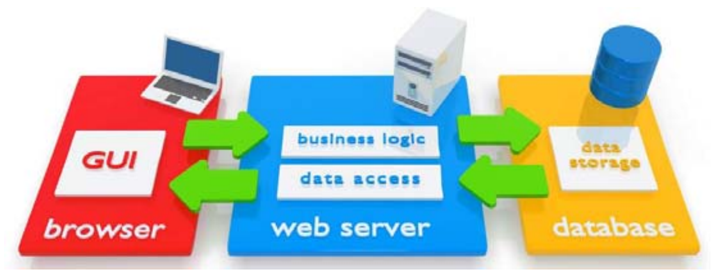
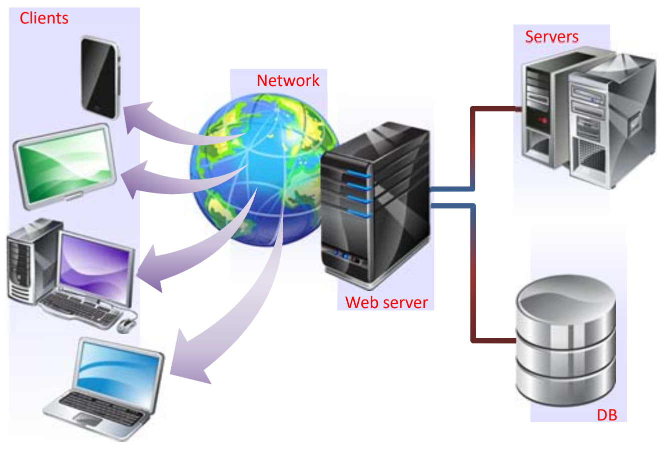
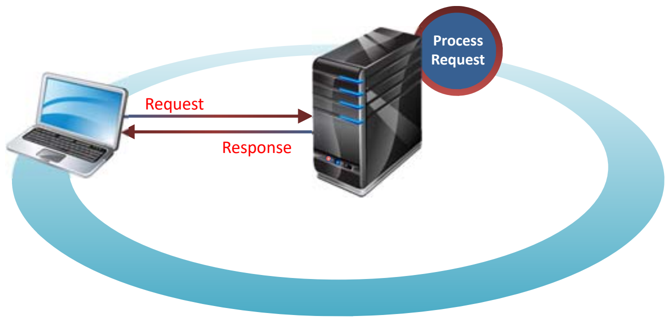


