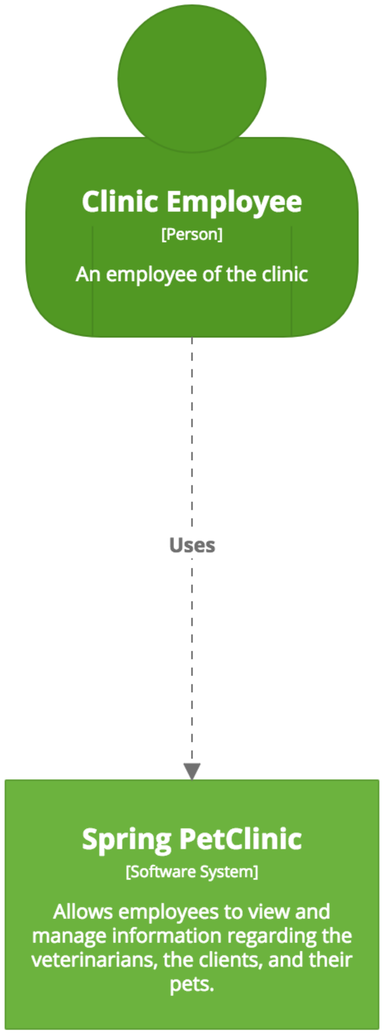
```java
// components
ComponentFinder componentFinder = new ComponentFinder(
    webApplication,
    "org.springframework.samples.petclinic",
    new SpringComponentFinderStrategy(
        new ReferencedTypesSupportingTypesStrategy()
    ),
    new SourceCodeComponentFinderStrategy(
        new File(sourceRoot, "/src/main/java/"), 150));

componentFinder.findComponents();
```

```
// connect components with other model elements
webApplication.getComponents().stream()
    .filter(c -> c.getTechnology().equals(SpringComponentFinderStrategy.SPRING_MVC_CONTROLLER))
    .forEach(c -> clinicEmployee.uses(c, "Uses", "HTTP"));

webApplication.getComponents().stream()
    .filter(c -> c.getTechnology().equals(SpringComponentFinderStrategy.SPRING_REPOSITORY))
    .forEach(c -> c.uses(relationalDatabase, "Reads from and writes to", "JDBC"));
```

```
// system context, container and component views
ViewSet viewSet = workspace.getViews();

SystemContextView contextView = viewSet.createContextView(
    springPetClinic, "context", "Context view for Spring PetClinic");
contextView.addAllSoftwareSystems();
contextView.addAllPeople();

ContainerView containerView = viewSet.createContainerView(
    springPetClinic, "containers", "Container view for Spring PetClinic");
containerView.addAllPeople();
containerView.addAllSoftwareSystems();
containerView.addAllContainers();

ComponentView componentView = viewSet.createComponentView(
    webApplication, "components", "Component view for the Spring PetClinic webapp.");
componentView.addAllComponents();
componentView.addAllPeople();
componentView.add(relationalDatabase);
```

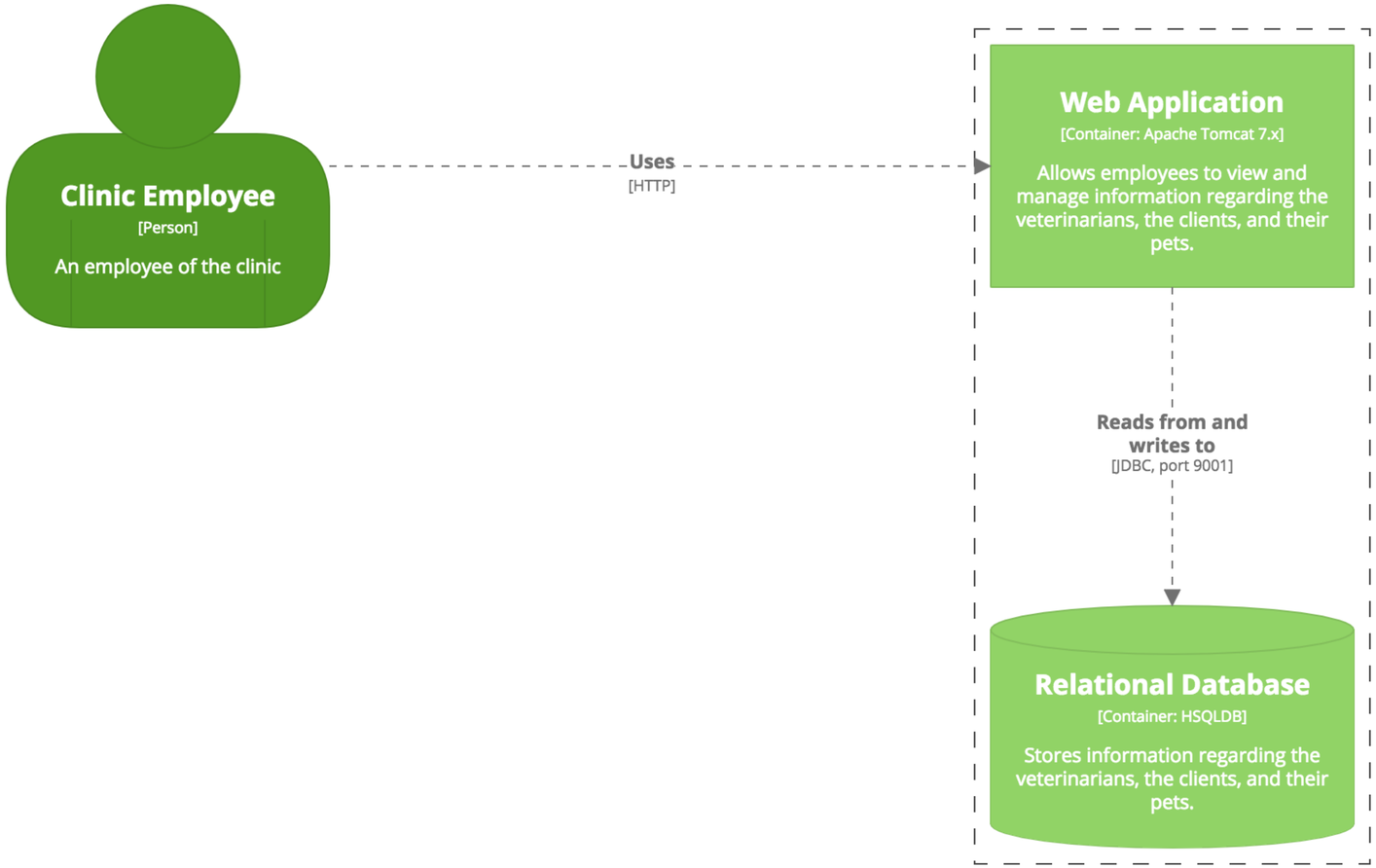
```
// upload the software architecture model to structurizr.com
StructurizrClient client = new StructurizrClient("key", "secret");
client.mergeWorkspace(1234, workspace);
```


```
{
 "id" : 0,
 "name" : "Spring PetClinic",
 "description" : "This is a C4 representation of the Spring PetClinic sample app (https://github.com/spring-projects/spring-petclinic/)",
 "model" : {
   "people" : [ {
     "tags" : "Element,Person",
     "id" : "2",
     "name" : "Clinic Employee",
     "description" : "An employee of the clinic",
     "relationships" : [ {
       "tags" : "Relationship,Synchronous",
       "id" : "3",
       "sourceId" : "2",
       "destinationId" : "1",
       "description" : "Uses",
       "interactionStyle" : "Synchronous"
     }, {
       "tags" : "Relationship,Synchronous",
       "id" : "6",
       "sourceId" : "2",
       "destinationId" : "4",
       "description" : "Uses",
       "technology" : "HTTP",
       "interactionStyle" : "Synchronous"
     }, {
       "tags" : "Relationship,Synchronous",
       "id" : "28",
       "sourceId" : "2",
       "destinationId" : "8",
       "description" : "Uses",
       "technology" : "HTTP",
       "interactionStyle" : "Synchronous"
```



