# COMS E6998: **Microservices and Cloud Applications**

*Lecture 1: Introduction*

Dr. Donald F. Ferguson
dff9@columbia.edu

© Donald F. Ferguson, 2017. All rights reserved.

# Contents



© Donald F. Ferguson, 2017. All rights reserved.

# Contents

- Introduction
  - A bit about your instructor.
  - Coursework and grading.
  - Logistics.
- Course Overview and Contents
  - Schedule and topics.
  - Cloud
  - Microservices
  - Cloud applications.
- An example “microservice, cloud application”
  - Demo.
  - Overall structure.
- 1st Assignment



© Donald F. Ferguson, 2017. All rights reserved.

# Introduction



© Donald F. Ferguson, 2017. All rights reserved.

# Your Instructor



© Donald F. Ferguson, 2017. All rights reserved.

# A Little About Me

- Columbia through and through
  - BA (Columbia College) '82, M.S. '84 (Engineering), M.Phil. '84 (GSAS).
  - Ph.D. '87 (GSAS) – Thesis: "The Application of Microeconomics to the Design of Resource Allocation and Control Algorithms."
  - Adjunct Professor at Columbia Univ: 2006, 2009, 2014-2017. 7 "Topics in CS" courses.
- 32 years of technology contribution and leadership.
  - VP, IBM Fellow: IBM Research, IBM Software Group.
  - Microsoft Technical Fellow.
  - Distinguished Engineer, CTO, Executive VP, CA technologies.
  - VP, Sr. Fellow, CTO, Dell Software Group.
  - Co-Founder, CTO Seeka TV (www.seeka.tv).
- Personal
  - Two fantastic daughters; high school and one is a senior at Barnard.
  - Black belt in Kenpo Karate (http://www.tracyskarateworldwide.com/about.html, www.elitedefensivetactics.com )
  - 2LT in New York Guard (http://dmna.ny.gov/nyg/)
  - Conversational Spanish, learning Arabic
  - Hobbies: Bicycling, Astronomy


© Donald F. Ferguson, 2017. All rights reserved.

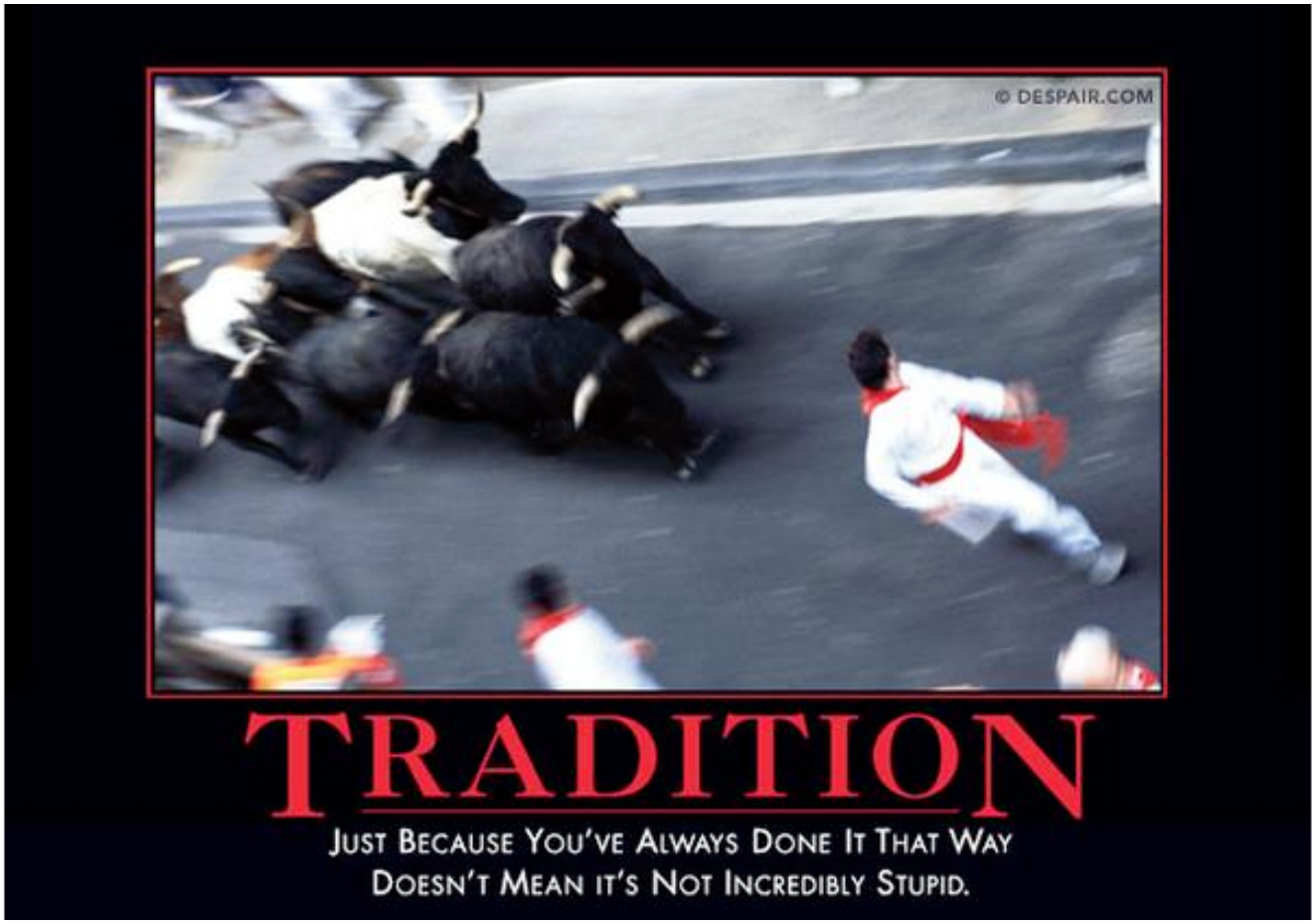

g), M.Phil. '84 (GSAS).

- Personal
  - Two daughters
  - Black belt in Kenpo Karate
  - 2LT in New York Guard (http://dmna.ny.gov/nyg/
  - Conversation Spanish, learning Arabic
  - Hobbies: Bicycling, Astronomy



© Donald F. Ferguson, 2017. All rights reserved.

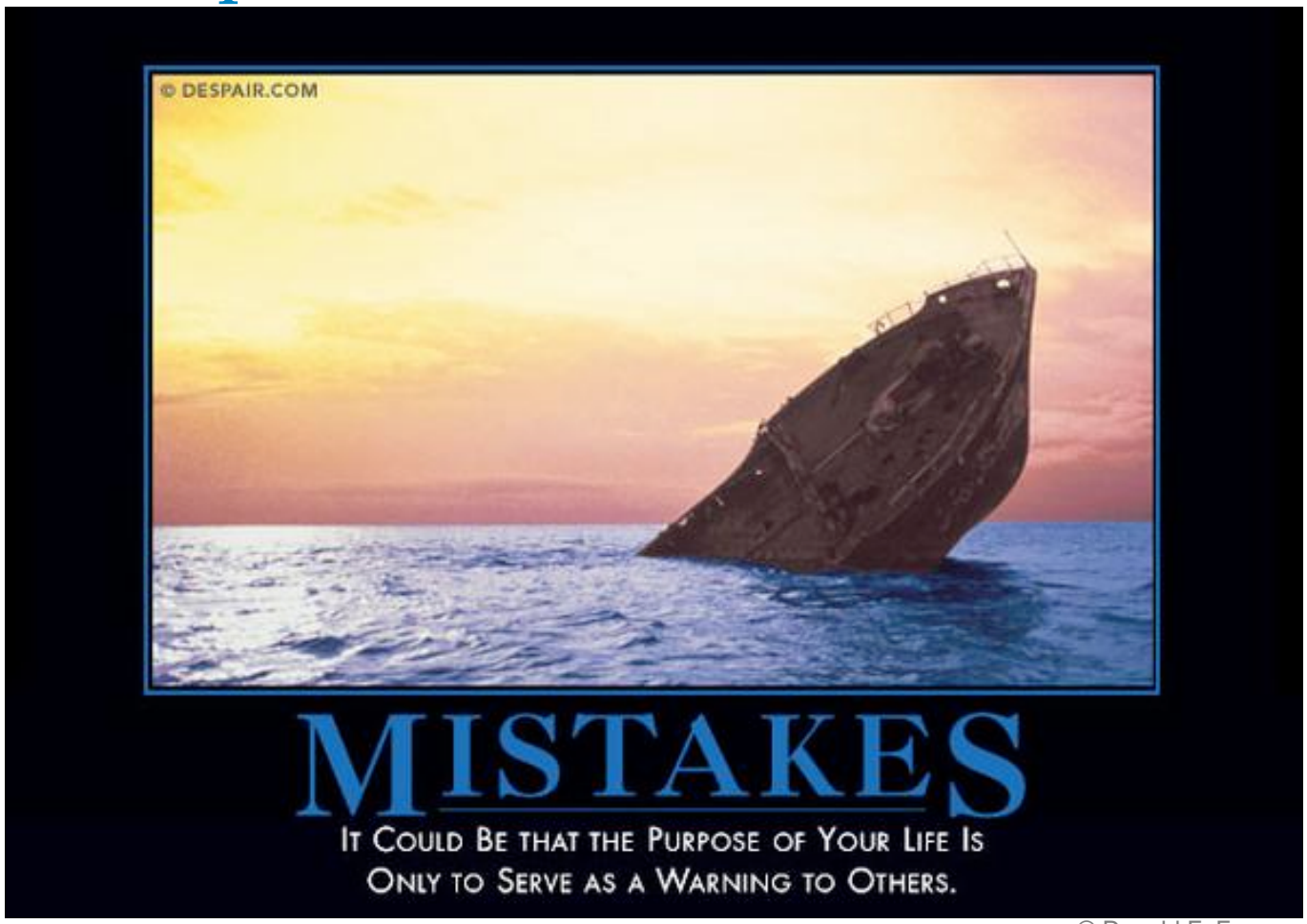
# A Lot of Experience






© Donald F. Ferguson, 2017. All rights reserved.

# A Lot of Experience






© Donald F. Ferguson, 2017. All rights reserved.

# A Lot of Experience

I may not be able to teach you the right way to do something,

but

I can show you plenty of wrong ways to avoid.



© Donald F. Ferguson, 2017. All rights reserved.

# Coursework Grading



© Donald F. Ferguson, 2017. All rights reserved.

