Introduction to Software Architecture

17-313 Spring 2025

Foundations of Software Engineering

https://cmu-313.github.io

Michael Hilton, Austin Henley, and Nadia Nahar

Administrivia

- “Regrade requests can be submitted via Gradescope. The regrade period is open for one week after grades have been released for a particular assignment.”

Smoking Section

- Last full row

Learning Goals

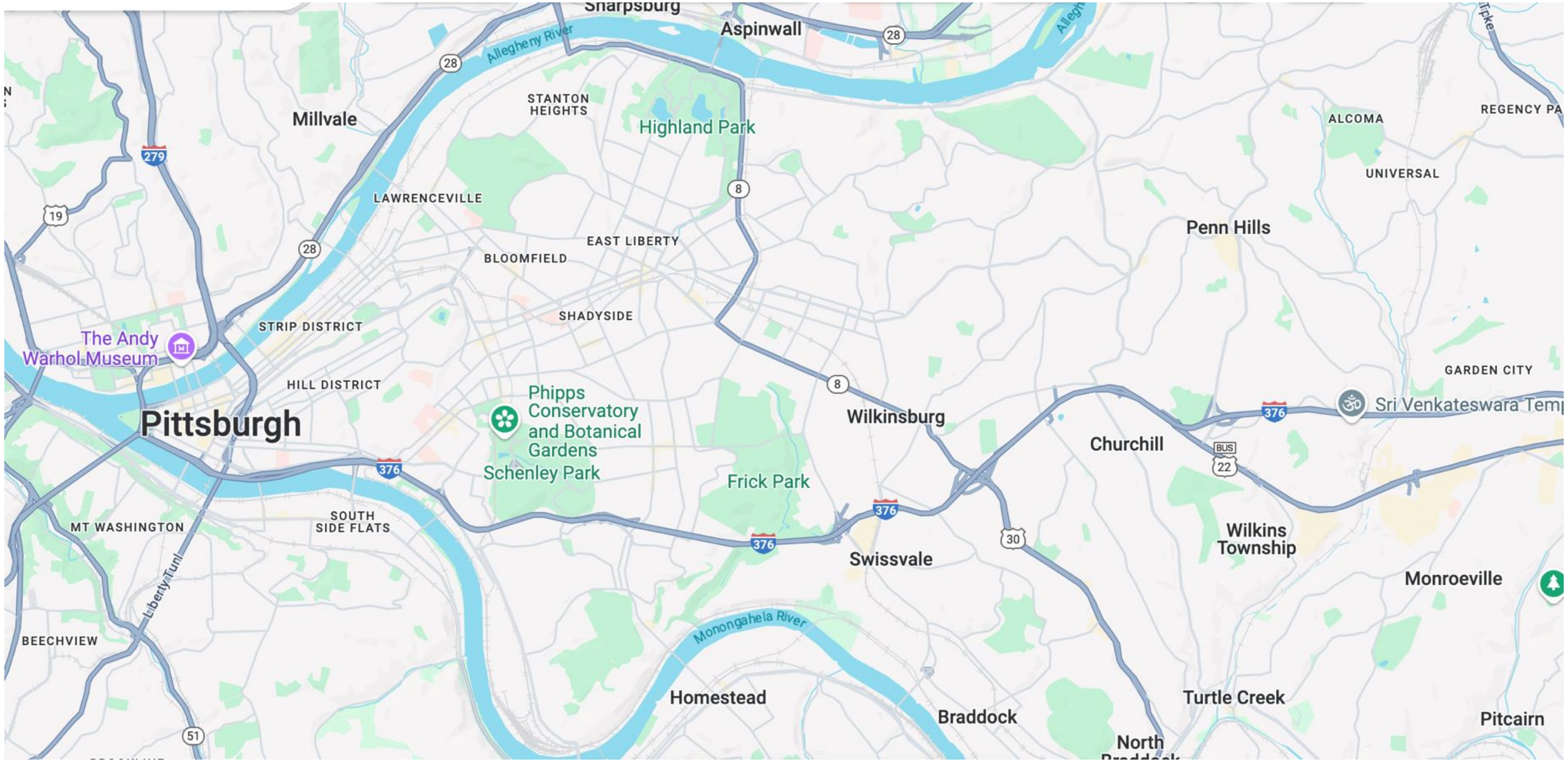
- Understand the abstraction level of architectural reasoning
- Appreciate how software systems can be viewed at different abstraction levels
- Distinguish software architecture from (object-oriented) software design
- Explain the importance of architectural decisions
- Integrate architectural decisions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

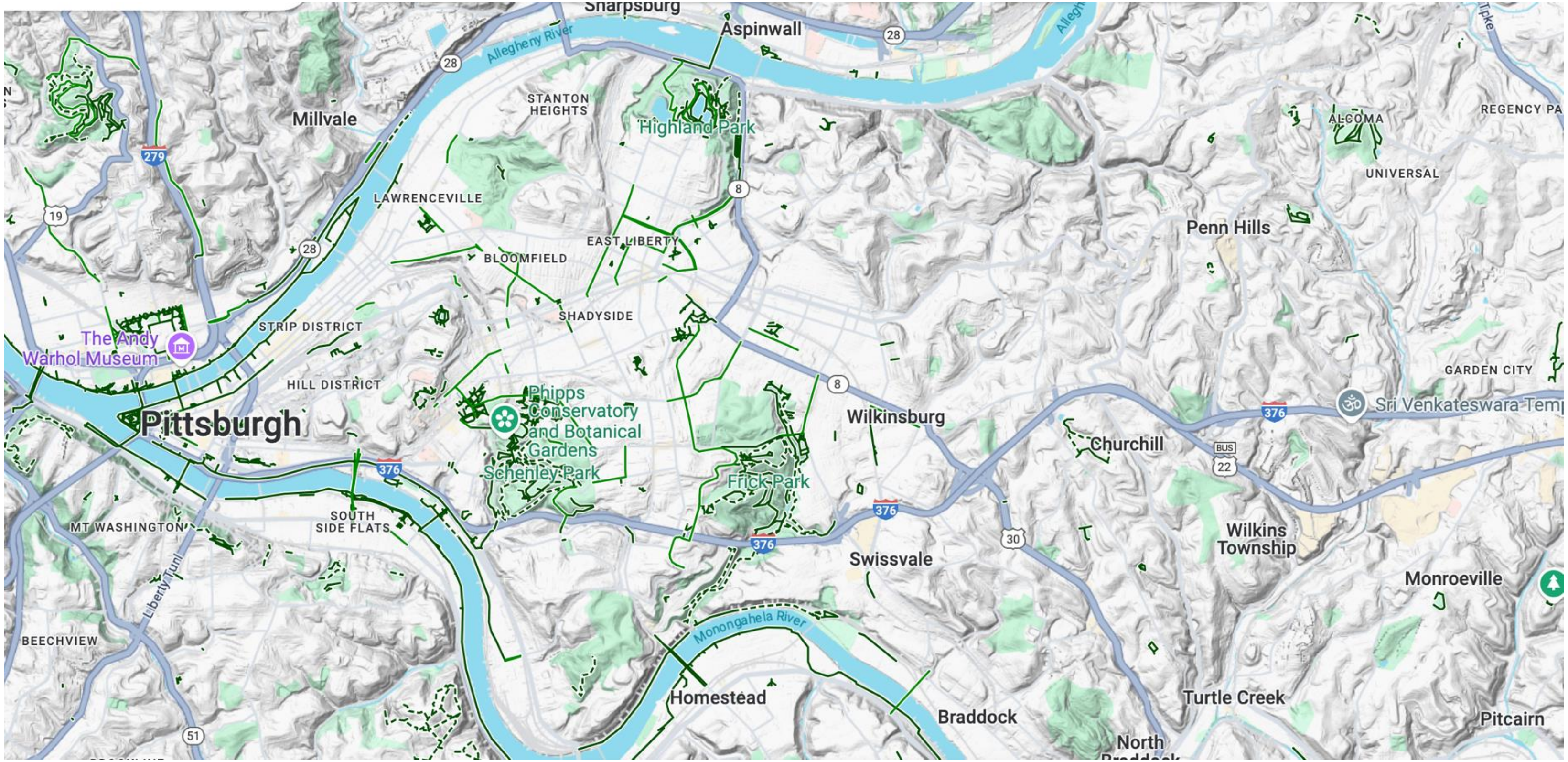
Outline

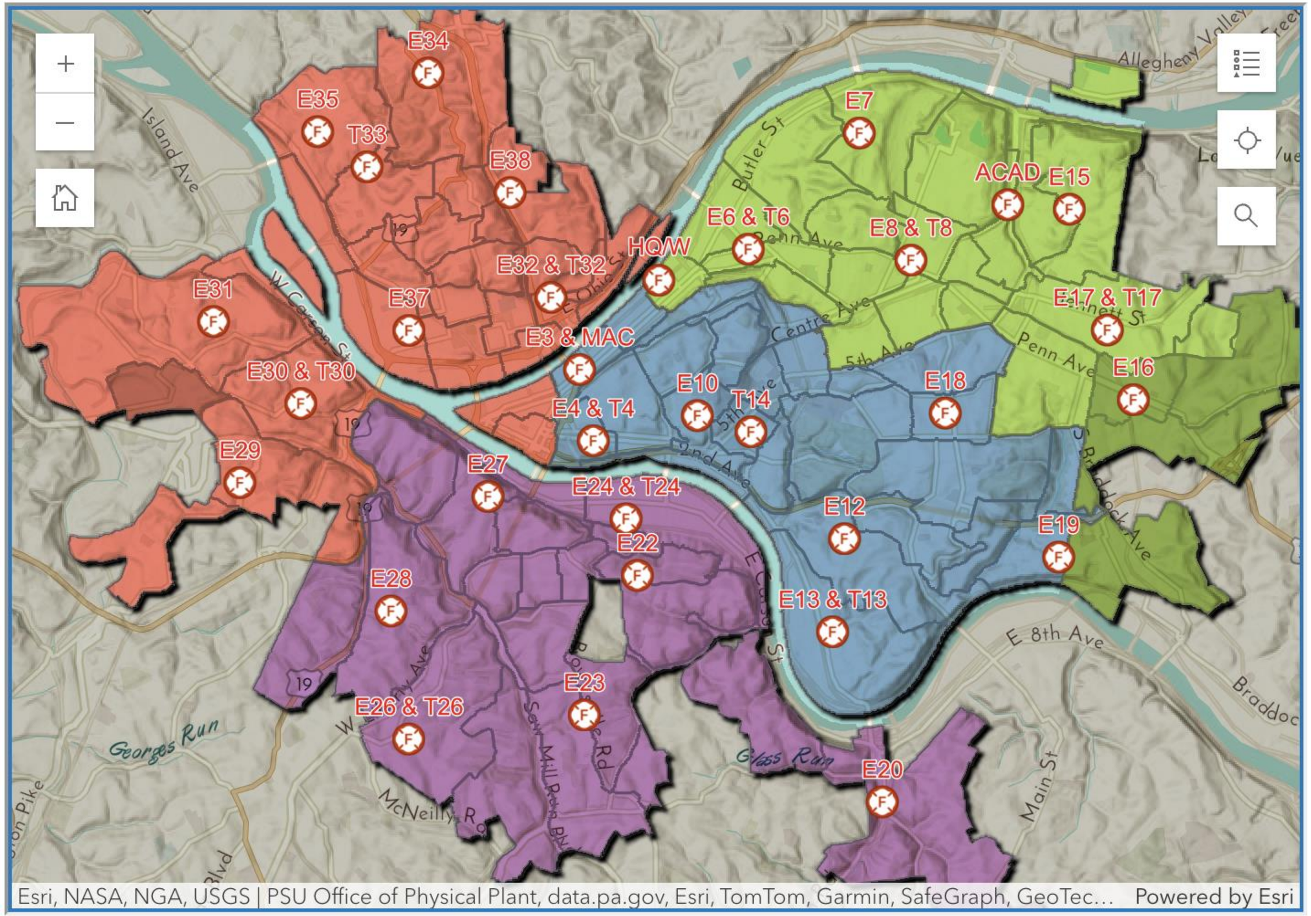
- Views and Abstraction
- Case Study: Autonomous Vehicles
- Software Architecture
    - Definitions, Importance
    - Software Design vs. Software Architecture
- Architecting software
    - Integrating Architectural Decisions into the SW Development Process
    - Common Software Architectures
    - Documentation

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


Sharpsburg
Aspinwall
Allegheny River
Millvale
Stanton Heights
Highland Park
Lawrenceville
East Liberty
Bloomfield
Shadyside
Strip District
The Andy Warhol Museum
Hill District
Pittsburgh
Phipps Conservatory and Botanical Gardens
Schenley Park
Frick Park
Wilkinsburg
Swissvale
South Side Flats
Mt Washington
Liberty Tunl
Beechview
Monongahela River
Homestead
Braddock
North Braddock
Turtle Creek
Pitcairn
Monroeville
Wilkins Township
Churchill
Penn Hills
Alcoma
Universal
Regency Pa
Garden City
Sri Venkateswara Temp
279
19
28
8
376
30
22
51


