
MAJOR
WELCOME
LEAGUE
TORONTO
CHICAGO
LONDON
KAUNAS
VILNIUS

Microservices – challenges for testing

Rimantas Benetis, Director of Technology

Does microservices mean micro testing?

Evolution

Single purpose applications run on mainframes

- Low user base
- Usage defined by developers
- No expectations

Single purpose applications run on personal computers

- User base increases
- Driven by technical capability rather than users
- Small expectations

Hosted applications in intranet

- Limited user base
- Driven by organization
- Some business defined requirements

Cloud services

- Large user base
- Driven by users
- User expects more than functionality
- High competition

Microservices

Meet the monolith

Self-contained application (does everything)

Multiple business purposes

Highly coupled

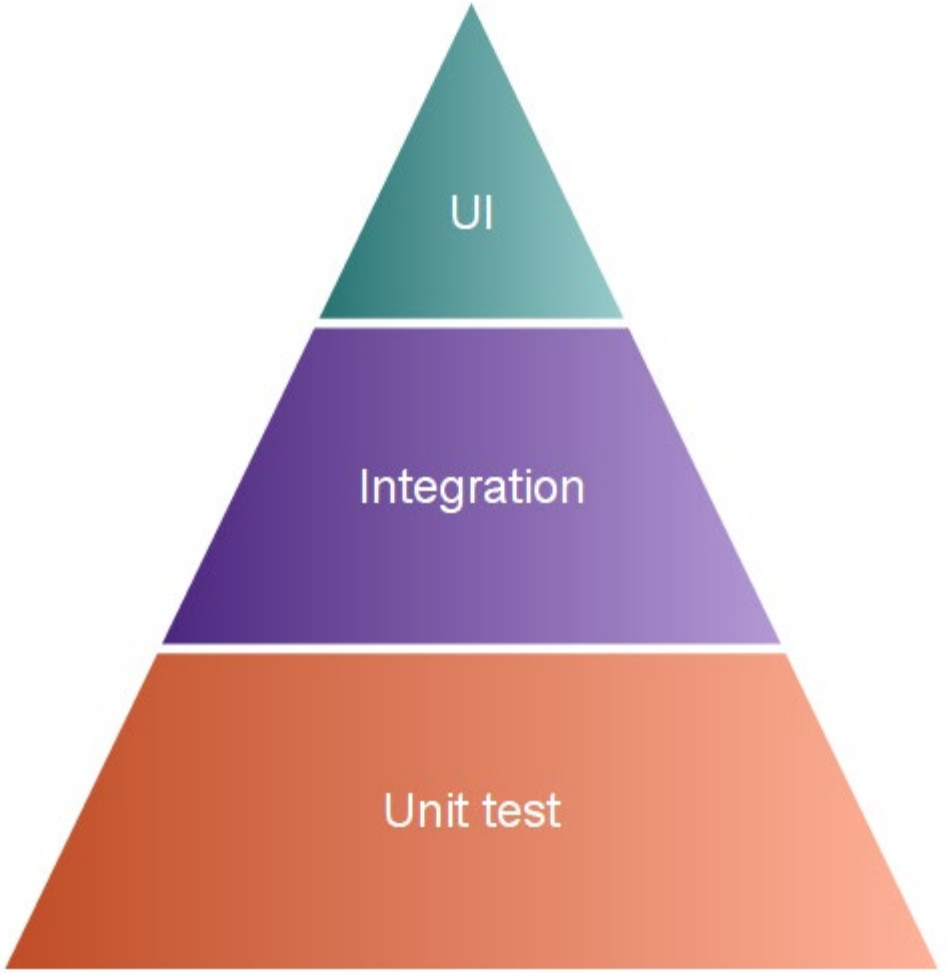
High complexity

Difficult scaling

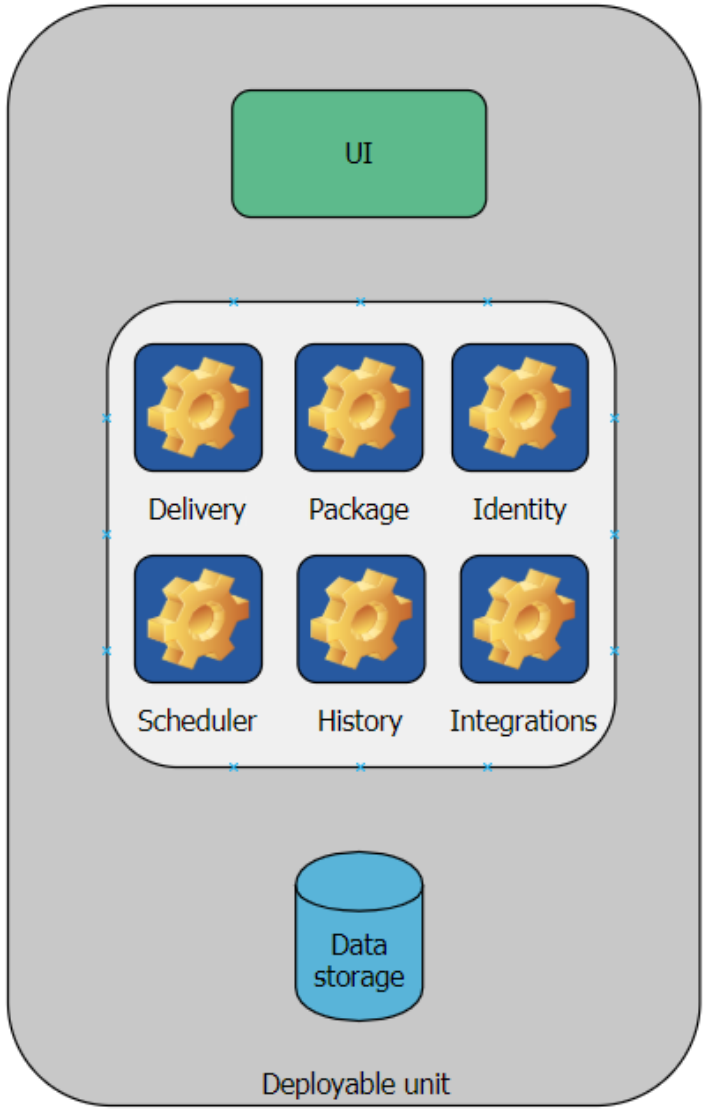
Continuous deployment is difficult

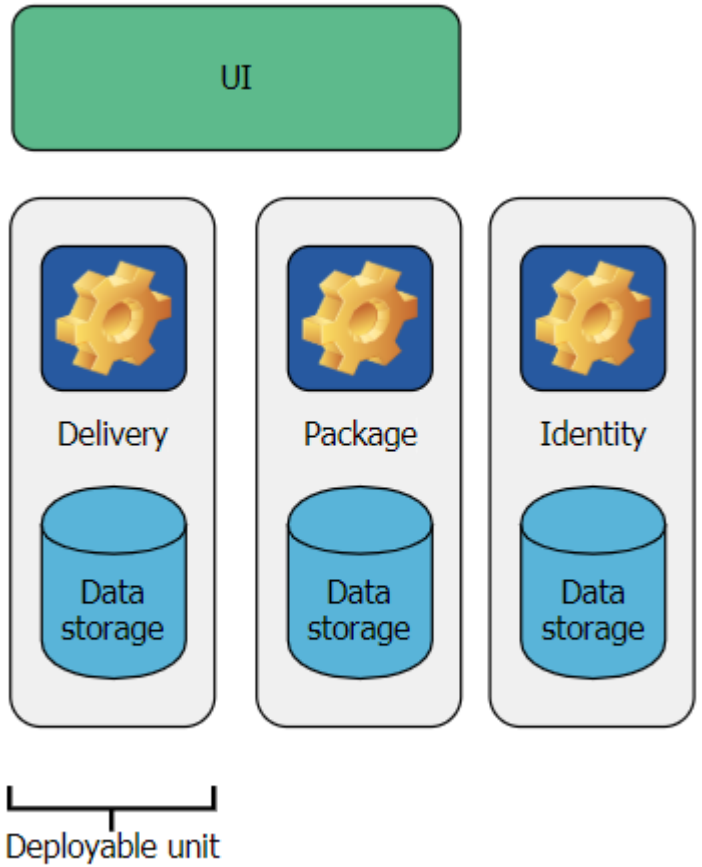
Monolith

- Complex unit tests
