Introduction to Software Architecture

17-313 Fall 2025

Foundations of Software Engineering

https://cmu-17313q.github.io

Eduardo Feo Flushing

Administrivia

- P2B due Sunday September 28th, 11:59PM
- Team Surveys due every Sunday, 11:59PM
  - “Storming” phase
  - Most teams doing well
  - Remember: communication, communication, …

Communication

Communication

Co

You can’t solve any problem without communication!

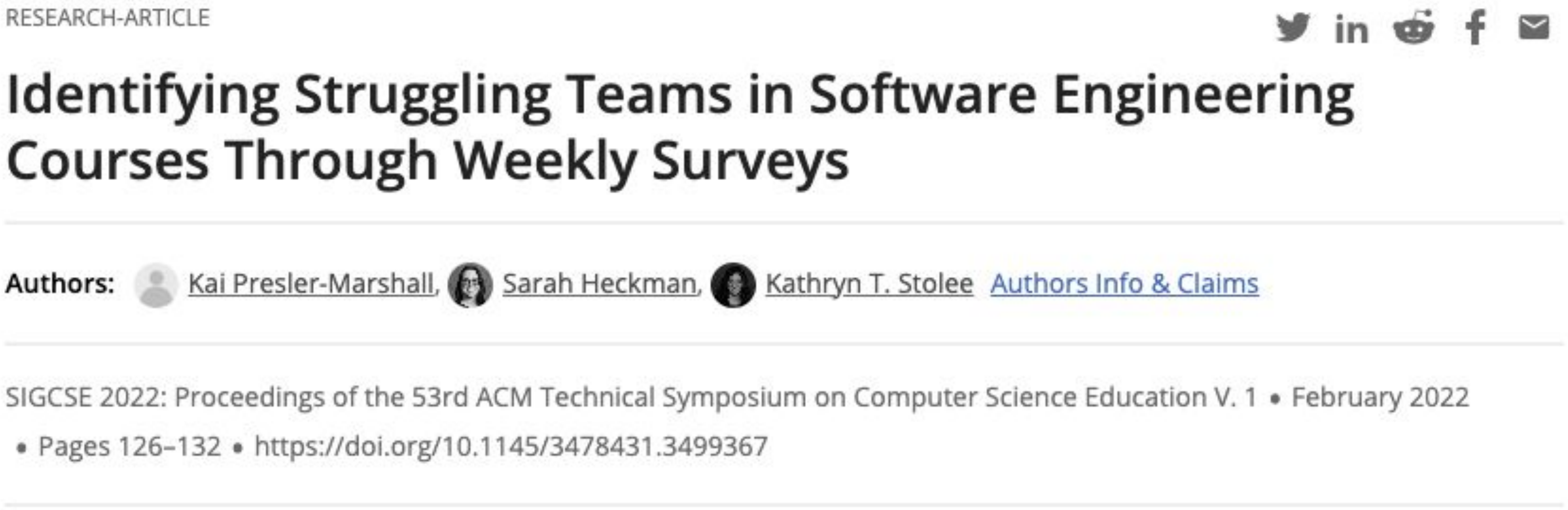
Co

on

Conflict Resolution

- Your goal: Find a solution to the problem and move forward.
- Make sure that everybody works from the same set of facts.
- Establish ground rules for your team's discussion.
  - Talk about how the situation made you feel. Never presume anything about anyone else.
- Remain calm and rational. If you feel triggered or threatened, extract yourself from the situation, wait an hour to chill out, and then try again.
- If you reach an impasse, talk to your team leader.
- If your team remains in conflict, escalate to your mentor CA.
  - Your mentor CA *will not solve* your problem. They will help *you* to solve your own problems.

Team survey



RESEARCH-ARTICLE

Identifying Struggling Teams in Software Engineering Courses Through Weekly Surveys

Authors: Kai Presler-Marshall, Sarah Heckman, Kathryn T. Stolee Authors Info & Claims

SIGCSE 2022: Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1 • February 2022 • Pages 126–132 • https://doi.org/10.1145/3478431.3499367

Smoking Section

- Last **two** full rows

Learning Goals

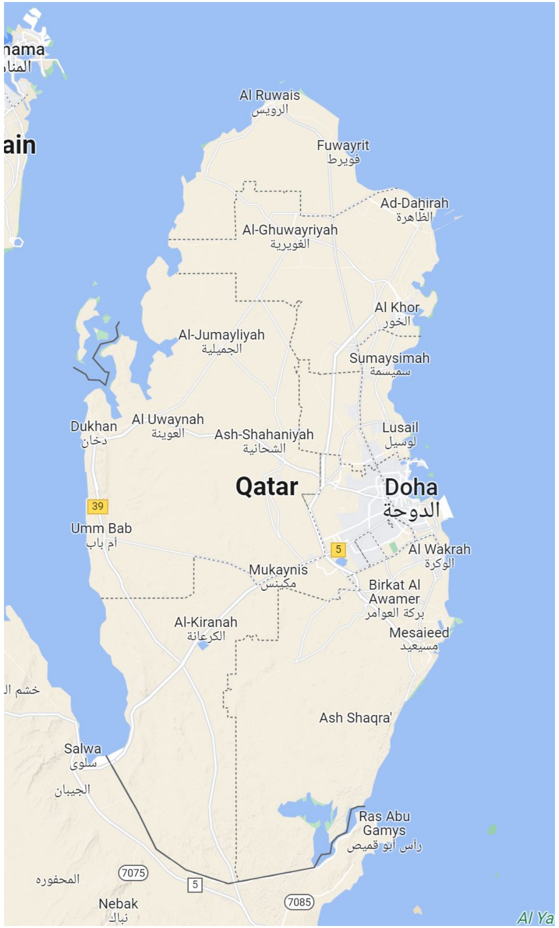
- Understand the abstraction level of architectural reasoning
- Appreciate how software systems can be viewed at different abstraction levels
- Distinguish software architecture from (object-oriented) software design
- Explain the importance of architectural decisions
- Integrate architectural decisions into the software development process
- Document architectures clearly, without ambiguity

Outline

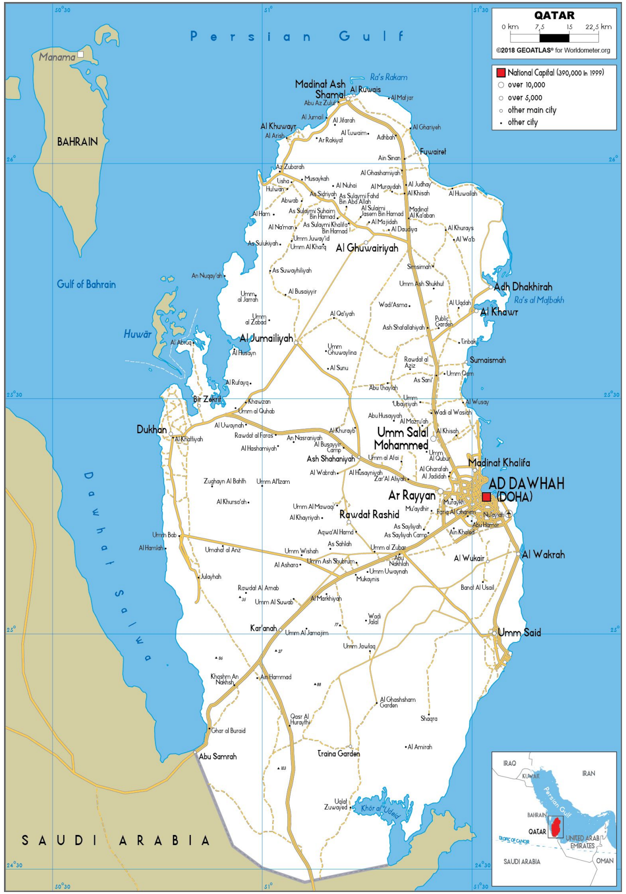
- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- Architecting software
  - Integrating Architectural Decisions into the SW Development Process
  - Common Software Architectures
  - Documentation

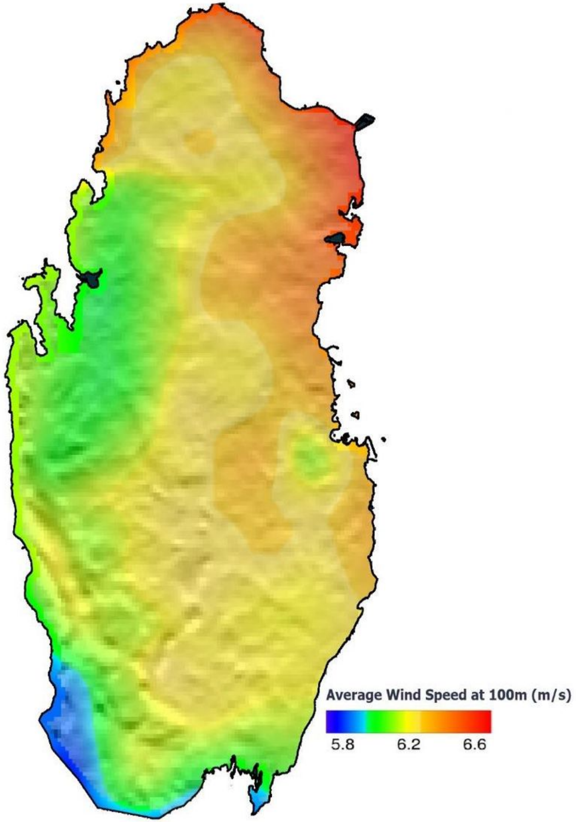
Outline

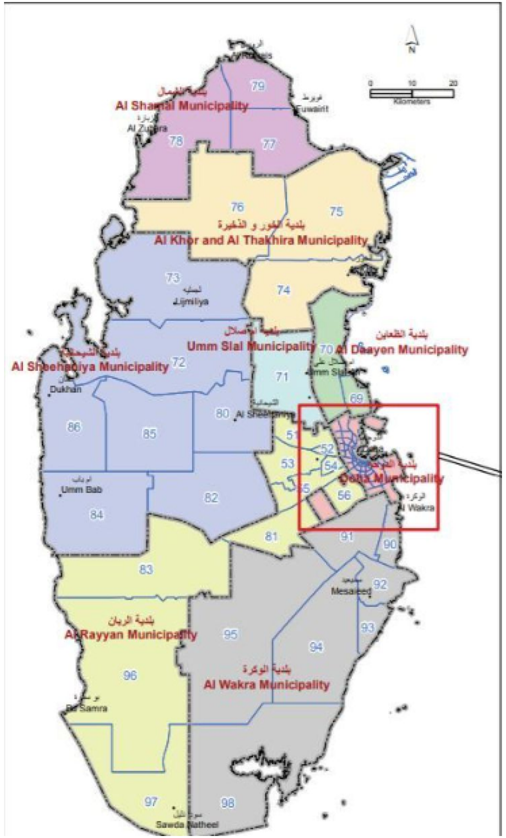
- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- Architecting software
  - Integrating Architectural Decisions into the SW Development Process
  - Common Software Architectures
  - Documentation



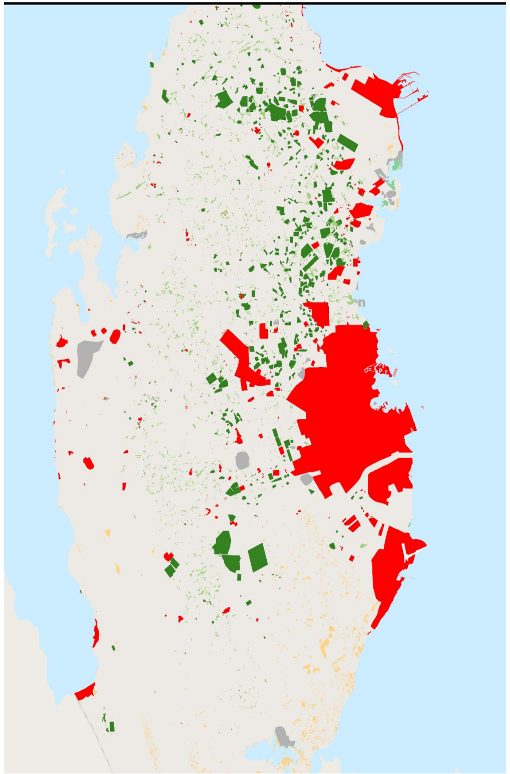
nama
المنا
ain
Al Ruwais
الرويس
Fuwayrit
فويرط
Ad-Dahirah
الظاهرة
Al-Ghuwayriyah
الغويرية
Al Khor
الخور
Al-Jumayliyah
الجميلية
Sumaysimah
سميسمة
Dukhan
دخان
Al Uwaynah
العوينة
Ash-Shahaniyah
الشحانية
Lusail
لوسيل
Qatar
Doha
الدوحة
39
Umm Bab
أم باب
5
Al Wakrah
الوكرة
Mukaynis
مكينس
Birkat Al Awamer
بركة العوامر
Al-Kiranah
الكرعانة
Mesaieed
مسيعيد
خشم ال
Ash Shaqra'
Salwa
سلوى
الجيبان
Ras Abu Gamys
راس أبو قميص
7075
المحفوره
5
Nebak
نباك
7085
Al Ya


QATAR
0 km 7,5 15 22,5 km
©2018 GEOATLAS® for Worldometer.org
National Capital (390,000 in 1999)
over 10,000
over 5,000
other main city
other city
Persian Gulf
Manama
BAHRAIN
Gulf of Bahrain
Huwār
Dawhat Salwa
SAUDI ARABIA
Ra's Rakam
Madinat Ash Shamal
Al Ruwais
Al Khuwayr
Fuwairet
Az Zubarah
Al Ghuwairiyah
Adh Dhakhirah
Ra's al Mafbakh
Al Khawr
Al Jumailiyah
Sumaismah
Bir Zekrit
Dukhan
Umm Salal Mohammed
Ash Shahaniyah
Madinat Khalifa
AD DAWHAH (DOHA)
Ar Rayyan
Rawdat Rashid
Al Wukair
Al Wakrah
Umm Said
Kar'anah
Abu Samrah
Traina Garden
Khor al 'Udeid
Al Ghashsham Garden
Shaqra
Al Amirah
IRAQ
KUWAIT
IRAN
Persian Gulf
BAHRAIN
QATAR
UNITED ARAB EMIRATES
TROPIC OF CANCER
SAUDI ARABIA
OMAN
50°30
51°
51°30
26°
25°30
25°
24°30


Average Wind Speed at 100m (m/s)
5.8
6.2
6.6


N
0 10 20
Kilometers
Al Shamal Municipality
Al Zubara
Fuwairit
79
78
77
76
75
بلدية الظعاين
Al Khor and Al Thakhira Municipality
73
Lijmiliya
74
Umm Slal Municipality
70
Al Daayen Municipality
Umm Slal Ali
Al Sheehaniya Municipality
72
Dukhan
71
69
80
Al Sheehaniya
86
85
51
52
53
54
Doha Municipality
55
56
Al Wakra
Umm Bab
84
82
81
91
90
83
92
Mesaieed
93
بلدية الريان
Al Rayyan Municipality
95
94
بلدية الوكرة
Al Wakra Municipality
96
Bu Samra
97
98
Sawda Natheel