Sharpsburg
Aspinwall
Allegheny River
Millvale
Stanton Heights
Highland Park
Lawrenceville
East Liberty
Bloomfield
Shadyside
Strip District
The Andy Warhol Museum
Hill District
Pittsburgh
Phipps Conservatory and Botanical Gardens
Schenley Park
Frick Park
Wilkinsburg
Swissvale
South Side Flats
Mt Washington
Beechview
Liberty Tunl
Monongahela River
Homestead
Braddock
North Braddock
Turtle Creek
Churchill
Penn Hills
Alcoma
Universal
Regency Pa
Garden City
Sri Venkateswara Temp
Wilkins Township
Monroeville
Pitcairn



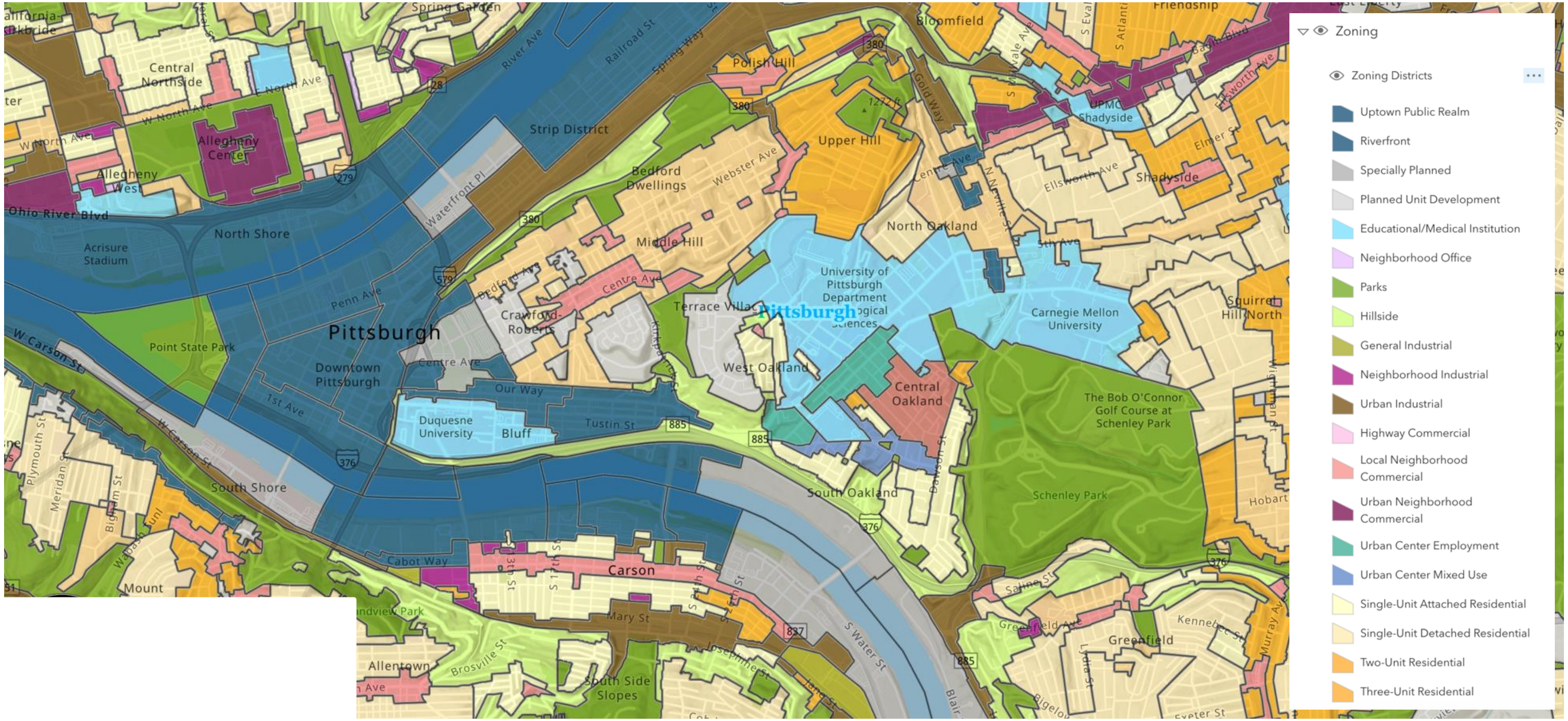
Esri, NASA, NGA, USGS | PSU Office of Physical Plant, data.pa.gov, Esri, TomTom, Garmin, SafeGraph, GeoTec… Powered by Esri

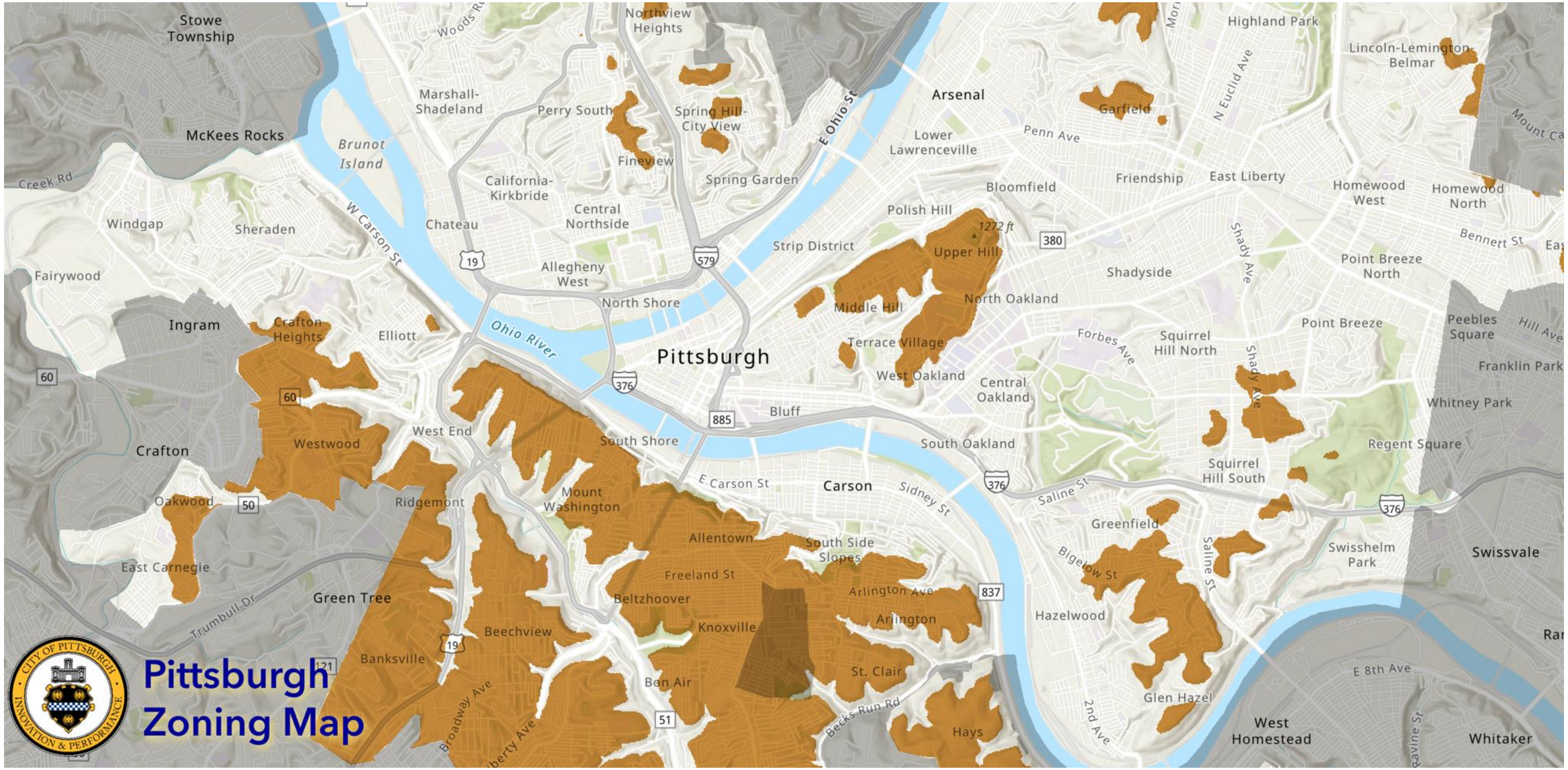
Fire Station

Ⓕ

Fire Districts

🟥	1
🟦	2
🟩	3
🟪	4


Zoning
Zoning Districts
Uptown Public Realm
Riverfront
Specially Planned
Planned Unit Development
Educational/Medical Institution
Neighborhood Office
Parks
Hillside
General Industrial
Neighborhood Industrial
Urban Industrial
Highway Commercial
Local Neighborhood Commercial
Urban Neighborhood Commercial
Urban Center Employment
Urban Center Mixed Use
Single-Unit Attached Residential
Single-Unit Detached Residential
Two-Unit Residential
Three-Unit Residential
Central Northside
Allegheny Center
Allegheny West
Strip District
North Shore
Acrisure Stadium
Pittsburgh
Downtown Pittsburgh
Point State Park
Bedford Dwellings
Middle Hill
Upper Hill
Polish Hill
Bloomfield
Crawford-Roberts
Terrace Village
West Oakland
North Oakland
Central Oakland
South Oakland
University of Pittsburgh
Carnegie Mellon University
UPMC Shadyside
Shadyside
Squirrel Hill North
The Bob O'Connor Golf Course at Schenley Park
Schenley Park
Duquesne University
Bluff
South Shore
Carson
Greenfield
South Side Slopes
Allentown
Mount


Pittsburgh
Zoning Map
Stowe Township
McKees Rocks
Marshall-Shadeland
Perry South
Spring Hill-City View
Northview Heights
Arsenal
Lower Lawrenceville
Highland Park
Lincoln-Lemington-Belmar
Garfield
Brunot Island
Windgap
Sheraden
Chateau
California-Kirkbride
Central Northside
Fineview
Spring Garden
Bloomfield
Friendship
East Liberty
Homewood West
Homewood North
Polish Hill
Strip District
Upper Hill
Shadyside
Point Breeze North
Fairywood
Allegheny West
North Shore
Middle Hill
North Oakland
Point Breeze
Peebles Square
Ingram
Crafton Heights
Elliott
Ohio River
Pittsburgh
Terrace Village
West Oakland
Squirrel Hill North
Central Oakland
Franklin Park
Whitney Park
Westwood
West End
Bluff
South Shore
South Oakland
Regent Square
Crafton
Mount Washington
Carson
Squirrel Hill South
Oakwood
Ridgemont
Allentown
South Side Slopes
Greenfield
Swisshelm Park
Swissvale
East Carnegie
Green Tree
Freeland St
Beltzhoover
Arlington
Hazelwood
Beechview
Knoxville
Banksville
Bon Air
St. Clair
Glen Hazel
Hays
West Homestead
Whitaker
E 8th Ave
1272 ft

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

Case Study: Apollo

Check out the “side pass” feature from the video:

Case Study: Apollo

Goal: Try to have a high-level understanding of how the **side pass feature** is built and integrated into the system.

Let's explore the code and the documentation of apollo to find parts associated with the **side pass feature**:

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

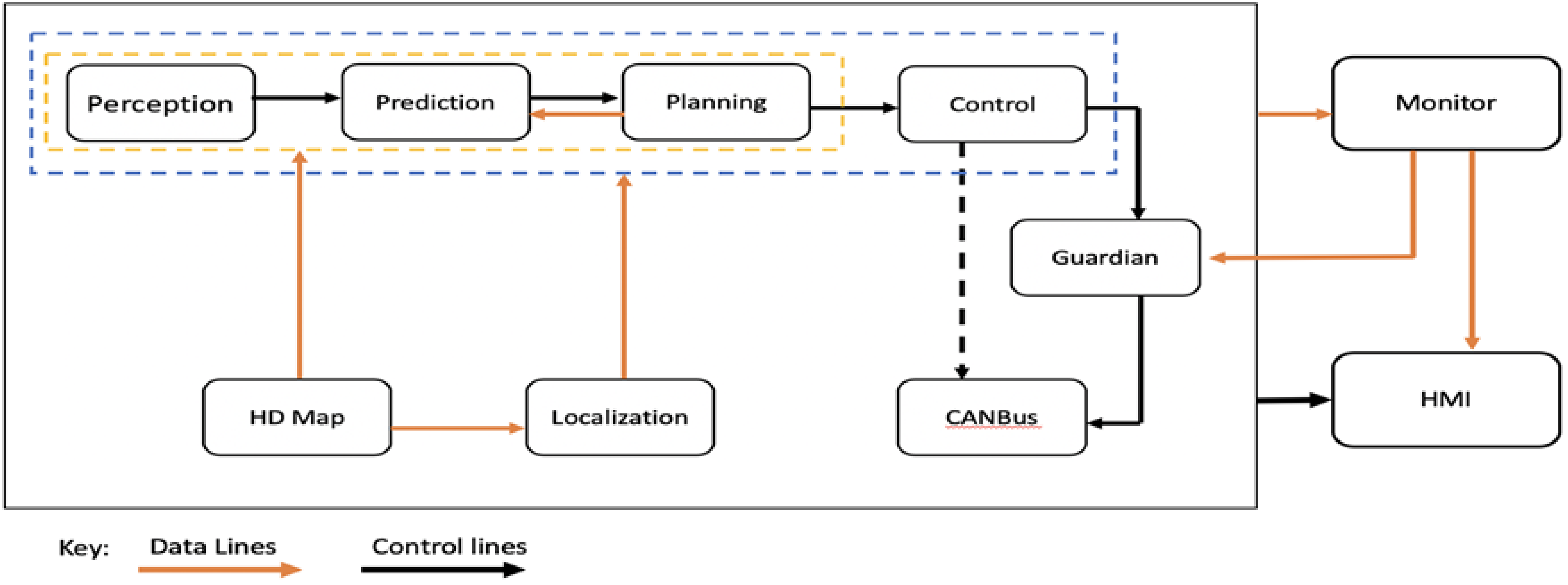
Doc: https://hidetoshi-furukawa.github.io/apollo-doxygen/index.html

