

Ardeim

*(Ar)chitecture, (De)sign &
(Im)plementation*

Copyright: SivaramaSundar (sivaramasundar@gmail.com), Karthik Kalkur (Karthikkalkur@tesco.com)

Thanks to: Rashmi Subbanna (Rashmi.Subbanna@tesco.com)

Credits: revealjs, stripthis ([kesiev](#))

Heads UP!



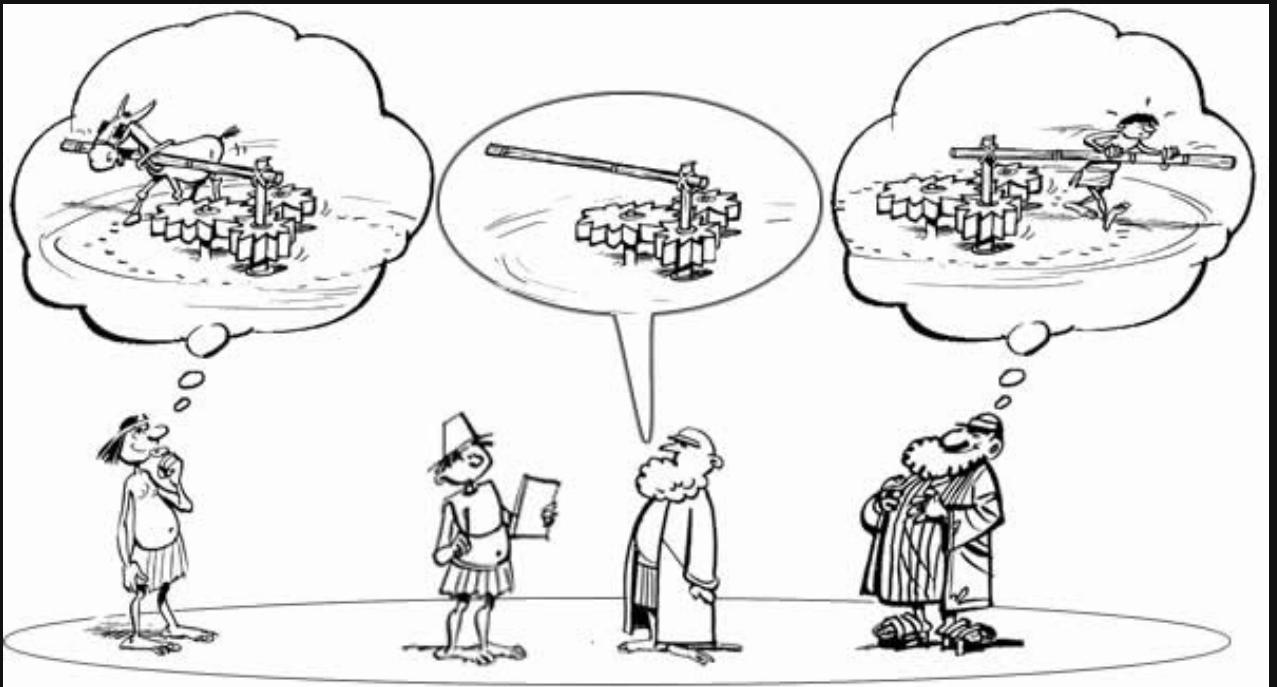
details

Raise relevant questions to understand the business benefits!

- Be Open and Make No Assumptions -

Heads UP!

What you think is what you get!

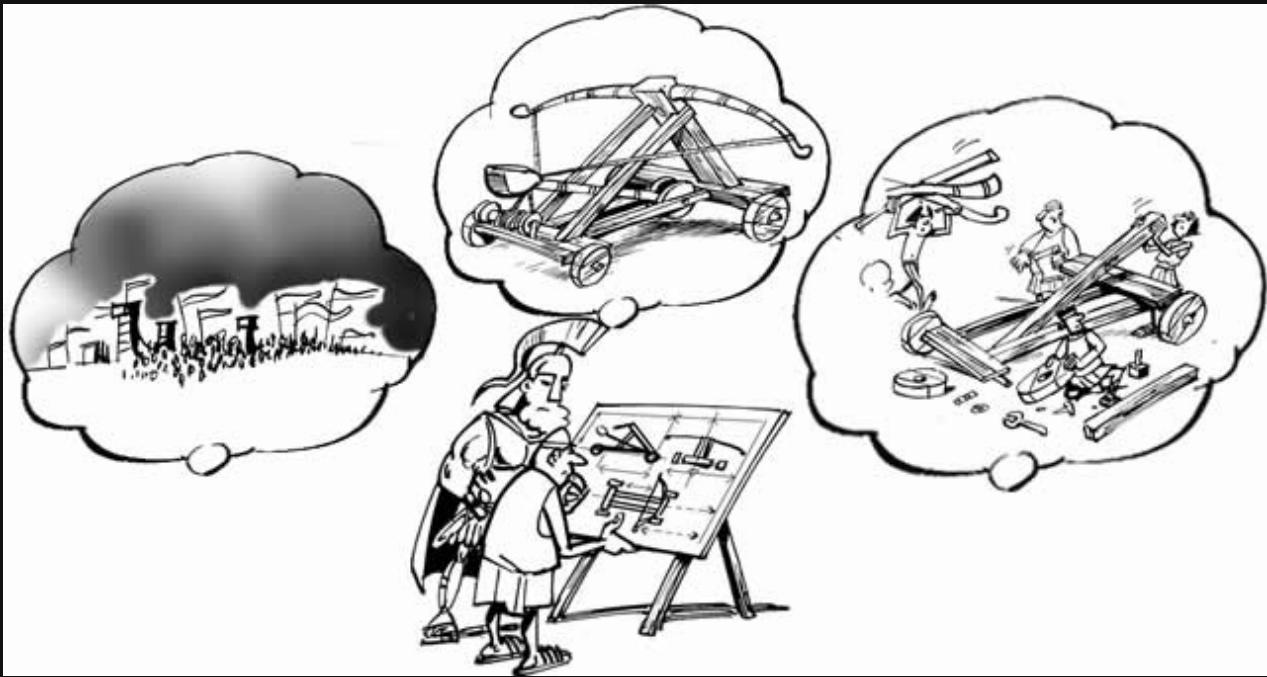


Specific & relevant details helps you think through a better solution!

- Stay Focused -

Heads UP!

Prespectives & Concerns Differ!



Understand Business Needs vs Functionalities vs Implementation
from the Stakeholder Perspective!

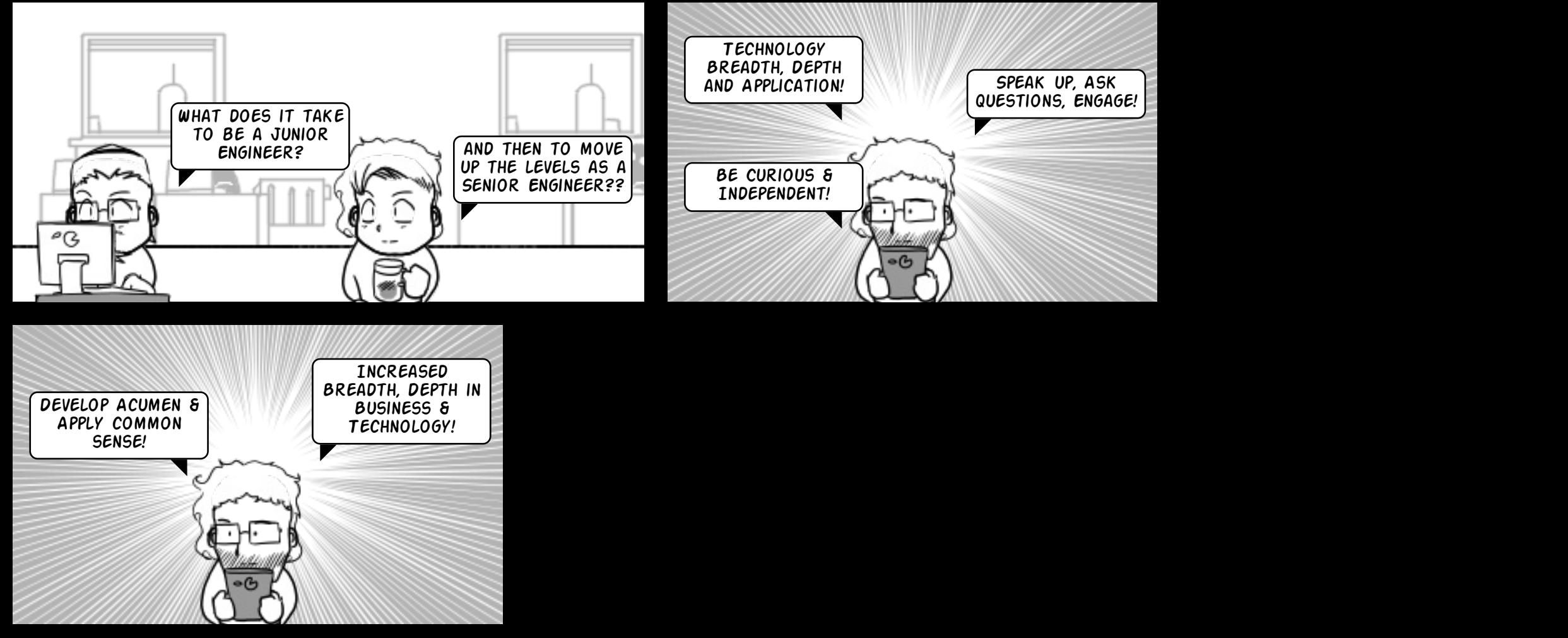
- Be an Enabler; Listen, Acknowledge & Accept Non Technical Needs! -

Context

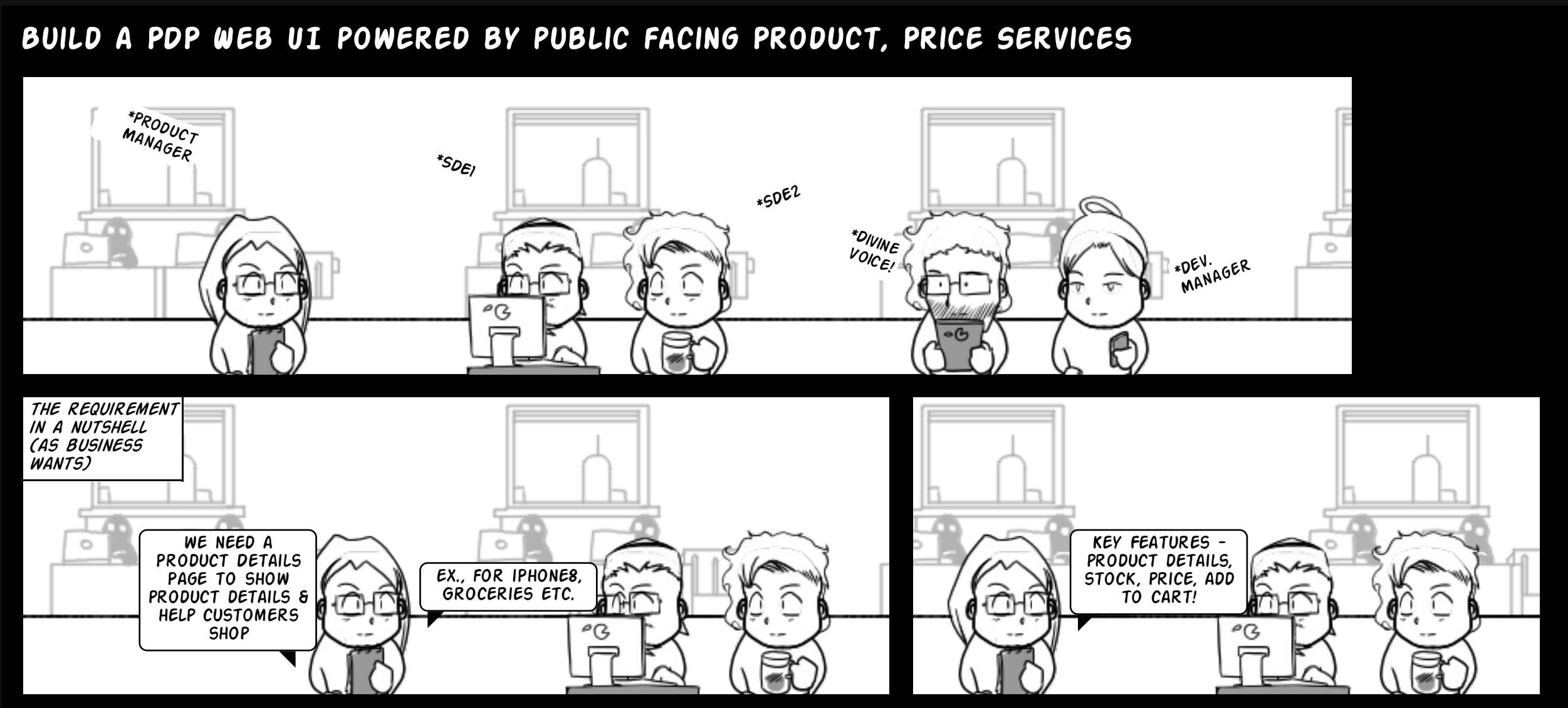


Context

THE ENGINEER'S DILEMMA



Context

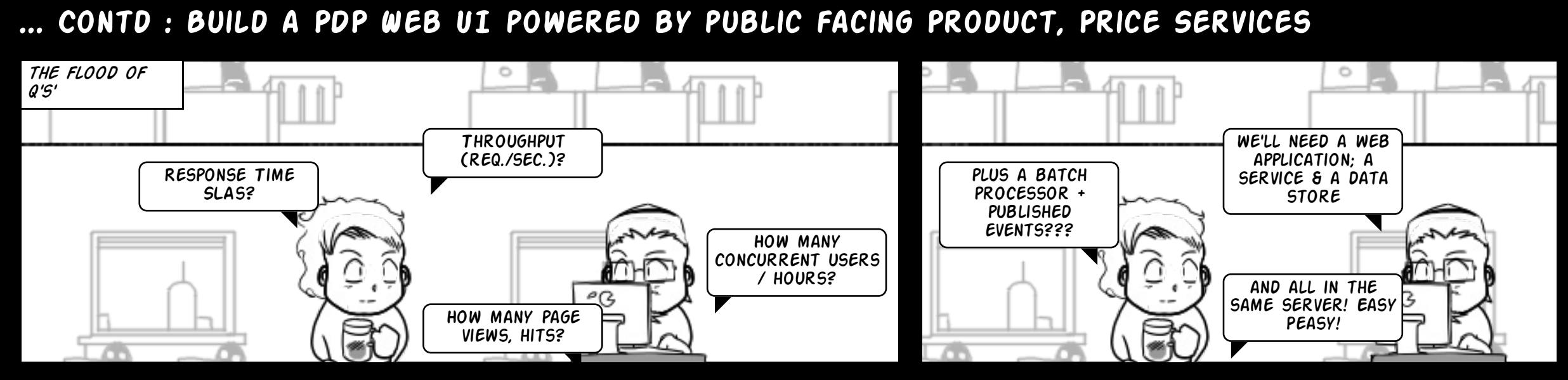


Context

... CONTD : BUILD A PDP WEB UI POWERED BY PUBLIC FACING PRODUCT, PRICE SERVICES

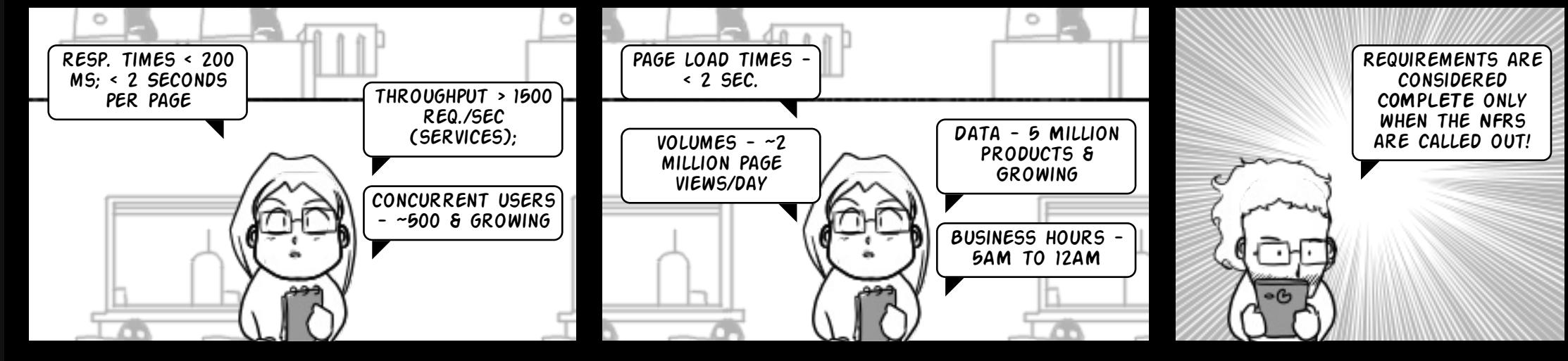


Context



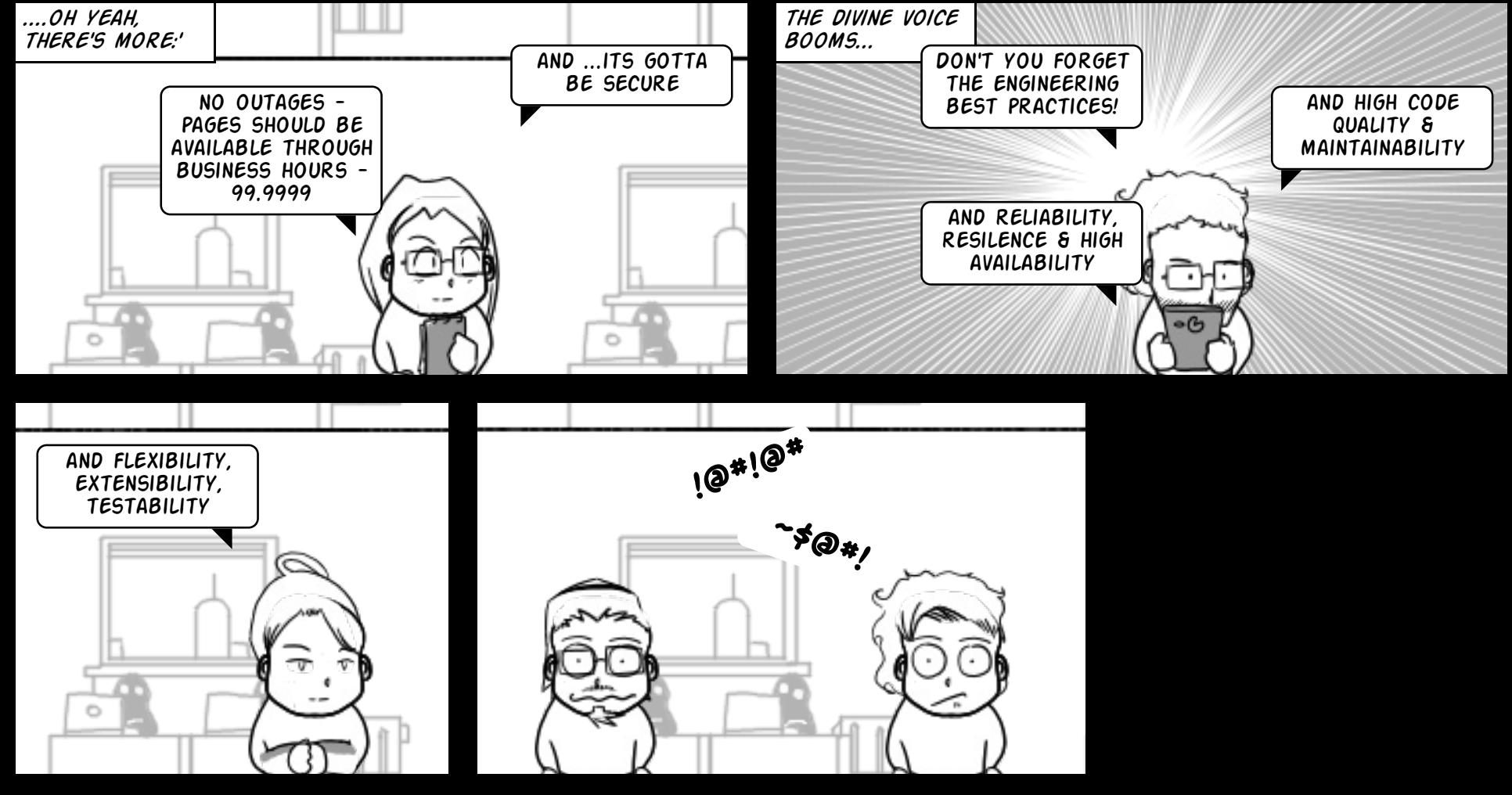
Context

... CONTD : BUILD A PDP WEB UI POWERED BY PUBLIC FACING PRODUCT, PRICE SERVICES



Context

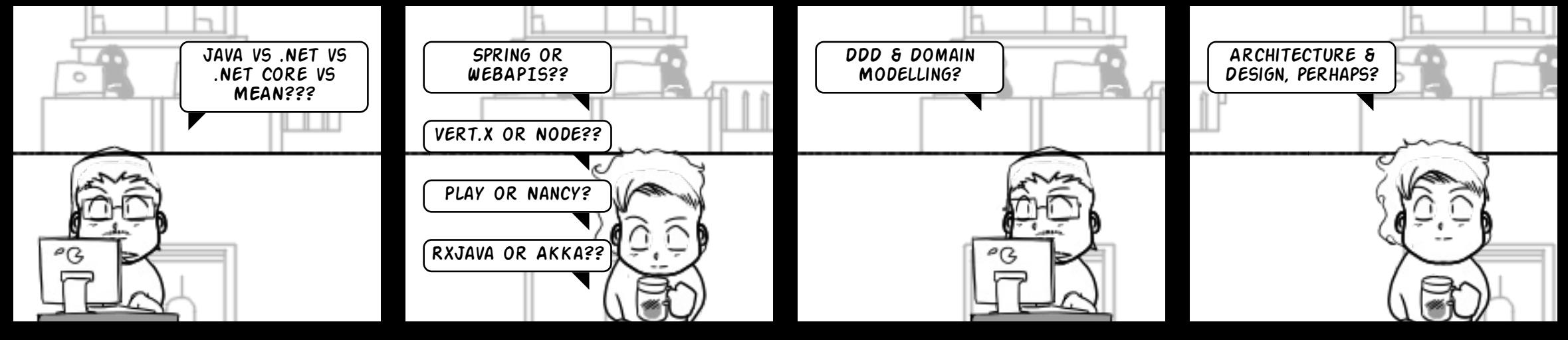
WAIT A MIN ... THERE'S MORE



Whew! So, What Next?

What next!?

TOO MUCH TO CHEW! WHERE DO WE START??!



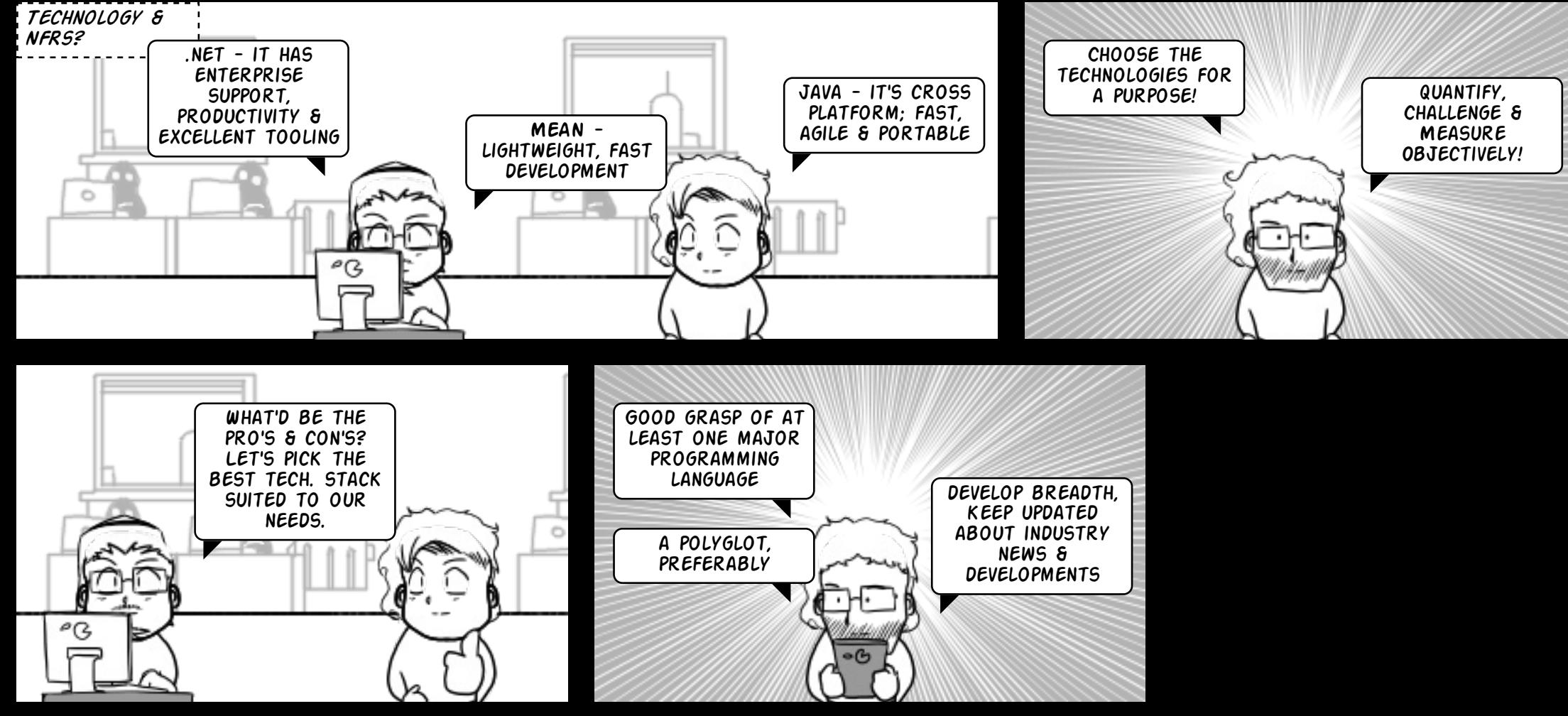
- Build Functionality first, then iterate addressing the NFRs -

What next!?