# Coursework and Grading

- No midterm, final or exams.
- Five 20 point projects, each of which extends previous projects.
  - Approximately every two weeks, with two weeks to complete.
  - Teams
    - Approximately 5 team members.
    - Each team has a designated *primary contact* and *team name*.
  - Each project has the following sub-deliverables, packaged in a zip file with a readme.
    - A set of user stories.
    - A short, simple architecture/design description with diagrams.
    - Demo
    - Application artefacts, e.g. code, DB schema/data, HTML, URL for testing, …
  - There will be a simulated "project review," which is a meeting with presentation, demo, Q&A, etc.
- In addition to the technology, I am trying to *mentor* you to learn how to be a successful technical professional. You will experience very basic concepts in
  - Product management.
  - Presentation and documentation skills.
  - **Transforming vague requirements into something clear and implementable.**
  - Q&A and discussion of alternatives.
  - Teamwork and team dynamics.

This will drive you crazy in the beginning.



© Donald F. Ferguson, 2017. All rights reserved.

# Logistics



© Donald F. Ferguson, 2017. All rights reserved.

# Logistics

- Course material
  - *No textbooks*. Technology changes too quickly, and we will cover concepts and technologies from several disciplines.
  - Lecture and lecture notes will provide links to online information.
  - Web search is your first and best resource.
- CAs:
  - We have two excellent CAs, with whom I have previous worked.
  - Will introduce themselves via email, Piazza and other collaboration environments.
- Office hours
  - Professor: Tu/Th 10:15 to 12:00, or by appt (typically Google Hangout).
  - CAs: TBA
- Collaboration environments:
  - CourseWorks for submitting projects.
  - Piazza: Slides, discussion (piazza.com/columbia/fall2017/comse6998_014_2017_3topicsincomputerscience)
  - Slack: Good way to have short discussions, get ahold of me, instant messages, etc.
    - Team: (https://join.slack.com/t/dff-columbia/shared_invite/MjM1MjA5NjAwOTk2LTE1MDQyODExOTQtNzEzM2FjZmMzMw).
    - Join channel #e6998-2017
- We will use Amazon Web Services, but the concepts are general.
- **Questions and discussion are expected, appreciated and essential.**



© Donald F. Ferguson, 2017. All rights reserved.

# Course Overview and Content

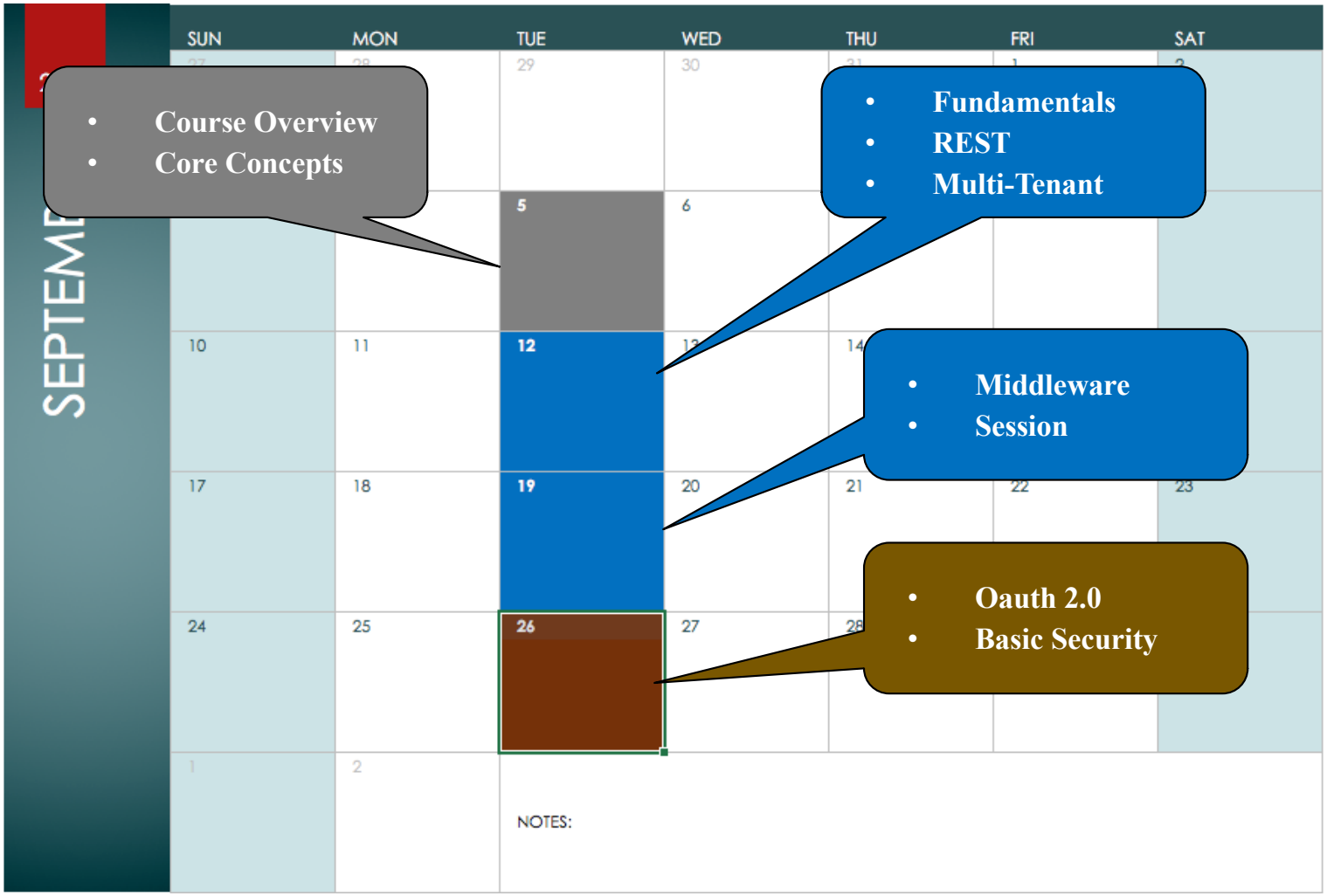


© Donald F. Ferguson, 2017. All rights reserved.

# Course Overview

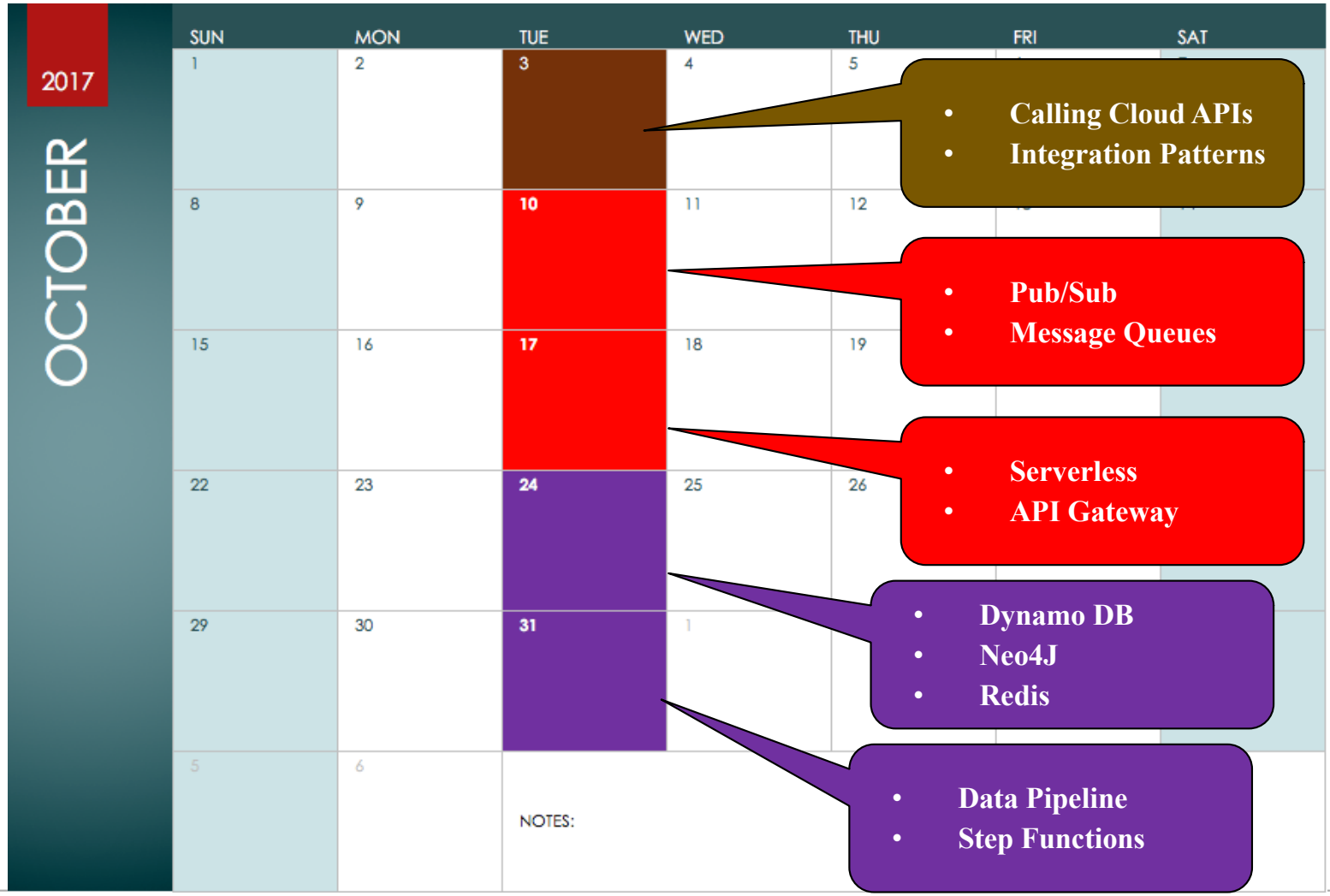


© Donald F. Ferguson, 2017. All rights reserved.



