Software architecture as a contributor to high performing teams

Simon Brown

DDD, agile, Spotify model, Team Topologies, feature teams, Conway's Law, platform teams, autonomy, trust, and other socio-technical techniques

will fail without

good technical foundations

and effective communication

You can’t optimise
team performance
without good
technical foundations

Autonomy

vs

alignment

Every **software** problem
is fundamentally
a **communication** problem

The **biggest impact**
a software architect can make
is by **improving communication**

“Leveling up Your Architecture Game” by Thomas Betts
https://www.infoq.com/presentations/progress-communication-collaboration/

I've witnessed a huge shift in how architecture is perceived over the past 15+ years

2008…

In 2008 I tried to pitch a book titled
"From developer to architect”
to a number of publishing companies

I was already an author/co-author,
and a technical reviewer,
so I had some contacts

we don't think that a book about architecture will sell

2011

goto; conference aarhus

THE FRUSTRATED ARCHITECT

Simon Brown

@simonbrown

coding {the} architecture

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

gotocon.com

Big up front design

and analysis paralysis

Waterfall

UML

Ivory Tower

PowerPoint Architect

Architecture Astronaut

Flat, self-organising teams are great but...

...they don't always work

Who is teaching the classics of the pre-agile era?

Does *agile* need architecture?

Yes, architecture provides

structure, firm foundations, vision and technical leadership

Does architecture need *agile*?

Yes, it helps software teams move away from

big design up front

and analysis paralysis

Just enough

Understand how the significant elements fit together

Mitigate the key risks

Provide the foundations and vision to move forward

you've presented a straw man

(i.e. oversimplifying, exaggerating, etc)

2012

coding {the} architecture

Master builder

The original generalising specialist?

The software architecture role is about coding, coaching, and collaboration

Do you have any evidence for anything in your talk?

(e.g. the value of technical leadership, architects who code, etc)

the plural of anecdote is not data

coding {the} architecture

Effective architecture

sketches

Did you bring your crayons?

Please give a warm welcome to Bob the Builder

Now...

Software Architecture for Developers

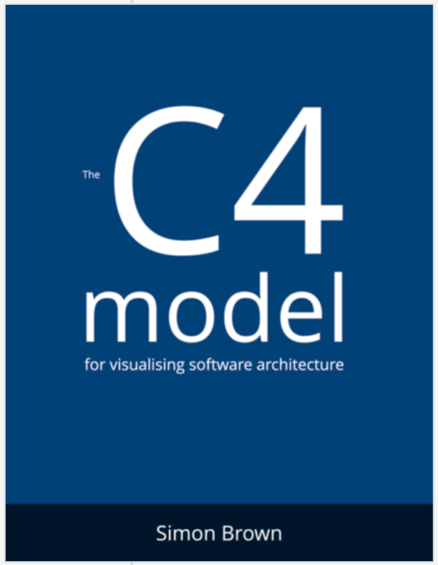
Technical leadership and the balance with agility

Simon Brown

27,377 READERS | 106 PAGES | 100% COMPLETE

PDF EPUB WEB ENGLISH

This book is a practical, pragmatic and lightweight guide to software architecture, specifically aimed at developers, and focussed around the software architecture role and process.



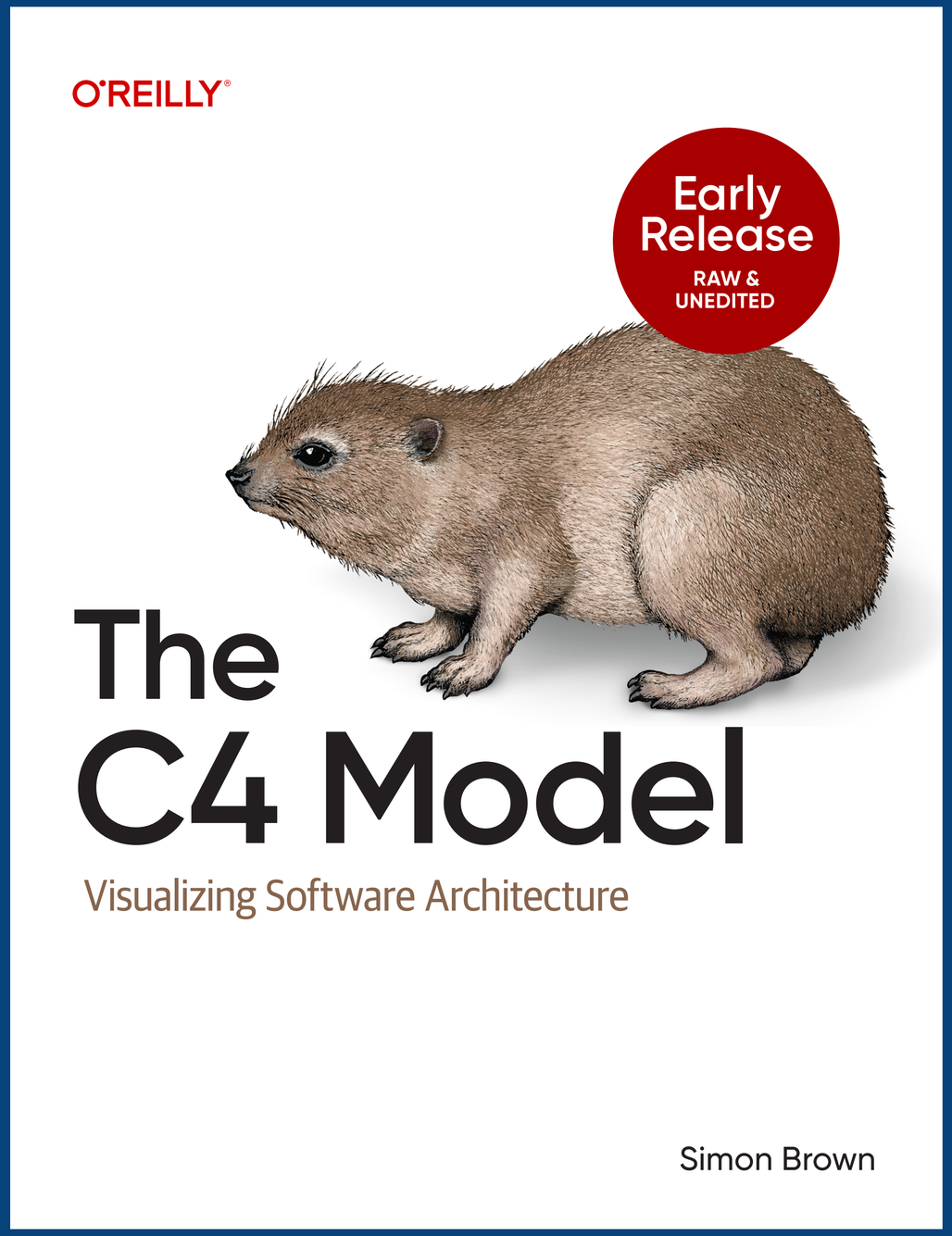
The C4 model for visualising software architecture

Simon Brown

26,427 READERS | 110 PAGES | 100% COMPLETE

PDF EPUB WEB ENGLISH

This book focusses on the visual communication of software architecture, based upon a collection of ideas and techniques that thousands of people across the world have found useful - the C4 model for visualising software architecture.



Scheduled for release July 2026,
early access available now
via the O'Reilly platform

I've been running software architecture workshops since 2008

the plural of anecdote is data

200+ clients

From startups and small country specific businesses,
to scale-ups and global household names

200+ clients

IT services and consulting, advertising, retail banking, investment banking, investment and wealth management, trading, FinTech, credit cards, payments, accounting, insurance, recruitment, law, e-commerce, airlines, ferries, travel agencies, universities, academia, research, community organisations, government, welfare, pensions, charity, cartography, property, weather, environmental science, energy, entertainment, music, TV, games, online gaming, electronics, technology, robotics, automotive, telephony, communications, publishing, health care, pharmaceutical, ...

(plus hundreds more companies represented by attendees at public workshops organised via conferences)

~40 countries

England, Wales, Scotland, Northern Ireland, Ireland, Jersey, France, Belgium, Netherlands, Denmark, Sweden, Norway, Finland, Lithuania, Latvia, Poland, Germany, Czechia, Austria, Switzerland, Slovakia, Hungary, Romania, Ukraine, Croatia, North Macedonia, Italy, Greece, Spain, Portugal, Iceland, United States of America, Canada, United Arab Emirates, India, China, South Korea, Singapore, Australia

Who do I usually engage with?

Engineering Manager, CTO, Head of Architecture Profession, Head of Enterprise Architecture, Director of Software Product, Architecture Lead, Director of Technology, Solution Architect, Director of Software Engineering, VP of Engineering, Tech Lead, etc...