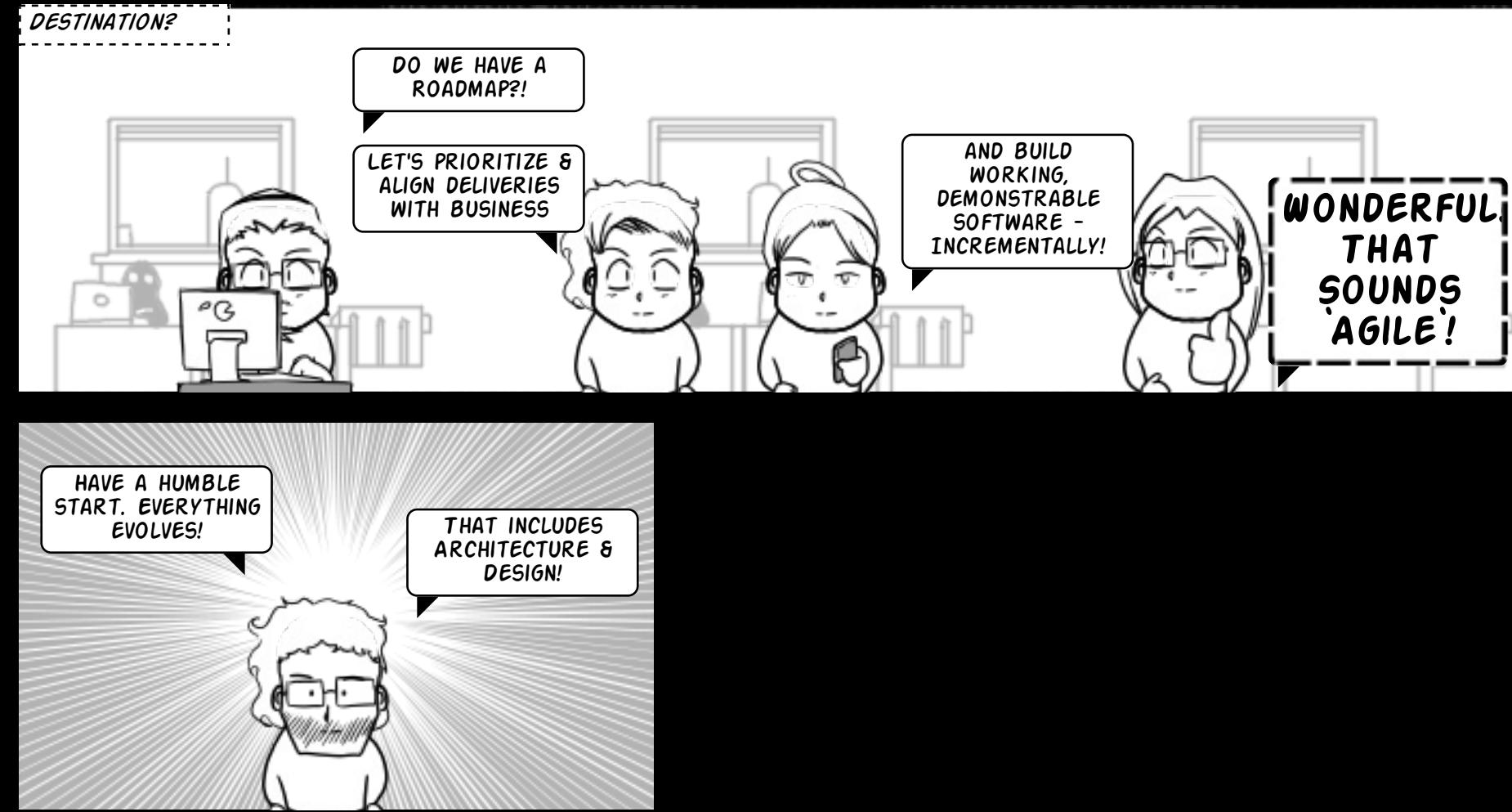
What next!?

TECHNOLOGY NEXT...



What next!?

DON'T LOSE SIGHT OF THE DESTINATION!



- Fret Not! Help is not far! -

What next!?

Data first, Design & Implementation follows next...

Core Concepts

- TDD, BDD, DDD

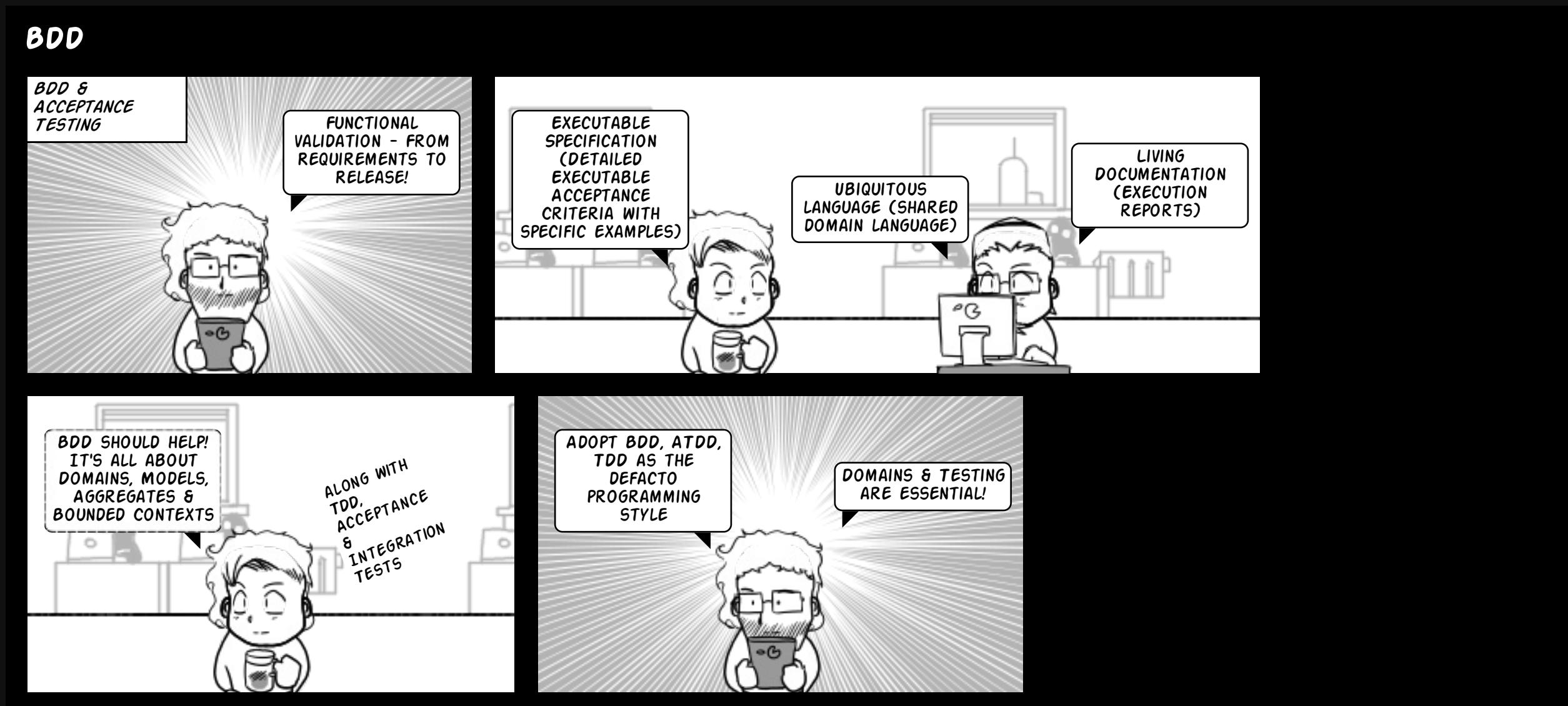
Architecture & Design

- Architectural Styles & Views
- Technology Selection (How & Why)
- Polyglot DataStores
- NFRs & Tradeoffs

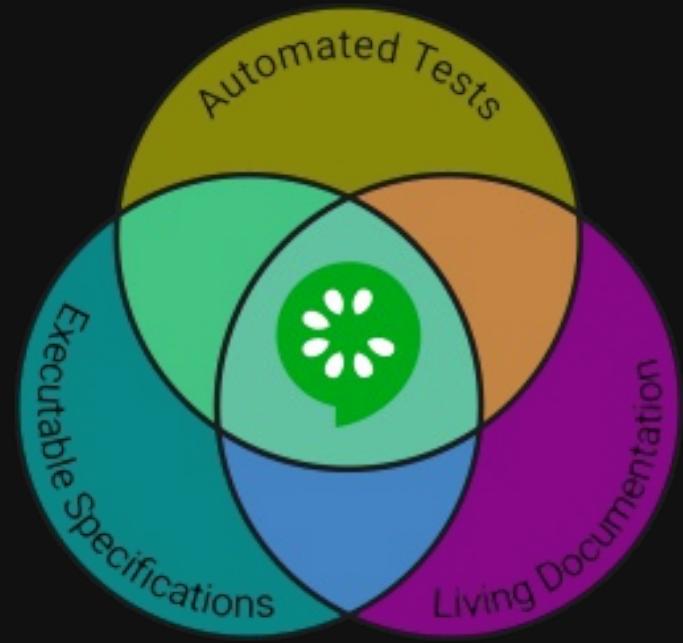
Engineering

- Best Practices
- Design Patterns
- Design Principles (SOLID)
- Code Smells
- CI-CD & DevOPS first culture
- Programming Paradigms (Functional, Imperative)

Core Concepts ...

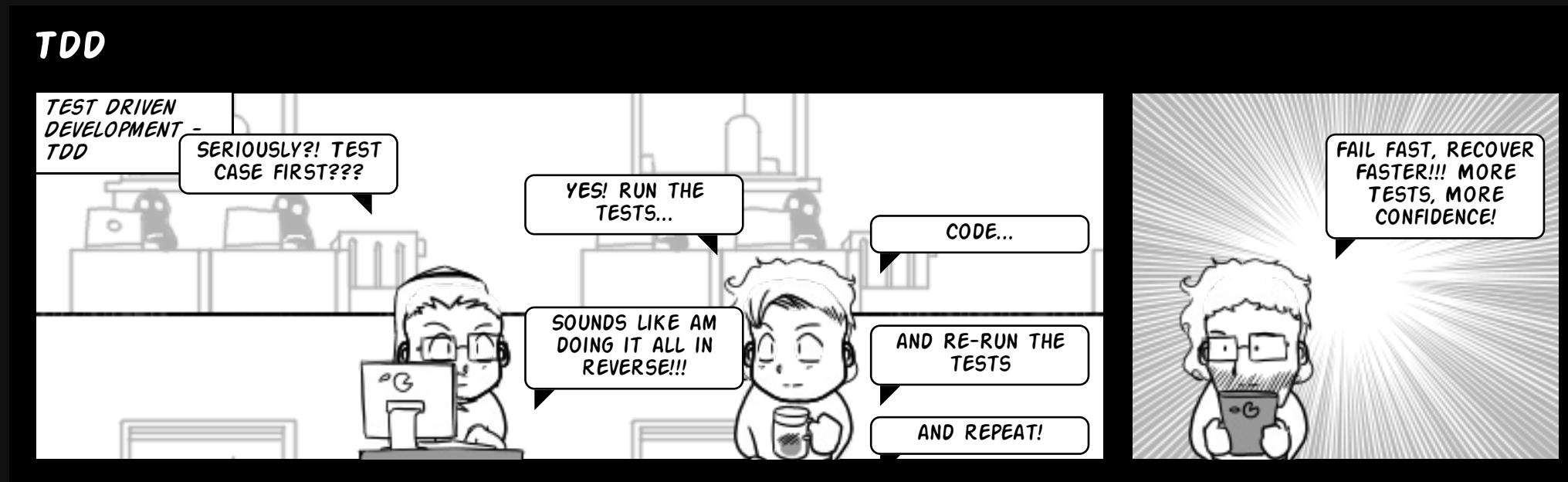


Core Concepts ...



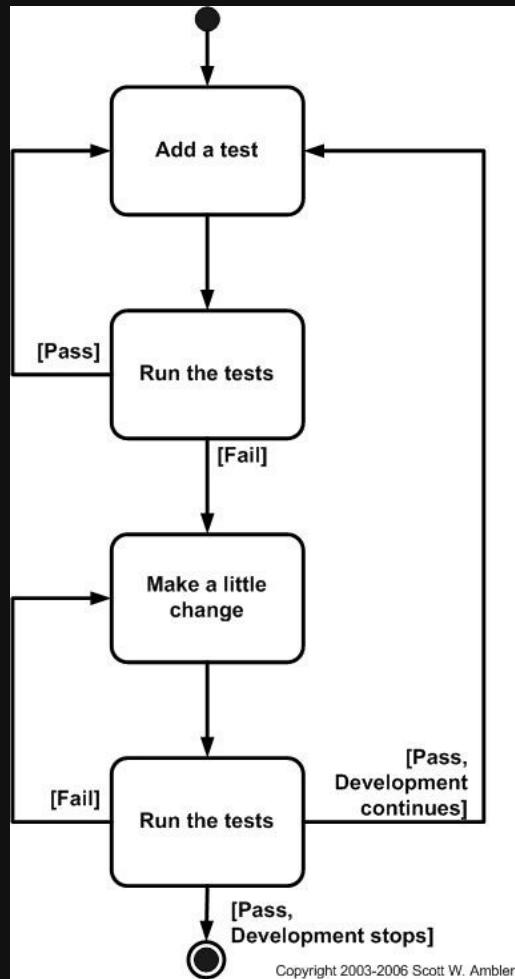
Benefits of BDD
BDD Frameworks
Book: BDD in Action
BDD - Introduction from Dan North

Core Concepts ...



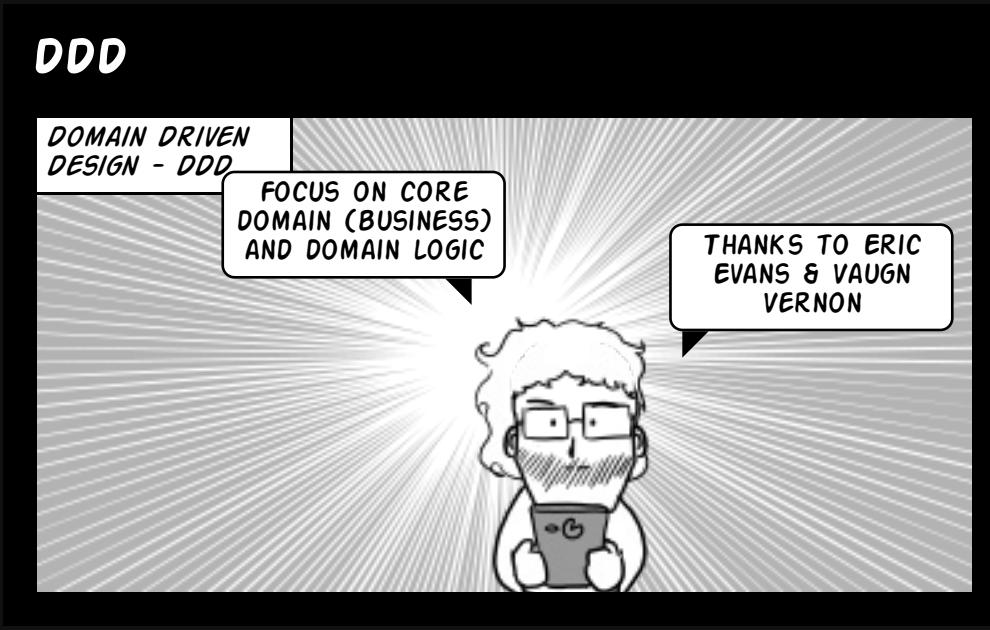
TDD Example
Introduction to TDD

Core Concepts ...



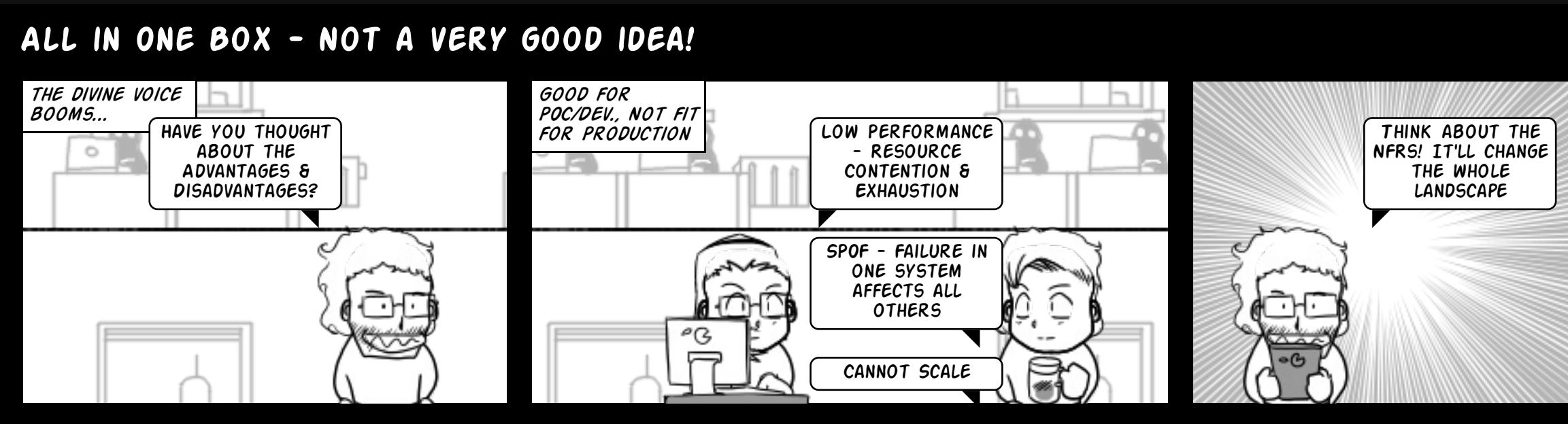
Copyright 2003-2006 Scott W. Ambler

Core Concepts ...



CQRS & Domain Driven Design
Awesome DDD - Nick Chamberlain
DDD Example
DDD Books
Domain Driven Design - Reference
The Ideal Domain-Driven Design Aggregate Store?
DDD & MicroServices

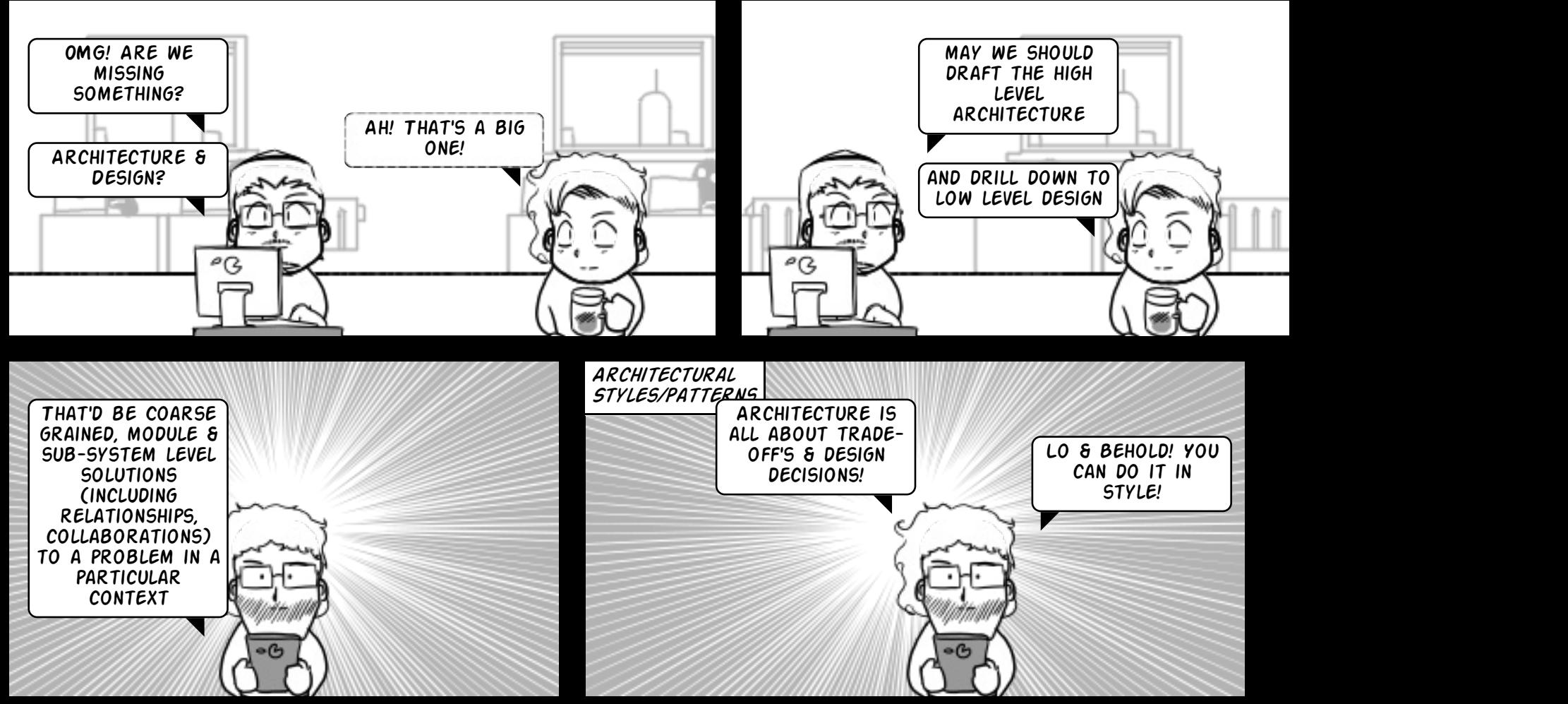
Architecture & Design ...



NFRs / QA (Quality Attributes)

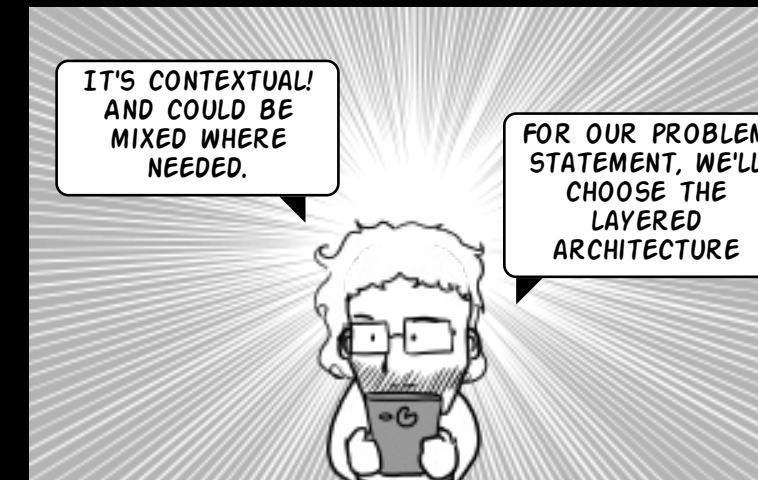
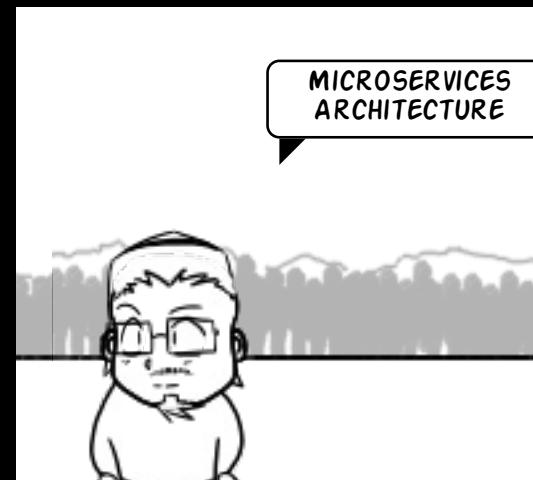
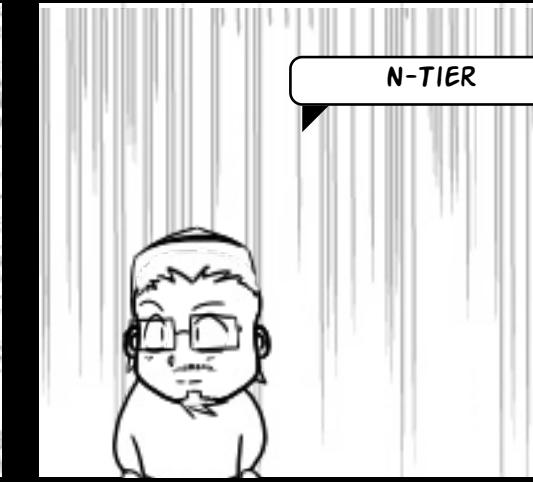
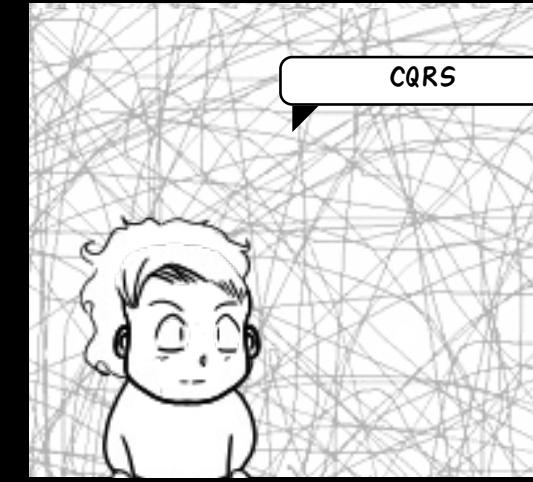
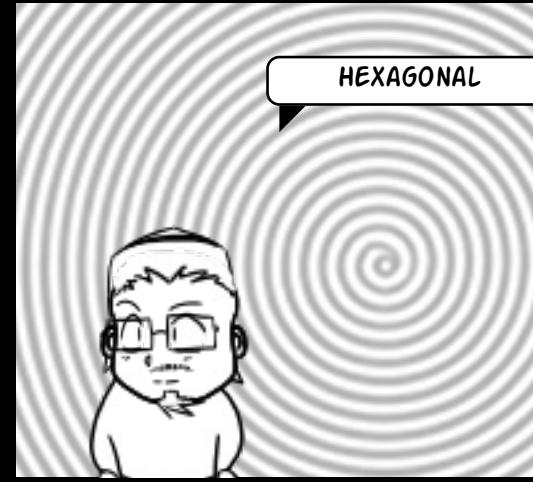
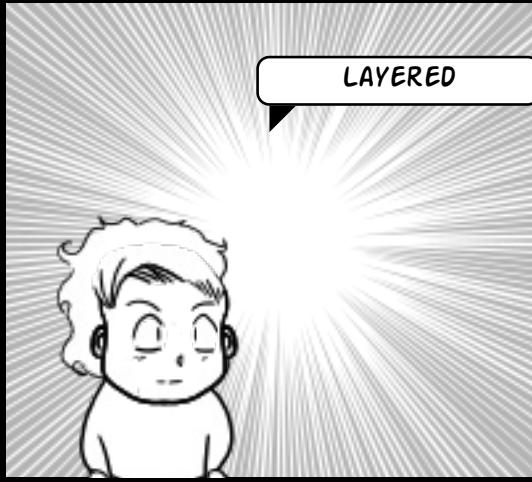
Architecture & Design ...