Activity: Apollo

Discuss in teams of 3 - 4 on what parts are associated with the **side pass feature** based on the 6 diagrams in the handout:

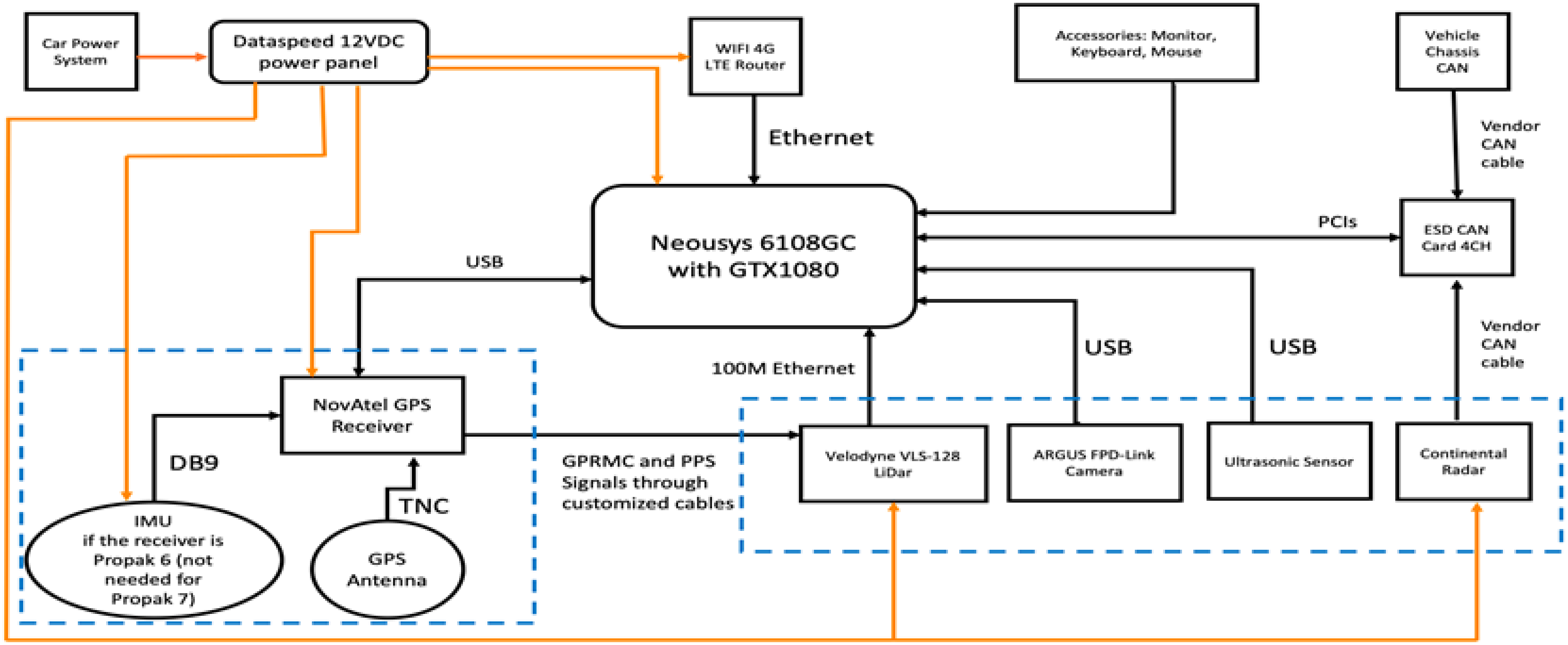
- Circle components that you think implement this feature.

Apollo Software Architecture



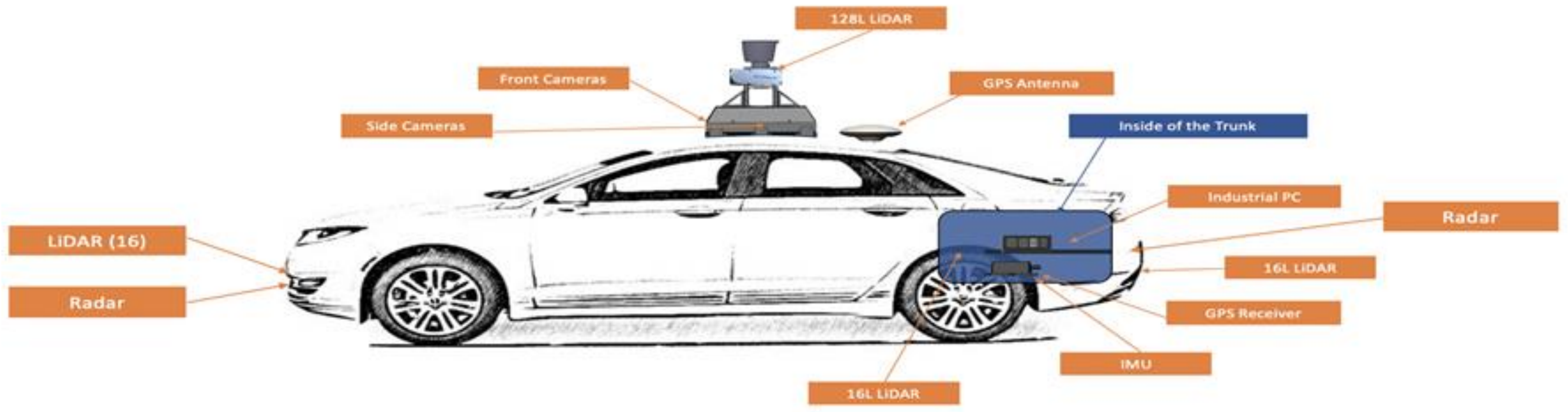
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/docs/specs/Apollo_5.5_Software_Architecture.md

Apollo Hardware Architecture



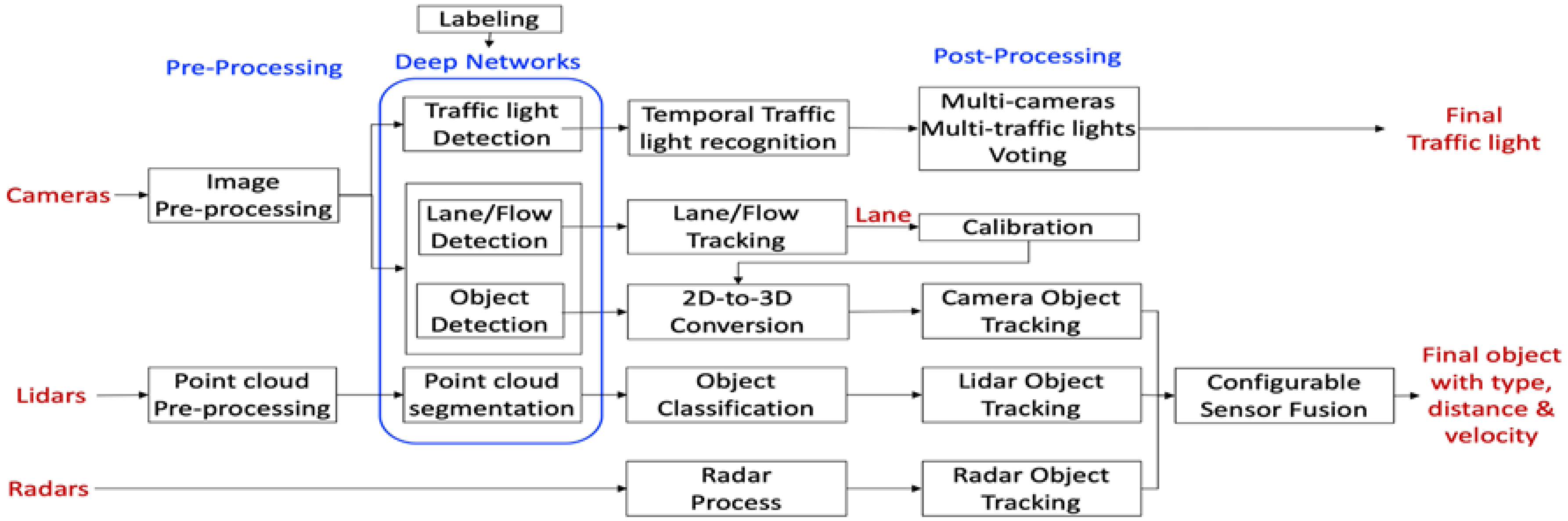
Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Hardware/Vehicle Overview

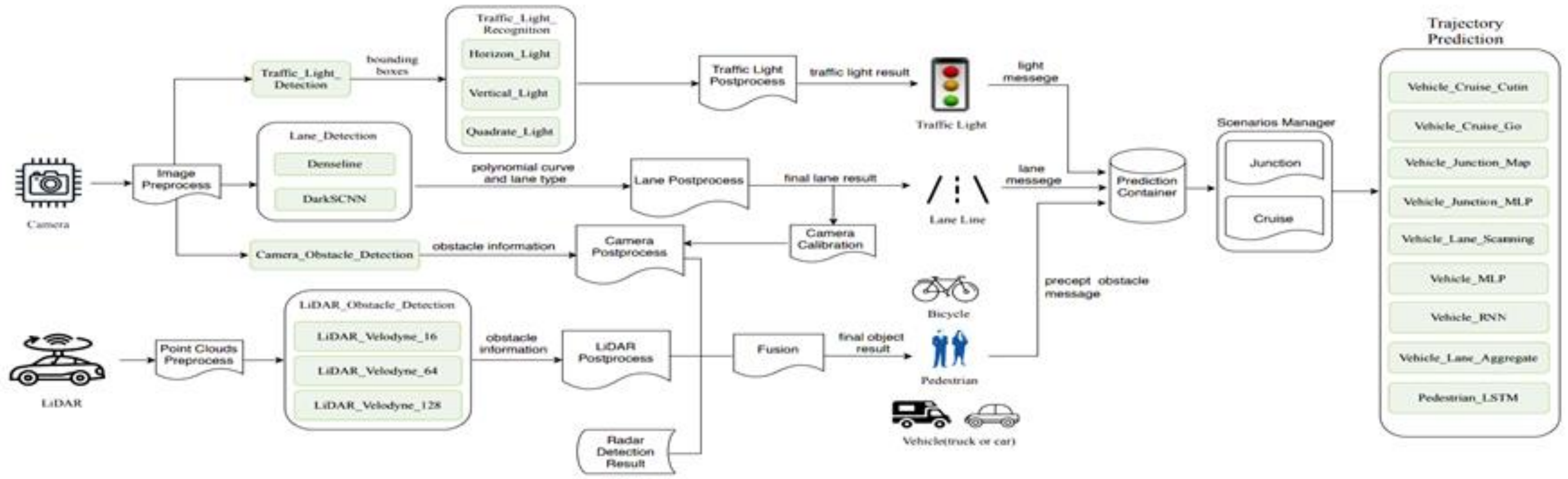


Source: https://github.com/ApolloAuto/apollo/blob/v6.0.0/README.md

Apollo Perception Module

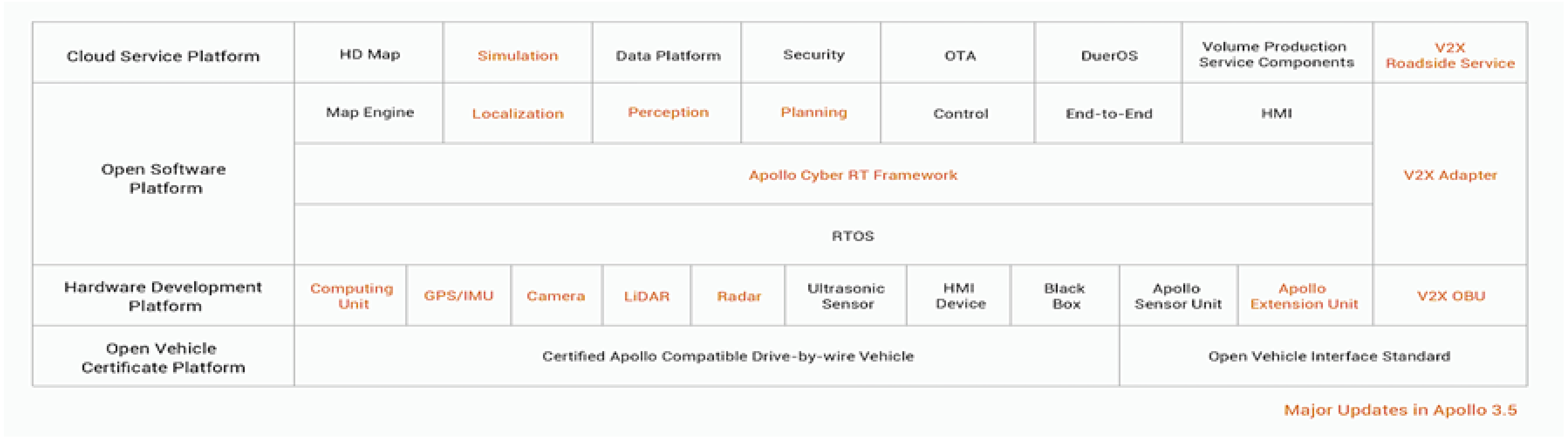


Apollo Machine Learning (ML) Models



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 Confer ence 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/

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

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!