- Difficult integration tests
- Long running automation



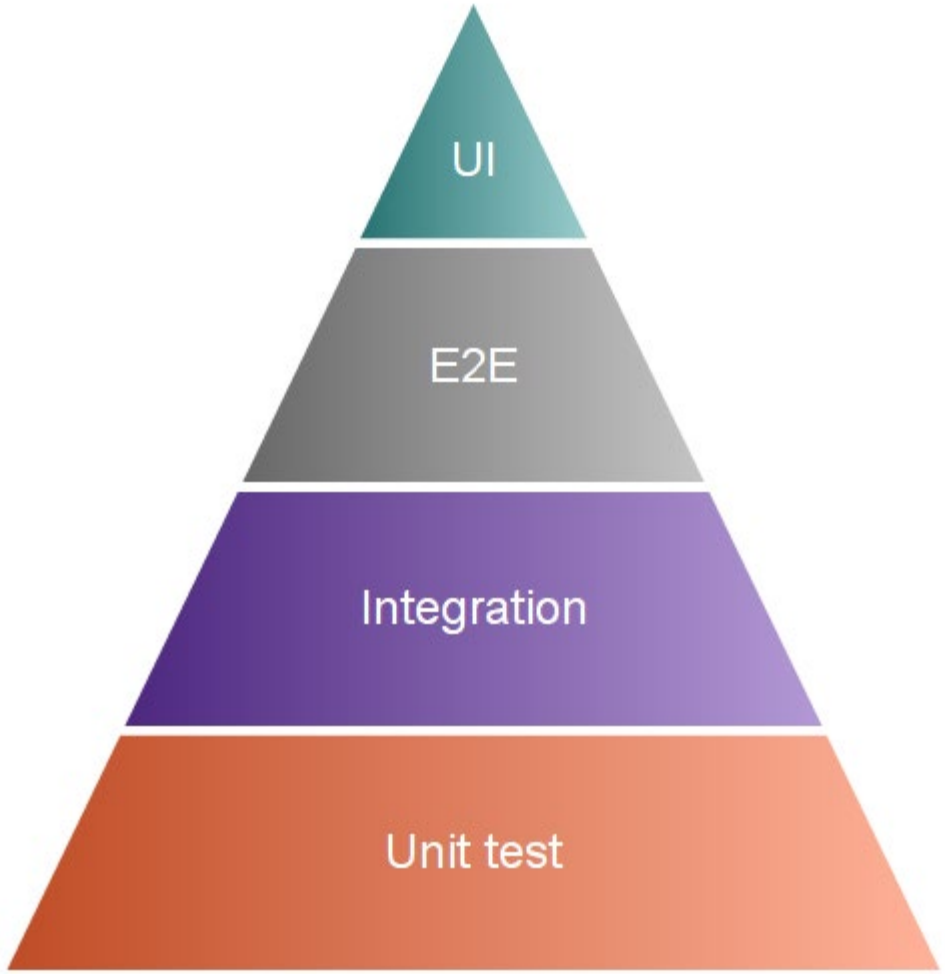
Monolith vs microservices





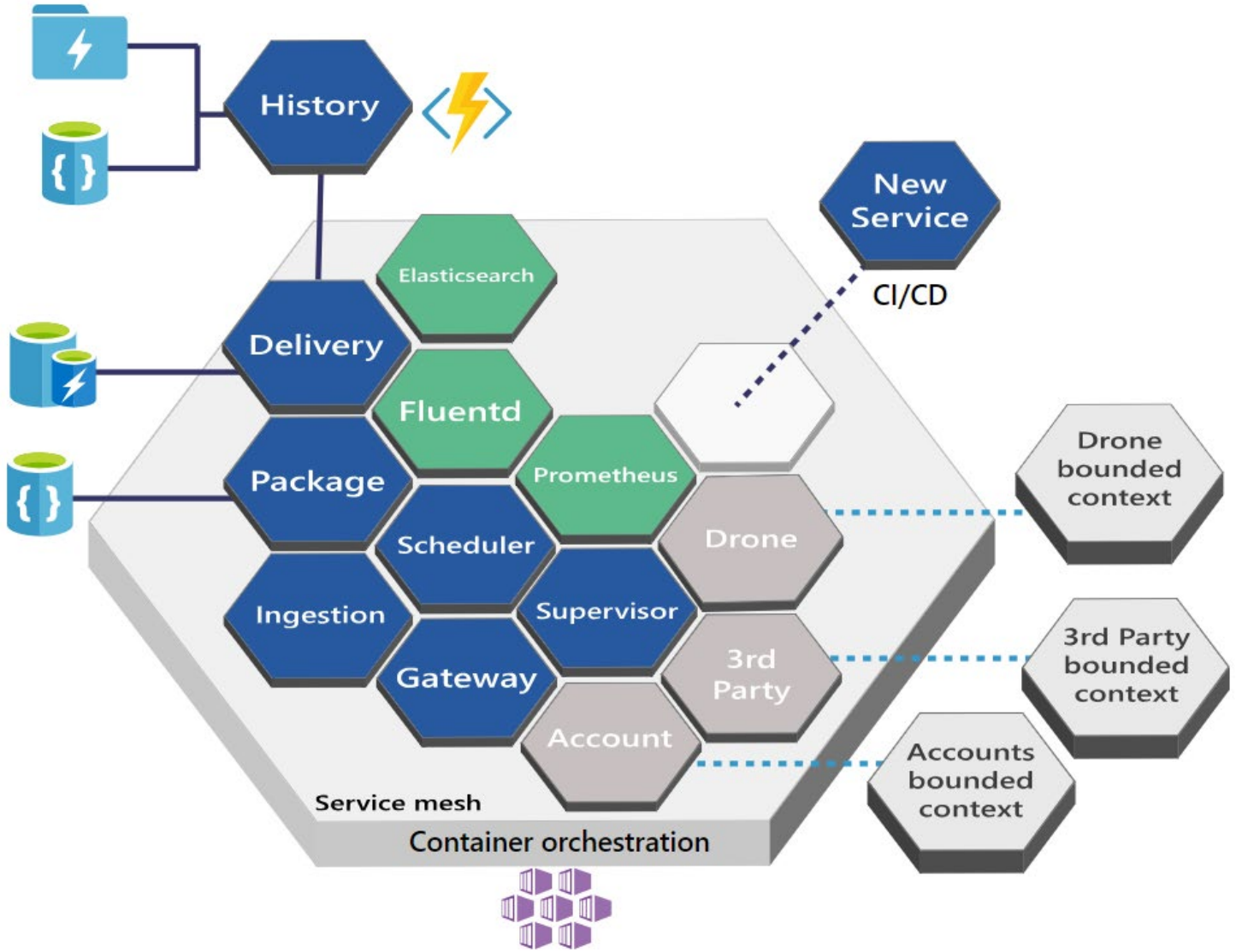
Microservice

- Simple unit tests
- Limited scope integration tests
- Stand alone e2e tests
- Quick running automation



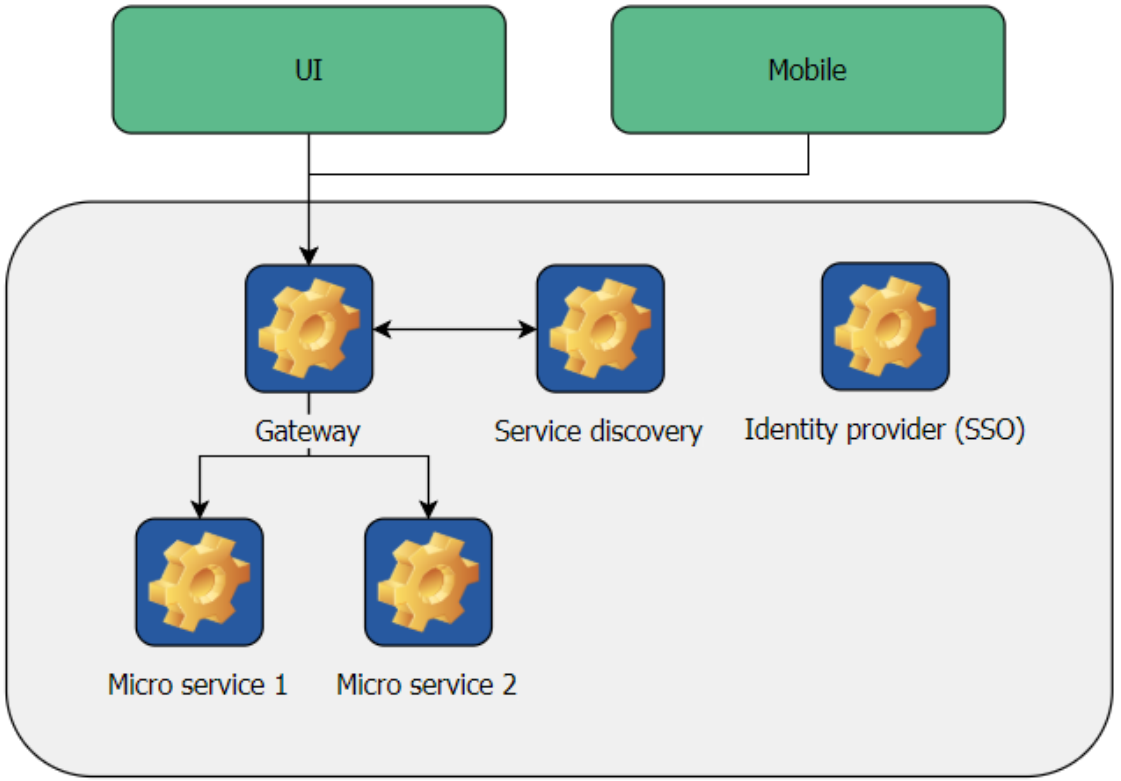
Doesn't it look the same?

