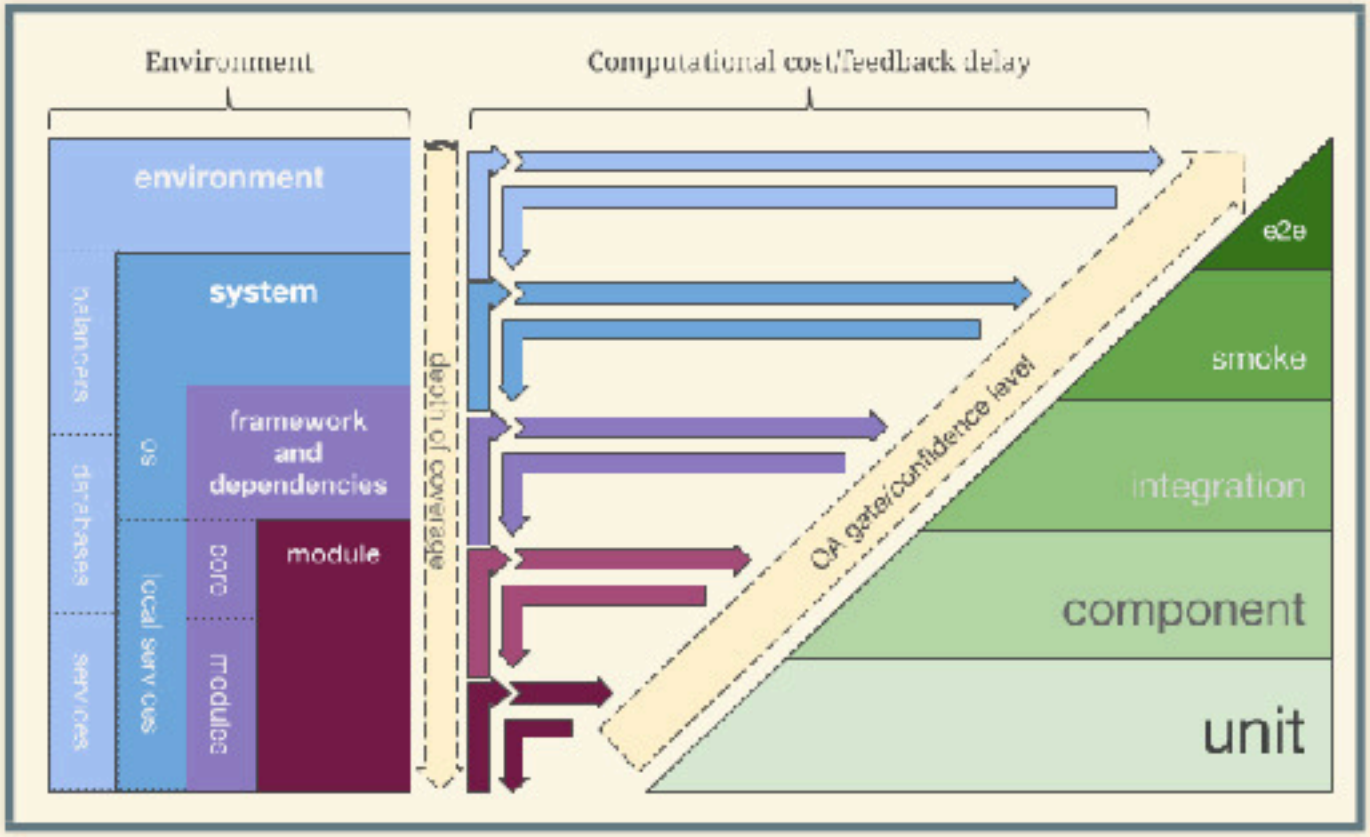
STREAMLINED SERVICE DELIVERY

WIKIMEDIA'S CONTINUOUS DELIVERY PIPELINE

WHY THE PIPELINE

- Developer empowerment
- Environment parity
- Feedback cycles
- Reproducibility and deployment confidence



Testing Pyramid

WHAT THE PIPELINE

- Promotion of application images
- Immutable progression
- Testing from the inside out

