Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής & Τηλεπικοινωνιών

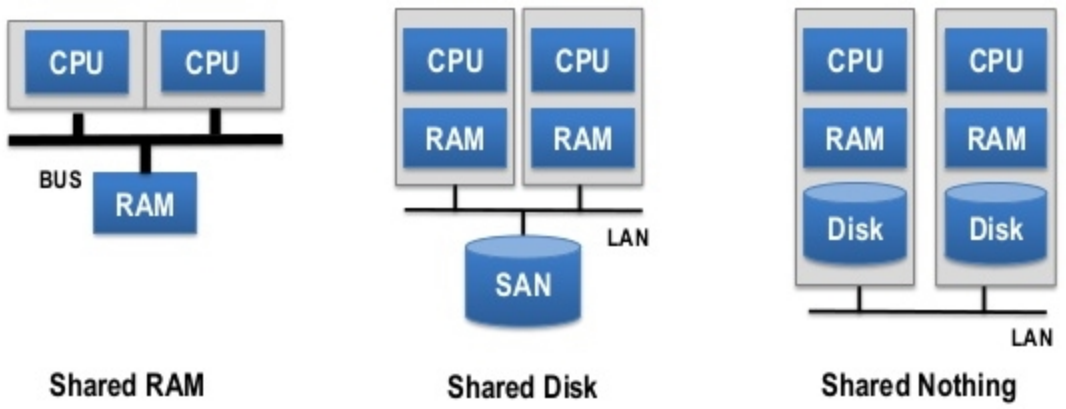
Πληροφοριακά Συστήματα

7ο Εξάμηνο 2021-22

Αρχιτεκτονική Πληροφοριακών Συστημάτων II

Δρ. Κώστας Σαΐδης (saiko@di.uoa.gr)

Share-Nothing Architecture



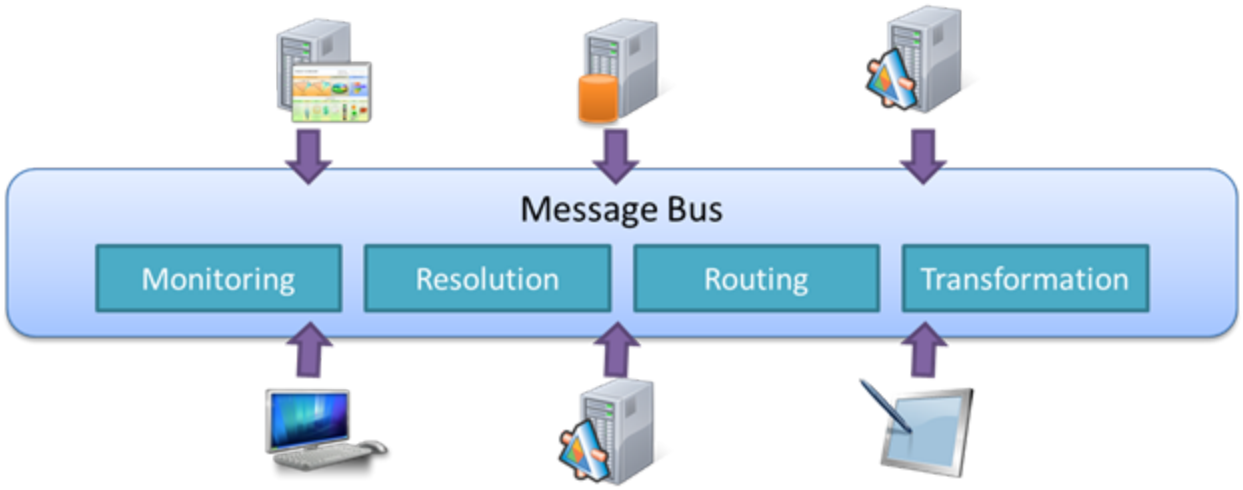
Χαρακτηριστικά

- Κάθε κόμβος είναι ανεξάρτητος και αυτοτελής.
- No single point of contention (δεν διαμοιράζονται πόροι, π.χ. μνήμη ή δίσκος).
- Sharding: οριζόντια επιμέρηση των δεδομένων.
- Οριζόντια κλιμάκωση (horizontal scalability) - απλή προσθήκη κόμβων.
- Η αρχιτεκτονική πολλών NoSQL συστημάτων.

