Technology for Microservices

Rag Dhiman

ragcode.com | @RagDhiman

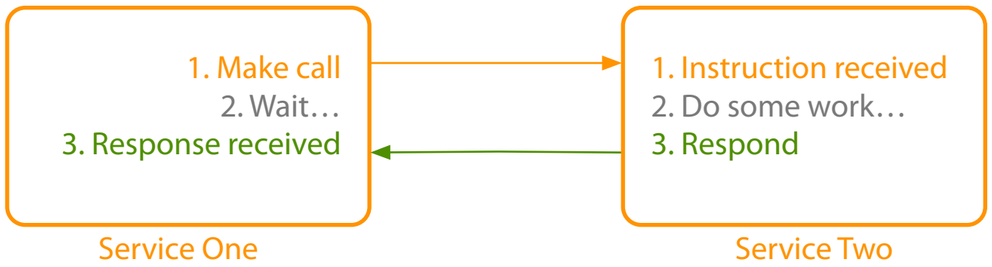
Module Overview

- Communication
- Hosting Platforms
- Observable Microservices
- Performance
- Automation Tools

Communication

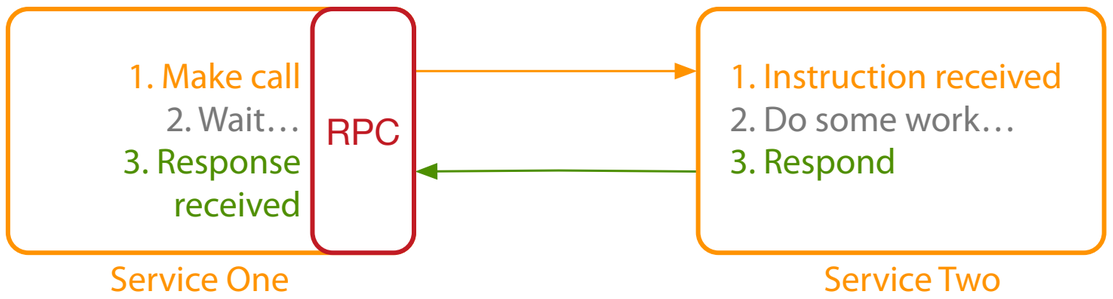
Synchronous | Asynchronous

Communication: Synchronous



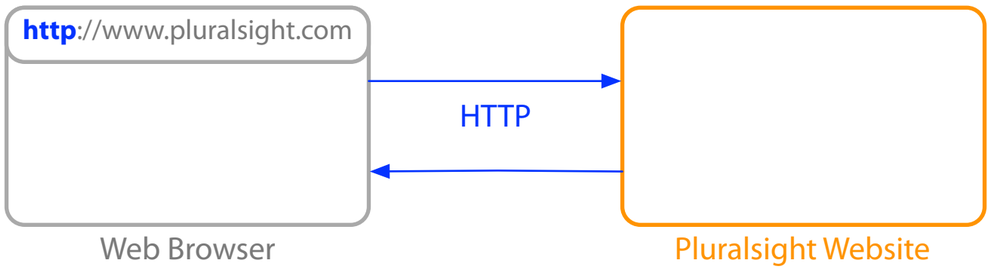
- Request response communication
  - Client to service
  - Service to service
  - Service to external
- Remote procedure call
  - Sensitive to change
- HTTP
  - Work across the internet
  - Firewall friendly
- REST
  - CRUD using HTTP verbs
  - Natural decoupling
  - Open communication protocol
  - REST with HATEOS
- Synchronous issues
  - Both parties have to be available
  - Performance subject to network quality
  - Clients must know location of service (host\port)

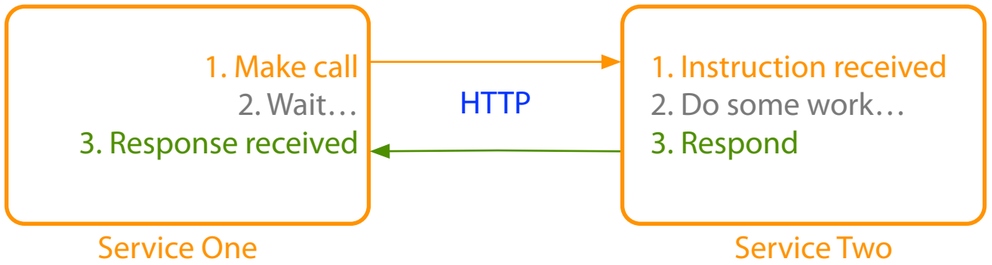
Communication: Synchronous



- Request response communication
  - Client to service
  - Service to service
  - Service to external
- Remote procedure call
  - Sensitive to change
- HTTP
  - Work across the internet
  - Firewall friendly
- REST
  - CRUD using HTTP verbs
  - Natural decoupling
  - Open communication protocol
  - REST with HATEOS
- Synchronous issues
  - Both parties have to be available
  - Performance subject to network quality
  - Clients must know location of service (host\port)