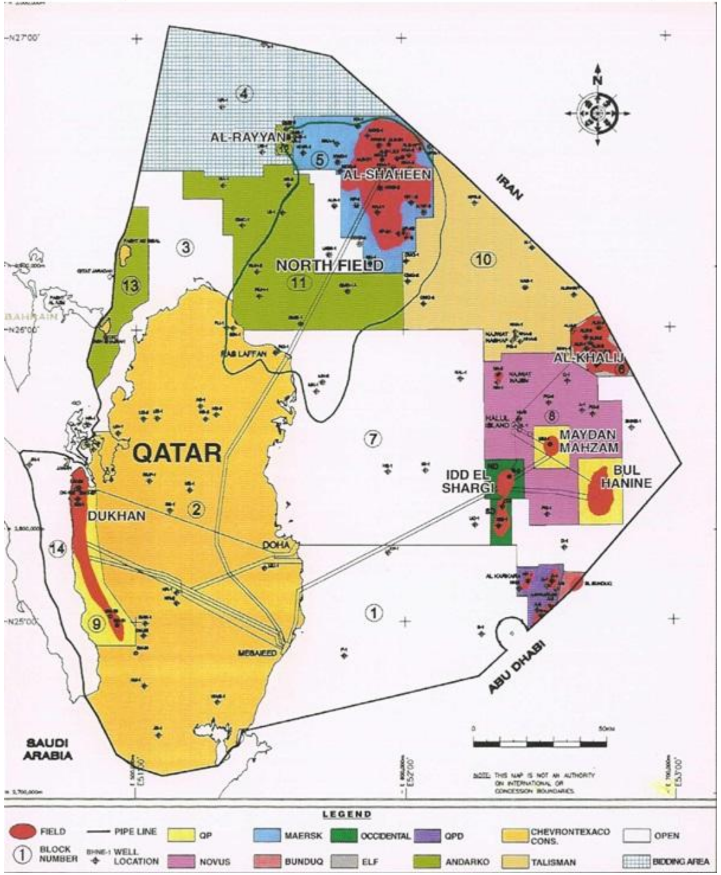
LandCover 2020



LandCover 2020

🟥	Settlements/Built-up	🟩	Shrubs/Rawada	⬜	Barren Land
🟩	Farms/Agriculture	🟩	Mangroves/ Forests	⬜	Others
		🟨	Sand/Sand Dunes		

https://www.qnb.co.id/assets/62378773_322_Qatar%20Economic%20Review%202010.pdf



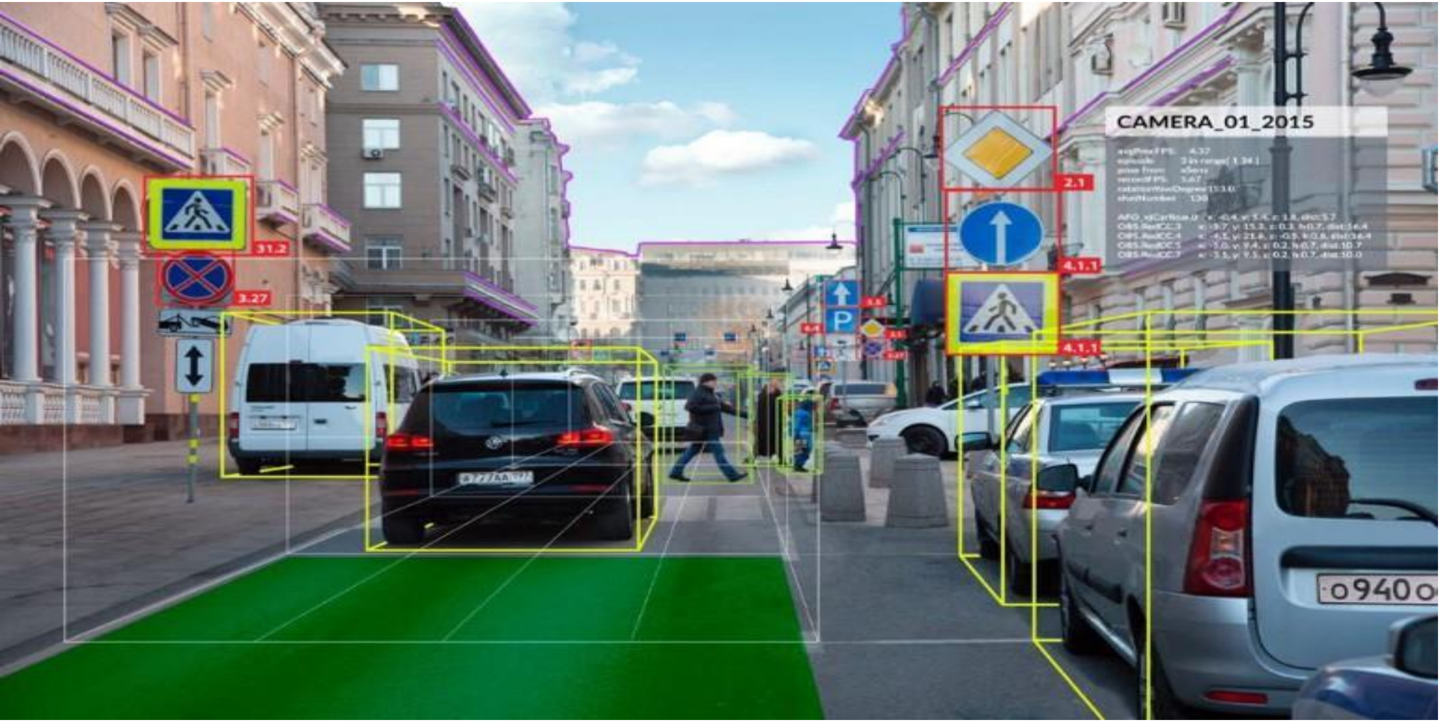
Abstracted views focus on conveying specific information

- They have a well-defined **purpose**
- Show **only necessary** information
- **Abstract** away unnecessary details
- Use legends/annotations to **remove ambiguity**
- **Multiple views** of the same object tell a larger story

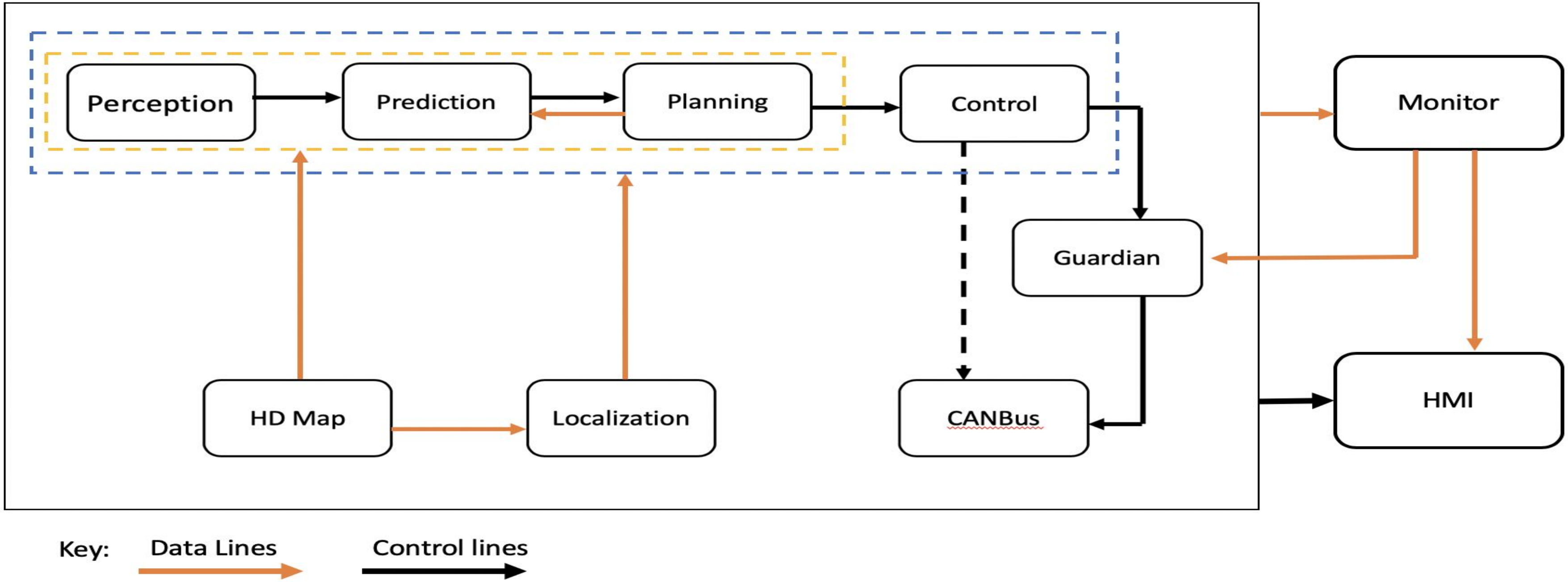
Outline

- Views and Abstraction
- **Case Study: Autonomous Vehicles**
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- Architecting software
  - Integrating Architectural Decisions into the SW Development Process
  - Common Software Architectures
  - Documentation

Case Study: Autonomous Vehicle Software

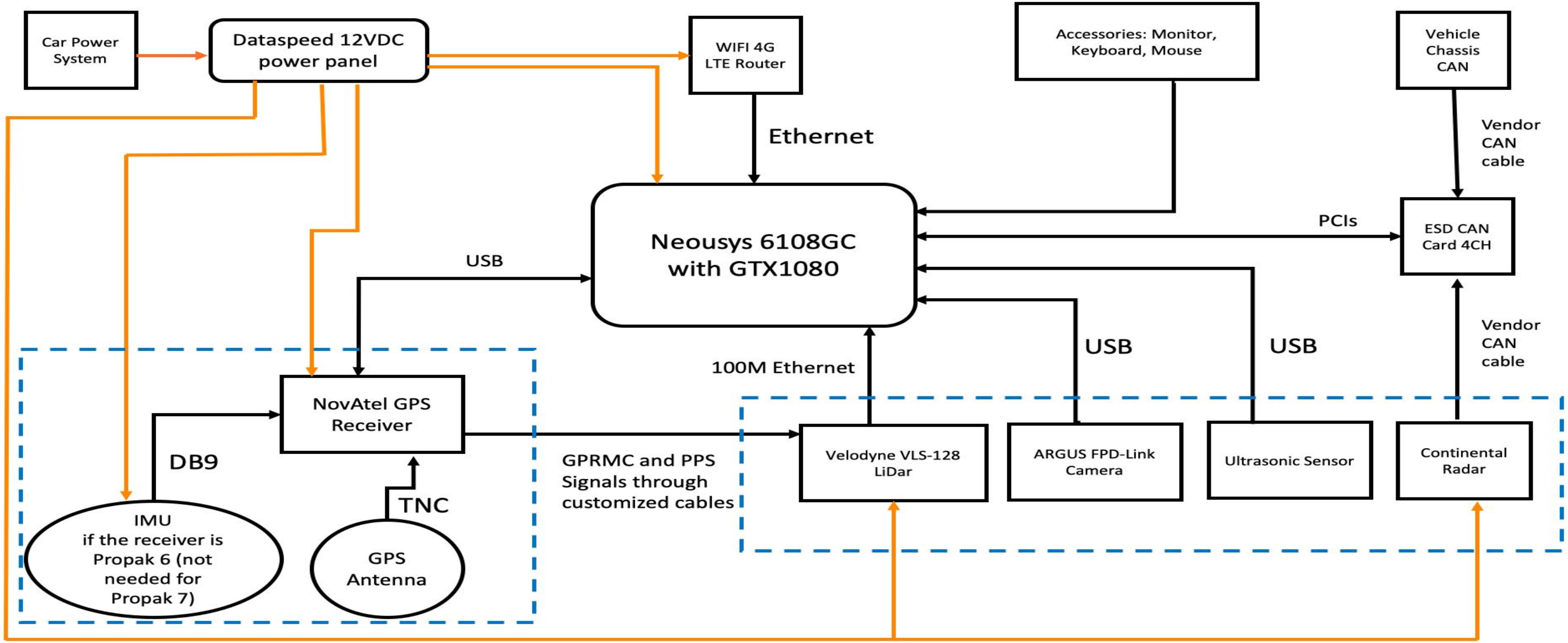


Apollo Software Architecture



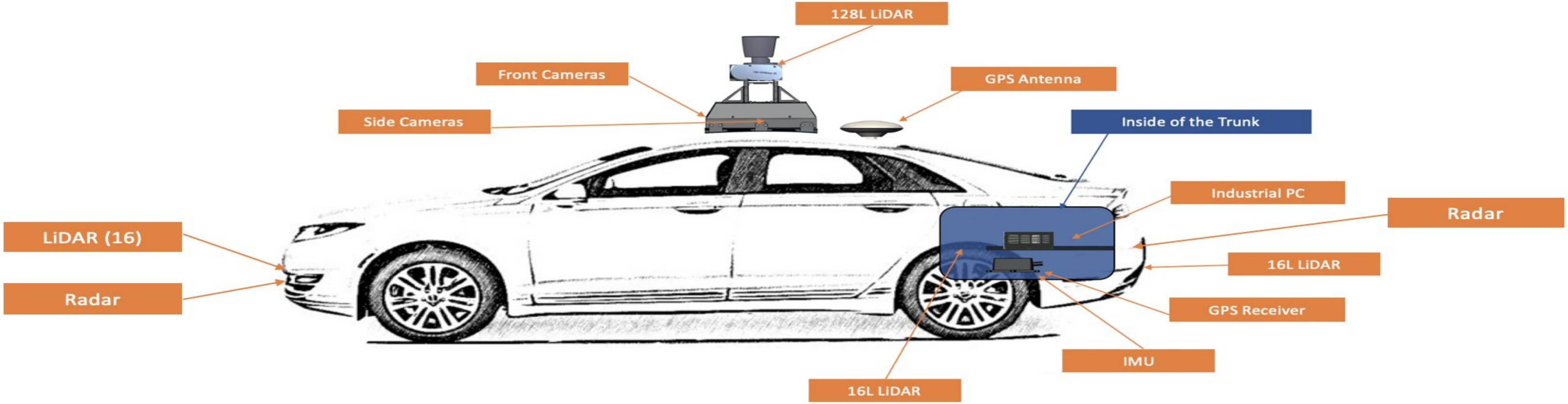
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/docs/specs/Apollo_5.5_Software_Architecture.md