Eventual Consistency

- **BASE Systems** (Basically Available, Soft state, Eventual consistency)
- Όταν πάψουν οι ενημερώσεις σε μια εγγραφή, τελικά (eventually) όλες οι αναγνώσεις της εγγραφής αυτής θα επιστρέψουν την πιο πρόσφατη ενημέρωση.
- Replica convergence (σύκλιση αντιγράφων)
- PA/EL (Επιλέγουν αυξημένη διαθεσιμότητα & μείωση καθυστέρησης αντί για συνέπεια)

Message-driven/Publish-subscribe



Χαρακτηριστικά

- Χαλαρή σύνδεση (loose coupling) μεταξύ συστατικών/εφαρμογών
- Publisher (producer): αποστολή μηνυμάτων
- Subscriber (consumer): λήψη μηνυμάτων
- Topics (channels): "κλάσεις/θέματα" μηνυμάτων
- Message Bus (broker): διαχερίση/δρομολόγηση μηνυμάτων σύγχρονα ή ασύγχρονα, με εγγυήσεις αποστολής ή όχι, με χρήση ουρών, με φιλτράρισμα ή όχι κτλ.

Εφαρμογές

- Middleware ολοκλήρωσης ετερογενών συστημάτων
- Επίτευξη υψηλής απόδοσης και κλιμάκωσης σε κατανεμημένα συστήματα
- Μειονέκτημα: δύσκολη η αλλαγή της δομής των μηνυμάτων

Ο Παγκόσμιος Ιστός

Βασικά συστατικά

- Client/Server
- HTTP (Hyper Text Transfer Protocol)
- HTML (Hyper Text Markup Language)
- Unified Resource Identifiers, Locators and Names (URIs, URLs, URNs)

Μορφή ενός URL

```
scheme://[user:password@]host[:port]/path[?query][#fragment]
```

HTTP

- Μέθοδοι (HEAD, GET, PUT, POST, DELETE, κ.ά)
- Headers (Host, Accept, Cookie, κ.ά)
- Status codes (200, 404, 500, κ.ά)
- Εκδόσεις: 1.0, 1.1, 2.0

Παράδειγμα

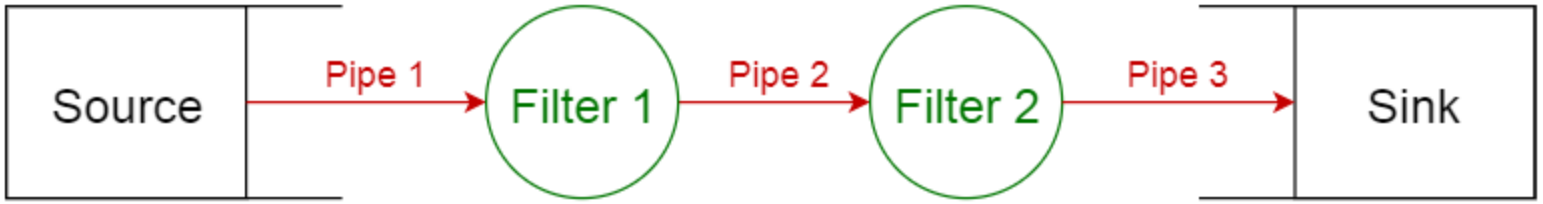
Αίτηση

```
GET /index.html HTTP/1.1
<line feed>
```

Απάντηση

```
HTTP/1.1 200 OK
Date: Wed, 29 Mar 2017 14:38:00 GMT
Server: Apache/1.3.27 (Unix) ...
Last-Modified: Wed, 29 Mar 2017 01:16:05 GMT
Accept-Ranges: bytes
Content-Length: 6188
Content-Type: text/html
<html>
<head>...</head><body>...</body>
</html>
```