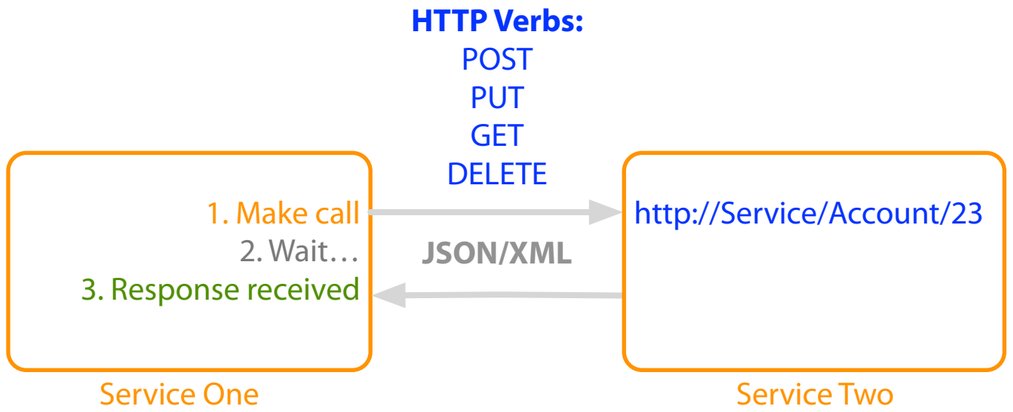
Communication: Synchronous





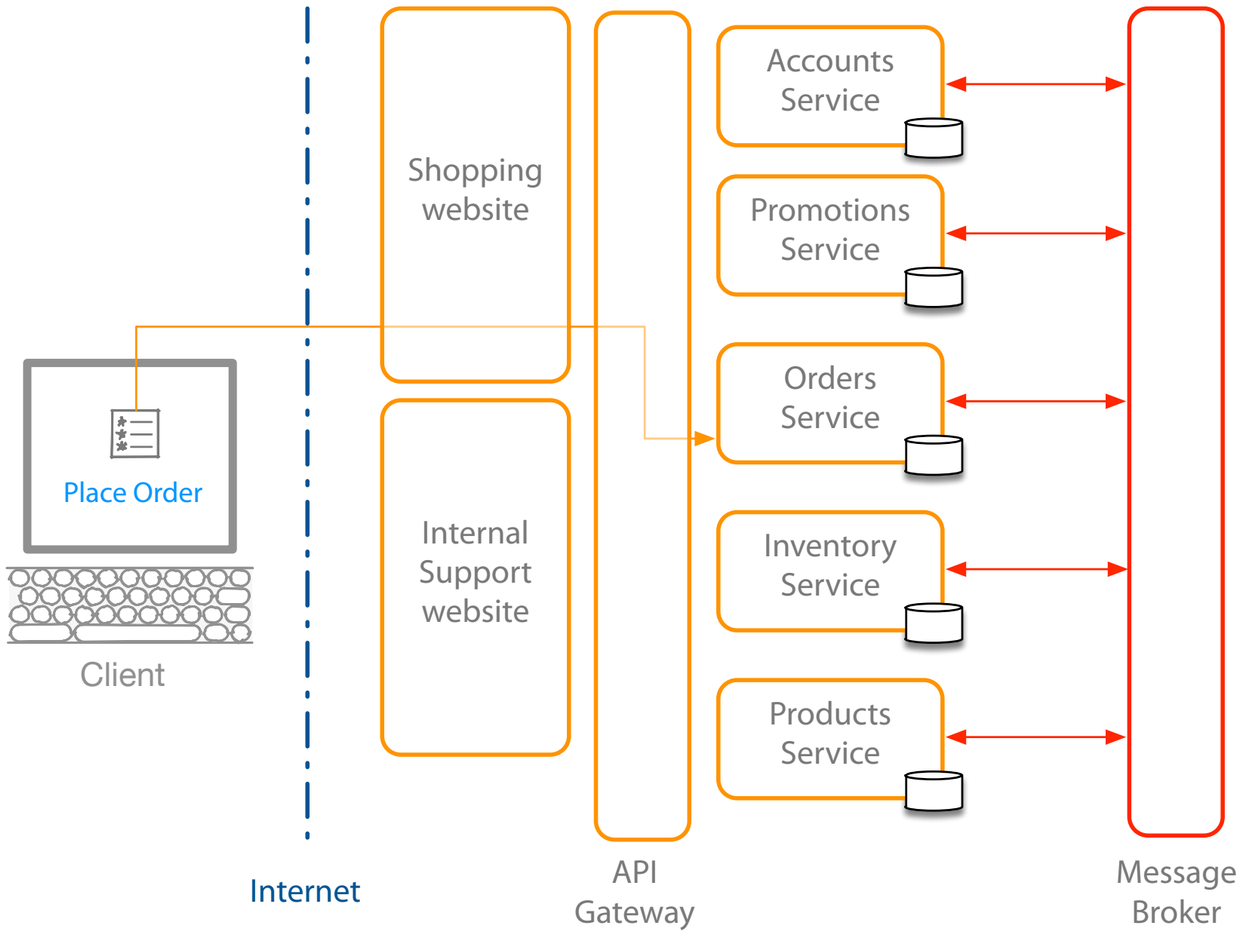
- Request response communication
  - Client to service
  - Service to service
  - Service to external
- Remote procedure call
  - Sensitive to change
- HTTP
  - Work across the internet
  - Firewall friendly
- REST
  - CRUD using HTTP verbs
  - Natural decoupling
  - Open communication protocol
  - REST with HATEOS
- Synchronous issues
  - Both parties have to be available
  - Performance subject to network quality
  - Clients must know location of service (host\port)

Communication: Synchronous



- Request response communication
  - Client to service
  - Service to service
  - Service to external
- Remote procedure call
  - Sensitive to change
- HTTP
  - Work across the internet
  - Firewall friendly
- REST
  - CRUD using HTTP verbs
  - Natural decoupling
  - Open communication protocol
  - REST with HATEOS
- Synchronous issues
  - Both parties have to be available
  - Performance subject to network quality
  - Clients must know location of service (host\port)

Communication: Asynchronous



Event based

- Mitigates the need of client and service availability
- Decouples client and service

Message queueing protocol

- Message Brokers
- Subscriber and publisher are decoupled
- Microsoft message queuing (MSMQ)
- RabbitMQ
- ATOM (HTTP to propagate events)

Asynchronous challenge

- Complicated
- Reliance on message broker
- Visibility of the transaction
- Managing the messaging queue