Bigger picture



Typical communication flow

- Previously owned communication moves to outside
- Standard flows/protocols/services emerge
- Tracing requests becomes an issue

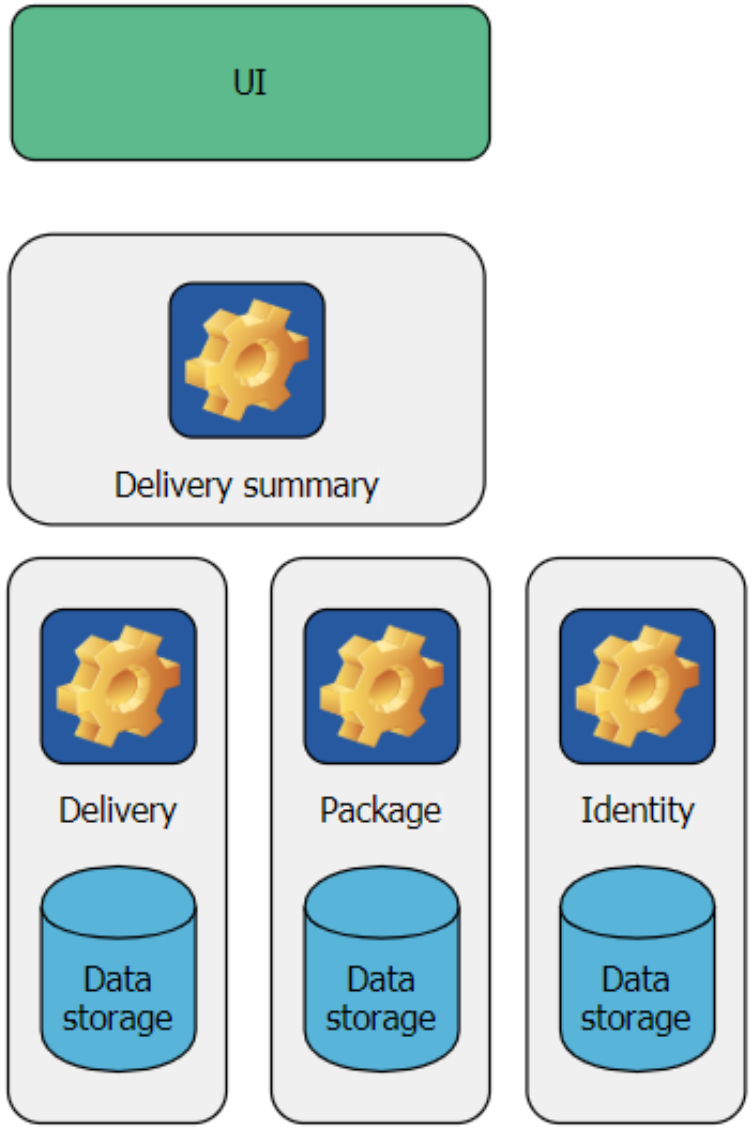


More moving parts - more things to go wrong

- Context boundaries
- Mocking dependencies
- Dependency configuration tests
- Tracing capability
- Multiple instance routing
- Graceful degradation