Pipeline / Pipe-filter



Χαρακτηριστικά

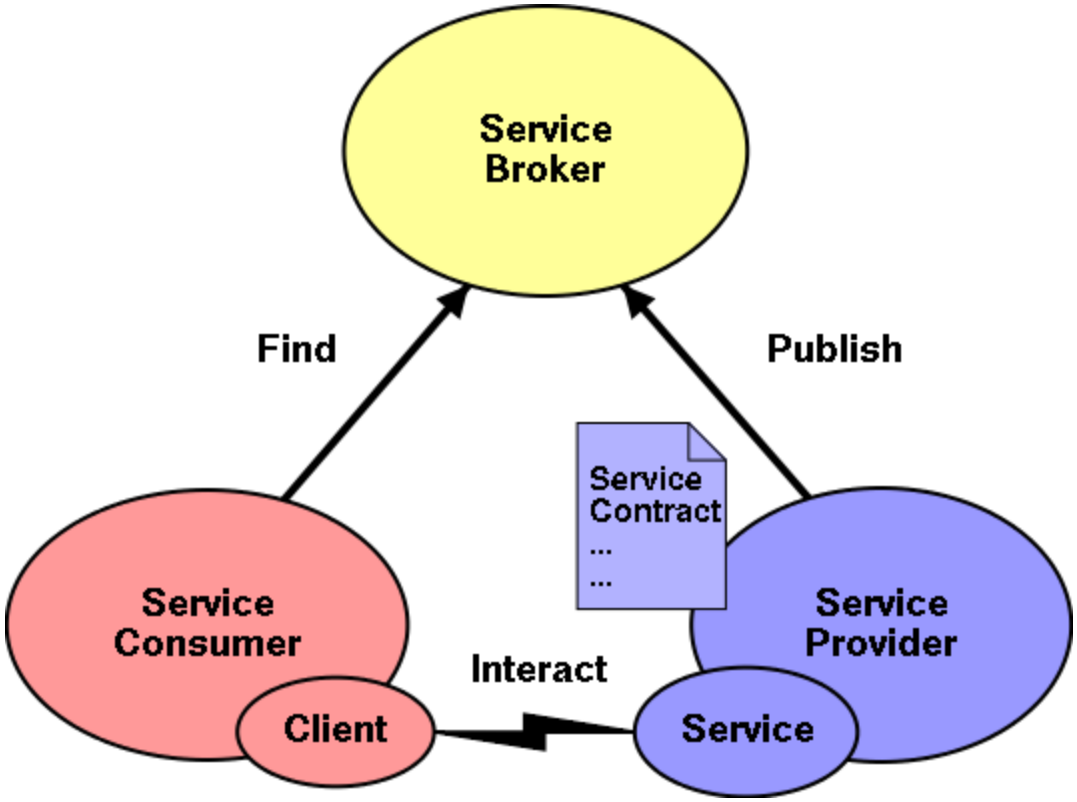
- Data streams, pipes and filters (data transformations)
- Συναρτησιακός προγραμματισμός
- Επαναχρησιμοποίηση, παραλληλισμός

Service-oriented architecture

Η κεντρική ιδέα

- Υπηρεσίες που επικοινωνούν μέσω ενός πρωτοκόλλου επικοινωνίας και είναι:
  - κατανεμημένες,
  - αυτοτελείς (separately maintained & deployed)
  - χαλάρα συνδεδεμένες (loosely-coupled)
  - ανεξάρτητες της τεχνολογίας υλοποίησης (technology-neutral)
  - ανεξάρτητες του κατασκευαστή (no vendor lock-in)
- Σύνθεση της εφαρμογής μέσω της ολοκλήρωσης (integration) των υπηρεσιών.