Software Architecture

Abstraction

Elements: roles, responsibilities, behaviors, properties

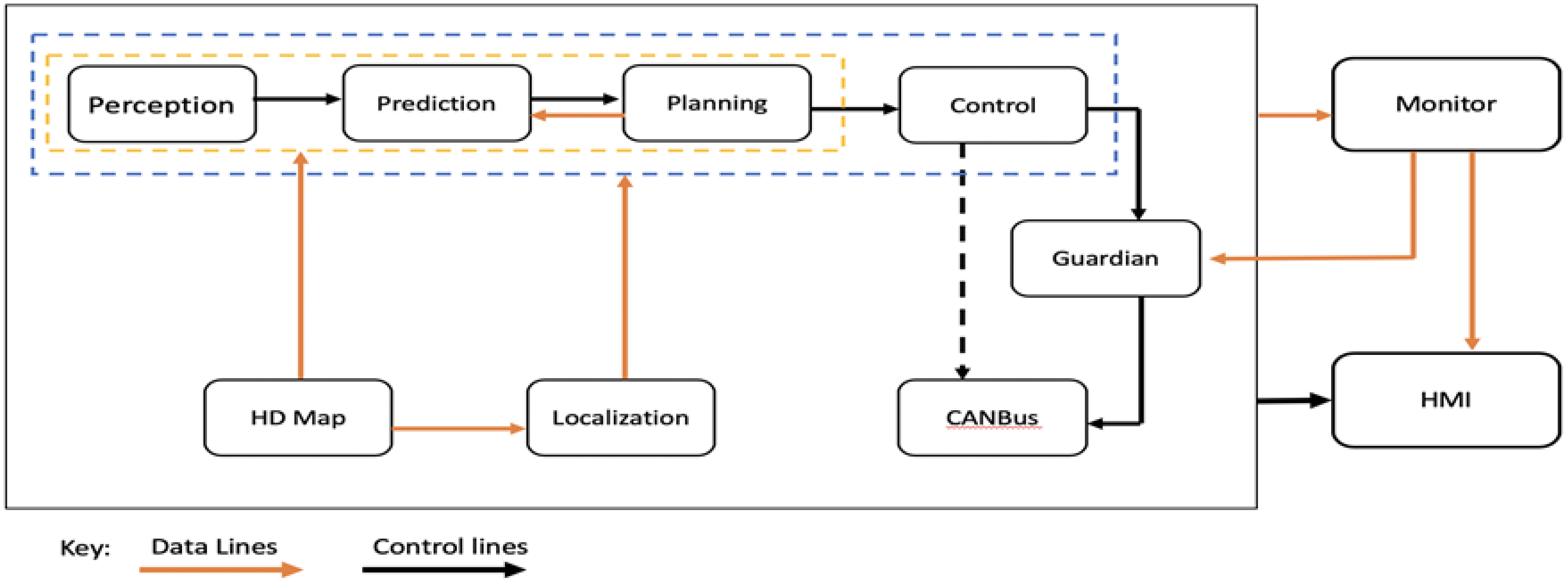
Relationships between elements

Relationships to non-software elements

- Hardware, external systems

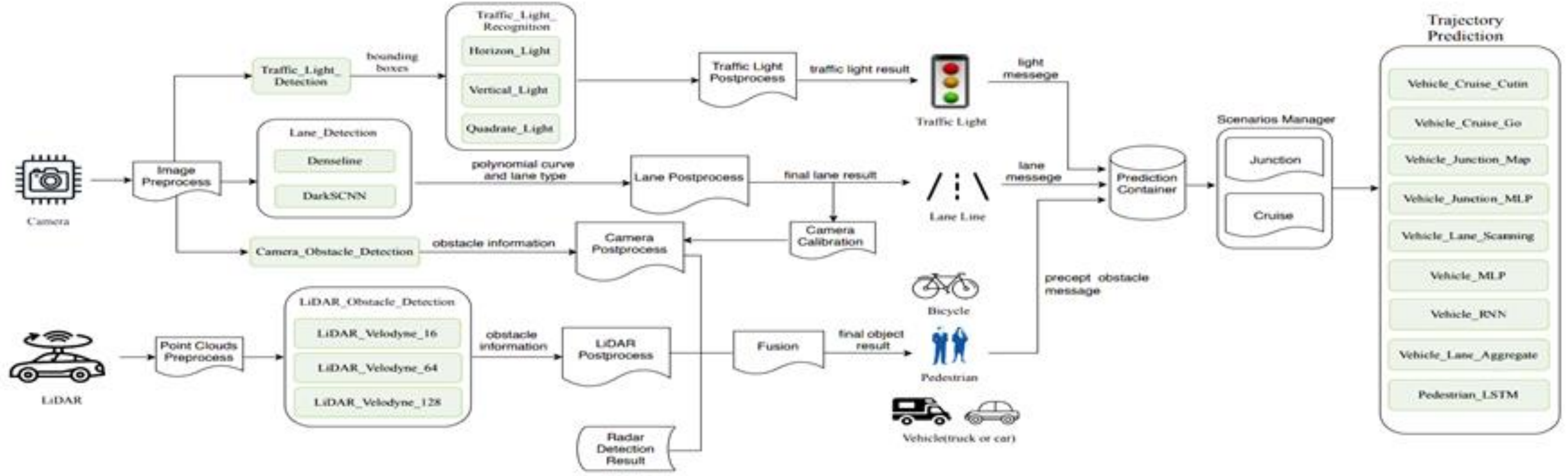
Described from many different perspectives (views)

Apollo Software Architecture



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

Apollo Machine Learning (ML) Models



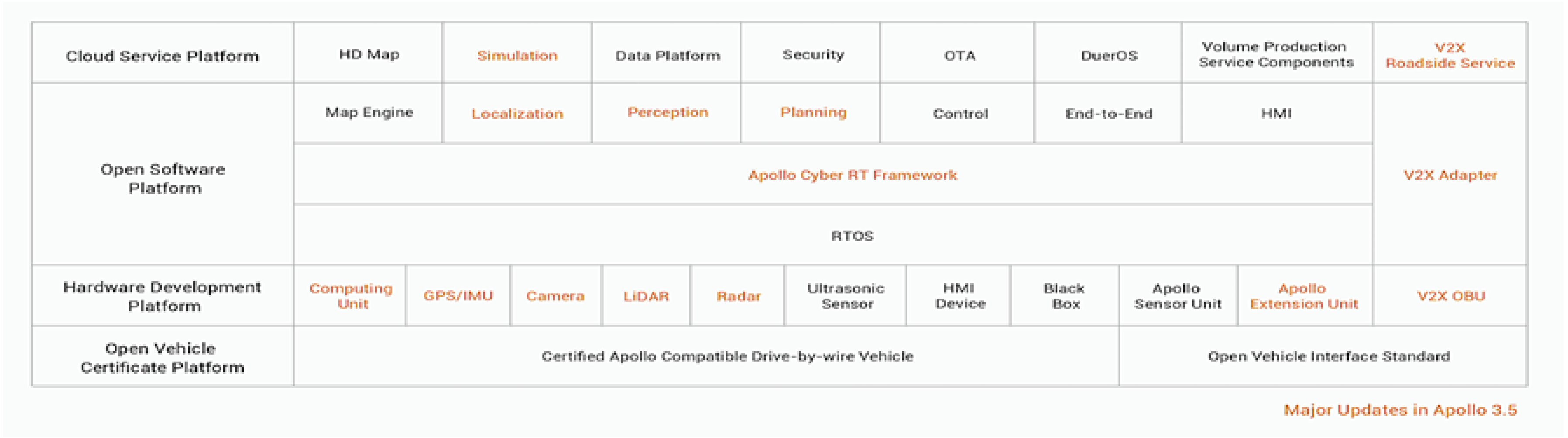
Source: Zi Peng, Jinqiu Yang, Tse-Hsun (Peter) Chen, and Lei Ma. 2020. A First Look at the Integration of Machine Learning Models in Complex Autonomous Driving Systems: A Case Study on Apollo. In Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations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

Software Architecture: Motivation

- Facilitates internal and external communication
- Describes design decisions and prescribes implementation constraints

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

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

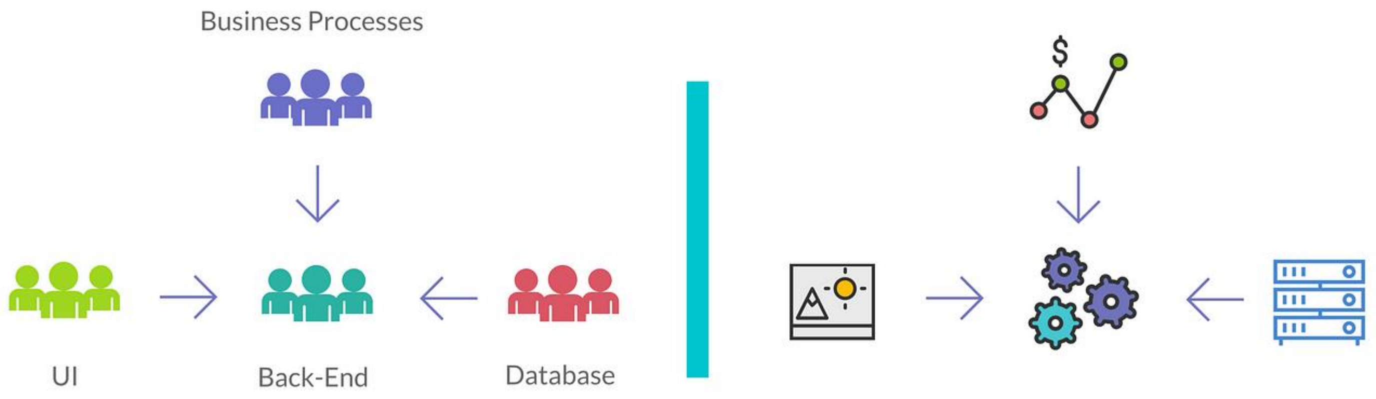
Source: https://github.com/ApolloAuto/

Software Architecture: Motivation

- Facilitates internal and external communication
- Describes design decisions and prescribes implementation constraints
- Relates to organizational structure

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

Conway’s Law



Company Structure

Software Architecture

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
- Control complexity
- Reason about and manage change
- Good basis for effort estimation
- …

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

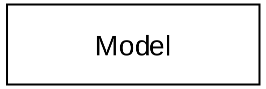
Design Questions

- 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?

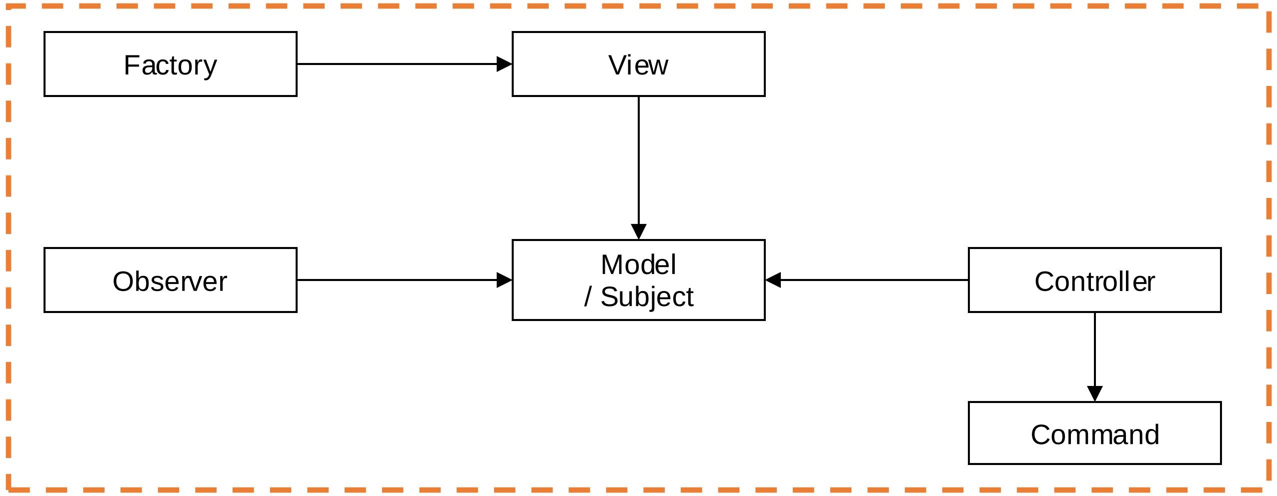
Architectural Questions

- 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?