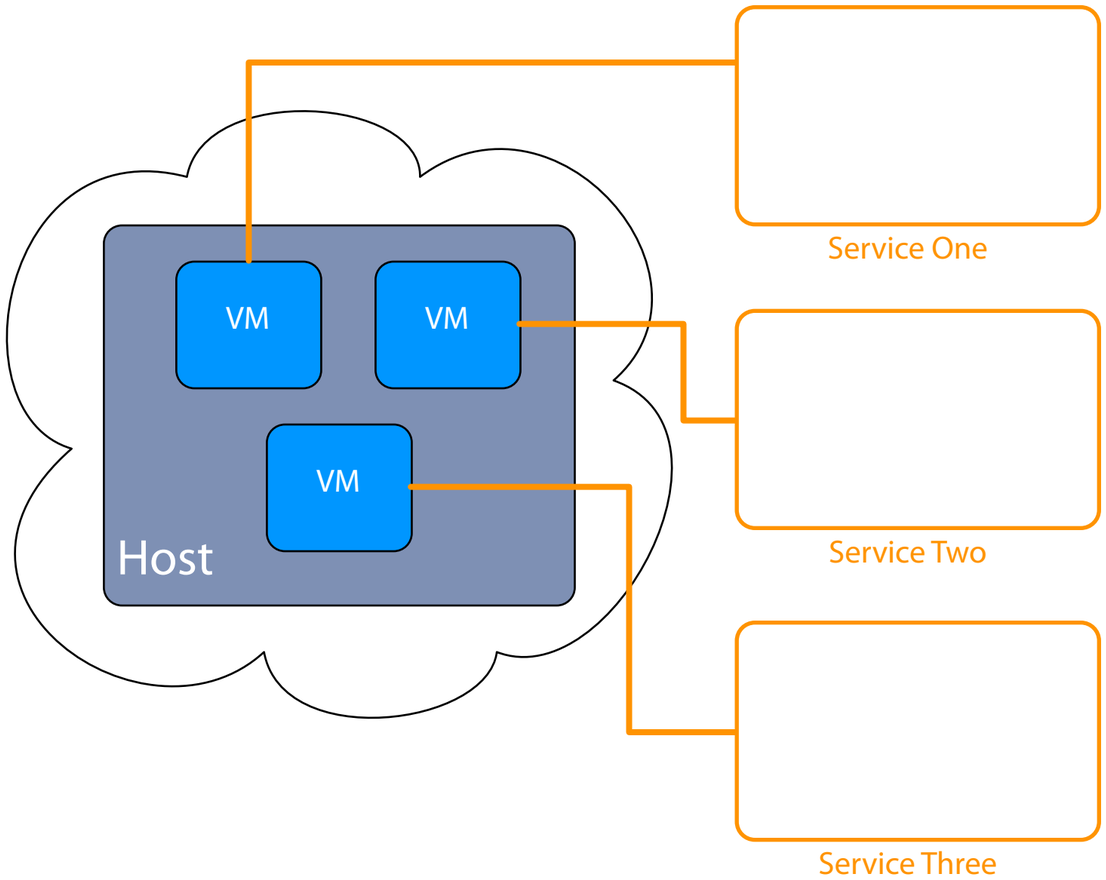
Real world systems

- Would use both synchronous and asynchronous

Hosting Platforms

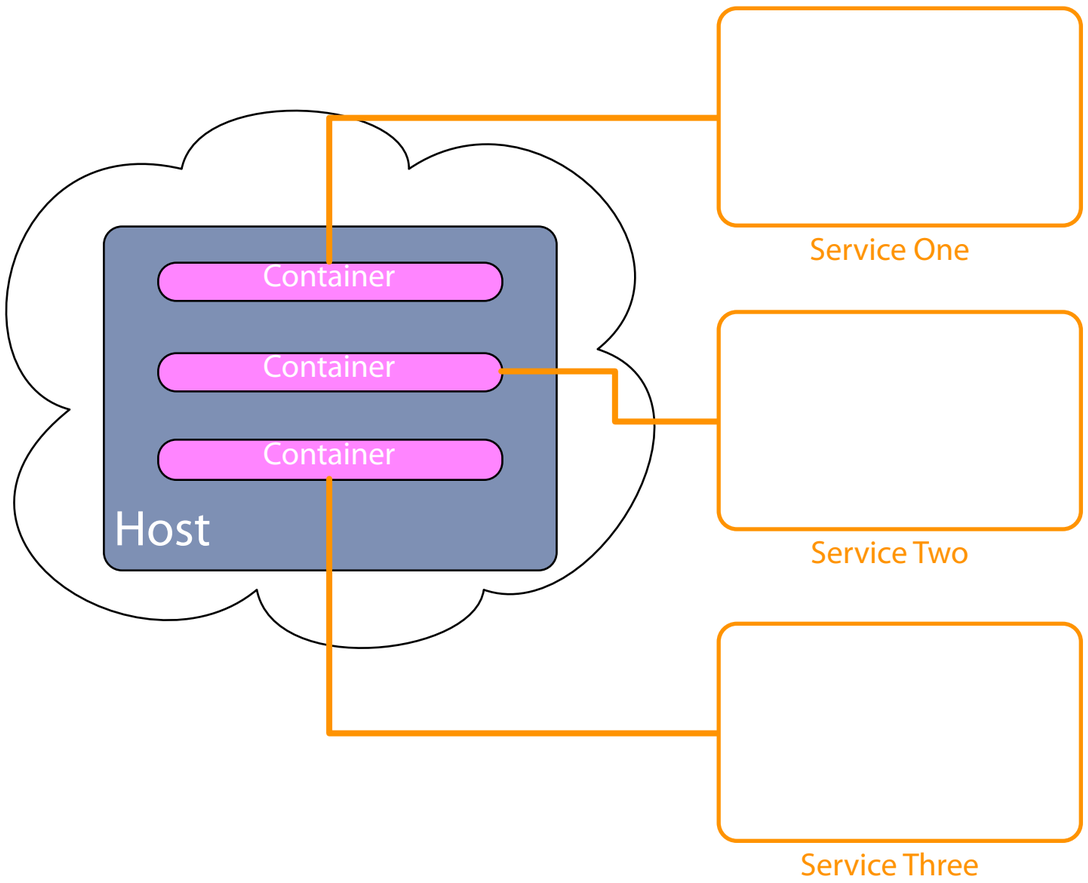
Virtualization | Containers | Self Hosting | Registry and Discovery

Hosting Platforms: Virtualization



- A virtual machine as a host
- Foundation of cloud platforms
  - Platform as a service (PAAS)
    - Microsoft Azure
    - Amazon web services
    - Your own cloud (for example vSphere)
- Could be more efficient
  - Takes time to setup
  - Takes time to load
  - Take quite a bit of resource
- Unique features
  - Take snapshot
  - Clone instances
- Standardised and mature