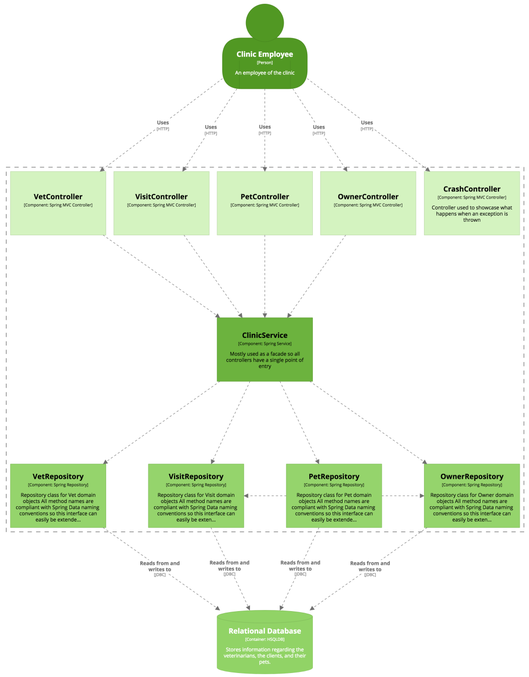
System Context diagram for Spring PetClinic

The System Context diagram for the Spring PetClinic system.
Friday 18 November 2016 22:21 UTC



Container diagram for Spring PetClinic

The Containers diagram for the Spring PetClinic system.
Friday 18 November 2016 22:21 UTC



Component diagram for Spring PetClinic - Web Application

The Components diagram for the Spring PetClinic web application.
Friday 18 November 2016 22:21 UTC