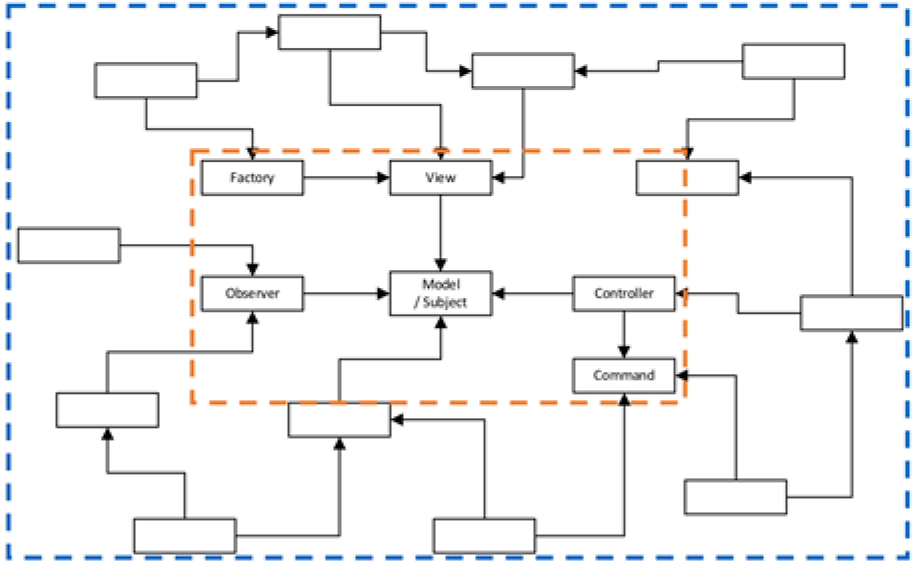
Objects



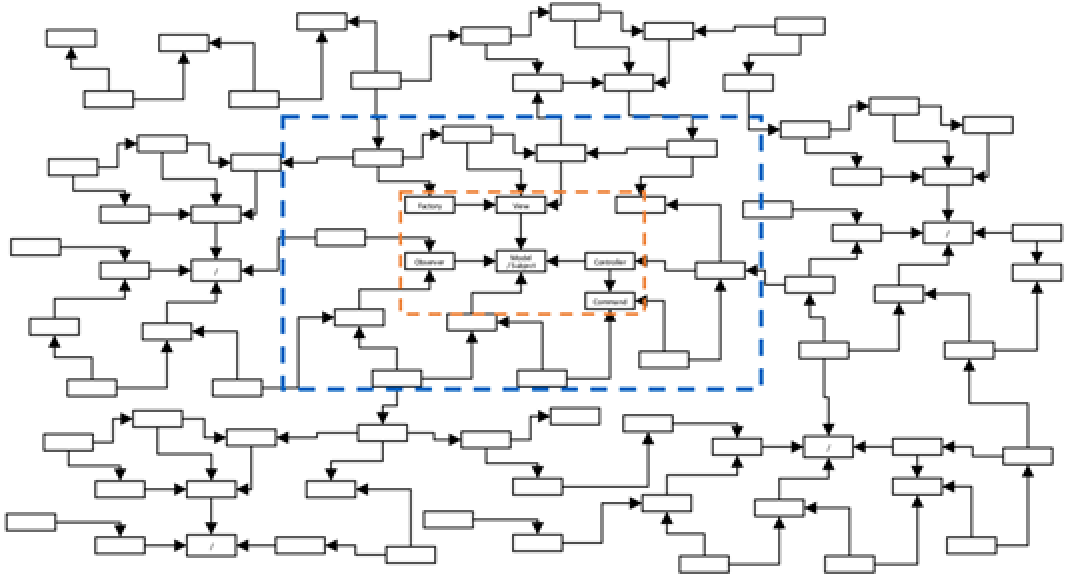
Design Patterns



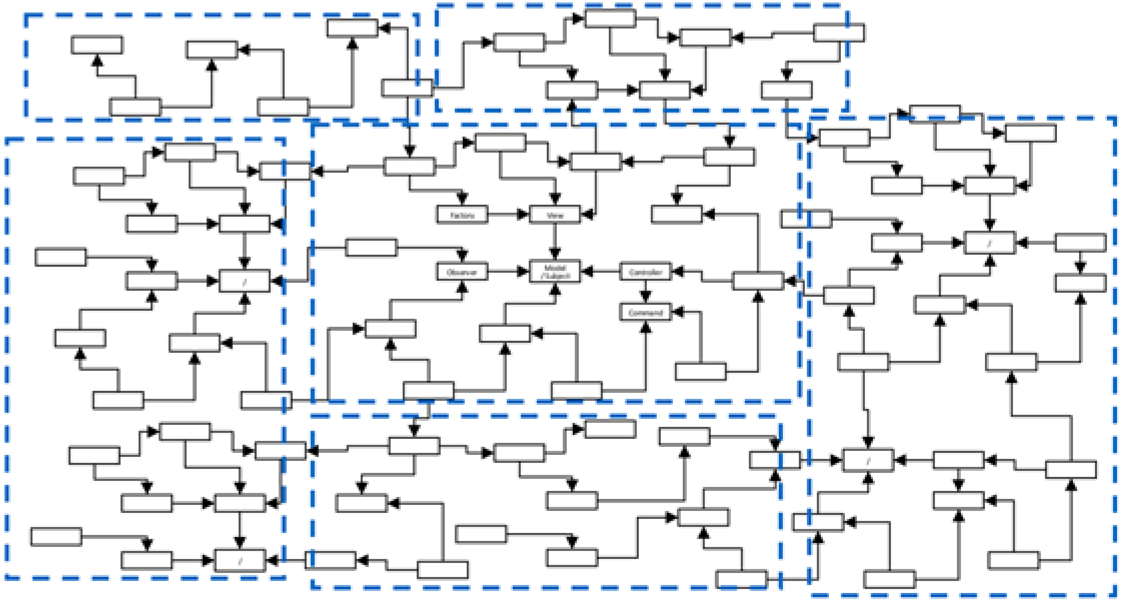
Design Patterns



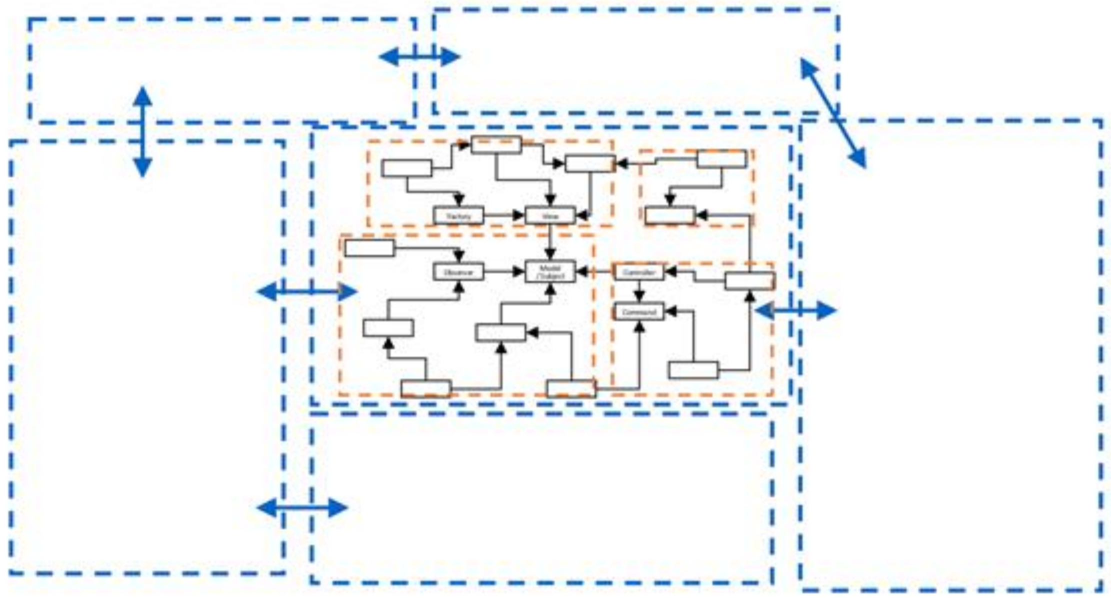
Design Patterns



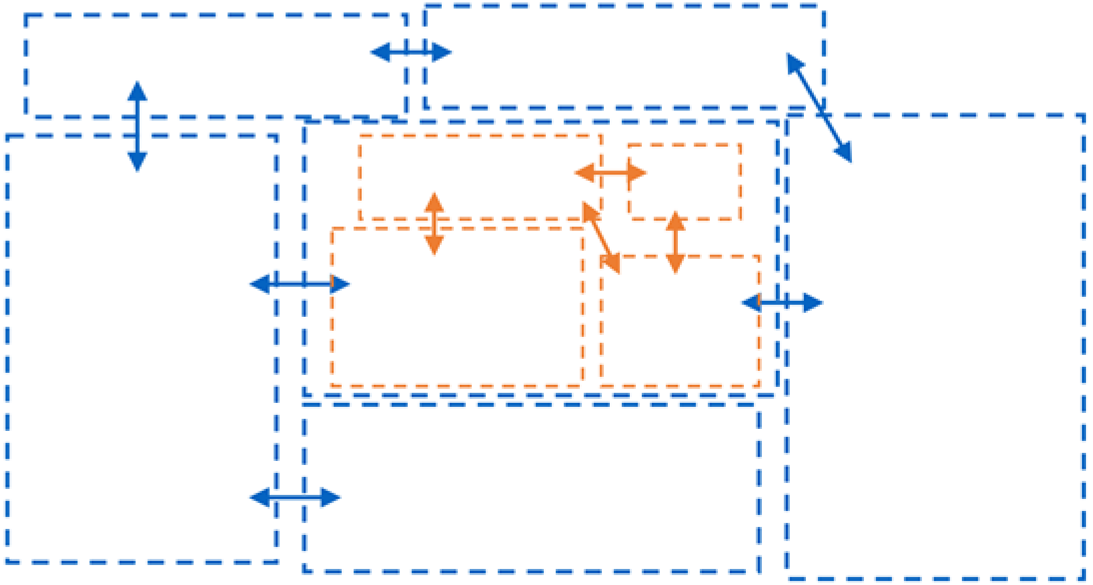
Architecture



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**

https://www.archdaily.com/

https://www.instagram.com/architectanddesign

https://www.mykonosceramica.com/

www.over-view.com

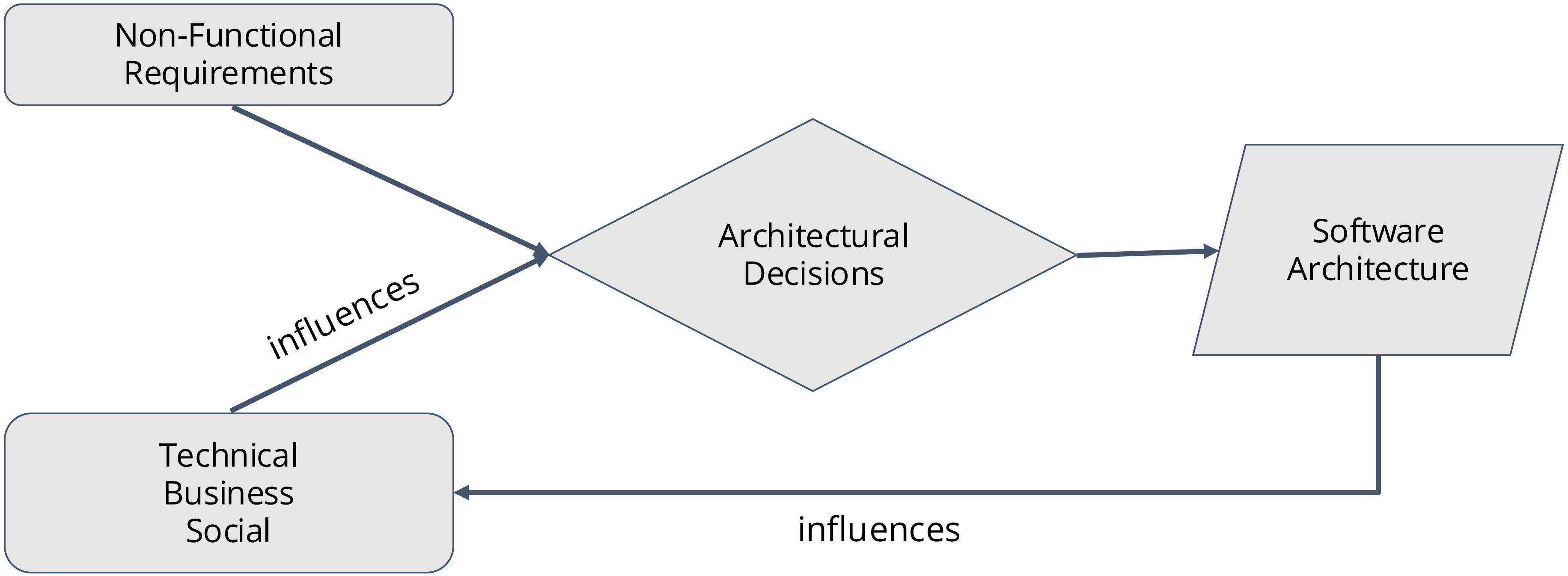
Every software system has an architecture

- 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!