Hosting Platforms: Containers

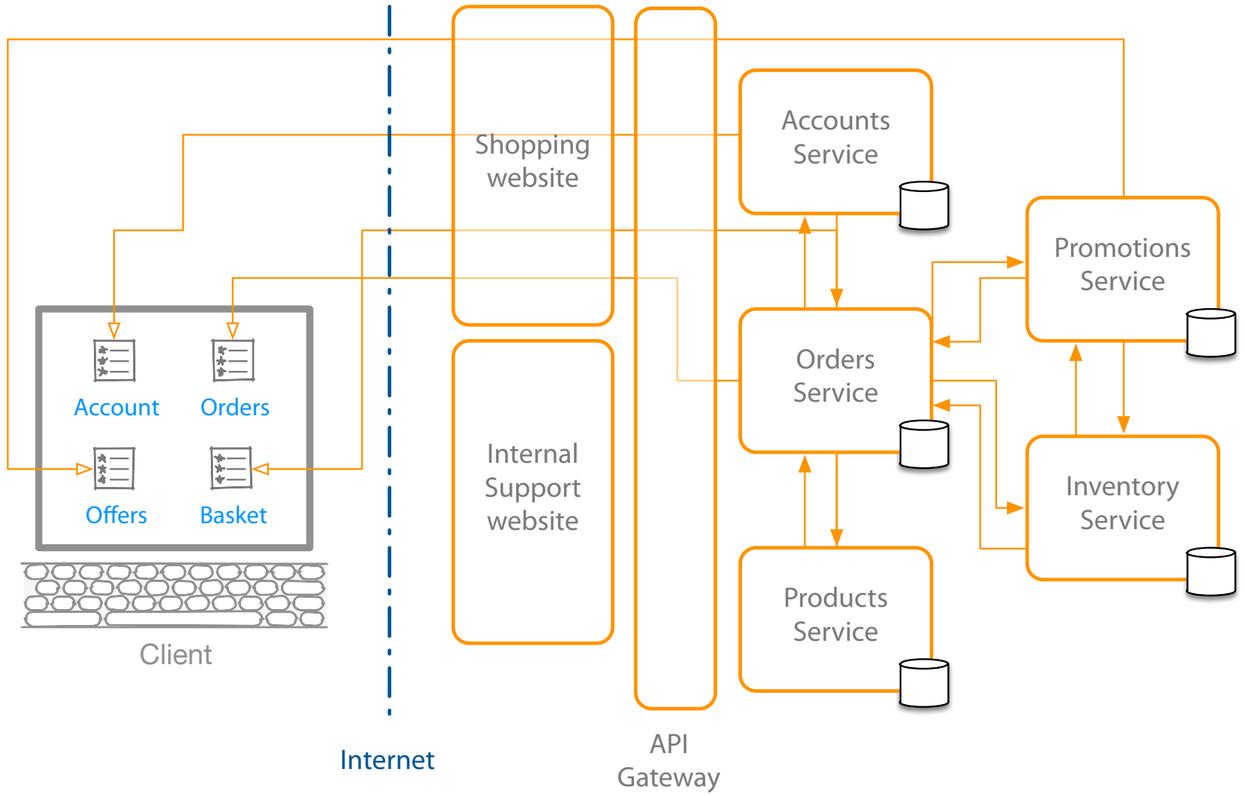


- Type of virtualization
- Isolate services from each other
- Single service per container
- Different to a virtual machine
  - Use less resource than VM
  - Faster than VM
  - Quicker to create new instances
- Future of hosted apps
- Cloud platform support growing
- Mainly Linux based
- Not as established as virtual machines
  - Not standardised
  - Limited features and tooling
  - Infrastructure support in its infancy
  - Complex to setup
- Examples
  - Docker
  - Rocker
  - Glassware

Hosting Platforms: Self Hosting

- Implement your own cloud
  - Virtualization platform
  - Implement containers
- Use of physical machines
  - Single service on a server
  - Multiple services on a server
- Challenges
  - Long-term maintenance
  - Need for technicians
  - Training
  - Need for space
  - Scaling is not as immediate

Hosting Platforms: Registration and Discovery



Where?