WEB APPLICATION ARCHITECTURES

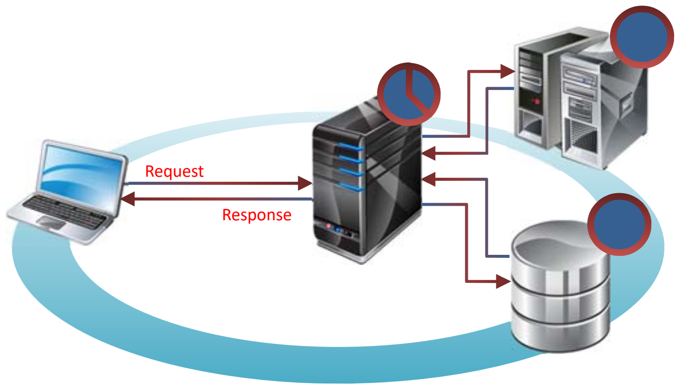
Web Architecture



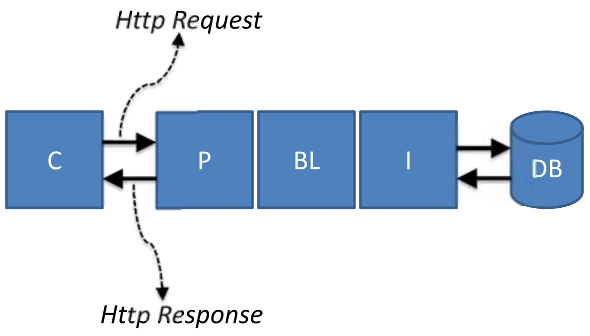
How Web Servers Work



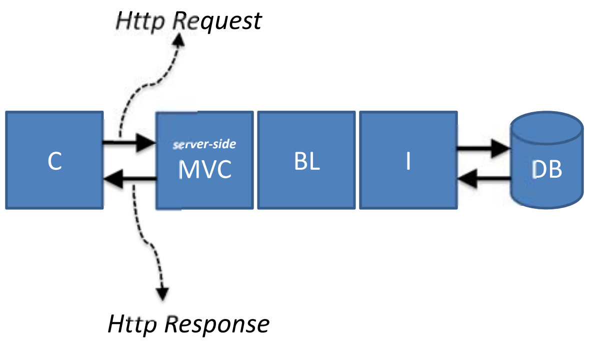
How Web Servers Work



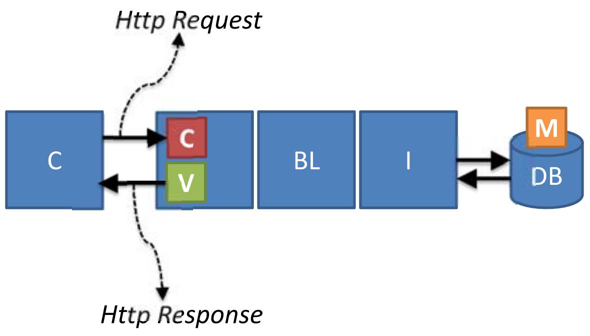
Request-Response



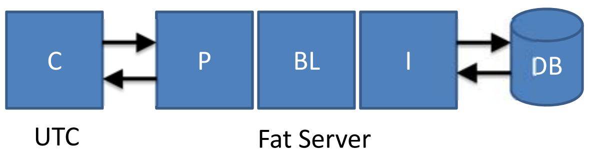
Request-Response



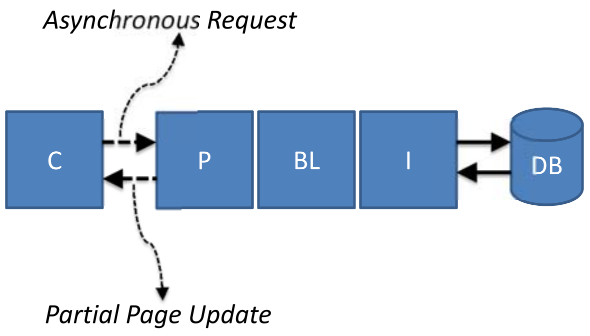
Request-Response

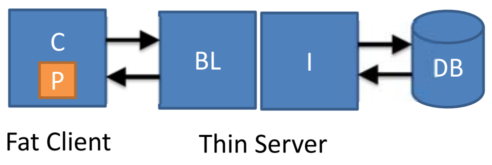


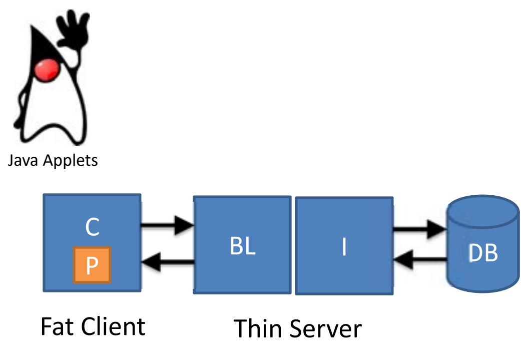
Request-Response

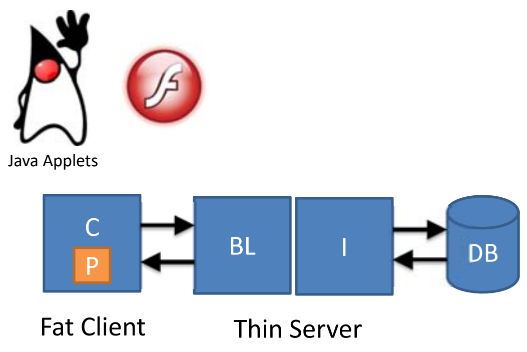


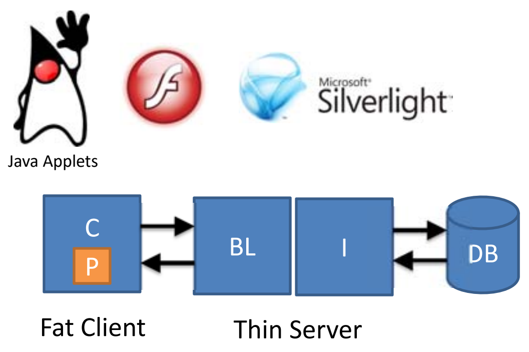
Ajax

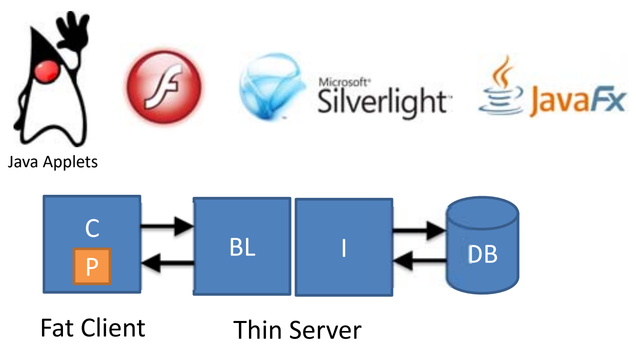

Asynchronous Request
C
P
BL
I
DB
Partial Page Update

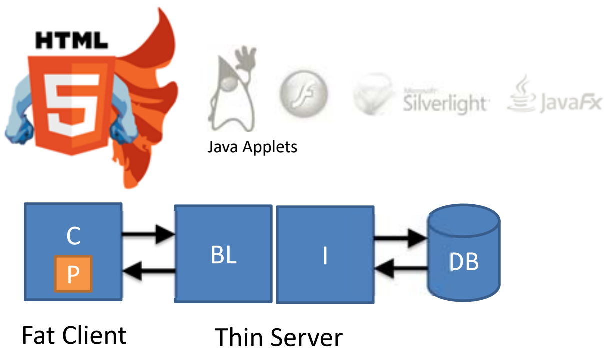

C
P
BL
I
DB
Fat Client
Thin Server


Java Applets
C
P
BL
I
DB
Fat Client
Thin Server

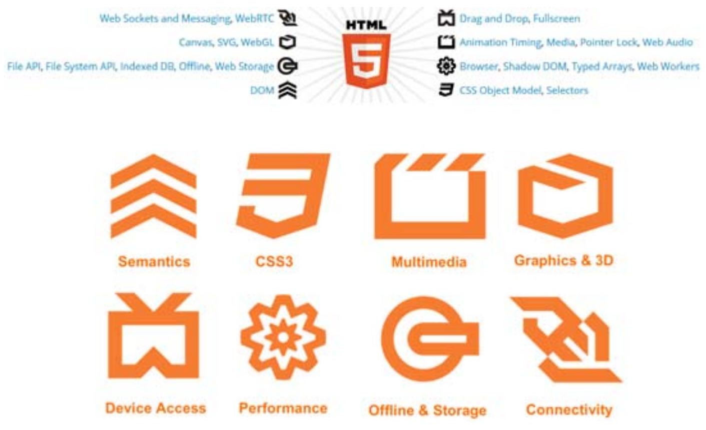

Java Applets
C
P
BL
I
DB
Fat Client
Thin Server


Java Applets
Microsoft Silverlight
C
P
BL
I
DB
Fat Client
Thin Server

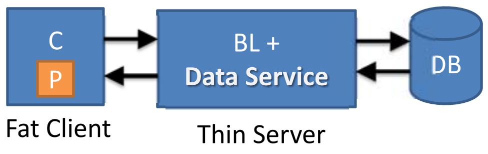

Java Applets
Microsoft Silverlight
JavaFx
C
P
BL
I
DB
Fat Client
Thin Server


HTML
5
Java Applets
Silverlight
JavaFx
C
P
BL
I
DB
Fat Client
Thin Server

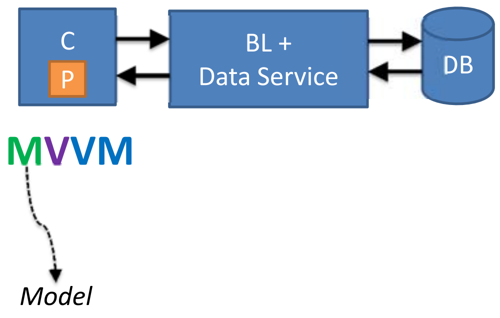
HTML5 APIs


Web Sockets and Messaging, WebRTC
Canvas, SVG, WebGL
File API, File System API, Indexed DB, Offline, Web Storage
DOM
HTML
5
Drag and Drop, Fullscreen
Animation Timing, Media, Pointer Lock, Web Audio
Browser, Shadow DOM, Typed Arrays, Web Workers
CSS Object Model, Selectors
Semantics
CSS3
Multimedia
Graphics & 3D
Device Access
Performance
Offline & Storage
Connectivity