DEVS THINKING...



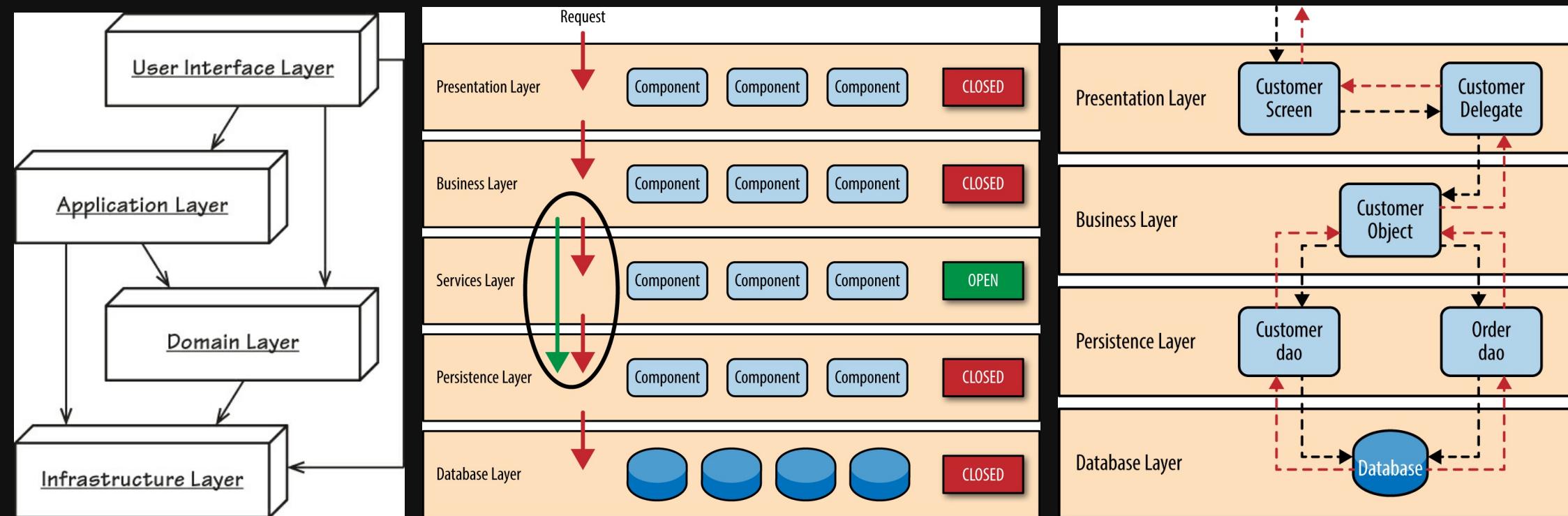
Architecture & Design ...

ARCHITECTURAL STYLES/PATTERNS & SELECTION



Architecture & Design ...

Layered Architecture

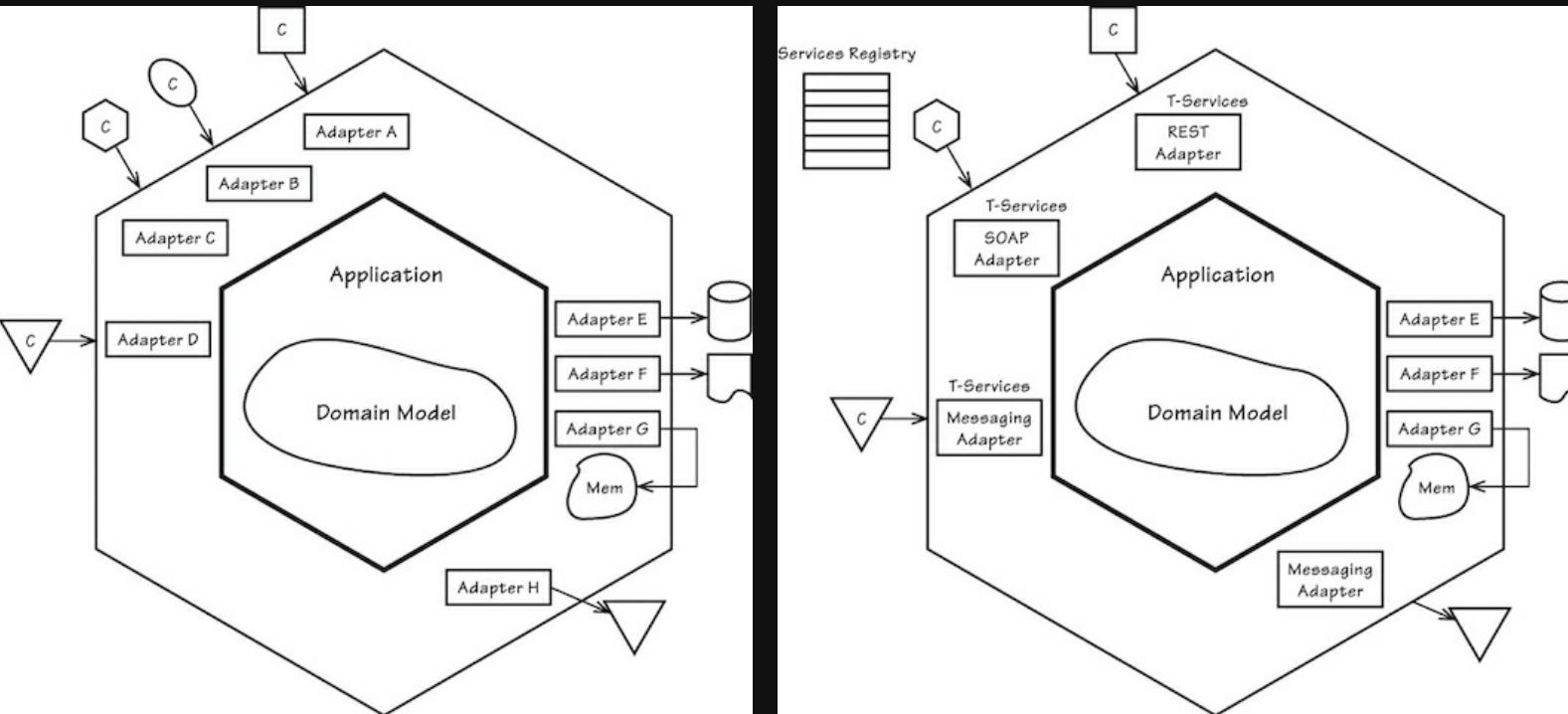


- Separation of Concerns - Simple Logical Decomposition, Top Down with Isolation -

Layered Architecture
N-Tier

Architecture & Design ...

Hexagonal Architecture



-Allow an application to equally be driven by users, programs, automated test or batch scripts, and to be developed and tested in isolation from its eventual run-time devices and databases-

[Hexagonal Architecture](#)

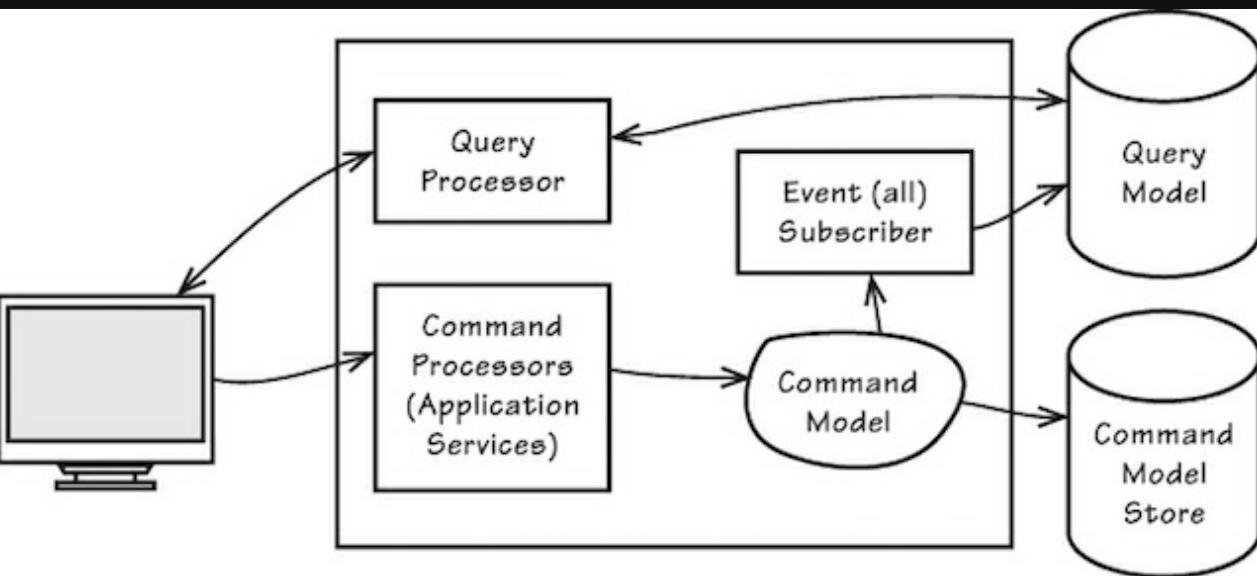
[Hexagonal Architecture FAQ](#)

[Hexagonal Architecture Example #1](#)

[Hexagonal Architecture Example #2](#)

Architecture & Design ...

CQRS

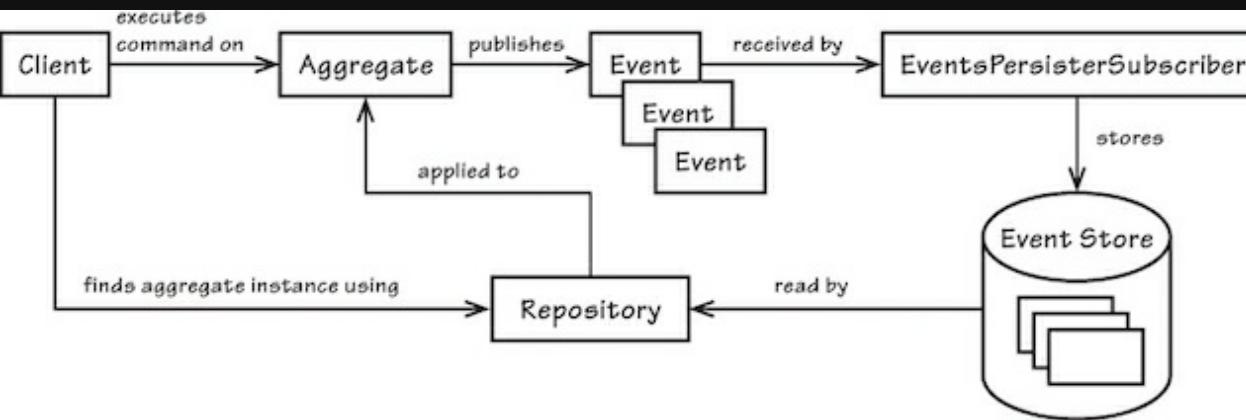


- Separate Flow for Queries (Reads) and Create (Writes) -

CQRS

Architecture & Design ...

Event Sourcing



- Events are immutable! There are no Updates & Deletes. Changes in state are stored as new events -

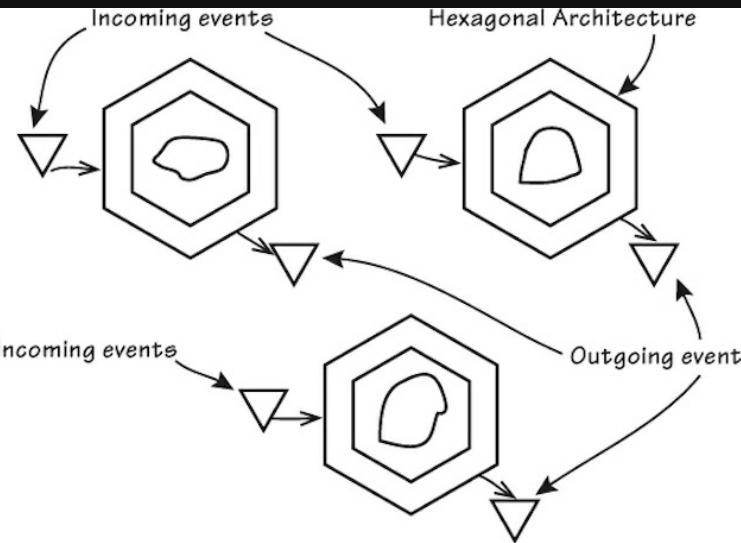
[Event Sourcing Concept - Martin Fowler](#)

[Event Sourcing Example #1](#)

[Event Sourcing Example #2](#)

Architecture & Design ...

Event Driven



- Different Systems integrate using Domain Events -

Event Driven Architecture

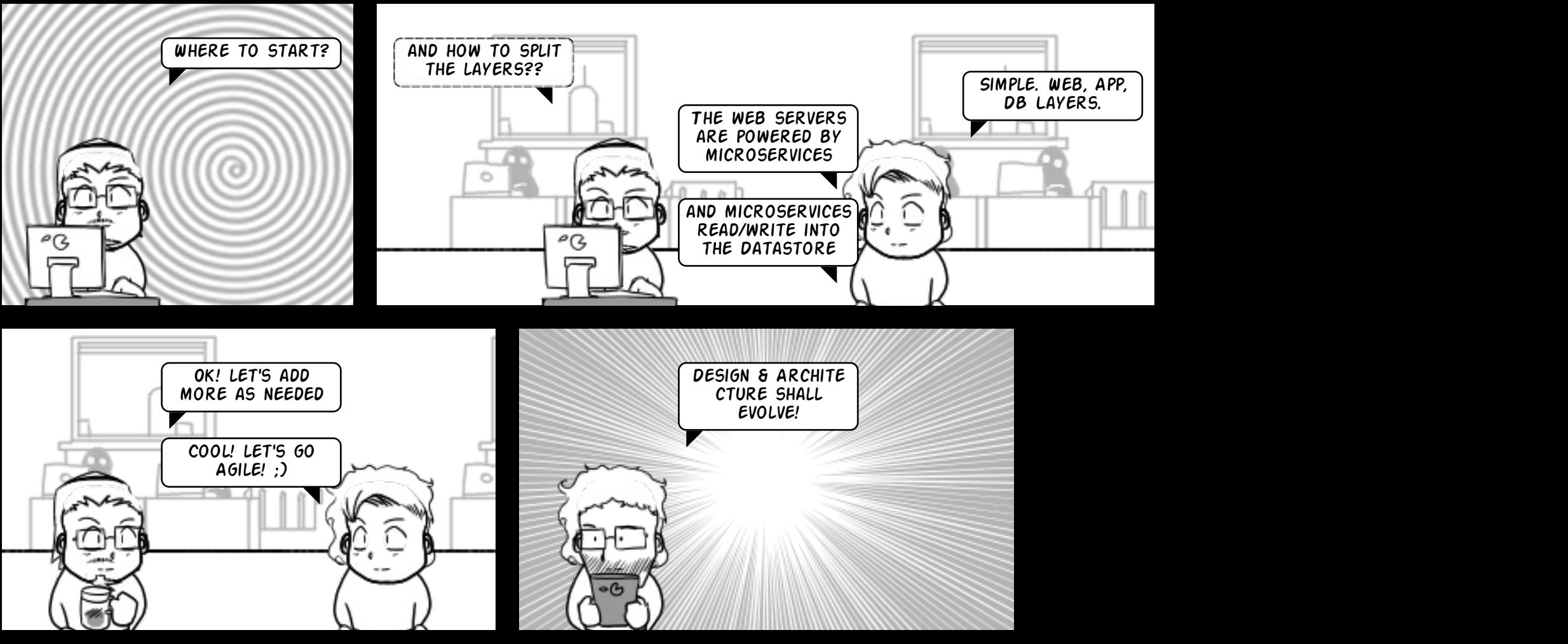
Event Driven Data Management for Microservices

Event Driven Architecture - Example #1

Event Driven Architecture - Example #2 - Saga Pattern

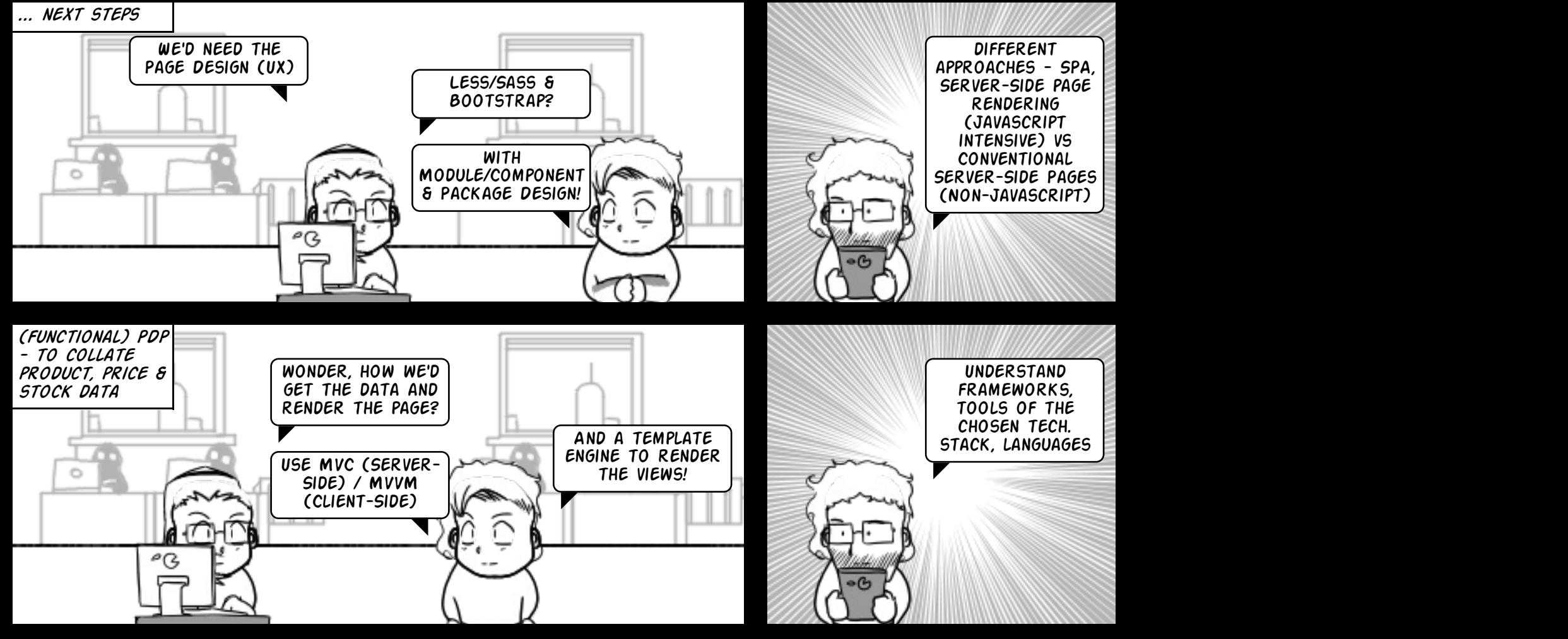
Architecture & Design ...

N-TIER ARCHITECTURE & LAYERS

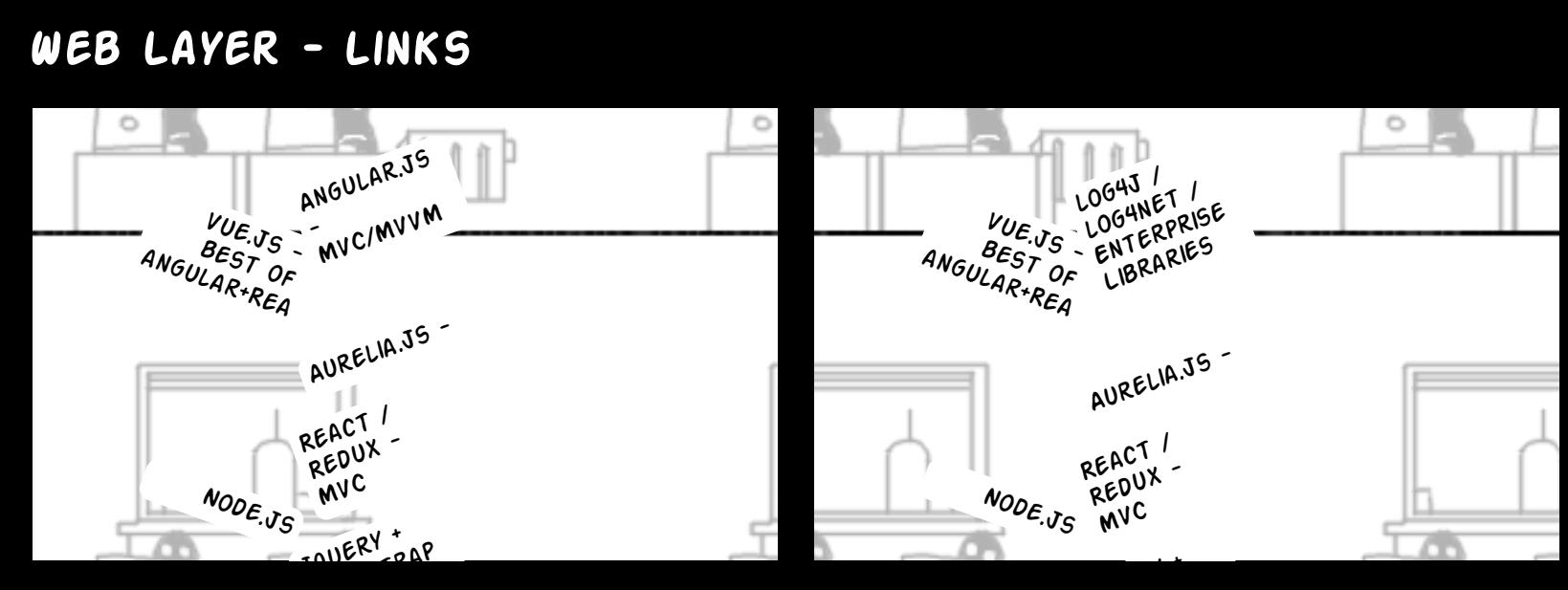


Architecture & Design ...

BREAKING DOWN INTO MANAGEABLE PIECES - FUNCTIONAL COMPONENTS

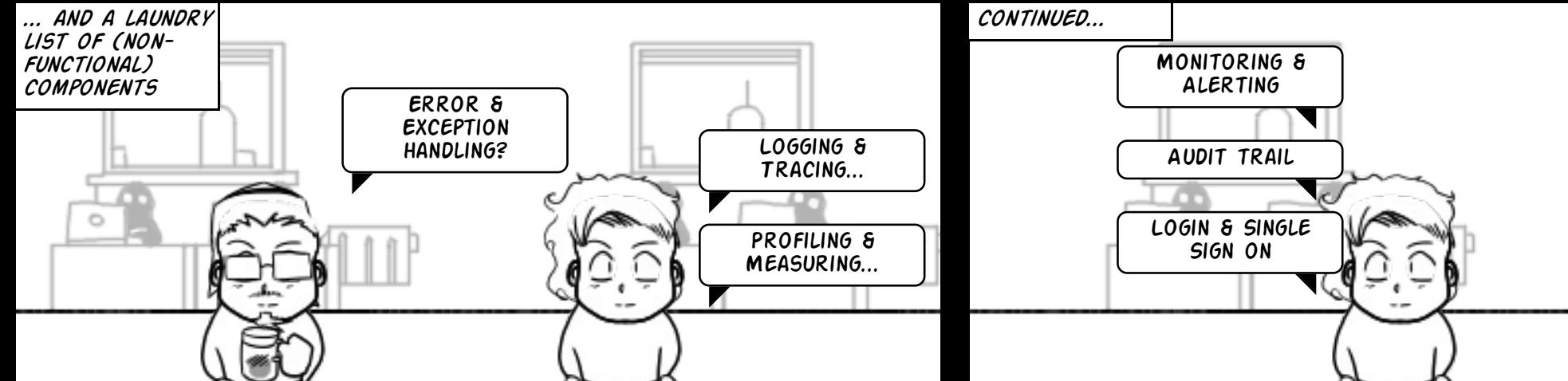


Architecture & Design ...



Architecture & Design ...

BREAKING DOWN INTO MANAGEABLE PIECES - NON-FUNCTIONAL COMPONENTS



Architecture & Design ...