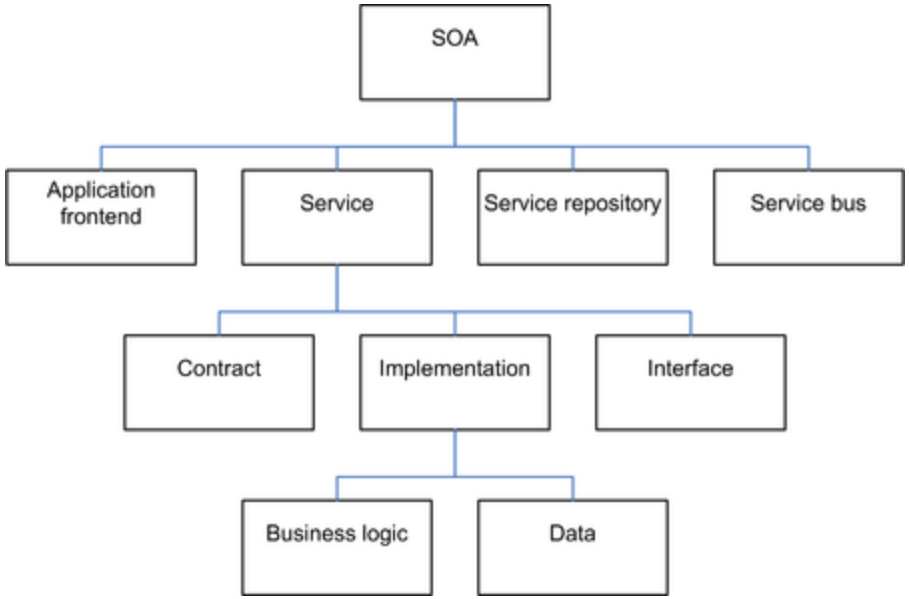
Ρόλοι



- Service consumer
- Service producer
- Service broker

SOA Αρχές

- Service contract
- Metadata
- Composition
- Autonomy
- Discovery
- Reusability


SOA
Application frontend
Service
Service repository
Service bus
Contract
Implementation
Interface
Business logic
Data

Υλοποίηση

- Web Services (SOAP, WSDL, UDDI)
- Remote-procedure call (RPC)
- Message-driven middleware
- RESTful APIs
- κ.ά

Ζητήματα

- Stateful vs Stateless
- Απόδοση
- Πολυπλοκότητα
- Έλεγχος και επαλήθευση (testing)

SOA & REST

- Ενδεχομένως ο απλούστερος τρόπος υλοποίησης μιας SOA
- Ένα σύνολο από RESTful HTTP end-points που ανταλλάσσουν δεδομένα σε μορφή JSON

Η Αρχιτεκτονική Representational State Transfer (REST)

RESTful APIs

- Υπάρχουν παντού (Web, Microservices, IoT)

Τι είναι το REST

- Ένα αρχιτεκτονικό στυλ για τη λειτουργία του Παγκόσμιου Ιστού (ή, γενικά, κατανεμημένων συστημάτων)
- Οριοθετεί αρχές, περιορισμούς και βασικές λειτουργίες
- Ανεξάρτητα της γλώσσας προγραμματισμού, του πρωτοκόλλου επικοινωνίας ή του είδους των δεδομένων

Συνοπτικά

Τέσσερις βασικές έννοιες:

- Resources, Representations, Requests, Responses

Έξι βασικές αρχές:

- Client-server, Stateless, Cacheable, Layered System, Uniform Interface, Code on demand (προαιρετικά)

Οφέλη

- Η σωστή χρήση των RESTful αρχών βελτιώνει όλα τα σημαντικά χαρακτηριστικά μιας αρχιτεκτονικής:
  - Απόδοση
  - Κλιμάκωση
  - Απλότητα
  - Επεκτασιμότητα
  - Αξιοπιστία

RESTful Web Service (API)

Αποτελείται από:

- Ένα HTTP base URL (REST endpoint)
- Ένα ή περισσότερα MimeType για τα representations
- HTTP Methods: GET, PUT ή PATCH, POST, DELETE

Παράδειγμα

- Έστω μια συλλογή (collection) από στοιχεία (items).
- Υπάρχουν δύο βασικά endpoints:
  - `[baseURL]/items` : Ενέργειες στη συλλογή (εν συνόλω).
  - `[baseURL]/items/id` : Ενέργειες στο στοιχείο (ειδικά).
- REST Calls
  - `GET [baseURL]/items`
  - `PUT [baseURL]/items/32`

Συλλογές

HTTP Method	Ενέργεια
GET	Εμφάνιση της λίστας με τα elements του collection.
PUT	Αντικατάσταση του υπάρχοντος collection με νέο collection (αν υποστηρίζεται).
POST	Προσθήκη νέου element στο υπάρχον collection.
DELETE	Διαγραφή του υπάρχοντος collection (αν υποστηρίζεται).