The software architecture diagramming exercise

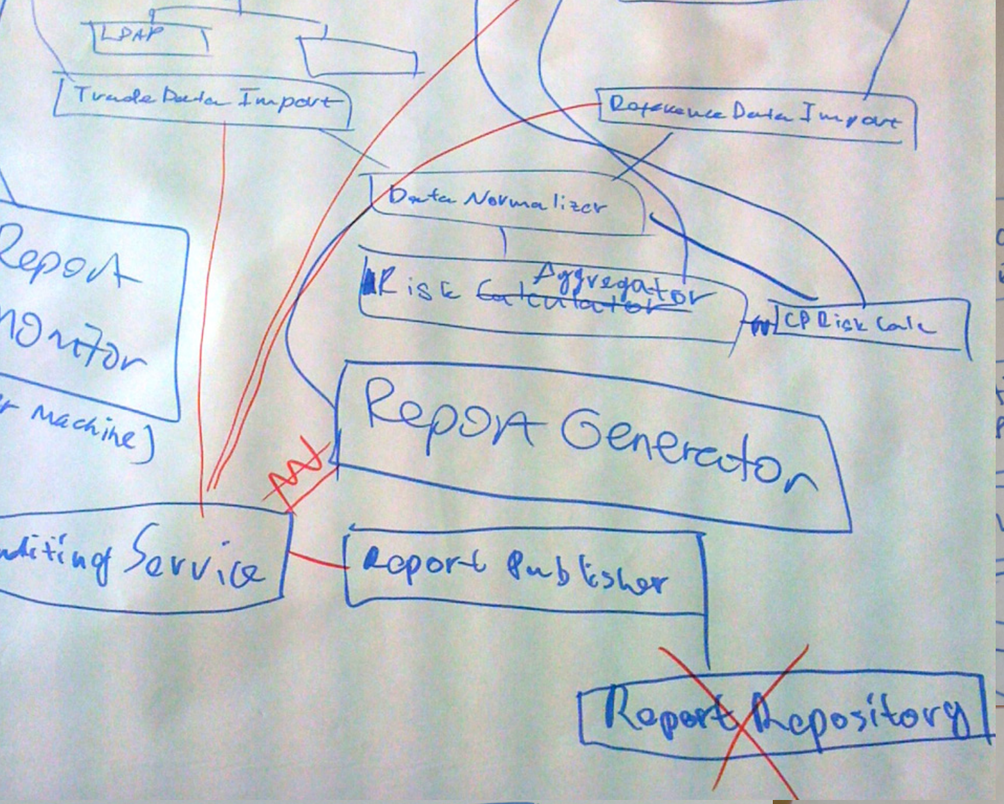
Design a software solution for the ”Financial Risk System”, and **draw** one or more architecture diagrams to describe your solution

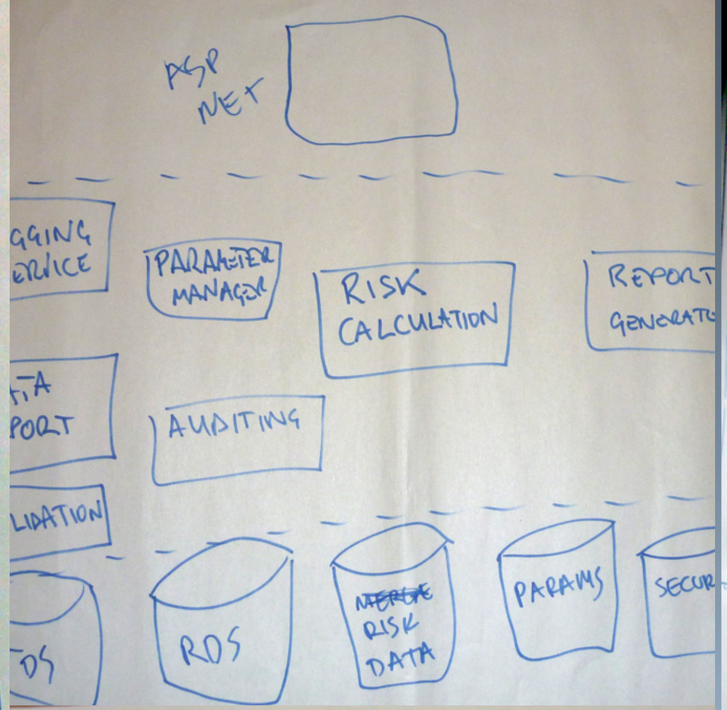
60-90 minutes

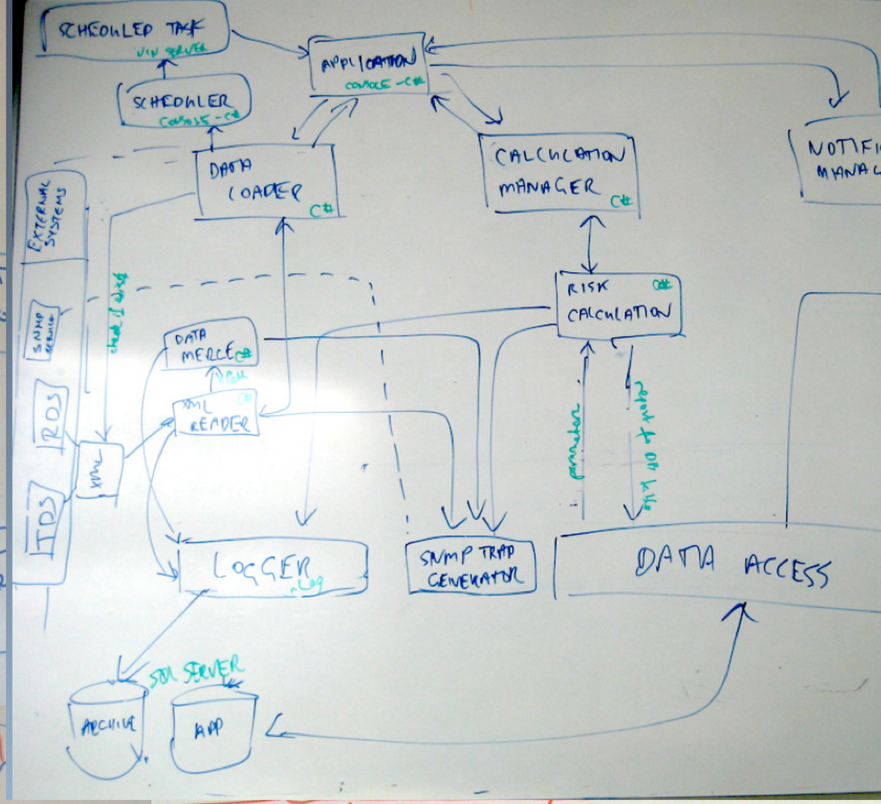
Draw one or more software architecture diagrams to describe a software system/service/product/etc

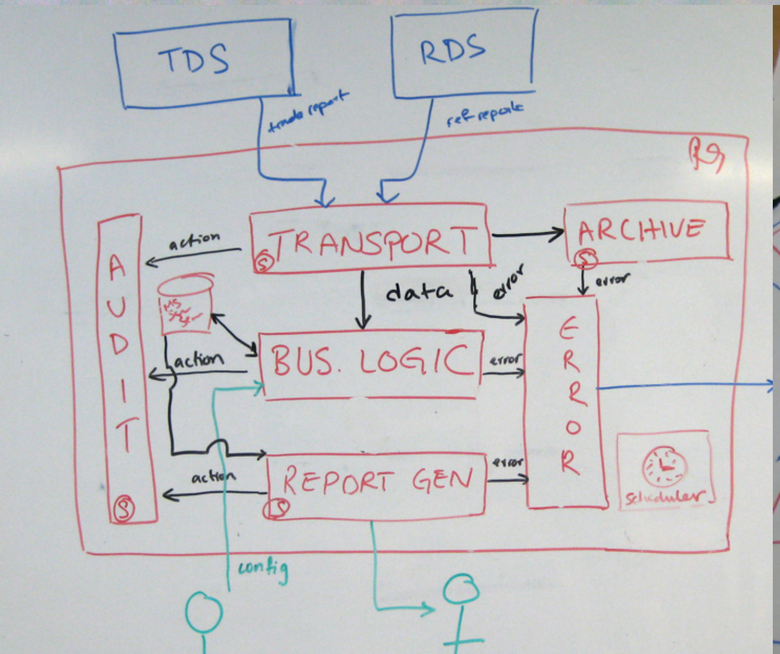
30 minutes

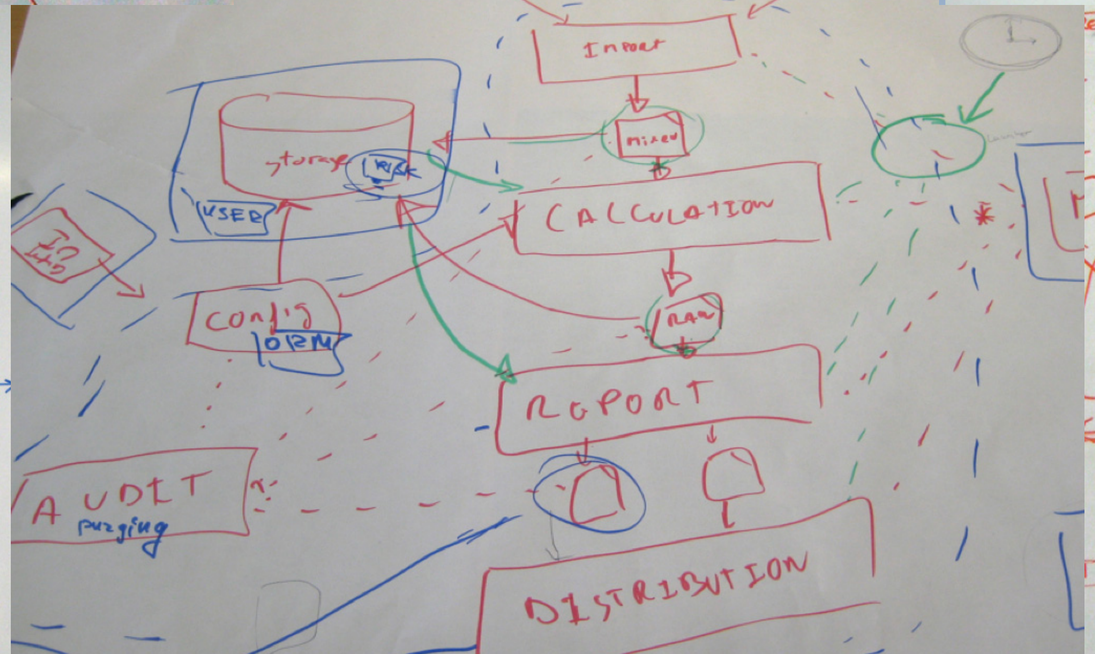
Architecture is more than “boxes and arrows” but…