Microsoft Patterns & Practices

Microsoft Application & Architecture Guide

Shaping Software - Architecture

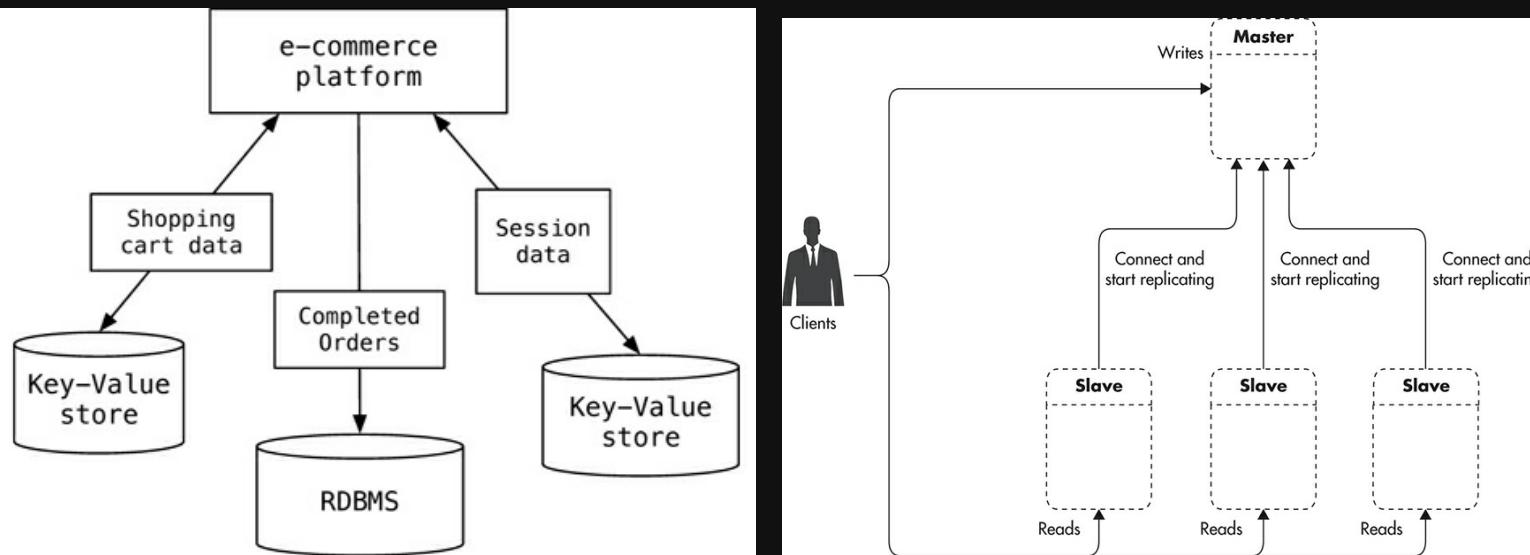
Clean MicroService Architecture

Coding the Architecture

Software & Systems Architecture

Architecture & Design ...

Choose Polyglot DataStores for different data needs

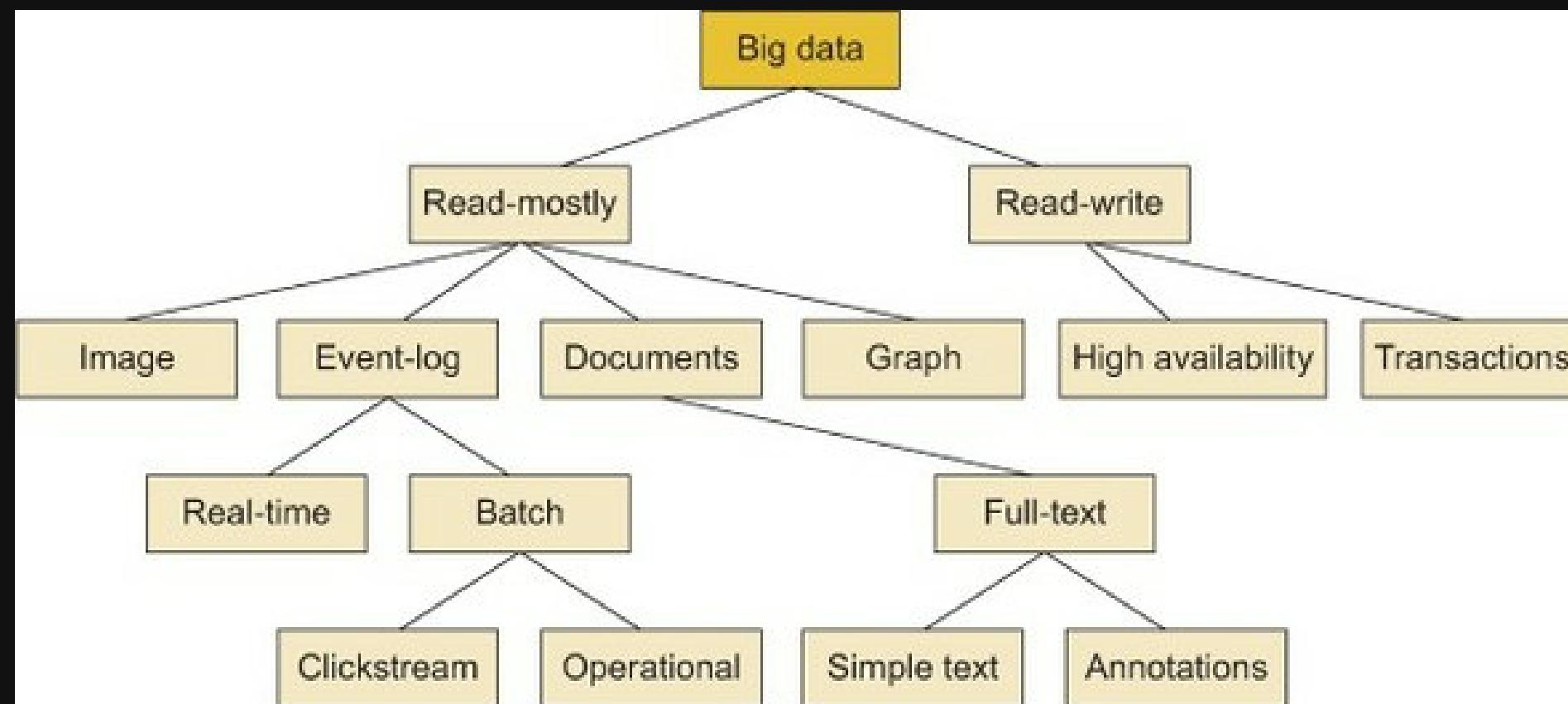


- RDBMS's for transactional needs (w replication), NoSQL for Scalability -

Microsoft Patterns & Practices - Data Guidance

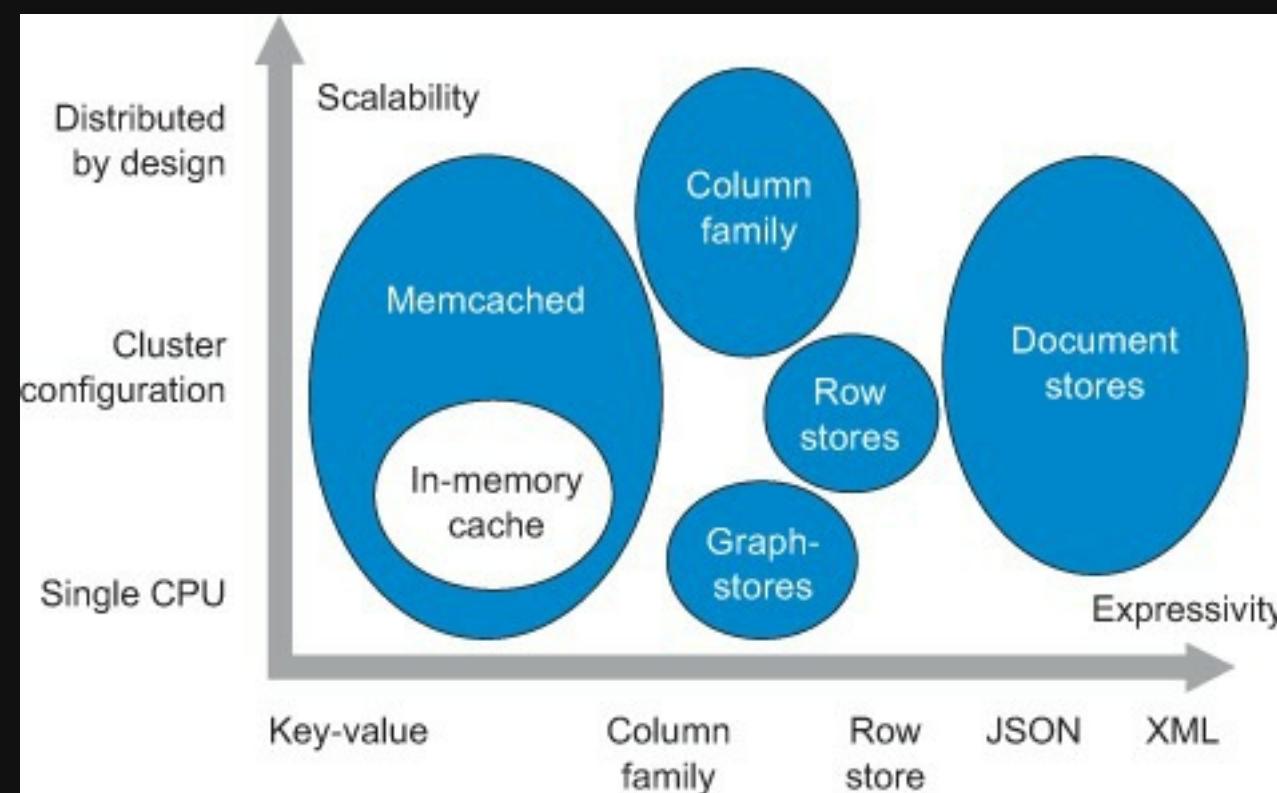
Architecture & Design ...

Tips to choose the right data store based on the use case



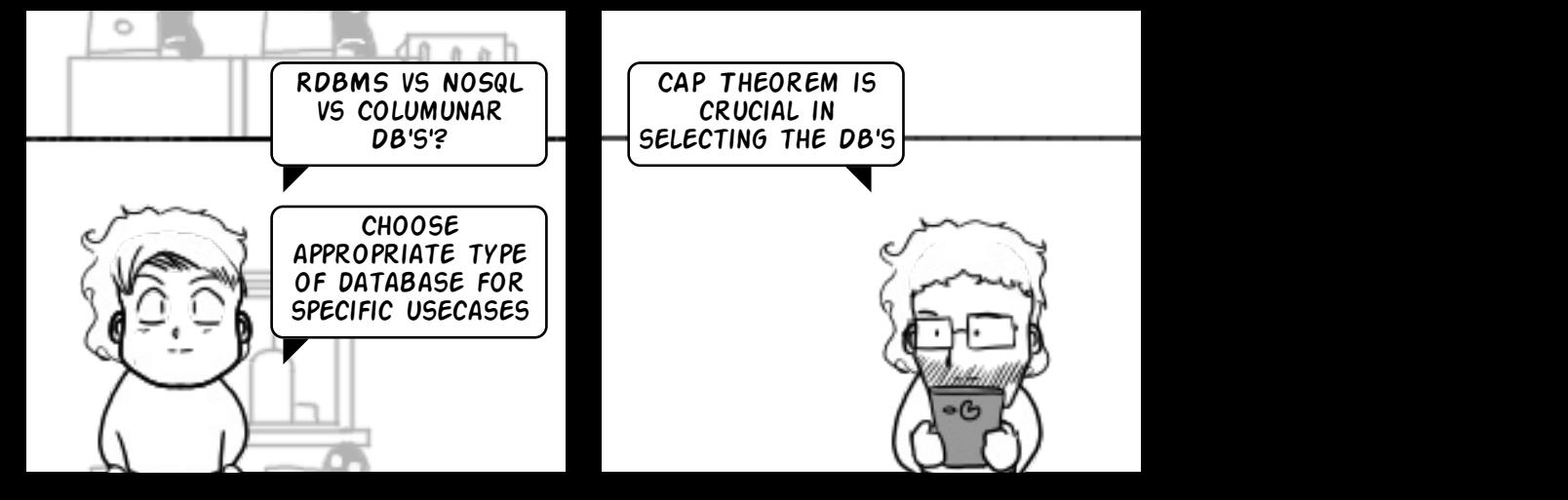
Architecture & Design ...

Tips to choose the right data store based on the type of data stored



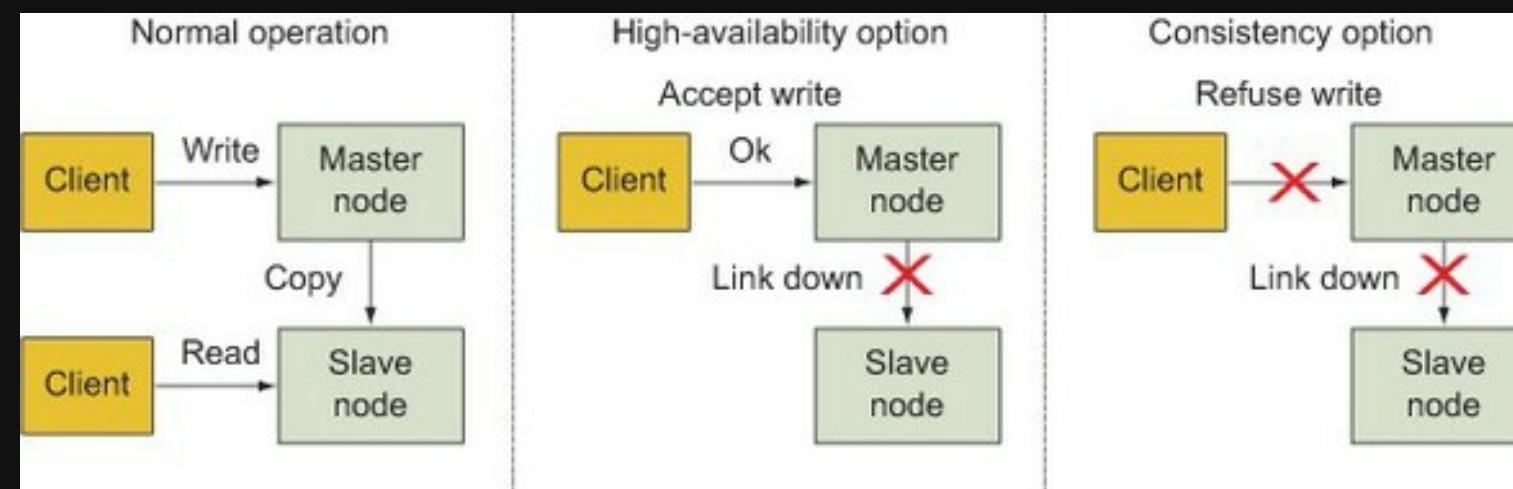
Architecture & Design ...

CHOOSE THE TECHNOLOGIES, FRAMEWORKS & TOOLS FOR DB LAYER



Architecture & Design ...

CAP Theorem



- Choosing between Availability, Consistency + Partition Tolerance -

Engineering ...

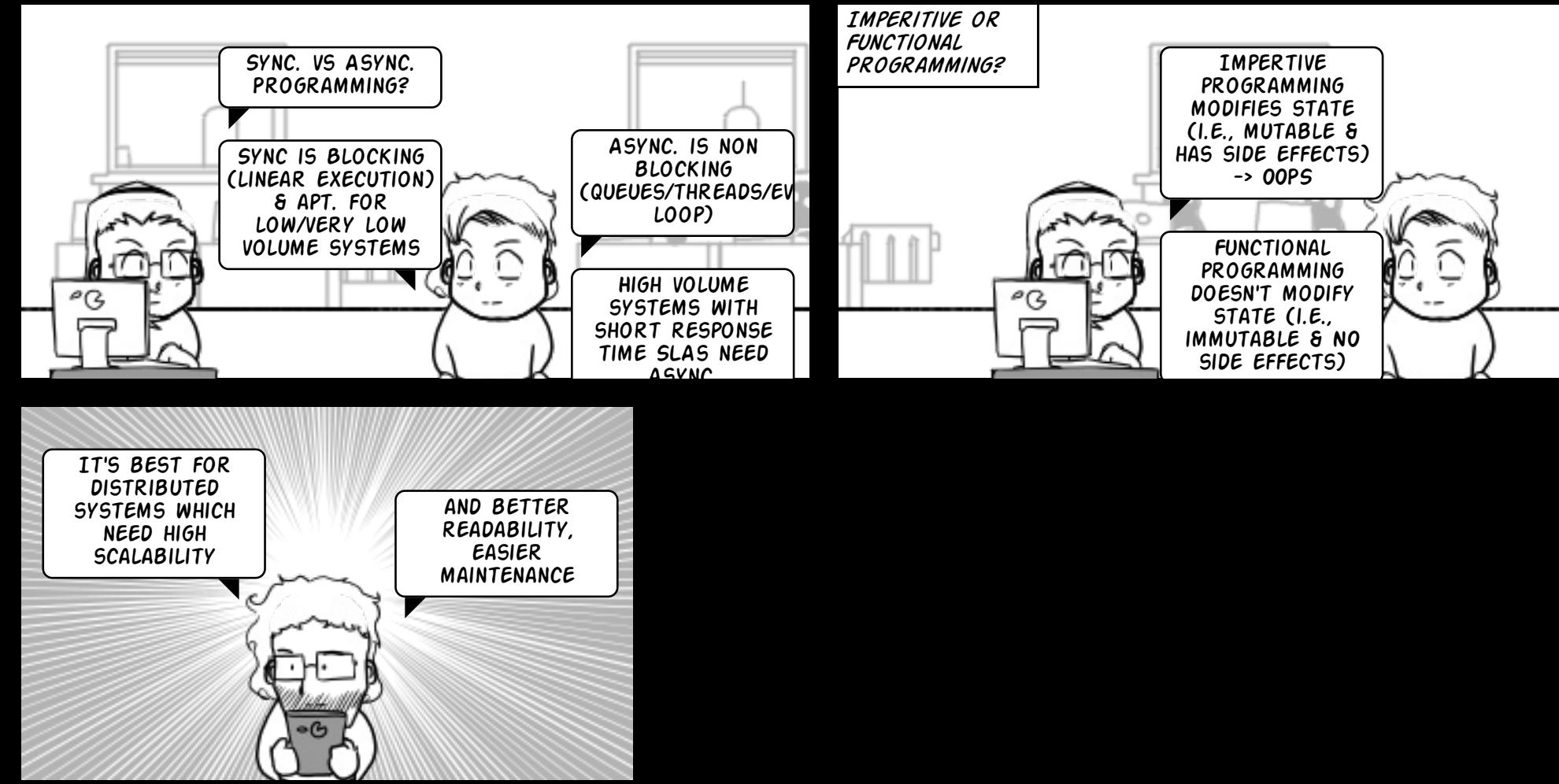
APPLY DESIGN PRINCIPLES & BEST PRACTICES



Follow Alistair Cockburn - Agile & Engineering Guru!
Microsoft Engineering Best Practices & Guidance

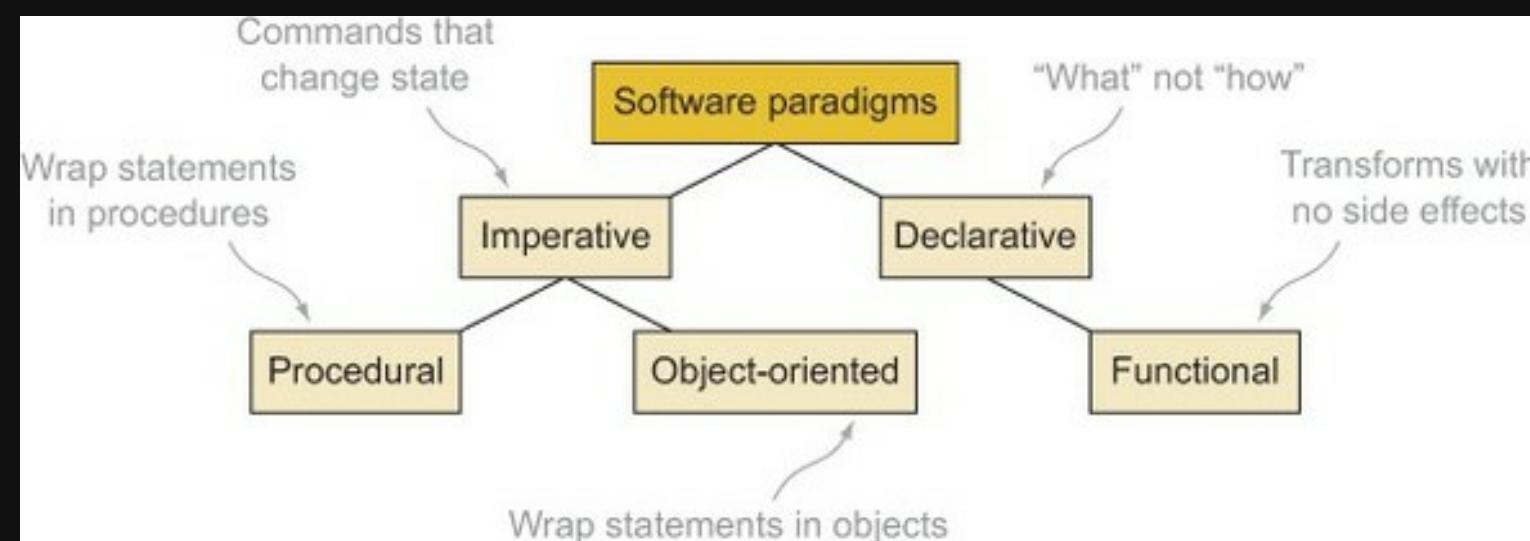
Engineering ...

BUILDING THE APP - CHOICE OF THE PROGRAMMING STYLE



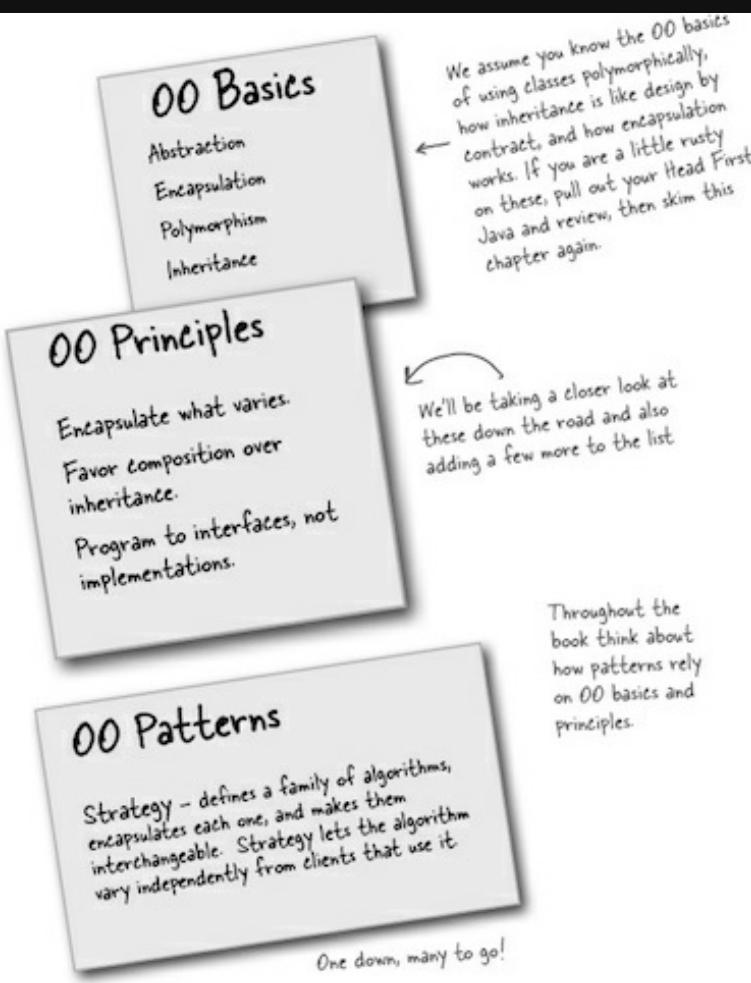
Engineering ...

Software Paradigms



Engineering ...

OOPS!

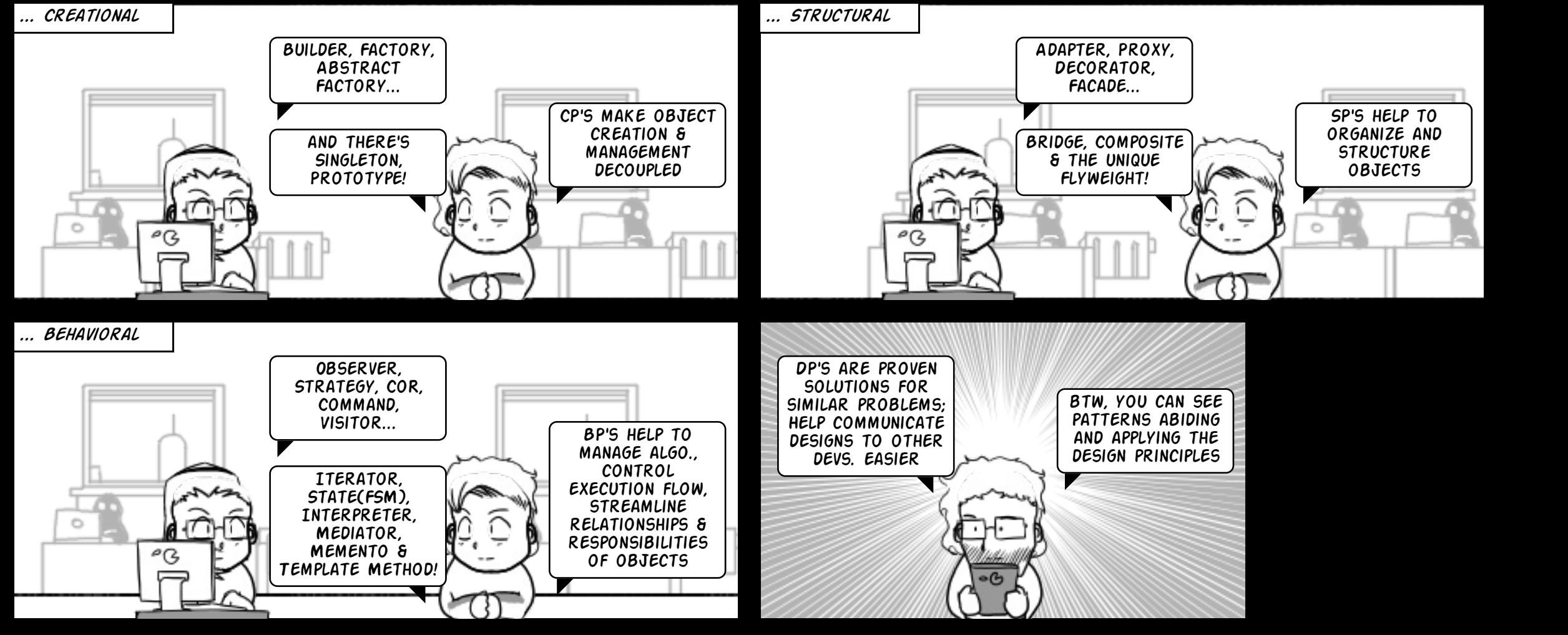


Engineering ...