Στοιχεία

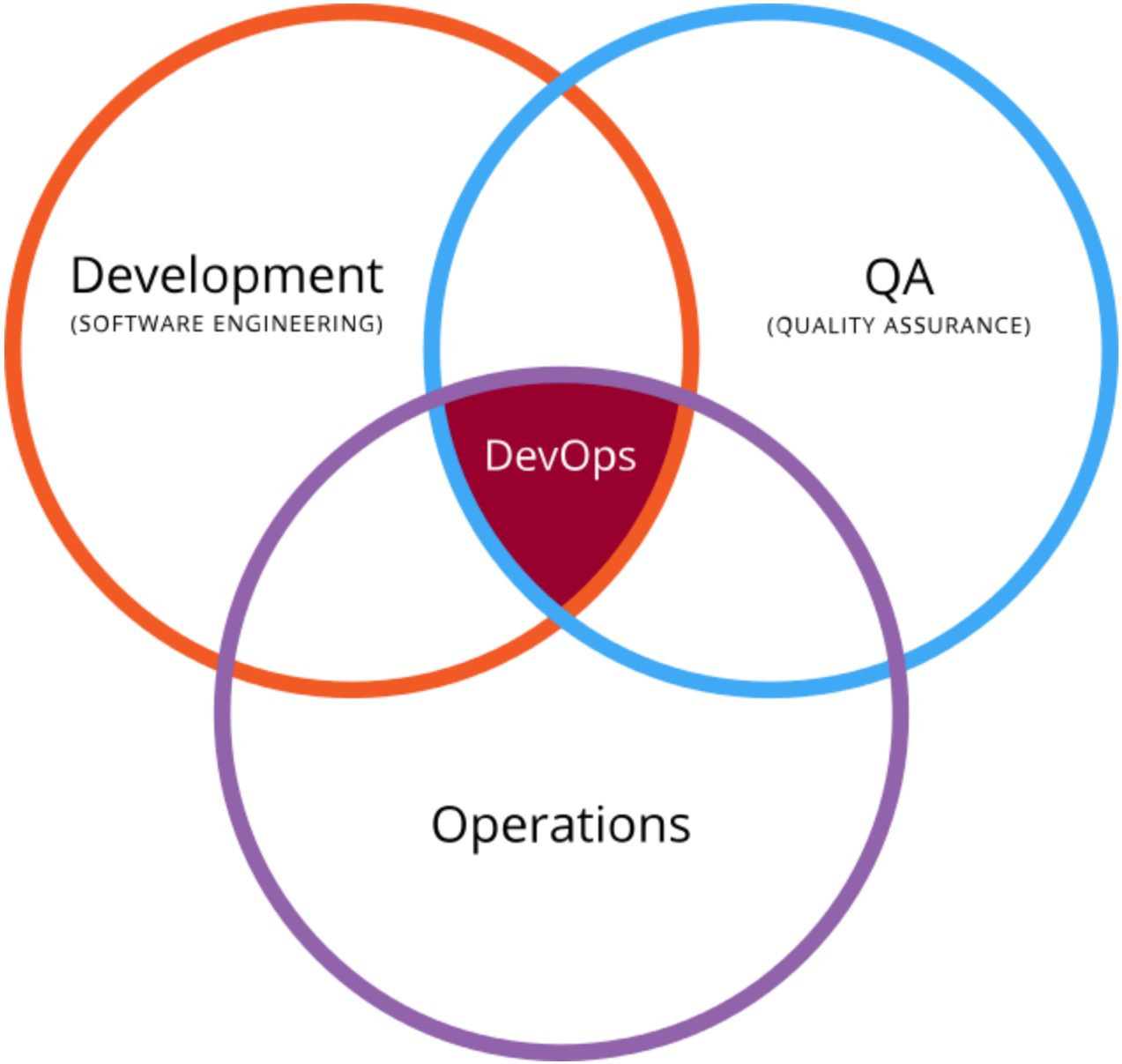
HTTP Method	Ενέργεια
GET	Ανάκτηση μιας αναπαράστασης του συγκεκριμένου item.
PUT ή PATCH	Τροποποίηση του συγκεκριμένου item.
POST	Δε χρησιμοποιείται ευρέως για items.
DELETE	Διαγραφή του συγεκριμένου item.