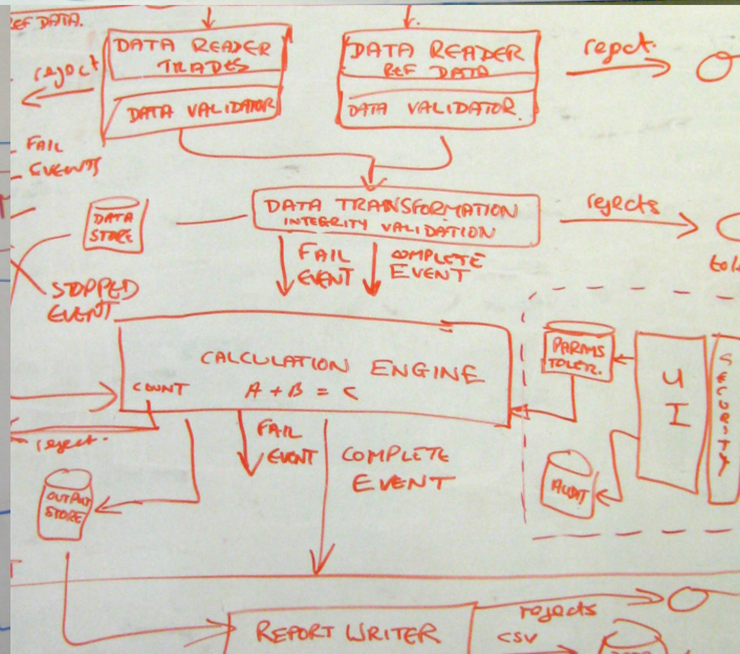

LDAP
Trade Data Import
Reference Data Import
Data Normalizer
Aggregator
Risk Calculator
CP Risk Calc
Report Generator
Report Publisher
Report Repository


ASP NET
PARAMETER MANAGER
RISK CALCULATION
AUDITING
RDS
RISK DATA
PARAMS


SCHEDULED TASK
SCHEDULER
APPLICATION
DATA LOADER
CALCULATION MANAGER
RISK CALCULATION
DATA MERGER
XML READER
LOGGER
SNMP TRAP GENERATOR
DATA ACCESS
SQL SERVER
ARCHIVE
APP


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


TDS
RDS
Input
mixer
storage
CALCULATION
config
ORM
REPORT
AUDIT
purging
DISTRIBUTION


DATA READER TRADES
DATA READER REF DATA
DATA VALIDATOR
DATA VALIDATOR
reject
FAIL EVENT
DATA TRANSFORMATION
INTEGRITY VALIDATION
FAIL EVENT
COMPLETE EVENT
rejects
STOPPED EVENT
CALCULATION ENGINE
COMPLETE EVENT
REPORT WRITER
rejects
csv

We don't have a standard way to discuss our products from an engineering perspective

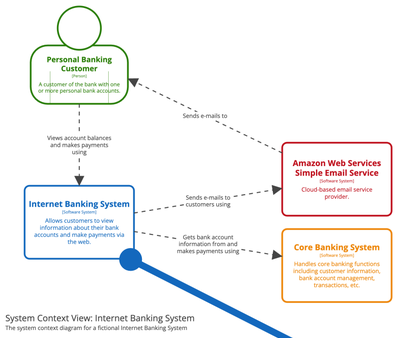
Our lack of documentation is slowing us down

It takes too long for new hires to be effective

We've failed our compliance audit through our lack of technical documentation

C4

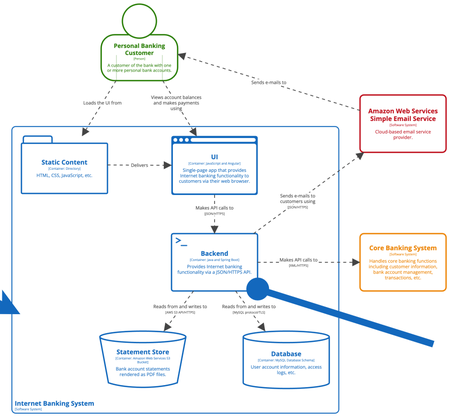
c4model.com



System Context View: Internet Banking System

The system context diagram for a fictional Internet Banking System

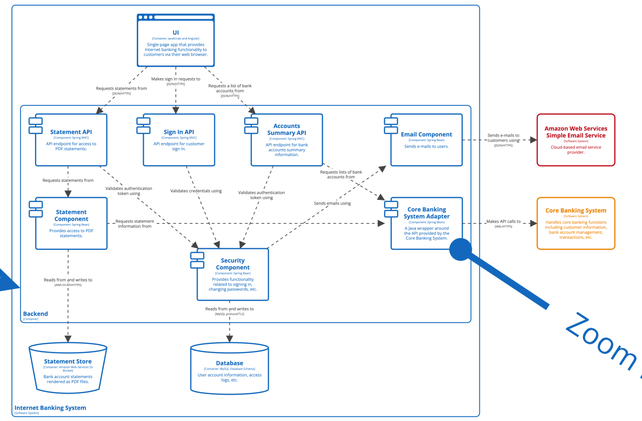
Zoom in



Container View: Internet Banking System

The container diagram for the Internet Banking System

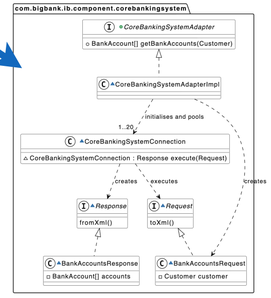
Zoom in



Component View: Internet Banking System - Backend

The component diagram for the Internet Banking System Backend

Zoom in



Code View: Internet Banking System - Backend - Core Banking System Adapter

A summary of the implementation details for the Core Banking System Adapter component

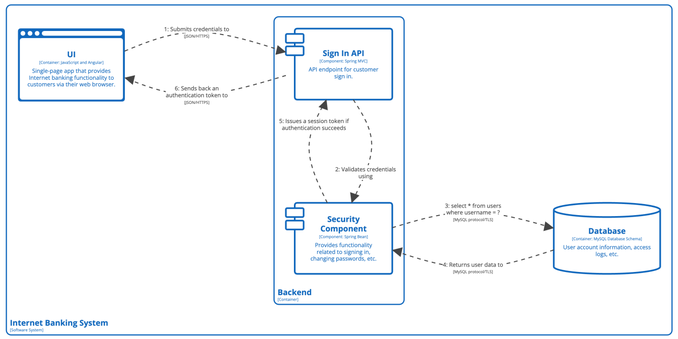
Static structure diagrams

System Context

Containers

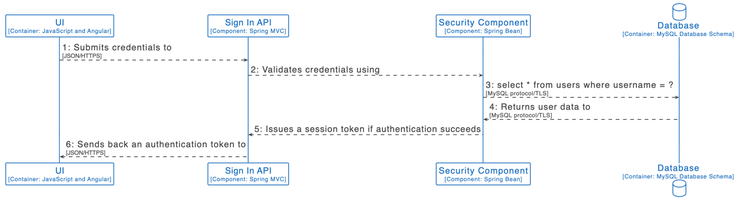
Components

Code



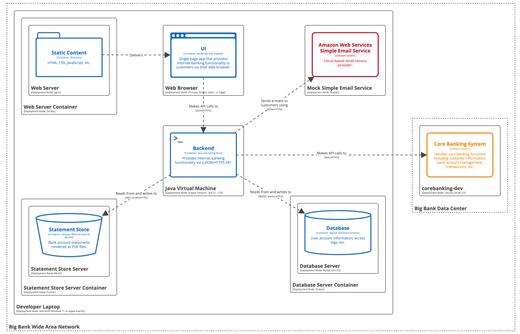
Dynamic View: Internet Banking System - Backend

Summarises how the sign in feature works in the single-page application



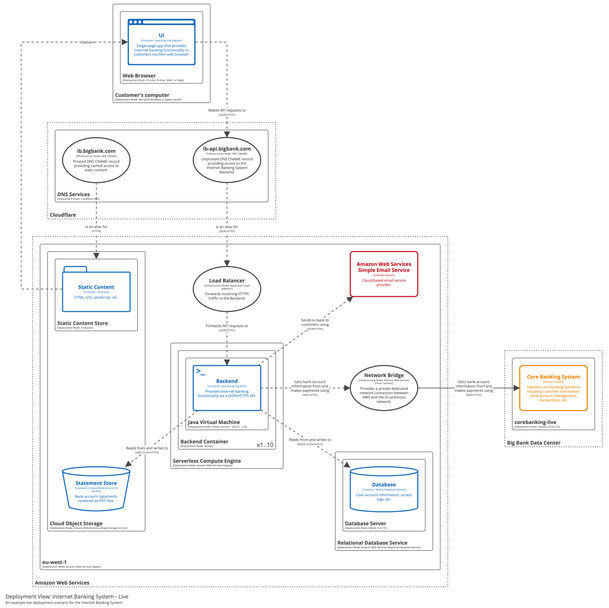
Dynamic View: Internet Banking System - Backend