Apollo Hardware Architecture



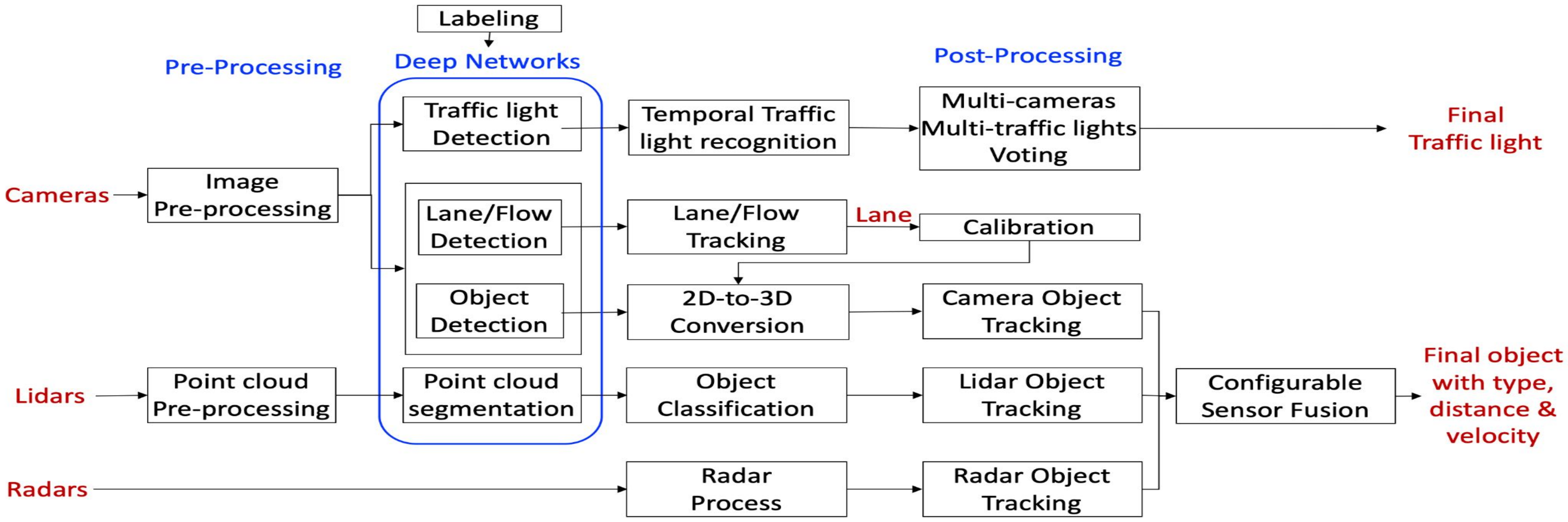
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Hardware/Vehicle Overview

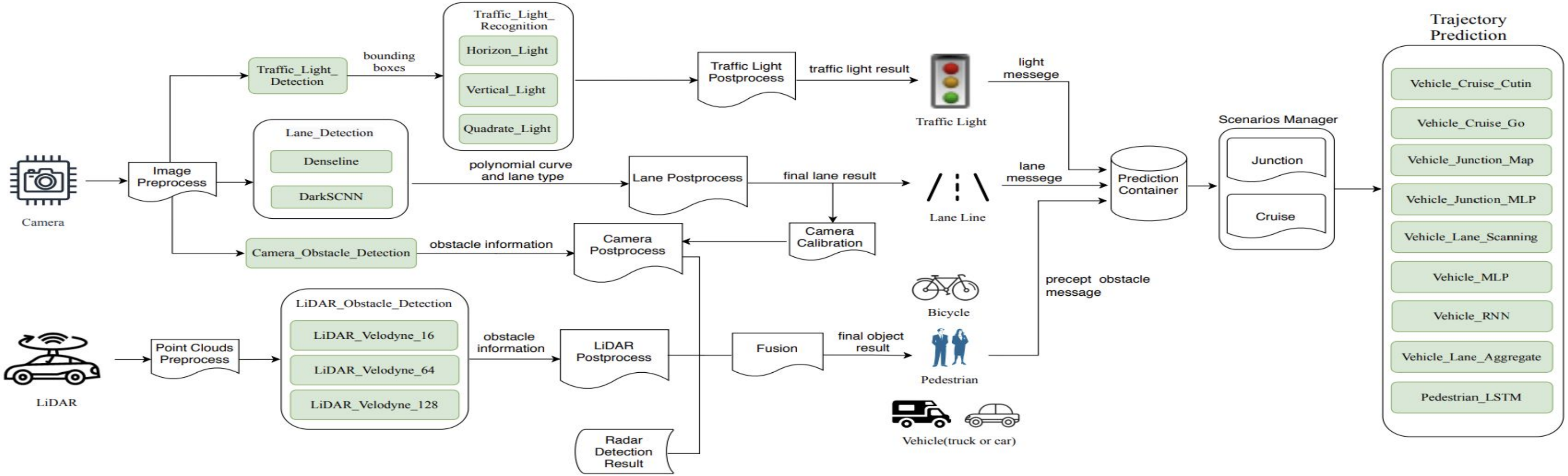


Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Perception Module

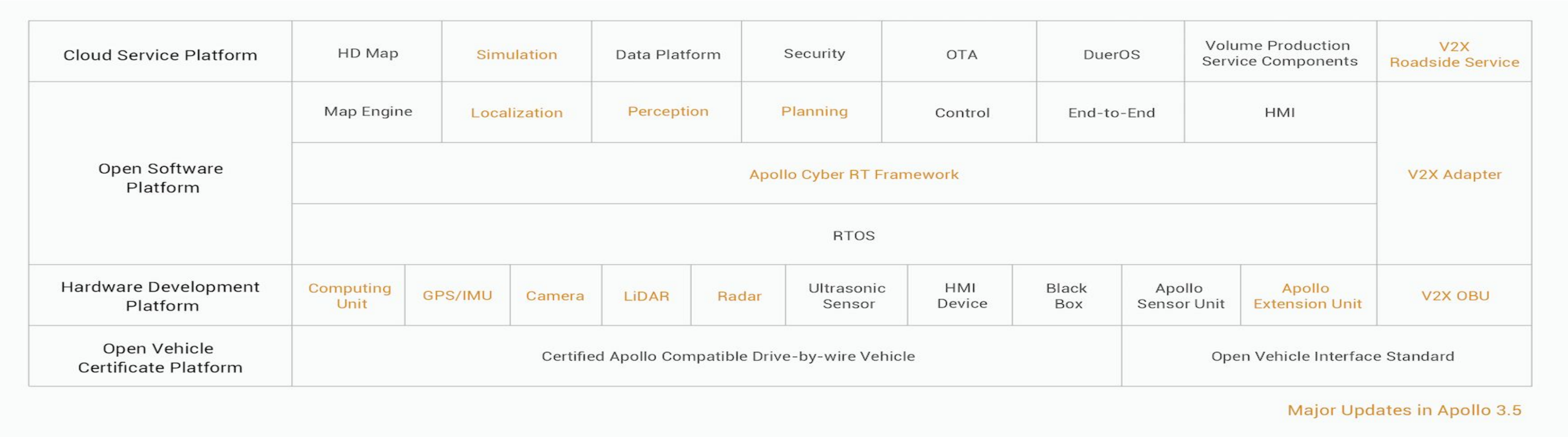


Apollo ML Models



Source: Zi Peng, Jinqiu Yang, Tse-Hsun (Peter) Chen, and Lei Ma. 2020. A First Look at the Integration of Machine Learning Models in Complex Autonomous Driving Systems: A Case Study on Apollo. In Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20), https://doi.org/10.1145/ 3368089.3417063

Apollo Software Stack

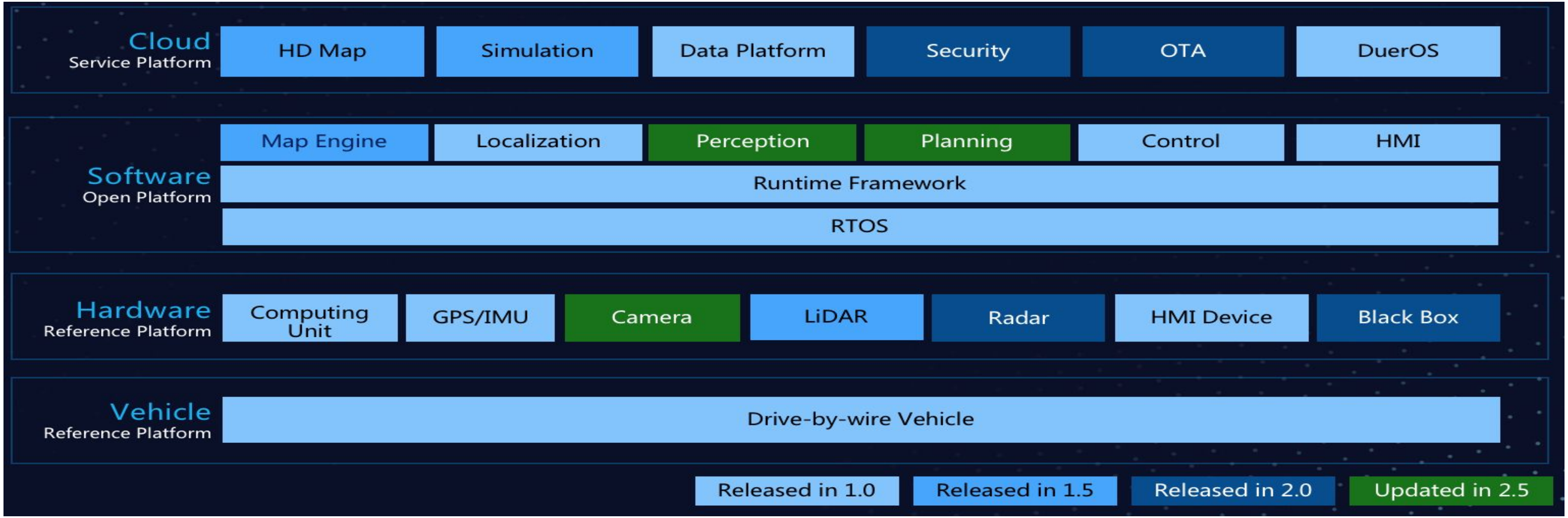


Cloud Service Platform	HD Map	Simulation	Data Platform	Security	OTA	DuerOS	Volume Production Service Components	V2X Roadside Service			
Open Software Platform	Map Engine	Localization	Perception	Planning	Control	End-to-End	HMI	V2X Adapter			
	Apollo Cyber RT Framework										
	RTOS										
Hardware Development Platform	Computing Unit	GPS/IMU	Camera	LiDAR	Radar	Ultrasonic Sensor	HMI Device	Black Box	Apollo Sensor Unit	Apollo Extension Unit	V2X OBU
Open Vehicle Certificate Platform	Certified Apollo Compatible Drive-by-wire Vehicle	Open Vehicle Interface Standard									

Major Updates in Apollo 3.5

Source: https://github.com/ApolloAuto/

Feature Evolution (Software Stack View)



Cloud Service Platform: HD Map | Simulation | Data Platform | Security | OTA | DuerOS

Software Open Platform: Map Engine | Localization | Perception | Planning | Control | HMI

Runtime Framework

RTOS

Hardware Reference Platform: Computing Unit | GPS/IMU | Camera | LiDAR | Radar | HMI Device | Black Box

Vehicle Reference Platform: Drive-by-wire Vehicle

Released in 1.0 | Released in 1.5 | Released in 2.0 | Updated in 2.5

Source: https://github.com/ApolloAuto/apollo

Case Study: Apollo

Check out the **“side pass”** feature from the video:
https://www.youtube.com/watch?v=BXNDUtNZdM4

- Which **modules or components** are involved in enabling the side pass feature?

Other resources:

- **Source: https://github.com/ApolloAuto/apollo**
- **Doxygen: https://hidetoshi-furukawa.github.io/apollo-doxygen/index.html**

Outline

- Views and Abstraction
- Case Study: Autonomous Vehicles
- **Software Architecture**
  - **Definitions, Importance**
  - **Software Design vs. Software Architecture**
- Architecting software
  - Integrating Architectural Decisions into the SW Development Process
  - Common Software Architectures
  - Documentation

Software Architecture

“Architecture is about the important stuff. Whatever that is.”

Ralph Johnson

design

Editor: Martin Fowler ■ ThoughtWorks ■ fowler@acm.org

Who Needs an Architect?

Martin Fowler

Wandering down our corridor a while ago, I saw my colleague Dave Rice in a particularly grumpy mood. My brief question caused a violent statement, “We shouldn’t interview anyone who has ‘architect’ on his resume.” At first blush, this was an odd turn of phrase, because we usually introduce Dave as one of our leading architects.

The reason for his title schizophrenia is the fact that, even by our industry’s standards, “architect” and “architecture” are terribly overloaded words. For many, the term “software architect” fits perfectly with the smug controlling image at the end of *Matrix Reloaded*. Yet even in firms that have the greatest contempt for that image, there’s a vital role for the technical leadership that an architect such as Dave plays.

What is architecture?

When I was fretting over the title for *Patterns of Enterprise Application Architecture* (Addison-Wesley, 2002), everyone who reviewed it agreed that “architecture” belonged

chitect.) However, as so often occurs, inside the blighted cynicism is a pinch of truth. Understanding came to me after reading a posting from Ralph Johnson on the Extreme Programming mailing list. It’s so good I’ll quote it all.

A previous posting said

> The RUP, working off the IEEE definition, defines architecture as “the highest level concept of a system in its environment. The architecture of a software system (at a given point in time) is its organization or structure of significant components interacting through interfaces, those components being composed of successively smaller components and interfaces.”

Johnson responded:

> I was a reviewer on the IEEE standard that used that, and I argued uselessly that this was clearly a completely bogus definition. There is no highest level concept of a system. Customers have a different concept than developers. Customers do not care at all about the structure of significant components. So, perhaps an architecture is the highest level concept that developers have of a system in its environment. Let’s forget the devel-

Software Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them.

[Bass et al. 2003]

Note: this definition is ambivalent to whether the architecture is known or whether it's any good!

Elements of Software Architecture

- Abstraction
- Elements: roles, responsibilities, behaviors, properties
- Relationships between elements
- Relationships to non-software elements
  - Hardware, external systems
- Described from many different “external” perspectives
  - Hides “internal” details

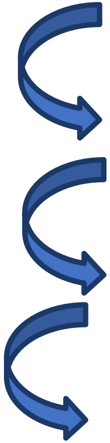
Software Architecture: Motivation

- Facilitates internal and external communication
- Describes design decisions and prescribes implementation constraints
- Relates to organizational structure
- **Permits/precludes achieving non-functional requirements**
- Allows to control complexity, manage change, and to (better) estimate effort

Architecting Software the SEI Way - Software Architecture Fundamentals: Technical, Business, and Social Influences. Robert Wojcik. 2012

Software Design vs. Architecture

Levels of Abstraction



- Requirements
  - high-level “what” needs to be done
- Architecture (High-level design)
  - high-level “how”, mid-level “what”
- OO-Design (Low-level design, e.g. design patterns)
  - mid-level “how”, low-level “what”
- Code
  - low-level “how”

Design vs. Architecture

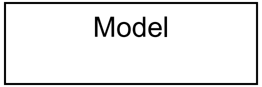
Design Questions

- How do I add a menu item in NodeBB?
- How can I make it easy to create posts in NodeBB?
- What lock protects this data?
- How does Google rank pages?
- What encoder should I use for secure communication?
- What is the interface between objects?