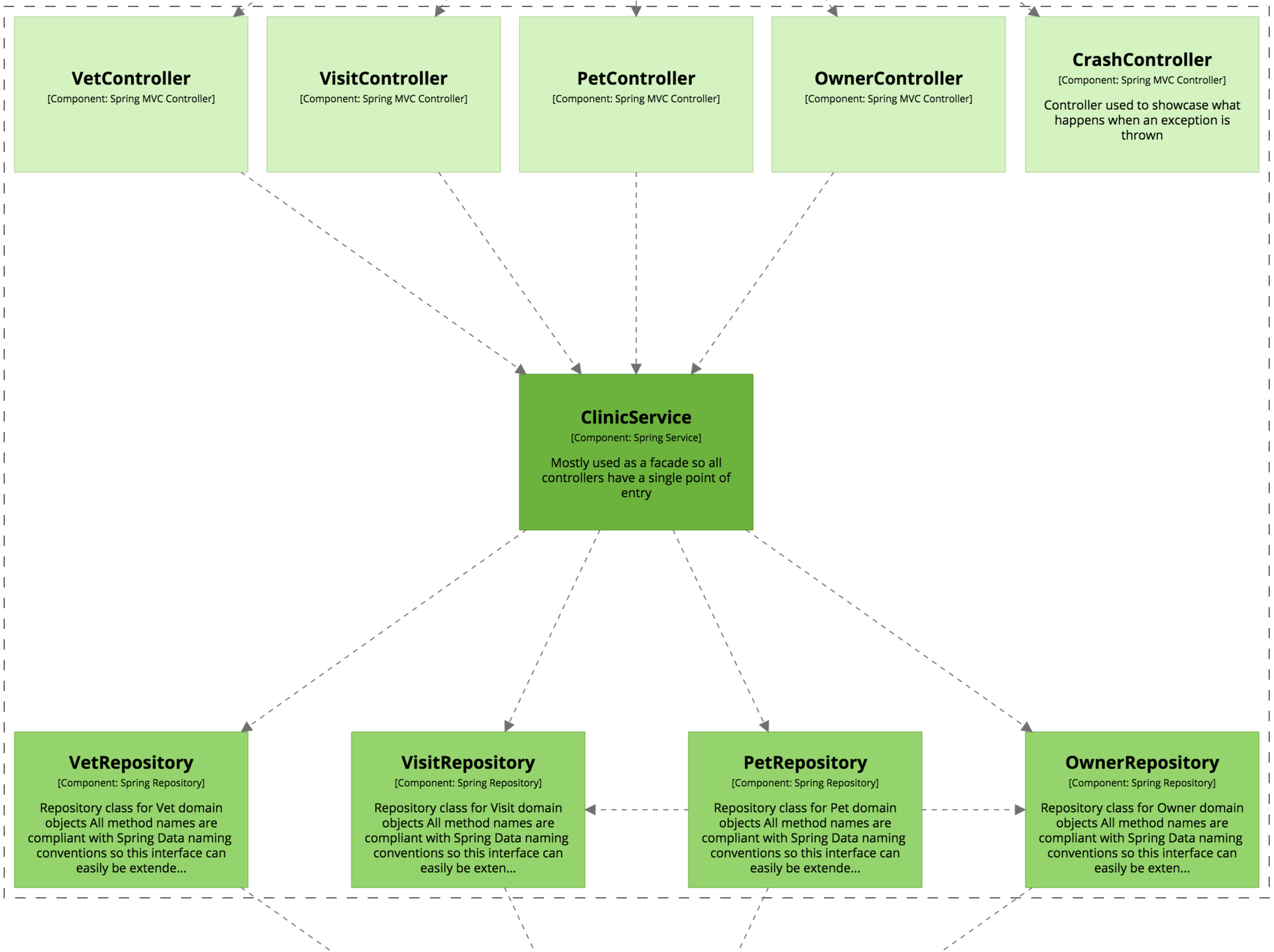
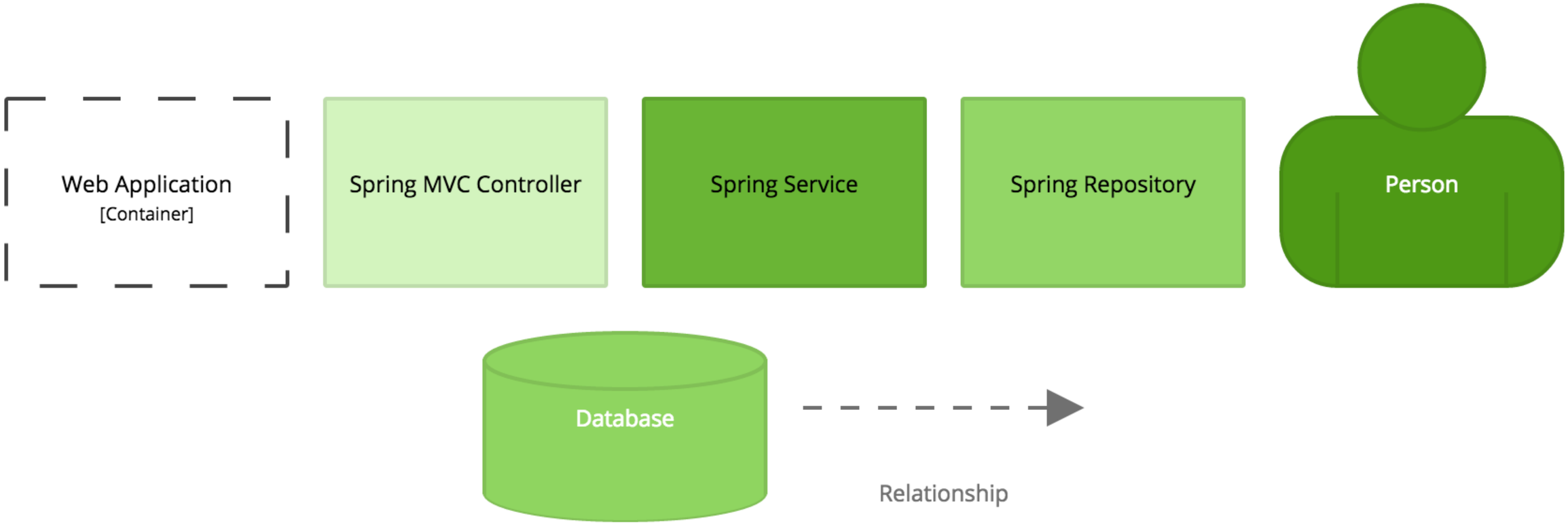
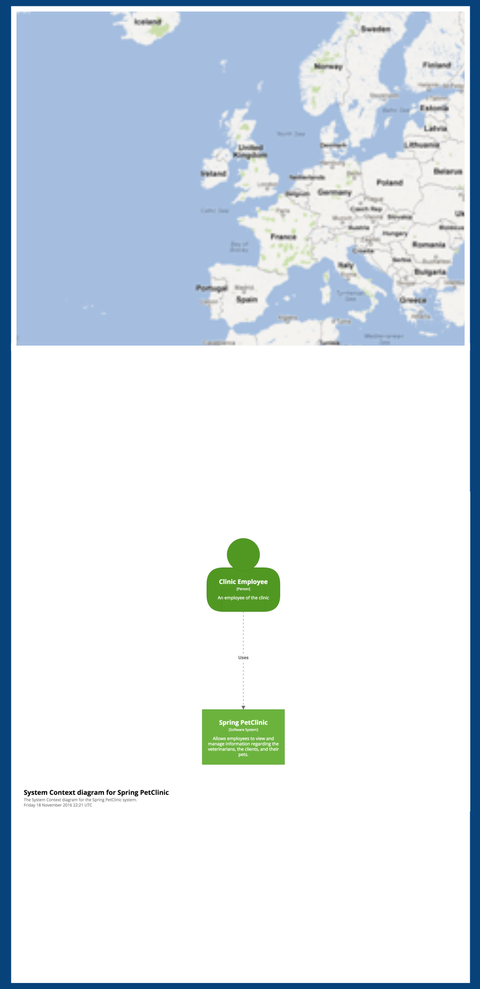