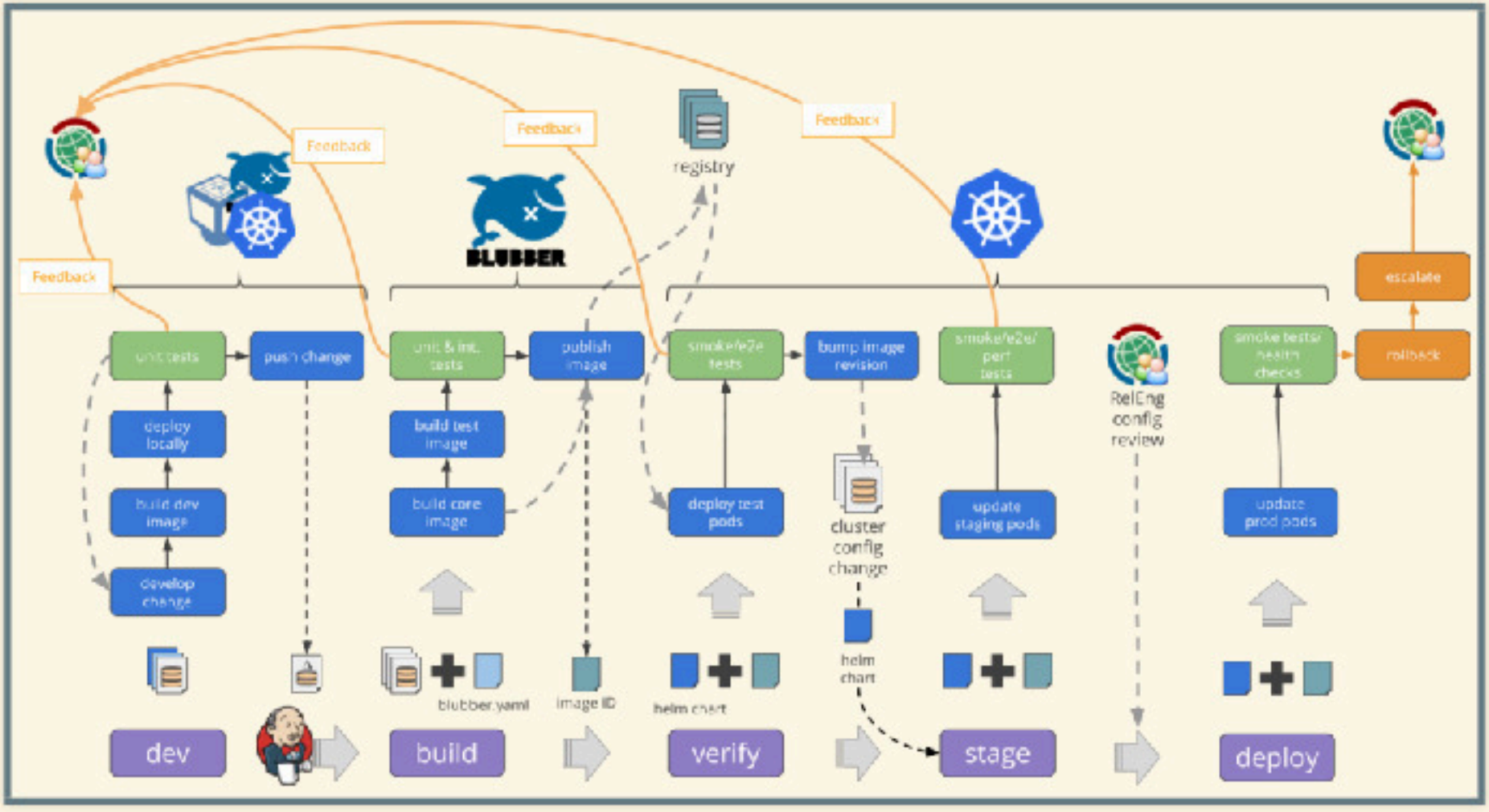