Architectural Questions

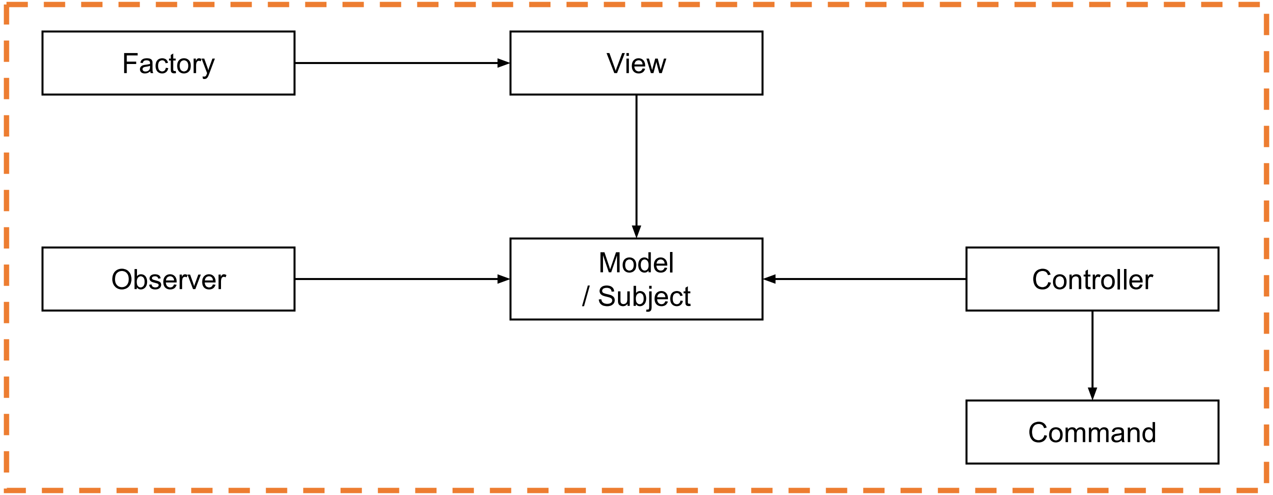
- How do I extend NodeBB with a plugin?
- What threads exist and how do they coordinate?
- How does Google scale to billions of hits per day?
- Where should I put my firewalls?
- What is the interface between subsystems?

Objects

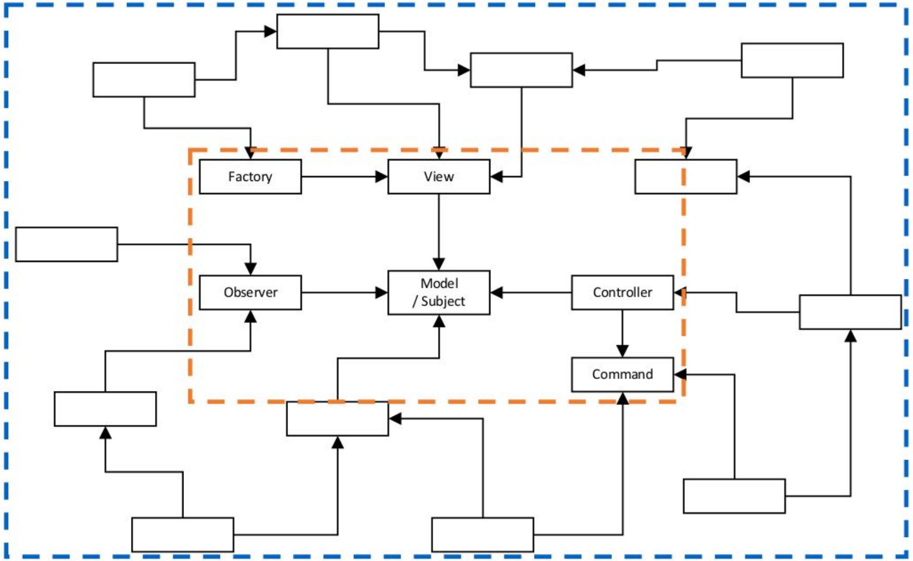


Model

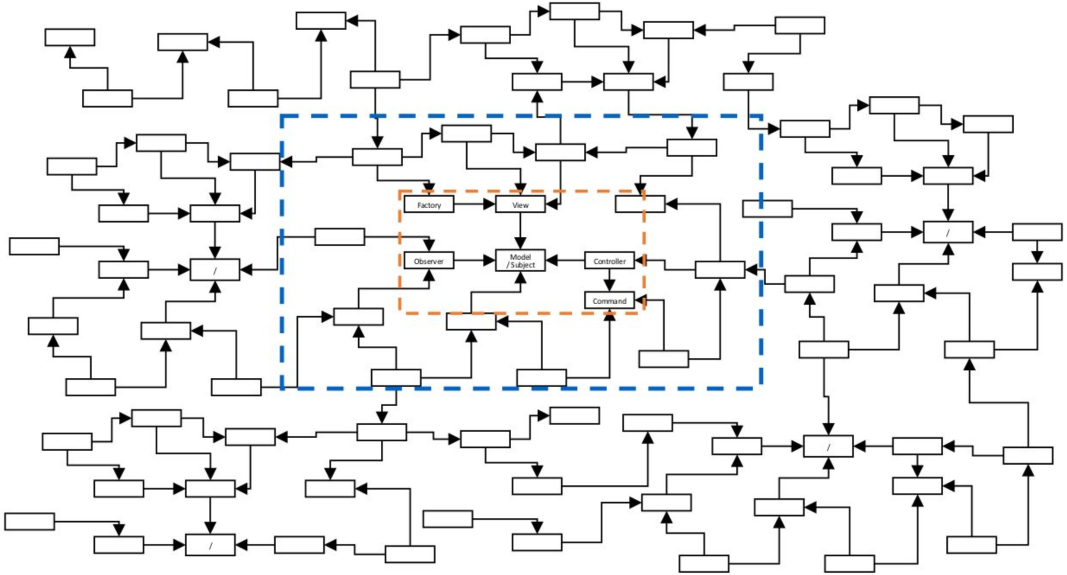
Design Patterns



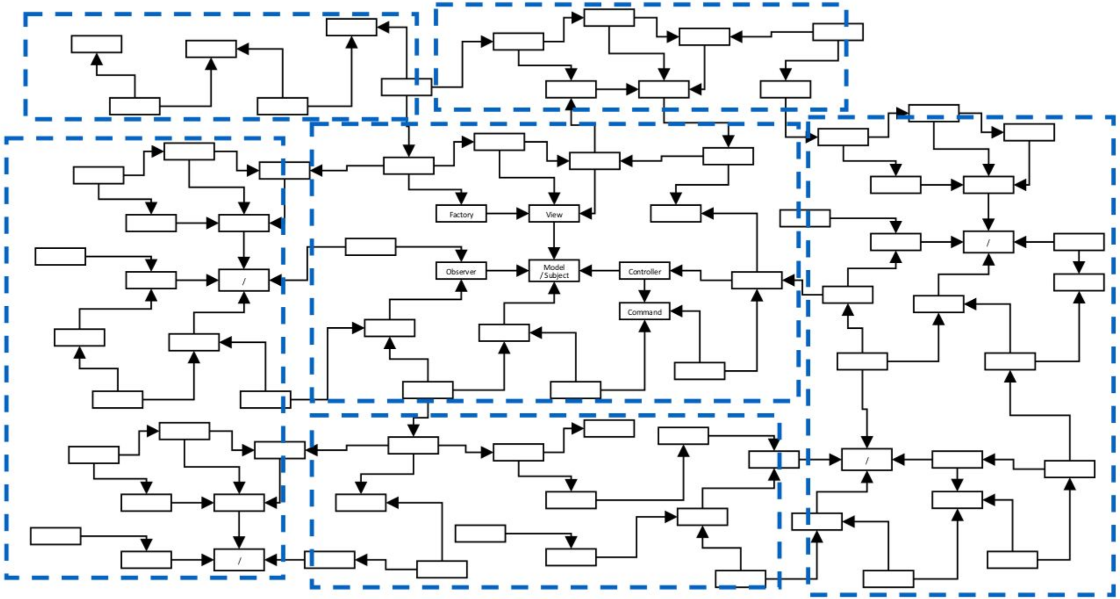
Design Patterns



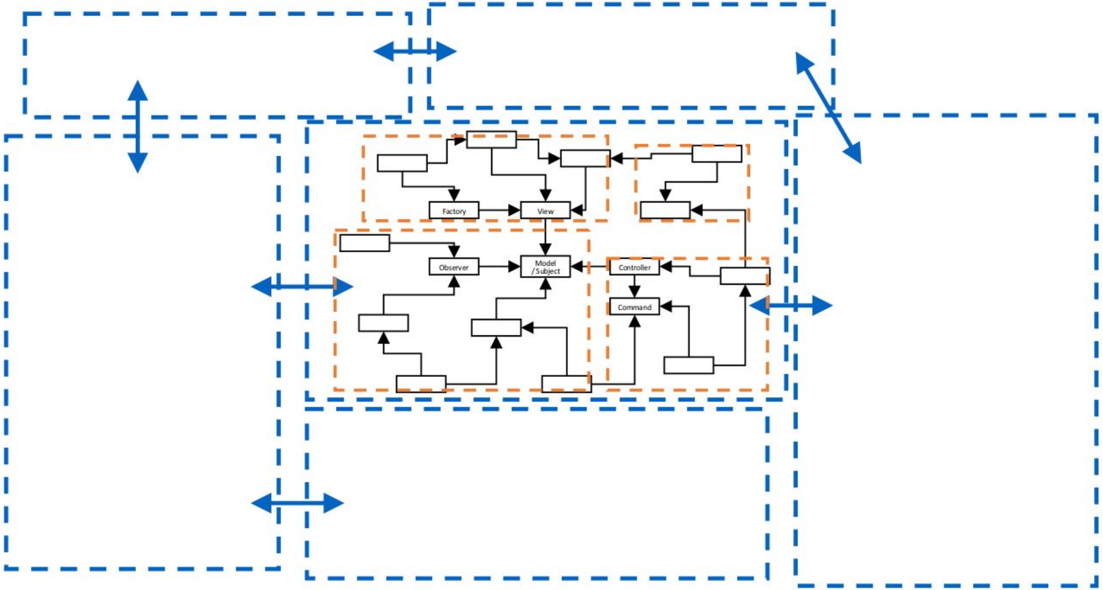
Design Patterns



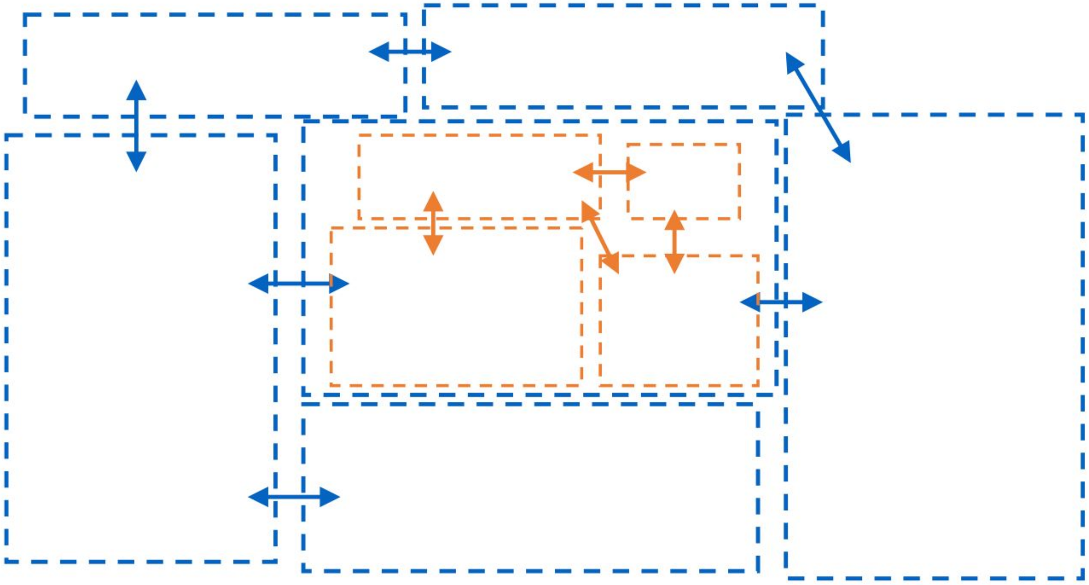
Architecture



Architecture



Architecture



Outline

- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- **Architecting software**
  - **Integrating Architectural Decisions into the SW Development Process**
  - **Common Software Architectures**
  - **Documentation**

Every software system has an architecture

- Whether you know it or not
- Whether you like it or not
- Whether it's documented or not

If you don't consciously elaborate the architecture, it will evolve by itself!

… so don't complain later

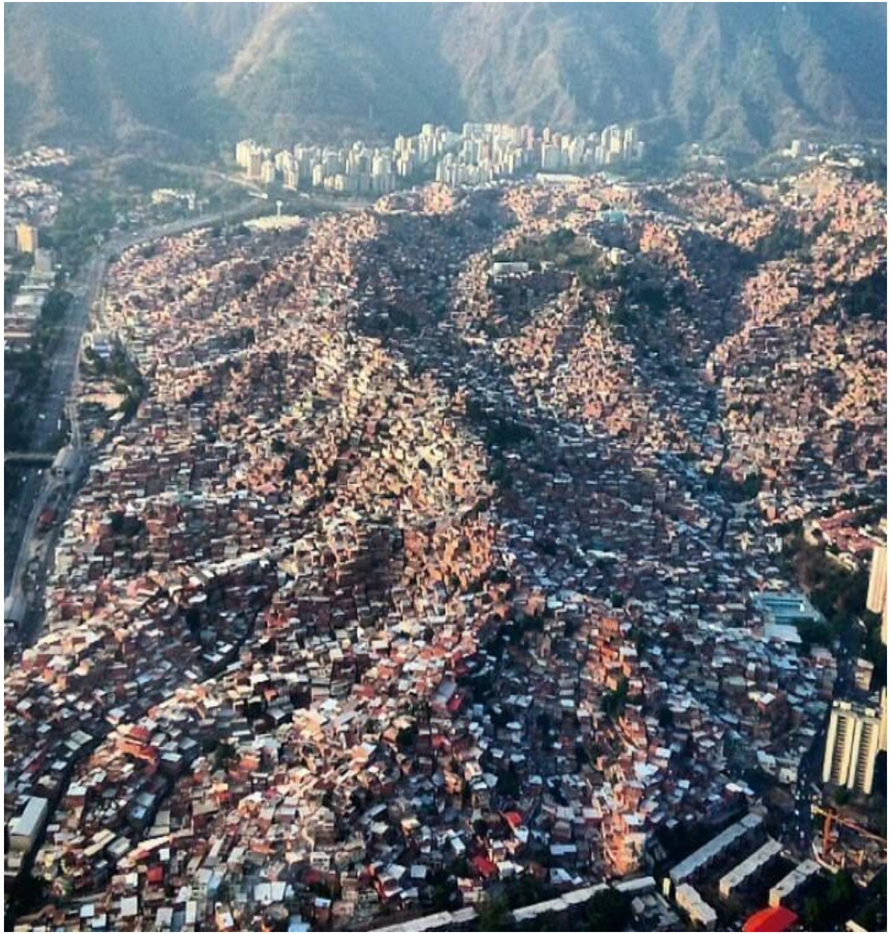
OUR SOFTWARE PRODUCT
DOESN'T MEET STAKEHOLDERS' EXPECTATIONS
DID YOU CONSCIOUSLY MAKE ARCHITECTURAL DECISIONS TO MEET THEIR EXPECTATIONS?
imgflip.com