- Host, port and version

Service registry database

Register on startup

Deregister service on failure

Cloud platforms make it easy

Local platform registration options

- Self registration
- Third-party registration

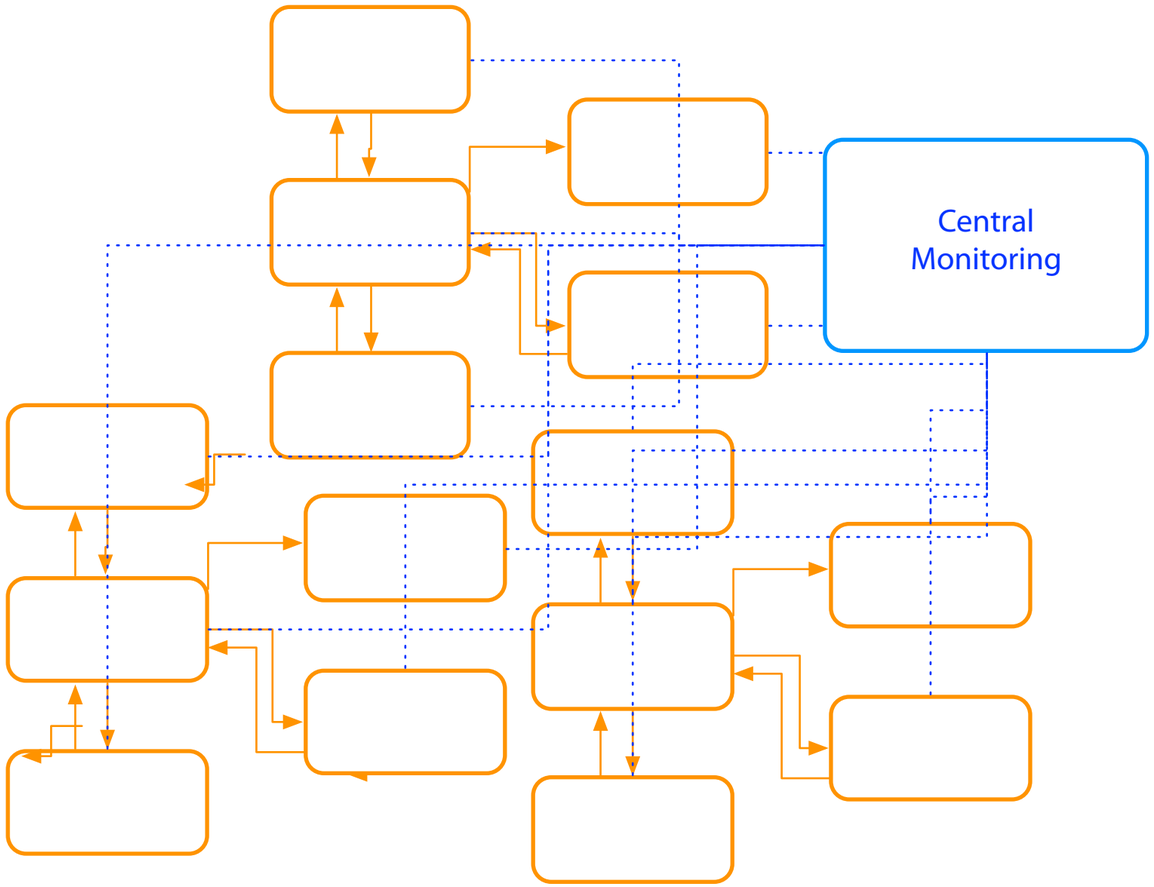
Local platform discovery options

- Client-side discovery
- Server-side discovery

Observable Microservices

Monitoring Tech | Logging Tech

Observable Microservices: Monitoring Tech

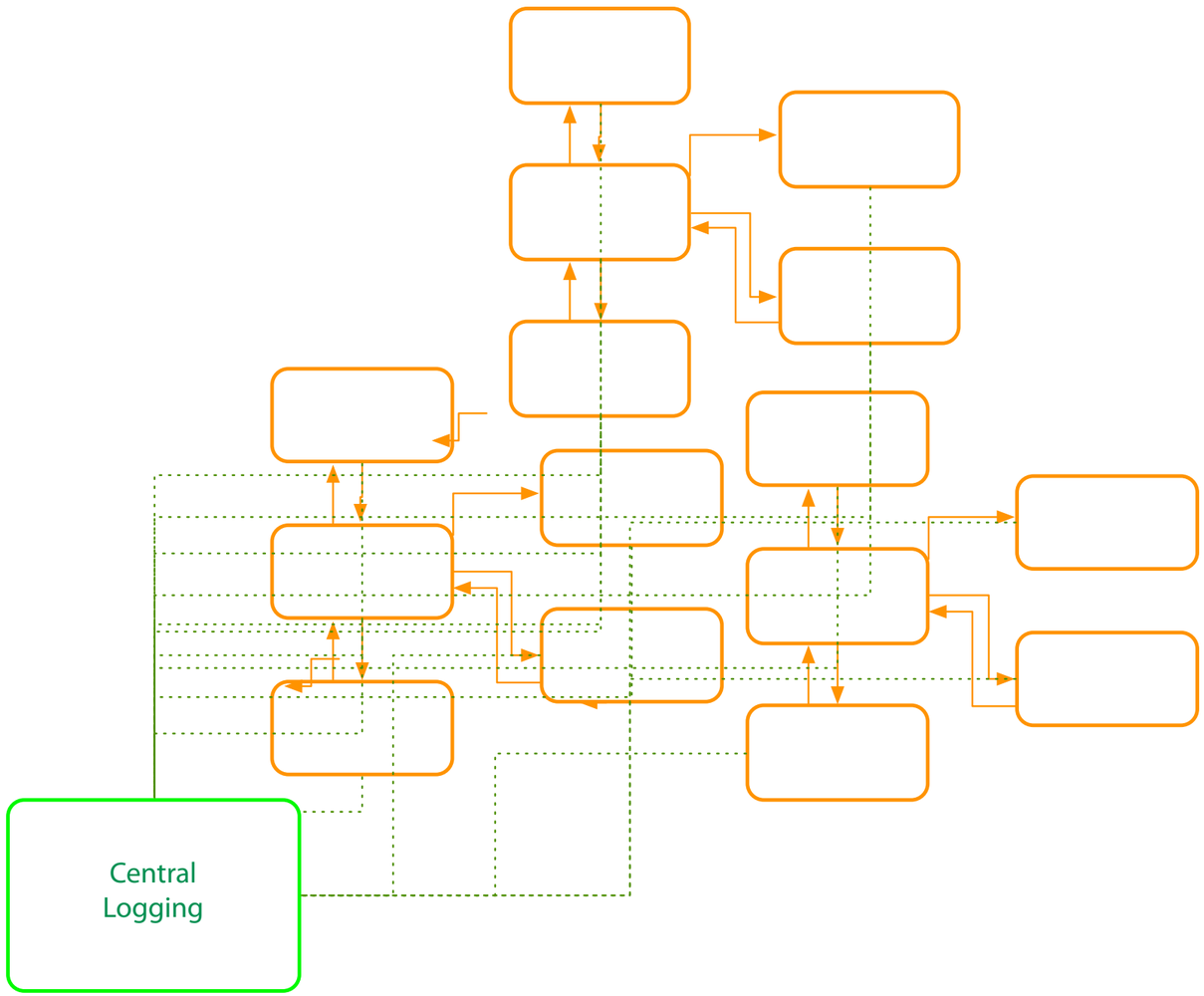


- Centralised tools
  - Nagios
  - PRTG
  - Load balancers
  - New Relic
- Desired features
  - Metrics across servers
  - Automatic or minimal configuration
  - Client libraries to send metrics
  - Test transactions support
  - Alerting
- Network monitoring
- Standardise monitoring
  - Central tool
  - Preconfigured virtual machines or containers
- Real-time monitoring