Context boundaries

- Help identify domain contexts
- Verify context spill
- Data overexposing



Mocking dependencies

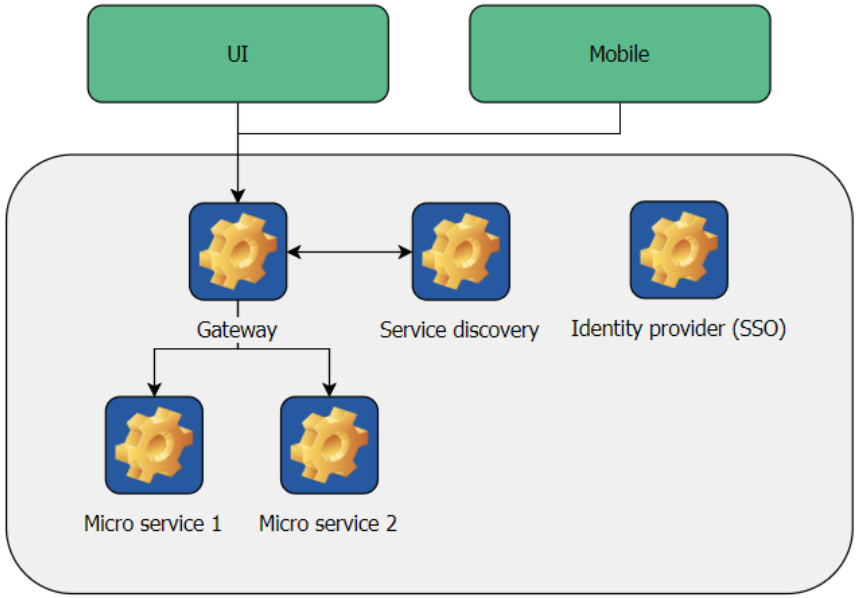
- More “external” dependencies
- Mock dependent services for isolated e2e tests
- Automate dependency setup for testing
- Decouples from other teams
- Isolates from system configuration changes

Moutebank

POSTMAN

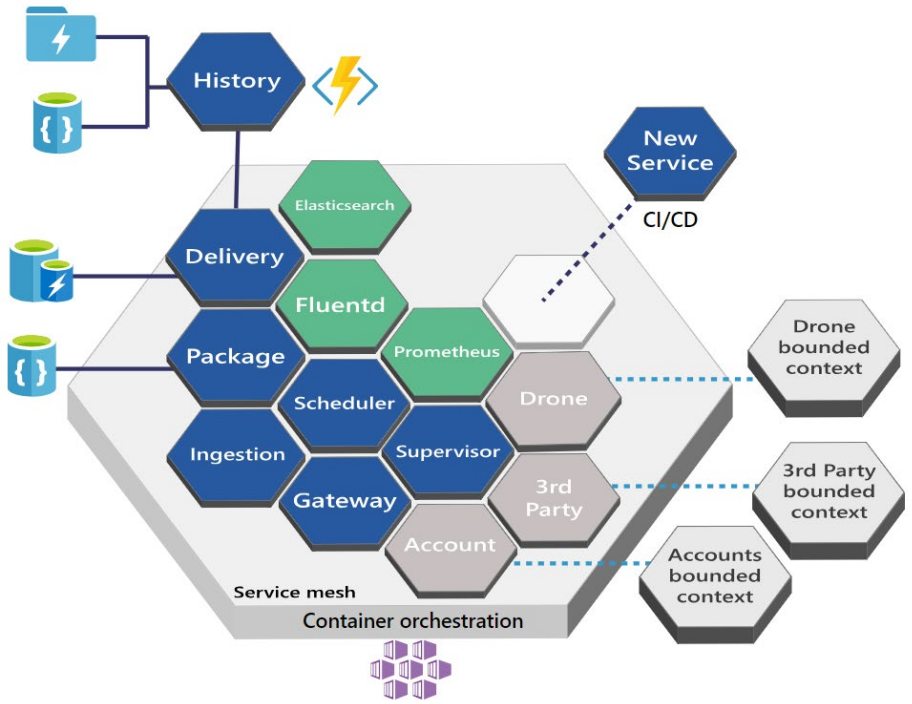
Environment configuration tests

- More moving components
- Infrastructure services are exposed as SaaS
- Other services may be resolved via service
- Requires preparation for test data