Architecting Software the SEI Way - Software Architecture Fundamentals: Technical, Business, and Social Influences. Robert Wojcik. 2012

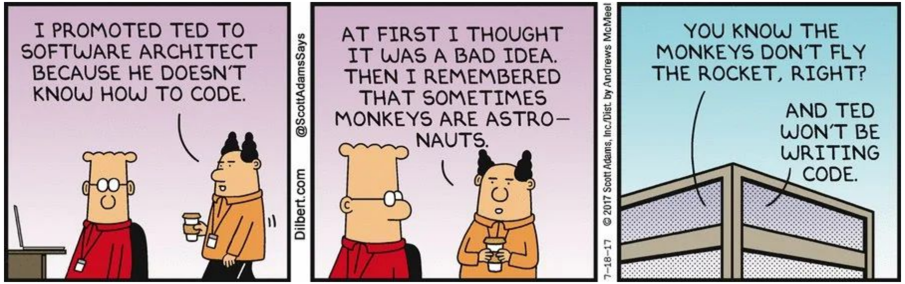
https://www.archdaily.com/

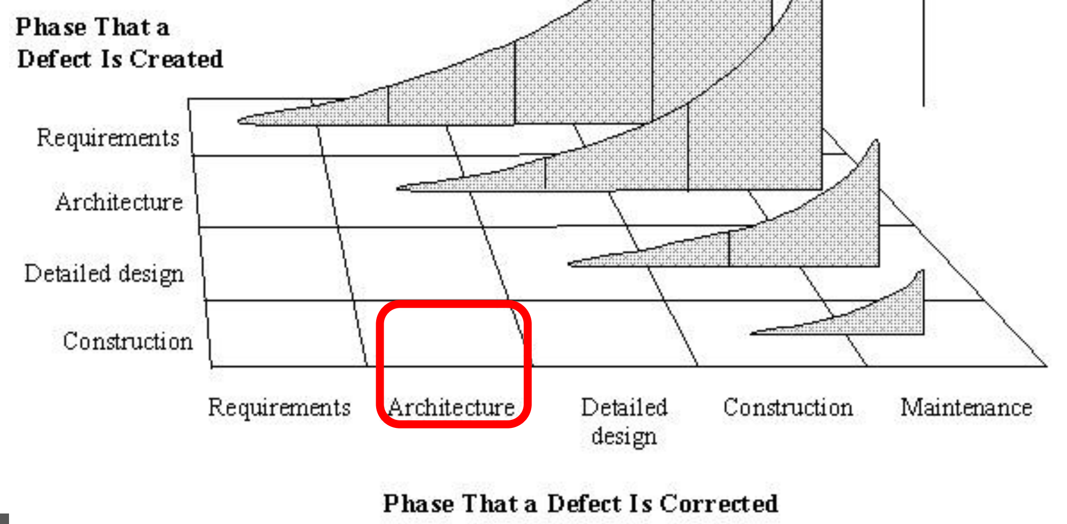
https://www.instagram.com/architectanddesign

https://www.mykonosceramica.com/



The costs of a wrong architecture


I PROMOTED TED TO SOFTWARE ARCHITECT BECAUSE HE DOESN'T KNOW HOW TO CODE.
Dilbert.com @ScottAdamsSays
AT FIRST I THOUGHT IT WAS A BAD IDEA. THEN I REMEMBERED THAT SOMETIMES MONKEYS ARE ASTRO-NAUTS.
7-18-17 © 2017 Scott Adams, Inc./Dist. by Andrews McMeel
YOU KNOW THE MONKEYS DON'T FLY THE ROCKET, RIGHT?
AND TED WON'T BE WRITING CODE.

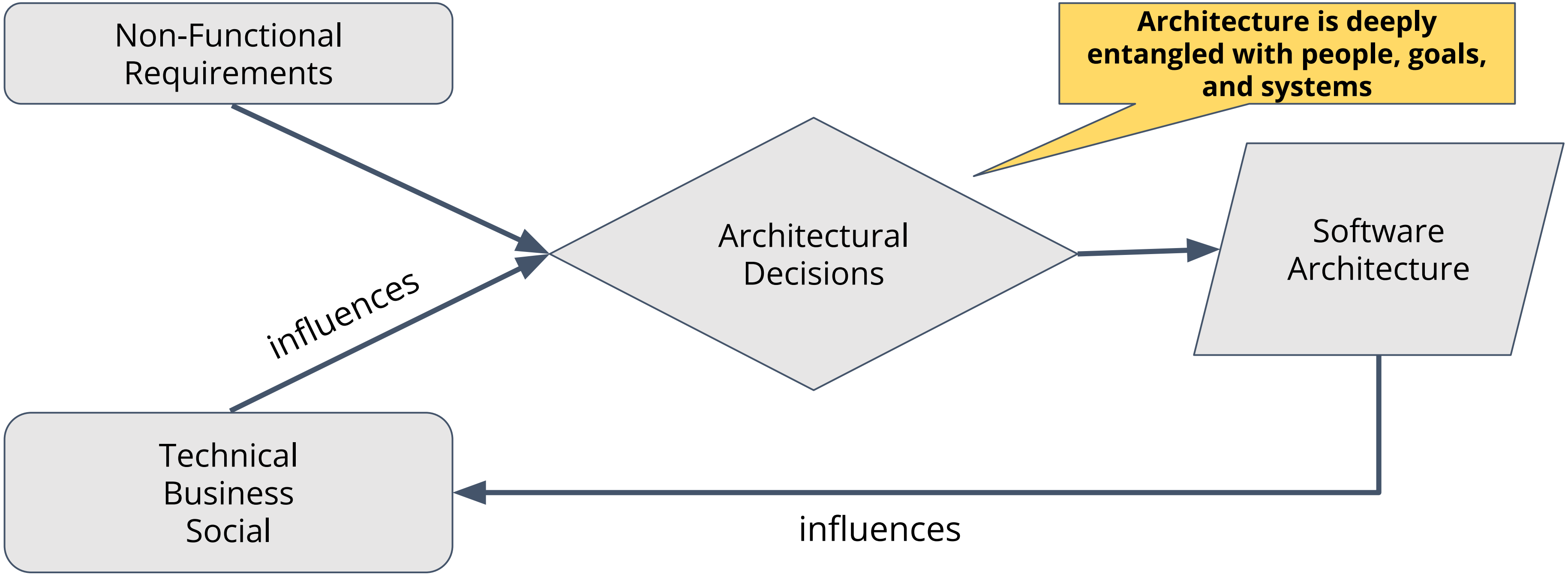


Copyright 1998 Steven C. McConnell. Reprinted with permission from Software Project Survival Guide (Microsoft Press, 1998).

How to make architectural decisions

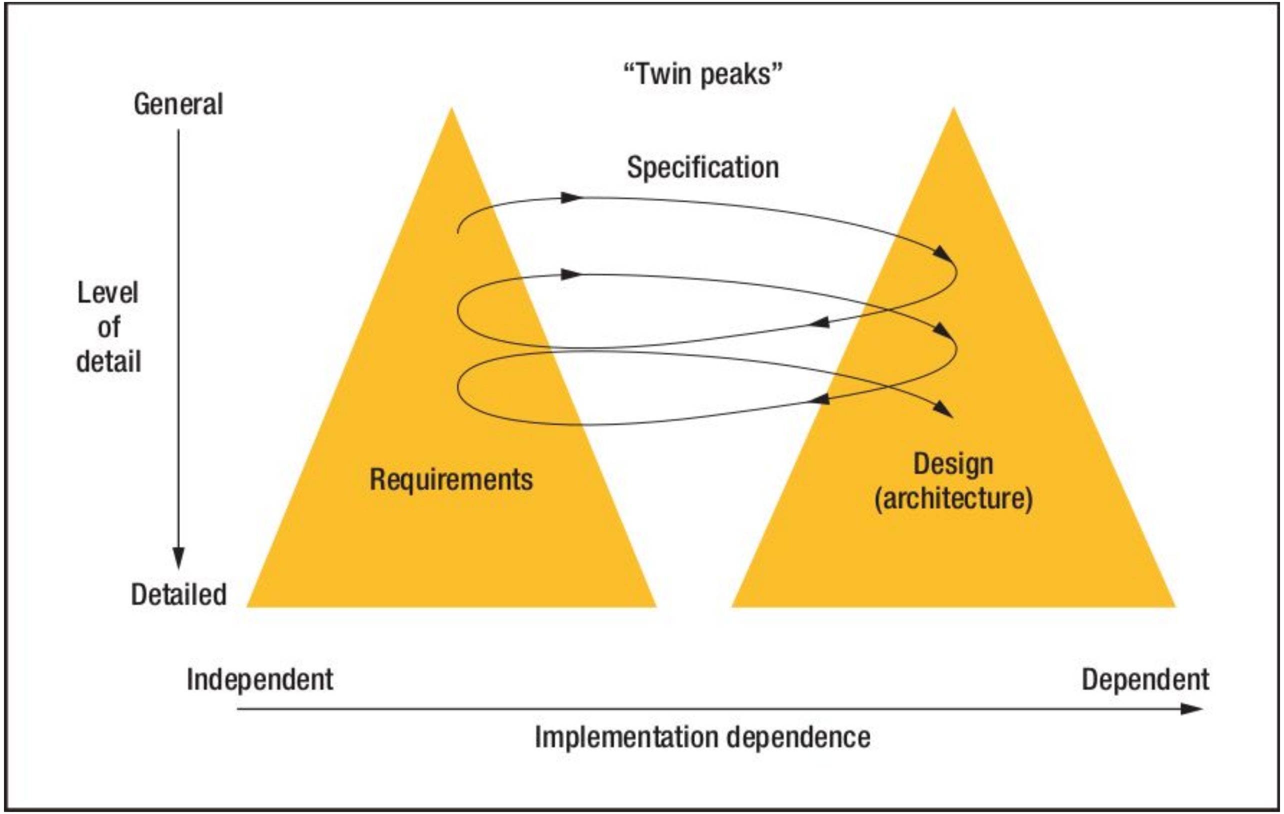
More than one answer

The Ecosystem of Architectural Decisions



Architecture is deeply entangled with people, goals, and systems

Architecting Software the SEI Way - Software Architecture Fundamentals: Technical, Business, and Social Influences. Robert Wojcik. 2012

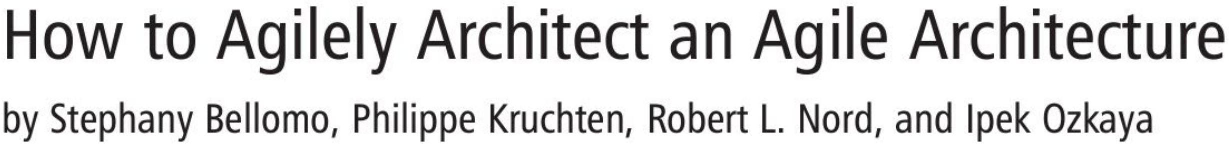


B. Nuseibeh, "Weaving together requirements and architectures". 2001

Agile and Architecture

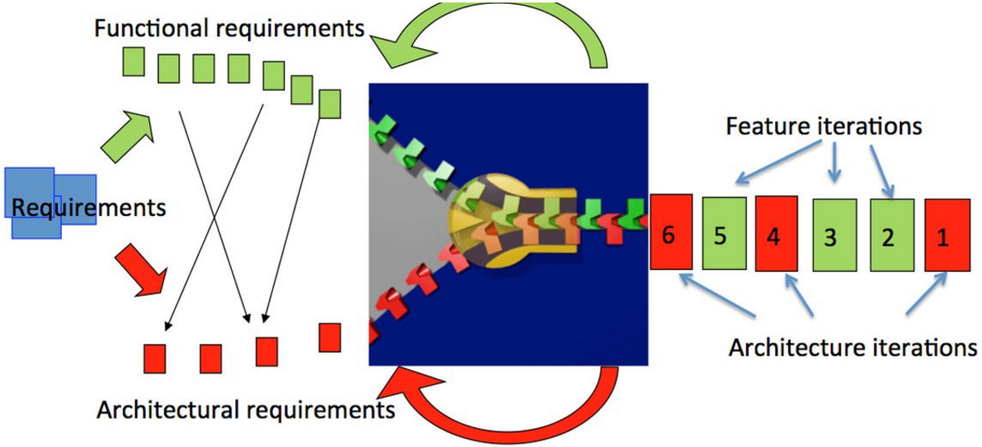
“The best architectures, requirements, and designs emerge from self-organizing teams”

The Zipper Model



How to Agilely Architect an Agile Architecture

by Stephany Bellomo, Philippe Kruchten, Robert L. Nord, and Ipek Ozkaya

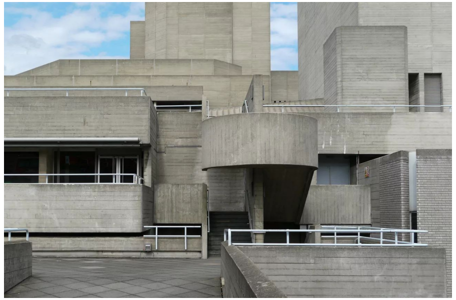


Elicit architecturally significant user stories in early iterations

Outline

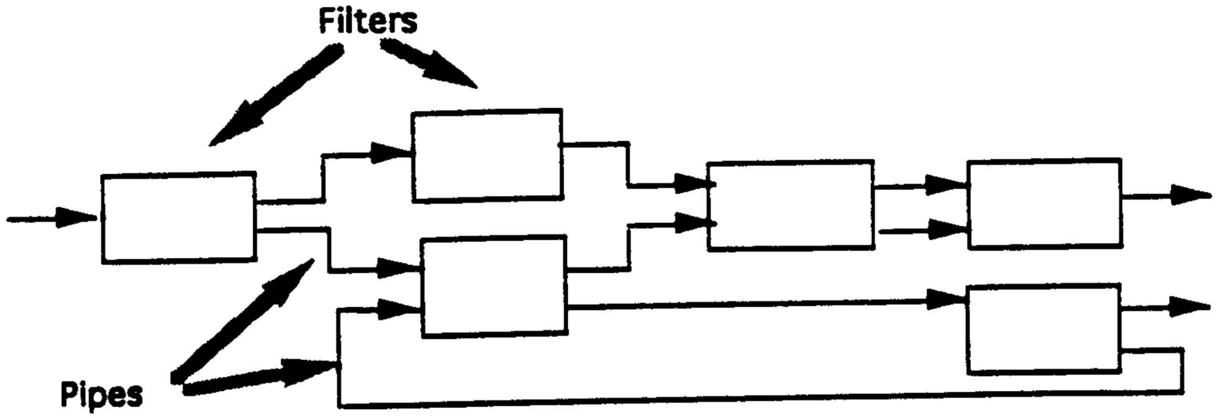
- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- **Architecting software**
  - **Integrating Architectural Decisions into the SW Development Process**
  - **Common Software Architectures**
  - **Documentation**