A Practical Intro to Functional Programming

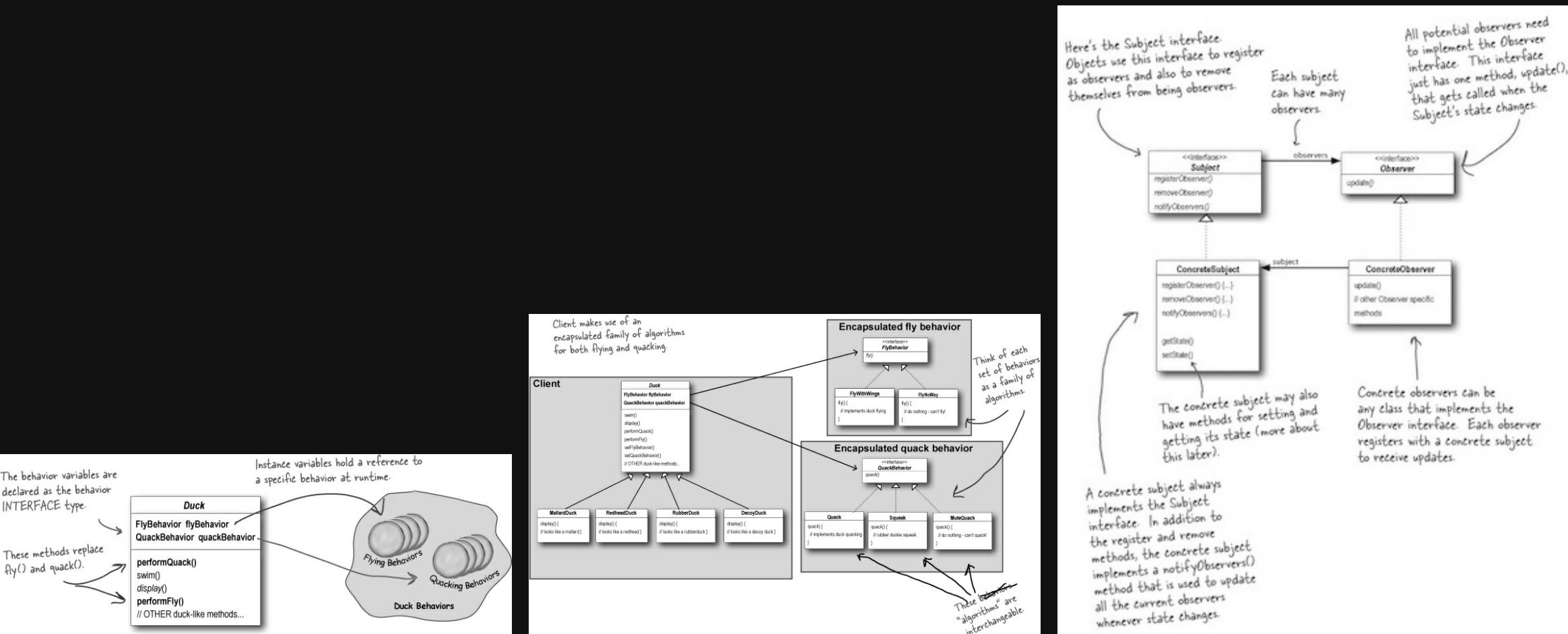
Engineering ...

DESIGN PATTERNS - FINE GRAINED, CLASS LEVEL SOLUTION TO A PROBLEM IN A PARTICULAR CONTEXT



Design Patterns

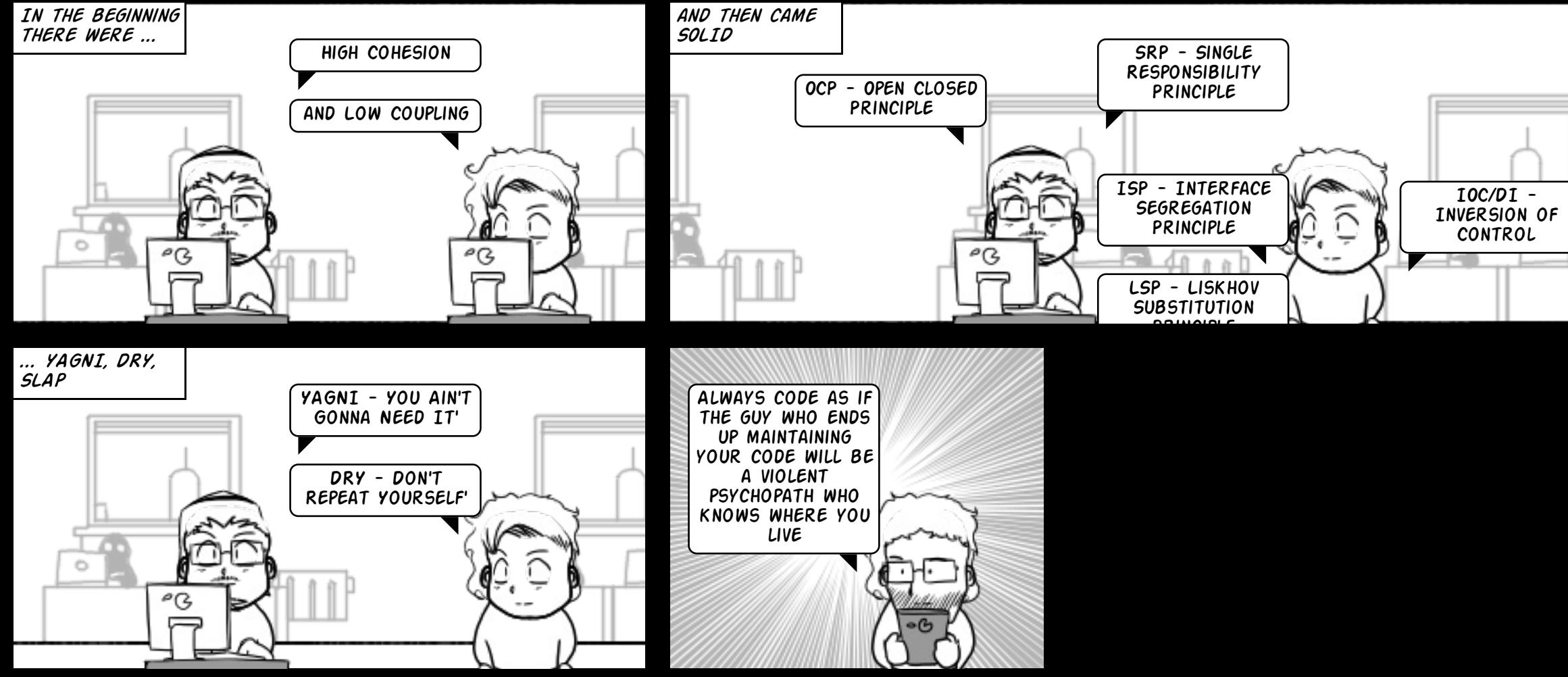
Engineering ...



Design Patterns Catalog Patterns & Principles w Examples - Exhaustive List

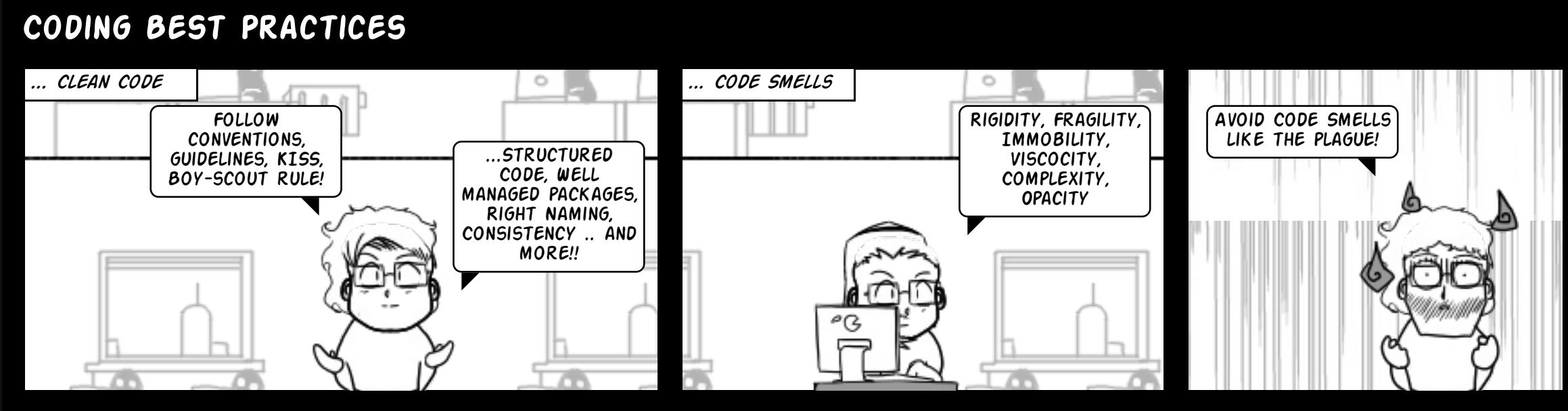
Engineering ...

DESIGN PRINCIPLES



Principles Wiki

NFR's : Quality....

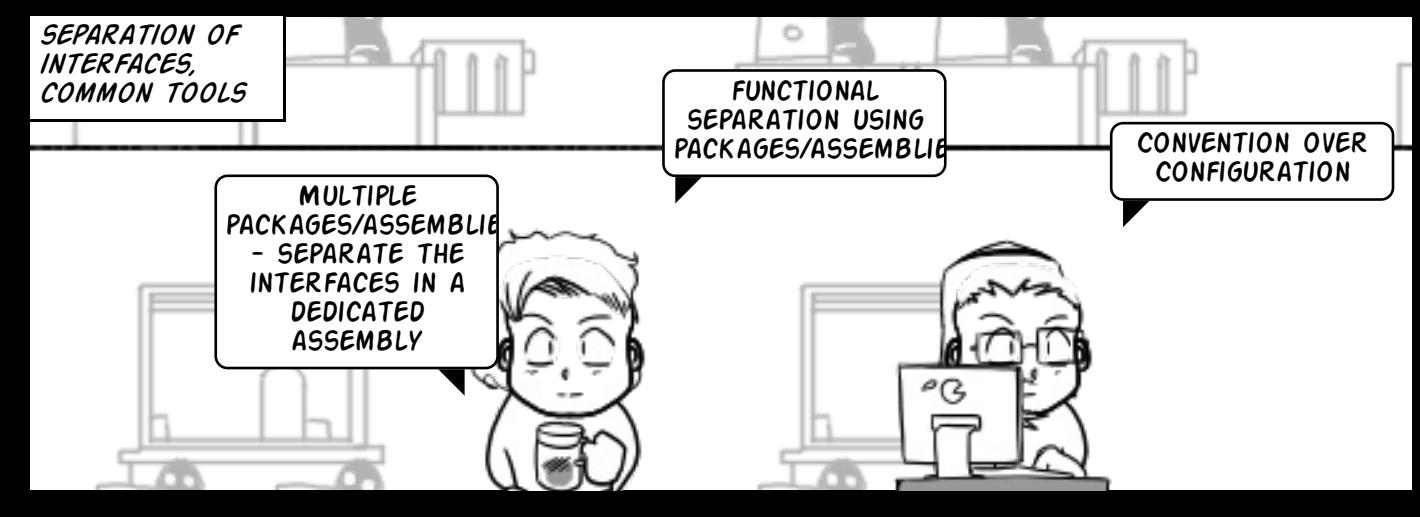


NFR's : Quality....

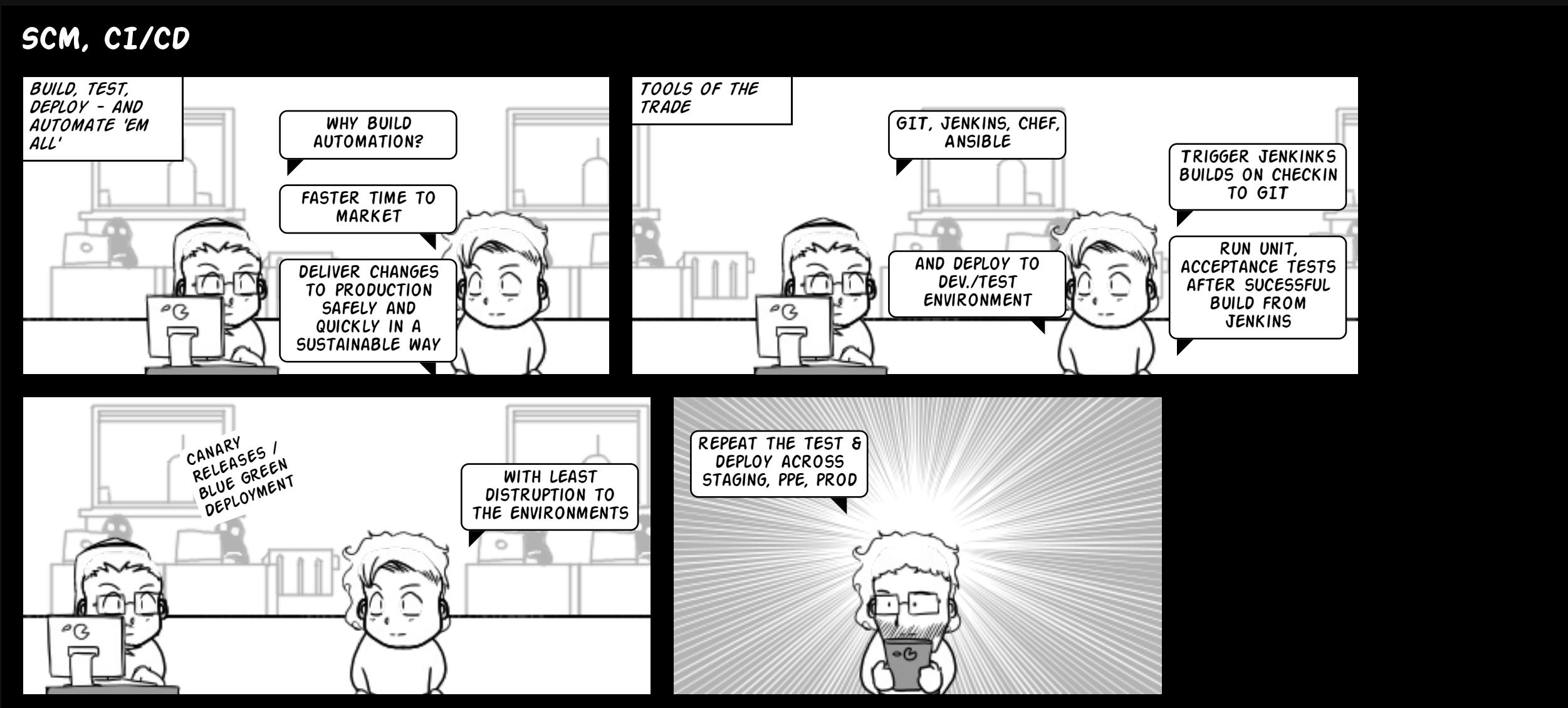
Clean Code Cheatsheet
Software Craftsmanship
Essential Programming Books - Bibles
Influential CS Books
Code Smells

NFR's : Quality....

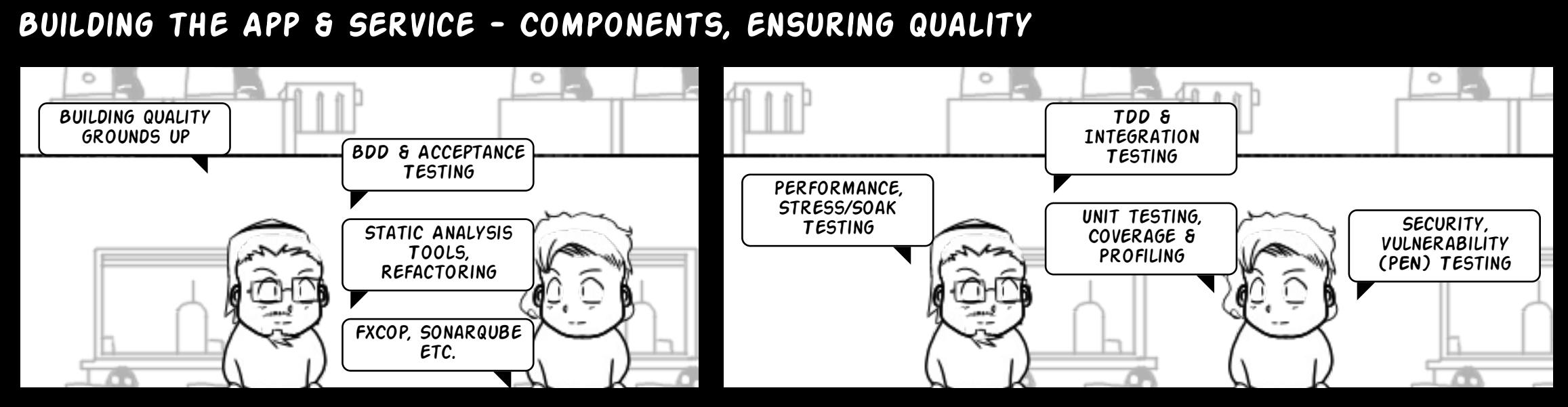
CONTD.. ENSURING QUALITY - PACKAGING FOR DELIVERY



NFR's : Quality....

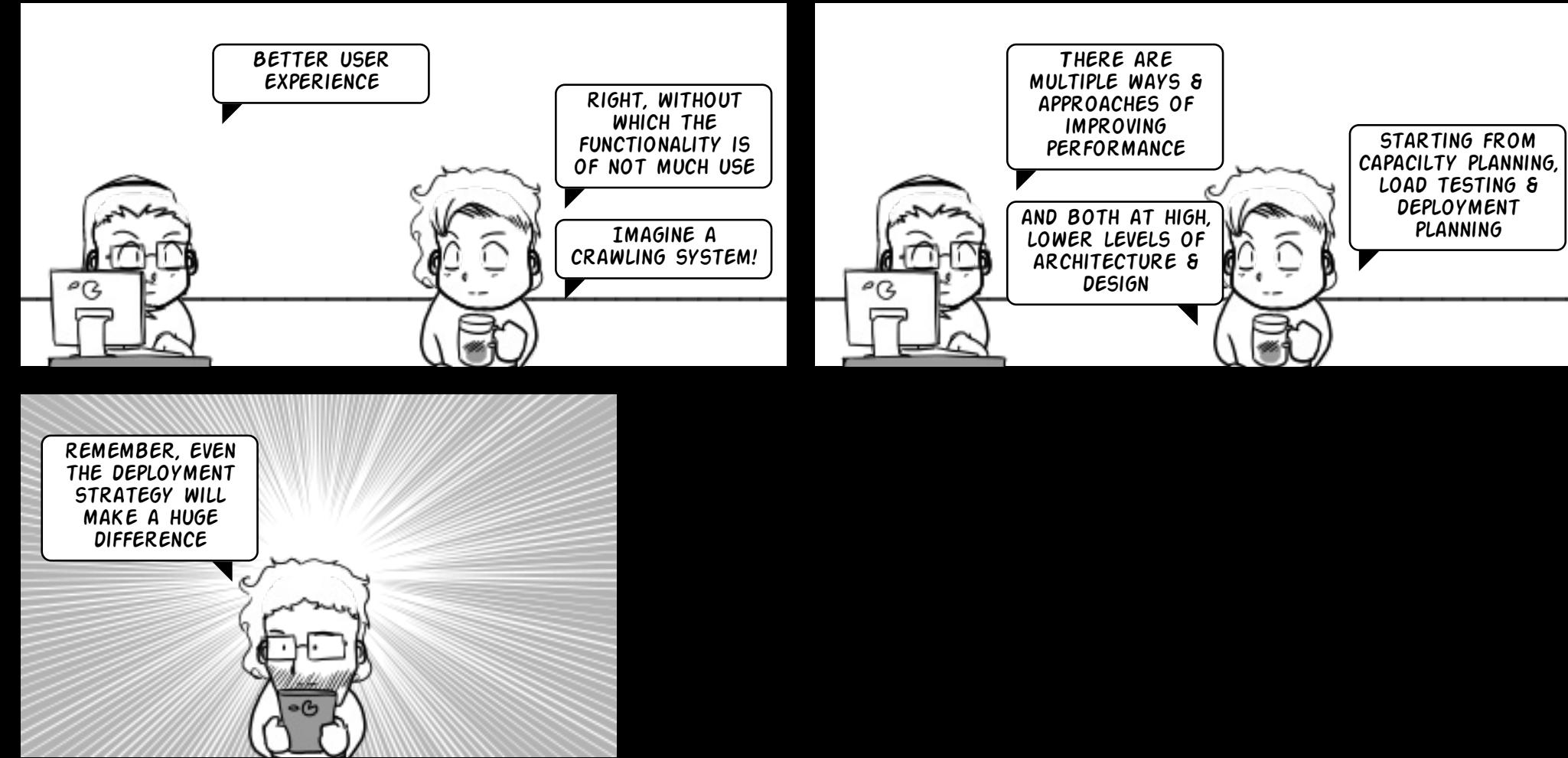


NFR's : Quality....



NFR's: Performance

WHY IMPROVE PERFORMANCE?

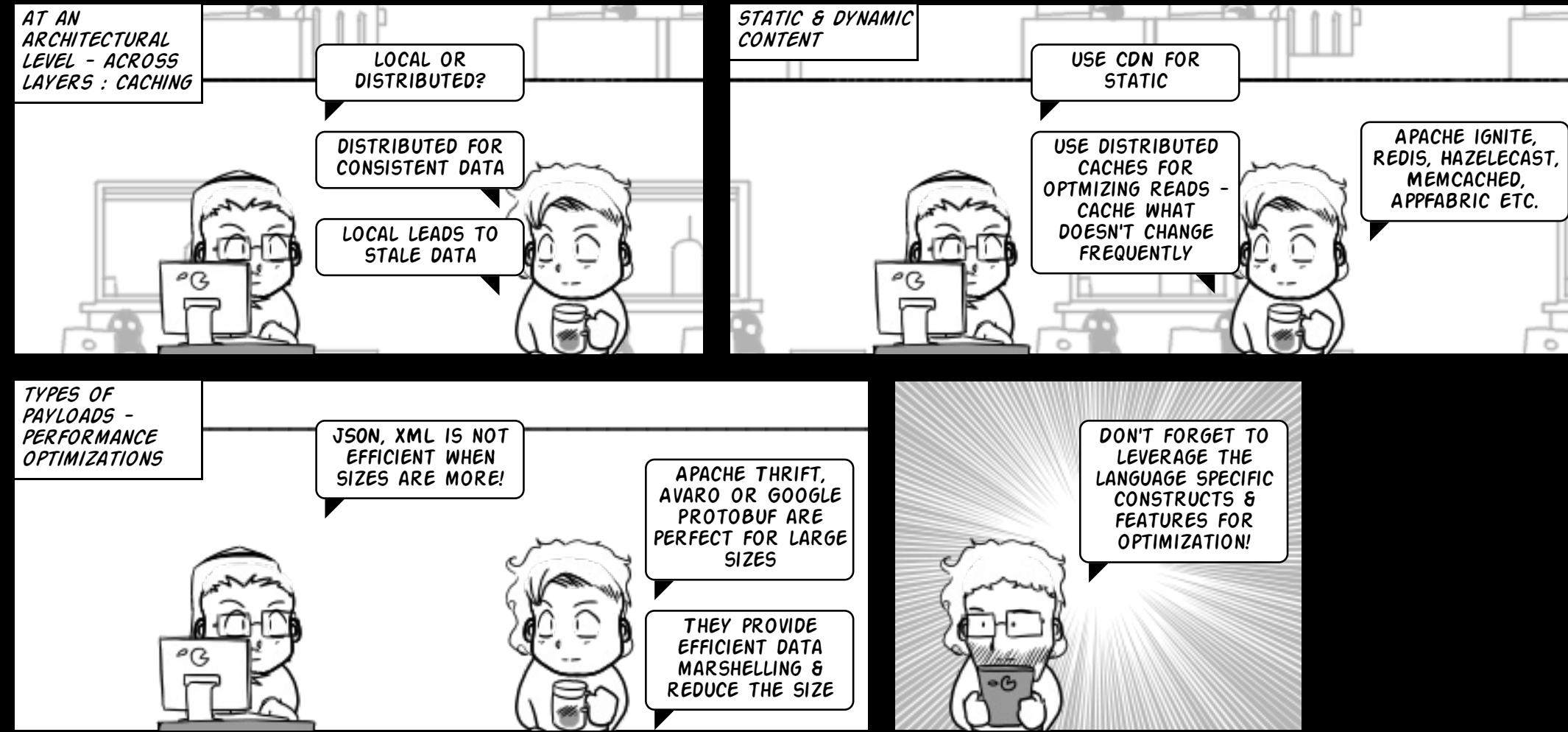


NFR's : Performance

Performance Testing
Awesome Web Performance Optimization
Path To Performance - Podcast