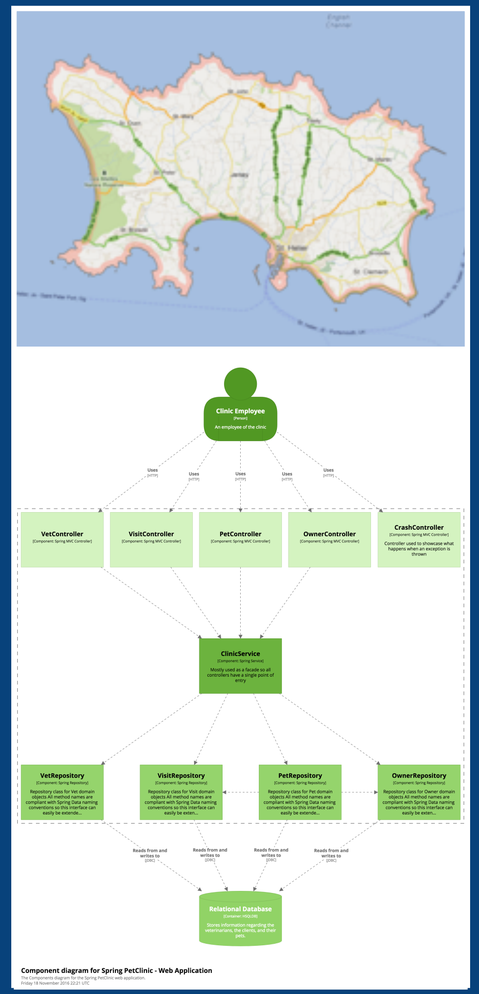
Diagram key

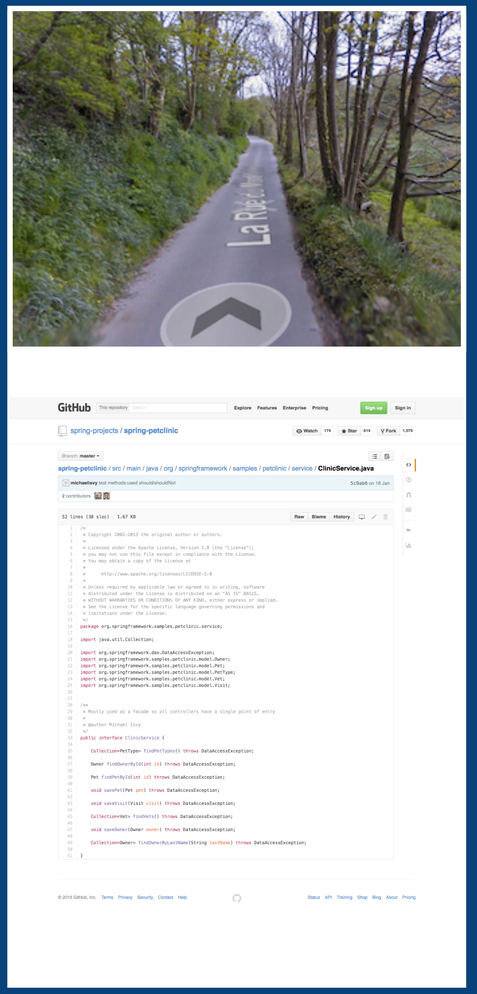
Here are the styles that have been used on this diagram.



Close





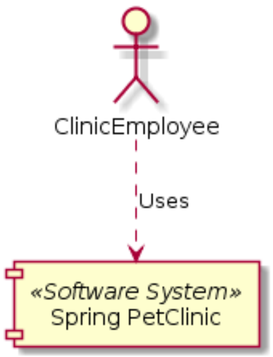


Diagrams are maps

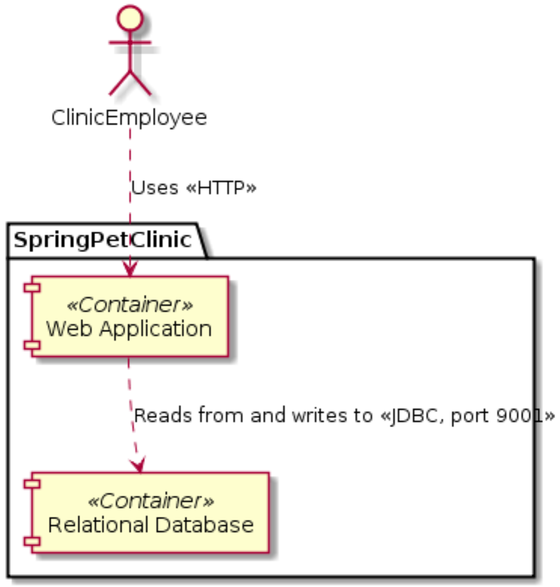
that help software developers navigate a large and/or complex codebase