Observable Microservices

Monitoring Tech | Logging Tech

Observable Microservices: Logging Tech

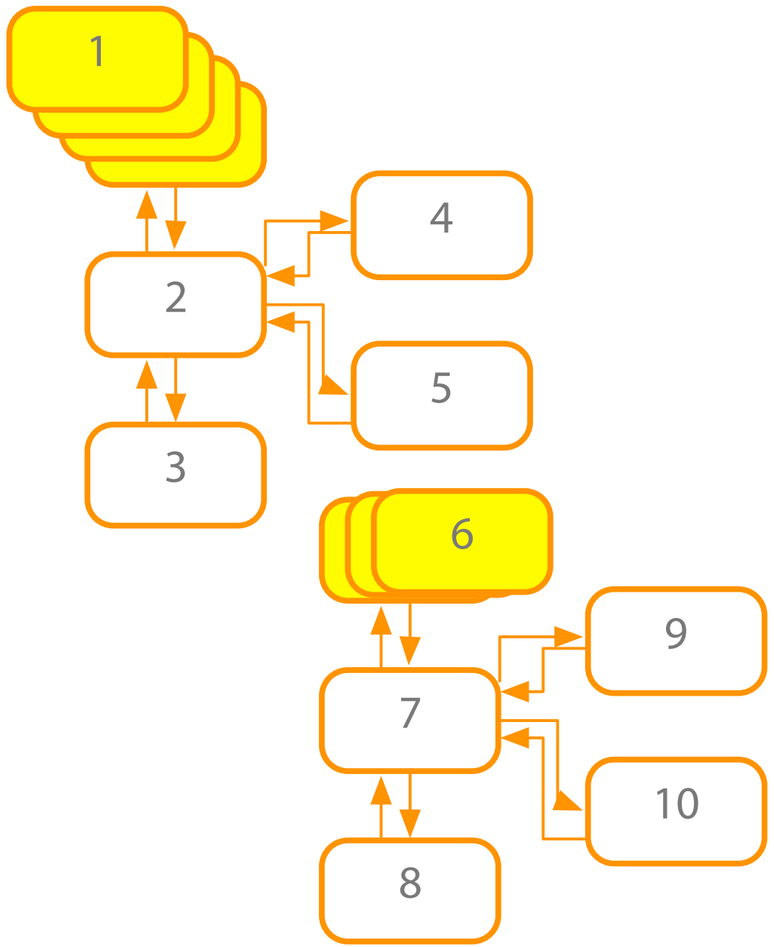


- Portal for centralised logging data
  - Elastic log
  - Log stash
  - Splunk
  - Kibana
  - Graphite
- Client logging libraries
  - Serilog
  - and many more…
- Desired features
  - Structured logging
  - Logging across servers
  - Automatic or minimal configuration
  - Correlation\Context ID for transactions
- Standardise logging
  - Central tool
  - Template for client library

Microservices Performance

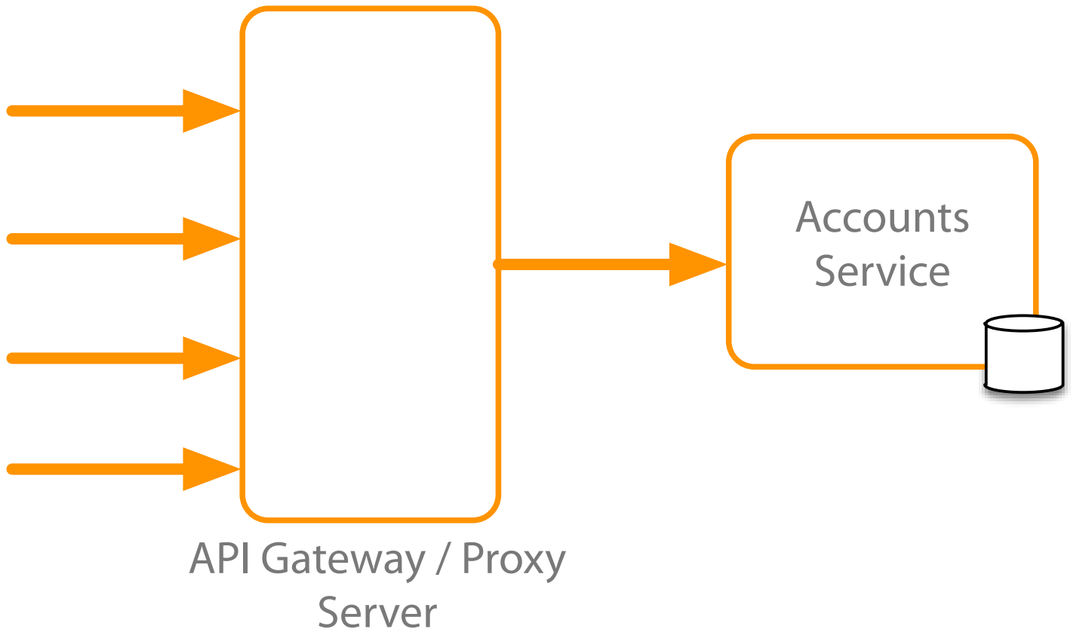
Scaling | Caching | API Gateway

Microservices Performance: Scaling



- How
  - Creating multiple instances of service
  - Adding resource to existing service
- Automated or on-demand
- PAAS auto scaling options
- Virtualization and containers
- Physical host servers
- Load balancers
  - API Gateway
- When to scale up
  - Performance issues
  - Monitoring data
  - Capacity planning

Microservices Performance: Caching



Caching to reduce

- Client calls to services
- Service calls to databases
- Service to service calls