Common Architectural Styles



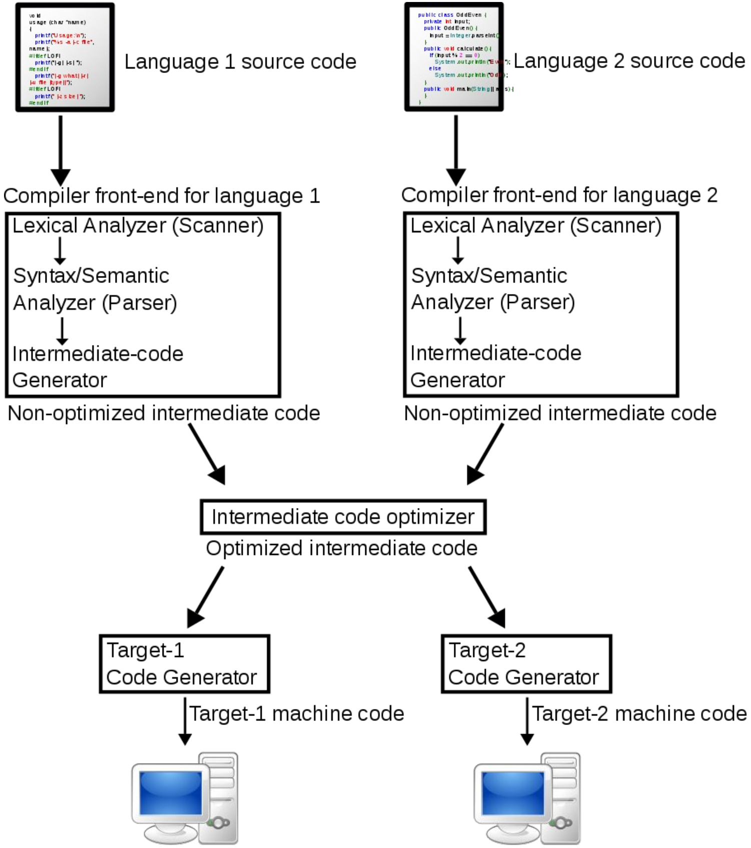
https://www.thespruce.com/top-architectural-styles-4802083

1. Pipes and Filters



© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

Example: Compilers



Language 1 source code

Language 2 source code

Compiler front-end for language 1

Lexical Analyzer (Scanner)

Syntax/Semantic Analyzer (Parser)

Intermediate-code Generator

Non-optimized intermediate code

Compiler front-end for language 2

Lexical Analyzer (Scanner)

Syntax/Semantic Analyzer (Parser)

Intermediate-code Generator

Non-optimized intermediate code

Intermediate code optimizer

Optimized intermediate code

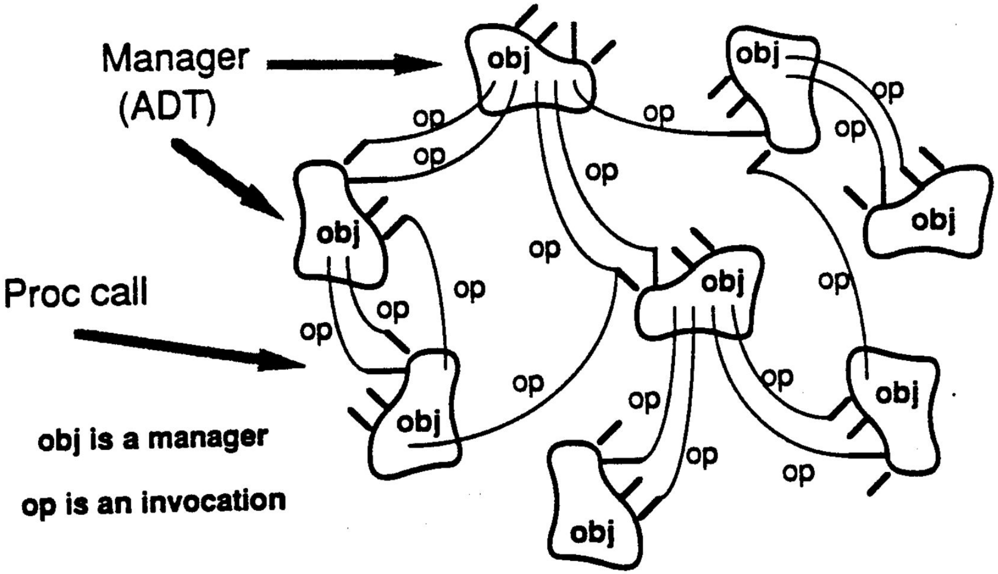
Target-1 Code Generator

Target-1 machine code

Target-2 Code Generator

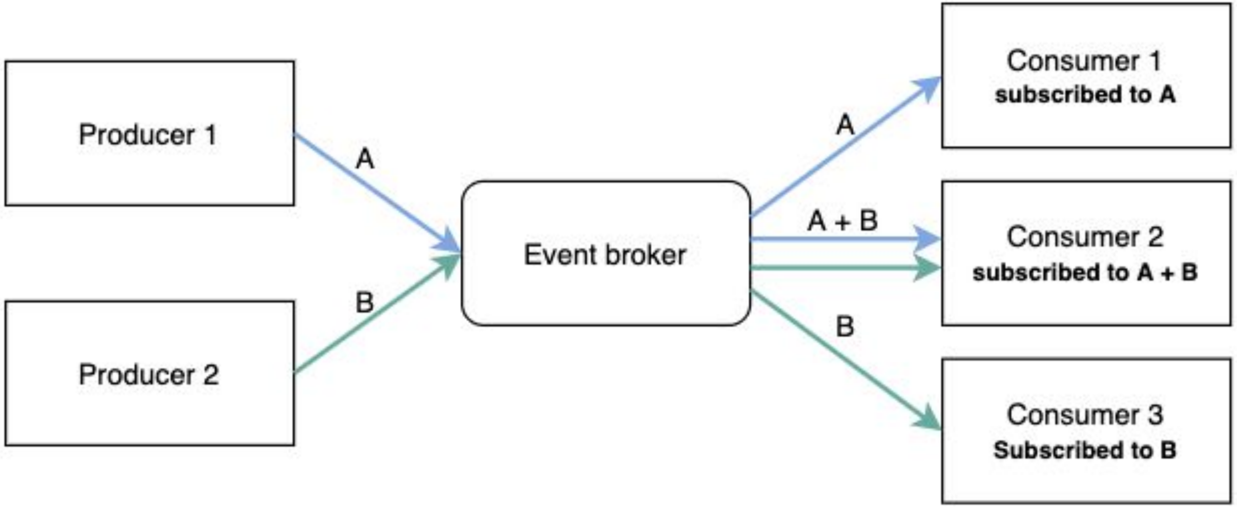
Target-2 machine code

2. Object-Oriented Organization

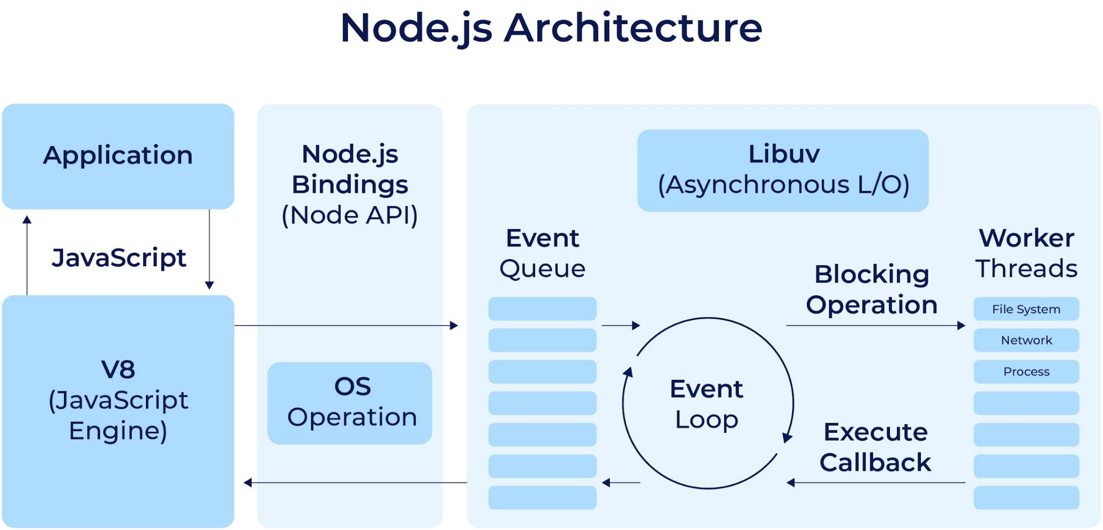


© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

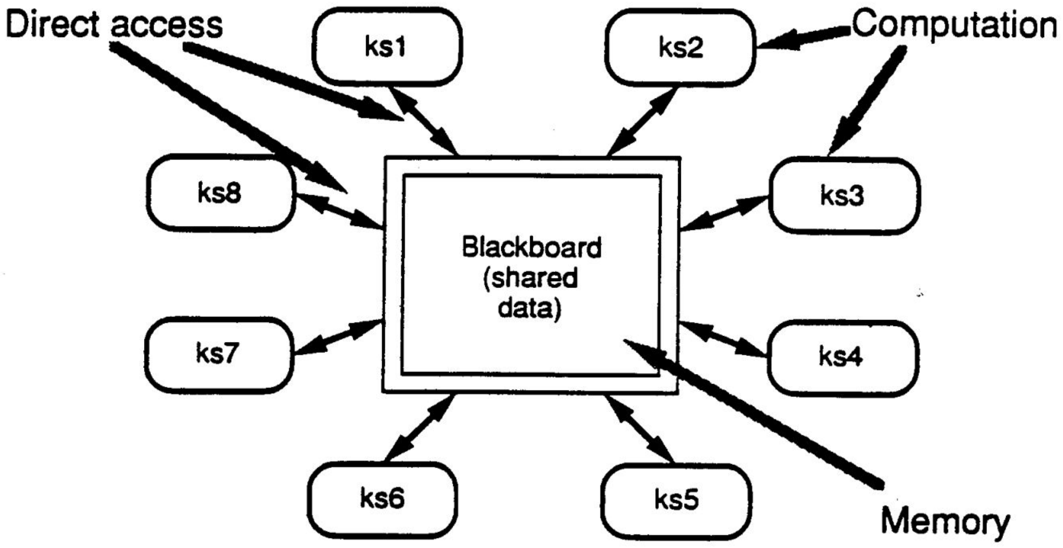
3. Event-Driven Architecture



Example: Node.js

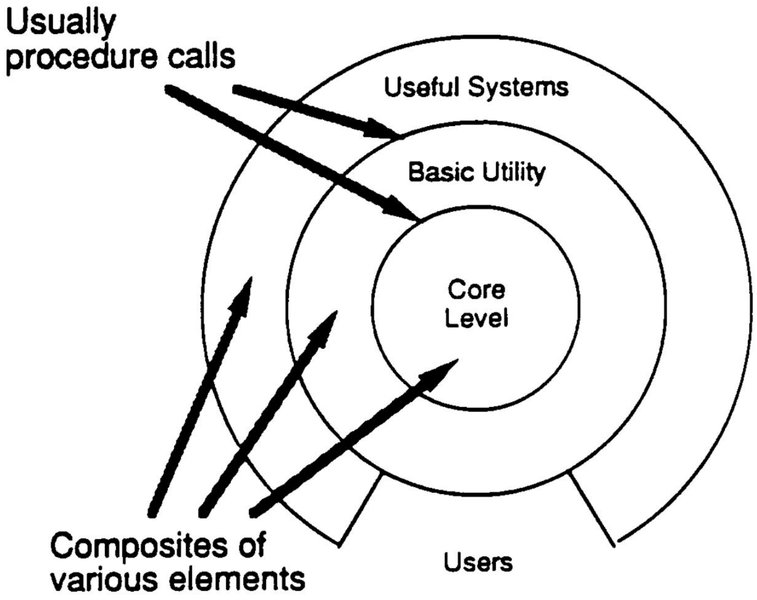


4. Blackboard Architecture



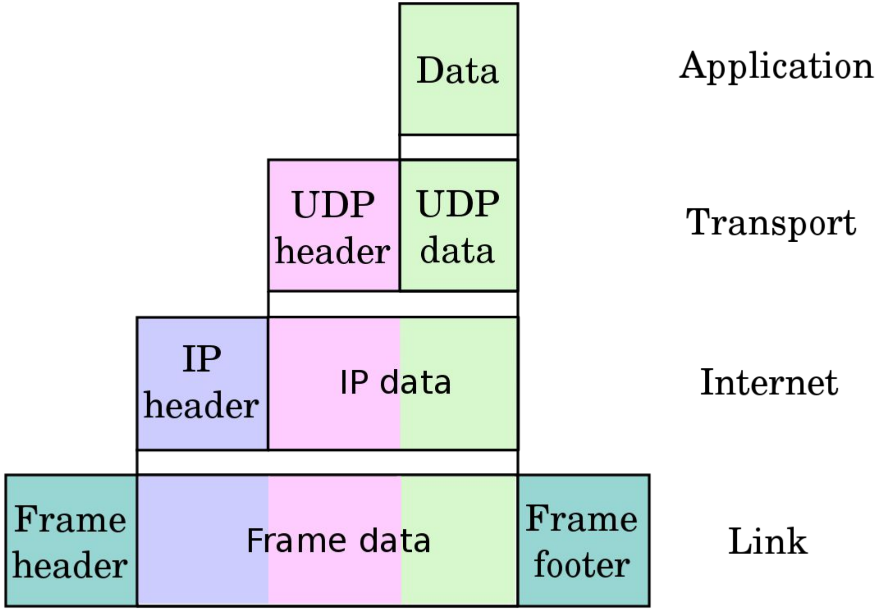
© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