You can create
many **visualisations**
from a single model

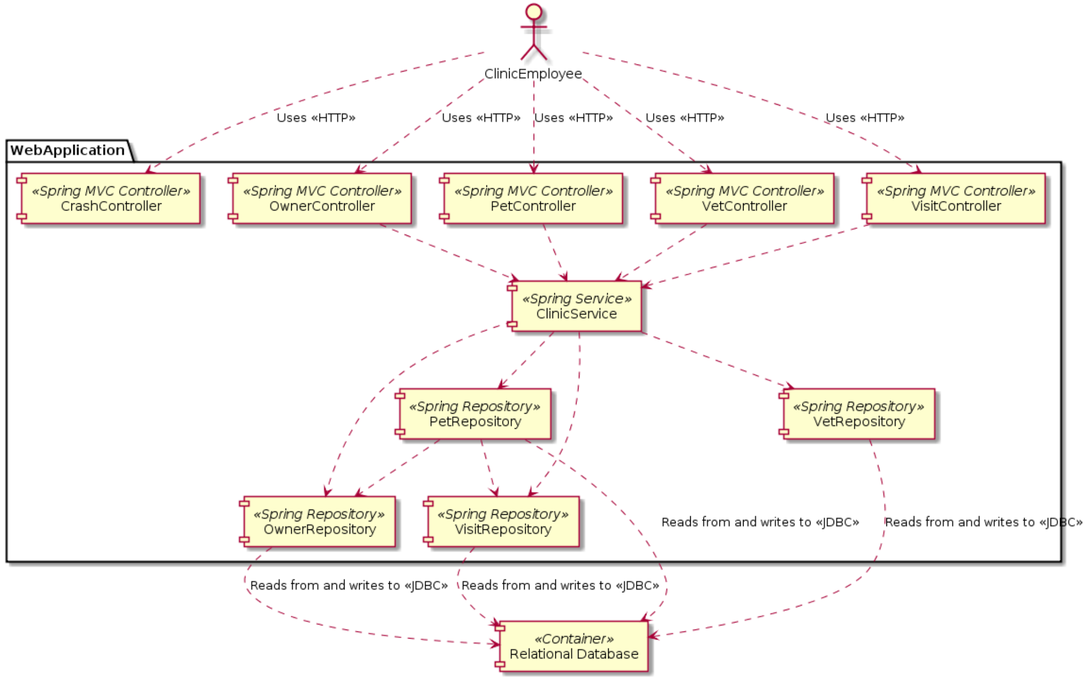
Spring PetClinic - System Context



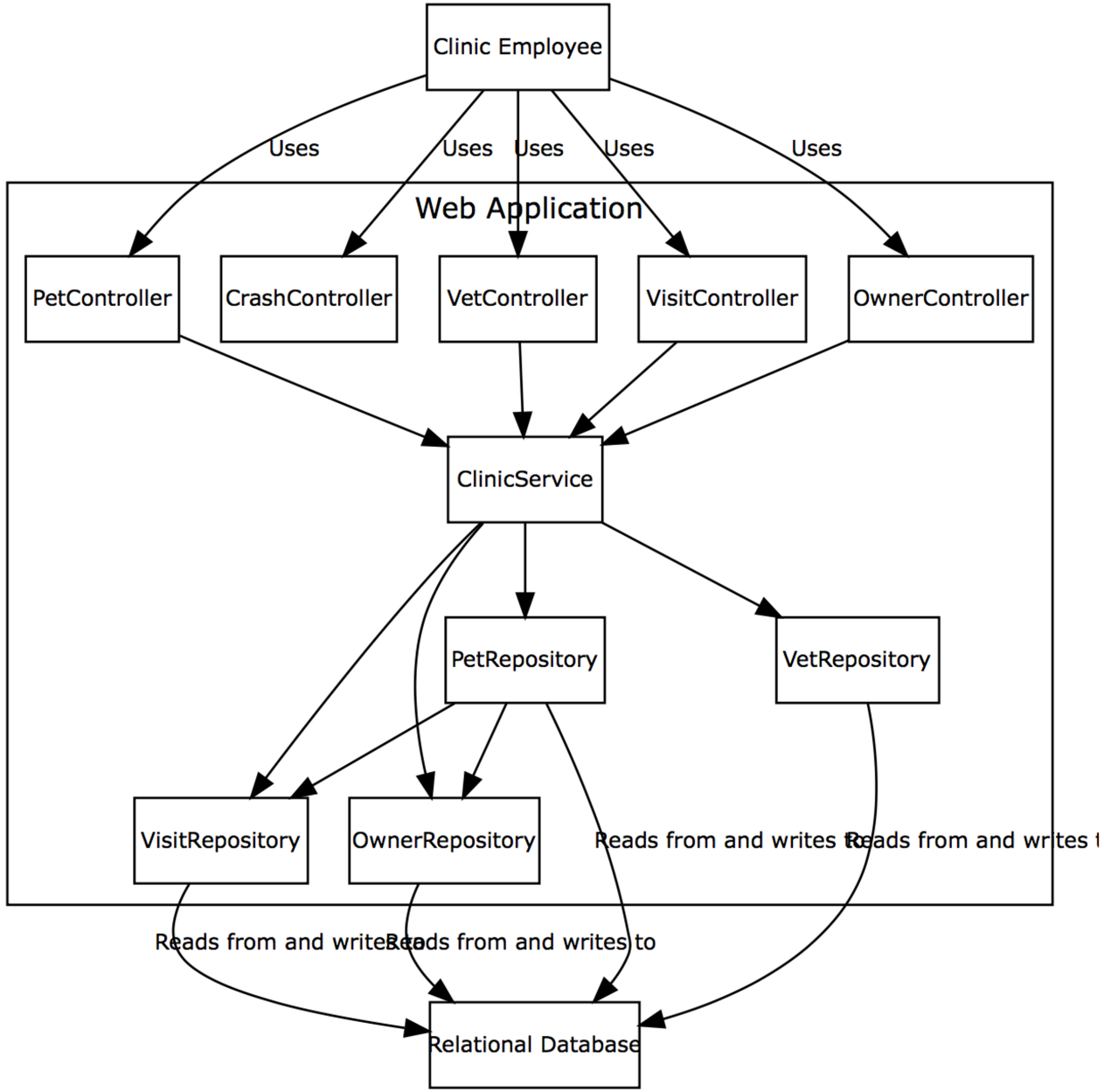
Spring PetClinic - Containers