API Gateway\Proxy level

Client side

Service level

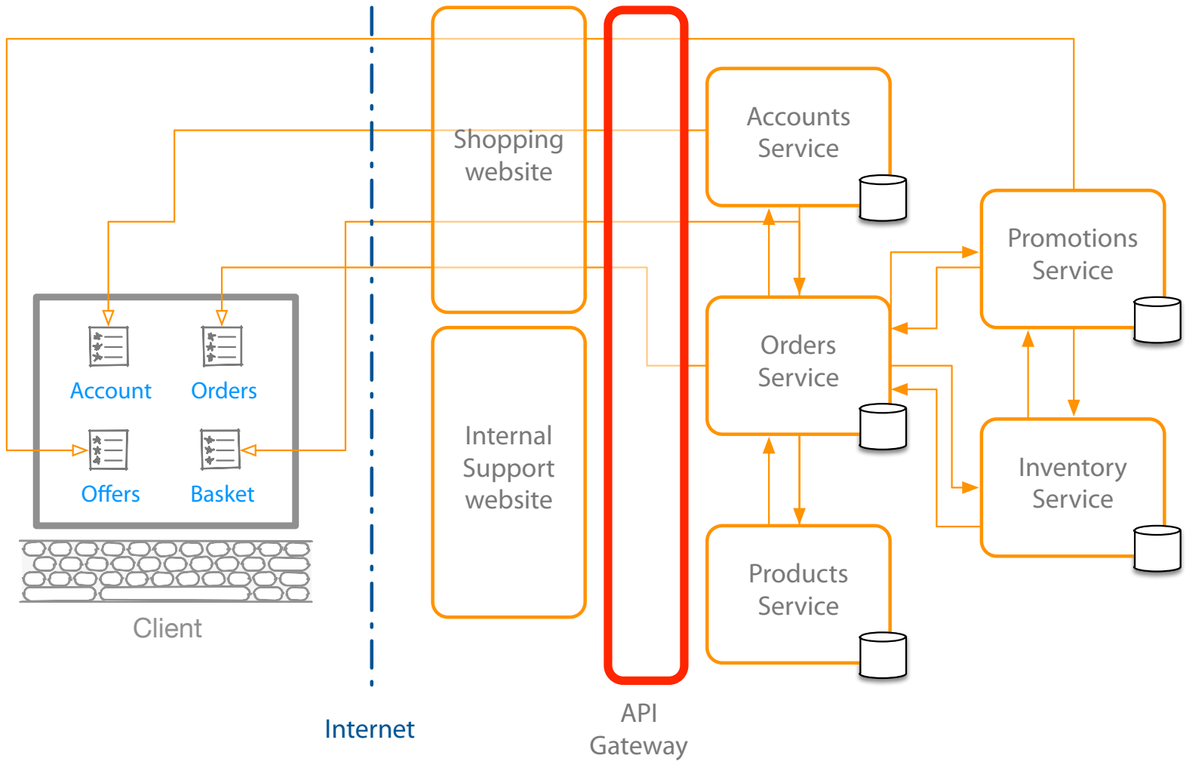
Considerations

- Simple to setup and manage
- Data leaks

Microservices Performance

Scaling | Caching | API Gateway

Microservices Performance: API Gateway



Help with performance

- Load balancing
- Caching

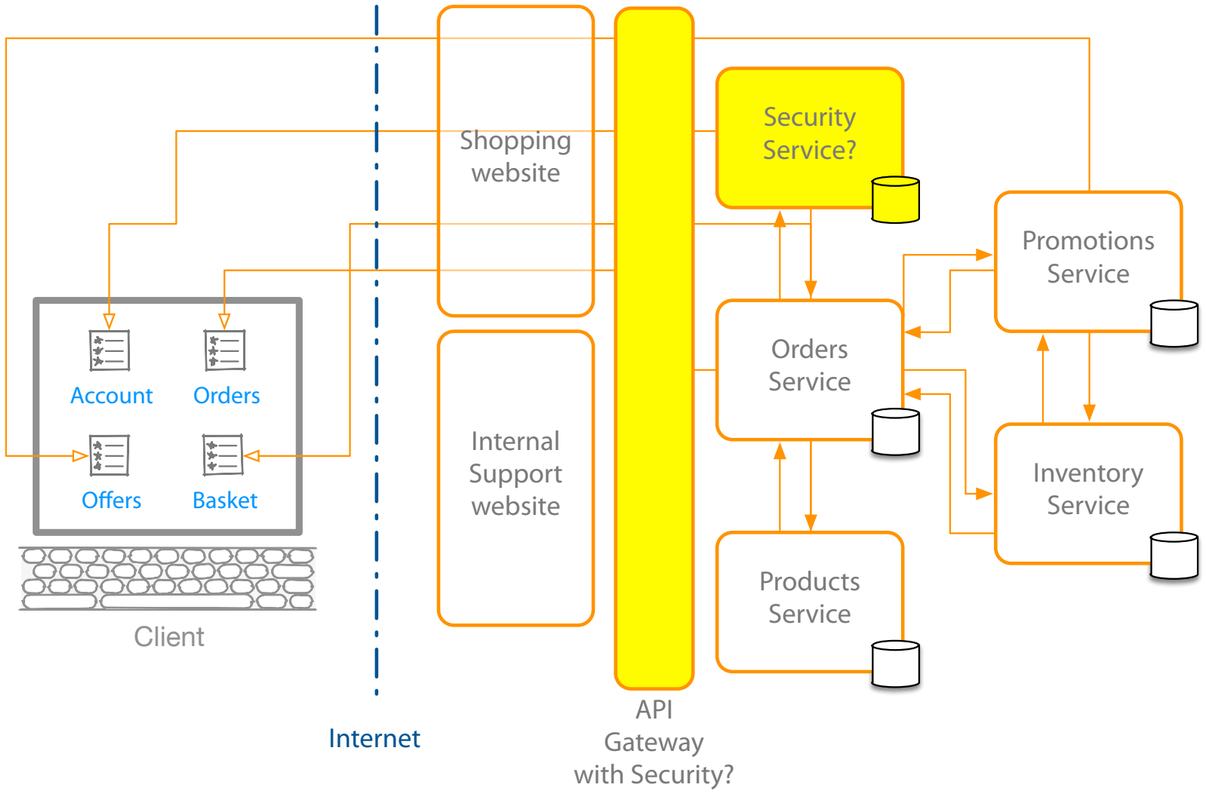
Help with

- Creating central entry point
- Exposing services to clients
- One interface to many services
- Dynamic location of services
- Routing to specific instance of service
- Service registry database

Security

- API Gateway
- Dedicated security service
- Central security vs service level

Microservices Performance: API Gateway



Help with performance

- Load balancing
- Caching

Help with

- Creating central entry point
- Exposing services to clients
- One interface to many services
- Dynamic location of services
- Routing to specific instance of service
- Service registry database