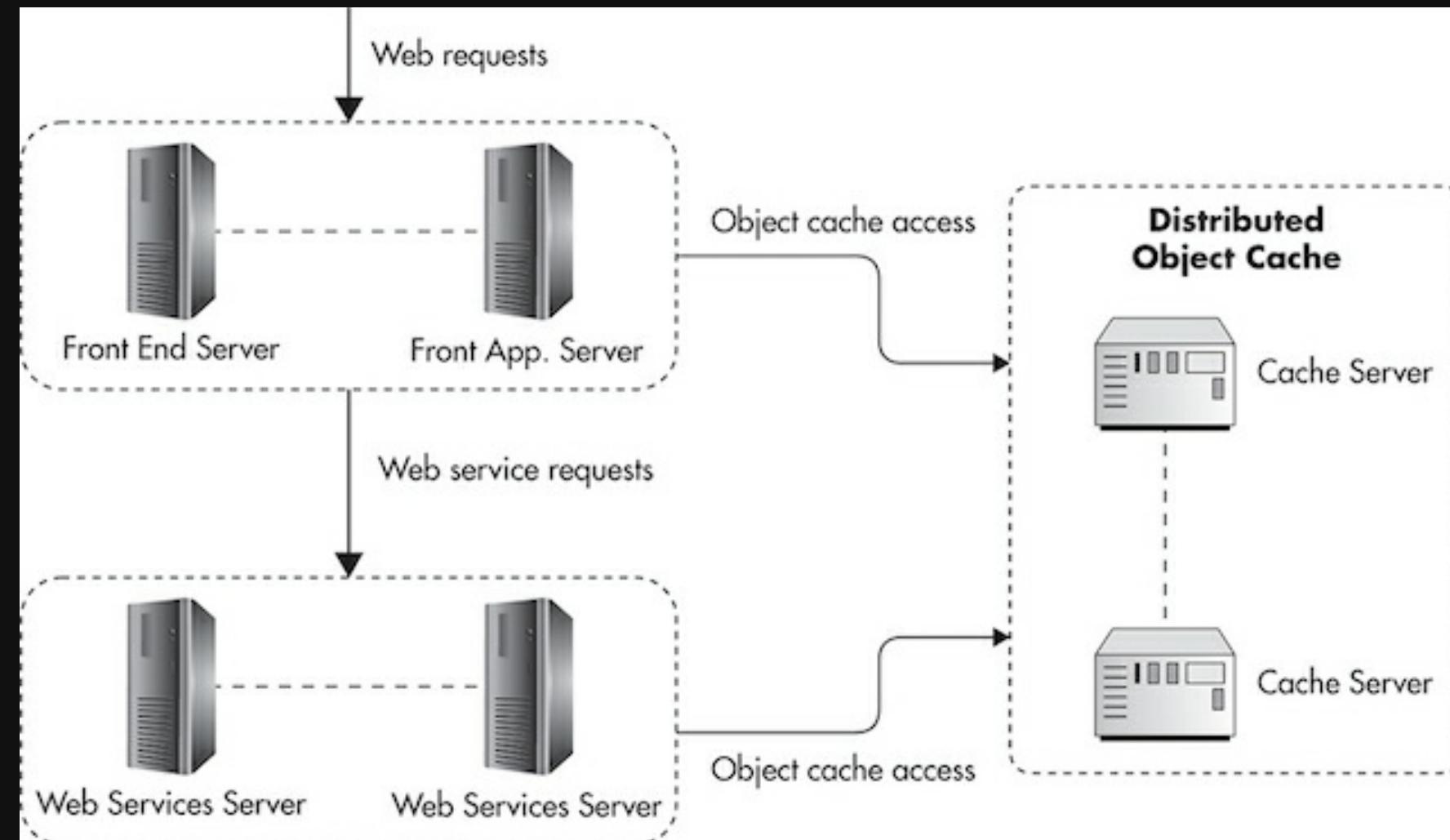
NFR's: Performance

HOW TO IMPROVE PERFORMANCE



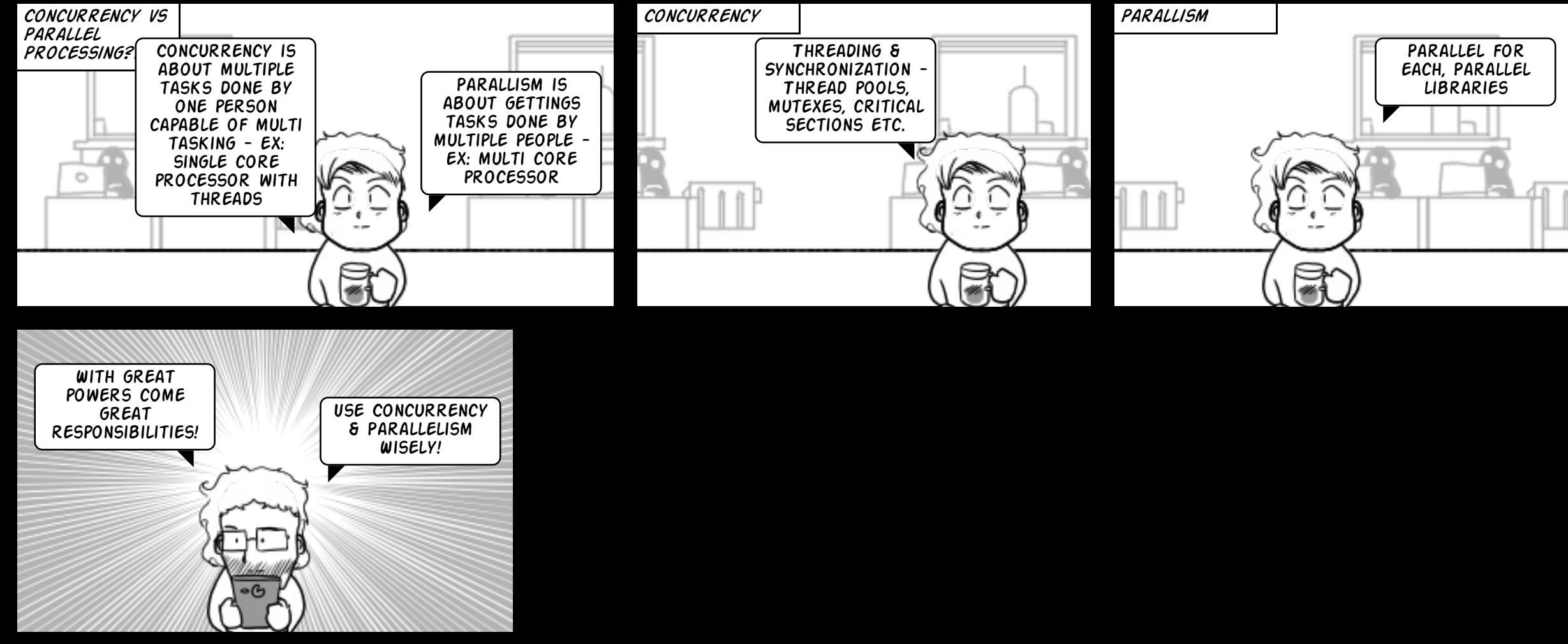
C# Internals
C# async/await
CLR via C#
Java Performance Tuning

NFR's: Performance

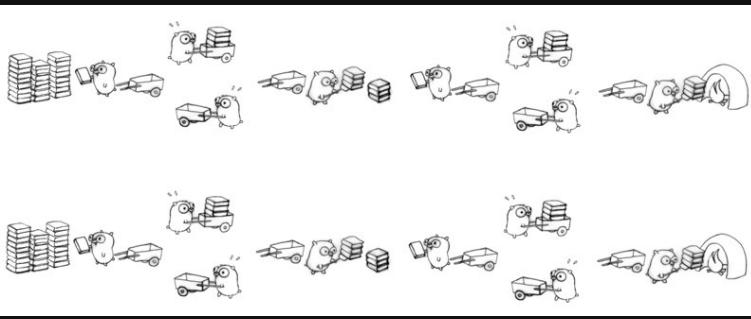
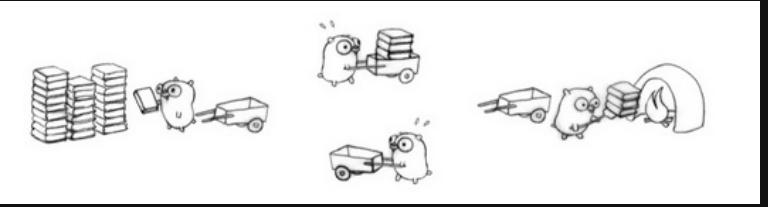
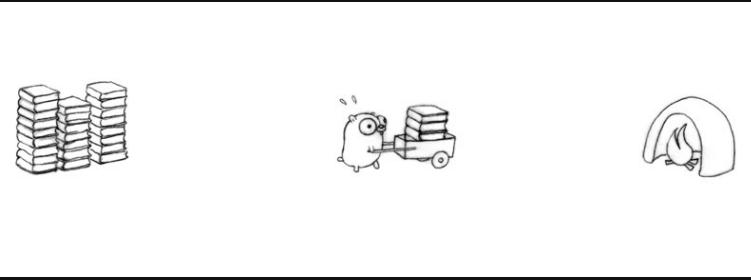


NFR's: Performance

CONCURRENCY & PARALLELISM

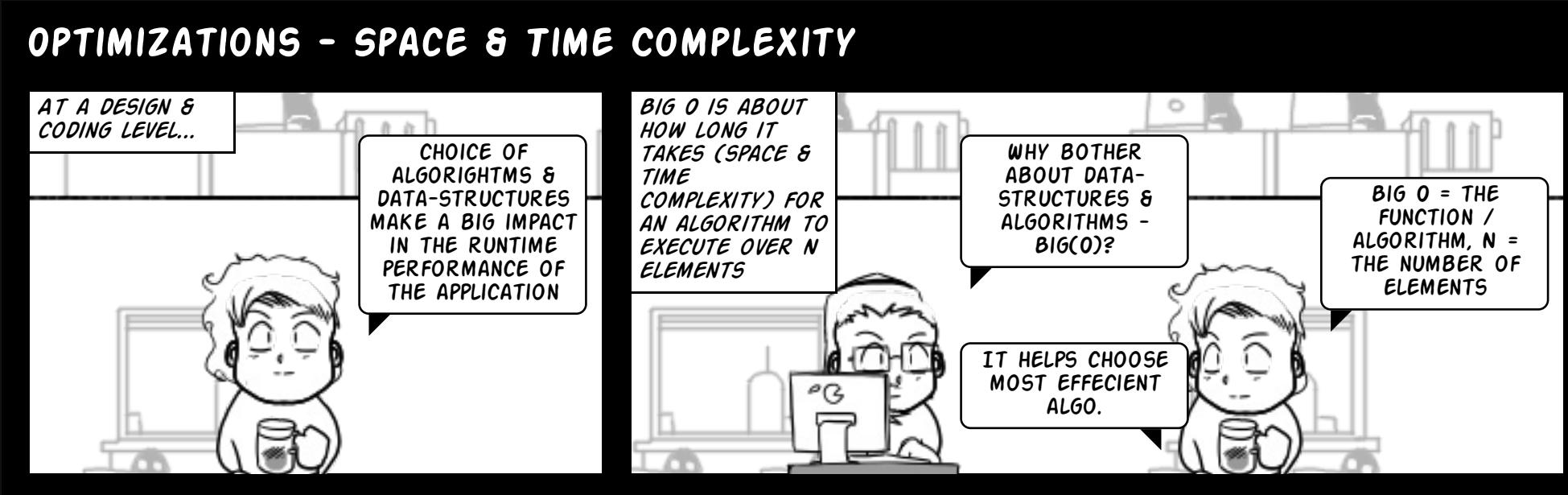


NFR's: Performance



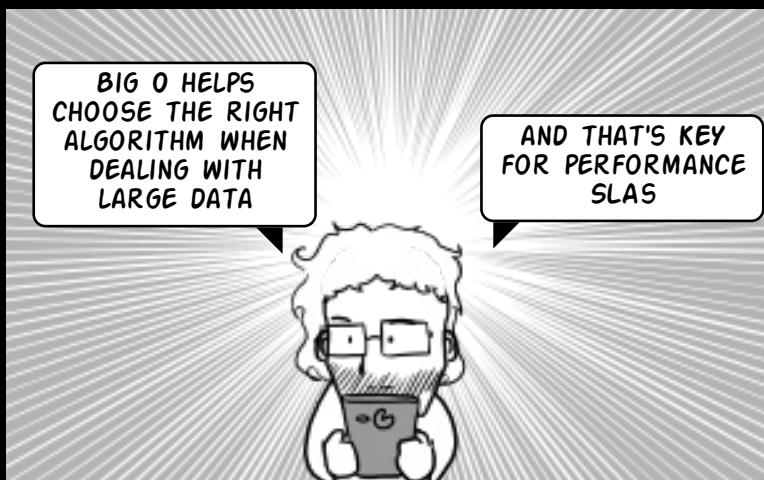
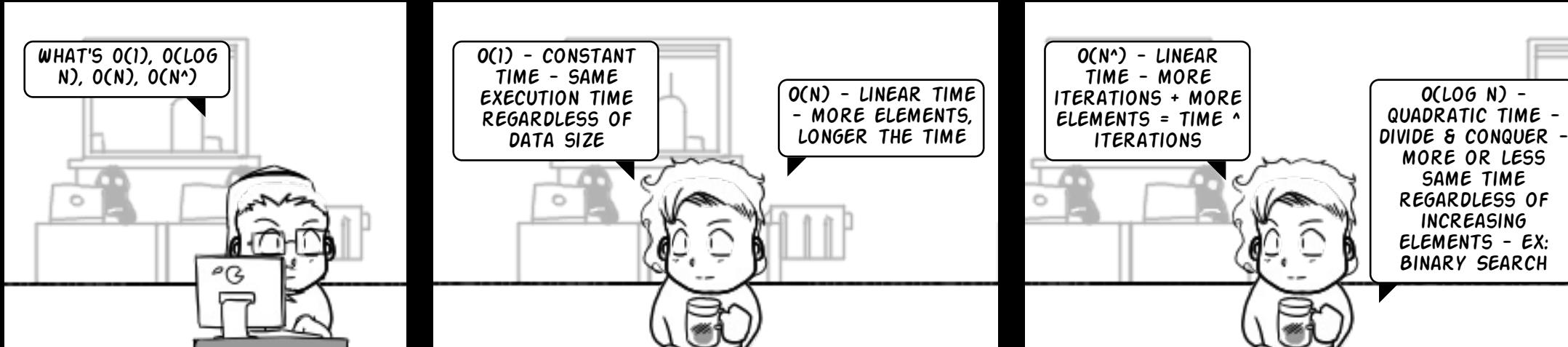
Concurrency vs Parallelism
Threading in C#

NFR's: Performance



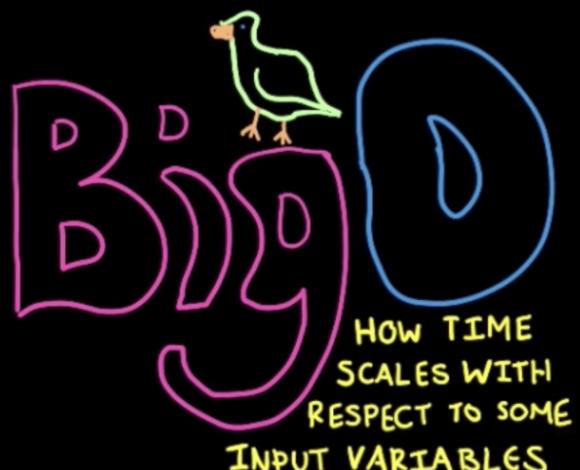
NFR's: Performance

...CONTD. - OPTIMIZATIONS - SPACE & TIME COMPLEXITY



NFR's: Performance

④ Drop non-dominant terms



```
function whyWouldIDoThis(array){  
    max = NULL  
    O(n) {  
        foreach a in array {  
            max = MAX(a, max)  
            print max  
        }  
    }  
    O(n^2) {  
        foreach a in array {  
            foreach b in array {  
                print a,b  
            }  
        }  
    }  
}  
  
 $O(n^2) \leq O(n+n^2) \leq O(n^2+n^2)$   
*if LEFT and RIGHT are equivalent  
(see RULE 2), then CENTER is too*  
 $O(n+n^2) \Rightarrow O(n^2)$ 
```

Grokking Algorithms!

