ARCHITECTING FOR CONTINUOUS DELIVERY

Ken Mugrage – ThoughtWorks Technology Advocate

@kmugrage
CONWAY’S LAW

Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.
“...It also brings them into day-to-day contact with the customer. This customer feedback loop is essential for improving the quality of the service.”
CODE MANAGEMENT
TRUNK BASED DEVELOPMENT

@kmugrage

https://martinfowler.com/bliki/FeatureBranch.html

https://go.cd.org/
TRUNK BASED DEVELOPMENT

@kmugrage

https://martinfowler.com/bliki/FeatureBranch.html

https://gocd.org/
TRUNK BASED DEVELOPMENT

https://martinfowler.com/bliki/FeatureBranch.html
https://gocd.org/
The purpose of a Continuous Delivery pipeline is to kill release candidates
FEATURE TOGGLIES

function reticulateSplines()
{
    // current implementation lives here
}

@kmugrage  https://martinfowler.com/articles/feature-toggles.html by Pete Hodgson  https://gocd.org/
function reticulateSplines() {
    var useNewAlgorithm = false;
    // useNewAlgorithm = true; // UNCOMMENT IF YOU ARE WORKING ON THE NEW SR ALGORITHM

    if (useNewAlgorithm) {
        return enhancedSplineReticulation();
    } else {
        return oldFashionedSplineReticulation();
    }
}

function oldFashionedSplineReticulation() {
    // current implementation lives here
}

function enhancedSplineReticulation() {
    // TODO: implement better SR algorithm
}
FEATURE TOGGLES

we put toggle points in our code to switch behavior for the new feature.

several toggle points for the same feature use a single toggle router to determine their state.

the toggle router may need to consider toggle context (e.g., which user is making the request) in order to make a routing decision.

a toggle router is controlled via the toggle configuration for this environment.

@kmugrage  https://martinfowler.com/articles/feature-toggles.html by Pete Hodgson  https://gocd.org/
THE PIPELINE

Build
  ▶ Functional Tests
  ▶ Regression Tests
  ▶ Deploy To Stage
  ▶ Deploy To Production

Build
  ▶ Toggles Off
  ▶ Toggles On
  ▶ Deploy To Stage
  ▶ Deploy To Production

@kmugrage
https://gocd.org/
BRANCH BY ABSTRACTION
BRANCH BY ABSTRACTION

Client Code → Abstraction Layer

Client Code → Old Supplier

Client Code

https://martinfowler.com/bliki/BranchByAbstraction.html

https://gocd.org/
BRANCH BY ABSTRACTION

Client Code → Abstraction Layer → Old Supplier

Client Code → Abstraction Layer

Client Code → Abstraction Layer
BRANCH BY ABSTRACTION

Client Code

Client Code

Abstraction Layer

Old Supplier

Abstraction Layer

New Supplier

@kmugrage

https://martinfowler.com/bliki/BranchByAbstraction.html

https://gocd.org/
BRANCH BY ABSTRACTION

Client Code

Client Code

Client Code

Abstraction Layer

New Supplier

https://martinfowler.com/bliki/BranchByAbstraction.html

@kmugrage

https://goCd.org/
ARCHITECTURE
WELL VERSIONED APIS

/api/users/v1
- ID
- Name
- Email

• Consumer 1
• Consumer 2
• Consumer 3
WELL VERSIONED APIS

/api/users/v1
- ID
- Name
- Email

/api/users/v2
- ID
- FirstName
- LastName
- Email

Consumer 1
Consumer 2
Consumer 3
WELL VERSIONED APIs

/api/users/v1
- ID
- Name
- Email

/api/users/v2
- ID
- FirstName
- LastName
- Email

Consumer 1
Consumer 2
Consumer 3
WELL VERSIONED APIS

/api/users/v2
• ID
• FirstName
• LastName
• Email

Consumer 1
Consumer 2
Consumer 3
NO SHARED DATA STORE

Service One

Service Two

DB
NO SHARED DATA STORE

Service One

DB

Service Two

DB

@kmugrage

https://go.cd.org/
EVENT DRIVEN APPLICATIONS
WITHOUT EVENTS

@kmugrage

https://martinfowler.com/articles/201701-event-driven.html

https://gocd.org/
EVENT NOTIFICATION

@kmugrage

https://martinfowler.com/articles/201701-event-driven.html

https://gocd.org/
EVENT SOURCING

• You don’t write to the data store, you create an event which writes to the store
• The test: You could completely blow away the store and recreate it from the event stream
• You use this model every day (I hope)
TESTING DISTRIBUTED APPS

• We’re using external services
• We can’t reliably test them
• Flaky tests are worse than no tests
TEST DOUBLES

On Every Continuous Integration Run

![Diagram showing Client Code connected to Test Double](https://gocd.org/)

@kmugrage
CONTRACT TESTING

On Every Continuous Integration Run

Client Code ➔ Test Double

Occasionally

Contract Test ➔ Their Test Service
CONSUMER DRIVEN CONTRACTS

• Provider runs tests which verify they won’t break consumers
• Shifts responsibility to the provider
• Acts as a communication method
TESTING WEB APPS

• Adding waits to tests makes them non-deterministic (flaky)
• Non-deterministic tests are bad
TESTING WEB APPS

- Test specifications in markdown
- Multiple language support
- Maintainable tests
- GPL v3.0

- Reliable browser automation
- Request/Response stubbing and mocking
- Smart Selectors
- MIT License

---

@kmugrage

https://gauge.org/

https://taiko.gauge.org/

https://gocd.org/
In control theory, observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs. The observability and controllability of a system are mathematical duals. The concept of observability was introduced by Hungarian-American engineer Rudolf E. Kálmán for linear dynamic systems.
applyDiscountCode(discountCode) {
  this.instrumentation.applyingDiscountCode(discountCode);

  let discount;
  try {
    discount = this.discountService.lookupDiscount(discountCode);
  } catch (error) {
    this.instrumentation.discountCodeLookupFailed(discountCode, error);
    return 0;
  }
  this.instrumentation.discountCodeLookupSucceeded(discountCode);

  const amountDiscounted = discount.applyToCart(this);
  this.instrumentation.discountApplied(discount, amountDiscounted);
  return amountDiscounted;
}
OUR ACTUAL PIPELINE
DEPLOYMENT
EXPAND / CONTRACT FOR DATABASES

DB Version 1
- ID (not null)
- Name (not null)
- Email (not null)

DB Version 2
- ID (not null)
- Name (not null)
- FirstName
- LastName
- Email (not null)

App Version 1

App Version 1

App Version 2
EXPAND / CONTRACT FOR DATABASES

**DB Version 2**
- ID (not null)
- Name (not null)
- FirstName
- LastName
- Email (not null)

**App Version 2**

```sql
select FirstName, LastName from users
If null select Name from users
Prompt user to update
Update FirstName, LastName
select FirstName, LastName from users
```
CANARY RELEASE

Release to a very small portion of our audience to “test in production”.

( Pro Tip: Test for business efficacy not just successful bits and bytes )
Release to a very small portion of our audience to “test in production”.

( Pro Tip: Test for business efficacy not just successful bits and bytes )
CANARY RELEASE

Release to a very small portion of our audience to “test in production”.

( Pro Tip: Test for business efficacy not just successful bits and bytes )
SUMMARY

• Small, well versioned pieces are best
• Small, well focused teams are best to create those
• Continuous Integration is the foundation
THANK YOU