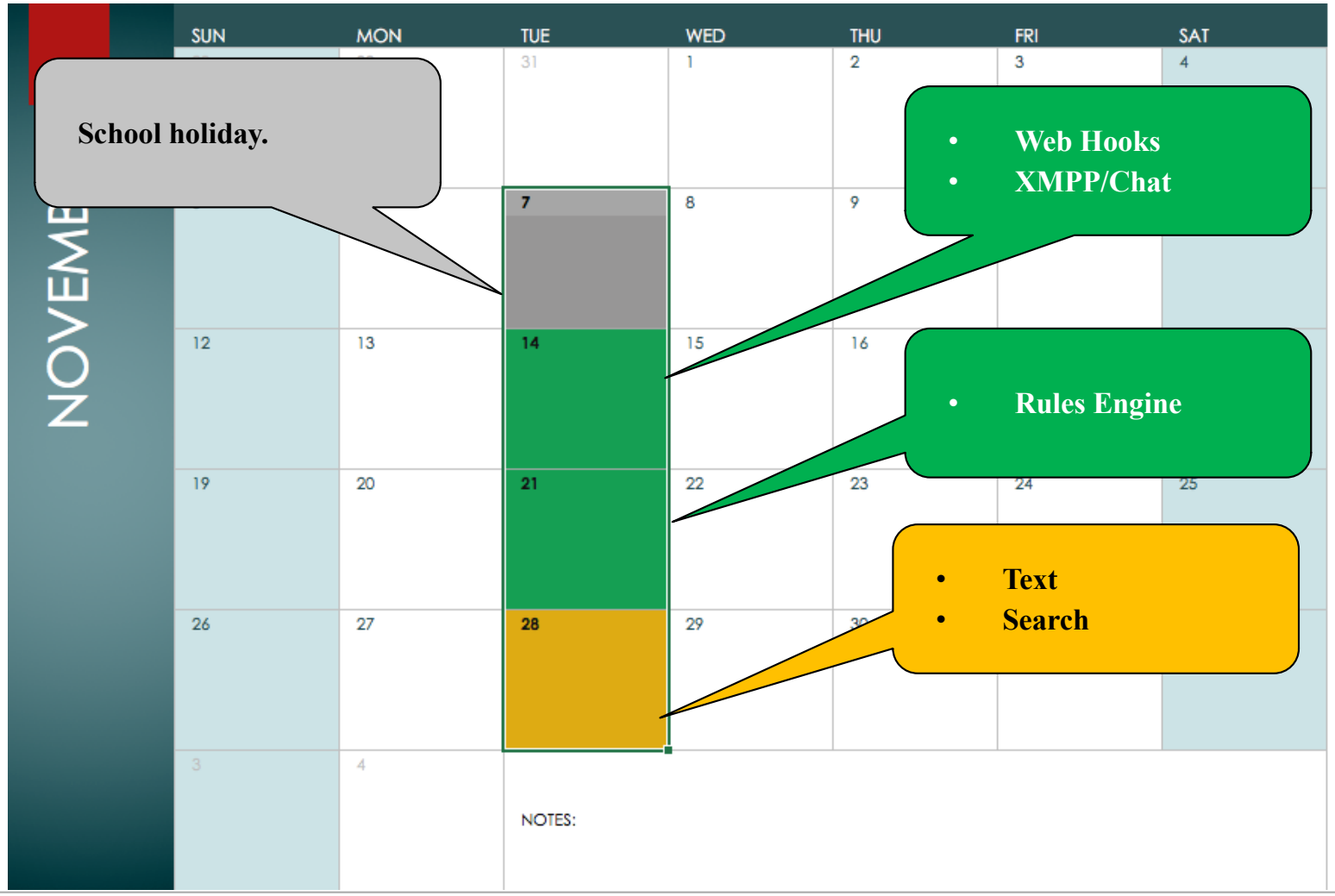



© Donald F. Ferguson, 2017. All rights reserved.



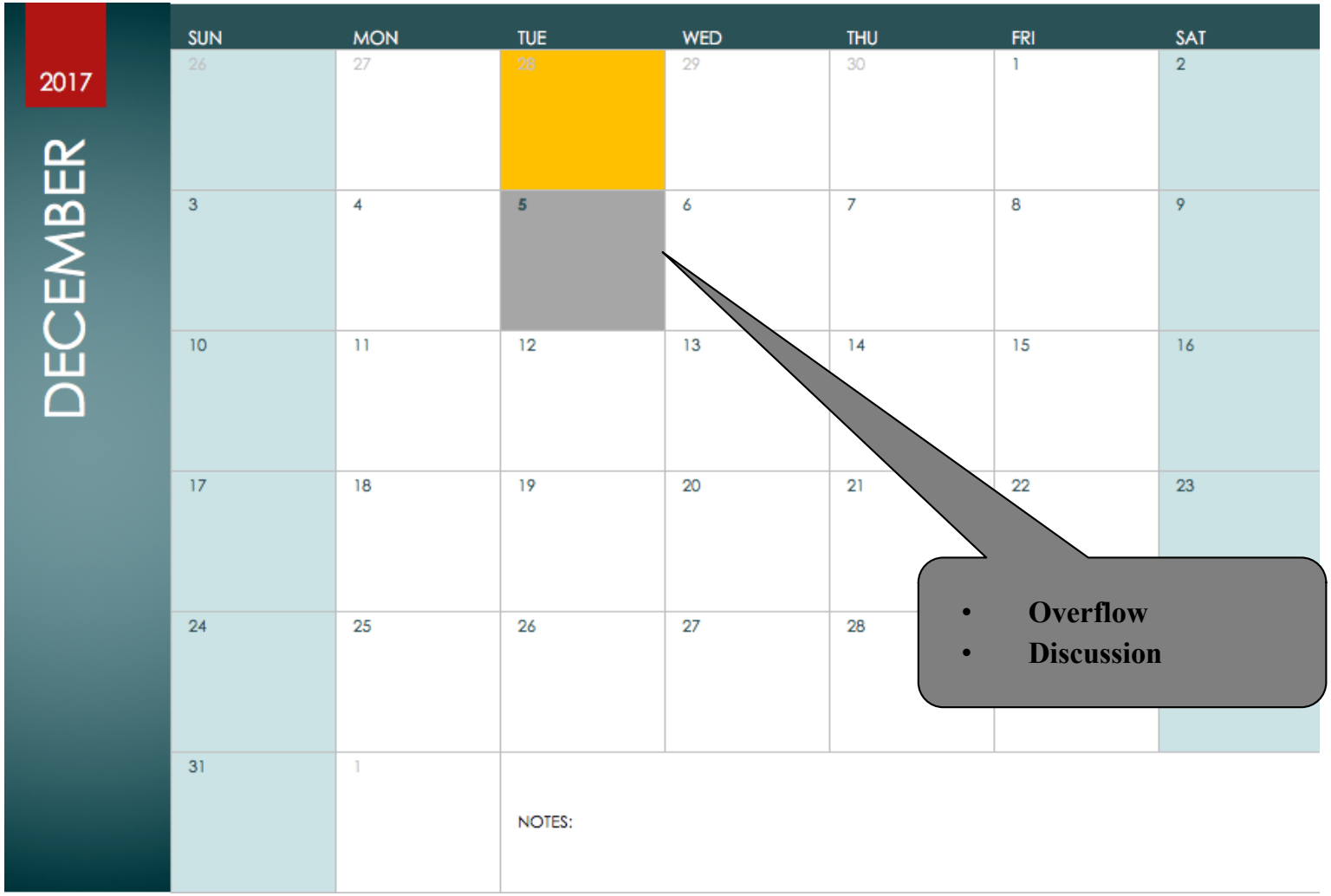



© Donald F. Ferguson, 2017. All rights reserved.






© Donald F. Ferguson, 2017. All rights reserved.






© Donald F. Ferguson, 2017. All rights reserved.

# Comments

- This course is about building applications
  - That apply best practices and design patterns,
  - Using a broad set of technologies,
  - Each of which is several lectures or a complete course unto itself.

  We will cover the essentials but not the depth of the individual technologies.
- One place where we will ”go deep” is on the basic structure of a well-designed application/set of microservices. Some example topics are
  - Model-View-Controller.
  - Application Tier-Data Tier.
  - Encapsulation/Abstraction.
  - REST API

  *My experience from prior courses has been that I have gone through these topics too quickly, causing students to do advanced topics with a poor design.*


© Donald F. Ferguson, 2017. All rights reserved.

# Course Content:

Cloud
Microservices
Applications



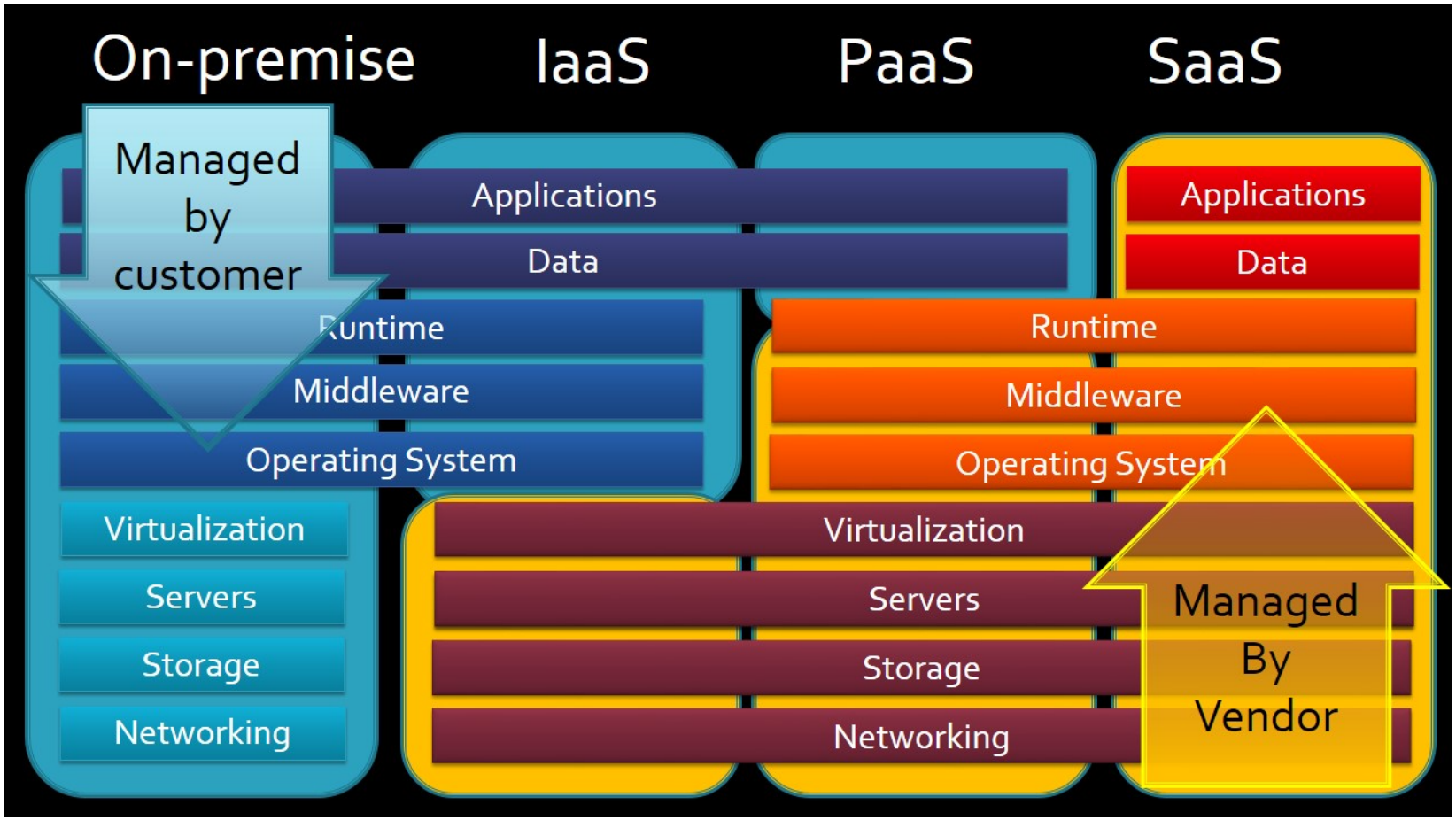
© Donald F. Ferguson, 2017. All rights reserved.