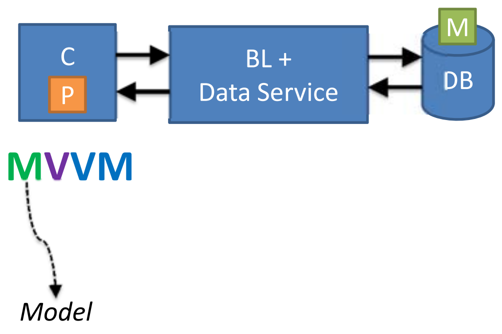
Data Service



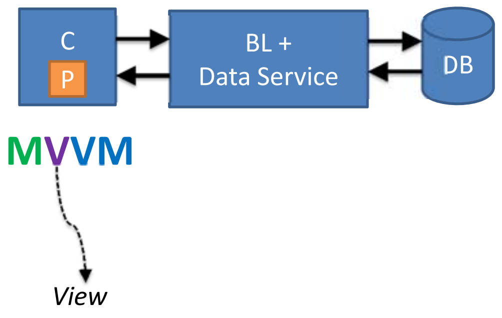
UI Logic



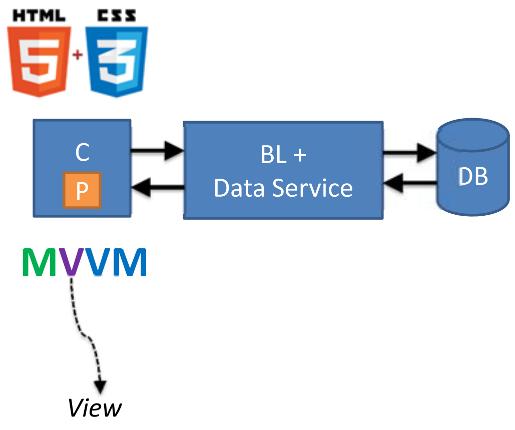
UI Logic



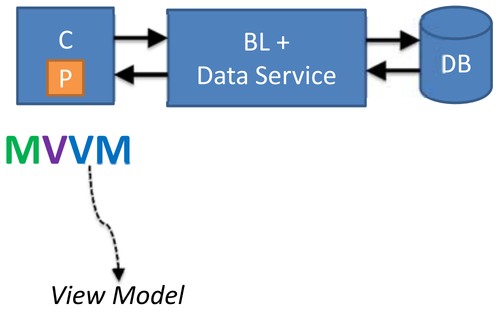
UI Logic



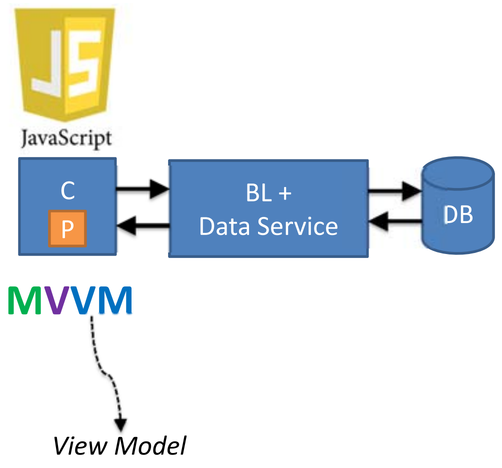
UI Logic



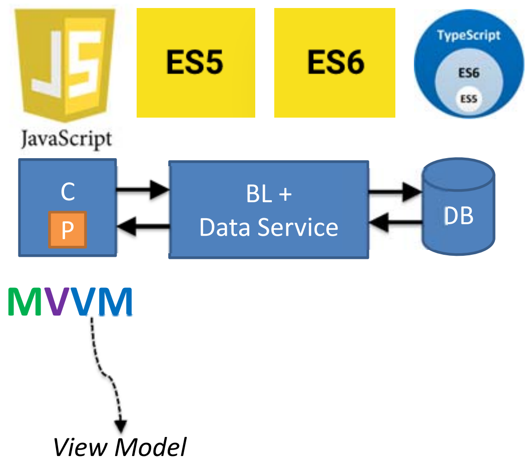
UI Logic



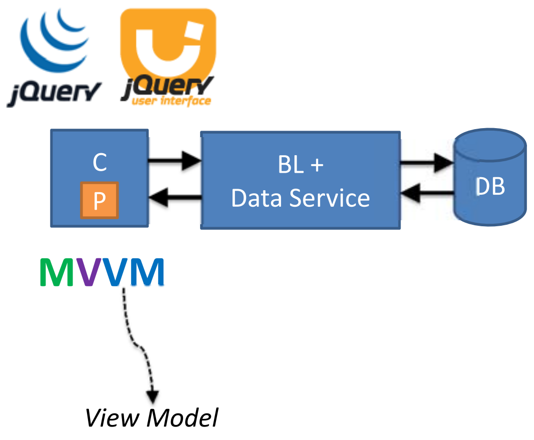
UI Logic



UI Logic

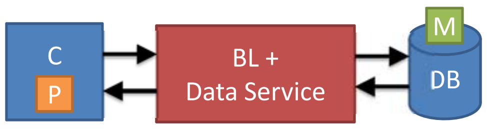


UI Logic

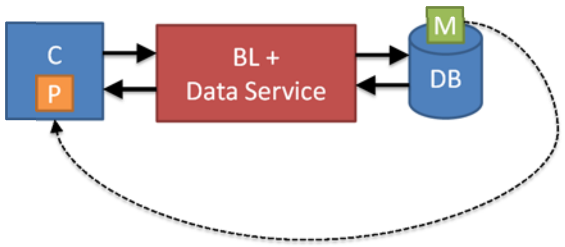

jQuery
jQuery user interface
C
P
BL +
Data Service
DB
MVVM
View Model

Knockout.
ember
METEOR
jQuery
jQuery user interface
THREE.JS
BABEL
ES6
TypeScript
Gulp
GRUNT
webpack
browserify
node JS
npm
yarn