5. Layered Systems



© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

Example: Internet Protocol Suite



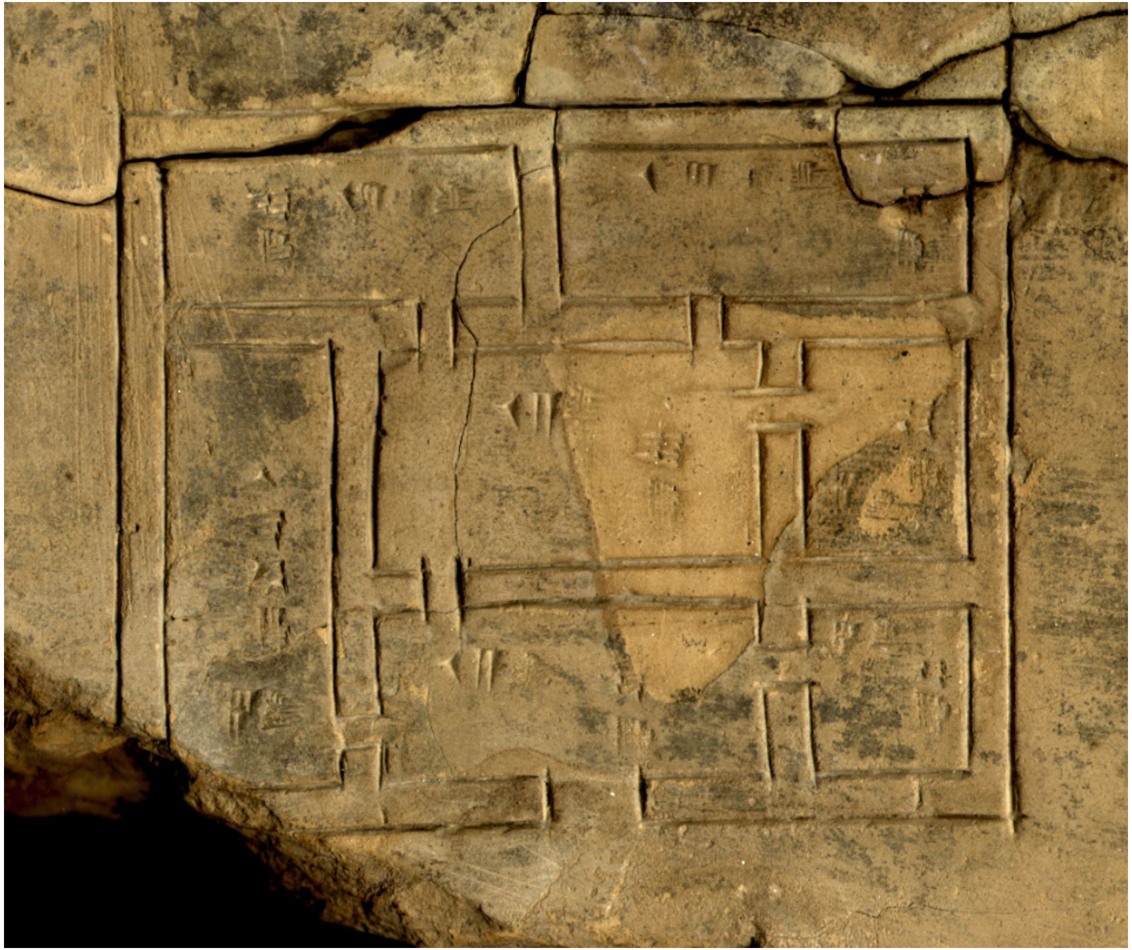
Outline

- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
  - Definitions, Importance
  - Software Design vs. Software Architecture
- **Architecting software**
  - **Integrating Architectural Decisions into the SW Development Process**
  - **Common Software Architectures**
  - **Documentation**

Why Document Architecture?

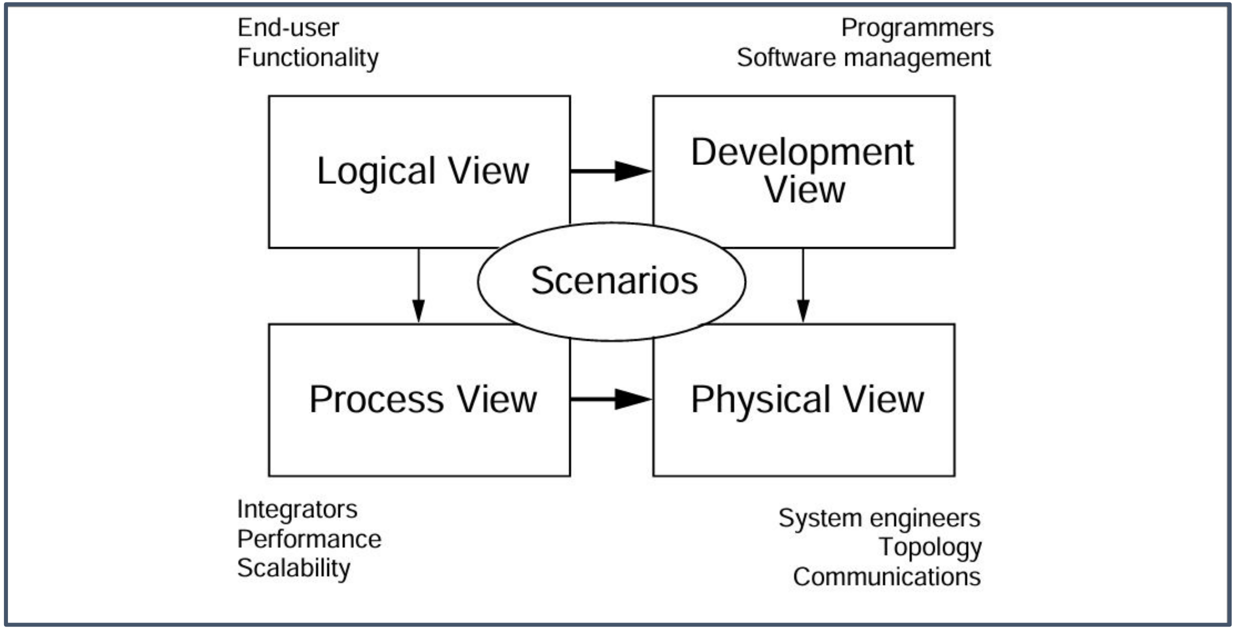
- Blueprint for the system
  - Artifact for early analysis
  - Primary carrier of quality attributes
  - Key to post-deployment maintenance and enhancement
- Documentation speaks for the architect, today and 20 years from today
  - As long as the system is built, maintained, and evolved according to its documented architecture
- Support traceability

5000 years old floorplan depicted on a tablet excavated in Umma (now Iraq), now kept in Vorderasiatisches Museum, Berlin, Germany



https://cdli.mpiwg-berlin.mpg.de/artifacts/125392
Plan or Drawing tablet excavated in Umma (mod. Tell Jokha), dated to the Ur III (ca. 2100-2000 BC) period and now kept in Vorderasiatisches Museum, Berlin, Germany

The “4+1” view model

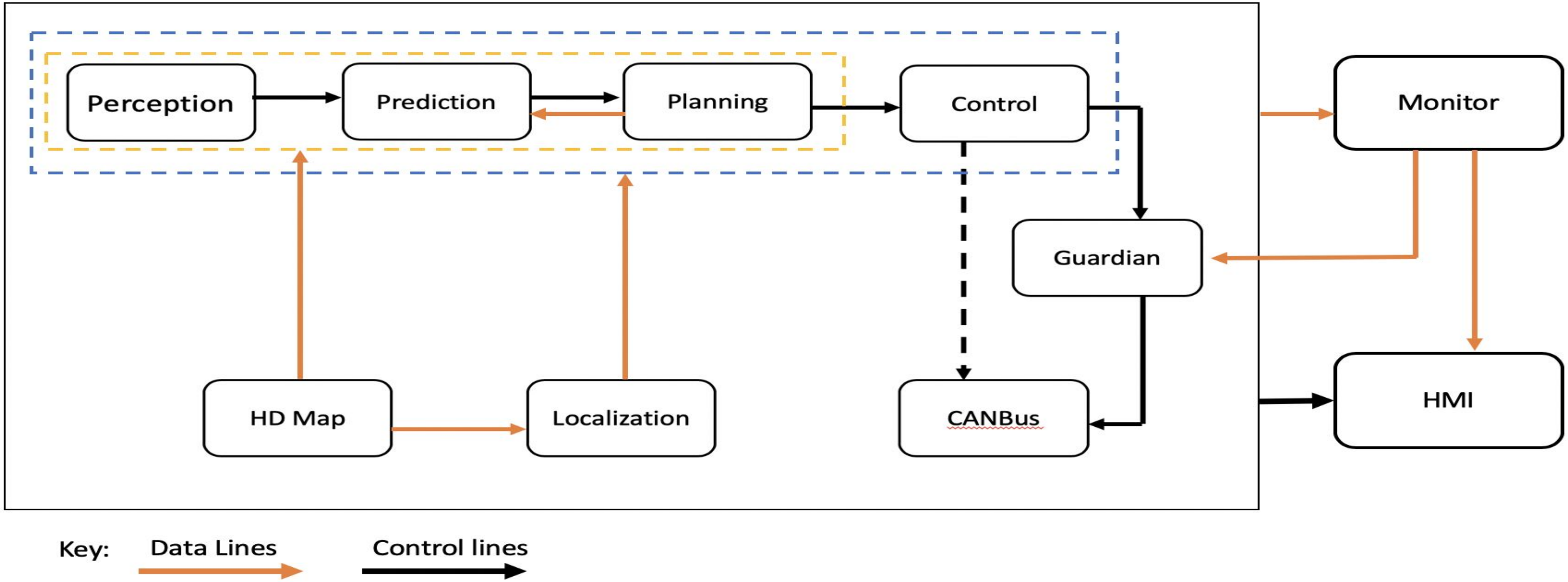


Philippe Kruchten, Architectural Blueprints—The “4+1” View Model of Software Architecture[

Common Views in Documenting Software Architecture

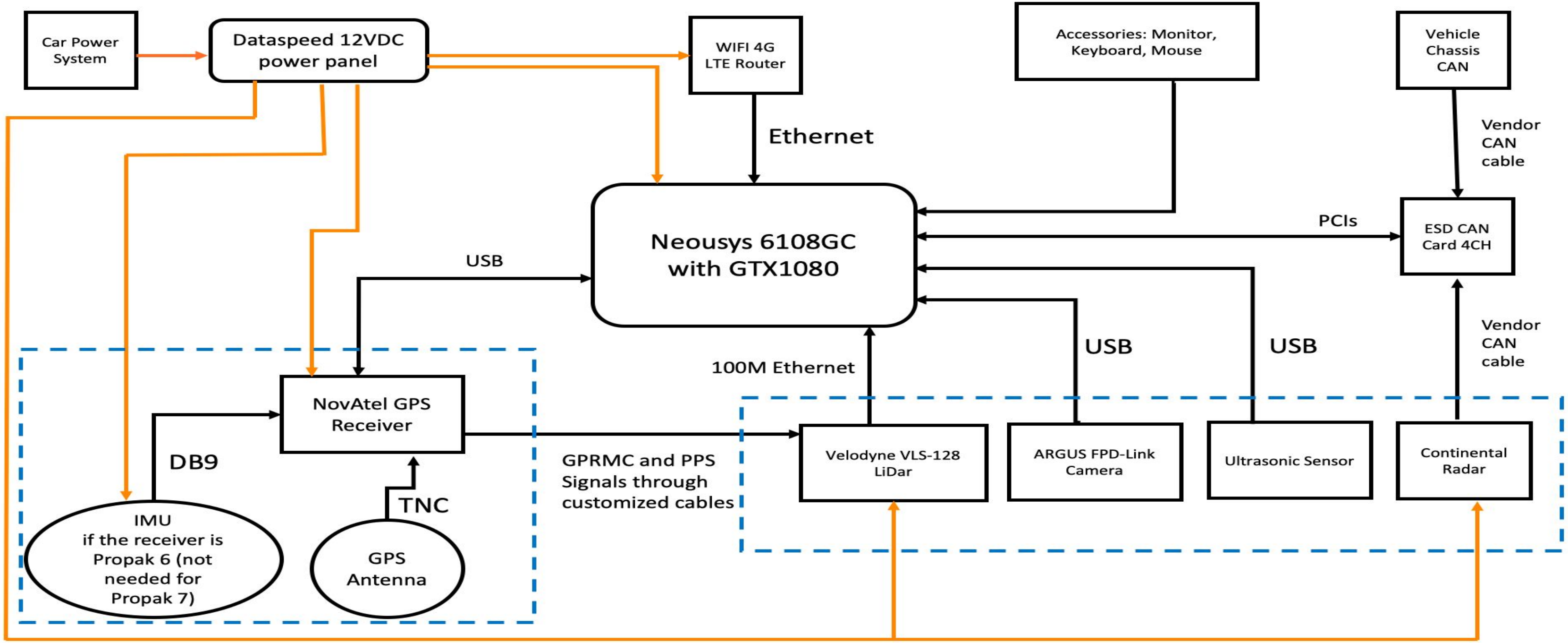
- Logical View (End user)
  - Functionality
  - Subsystems, structures and their relations (dependencies, ...)
- Process View (System Integration)
  - Non-functional aspects
  - Components (processes, runnable entities) and connectors (messages, data flow, ...)
- Development View (Developers)
  - Software modularity / decomposition
- Physical View (System Engineer/DevOps)
  - Hardware structures and their connections
  - Deployment
- Scenarios (All)
  - Outline tasks/use cases
  - Sequences of interactions between objects and processes

Apollo Software Architecture



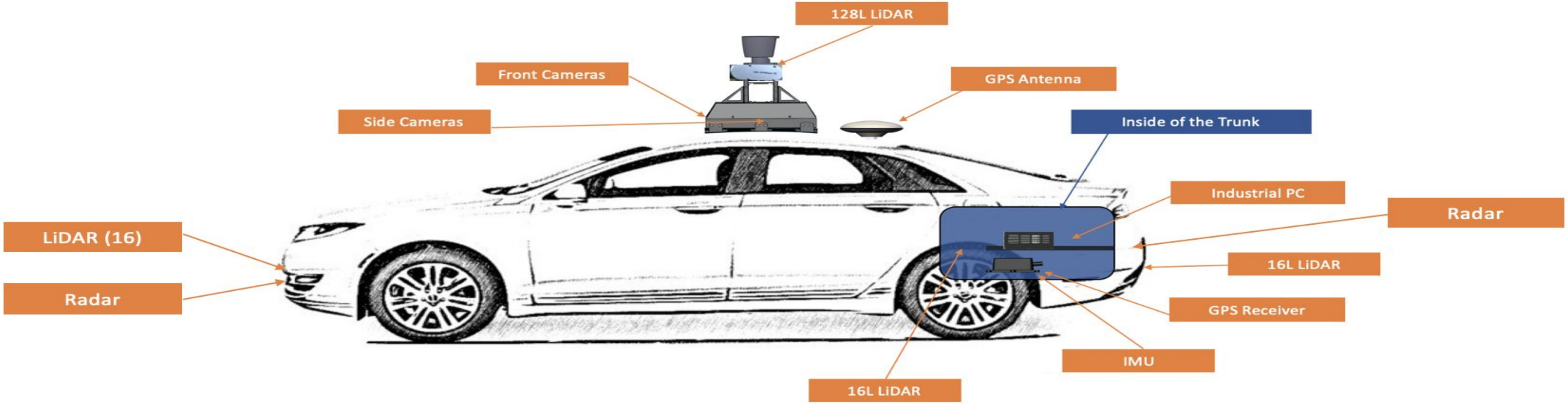
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/docs/specs/Apollo_5.5_Software_Architecture.md