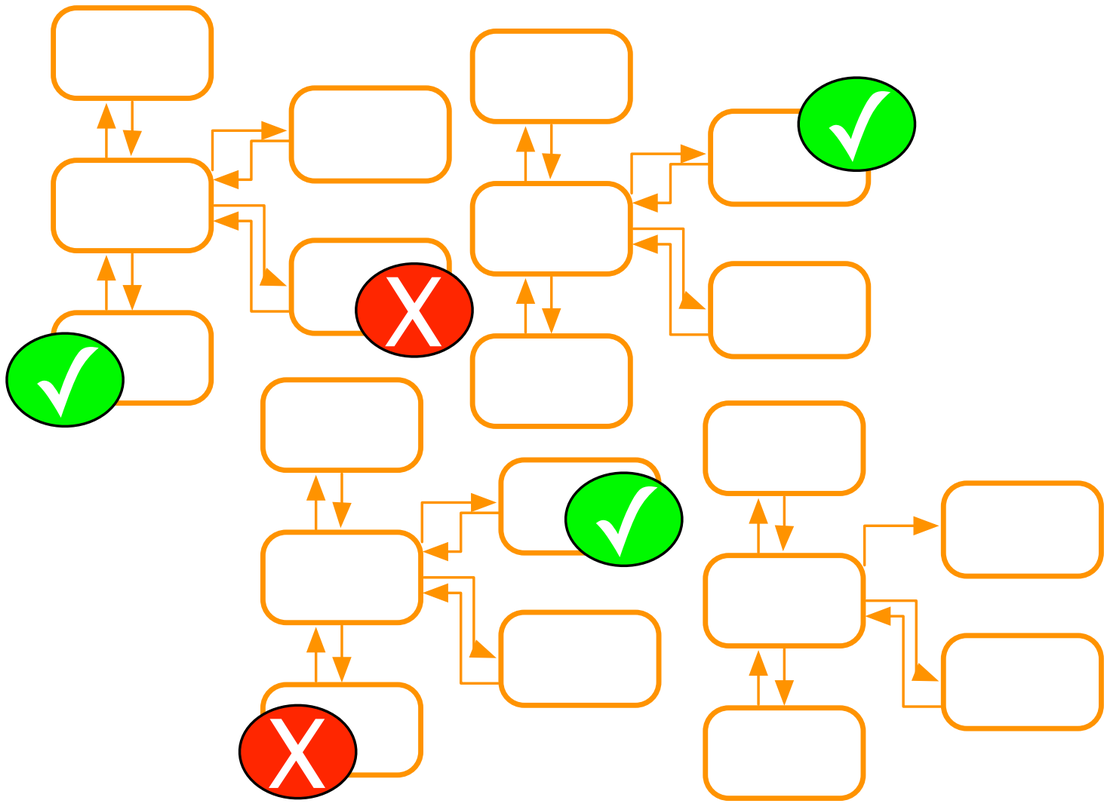
Security

- API Gateway
- Dedicated security service
- Central security vs service level

Automation Tools

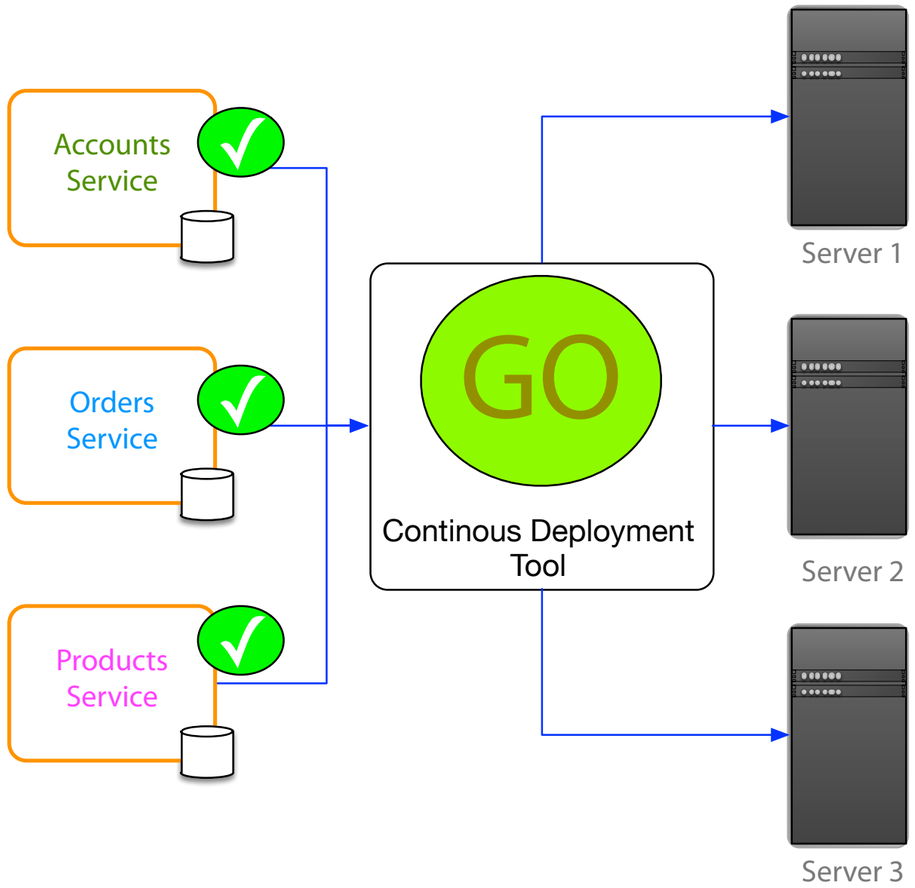
Continuous Integration | Continuous Deployment

Automation Tools: Continuous Integration



- Many CI tools
  - Team Foundation Server
  - TeamCity
  - Many more!
- Desired features
  - Cross platform
    - Windows builders, Java builders and others
  - Source control integration
  - Notifications
  - IDE Integration (optional)
- Map a microservice to a CI build
  - Code change triggers build of specific service
  - Feedback just received on that service
  - Builds and tests run quicker
  - Separate code repository for service
  - End product is in one place
  - CI builds to test database changes
  - Both microservice build and database upgrade are ready
- Avoid one CI build for all services

Automation Tools: Continuous Deployment



Many CD tools

- Aim for cross platform tools

Desired features

- Central control panel
- Simple to add deployment targets
- Support for scripting
- Support for build statuses
- Integration with CI tool
- Support for multiple environments
- Support for PAAS

Module Summary

- Communication
  - Synchronous
  - Asynchronous
- Hosting Platforms
  - Virtualization
  - Containers
  - Self Hosting
  - Registry and Discovery
- Observable Microservices
  - Monitoring Tech
  - Logging Tech
- Performance
  - Scaling
  - Caching
  - API Gateway
- Automation Tools
  - Continuous Integration
  - Continuous Deployment