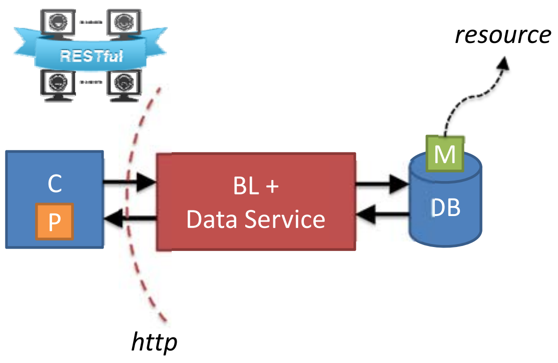
Data Service



Data Service



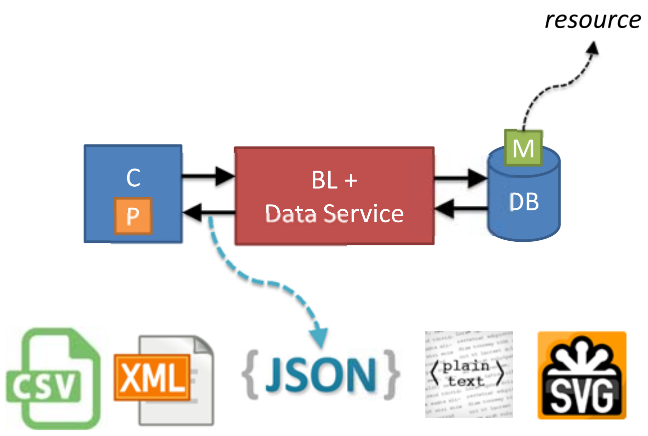
Data Service



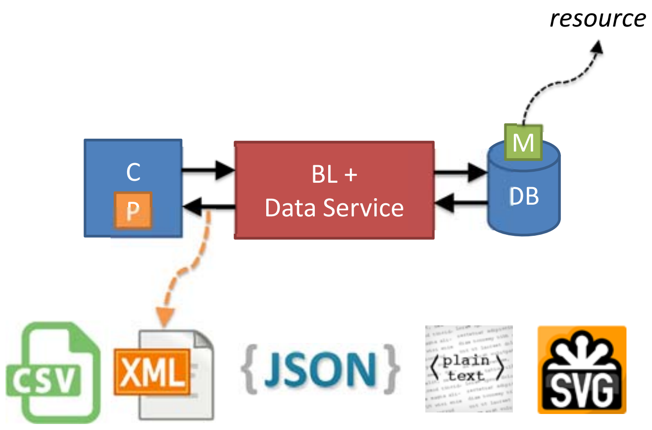
Data Service

HTTP	SQL
GET	SELECT
POST/PUT	INSERT
PUT/POST	UPDATE
DELETE	DELETE

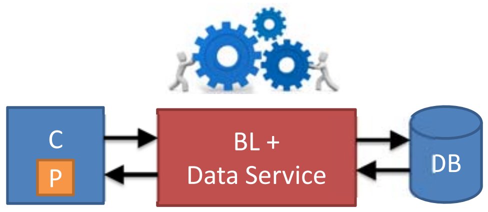
RESTful Data Service



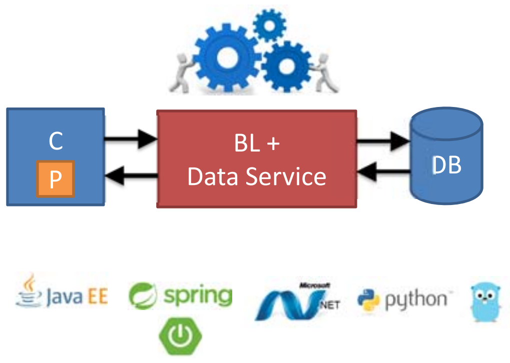
RESTful Data Service



RESTful Data Service

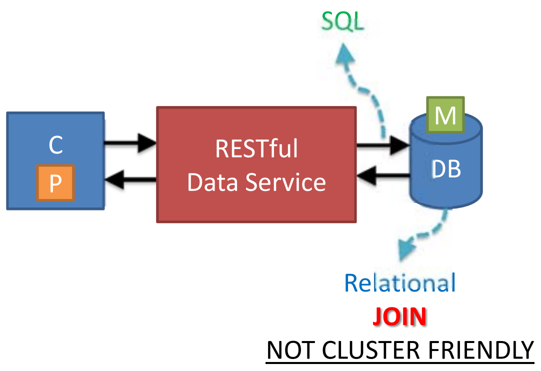


RESTful Data Service

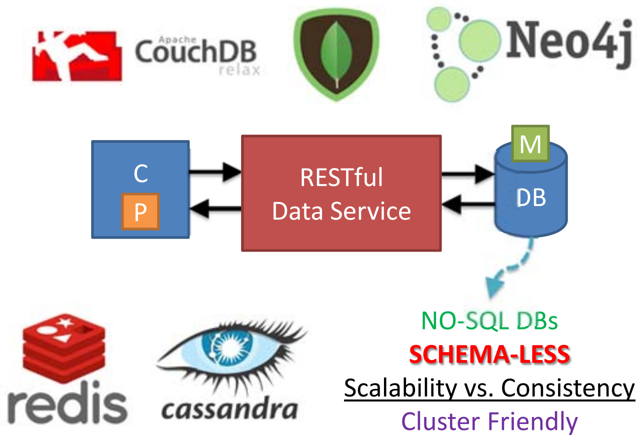


SCALABLE WEB APPLICATIONS

Data Service



Data Service



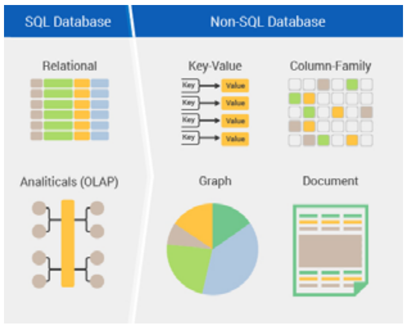
NO-SQL DBs

SCHEMA-LESS

<u>Scalability vs. Consistency</u>

Cluster Friendly

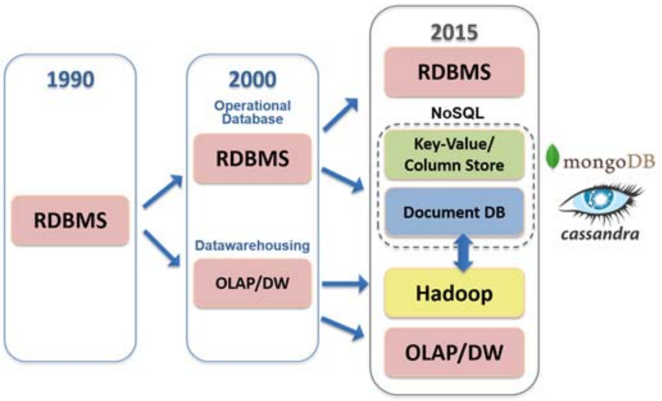
NoSQL Databases



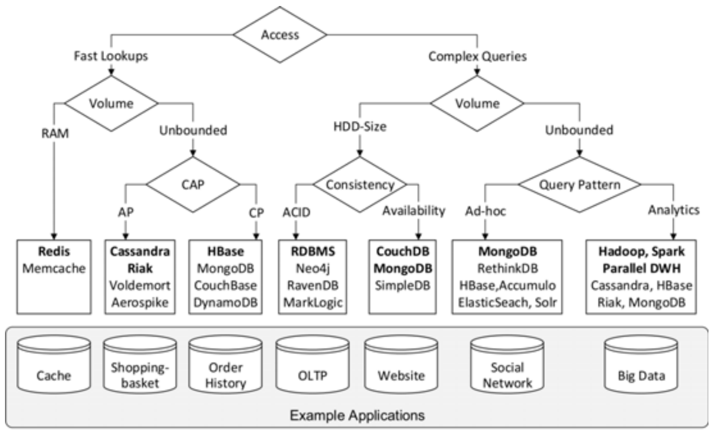
NoSQL Databases

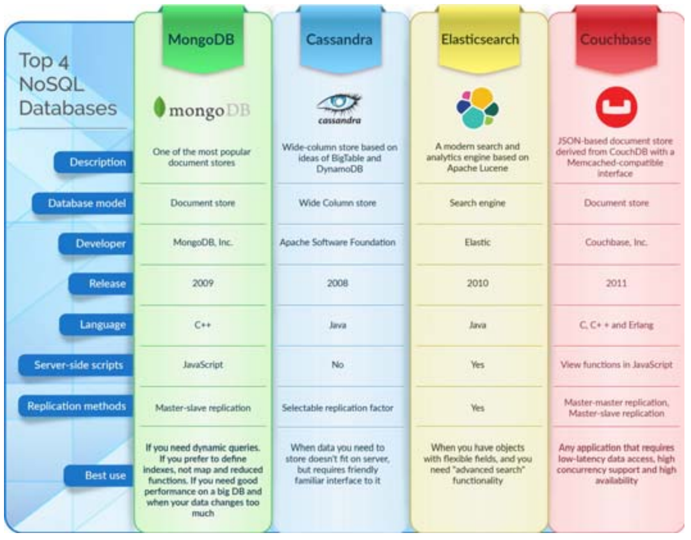
	Data Model	Query API
Cassandra	Columnfamily	Thrift
CouchDB	Document	map/reduce views
HBase	Columnfamily	Thrift, REST
MongoDB	Document	Cursor
Neo4J	Graph	Graph
Redis	Collection	Collection
Riak	Key/value	REST
Scalaris	Key/value	get/put
Tokyo Cabinet	Key/value	get/put
Voldemort	Key/value	get/put

NoSQL Databases



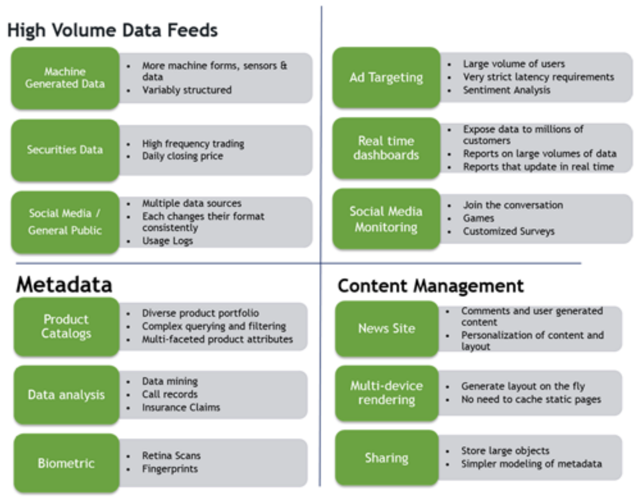
Decision Tree





Top 4 NoSQL Databases	MongoDB	Cassandra	Elasticsearch	Couchbase
Description	One of the most popular document stores	Wide-column store based on ideas of BigTable and DynamoDB	A modern search and analytics engine based on Apache Lucene	JSON-based document store derived from CouchDB with a Memcached-compatible interface
Database model	Document store	Wide Column store	Search engine	Document store
Developer	MongoDB, Inc.	Apache Software Foundation	Elastic	Couchbase, Inc.
Release	2009	2008	2010	2011
Language	C++	Java	Java	C, C+ + and Erlang
Server-side scripts	JavaScript	No	Yes	View functions in JavaScript
Replication methods	Master-slave replication	Selectable replication factor	Yes	Master-master replication, Master-slave replication
Best use	If you need dynamic queries. If you prefer to define indexes, not map and reduced functions. If you need good performance on a big DB and when your data changes too much	When data you need to store doesn't fit on server, but requires friendly familiar interface to it	When you have objects with flexible fields, and you need "advanced search" functionality	Any application that requires low-latency data access, high concurrency support and high availability

What are the Right Use Cases for NoSQL?