# Cloud



© Donald F. Ferguson, 2017. All rights reserved.

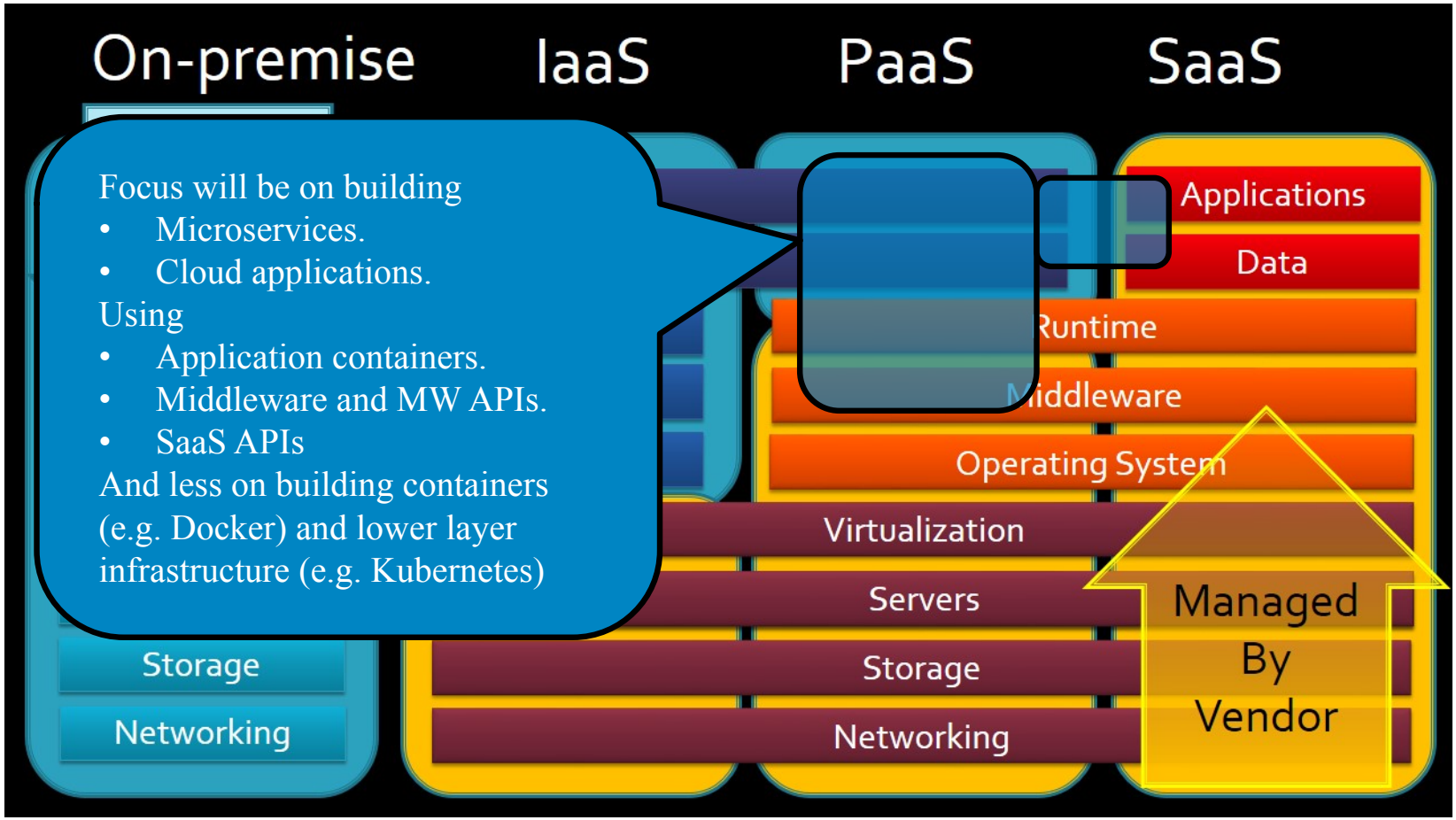
# IaaS, PaaS, SaaS






© Donald F. Ferguson, 2017. All rights reserved.

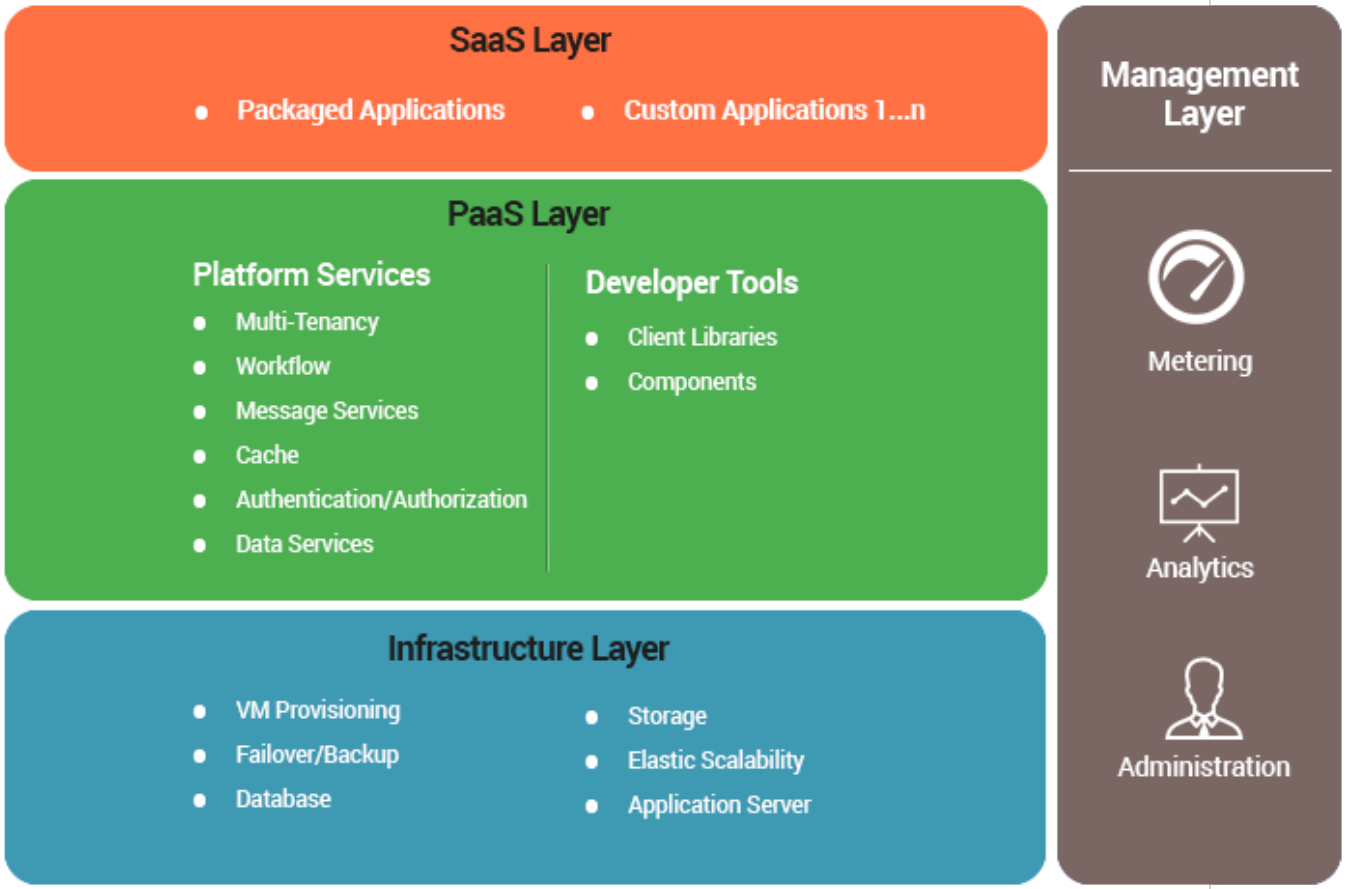
# IaaS, PaaS, SaaS






© Donald F. Ferguson, 2017. All rights reserved.

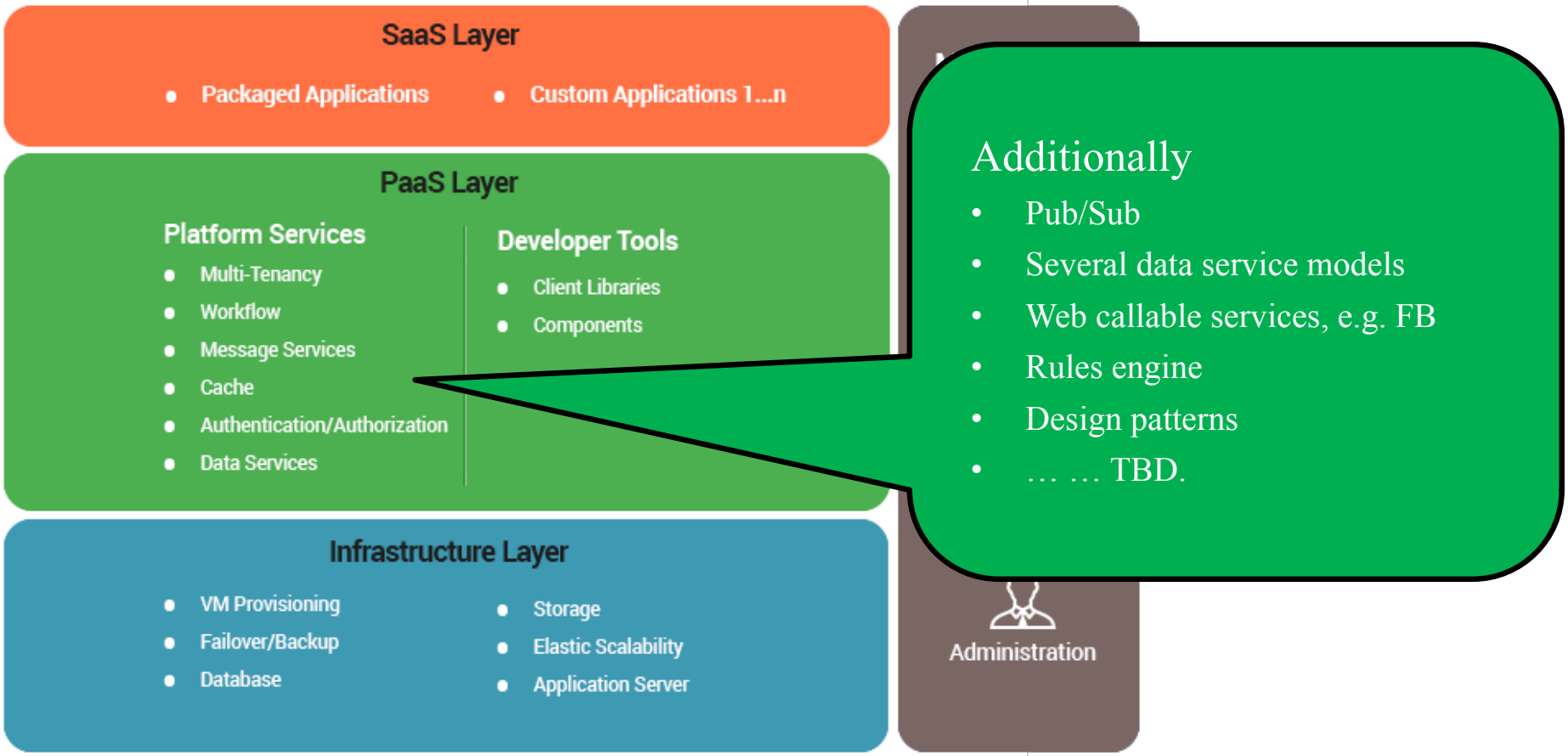
# IaaS, PaaS, SaaS – A Little More about PaaS