NFR's : Performance

A Beginners guide to big-o-notation

Big-O notation in 5 minutes — The basics

Big O Notation explained

Big O Notation: A Few Examples

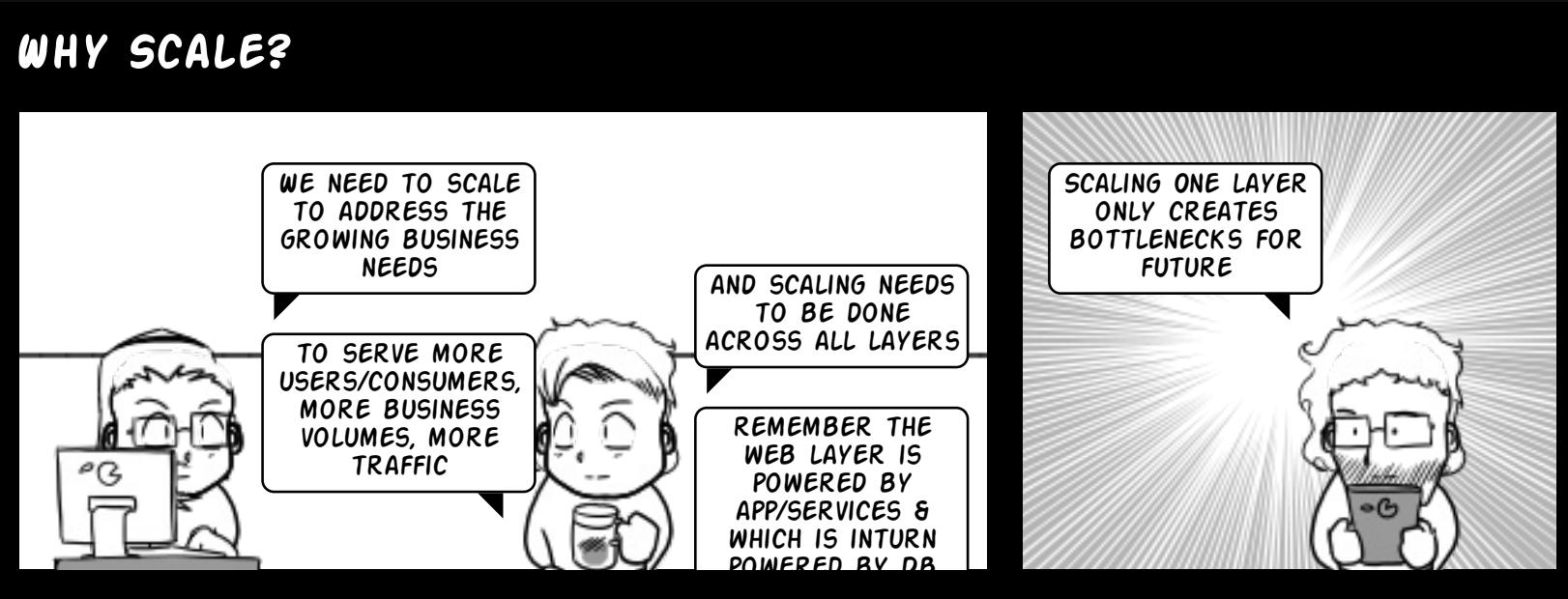
Big O CheatSheet

Algorithms for dummies

Algorithmic Complexity

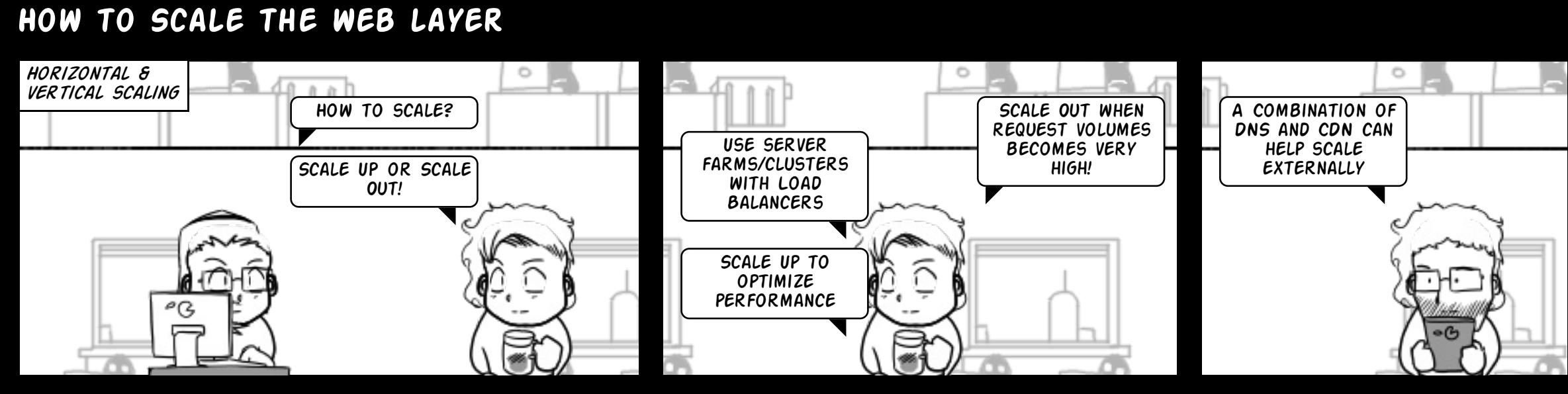
Understanding Big O

NFR's : Scalability

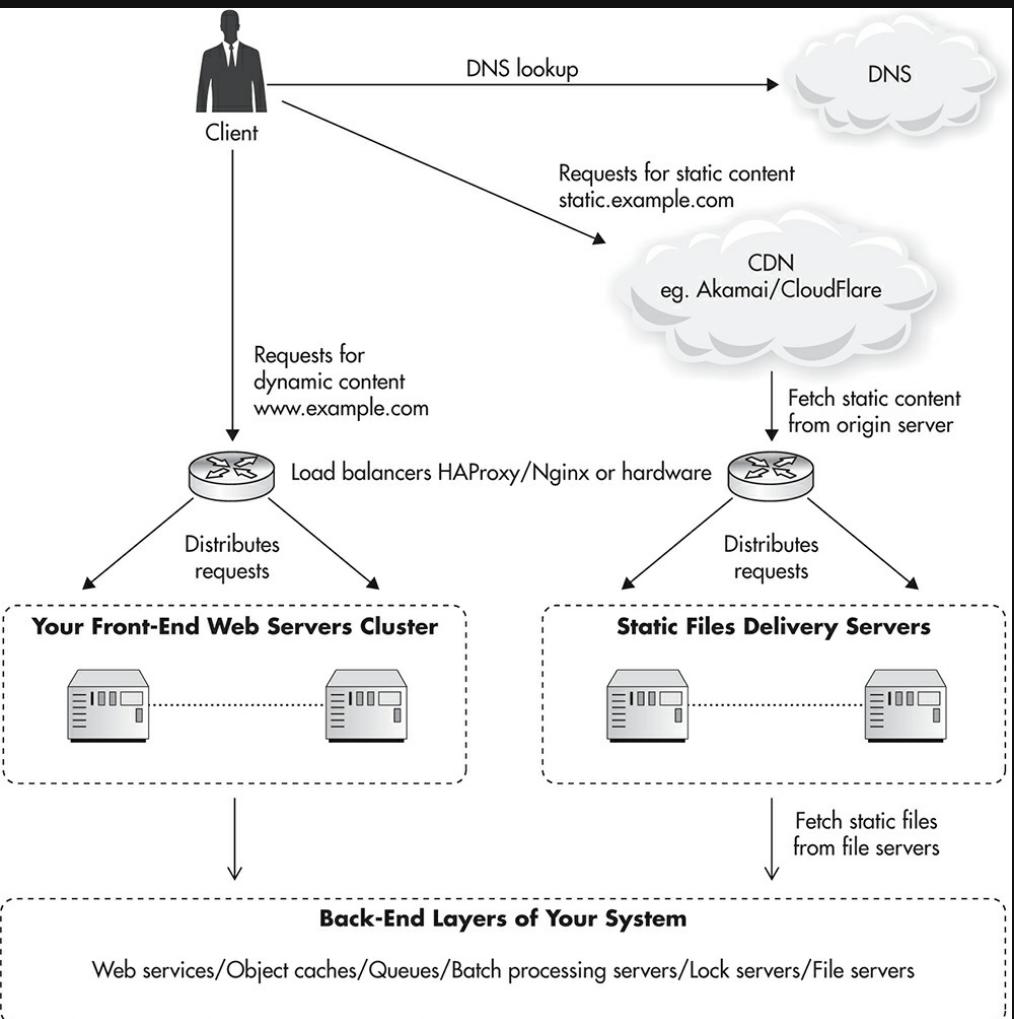


Awesome Scalability
Microsoft Improving Performance & Scalability

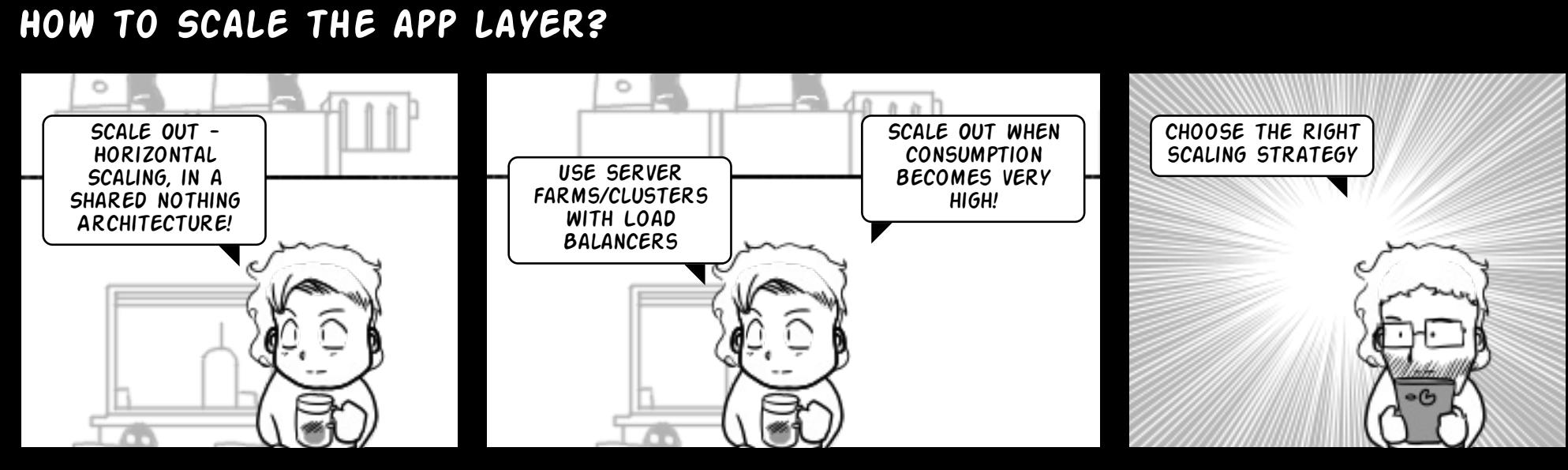
NFR's : Scalability



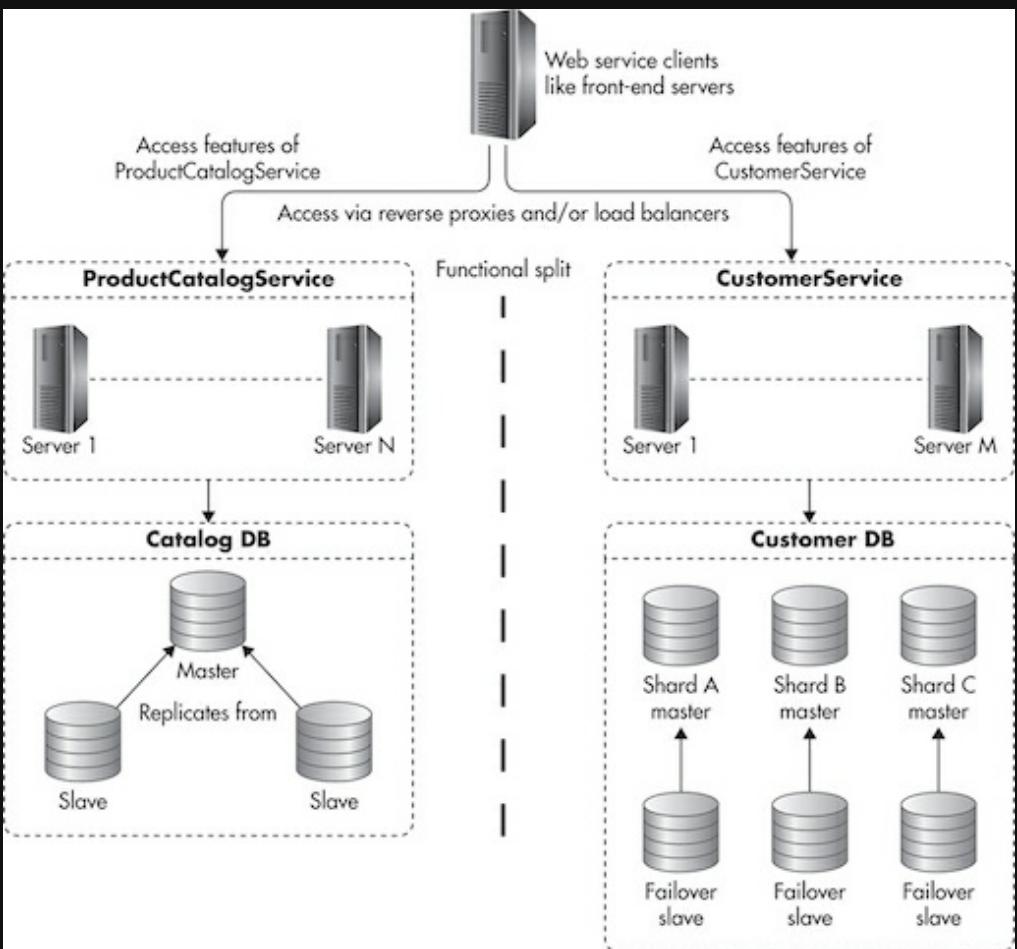
NFR's : Scalability



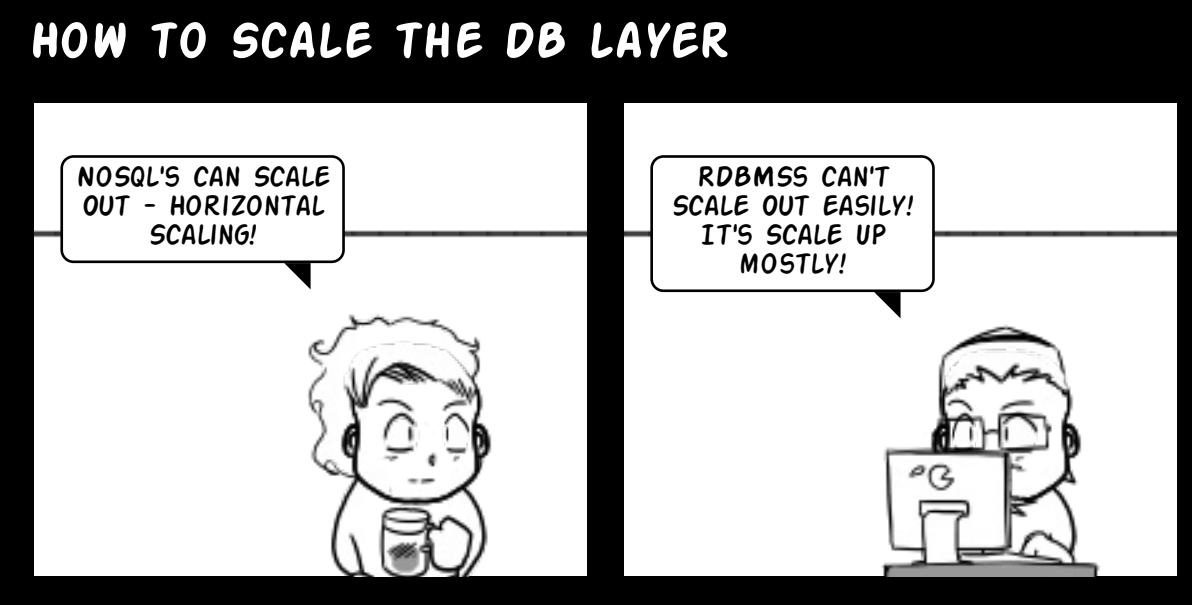
NFR's : Scalability



NFR's : Scalability

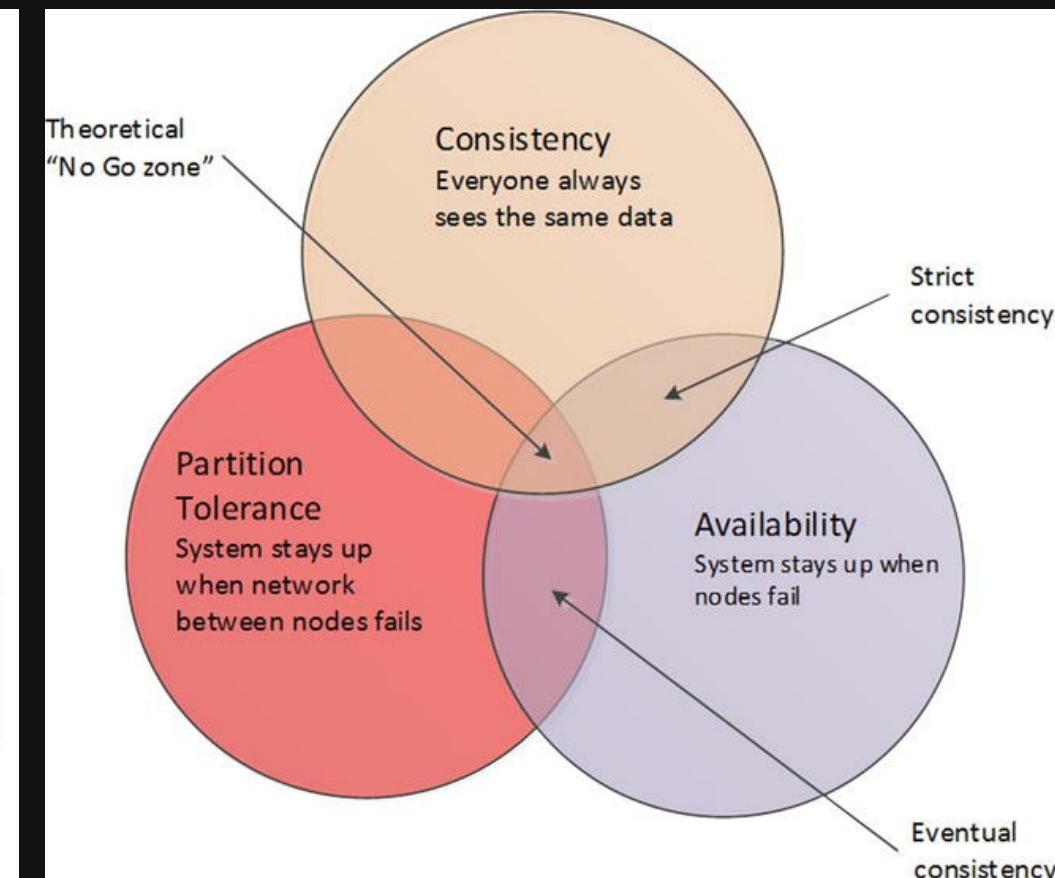
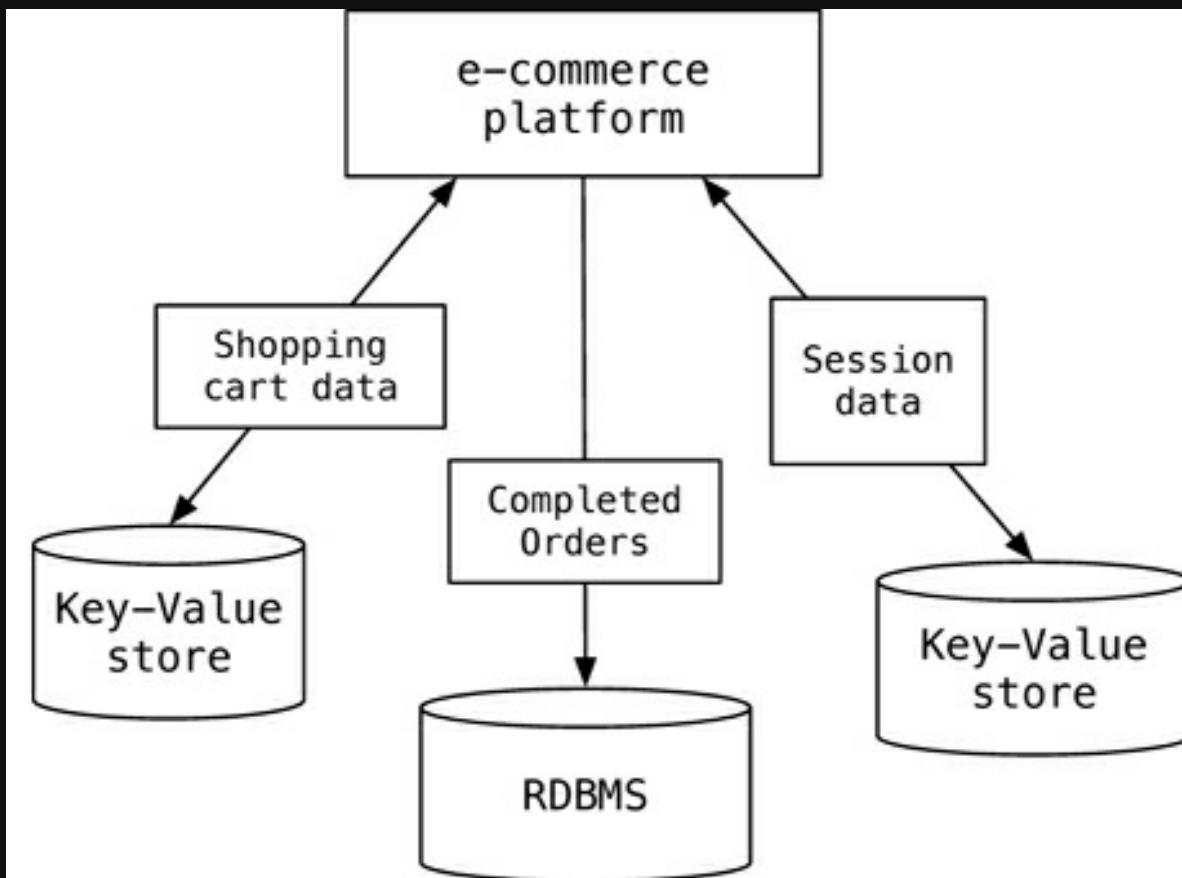


NFR's : Scalability



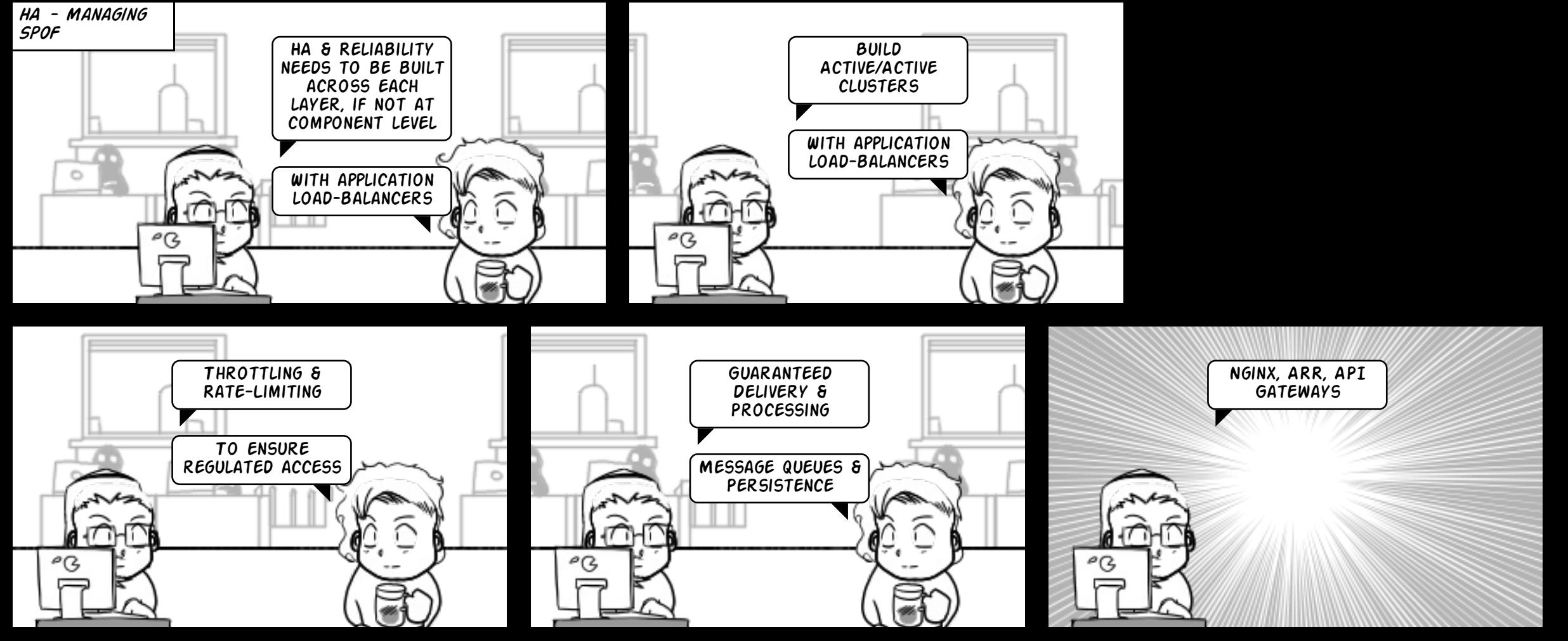
NFR's : Scalability

- Revisiting DB Scaling & CAP -



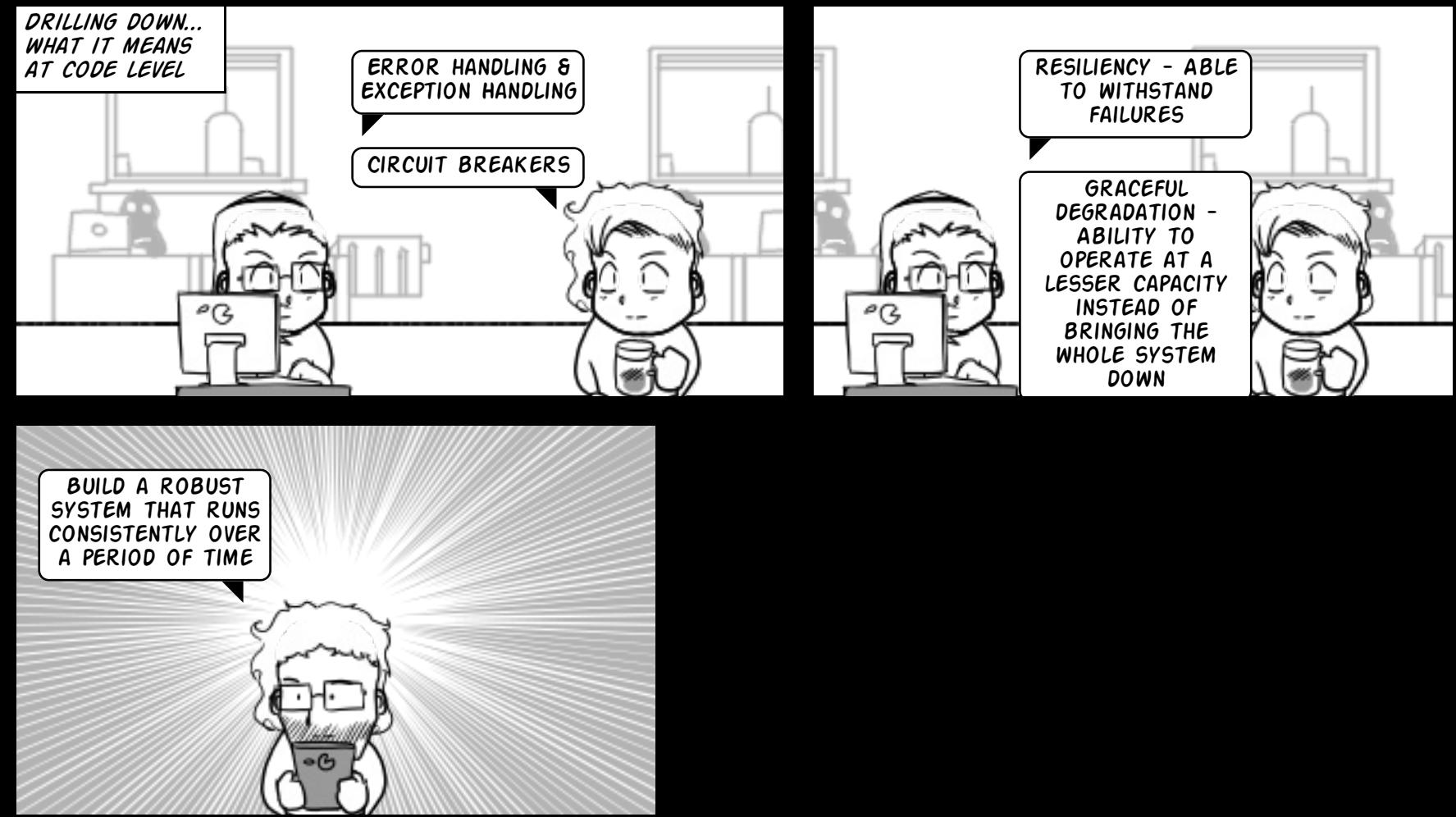
NFR's : HA & Reliability

HOW TO ENSURE HIGH AVAILABILITY OF THE WEB LAYER



NFR's : HA & Reliability

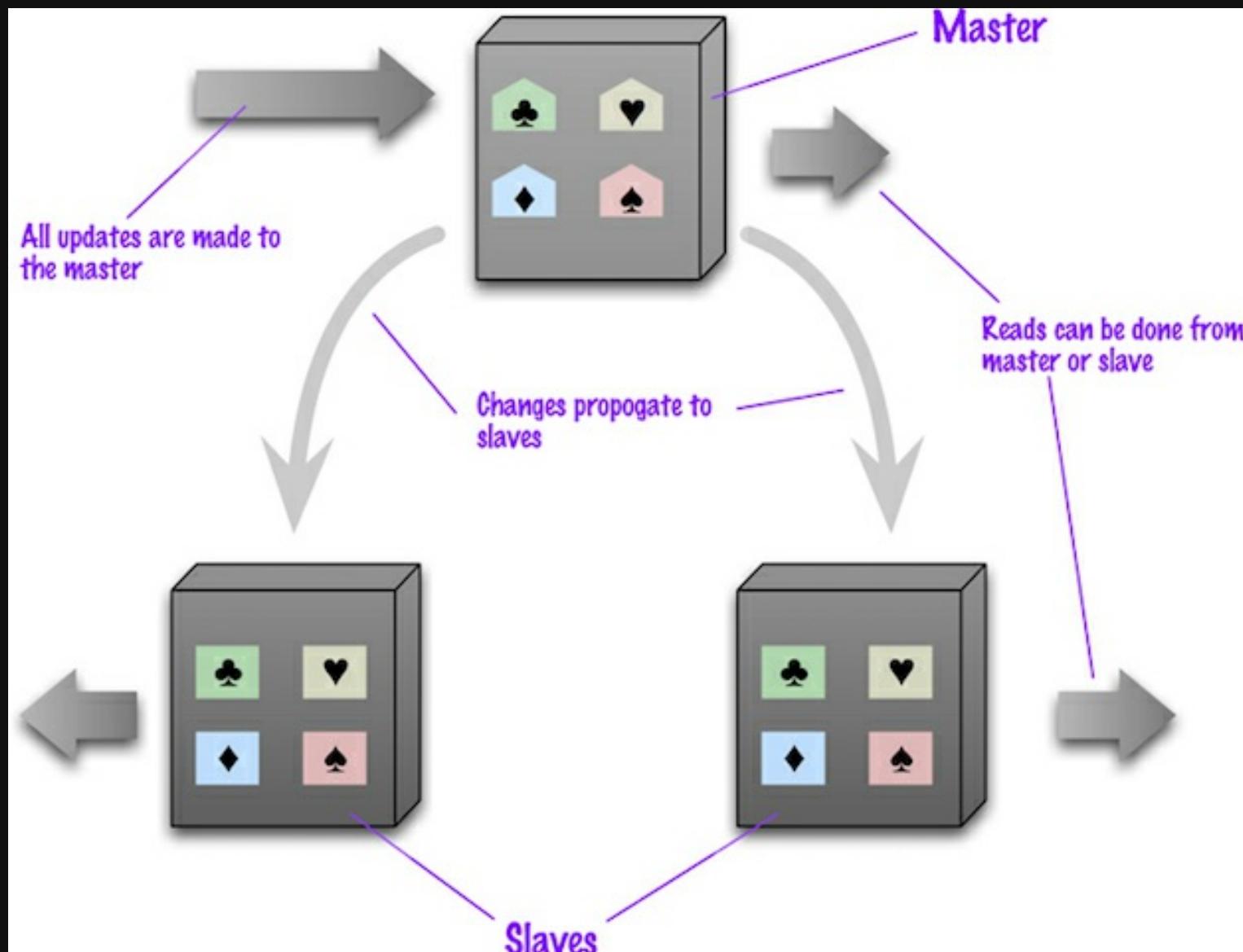
RELIABILITY, RESILIENCY & ROBUSTNESS



NFR's : HA & Reliability

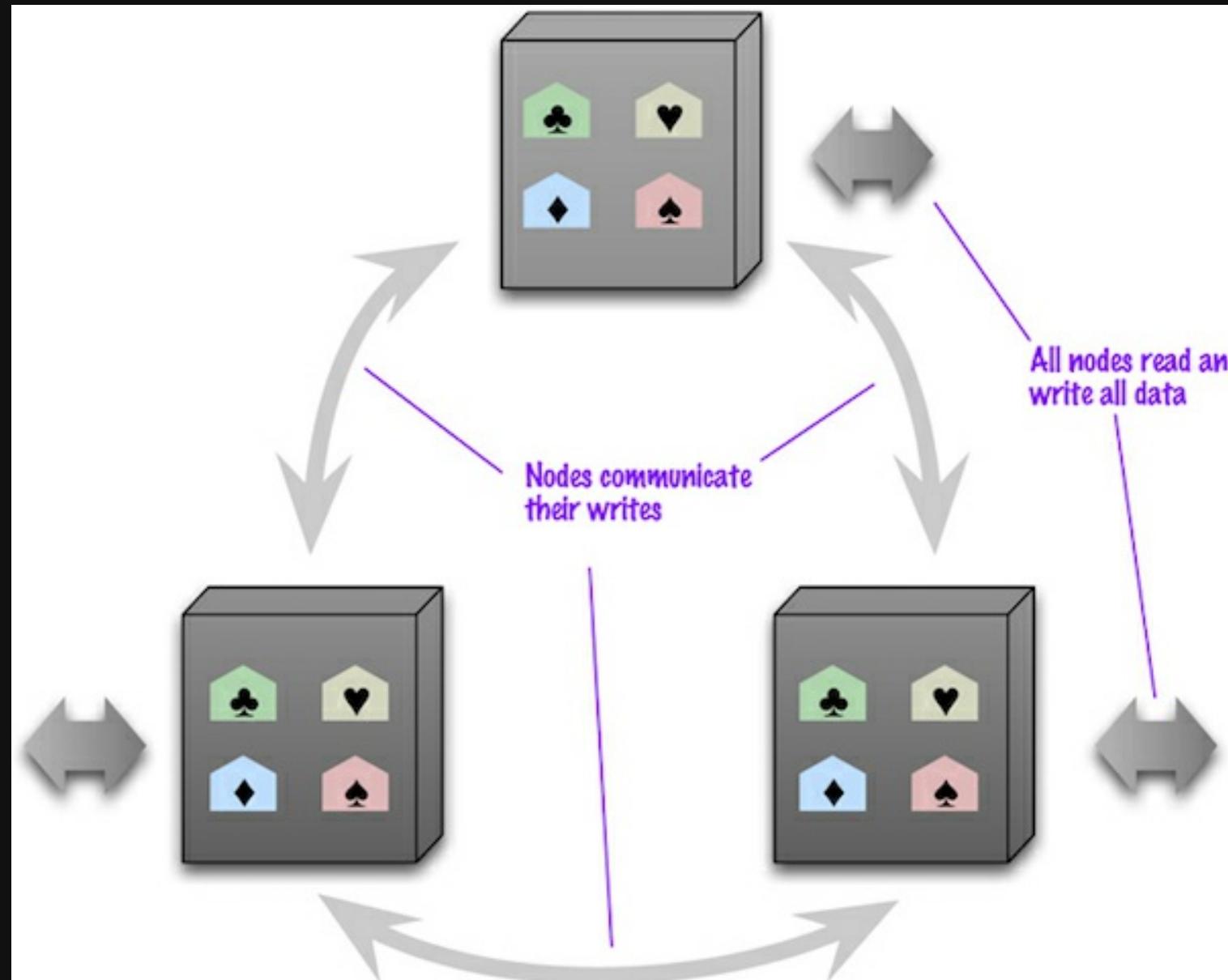
Scalability & HA go hand in hand!
HA Calls for more than one server;
An Active / Active HA cannot be possible without either Scale UP or Scale Out