High Volume Data Feeds

Machine Generated Data
- More machine forms, sensors & data
- Variably structured

Securities Data
- High frequency trading
- Daily closing price

Social Media / General Public
- Multiple data sources
- Each changes their format consistently
- Usage Logs

Ad Targeting
- Large volume of users
- Very strict latency requirements
- Sentiment Analysis

Real time dashboards
- Expose data to millions of customers
- Reports on large volumes of data
- Reports that update in real time

Social Media Monitoring
- Join the conversation
- Games
- Customized Surveys

Metadata

Product Catalogs
- Diverse product portfolio
- Complex querying and filtering
- Multi-faceted product attributes

Data analysis
- Data mining
- Call records
- Insurance Claims

Biometric
- Retina Scans
- Fingerprints

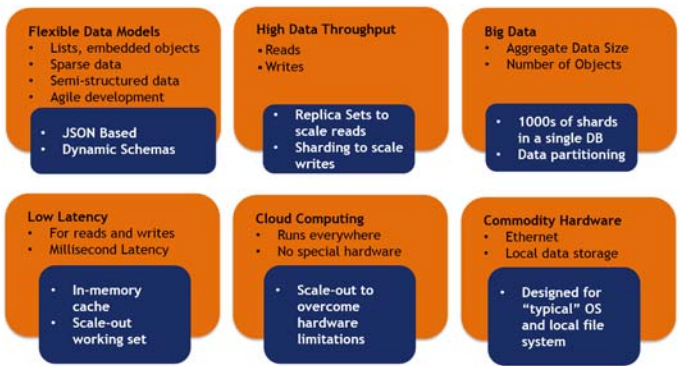
Content Management

News Site
- Comments and user generated content
- Personalization of content and layout

Multi-device rendering
- Generate layout on the fly
- No need to cache static pages

Sharing
- Store large objects
- Simpler modeling of metadata

NoSQL Database Features



Flexible Data Models
- Lists, embedded objects
- Sparse data
- Semi-structured data
- Agile development

- JSON Based
- Dynamic Schemas

High Data Throughput
- Reads
- Writes

- Replica Sets to scale reads
- Sharding to scale writes

Big Data
- Aggregate Data Size
- Number of Objects

- 1000s of shards in a single DB
- Data partitioning

Low Latency
- For reads and writes
- Millisecond Latency

- In-memory cache
- Scale-out working set

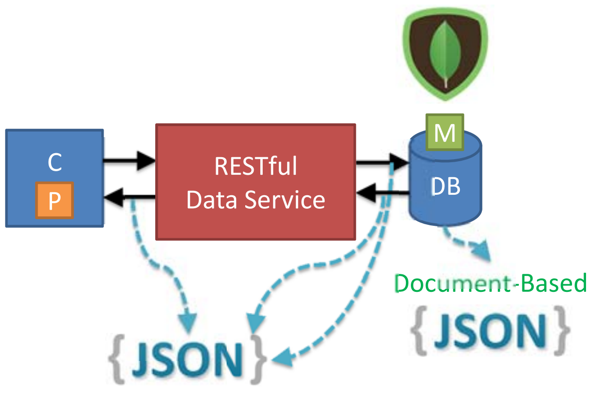
Cloud Computing
- Runs everywhere
- No special hardware

- Scale-out to overcome hardware limitations

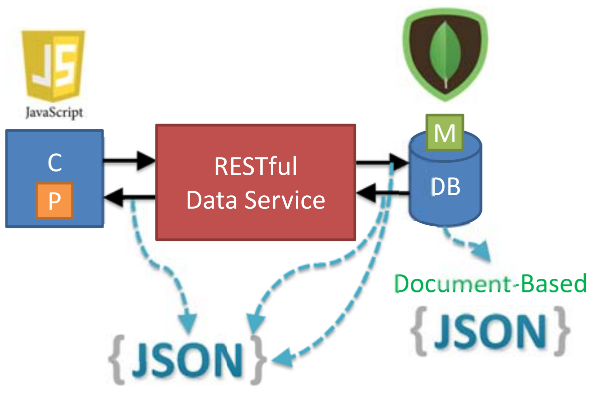
Commodity Hardware
- Ethernet
- Local data storage