© Donald F. Ferguson, 2017. All rights reserved.

# IaaS, PaaS, SaaS – A Little More about PaaS






© Donald F. Ferguson, 2017. All rights reserved.

# Cloud is about *- as-a-Service

(https://en.wikipedia.org/wiki/As_a_service)

"***aaS** is an acronym for **as a service**, and refers to something being made available over the Internet to a customer as a service.[1] Examples include:"

Like any list, this is an overview and

- Incomplete, e.g.
  - Business-APIs-as-a-Service: Payments is an example, but there are others, e.g.
    - Order Management, Inventory, Shipping, …
    - Calendar and events. Contact management.
    - Human Resources.
  - Information-as-a-Service
    - Demographics.
    - Credit history and information.
    - Market data, prices, etc.
- Overly simplistic, e.g.
  - IaaS is many types of compute, storage, networking, ...
  - There are many, many types of PaaS
    - Integration-Platform-as-a-Service
    - More classical app/container, e.g. Cloud Foundry
    - Force.com and high-level component assembly.

| Service | Abbr. |
|---|---|
| Data as a service | DaaS |
| Desktop as a Service | |
| Energy Storage as a Service | ESaaS |
| Function as a Service | |
| Identity as a service | IDaaS |
| Infrastructure as a service | IaaS |
| IT as a service | |
| Logging as a service | LaaS |
| Mobile backend as a service | |
| Network as a service | |
| Payments as a service | |
| Platform as a service | PaaS |
| Recovery as a service | |
| Robot as a service | |
| Search as a service | |
| Security as a service | |
| Software as a service | SaaS |
| Storage as a service | |

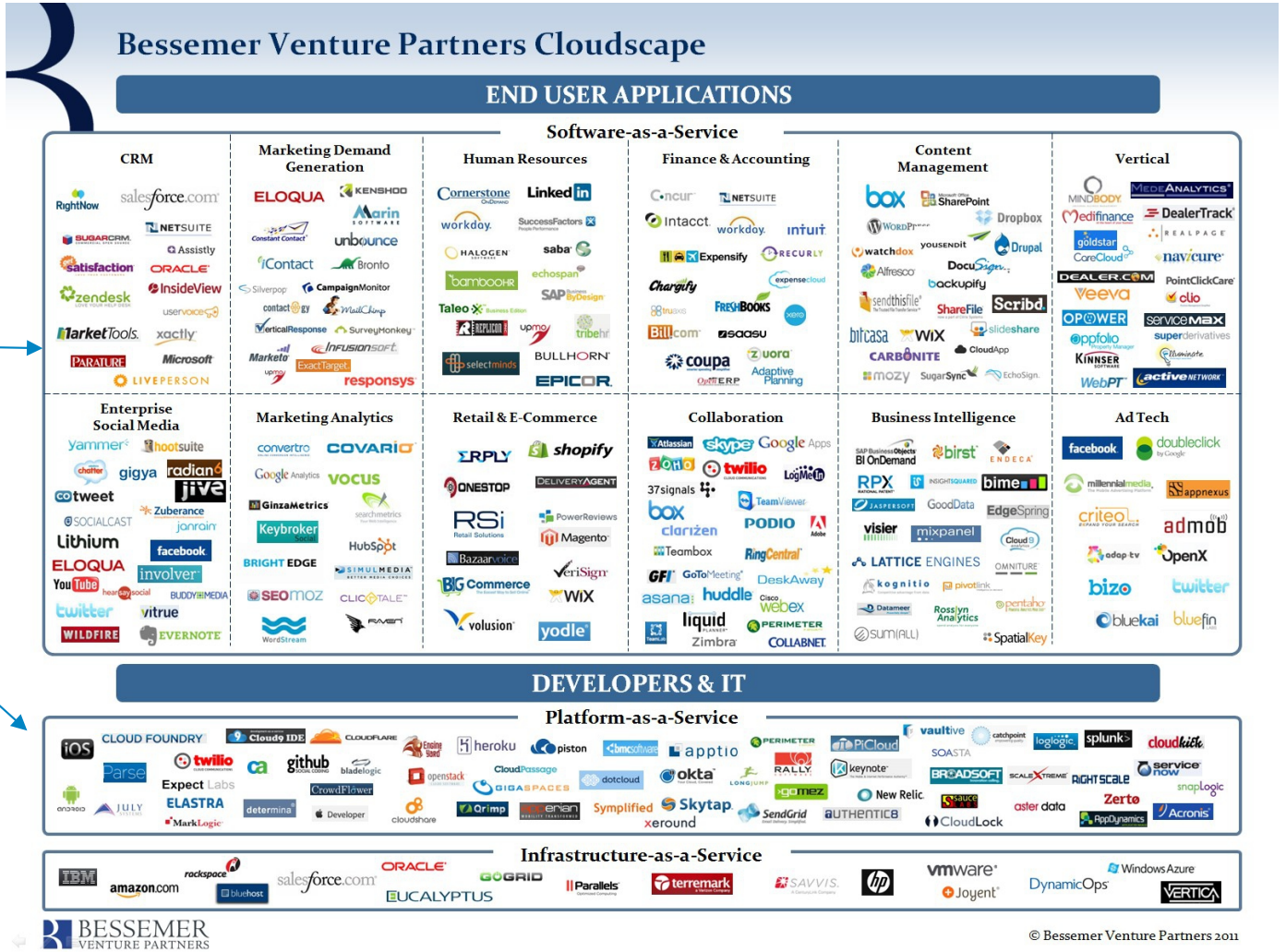


© Donald F. Ferguson, 2017. All rights reserved.

# Cloudscape

Our applications will

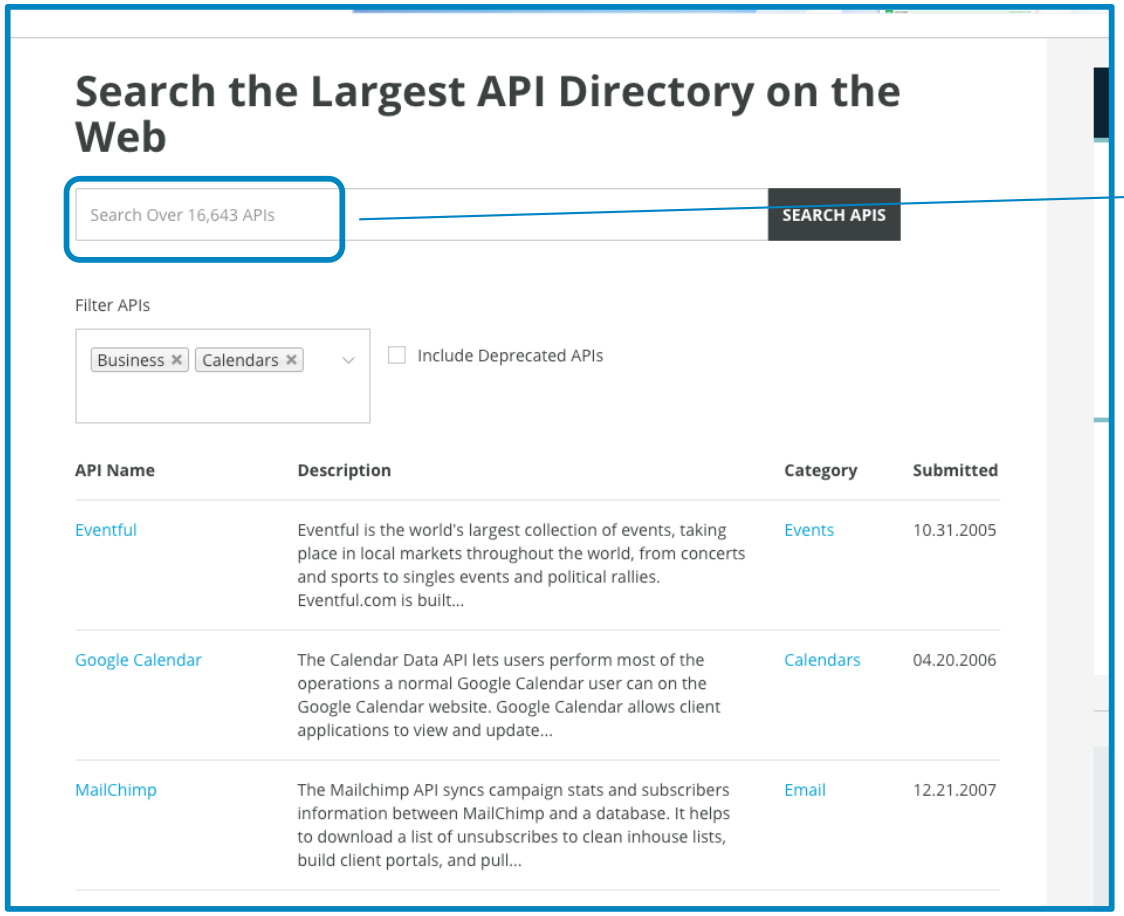
- Call APIs
- Use and run in PaaS






© Donald F. Ferguson, 2017. All rights reserved.

# www.programmableweb.com (Example)




Over 16,000 APIs



© Donald F. Ferguson, 2017. All rights reserved.

