

DEVELOPERS AND ARCHITECTS

STRATEGIES 2018

Oliver Sturm • @olivers • oliver@oliversturm.com



OLIVER STURM

- Training Director at DevExpress
- Consultant, trainer, author, software architect and developer for over 25 years
- Contact: oliver@oliversturm.com

AGENDA

Idea: Talk about technology

- Application building blocks
- Services
- Microservices
- Data persistence
- User Interfaces
- Programming Languages
- Mobile
- Cloud
- Open Source

APPLICATION BUILDING BLOCKS

- What is an "application" made of?
- Terminology check:
 - Client application
 - Server application
 - Web application
 - Application system
 - Enterprise application

SERVICES

- Part of most architectural concepts
- SOA?
- Web Services
- "Real-time web?" SignalR? socket.io?

SERVICES — SOA

Remember the four tenets Don Box got excited about?

- Boundaries are explicit
- Services are autonomous
- Services share schema and contract, not class
- Service compatibility is determined based on policy

SOA *resulted* in a very formal understanding of service architecture, which is fortunately not shared by too many architects today.

WEB SERVICES

- ASMX — WSE — WCF — WSDL — SOAP — Microsoft's world of enormous complexity intended to solve a very simple problem
- RESTful services: the most complicated part is the name
 - URLs and HTTP methods
 - JSON, XML and possibly other data formats, using content negotiation

SERVICES — REAL-TIME WEB

- WebSockets and their various ancestors
- Bi-directional communication

Reasoning for real-time web techniques:

MICROSERVICES

How big is a microservice? It depends.

- Do one "thing" well. What's a "thing"? It depends.
- Two-pizza team
- Throwawayable
- Focus on boundaries and business context, not on lines of code

MICROSERVICES — COMMUNICATION

- Direct communication between services
- Message Queues
- Service Bus (ESB)

MICROSERVICES — COMPOSITION

- Manual composition? Painful.
- Docker containers
- docker-compose
- Cloud container services (ecs-cli, Azure Docker VM extension)
 - Also support composition