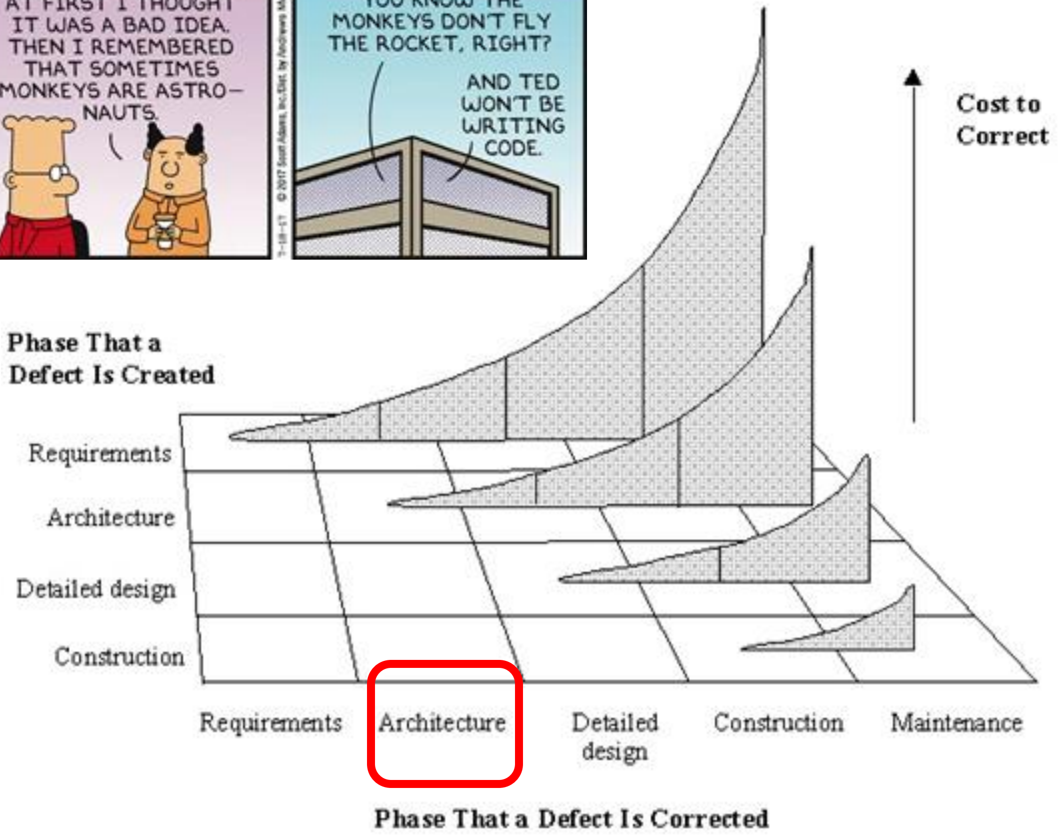
OUR SOFTWARE PRODUCT DOESN'T MEET STAKEHOLDERS' EXPECTATIONS
DID YOU CONSCIOUSLY MAKE ARCHITECTURAL DECISIONS TO MEET THEIR EXPECTATIONS?

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



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

I PROMOTED TED TO SOFTWARE ARCHITECT BECAUSE HE DOESN'T KNOW HOW TO CODE.
AT FIRST I THOUGHT IT WAS A BAD IDEA. THEN I REMEMBERED THAT SOMETIMES MONKEYS ARE ASTRONAUTS.
YOU KNOW THE MONKEYS DON'T FLY THE ROCKET, RIGHT?
AND TED WON'T BE WRITING CODE.
Dilbert.com
@ScottAdamsSays



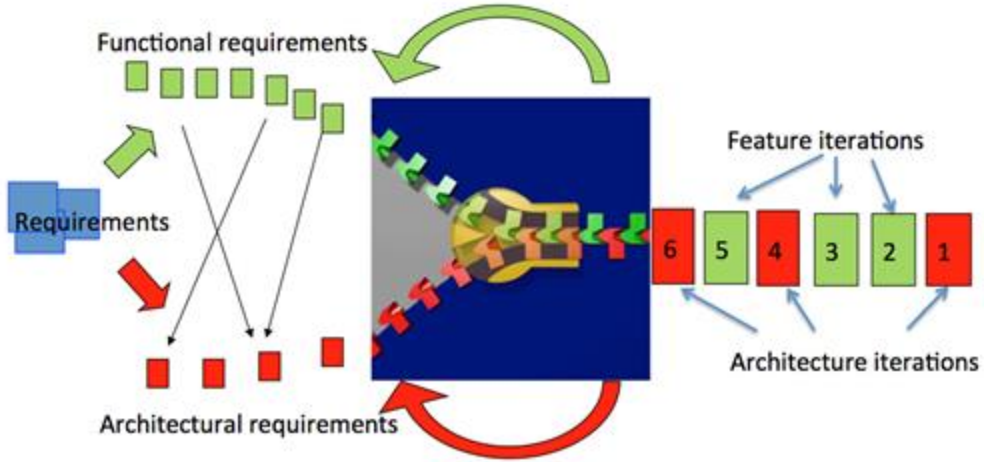
Copyright 1998 Steven C. McConnell. Reprinted with permission from *Software Project Survival Guide* (Microsoft Press, 1998).

Agile and Architecture

The Zipper Model

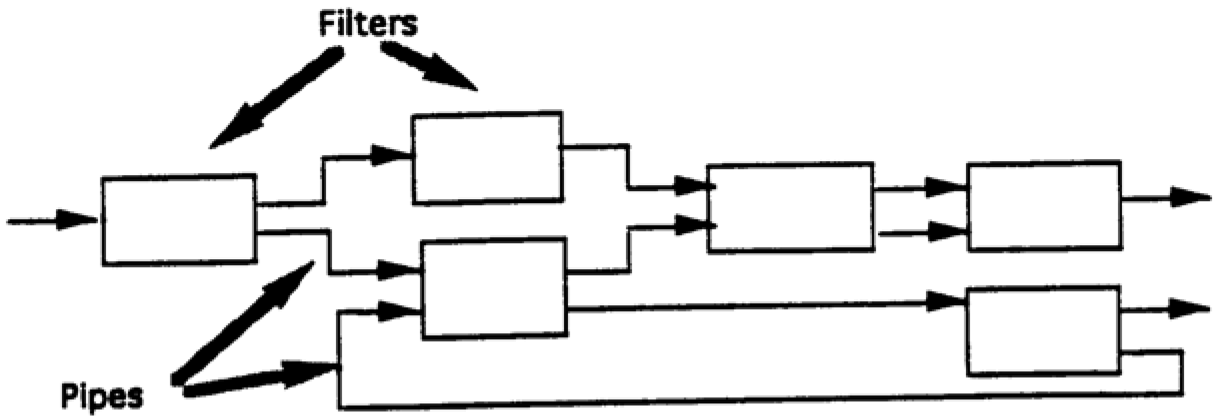
How to Agilely Architect an Agile Architecture

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



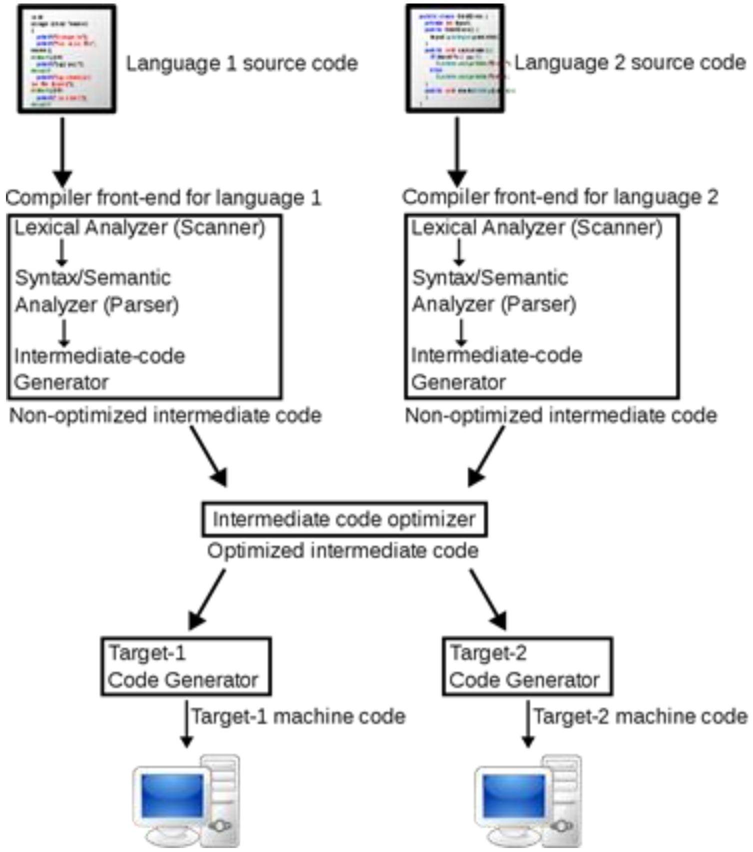
Common Software Architectures

1. Pipes and Filters

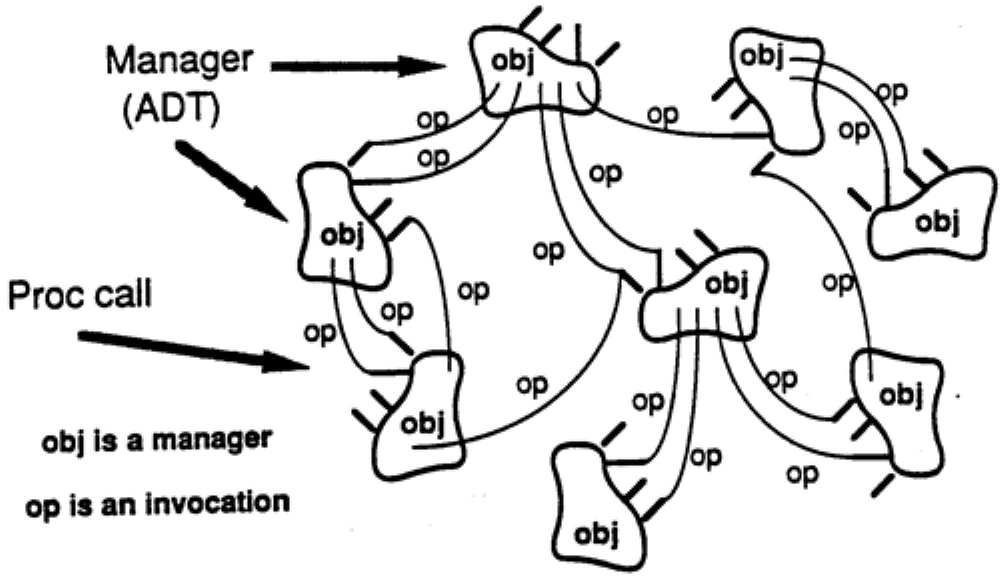


© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

Example: Compilers

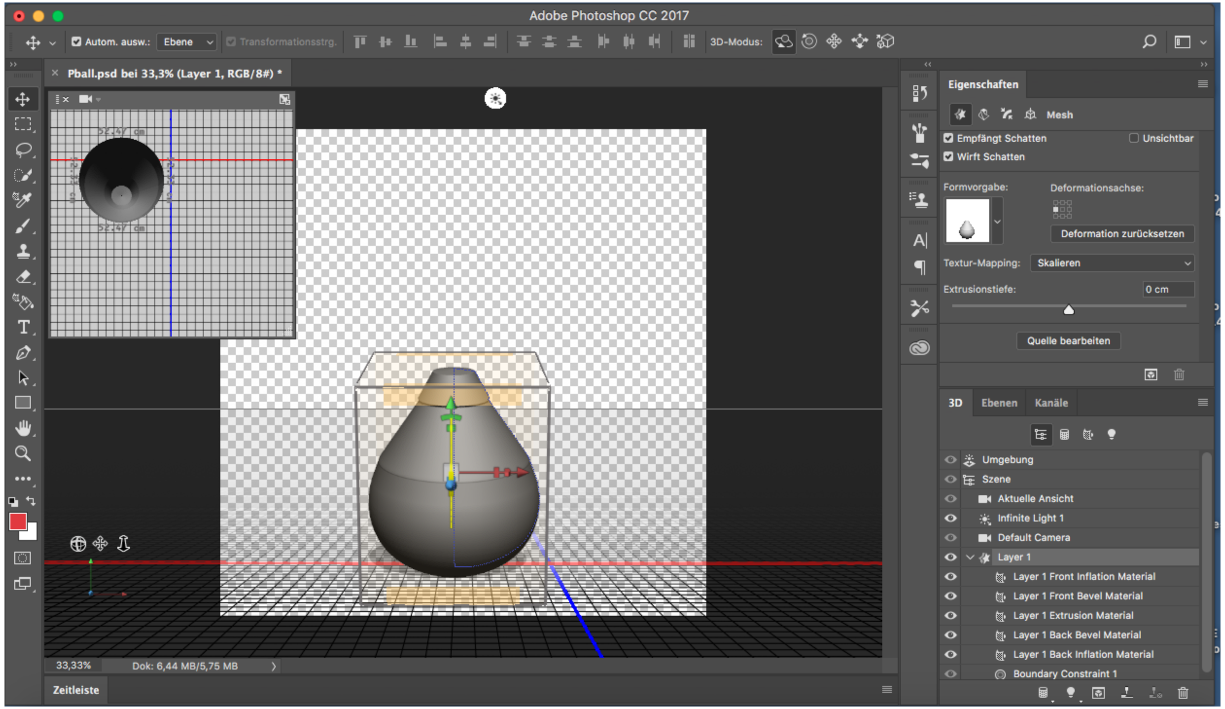

Language 1 source code
Language 2 source code
Compiler front-end for language 1
Compiler front-end for language 2
Lexical Analyzer (Scanner)
Syntax/Semantic Analyzer (Parser)
Intermediate-code Generator
Lexical Analyzer (Scanner)
Syntax/Semantic Analyzer (Parser)
Intermediate-code Generator
Non-optimized intermediate code
Non-optimized intermediate code
Intermediate code optimizer
Optimized intermediate code
Target-1 Code Generator
Target-2 Code Generator
Target-1 machine code
Target-2 machine code