# Microservices



© Donald F. Ferguson, 2017. All rights reserved.

# Microservices (https://martinfowler.com/articles/microservices.html)

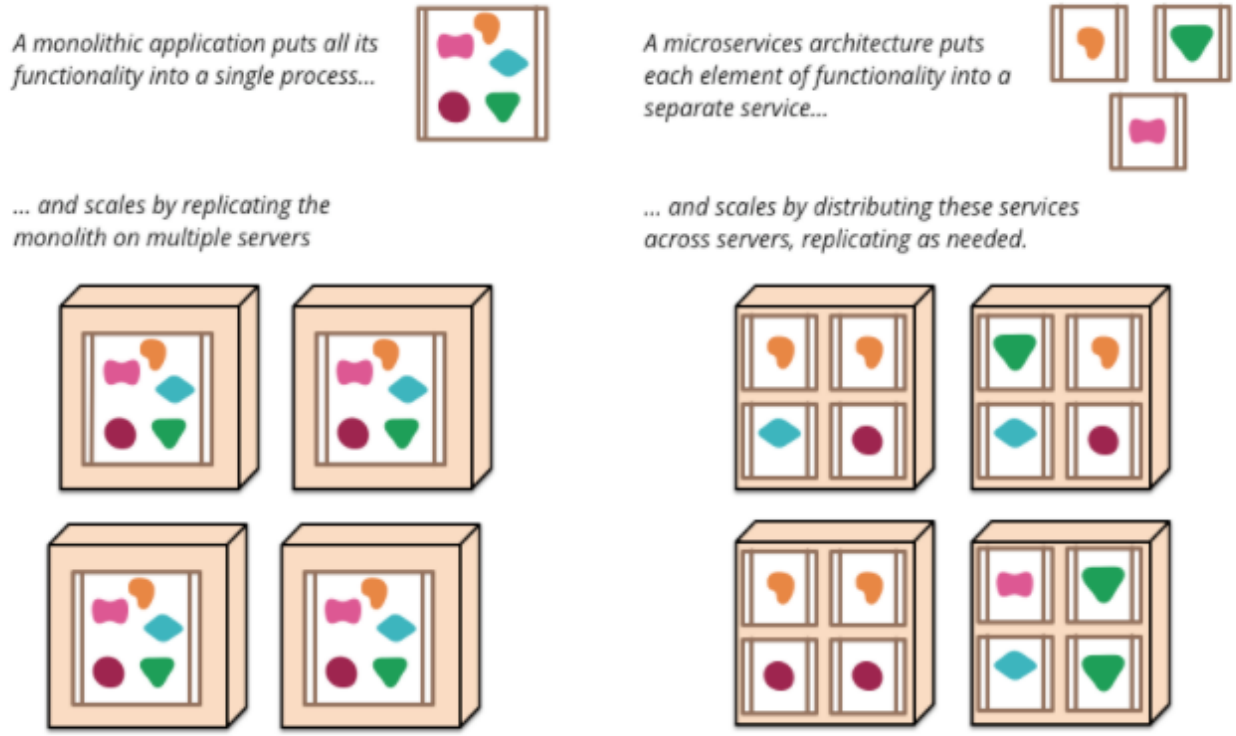
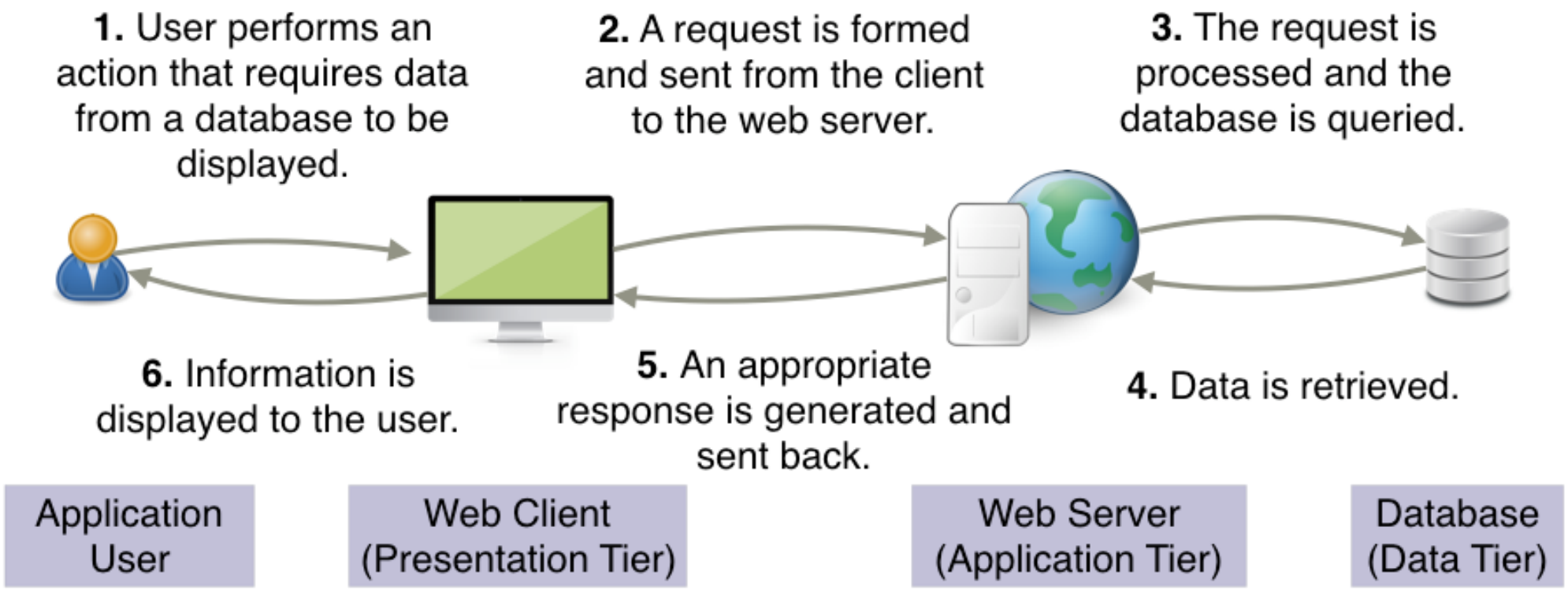



*Figure 1: Monoliths and Microservices*



© Donald F. Ferguson, 2017. All rights reserved.

# Web Application Basic Concepts






© Donald F. Ferguson, 2017. All rights reserved.

# "Traditional" Web Application Architecture

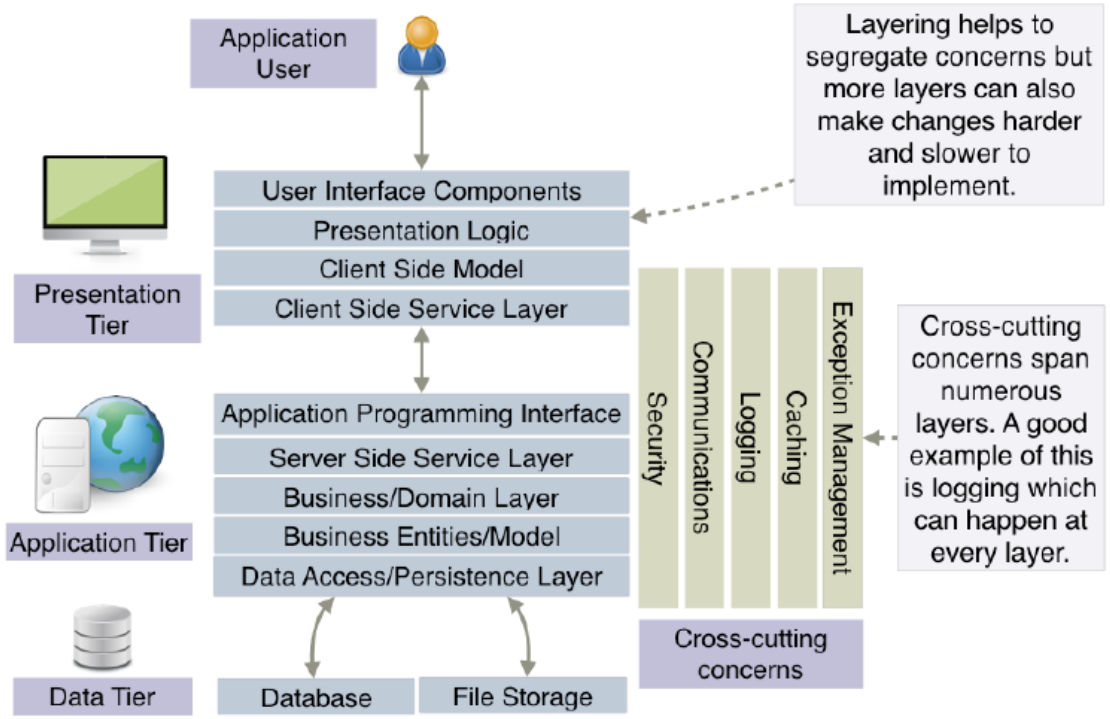
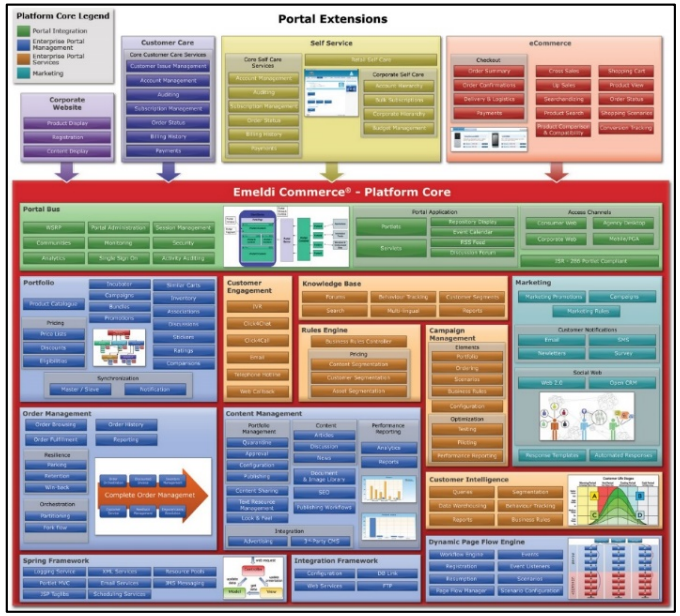