Apollo Hardware Architecture



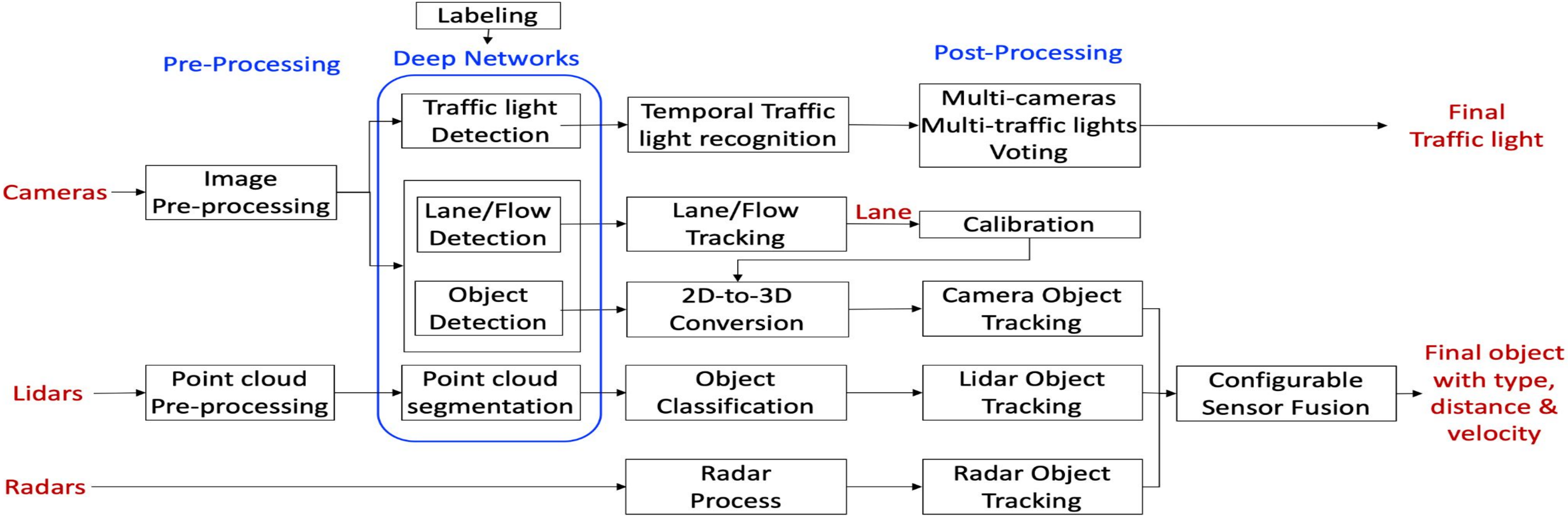
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Hardware/Vehicle Overview

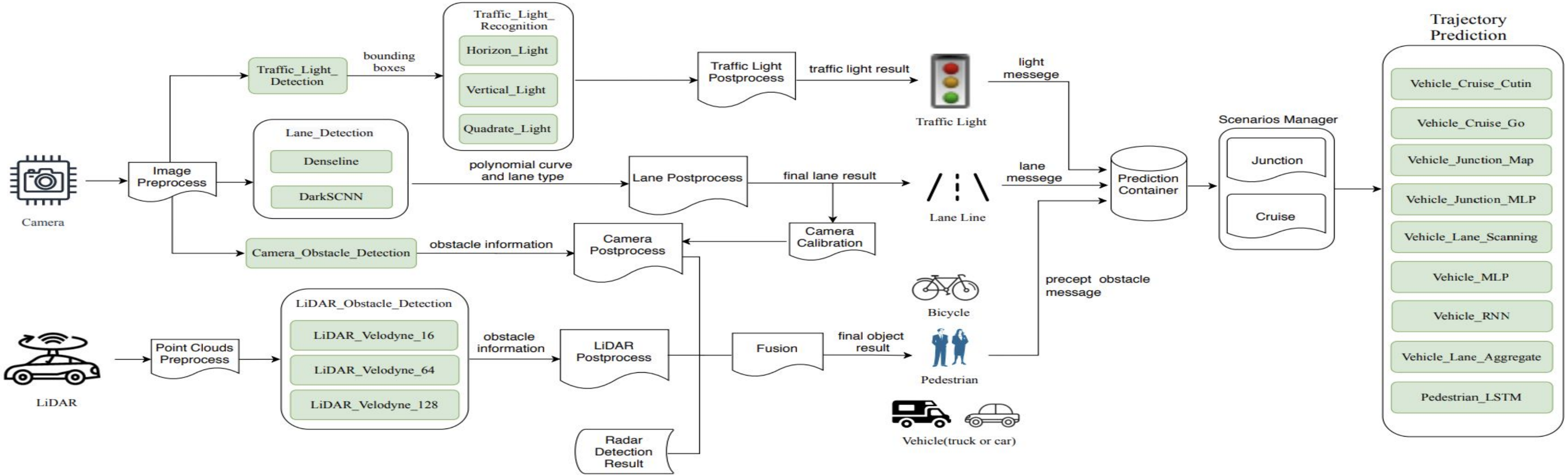


Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Perception Module

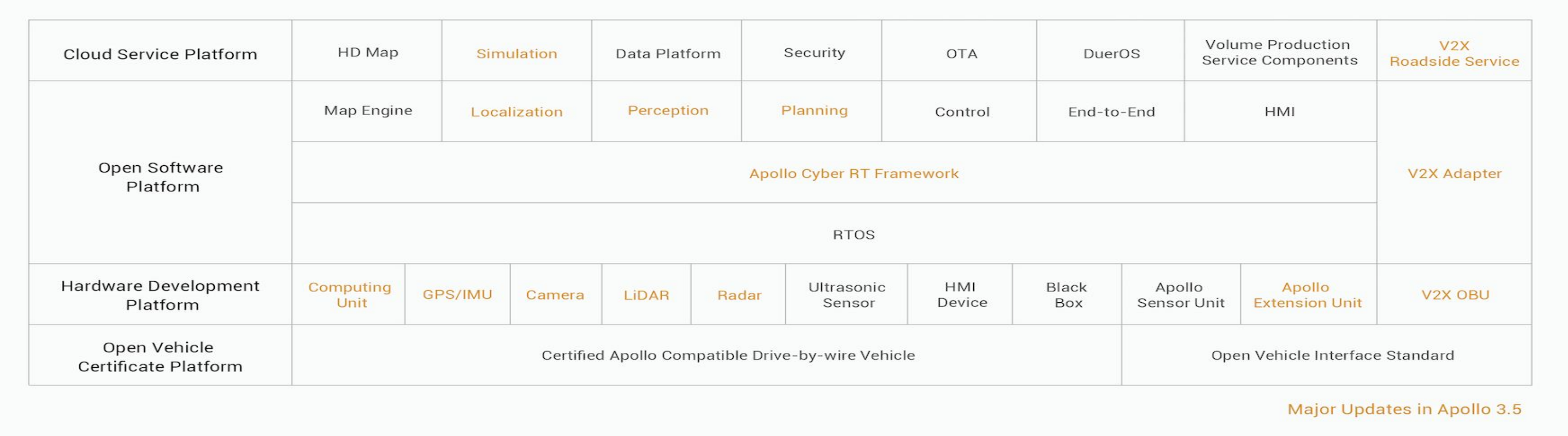


Apollo ML Models



Source: Zi Peng, Jinqiu Yang, Tse-Hsun (Peter) Chen, and Lei Ma. 2020. A First Look at the Integration of Machine Learning Models in Complex Autonomous Driving Systems: A Case Study on Apollo. In Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20), https://doi.org/10.1145/ 3368089.3417063

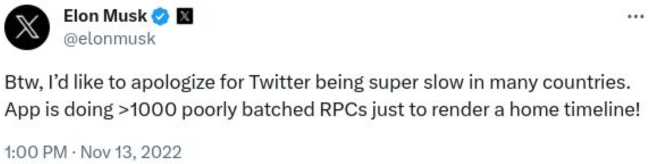
Apollo Software Stack



<table>
<tr><td>Cloud Service Platform</td><td colspan="2">HD Map</td><td colspan="2">Simulation</td><td colspan="3">Data Platform</td><td colspan="2">Security</td><td colspan="3">OTA</td><td colspan="2">DuerOS</td><td colspan="2">Volume Production Service Components</td><td>V2X Roadside Service</td></tr>
<tr><td rowspan="3">Open Software Platform</td><td colspan="2">Map Engine</td><td colspan="2">Localization</td><td colspan="3">Perception</td><td colspan="2">Planning</td><td colspan="3">Control</td><td colspan="2">End-to-End</td><td colspan="2">HMI</td><td rowspan="3">V2X Adapter</td></tr>
<tr><td colspan="16">Apollo Cyber RT Framework</td></tr>
<tr><td colspan="16">RTOS</td></tr>
<tr><td>Hardware Development Platform</td><td>Computing Unit</td><td colspan="2">GPS/IMU</td><td colspan="2">Camera</td><td>LiDAR</td><td colspan="2">Radar</td><td colspan="2">Ultrasonic Sensor</td><td>HMI Device</td><td colspan="2">Black Box</td><td colspan="2">Apollo Sensor Unit</td><td>Apollo Extension Unit</td><td>V2X OBU</td></tr>
<tr><td>Open Vehicle Certificate Platform</td><td colspan="13">Certified Apollo Compatible Drive-by-wire Vehicle</td><td colspan="4">Open Vehicle Interface Standard</td></tr>
</table>

Major Updates in Apollo 3.5

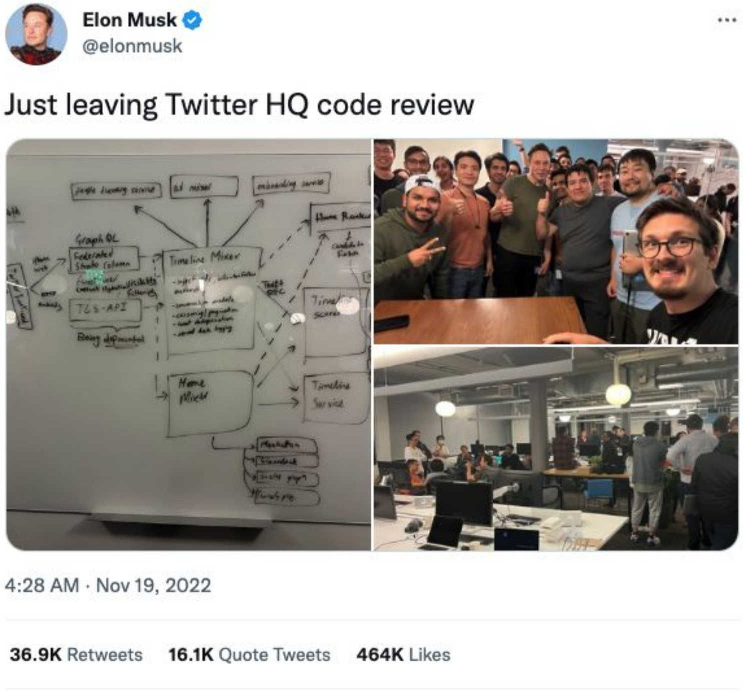
Source: https://github.com/ApolloAuto/



Elon Musk
@elonmusk

Btw, I'd like to apologize for Twitter being super slow in many countries. App is doing >1000 poorly batched RPCs just to render a home timeline!

1:00 PM · Nov 13, 2022

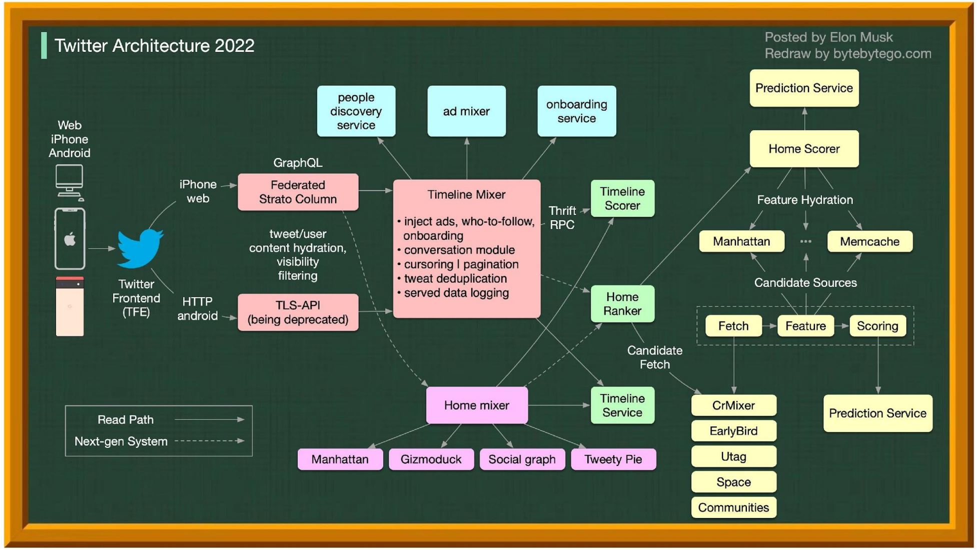


Elon Musk
@elonmusk

Just leaving Twitter HQ code review

4:28 AM · Nov 19, 2022

36.9K Retweets 16.1K Quote Tweets 464K Likes

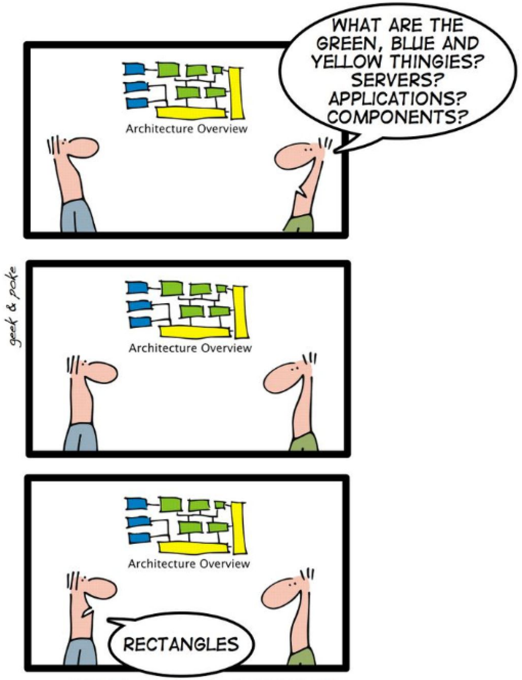


Twitter Architecture 2022

Posted by Elon Musk
Redraw by bytebytego.com

Guidelines for selecting a notation

- Suitable for purpose
- Often visual for compact representation
- Usually, boxes and arrows
- UML possible (semi-formal), but possibly constraining
    - Note the different abstraction level – Subsystems or processes, not classes or objects
- Formal notations available
- Decompose diagrams hierarchically and in views
- Always include a legend
- Define precisely what the boxes mean
- Define precisely what the lines mean
- Do not try to do too much in one diagram
    - Each view of architecture should fit on a page
    - Use hierarchy



Learning Goals

- Understand the abstraction level of architectural reasoning
- Appreciate how software systems can be viewed at different abstraction levels
- Distinguish software architecture from (object-oriented) software design
- Explain the importance of architectural decisions
- Integrate architectural decisions into the software development process
- Document architectures clearly, without ambiguity