- Designed for “typical” OS and local file system

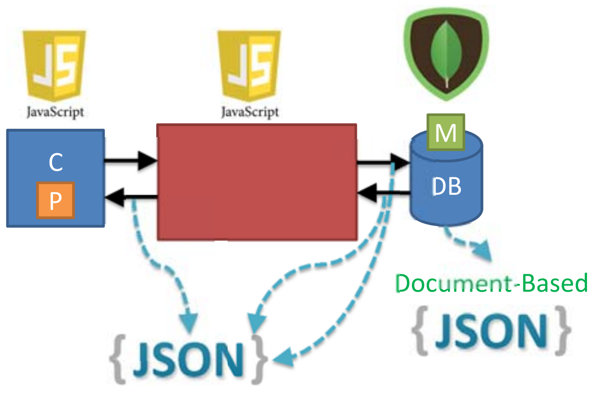
Data Service



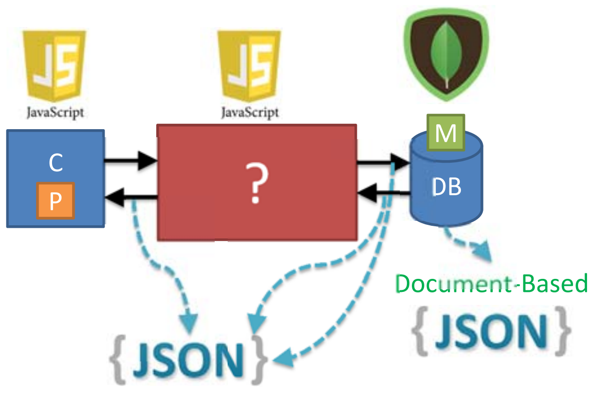
Data Service



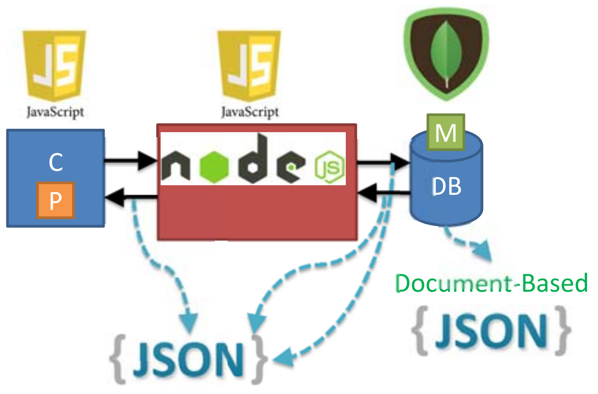
Data Service



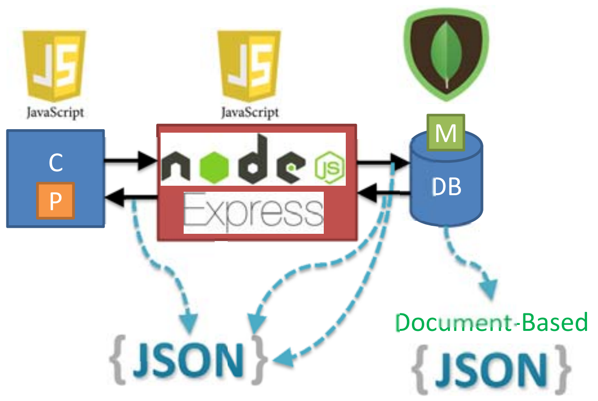
Data Service



Data Service

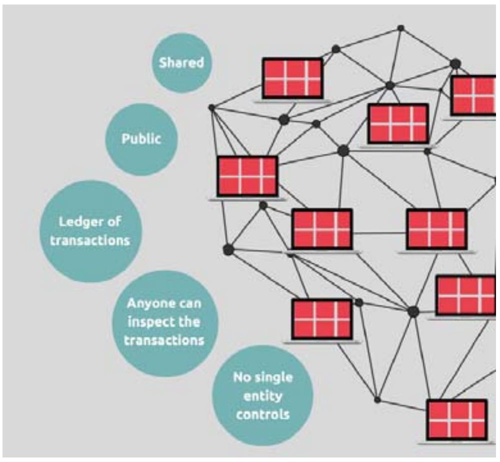


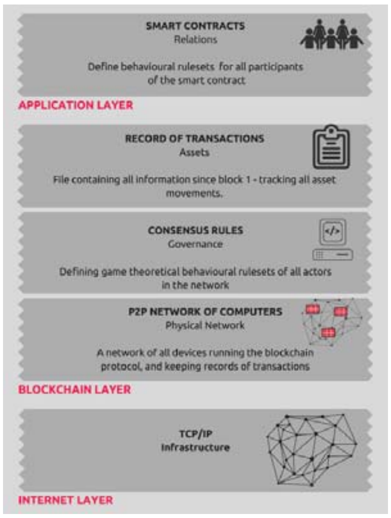
Data Service



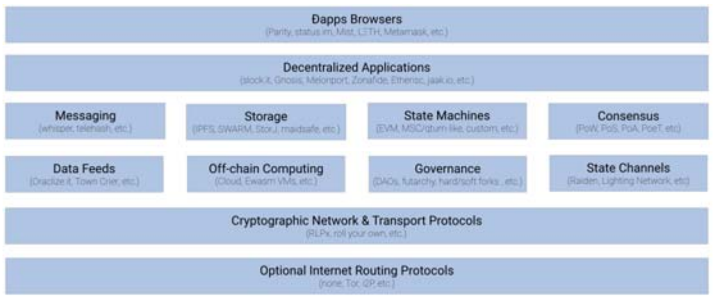
JS IS EVERYWHERE

Distributed Ledger

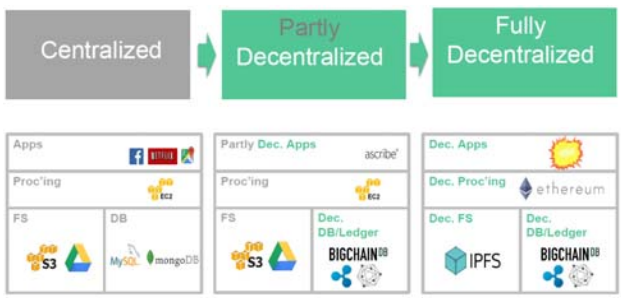

Shared
Public
Ledger of transactions
Anyone can inspect the transactions
No single entity controls


SMART CONTRACTS
Relations
Define behavioural rulesets for all participants of the smart contract
APPLICATION LAYER
RECORD OF TRANSACTIONS
Assets
File containing all information since block 1 - tracking all asset movements.
CONSENSUS RULES
Governance
Defining game theoretical behavioural rulesets of all actors in the network
P2P NETWORK OF COMPUTERS
Physical Network
A network of all devices running the blockchain protocol, and keeping records of transactions
BLOCKCHAIN LAYER
TCP/IP
Infrastructure
INTERNET LAYER

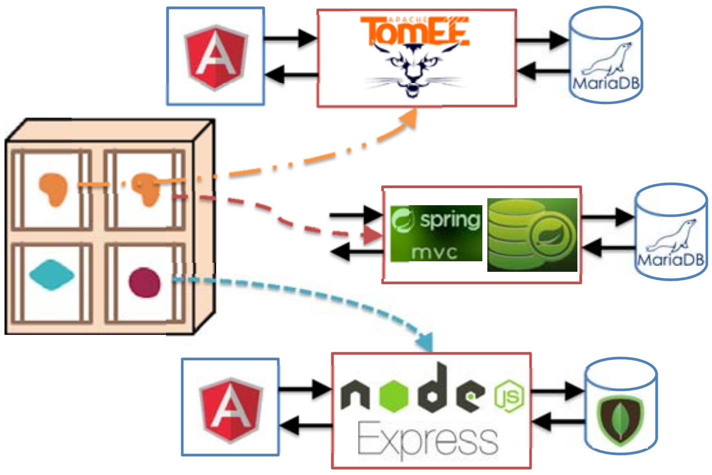
Web 3.0 Abstracted Stack


Đapps Browsers
(Parity, status.im, Mist, LETH, Metamask, etc.)
Decentralized Applications
(slock.it, Gnosis, Melonport, Zonafide, Etherisc, jaak.io, etc.)
Messaging
(whisper, telehash, etc.)
Storage
(IPFS, SWARM, StorJ, maidsafe, etc.)
State Machines
(EVM, MSC/qtum like, custom, etc.)
Consensus
(PoW, PoS, PoA, PoeT, etc)
Data Feeds
(Oraclize it, Town Crier, etc.)
Off-chain Computing
(Cloud, Ewasm VMs, etc.)
Governance
(DAOs, futarchy, hard/soft forks , etc.)
State Channels
(Raiden, Lighting Network, etc)
Cryptographic Network & Transport Protocols
(RLPx, roll your own, etc.)
Optional Internet Routing Protocols
(none, Tor, I2P, etc.)

Decentralization of the Cloud



Hybrid Architectures



TRENDS
IN
SOFTWARE DEVELOPMENT

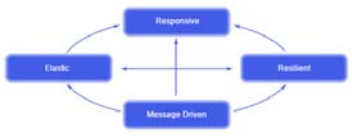
Trends in software development

Platform as a Service

Autonomous teams

Continuous Delivery

Agile Organization



Reactive manifesto

Trends in software development

> Backend
  – Micro-service Architecture
  – Reactive System Architecture
> Frontend
  – Reactive Programming
  – Progressive Web Application

Definitions of microservices

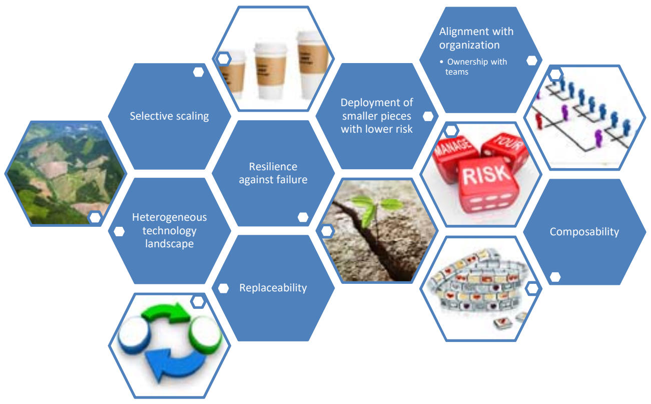
> Small and focused on doing one thing well
> Autonomous

"*Loosely coupled service oriented architecture with bounded contexts*"
Adrian Cockcroft (Netflix)

"*SOA done right*"
Anonymous

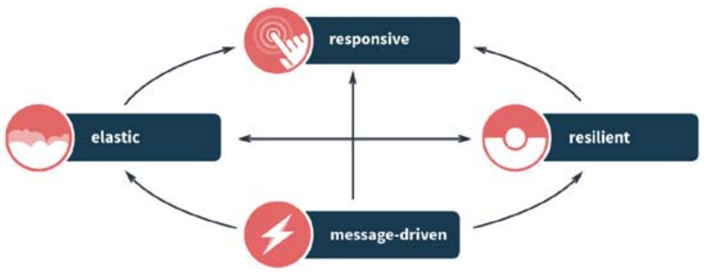
"… services are independently deployable and scalable, each service also provides a firm module boundary, even allowing for different services to be written in different programming languages."
Martin Fowler (Thoughtworks)

Benefits


Selective scaling
Alignment with organization
• Ownership with teams
Deployment of smaller pieces with lower risk
Resilience against failure
Heterogeneous technology landscape
Replaceability
Composability
MANAGE YOUR RISK