Spring PetClinic - Web Application - Components

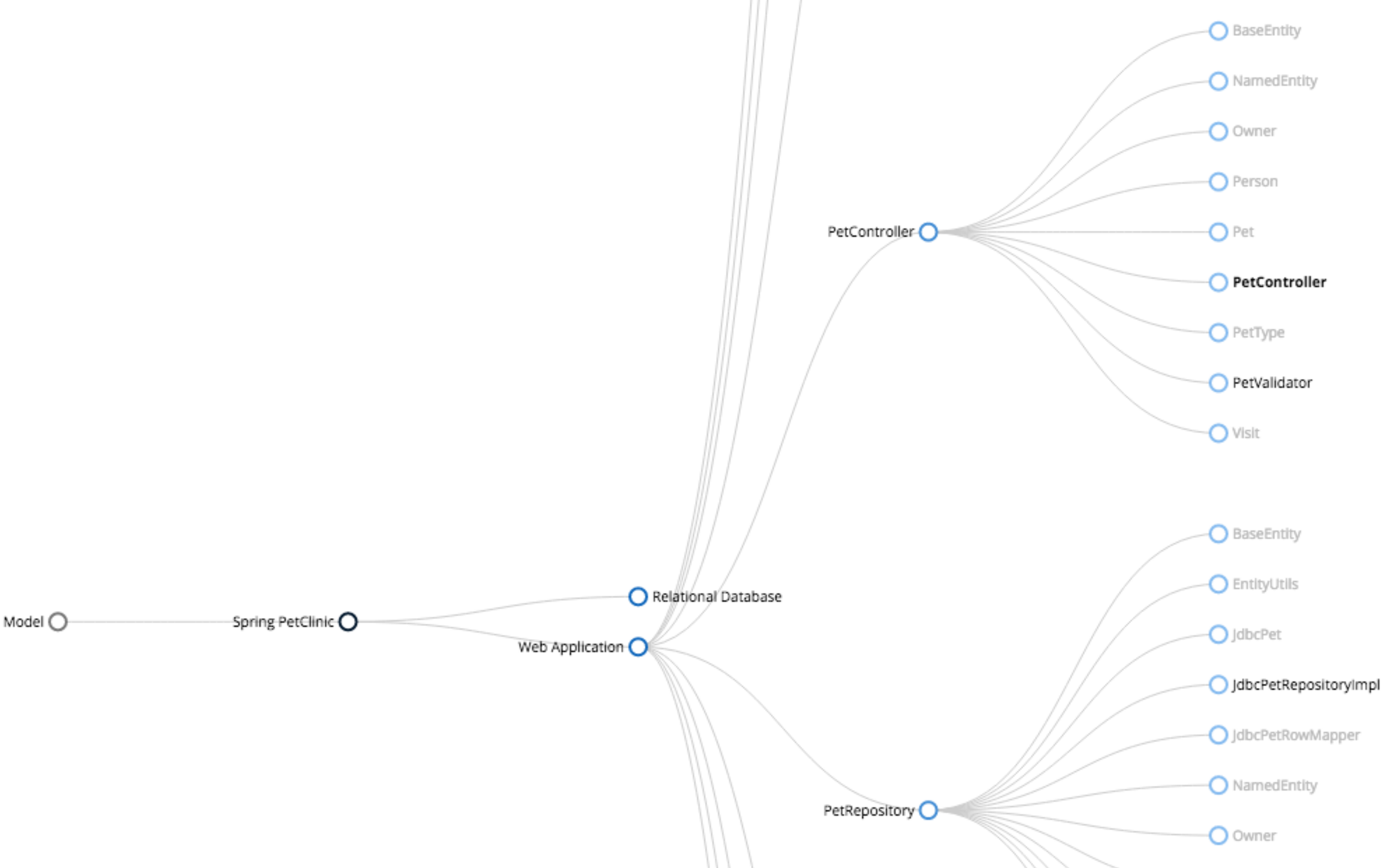


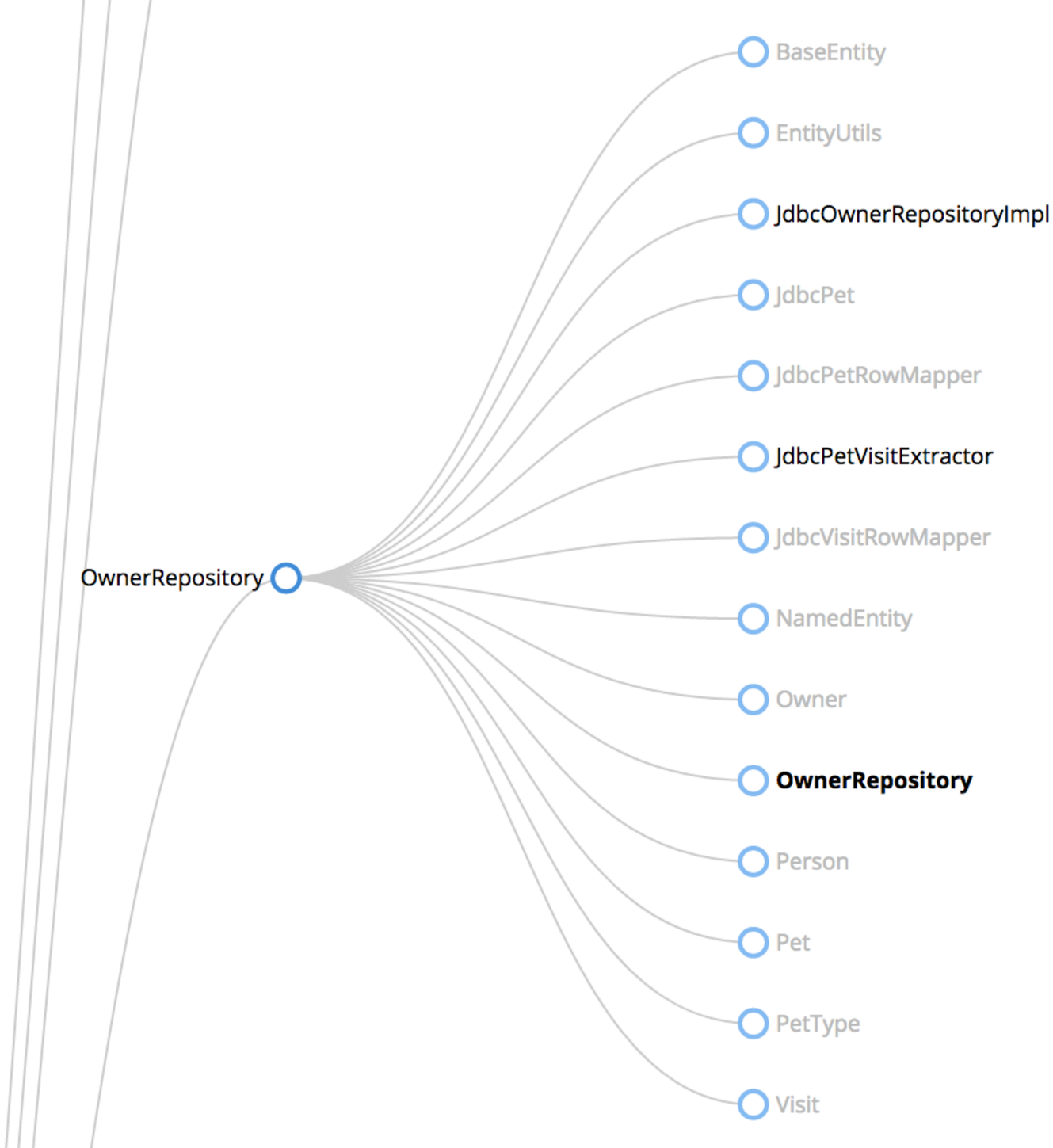
Spring PetClinic - Web Application - Components