Summarises how the sign in feature works in the single-page application



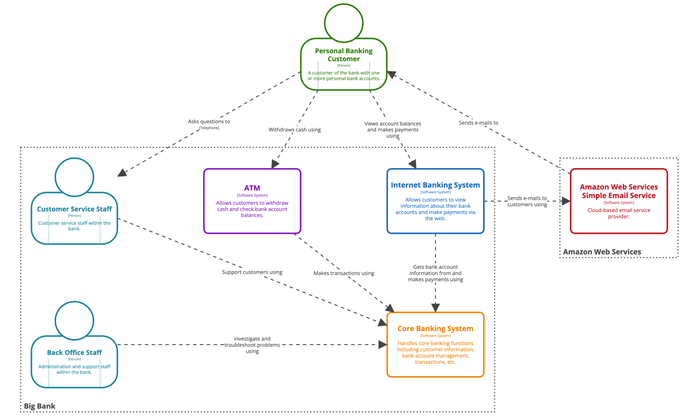
Deployment View: Internet Banking System - Development

An example development deployment scenario for the Internet Banking System



Deployment View: Internet Banking System - Live

An example live deployment scenario for the Internet Banking System



System Landscape View

A partial system landscape diagram for a fictional bank

Supporting diagrams

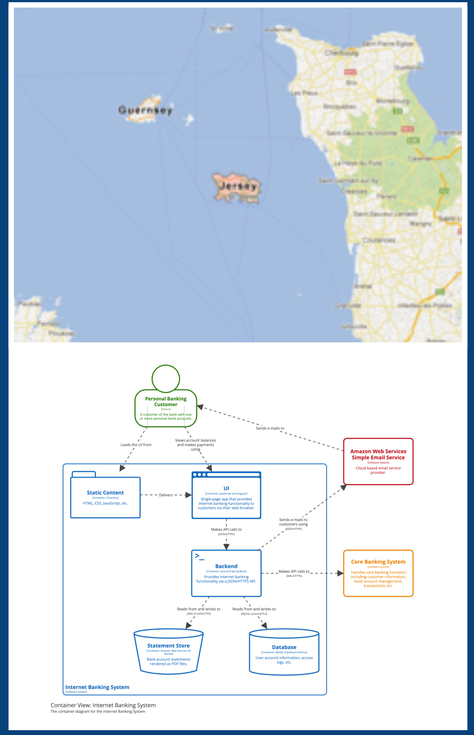
Dynamic

Deployment

System Landscape

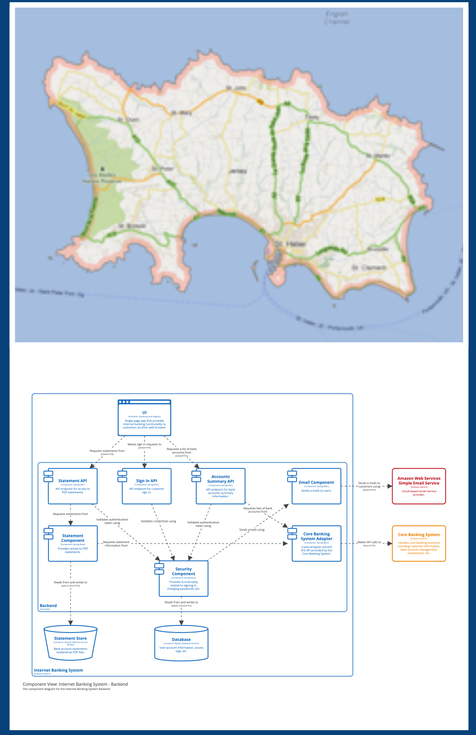
System Context View: Internet Banking System

The system context diagram for a fictional Internet Banking System



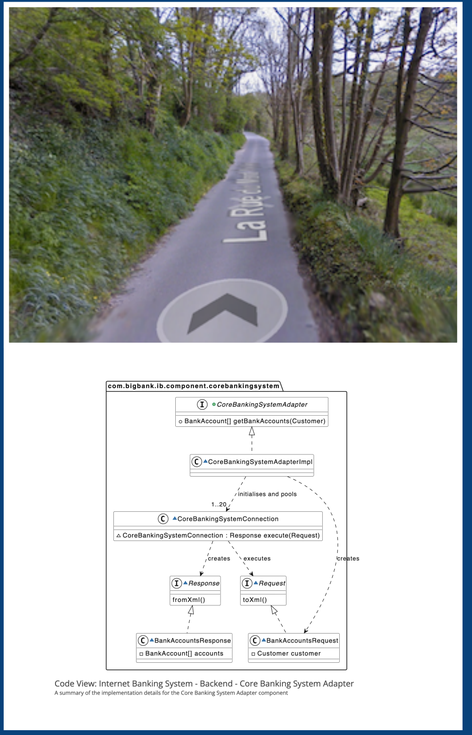
Container View: Internet Banking System

The container diagram for the Internet Banking System



Component View: Internet Banking System - Backend

The component diagram for the Internet Banking System backend



Code View: Internet Banking System - Backend - Core Banking System Adapter

A summary of the implementation details for the Core Banking System Adapter component

Diagrams are maps

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

The C4 model is...

A set of hierarchical abstractions

(software systems, containers, components, and code)

A set of hierarchical diagrams

(system context, containers, components, and code)