2. Object-Oriented Organization



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

Example: Adobe Photoshop



Pball.psd bei 33,3% (Layer 1, RGB/8#) *

Autom. ausw.: Ebene

Empfängt Schatten [✓]

Wirft Schatten [✓]

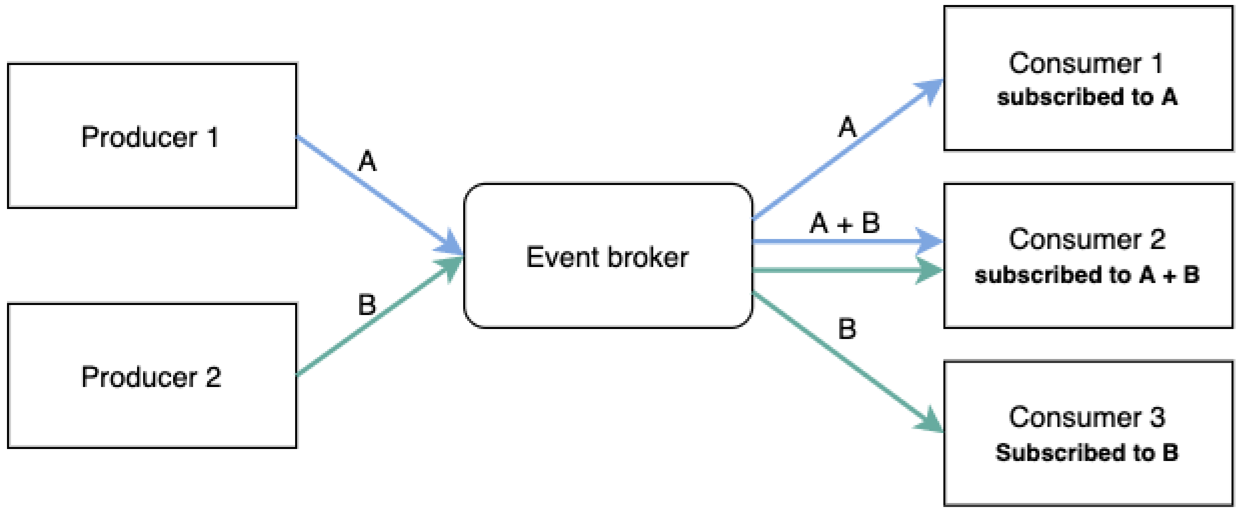
Unsichtbar [ ]

Textur-Mapping: Skalieren

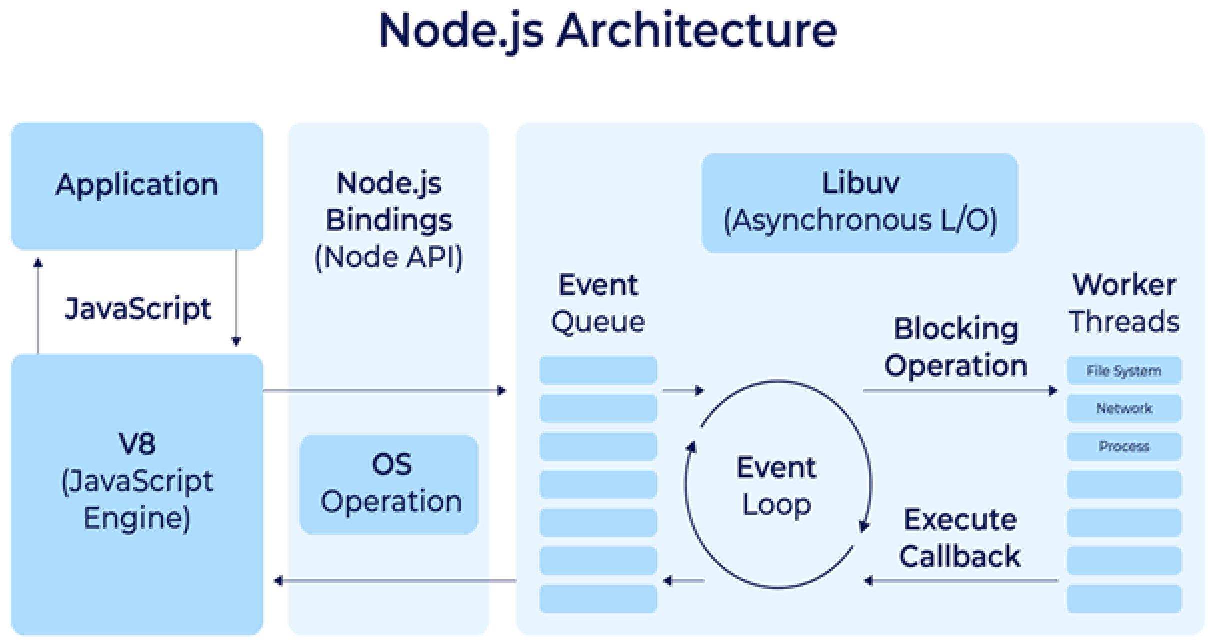
Extrusionstiefe: 0 cm

33,33% Dok: 6,44 MB/5,75 MB

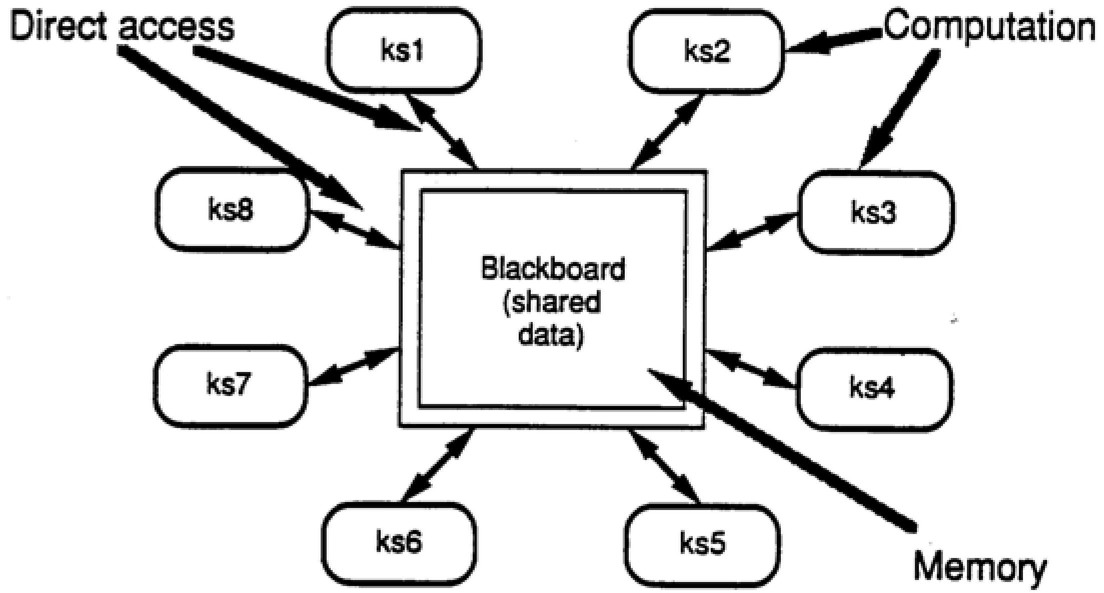
3. Event-Driven Architecture



Example: Node.js

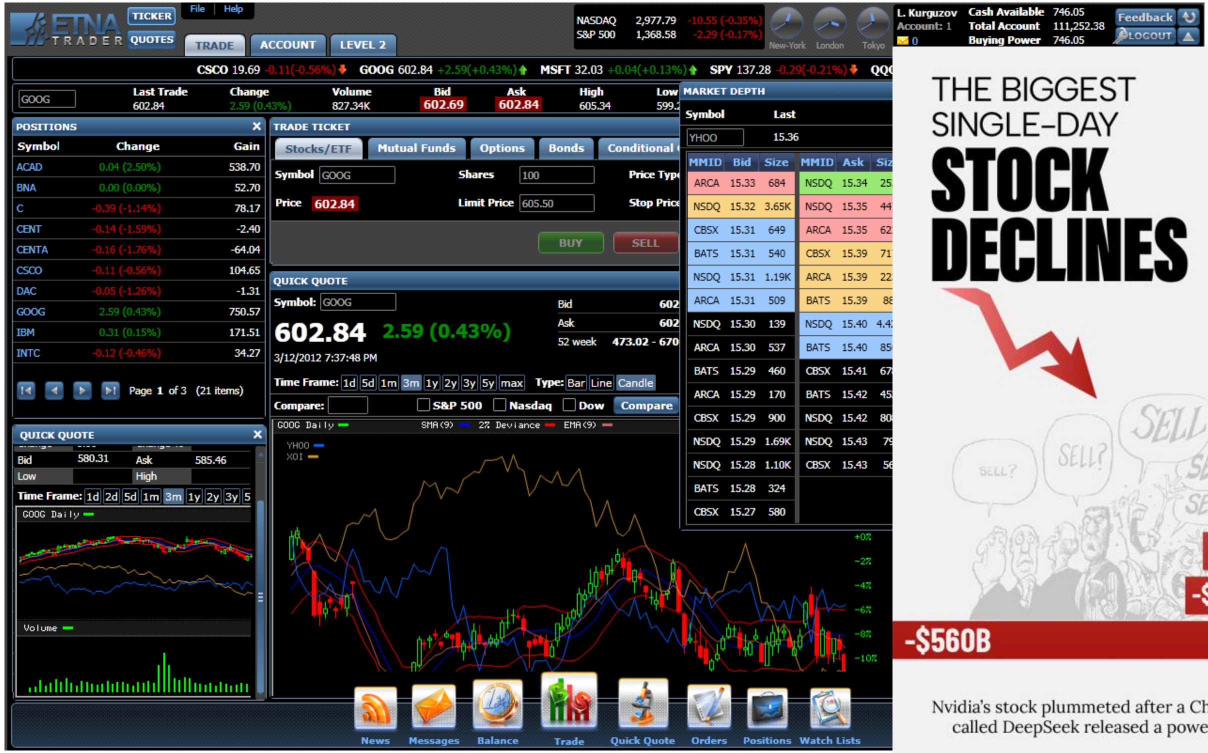


4. Blackboard Architecture



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

Example: Stock exchange



NASDAQ	2,977.79	-10.55 (-0.35%)
S&P 500	1,368.58	-2.29 (-0.17%)

L. Kurguzov Account: 1 ✉ 0

Cash Available	746.05
Total Account	111,252.38
Buying Power	746.05

CSCO 19.69 -0.11(-0.56%) GOOG 602.84 +2.59(+0.43%) MSFT 32.03 +0.04(+0.13%) SPY 137.28 -0.29(-0.21%) QQ

GOOG	Last Trade	Change	Volume	Bid	Ask	High	Low
	602.84	2.59 (0.43%)	827.34K	602.69	602.84	605.34	599.2

POSITIONS

Symbol	Change	Gain
ACAD	0.04 (2.50%)	538.70
BNA	0.00 (0.00%)	52.70
C	-0.39 (-1.14%)	78.17
CENT	-0.14 (-1.59%)	-2.40
CENTA	-0.16 (-1.76%)	-64.04
CSCO	-0.11 (-0.56%)	104.65
DAC	-0.05 (-1.26%)	-1.31
GOOG	2.59 (0.43%)	750.57
IBM	0.31 (0.15%)	171.51
INTC	-0.12 (-0.46%)	34.27

Page 1 of 3 (21 items)

QUICK QUOTE

Bid	580.31	Ask	585.46
Low		High	

Time Frame: 1d 2d 5d 1m 3m 1y 2y 3y 5

GOOG Daily
Volume

TRADE TICKET

Stocks/ETF | Mutual Funds | Options | Bonds | Conditional

Symbol GOOG Shares 100 Price Typ
Price 602.84 Limit Price 605.50 Stop Price

QUICK QUOTE

Symbol: GOOG

602.84 2.59 (0.43%)