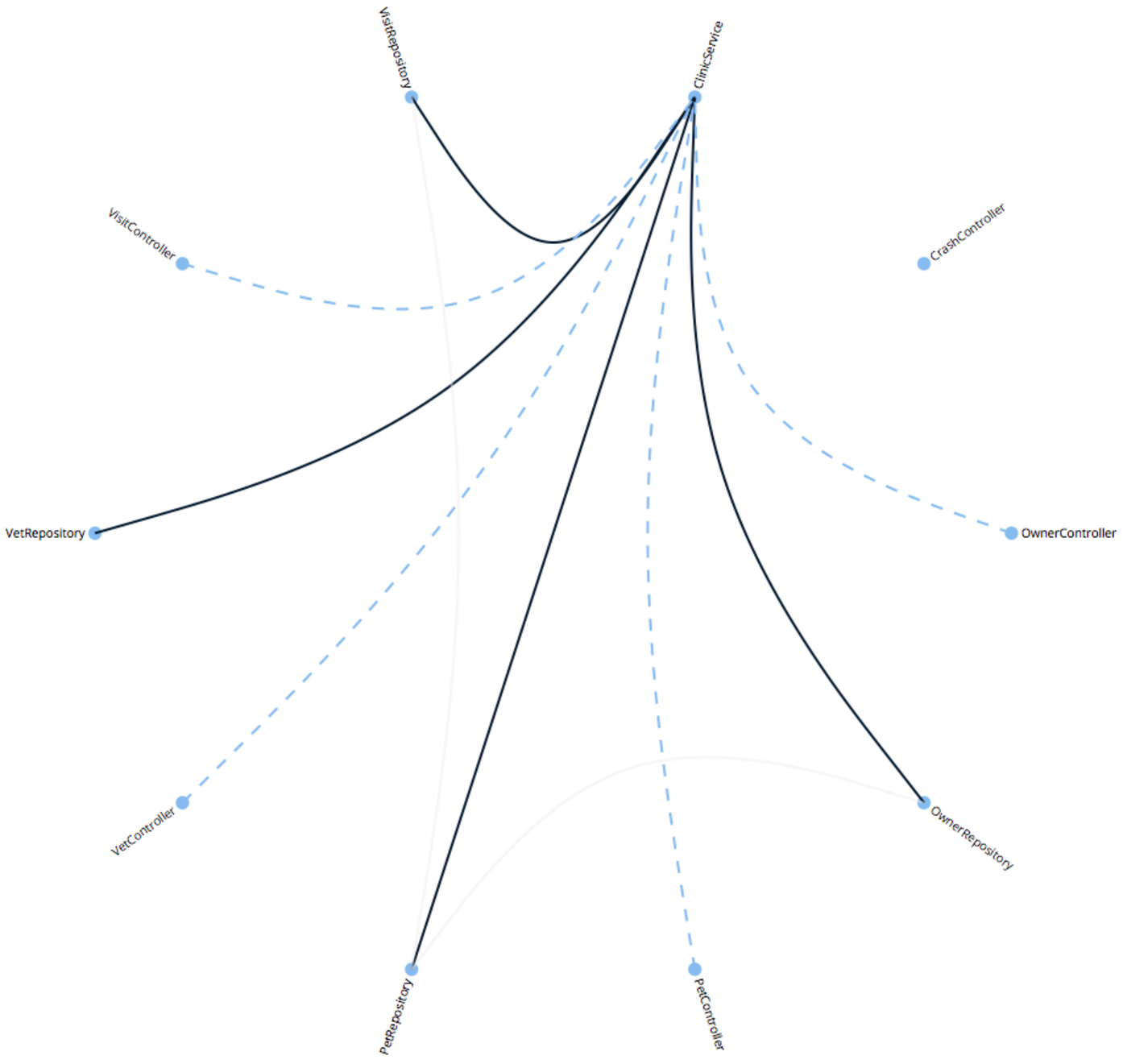


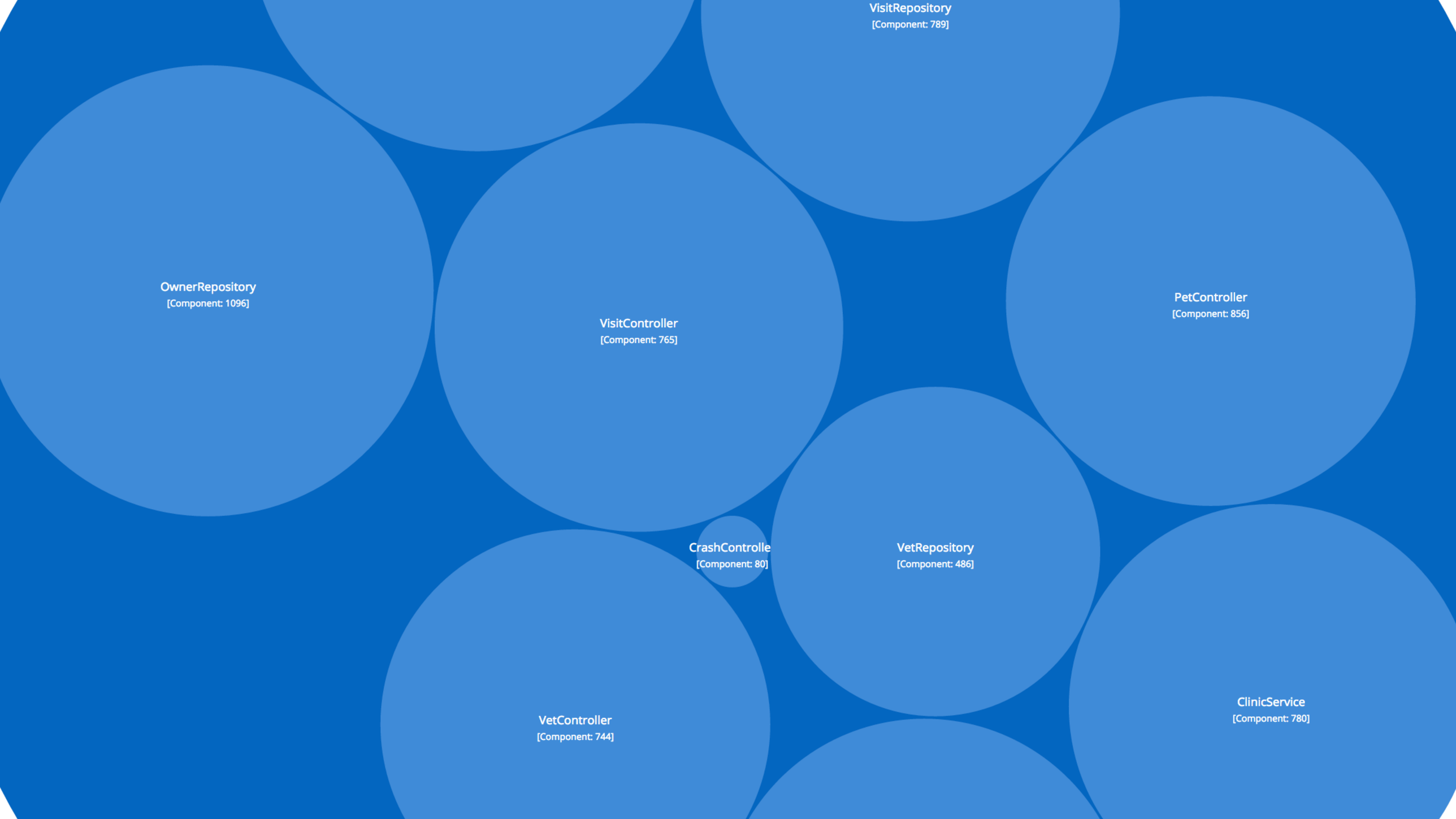
Integration with your build process keeps models up to date

Once you have a model of your software system, you can **explore** it


BaseEntity
NamedEntity
Owner
Person
Pet
PetController
PetType
PetValidator
Visit
PetController
BaseEntity
EntityUtils
JdbcPet
JdbcPetRepositoryImpl
JdbcPetRowMapper
NamedEntity
Owner
PetRepository
Relational Database
Model
Spring PetClinic
Web Application


BaseEntity
EntityUtils
JdbcOwnerRepositoryImpl
JdbcPet
JdbcPetRowMapper
JdbcPetVisitExtractor
JdbcVisitRowMapper
OwnerRepository
NamedEntity
Owner
OwnerRepository
Person
Pet
PetType
Visit


VisitRepository
ClinicService
VisitController
CrashController
VetRepository
OwnerController
VetController
OwnerRepository
PetRepository
PetController


VisitRepository
[Component: 789]
OwnerRepository
[Component: 1096]
VisitController
[Component: 765]
PetController
[Component: 856]
CrashControlle
[Component: 80]
VetRepository
[Component: 486]
VetController
[Component: 744]
ClinicService
[Component: 780]


OwnerRepository
[Code: 64]
Visit
[Code: 123]
BaseEntity
[Code: 47]
JdbcPetRowMapper
[Code: 43]
Person
[Code: 56]
PetType
[Code: 29]
JdbcVisitRowMapper
[Code: 42]
JdbcPetVisitExtractor
[Code: 54]
JdbcPet
[Code: 48]
Owner
[Code: 153]
NamedEntity
[Code: 48]
EntityUtils
[Code: 54]

Summary

The **1990**’s called and they want their tools back!

It’s 2016 and we shouldn’t be using a general purpose diagramming tool for software architecture

Abstractions first, notation second

Ensure that your team has a ubiquitous language to describe software architecture

Thank you!

simon.brown@codingthearchitecture.com
@simonbrown

Software Architecture for Developers

Volume I

Technical leadership and the balance with agility

Simon Brown

Leanpub

Software Architecture for Developers

Volume 2

Visualise, document and explore your software architecture

Simon Brown