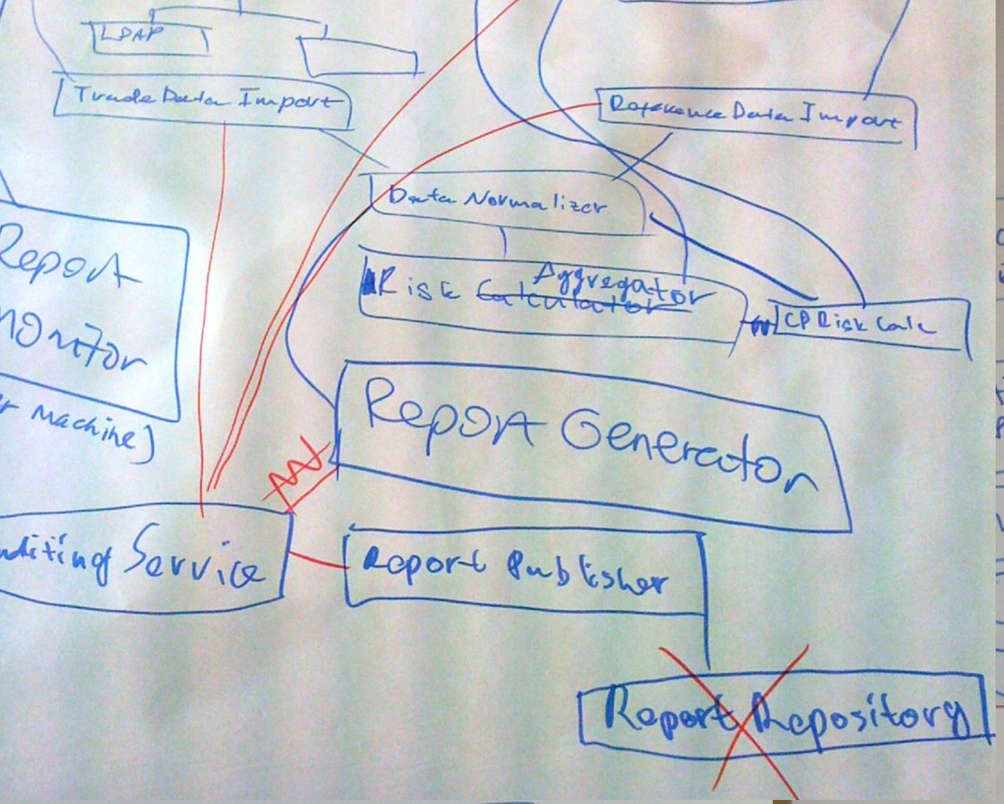
Notation independent

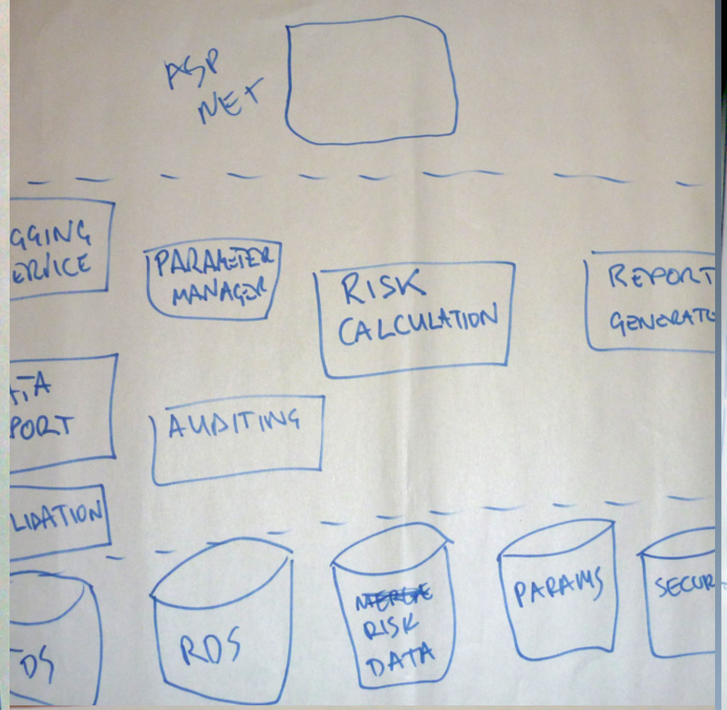
Tooling independent

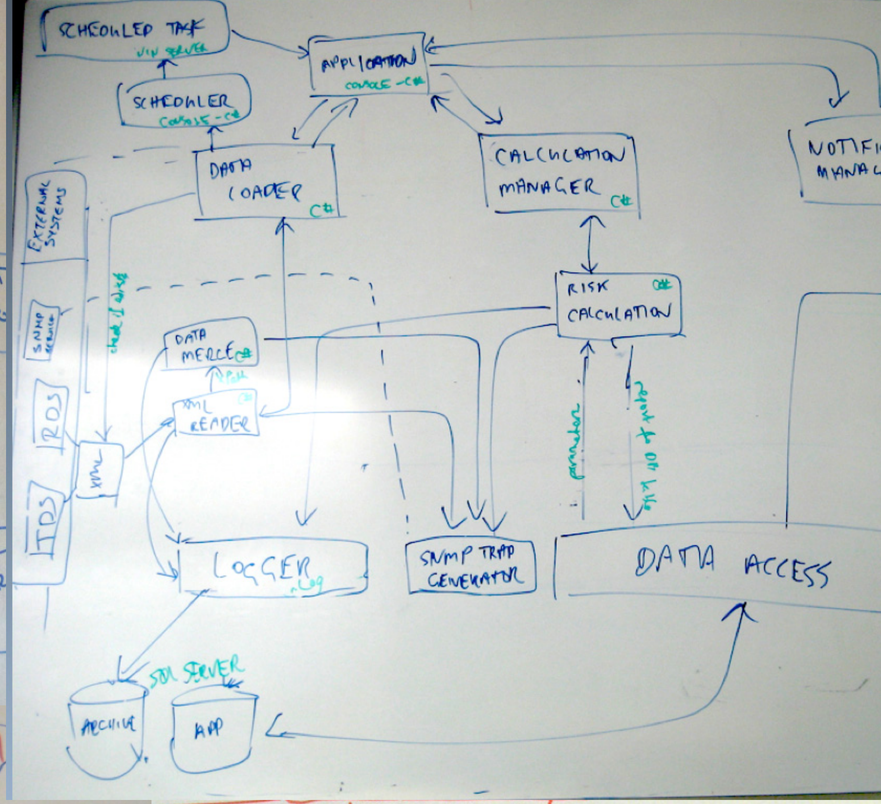
Design, decisions, direction

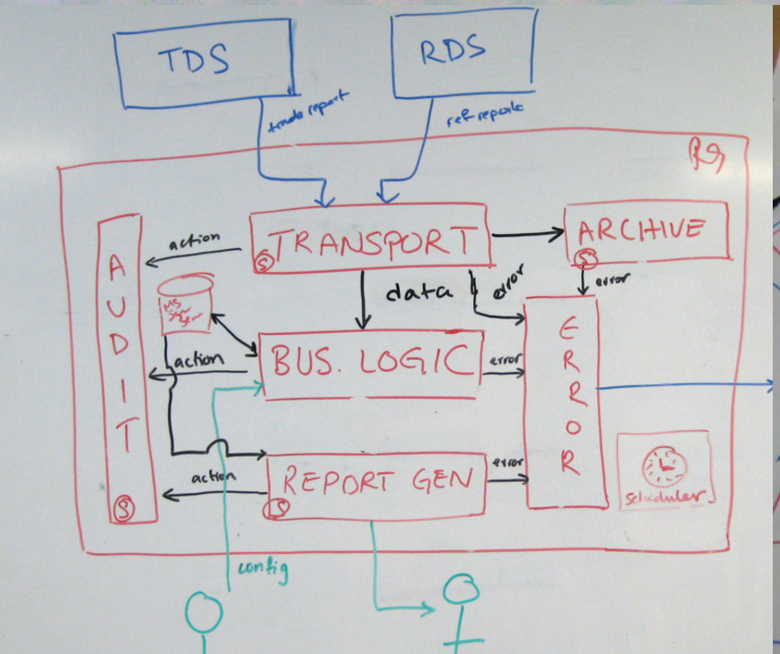
Our engineering teams are not considering architectural drivers - they’re missing key design inputs

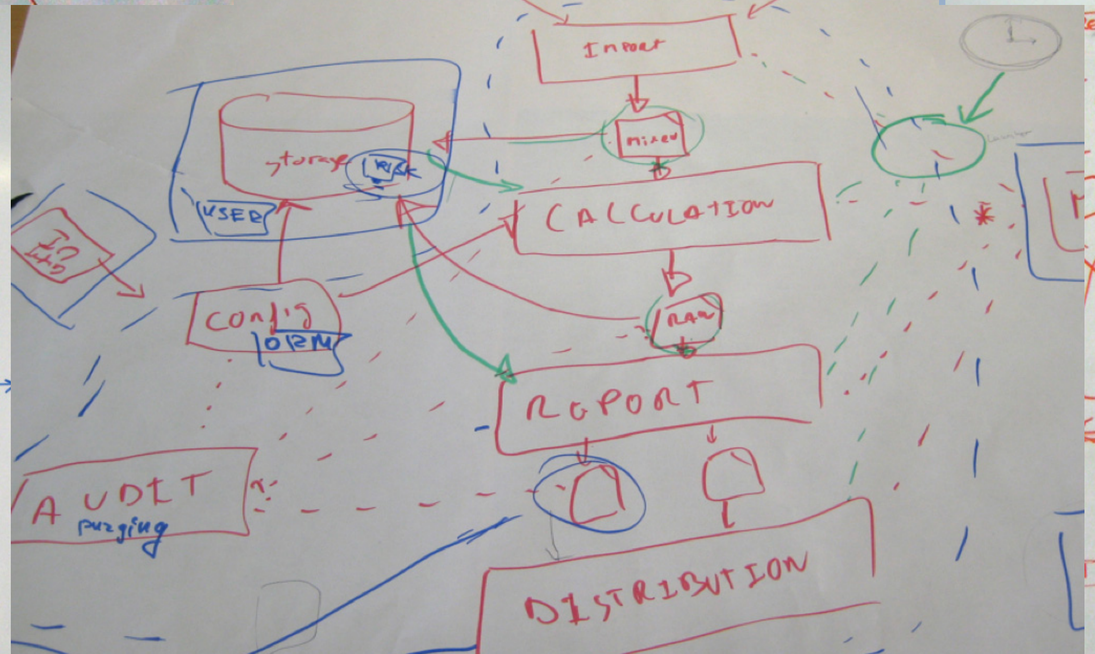
Our engineering teams are not considering the trade-offs of the decisions they are making

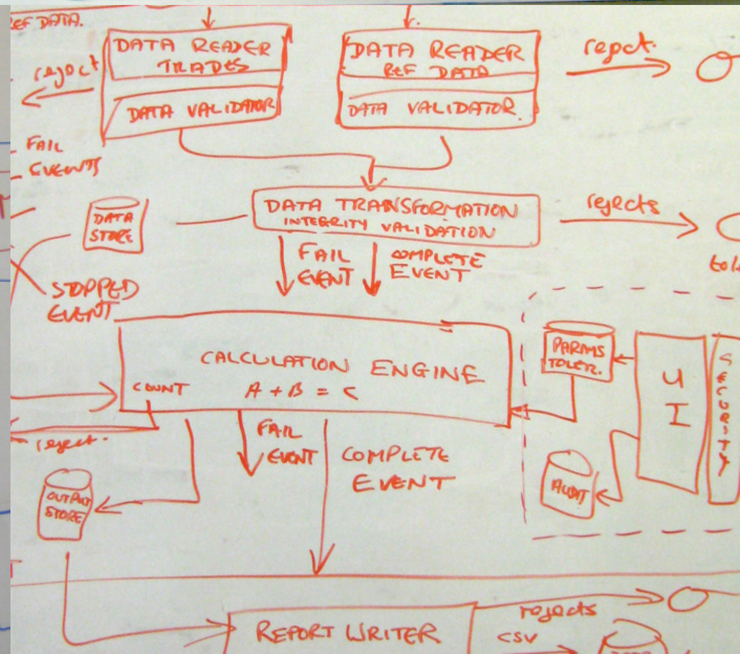

LDAP
Trade Data Import
Reference Data Import
Data Normalizer
Aggregator
Risk Calculator
CP Risk Calc
Report Generator
Report Publisher
Report Repository


ASP NET
PARAMETER MANAGER
RISK CALCULATION
AUDITING
RDS
RISK DATA
PARAMS


SCHEDULED TASK
SCHEDULER
APPLICATION
DATA LOADER
CALCULATION MANAGER
RISK CALCULATION
DATA MERGER
XML READER
LOGGER
SNMP TRAP GENERATOR
DATA ACCESS
SQL SERVER
ARCHIVE
APP


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


TDS
RDS
Input
mixer
storage
CALCULATION
config
ORM
REPORT
AUDIT
purging
DISTRIBUTION


DATA READER TRADES
DATA READER REF DATA
DATA VALIDATOR
DATA VALIDATOR
reject
FAIL EVENT
DATA TRANSFORMATION
INTEGRITY VALIDATION
FAIL EVENT
COMPLETE EVENT
rejects
STOPPED EVENT
CALCULATION ENGINE
COMPLETE EVENT
REPORT WRITER
rejects
csv

Search

Sign in

Agile meets Architecture

@ama_conf · 2.18K subscribers · 55 videos

The conference combining both worlds of agile and architecture – creating a community ...more

agile-meets-architecture.com and 2 more links

Subscribe

Home Videos Shorts Live Playlists Posts

Latest Popular Oldest

Simon Brown — The lost art of software design

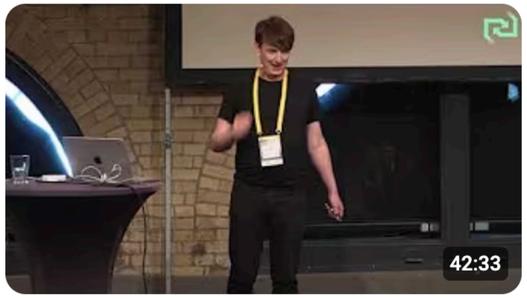
15K views • 2 years ago

Kevlin Henney — The Case for Technical Excellence

8.9K views • 1 year ago

Susanne Kaiser — Architecture for Flow with Wardley Mapping, DDD, and Team...