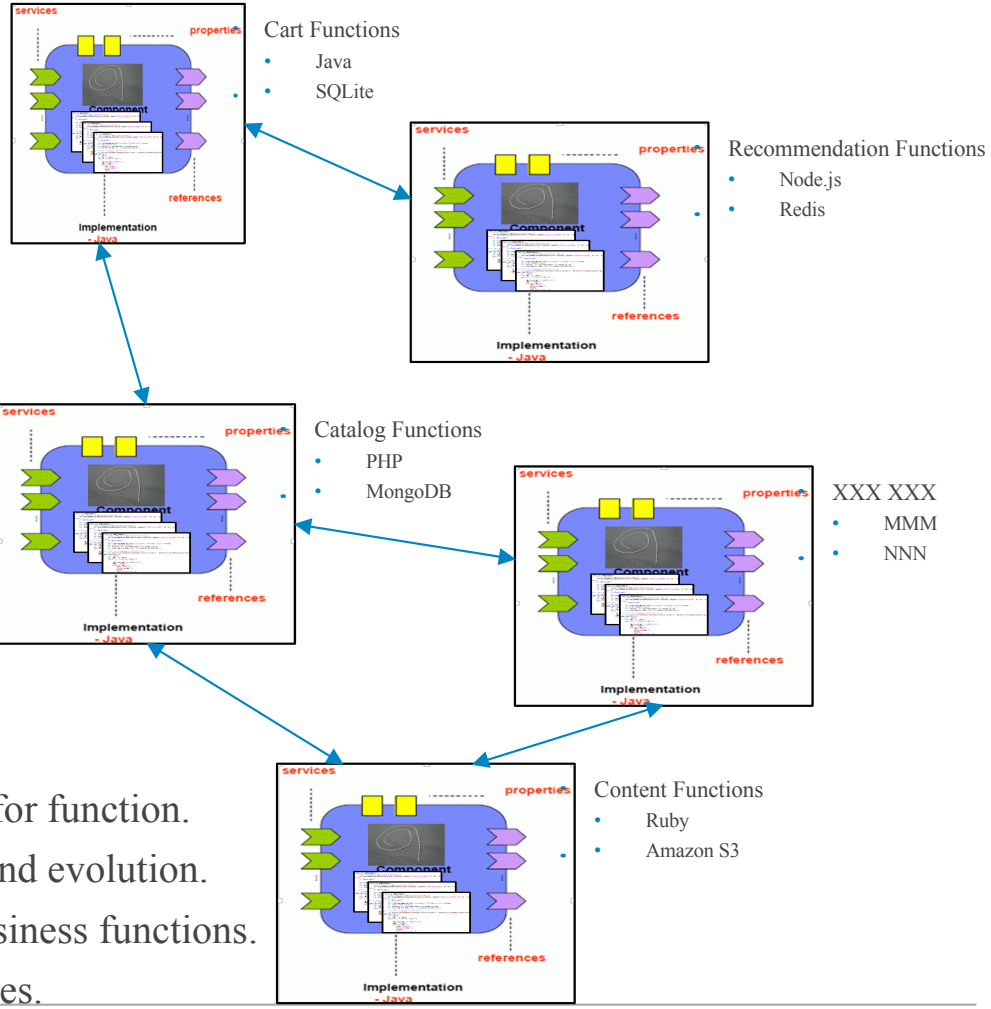

**Figure 1.2. A typical 3-tier application is usually made up of a presentation, application, and data tiers. In a tier there may be multiple layers that have specific responsibilities. A developer can choose how layers will interact with one another. This can be strictly top-down or in a loose way where layers can bypass their immediate neighbors to talk to other layers.**



© Donald F. Ferguson, 2017. All rights reserved.

# Monolithic to Micro




Motivations

- Enable best tools, languages, … for function.
- Simplifies change management and evolution.
- Better alignment of apps with business functions.
- Reuse of code and internet services.



© Donald F. Ferguson, 2017. All rights reserved.

# Micro-services Characteristics

- Componentization via Services
- Organized around Business Capabilities
- Products not Projects
- Smart endpoints and dumb pipes
- Decentralized Governance
- Decentralized Data Management
- Infrastructure Automation
- Design for failure
- Evolutionary Design



© Donald F. Ferguson, 2017. All rights reserved.

# SOA vs Microservices

| | Traditional SOA | Microservices |
|---|---|---|
| **Messaging type** | Smart, but dependency-laden ESB | Dumb, fast messaging (as with Apache Kafka) |
| **Programming style** | Imperative model | Reactive actor programming model that echoes agent-based systems |
| **Lines of code per service** | Hundreds or thousands of lines of code | 100 or fewer lines of code |
| **State** | Stateful | Stateless |
| **Messaging type** | Synchronous: wait to connect | Asynchronous: publish and subscribe |
| **Databases** | Large relational databases | NoSQL or micro-SQL databases blended with conventional databases |
| **Code type** | Procedural | Functional |
| **Means of evolution** | Each big service evolves | Each small service is immutable and can be abandoned or ignored |
| **Means of systemic change** | Modify the monolith | Create a new service |
| **Means of scaling** | Optimize the monolith | Add more powerful services and cluster by activity |
| **System-level awareness** | Less aware and event driven | More aware and event driven |

http://usblogs.pwc.com/emerging-technology/agile-coding-in-enterprise-it-code-small-and-local/



© Donald F. Ferguson, 2017. All rights reserved.

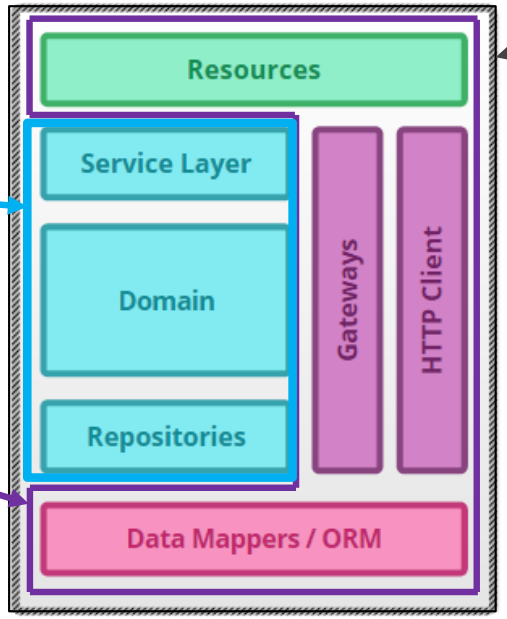
MARTIN FOWLER 4/25

# *Microservices can usually be split into similar kinds of modules*

Often, microservices display similar internal structure consisting of some or all of the displayed layers.

Inject application implementation into reusable SW containers

Reusable SW **containers** but with core technology and frameworks




Reusable infrastructure containers

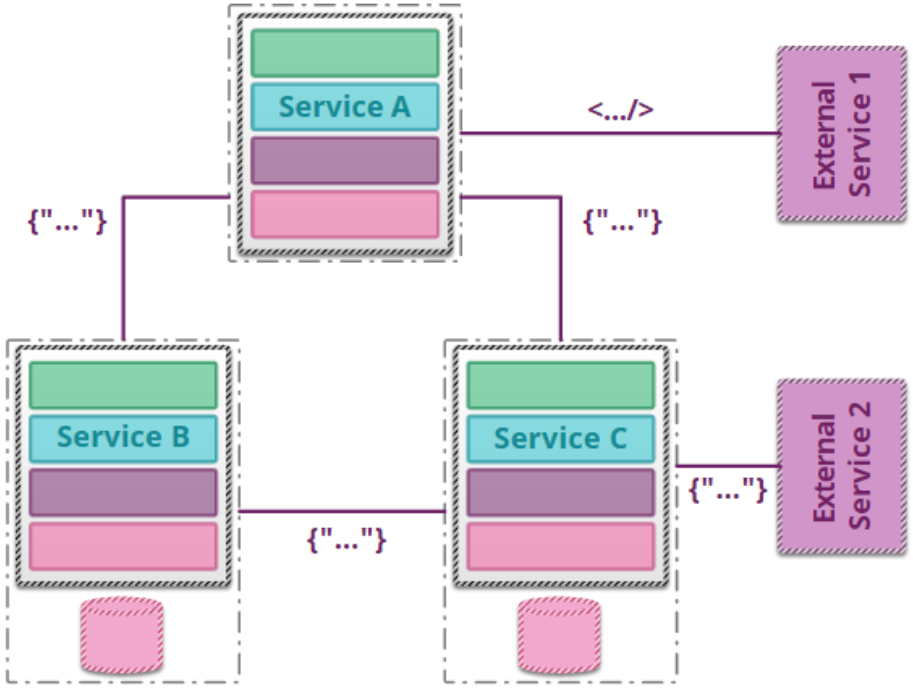
Network Boundary
Protocol
Domain
External
Persistence



© Donald F. Ferguson, 2017. All rights reserved.