NFR's : HA & Reliability



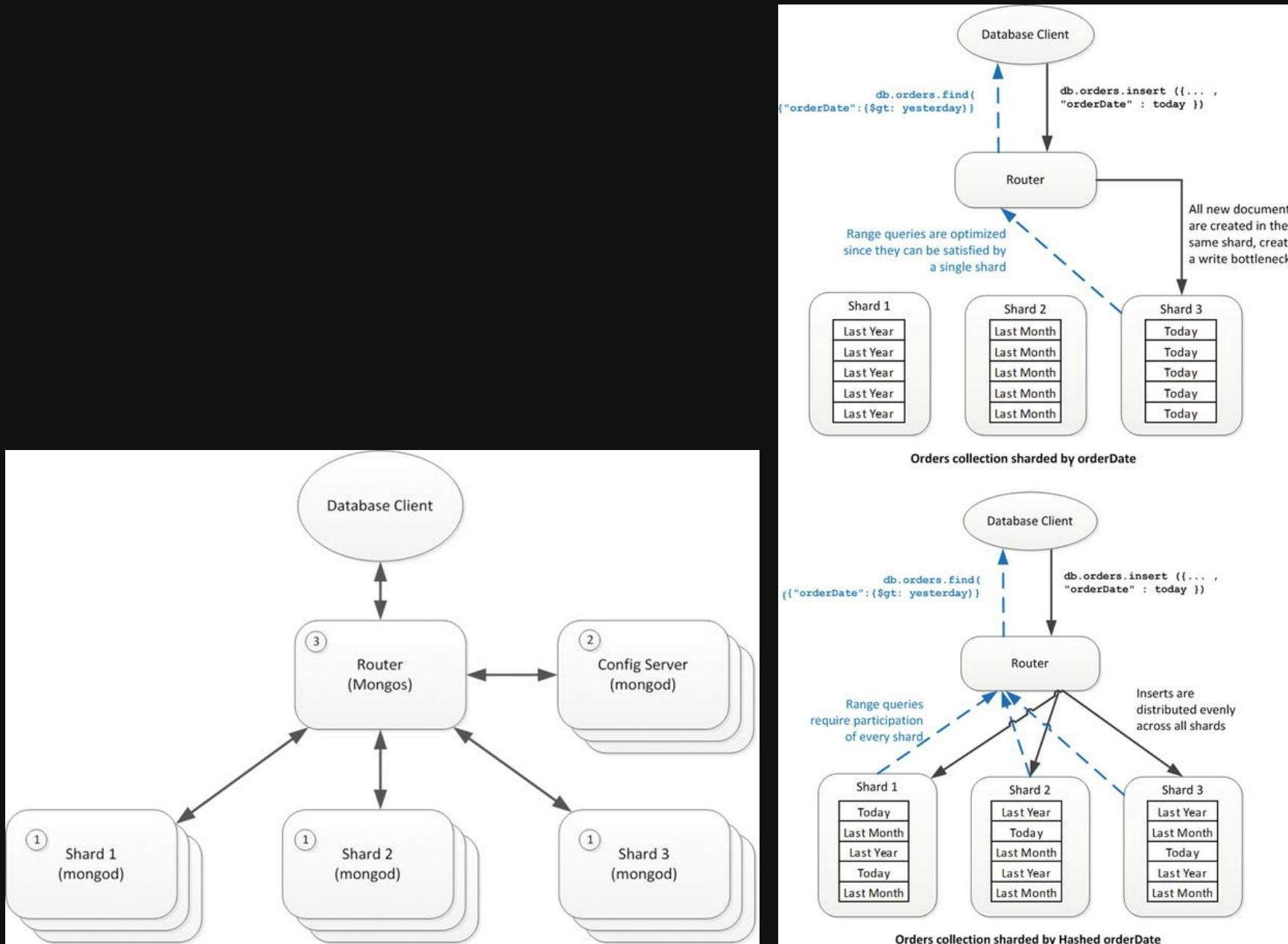
Master Slave (Active / Passive)

NFR's : HA & Reliability



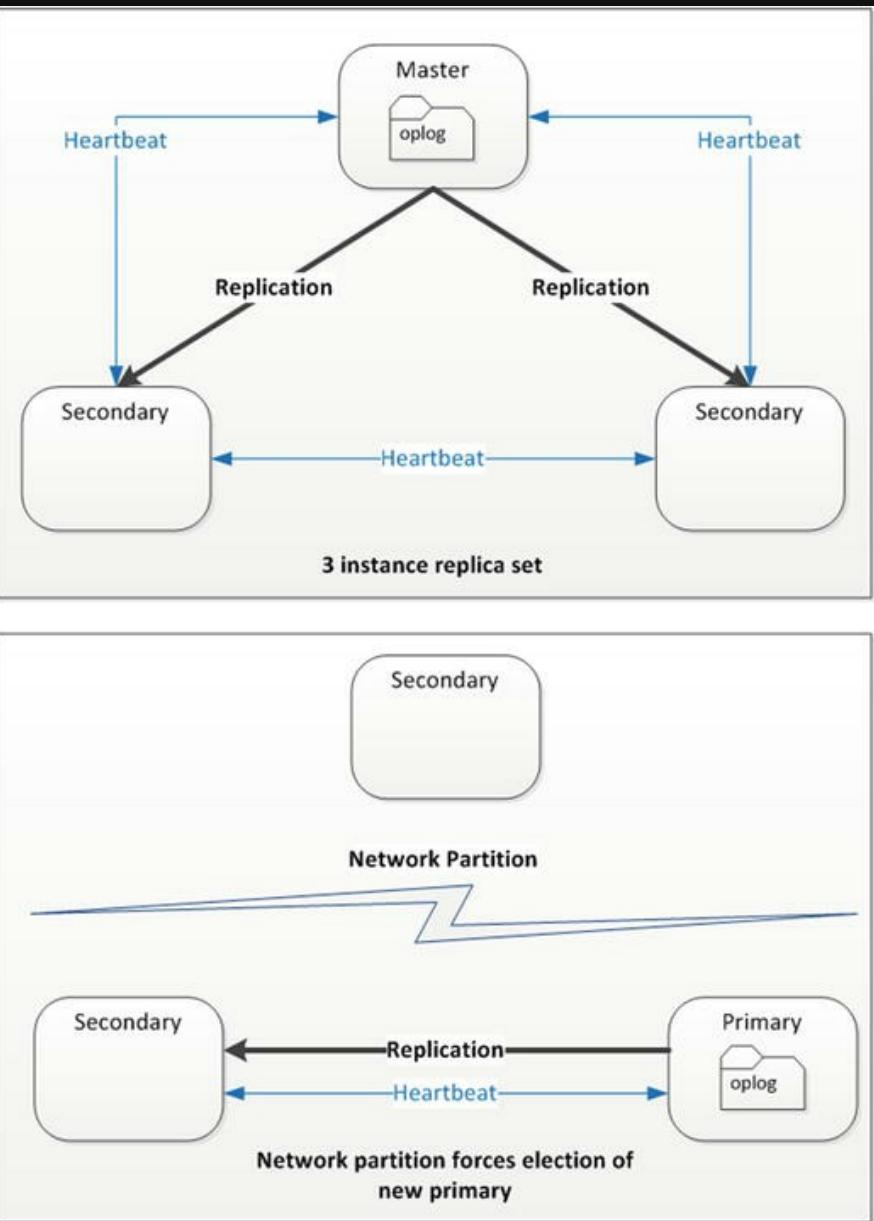
Master Slave : Peer - Peer (Active / Active)

NFR's : HA & Reliability



Sharding a NoSQL Database (Partition Data to manage scale)

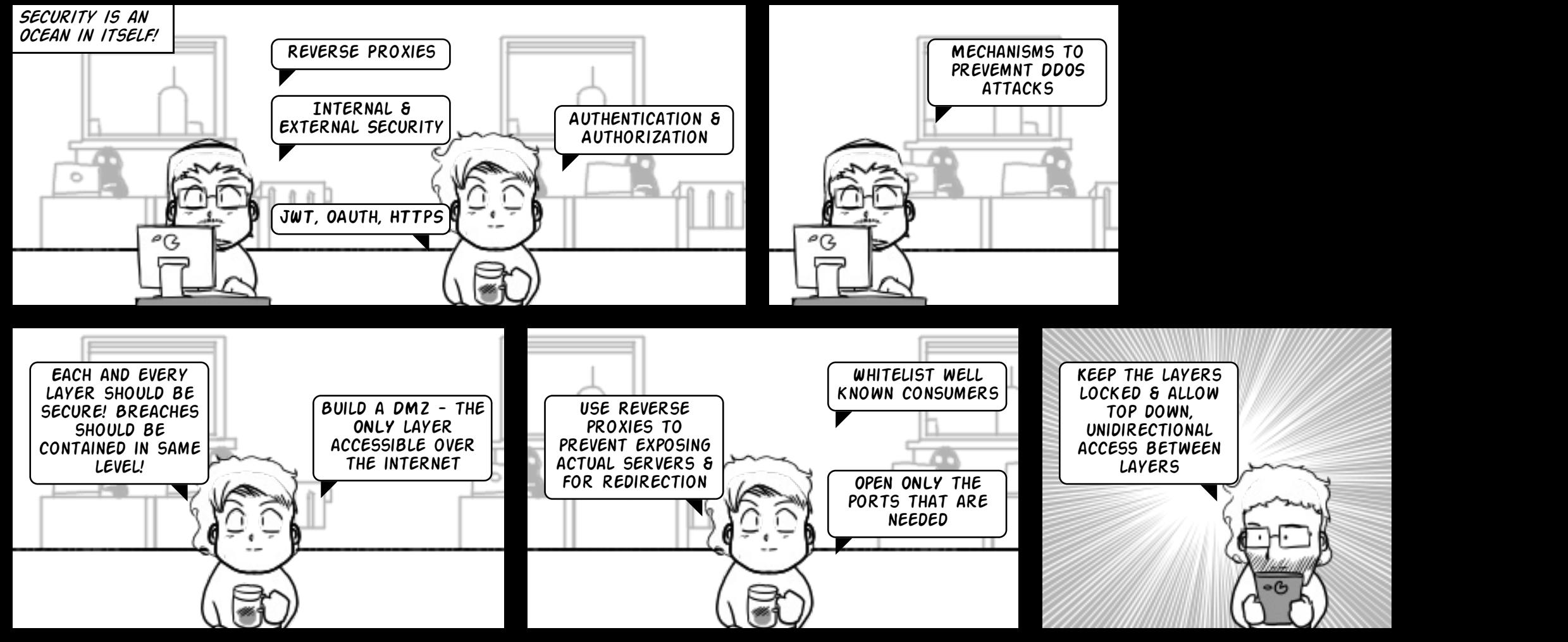
NFR's : HA & Reliability



Replication set (Active / Active, Active / Passive)

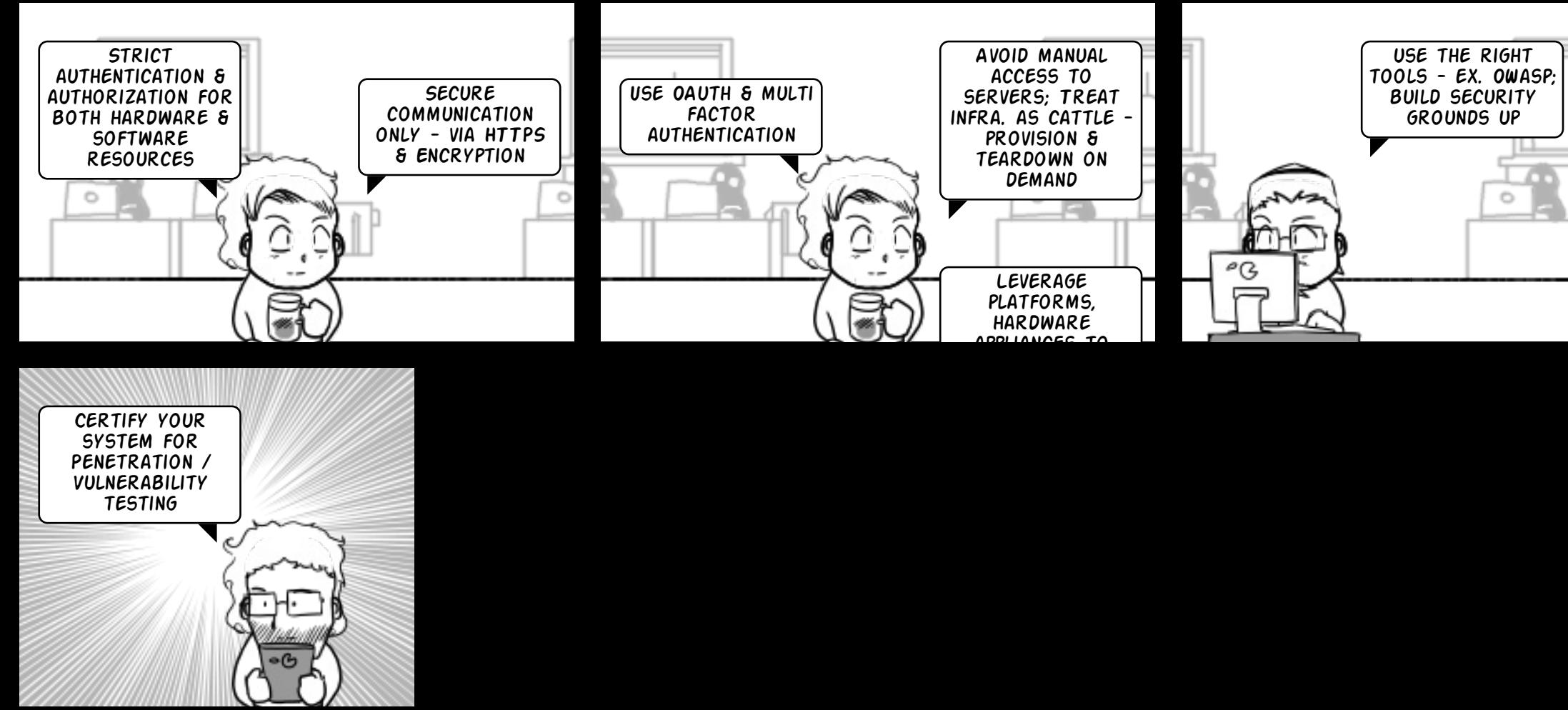
NFR's : Securing the System

SECURITY ACROSS LAYERS



NFR's : Securing the System

SECURE EVERYTHING - HARDWARE, SOFTWARE ACCESS



NFR's : Securing the System

Microsoft Patterns & Practices Guidance - Security

Microsoft Web Application Security: Threats and Countermeasures

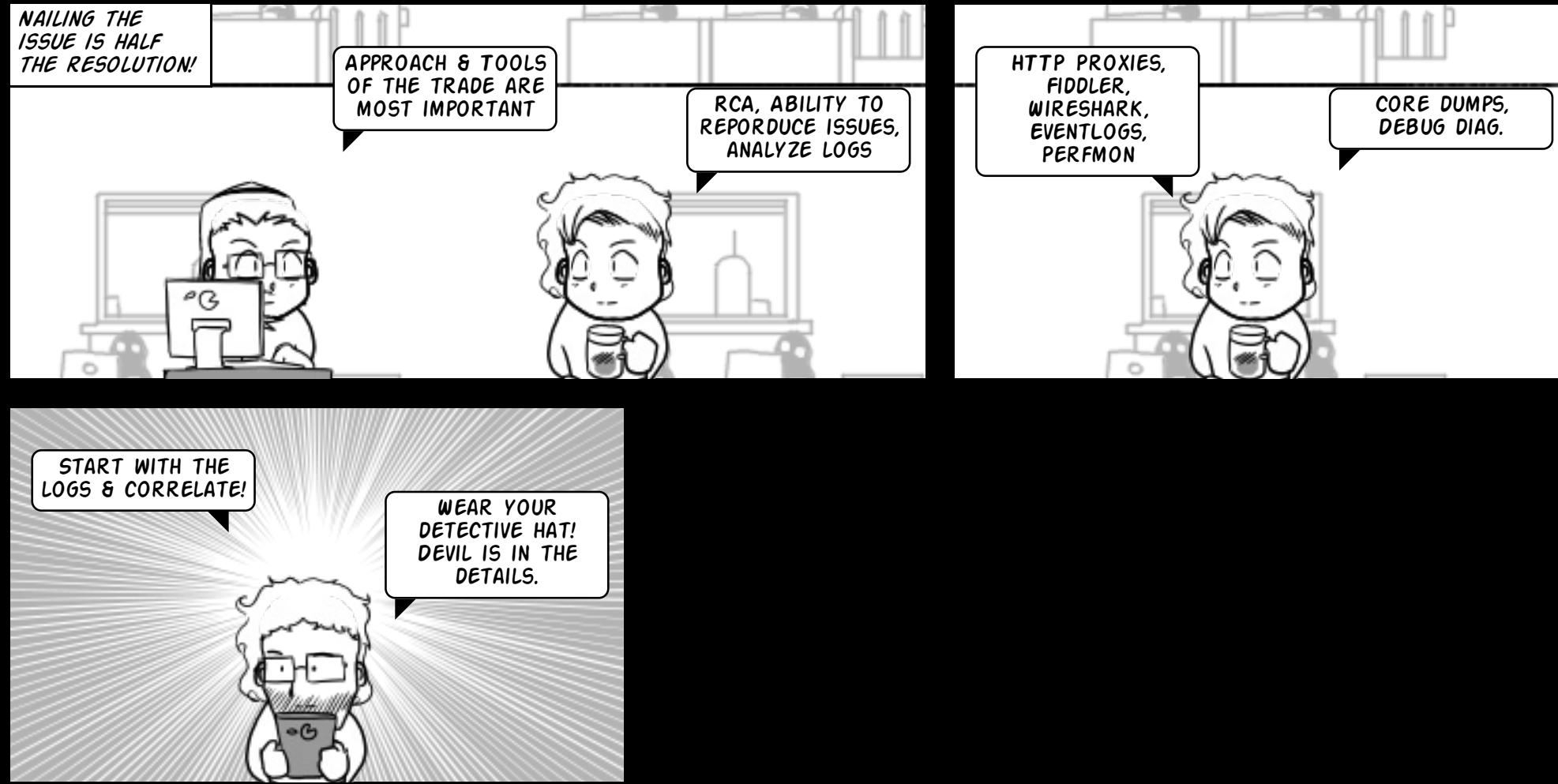
Azure Security

OWASP

Vulnerability Testing vs Penetration Testing

Operational needs

TROUBLESHOOTING & PROBLEM SOLVING



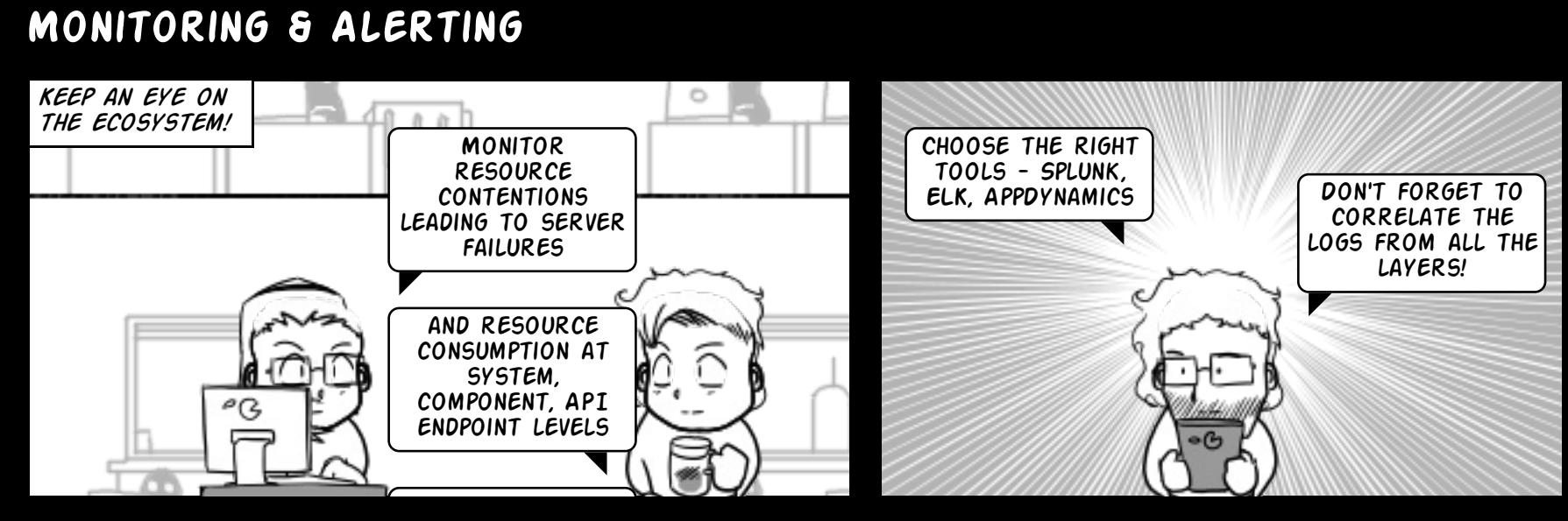
JConsole

VisualVM - All in One Java troubleshooting tool

JRockit Mission Control

Debugging .Net Framework Apps

Operational needs



Links

Donne Martin's System Design Primer

- The Single Most absolute reference and superbly organized collection of resources for system design -

Links

Sources every engineer & architect should follow!

InfoQ Tech. Hub

High Scalability - Real Time Architectures

DZone Tech. Hub

HackerNoon - Bleeding edge of tech.

Hacker News

Links

There's much much more, and these links give you good starting points

[NetFlix OSS](#)

[Everything about MicroServices](#)

[Cloud Foundry](#)

[Cloud Native Computing Foundation](#)

[A curated list of awesome resources](#)

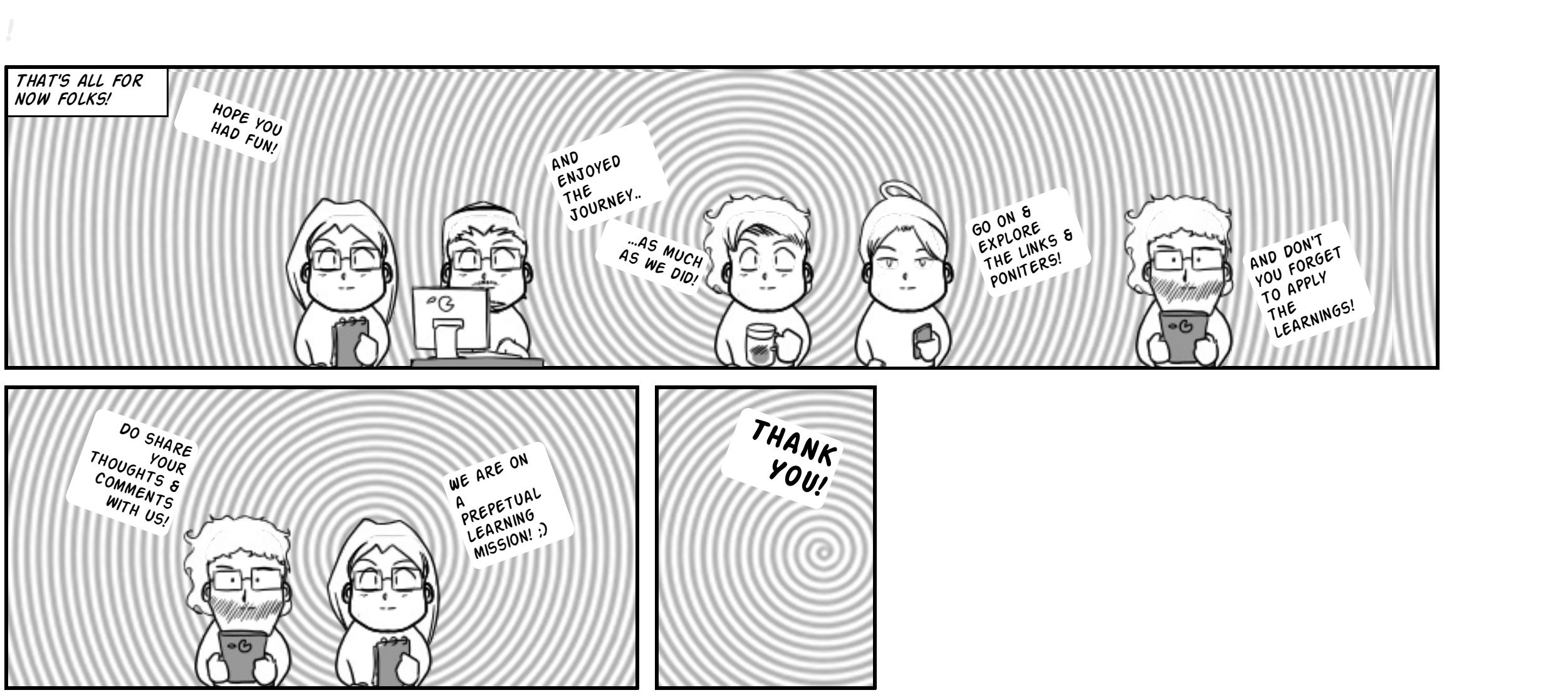
[Foundations of Software Engineering](#)

[System Design Inputs](#)

[Awesome Software Craftsmanship](#)

[Awesome BigData - All about data stores & databases](#)

[Microservice Patterns](#)





A journey of a thousand miles
begins with a single step
... And lots of coffee!

Visit Jim Hunt at facebook.com/huntcartoons