5.8K views • 2 years ago



Simon Rohrer — Modern Enterprise Architecture: architecting for outcomes...

5.5K views • 1 year ago

Diagrams are a visual checklist for design decisions

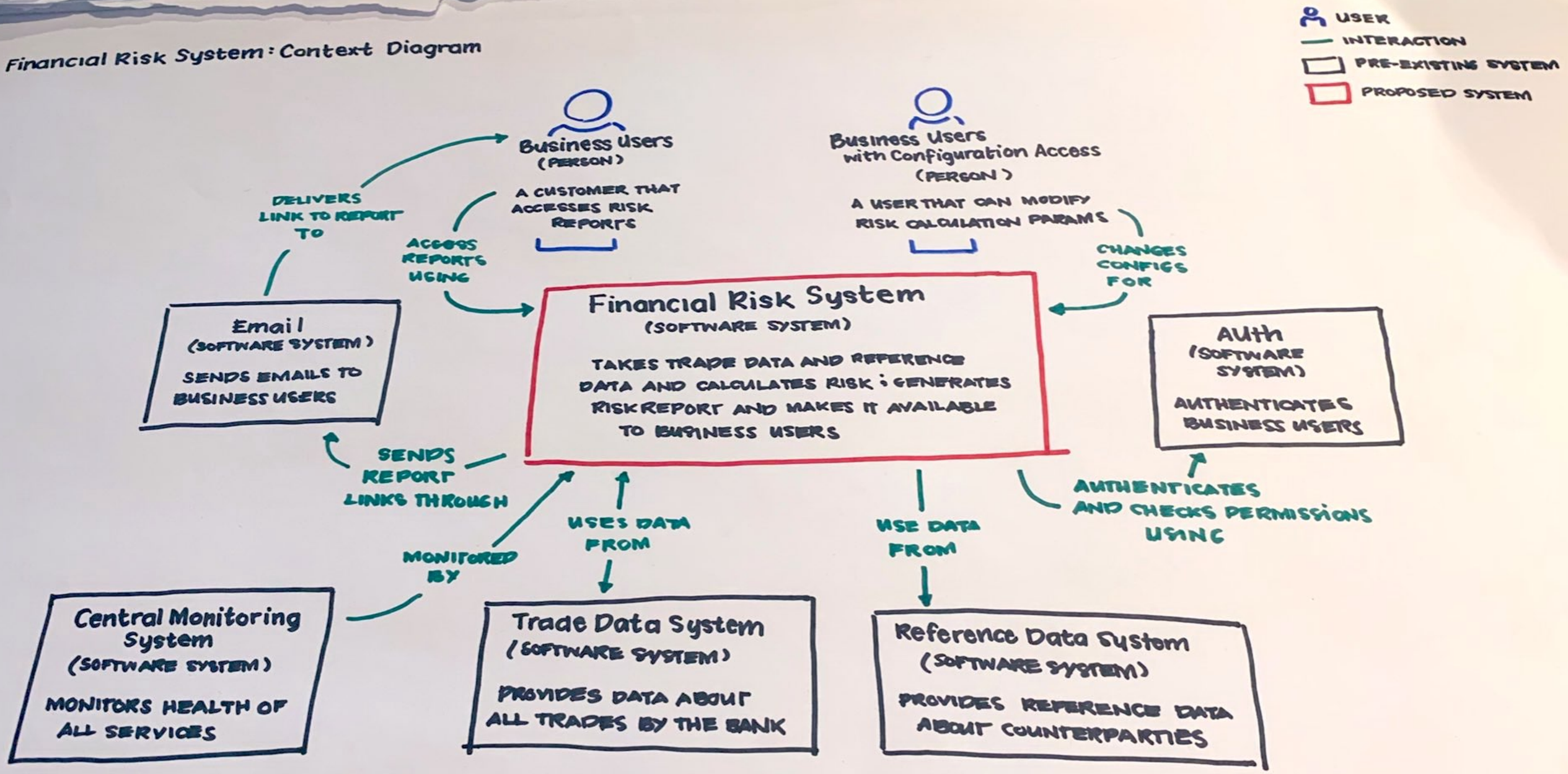
System Context diagram

What is the scope of the software system we’re building?

Who is using it? What are they doing?

What system integrations does it need to support?

Financial Risk System: Context Diagram

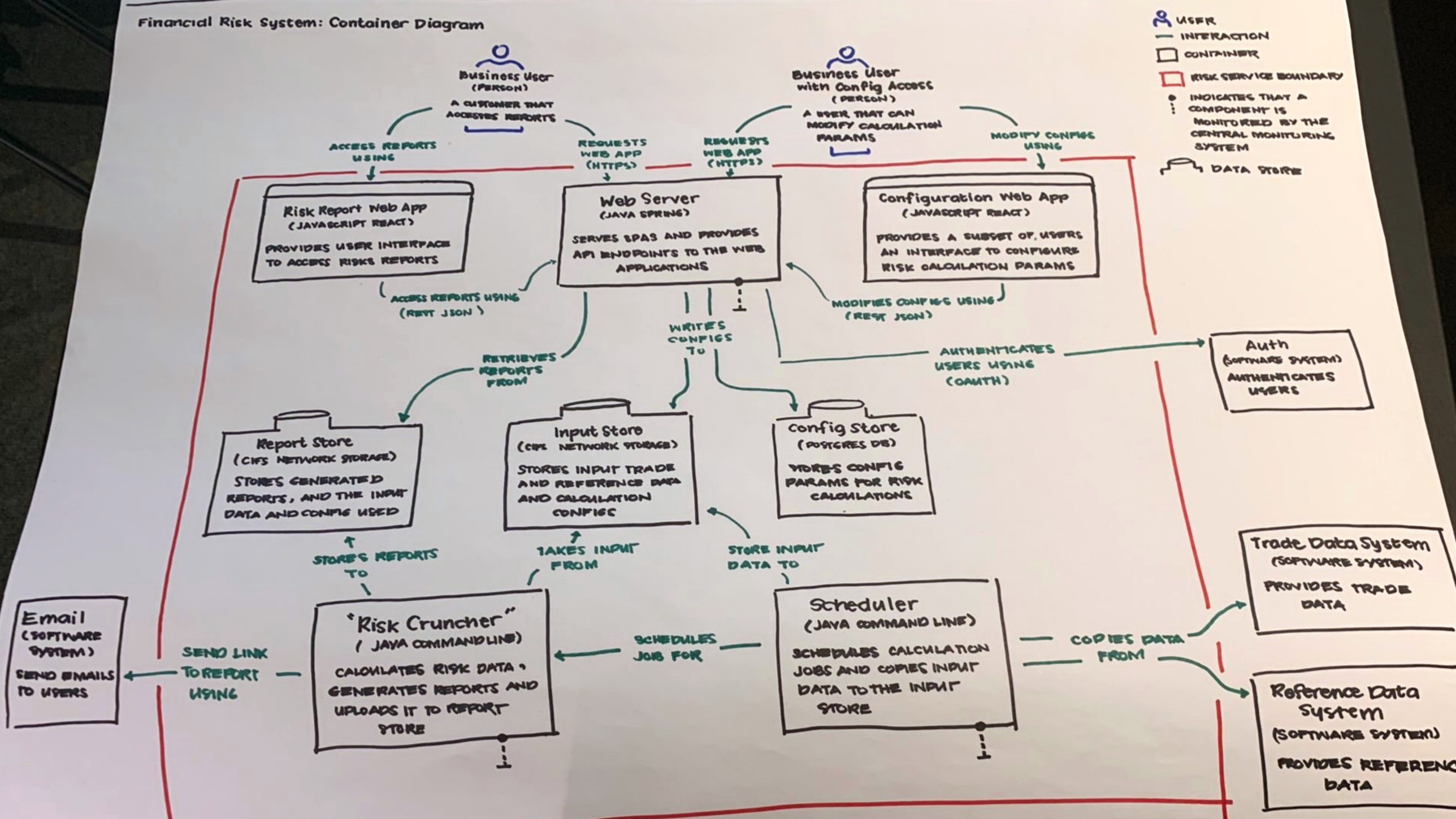


Container diagram

What are the major technology building blocks?

What are their responsibilities?

How do they communicate?