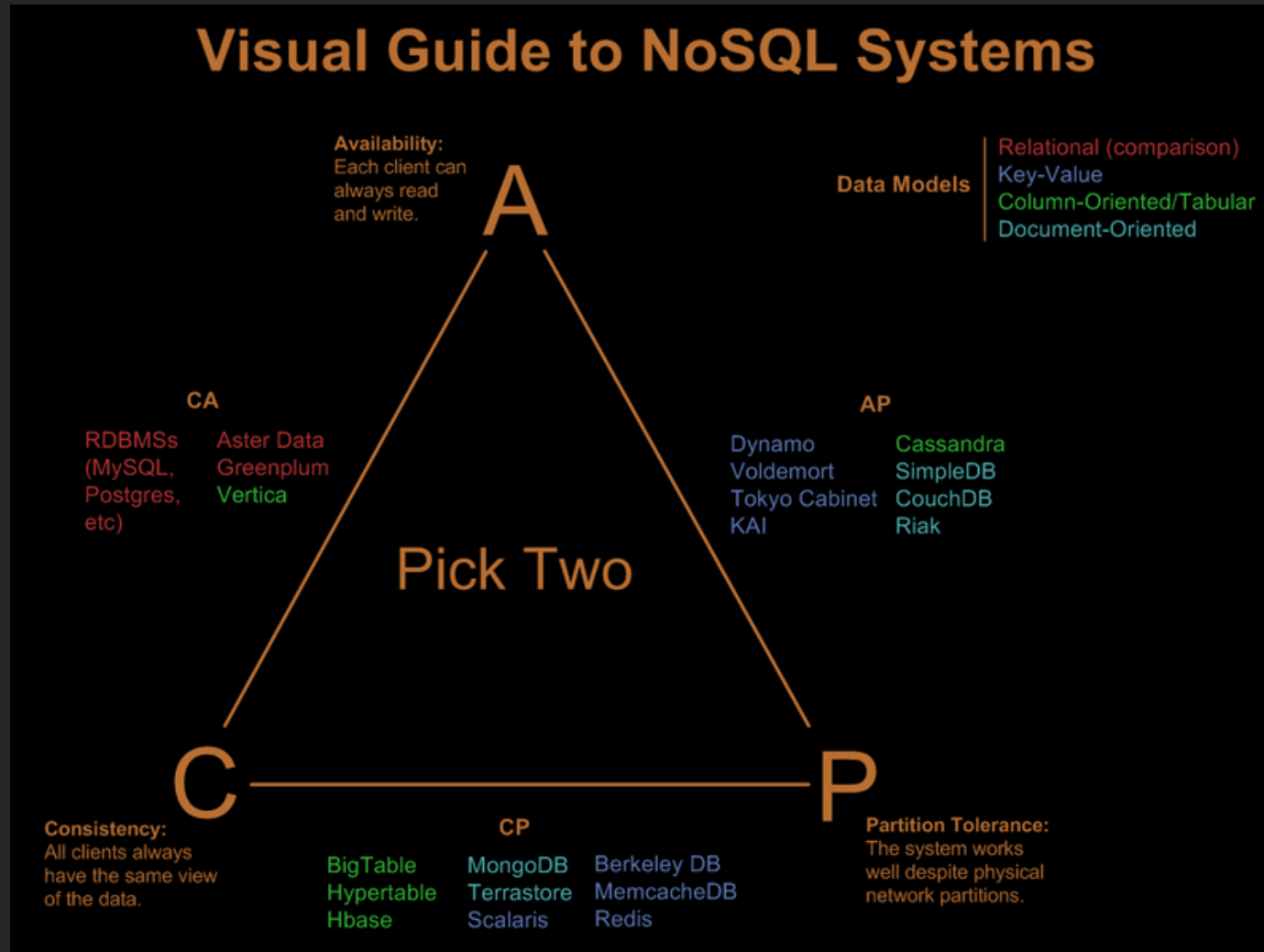
MICROSERVICES — SERVERLESS

- Function level composition: AWS Lambda, Azure Functions, Google Cloud Functions, ...
 - Integration with cloud infrastructure for triggering and output generation
- Event driven, scalable systems with minimal infrastructure management requirements
- Pay as you go
- Lock-in effect
- Debugging, Testing...

DATA PERSISTENCE

- Relational databases
- NoSQL options
 - Key/value and column family stores
 - Document
 - Data analytics (e.g. MapReduce)

DATA PERSISTENCE — NoSQL



DATA PERSISTENCE — NoSQL

Visual Guide to NoSQL Systems

THANKS

... to Nathan Hurst <nathan@developersforgood.org> for the only image in this presentation, used with permission.

<http://blog.nahurst.com/visual-guide-to-nosql-systems>



DATA PERSISTENCE — ORM

- Choice of frameworks
- Top Down or Bottom Up?
- DB Independence

Reasoning:

DATA PERSISTENCE — CQRS

Command/Query Responsibility Segregation

- Separate query and command models
- Conflicts with ORM?
- Event Sourcing
 - Eventual consistency

User Interfaces

- Platforms
 - Native: WinForms, XAML
 - HTML
 - Electron

Reasoning for native UI platforms:

UI APPLICATION PATTERNS

- MVVM
- Flux

HTML UI — WHERE TO RENDER

- Traditional web-server based rendering?

Reasoning:

PROGRAMMING LANGUAGES

- .NET: C#, VB.NET, F#, others?
- JavaScript: Native, TypeScript, CoffeeScript, LiveScript, others?

MOBILE

- Mobile support as a conceptual module
- Strategic platform?

"NATIVE" MOBILE

- iOS SDK
- Android SDK
- Windows Phone?

Reasoning:

MOBILE.NET

- Xamarin
 - Native
 - Forms

Reasoning:

MOBILE — HTML/HYBRID

- HTML (5), JavaScript, CSS
- PhoneGap/Cordova, CrossWalk, nw.js, ...
- Cross-platform

Reasoning:

Cloud

- Deployment option
 - Related: Docker, related orchestration (Kubernetes, ...)?
- Managed infrastructure

CLOUD FUNCTIONALITY

- Supplied services, vertical features
- Base platform functionality
 - Dynamic scalability
 - SLA
- Serverless computing

CLOUD — LEGAL CONSIDERATIONS

- Locations
- Industry/governmental requirements

CLOUD OPTIONS

- Azure, Amazon Web Services, Google (PaaS, IaaS, FaaS, ...)
- PaaS: Heroku, others
- SaaS: Office 365, Azure/AWS Websites, ...

CLOUD REASONING

- For/against cloud:
- For/against specific platforms, IaaS, PaaS:

OPEN SOURCE

- Everybody does it, right?
- Give and take...

Reasoning:

SOURCES

- This presentation:
 - <https://oliversturm.github.io/developers-and-architects/template>
 - PDF download:
<https://oliversturm.github.io/developers-and-architects/template/slides.pdf>

THANK You

Please feel free to contact me about the content anytime.

Oliver Sturm • @olivers • oliver@oliversturm.com