Microservices

Κωδικοποιημένα

Microservices = SOA + Unix Principles + Agile + DevOps

DevOps

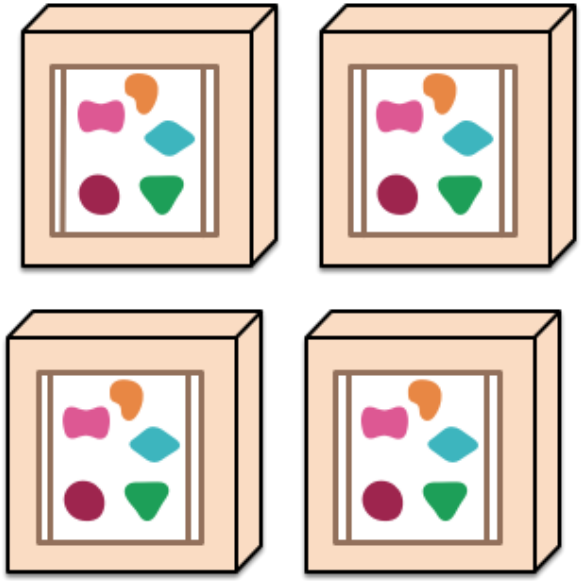

Development
(SOFTWARE ENGINEERING)
QA
(QUALITY ASSURANCE)
DevOps
Operations

Monolithic Server-side Applications

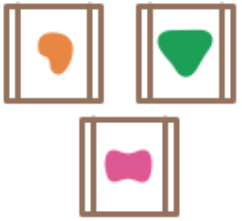
- A single logical executable
- Cloud deployment issues:
  - Small changes -> rebuild and redeploy the whole app
  - Scalability -> scale it all or not at all

A monolithic application puts all its functionality into a single process...

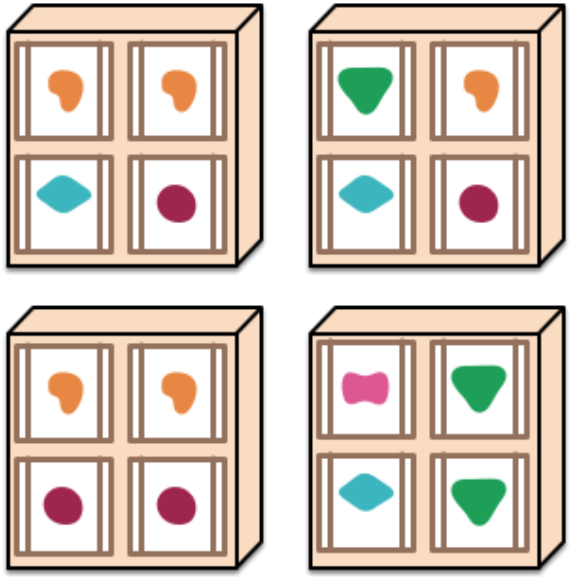
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...



... and scales by distributing these services across servers, replicating as needed.



Χαρακτηριστικά

- Componentization via services
  - As in SOA
- Organized around business capabilities
  - Cross-functional teams, as in Agile
- Products not projects
  - A team should own a product
  - You build it, you run it
- Smart end-points, dumb pipes
  - Simple communication (Unix-style)

- Decentralized governance
  - Each team governs the implementation details of its product
- Decentralized data managument
  - Different datastore per service/app
- Infrastructure automation
  - Build & test automation
  - Continuous delivery & deployment
- Design for failure
  - Sophisticated monitoring & logging

Devops toolchain for Microservices