Financial Risk System: Container Diagram
USER
INTERACTION
CONTAINER
RISK SERVICE BOUNDARY
INDICATES THAT A COMPONENT IS MONITORED BY THE CENTRAL MONITORING SYSTEM
DATA STORE
Business User
(PERSON)
A CUSTOMER THAT ACCESSES REPORTS
Business User with Config Access
(PERSON)
A USER THAT CAN MODIFY CALCULATION PARAMS
ACCESS REPORTS USING
REQUESTS WEB APP (HTTPS)
REQUESTS WEB APP (HTTPS)
MODIFY CONFIGS USING
Risk Report Web App
(JAVASCRIPT REACT)
PROVIDES USER INTERFACE TO ACCESS RISK REPORTS
Web Server
(JAVA SPRING)
SERVES SPAS AND PROVIDES API ENDPOINTS TO THE WEB APPLICATIONS
Configuration Web App
(JAVASCRIPT REACT)
PROVIDES A SUBSET OF USERS AN INTERFACE TO CONFIGURE RISK CALCULATION PARAMS
ACCESS REPORTS USING (REST JSON)
MODIFIES CONFIGS USING (REST JSON)
WRITES CONFIGS TO
RETRIEVES REPORTS FROM
AUTHENTICATES USERS USING (OAUTH)
Auth
(SOFTWARE SYSTEM)
AUTHENTICATES USERS
Report Store
(CIFS NETWORK STORAGE)
STORES GENERATED REPORTS, AND THE INPUT DATA AND CONFIG USED
Input Store
(CIFS NETWORK STORAGE)
STORES INPUT TRADE AND REFERENCE DATA AND CALCULATION CONFIGS
Config Store
(POSTGRES DB)
STORES CONFIG PARAMS FOR RISK CALCULATIONS
STORES REPORTS TO
TAKES INPUT FROM
STORE INPUT DATA TO
Email
(SOFTWARE SYSTEM)
SEND EMAILS TO USERS
SEND LINK TO REPORT USING
"Risk Cruncher"
(JAVA COMMANDLINE)
CALCULATES RISK DATA, GENERATES REPORTS AND UPLOADS IT TO REPORT STORE
SCHEDULES JOB FOR
Scheduler
(JAVA COMMAND LINE)
SCHEDULES CALCULATION JOBS AND COPIES INPUT DATA TO THE INPUT STORE
COPIES DATA FROM
Trade Data System
(SOFTWARE SYSTEM)
PROVIDES TRADE DATA
Reference Data System
(SOFTWARE SYSTEM)
PROVIDES REFERENC
DATA

The diagrams should spark **meaningful questions**

No

“What does that arrow mean?”

“Why are some boxes red?”

“Is that a Java application?”

“Is that a monolithic application, or a collection of microservices?”

“How do the users get their reports?”

Yes

“What protocol are your two Java applications using
to communicate with each other?”
“Why do you have two separate C# applications instead of one?”
“Why are you using MongoDB?”
“Why are you using MySQL when our standard is Oracle?”
“Should we really build new applications with .NET Framework
rather than .NET Core?”

Richer diagrams lead to
richer **design discussions**

Our developers are constantly revisiting past decisions

Title These documents have names that are short noun phrases. For example, "ADR 1: Deployment on Ruby on Rails 3.0.10" or "ADR 9: LDAP for Multitenant Integration"

Context This section describes the forces at play, including technological, political, social, and project local. These forces are probably in tension, and should be called out as such. The language in this section is value-neutral. It is simply describing facts.

Decision This section describes our response to these forces. It is stated in full sentences, with active voice. "We will ..."

Status A decision may be "proposed" if the project stakeholders haven't agreed with it yet, or "accepted" once it is agreed. If a later ADR changes or reverses a decision, it may be marked as "deprecated" or "superseded" with a reference to its replacement.

Consequences This section describes the resulting context, after applying the decision. All consequences should be listed here, not just the "positive" ones. A particular decision may have positive, negative, and neutral consequences, but all of them affect the team and project in the future.

“Architecture Decision Record”

A short description of an architecturally significant decision

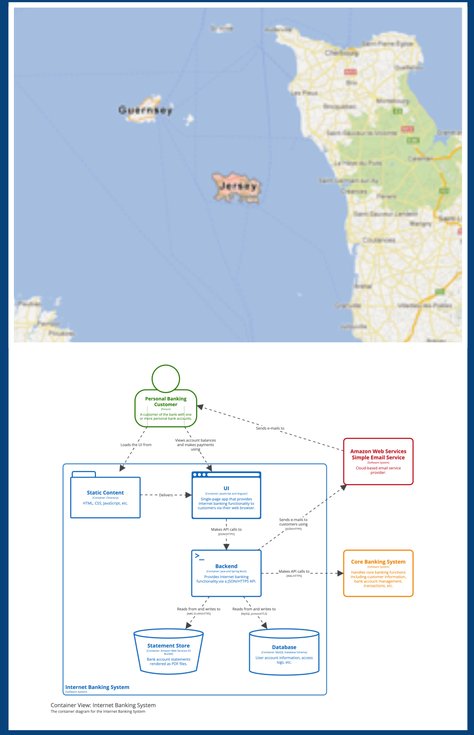
http://thinkrelevance.com/blog/2011/11/15/documenting-architecture-decisions (Michael Nygard)

Architecture isn’t only for architects

The value of a common language shouldn't be underestimated!

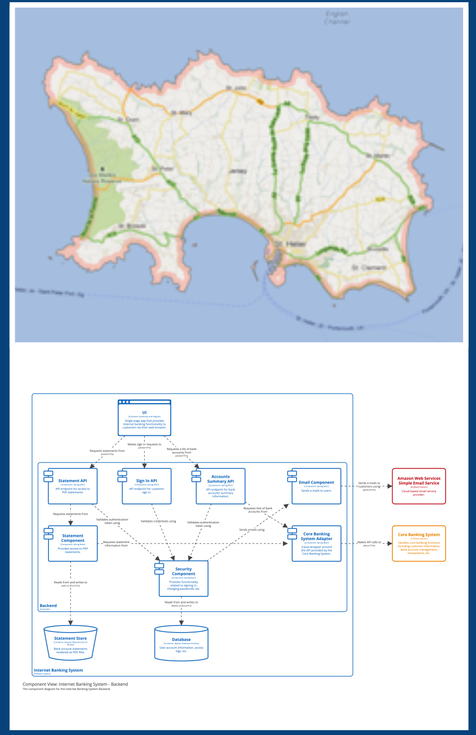
System Context View: Internet Banking System

The system context diagram for a fictional Internet Banking System



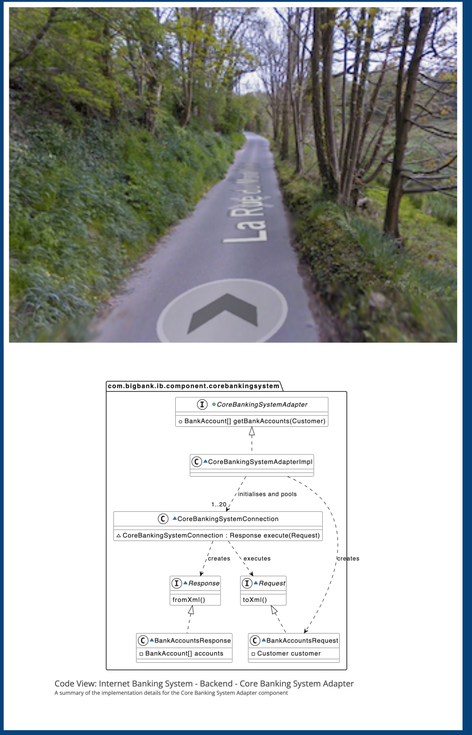
Container View: Internet Banking System

The container diagram for the Internet Banking System



Component View: Internet Banking System - Backend

The component diagram for the Internet Banking System backend



Code View: Internet Banking System - Backend - Core Banking System Adapter

A summary of the implementation details for the Core Banking System Adapter component

Diagrams are maps

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

We still have a gap between
dev and ops because
they speak different languages

Our product owners, testers, domain experts, etc don’t fully understand what the engineers are building

The UX lead

show me your system context
and container diagrams
[for this feature/change]

Product Owner

show me which C4 containers
you are changing
[for this feature/change]

Product Owner

Takeaways

The backlash against “architects” is real, but the technical leadership role never went away

Technical leadership is essential for all software engineering teams, irrespective of whether this is performed by "architects" or not

Many software teams are still missing some of the engineering fundamentals that prevent them from becoming high performing teams

DDD, agile, Spotify model, Team Topologies, feature teams, Conway's Law, platform teams, autonomy, trust, and other socio-technical techniques