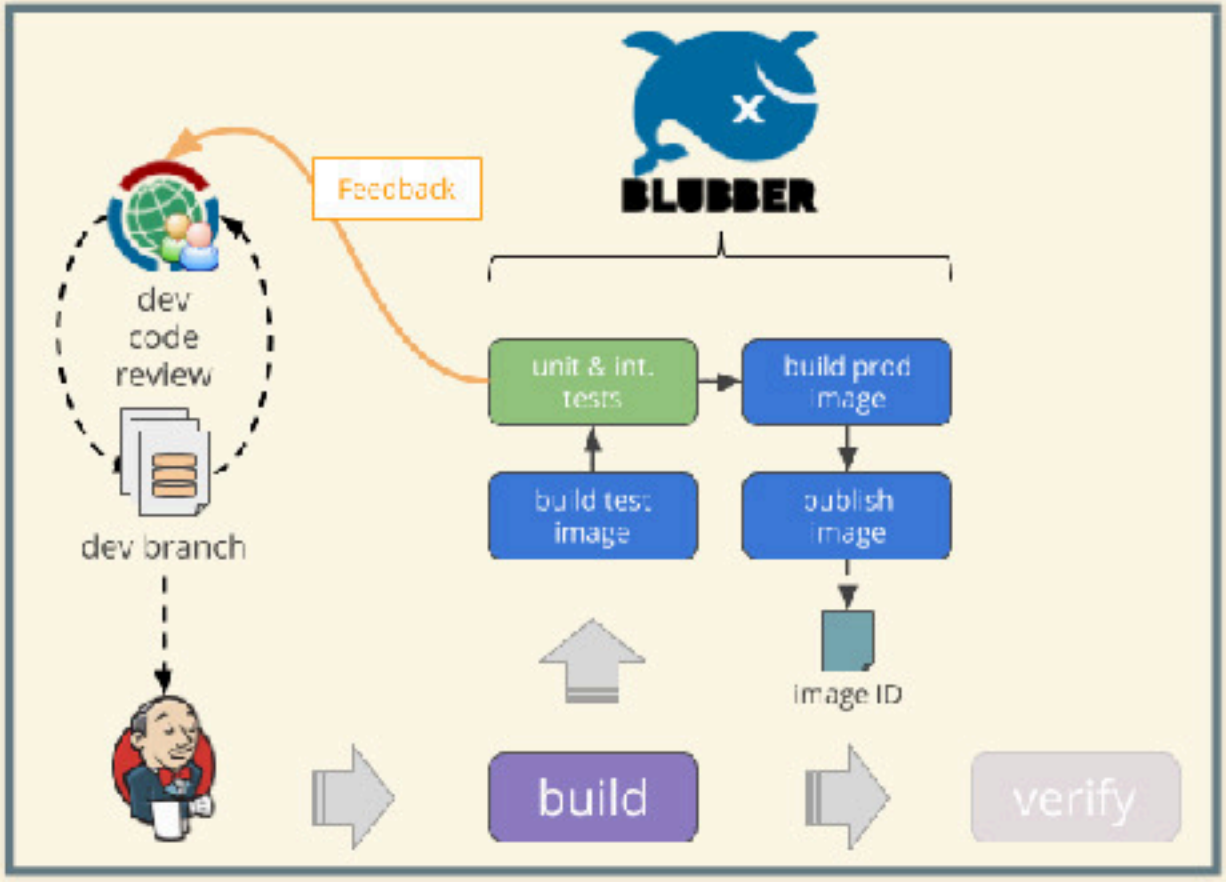