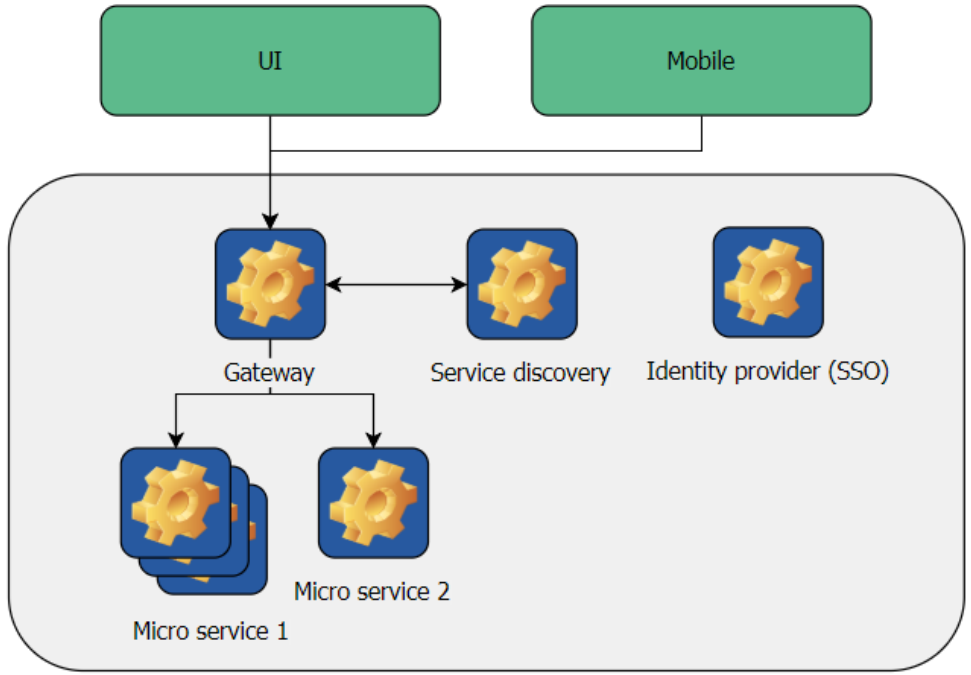
Tracing capability

- This needs to be solved by developers
- Identify “call stack”
- Identify bottle necks



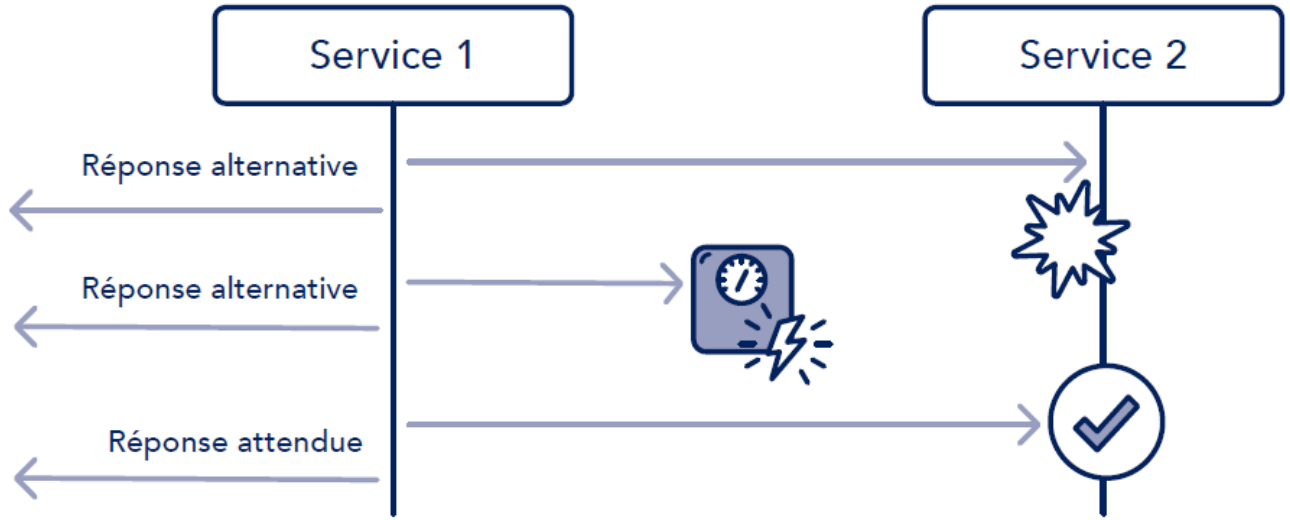
Multiple instance routing

- Verify that balancing works
- Verify multi instance behavior
- Verify limiting



Graceful degradation

- More test cases involved
- Requires specific setup



Developpez.com

Tester skill set expansion

- Requires infrastructure knowledge
- Requires deeper application architecture knowledge
- Requires technical knowledge diversity
- Requires higher automation skills

Does microservices mean micro testing?

Q&A

Thank you