will fail without

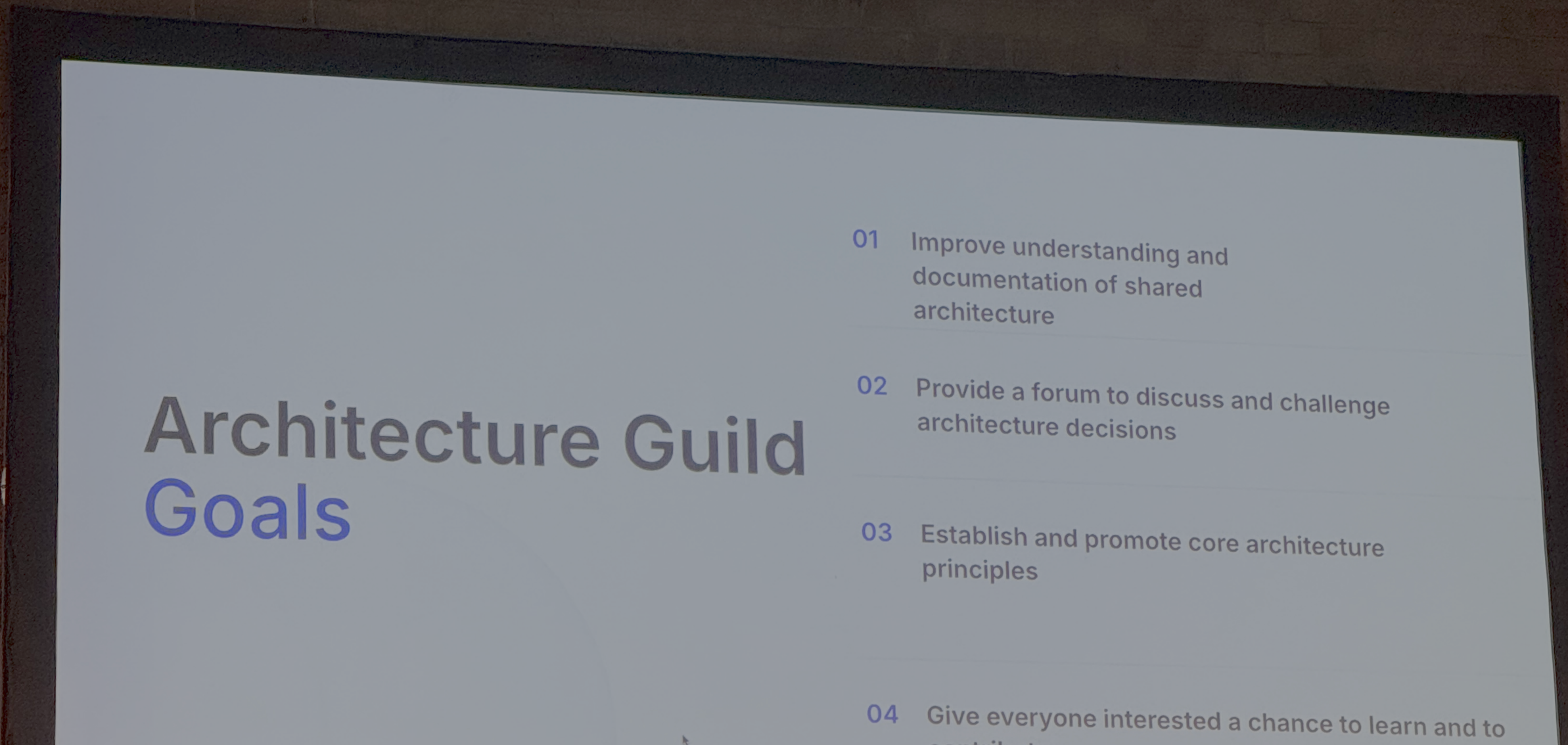
good technical foundations

and effective communication

You can’t optimise
team performance
without good
technical foundations

Improving architectural capability and maturity

Improve understanding and documentation of shared architecture



The C4 model provides
a common language for
technical and non-technical
team members
to communicate effectively

ARCHITECTURE & DESIGN

Navigating Software Architecture at Scale: Insights from Decathlon's Architecture Process

JUL 24, 2024 • 6 MIN READ

LIKE

by

Eran Stiller FOLLOW

Chief Software Architect

Raphaël Tahar, staff engineer at Decathlon, recently published his insights from co-leading an architecture process at scale. In a 4-part blog post series, Tahar depicts how, by combining methodologies like architecture committees, the C4 model, and System Thinking and emphasizing the importance of ADRs and centralized documentation, Decathlon ensures its teams are well-equipped to make informed, strategic decisions.

He is part of a group supporting over 120 engineers across 23 feature teams comprising one domain out of 1500+ engineers globally at Decathlon. Supporting this scale of developers is no small feat and involves providing support with designing new systems, optimizing existing ones, and ensuring alignment with global guidelines. To tackle this, Decathlon established an architecture committee, which plays a crucial role in guiding teams through the intricate decision-making process.

RELATED CONTENT

Java News Roundup: Jakarta EE 11 and Spring AI Updates, WildFly 36 Beta, Infinispan, JNoSQL
MAR 31, 2025

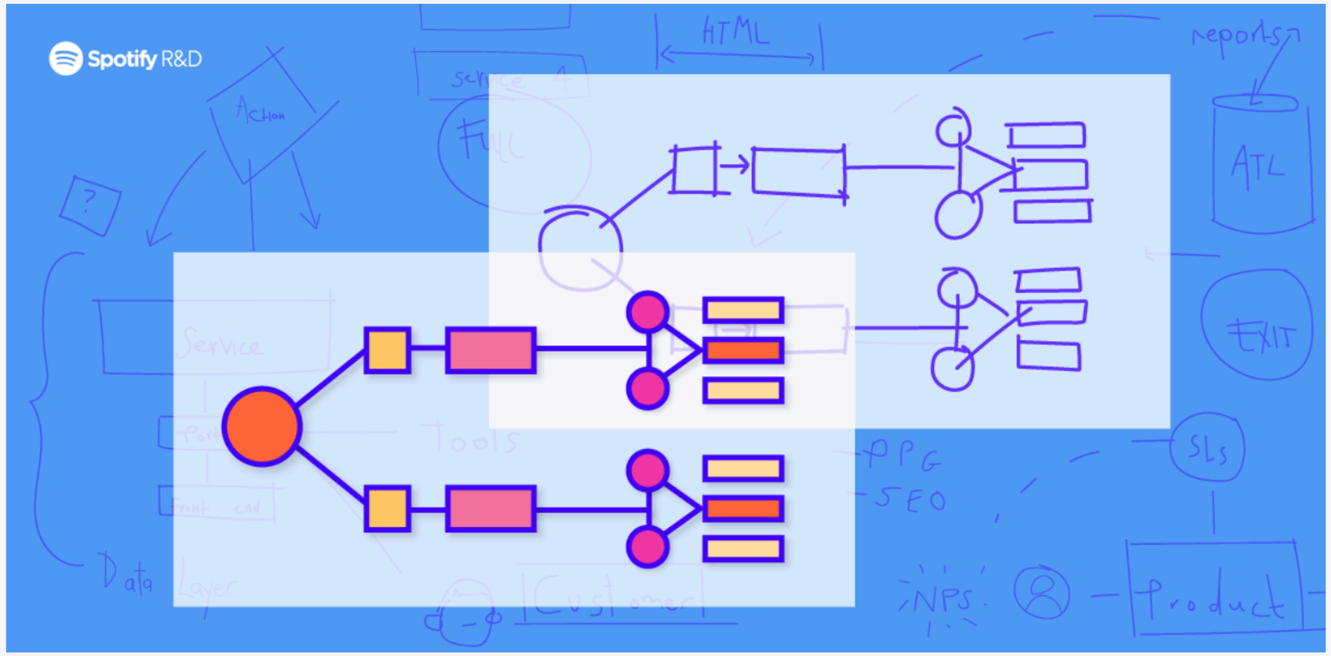
Google DeepMind Launches TxGemma: Advancing AI-Driven Drug Discovery and Development
MAR 30, 2025

ASP.NET Core 10 Preview 2 Streamlines Blazor Navigation, Updates OpenAPI
MAR 30, 2025

Atlassian Announces Opsgenie Consolidation into JIRA Service Management
MAR 29, 2025

Software Visualization — Challenge, Accepted

July 25, 2022
Published by Renato Kalman (Sr. Engineer) and Johan Wallin (Staff Engineer)



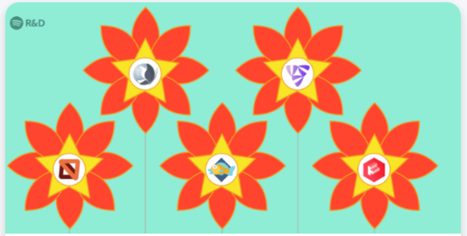
TL;DR Architectural diagrams are the bread and butter of software design and a foundational tool for communication and collaboration on software development. At Spotify, we have an incredibly complex network of thousands of interlinked software systems owned by hundreds of teams, so having a simple way to visualize these connections is essential. While capturing all of our software in one large diagram is technically possible, it would be very hard to understand and navigate. We need tools to

CATEGORIES

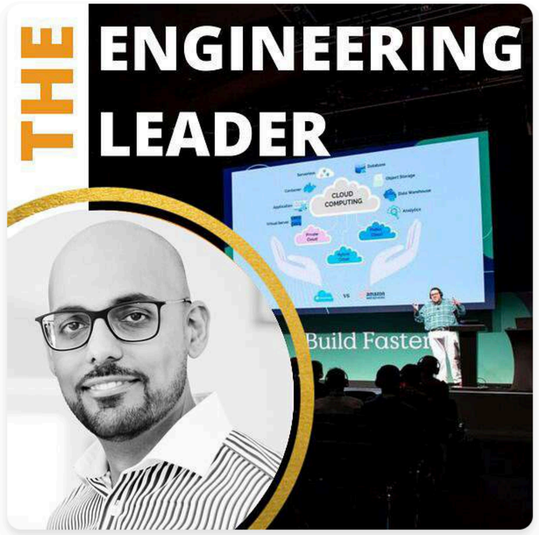
Developer Tools Platform

Related articles

An Insider's Tips for Taking the Certified Backstage Associate (CBA) Exam



Congratulations to the Recipients of the 2024 Spotify FOSS Fund



The Engineering Leader is an unscripted exploration of how to build great software Engineering Teams and drive effective Digital Transformation....
Show More

All Episodes

The Engineering Leader

A Moment on C4 Architectural Design with Amar Mehan

May 19, 2022 • Steve Westgarth • Season 1 • Episode 23

Share

15 30 1× 00:00 | 02:57

Show Notes

In this moment Amar Mehan, Lead Technologist at Boots describes how the C4 Model revolutionised the Boots approach to Architectural Design.For more information about C4 check out http://c4model.com

People on this episode

Steve Westgarth
Host

Follow

Apple Podcasts
Spotify
Amazon Music
RSS Feed

See All

PETE WILLIAMS

Rebuilding an Architecture Practice via DDD

Your software engineers today
are your technical leaders
of tomorrow

Thank you!

Simon Brown