MARTIN FOWLER 6/25

# *Multiple services work together as a system...*
# *...to provide business valuable features*

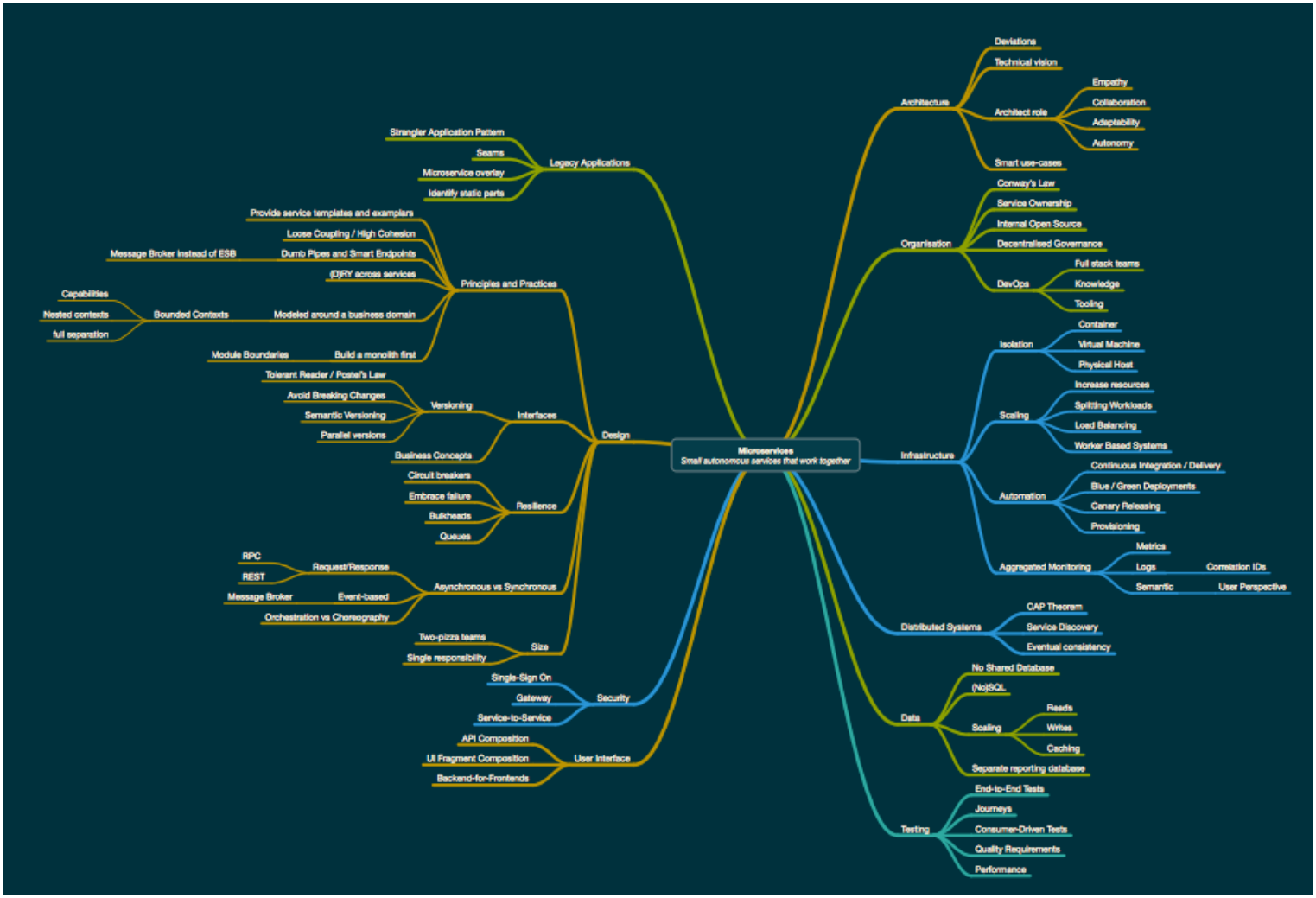



In larger systems, there are often multiple teams each with responsibility for different bounded contexts.



© Donald F. Ferguson, 2017. All rights reserved.

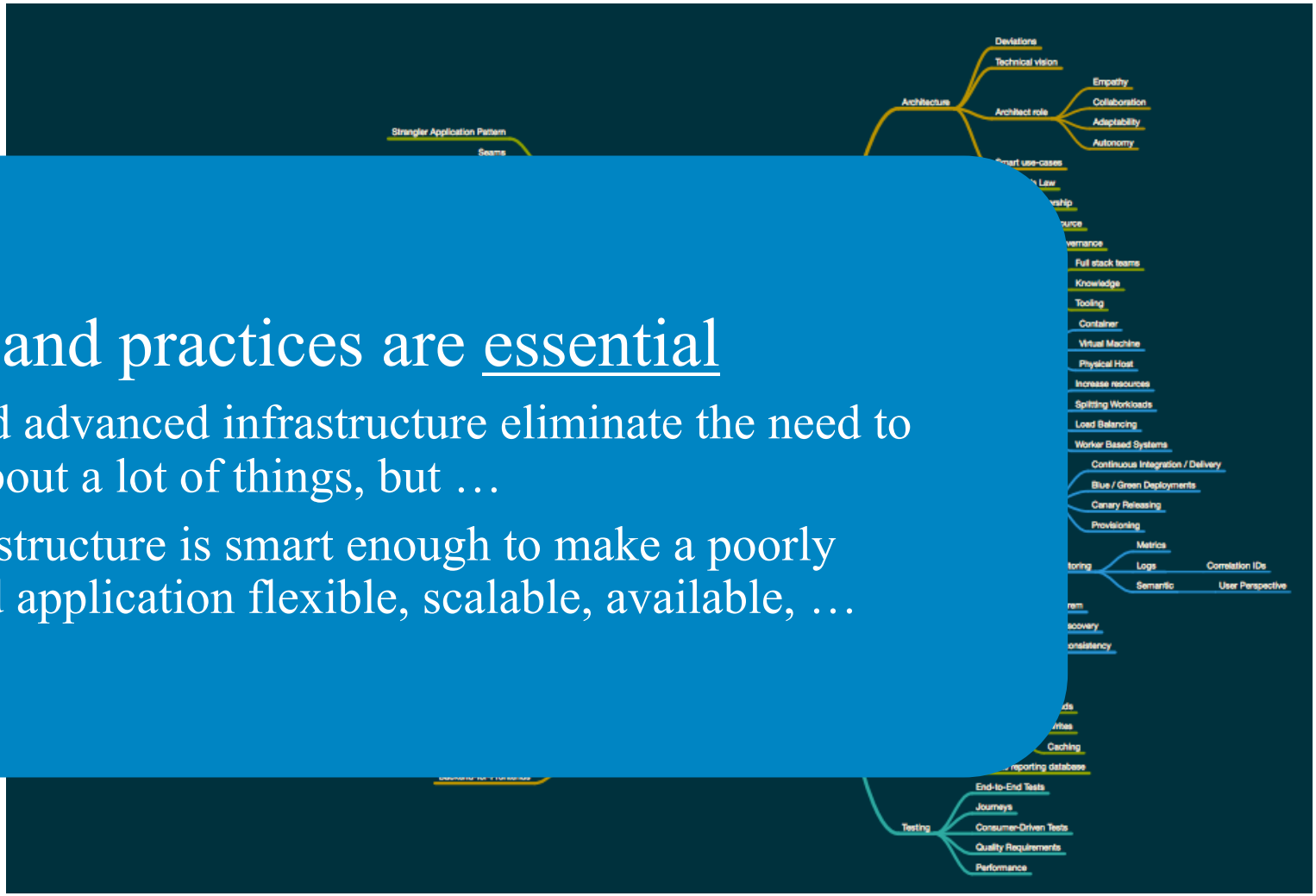
# Patterns






© Donald F. Ferguson, 2017. All rights reserved.

# Patterns




## Patterns and practices are essential

- PaaS and advanced infrastructure eliminate the need to worry about a lot of things, but …
- No infrastructure is smart enough to make a poorly designed application flexible, scalable, available, …



© Donald F. Ferguson, 2017. All rights reserved.

# Weinberg's Second Law

"If builders built buildings the way programmers wrote programs, then the first woodpecker that came along would destroy civilization."



© Donald F. Ferguson, 2017. All rights reserved.

# An Example



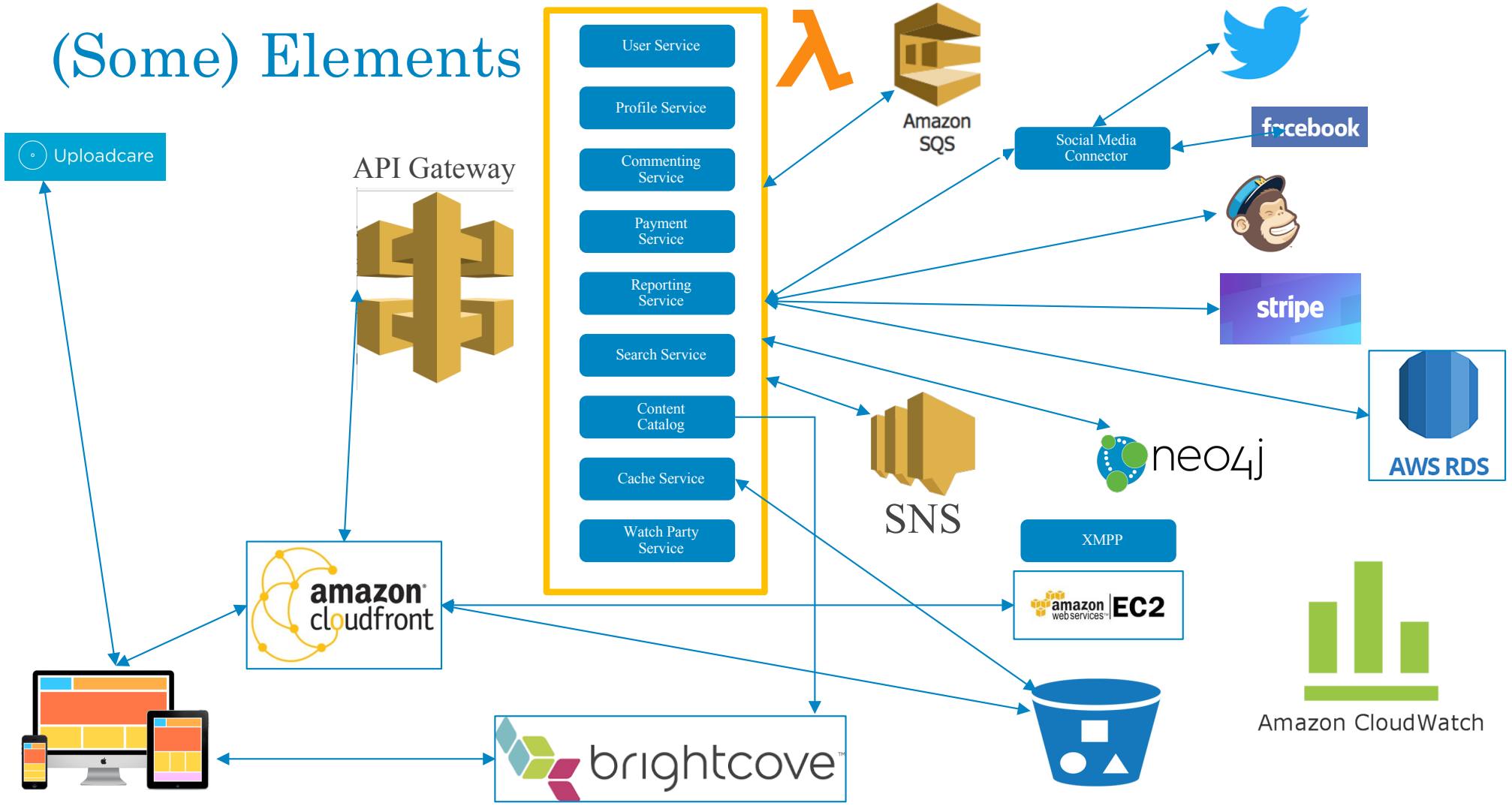
© Donald F. Ferguson, 2017. All rights reserved.

# Seeka Demo



© Donald F. Ferguson, 2017. All rights reserved.

# (Some) Elements






© Donald F. Ferguson, 2017. All rights reserved.

# 0th Project



© Donald F. Ferguson, 2017. All rights reserved.

# 0th Project

- Teams
  - Form your teams (approx. 5 people)
  - Identify contact focal point.
  - Give your team a "cool" name.
- Signup (reuse) and Amazon Web Service Account
  - Free Tier should be fine.
  - Provide access to team members.
- Create an Elastic Beanstalk instance/application.
  - Use one of the sample application.
  - Will have to make more sophisticated starting next week;
    I will use Node JS with Express for Elastic Beanstalk examples.



© Donald F. Ferguson, 2017. All rights reserved.