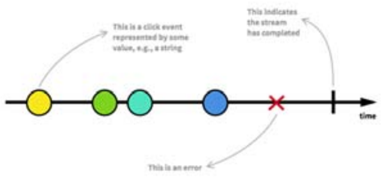
REACTIVE SYSTEMS AND PROGRAMMING

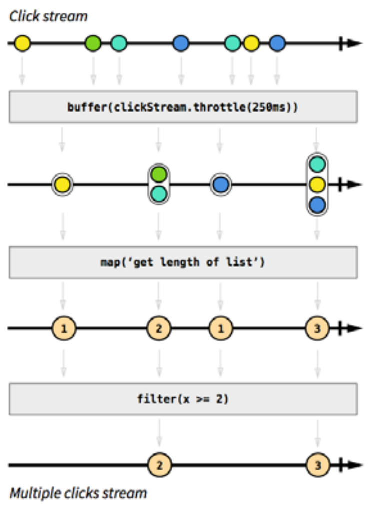
Principles of Reactive Systems



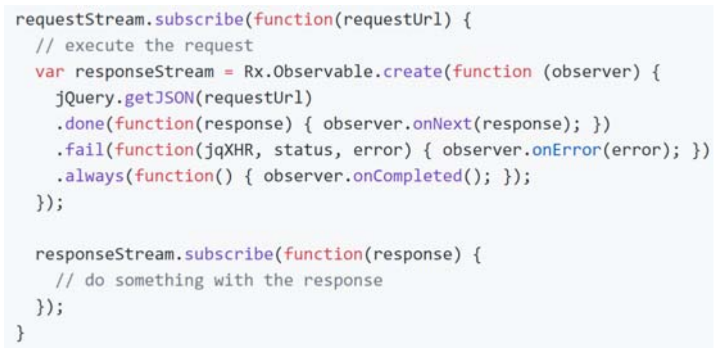
RX Programming

> Reactive programming is programming with asynchronous data streams.
> Functions to combine, create and filter of streams
> A stream is a sequence of **ongoing events ordered in time**.
  – Emit three different things
    • a value (of some type)
    • an error
    • a "completed" signal.


This is a click event represented by some value, e.g., a string
This indicates the stream has completed
time
This is an error


Click stream
buffer(clickStream.throttle(250ms))
map('get length of list')
1
2
1
3
filter(x >= 2)
2
3
Multiple clicks stream

Request and response



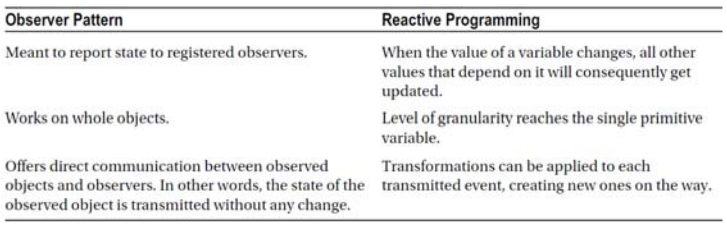
```
requestStream.subscribe(function(requestUrl) {
  // execute the request
  var responseStream = Rx.Observable.create(function (observer) {
    jQuery.getJSON(requestUrl)
    .done(function(response) { observer.onNext(response); })
    .fail(function(jqXHR, status, error) { observer.onError(error); })
    .always(function() { observer.onCompleted(); });
  });

  responseStream.subscribe(function(response) {
    // do something with the response
  });
}
```

Promise is an Observable?

- Yes
- Observable is Promise++
- Easily convert a Promise to an Observable

```
Rx.Observable.fromPromise(promise)
```

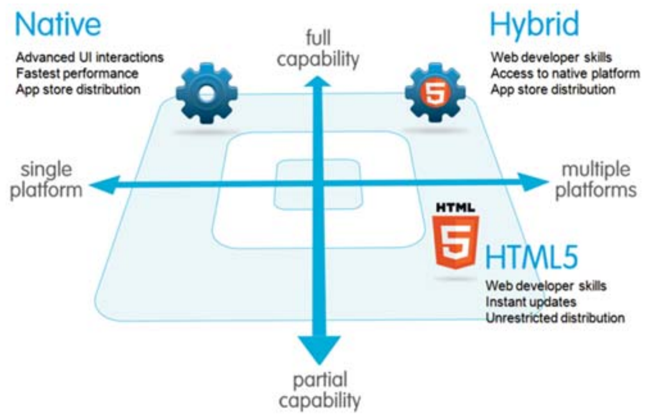
Observer Pattern *vs.* RP Paradigm



Observer Pattern	Reactive Programming
Meant to report state to registered observers.	When the value of a variable changes, all other values that depend on it will consequently get updated.
Works on whole objects.	Level of granularity reaches the single primitive variable.
Offers direct communication between observed objects and observers. In other words, the state of the observed object is transmitted without any change.	Transformations can be applied to each transmitted event, creating new ones on the way.

MOBILE APPLICATIONS

Native vs Hybrid



Native

- SDK-s evolve all the time
- Languages/tooling is different on every platform
- iOS – Swift/Objective-C
- Android – Java (C++ for low level)
- Windows Phone – MS languages - mostly C#

HTML5 Hybrid Apps

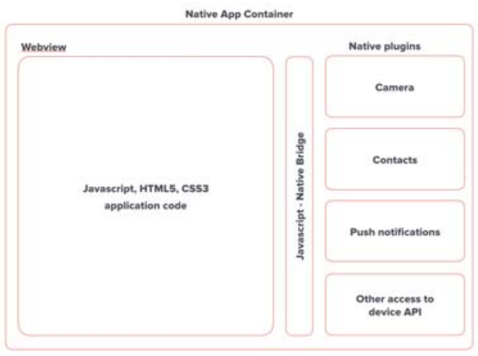
- Javascript/HTML5/CSS code
- Internal Web browser (webview)
- Inside native app frame
- All the code in Javascript, browser interprets and renders into UI
- Problems with performance, lots of devices don’t reach 60fps
- Usability problems – apps often don’t feel native to the specific platform
- Platform web engines are fragmented!

HTML5 Hybrid Apps

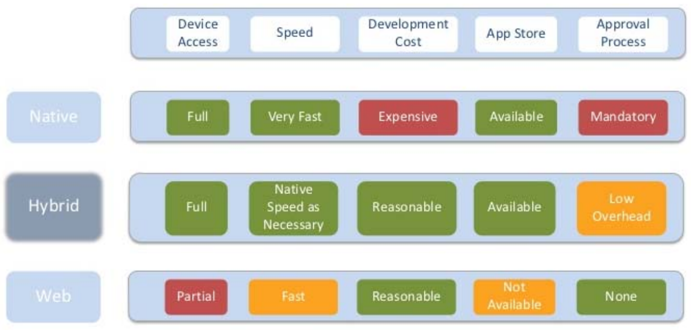
> Simplistic architecture

> Cordova/Phonegap

> Ionic (AngularJs)


Native App Container
Webview
Javascript, HTML5, CSS3 application code
Javascript - Native Bridge
Native plugins
Camera
Contacts
Push notifications
Other access to device API