3/12/2012 7:37:48 PM

Bid	602
Ask	602
52 week	473.02 - 670

Time Frame: 1d 5d 1m 3m 1y 2y 3y 5y max Type: Bar Line Candle

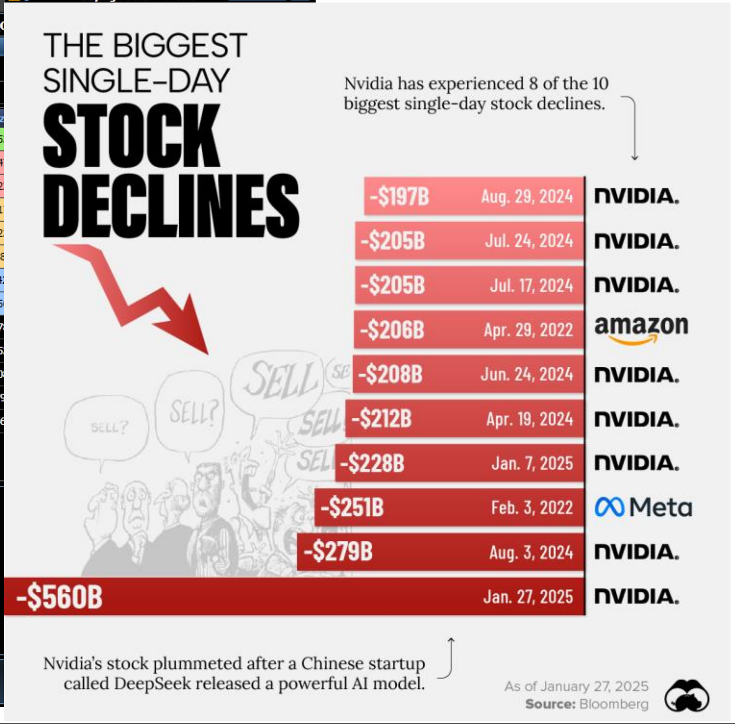
Compare: ☐ S&P 500 ☐ Nasdaq ☐ Dow Compare

GOOG Daily SMA(9) 2% Deviance EMA(9)
YHOO
XOI
+0% -2% -4% -6% -8% -10%

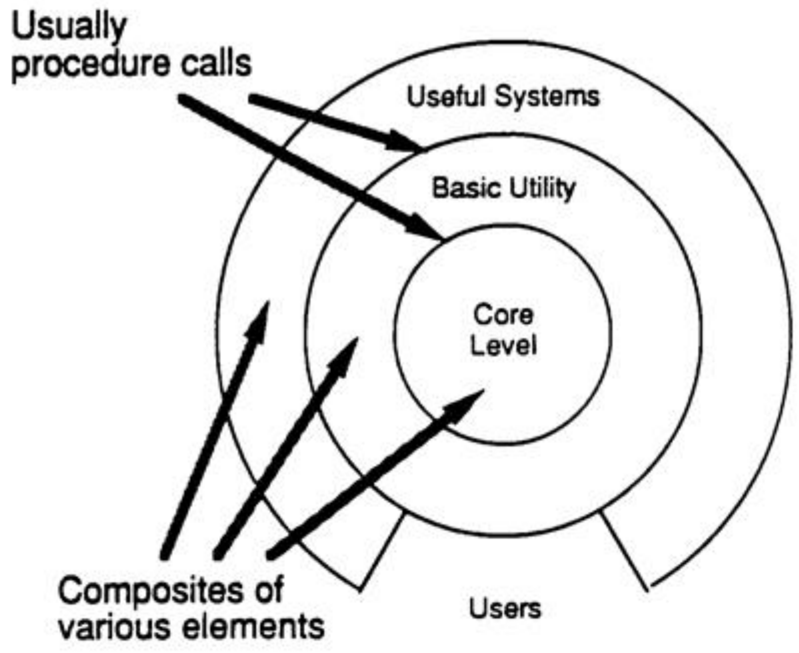
MARKET DEPTH

Symbol	Last
YHOO	15.36

MMID	Bid	Size	MMID	Ask	Siz
ARCA	15.33	684	NSDQ	15.34	25
NSDQ	15.32	3.65K	NSDQ	15.35	44
CBSX	15.31	649	ARCA	15.35	62
BATS	15.31	540	CBSX	15.39	71
NSDQ	15.31	1.19K	ARCA	15.39	22
ARCA	15.31	509	BATS	15.39	88
NSDQ	15.30	139	NSDQ	15.40	4.4
ARCA	15.30	537	BATS	15.40	85
BATS	15.29	460	CBSX	15.41	67
ARCA	15.29	170	BATS	15.42	45
CBSX	15.29	900	NSDQ	15.42	80
NSDQ	15.29	1.69K	NSDQ	15.43	79
NSDQ	15.28	1.10K	CBSX	15.43	56
BATS	15.28	324			
CBSX	15.27	580			

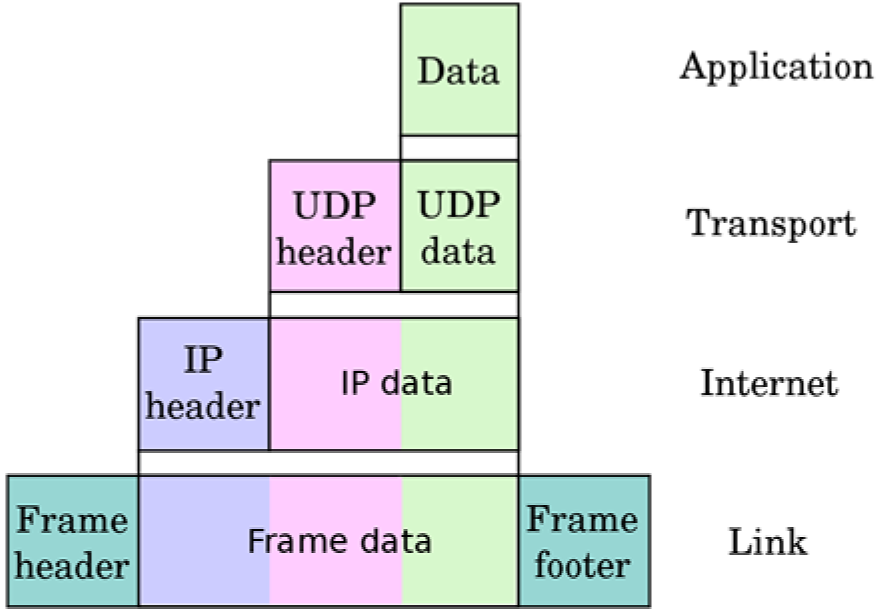


5. Layered Systems



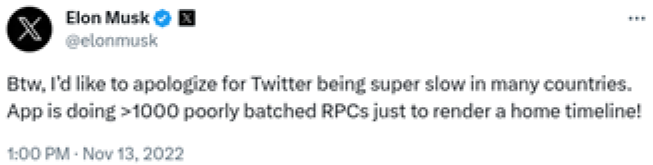
© David Garlan and Mary Shaw, CMU/SEI-94-TR-021

Example: Internet Protocol Suite



Why Document Architecture?

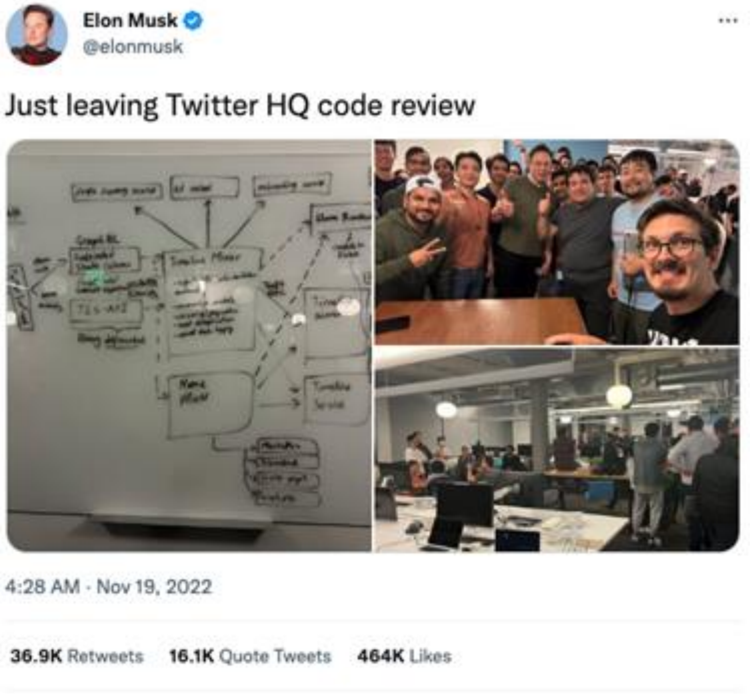
- Blueprint for the system
  - Artifact for early analysis
  - Primary carrier of quality attributes
  - Key to post-deployment maintenance and enhancement
- Documentation speaks for the architect, today and 20 years from today
  - As long as the system is built, maintained, and evolved according to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



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

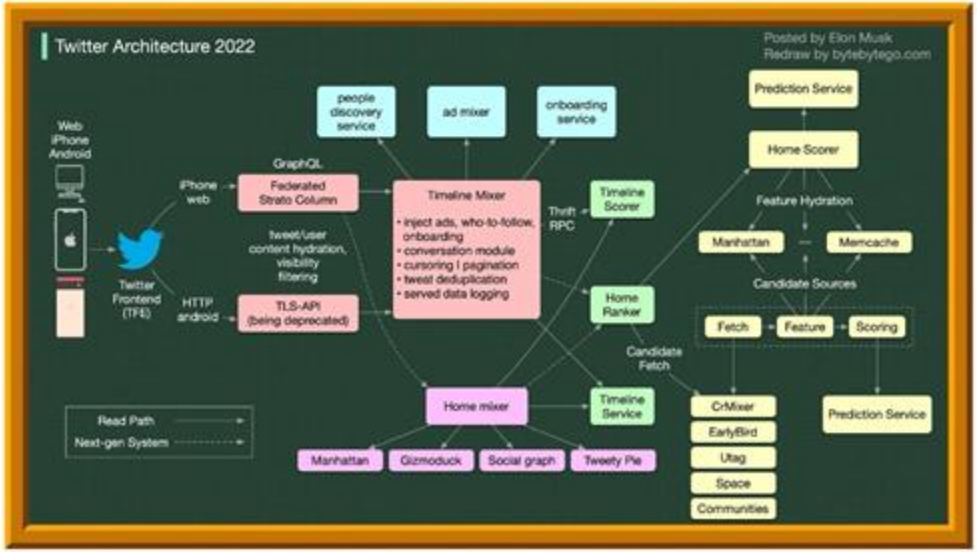


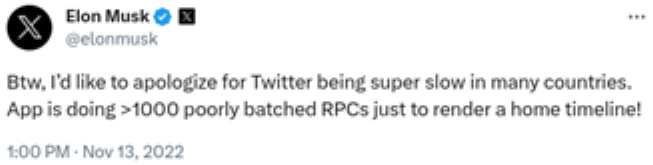
Elon Musk
@elonmusk

Just leaving Twitter HQ code review

4:28 AM · Nov 19, 2022

36.9K Retweets 16.1K Quote Tweets 464K Likes



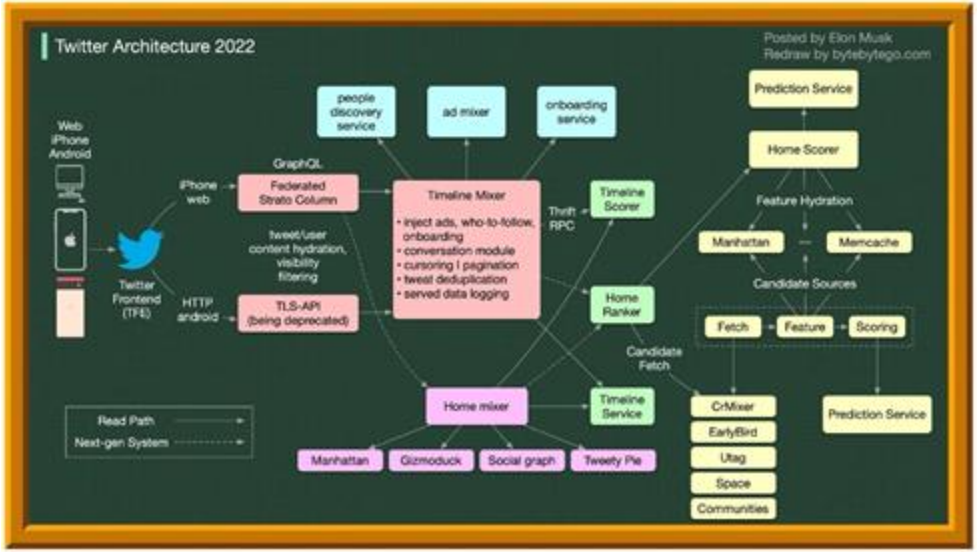


Elon Musk @elonmusk

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

From now on, y'all need to make sure you pay attention to the integration layer in order to map the domains to the interfaces within the virtual framework.
Who left the Software Architecture guide within the reach of the new executive?
Software Architecture



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