Application Development Comparison

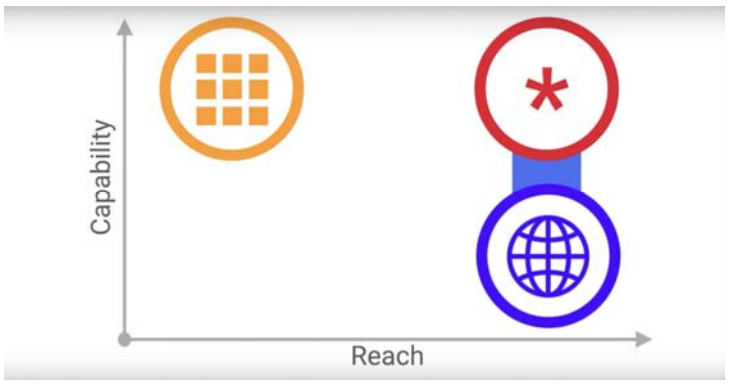


	Device Access	Speed	Development Cost	App Store	Approval Process
Native	Full	Very Fast	Expensive	Available	Mandatory
Hybrid	Full	Native Speed as Necessary	Reasonable	Available	Low Overhead
Web	Partial	Fast	Reasonable	Not Available	None

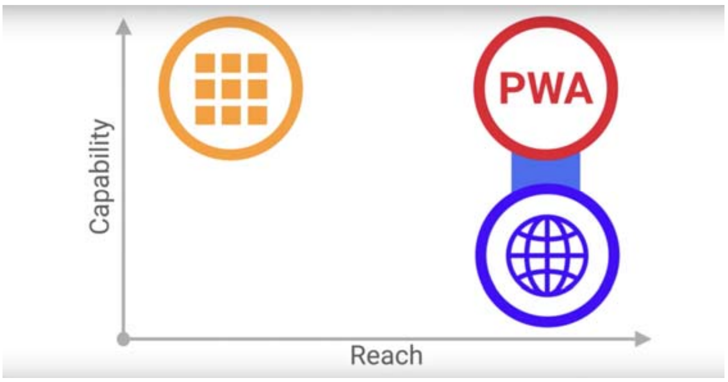
Progressive Web App

Capability

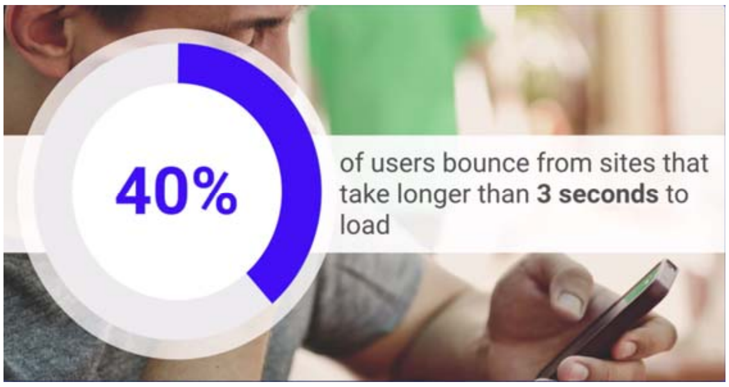
Progressive Web App



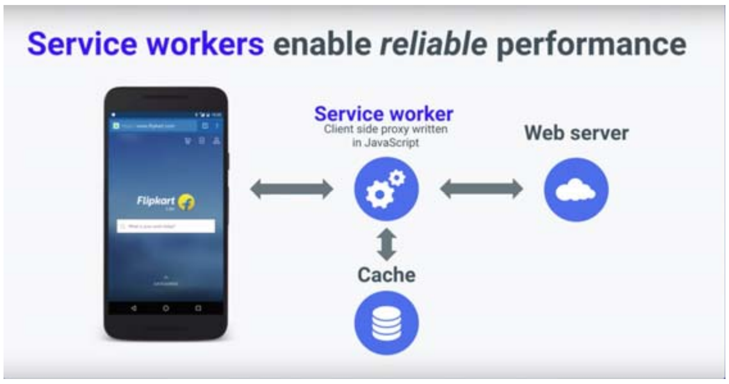
Progressive Web App



Progressive Web App

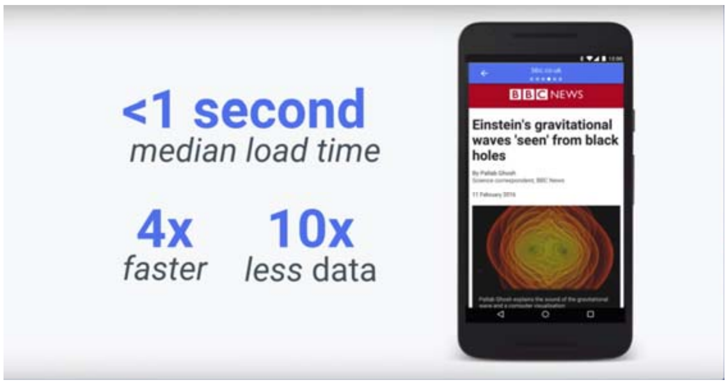


Progressive Web App



Service workers enable *reliable* performance

Progressive Web App



<1 second
median load time

4x
faster

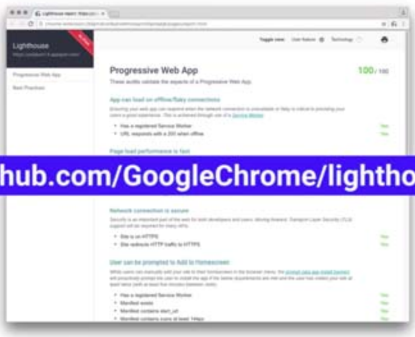
10x
less data

BBC NEWS
Einstein's gravitational waves 'seen' from black holes

Progressive Web App

`<amp-install-serviceworker>`

Progressive Web App


Lighthouse

Progressive Web App

Best Practices

Progressive Web App 100 / 100

These audits validate the aspects of a Progressive Web App.

App can load on offline/flaky connections

- Has a registered Service Worker Yes
- URL responds with a 200 when offline Yes

Page load performance is fast

Network connection is secure

- Site is on HTTPS Yes
- Site redirects HTTP traffic to HTTPS Yes

User can be prompted to Add to Homescreen

- Has a registered Service Worker Yes
- Manifest exists Yes
- Manifest contains start_url Yes
- Manifest contains icons at least 144px Yes

github.com/GoogleChrome/lighthouse

WEB ASSEMBLY

http://webassembly.org

Web Assembly

> WebAssembly or *wasm* is a new portable, size- and load-time-efficient format suitable for compilation to the web.

> An open standard by a W3C Community Group

> Includes representatives from all major browsers.

Web Assembly

> WebAssembly or *wasm* is a new portable, size- and load-time-efficient format suitable for compilation to the web.

> An open standard by a W3C Community Group

> Includes representatives from all major browsers.

A virtual machine for the web!

Web Assembly

> Exposes internal browser VM

> Cross-browser support

> JS bindings

Why?

> Performance
> Language freedom

Ready?

> Almost!

Efficient and fast

> The wasm stack machine is designed to be encoded in a size- and load-time-efficient binary format.

> WebAssembly aims to execute at native speed by taking advantage of common hardware capabilities available on a wide range of platforms.

Safe

> Describes a memory-safe, sandboxed execution environment

> Could be implemented inside existing JavaScript virtual machines.

> Enforce the same-origin and permissions security policies of the browser.

Open and debuggable

> Designed to be pretty-printed in a textual format for debugging, testing, experimenting, optimizing, learning, teaching, and writing programs by hand.

> The textual format will be used when viewing the source of wasm modules on the web.

Part of the open web platform

> Designed to maintain the version-less, feature-tested, and backwards-compatible nature of the web.

> WebAssembly modules will be able to call into and out of the JavaScript context and access browser functionality through the same Web APIs accessible from JavaScript.

> Supports non-web embeddings.

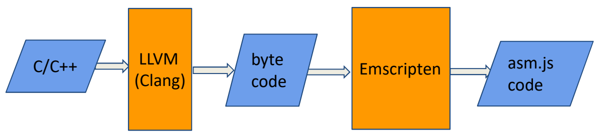
What can I do now?

> Execute WASM

> Need compiler to emit WASM

> LLVM

> C, C++, Rust


C/C++
LLVM (Clang)
byte code
Emscripten
asm.js code