Feedback
Feedback
Feedback
Feedback
registry
BLUBBER
escalate
unit tests
push change
unit & int. tests
publish image
smoke/e2e tests
bump image revision
smoke/e2e/ perf tests
RelEng config review
smoke tests/ health checks
rollback
deploy locally
build dev image
develop change
build test image
build core image
deploy test pods
cluster config change
update staging pods
update prod pods
blubber.yaml
image ID
helm chart
helm chart
dev
build
verify
stage
deploy

IMAGE BUILDING USING BLUBBER

- Docker wrapper
- Efficiency
- Security
- Empowerment



Pipeline Build Stage

BLUBBER – EFFICIENCY

- Dockerfiles are... idiomatic? Blubber uses declarative YAML
- Blubber knows how to write cacheable instructions
- Supports multi-stage builds (smaller resulting images)

BLUBBER – SECURITY

- Enforces a security model
  - `root`: Install system packages, etc.
  - `somebody`: Owns app files, installs dependencies (e.g. npm), etc.
  - `runuser`: Runs application entry-point

BLUBBER – EMPOWERMENT

- *Developers* specify system dependencies, package managers, entrypoints in their repos
- *Developers* define their tests in their repo
- *Developers* deploy (eventually)

DEVELOPERS,
DEVELOPERS,
DEVELOPERS

blubber.yaml

```yaml
base: docker-registry.wikimedia.org/nodejs-slim
apt: { packages: [librsvg2-2] }

variants:
  build:
    base: docker-registry.wikimedia.org/nodejs-devel
    apt: { packages: [librsvg2-dev, git, build-essential] }
    node: { requirements: [package.json, package-lock.json] }
  test:
    includes: [build]
    entrypoint: [npm, test]
  production:
    copies: build
    node: { env: production }
    entrypoint: [node, server.js]
```

BLUBBER

VARIANTS – WHY?

- Parity (development -> testing -> production)
- But some things are simply different in dev/testing
- A balance is struck by *selectively* overriding

BLUBBER

VARIANTS – HOW?

```
base: docker-registry.wikimedia.org/nodejs-slim
apt: { packages: [librsvg2-2] }

variants:
  build:
    base: docker-registry.wikimedia.org/nodejs-devel
    apt: { packages: [librsvg2-dev, git, build-essential] }
    node: { requirements: [package.json, package-lock.json] }
```

BLUBBER

VARIANTS – HOW?

```
variants:
  build:
      # ...
  test:
    includes: [build]
    entrypoint: [npm, test]
  prep:
    includes: [build]
    node: { env: production }
```

BLUBBER

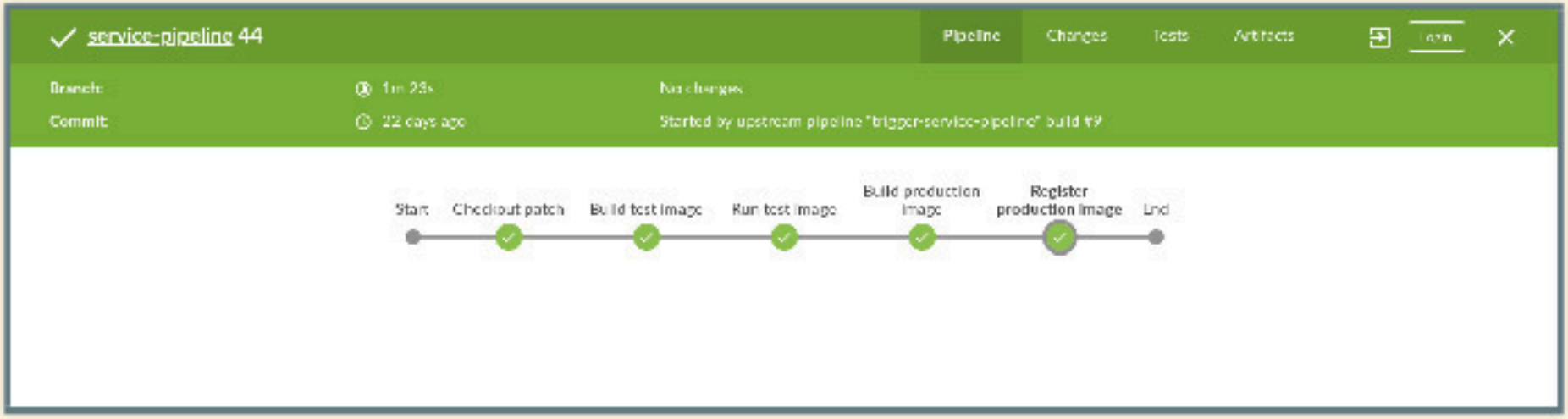
VARIANTS – HOW?

```
variants:
  # ...
  prep:
    includes: [build]
    node: { env: production }
  production:
    copies: prep
    node: { env: production }
    entrypoint: [node, server.js]
```

IMAGE BUILT

NOW WHAT?

JENKINS PIPELINE



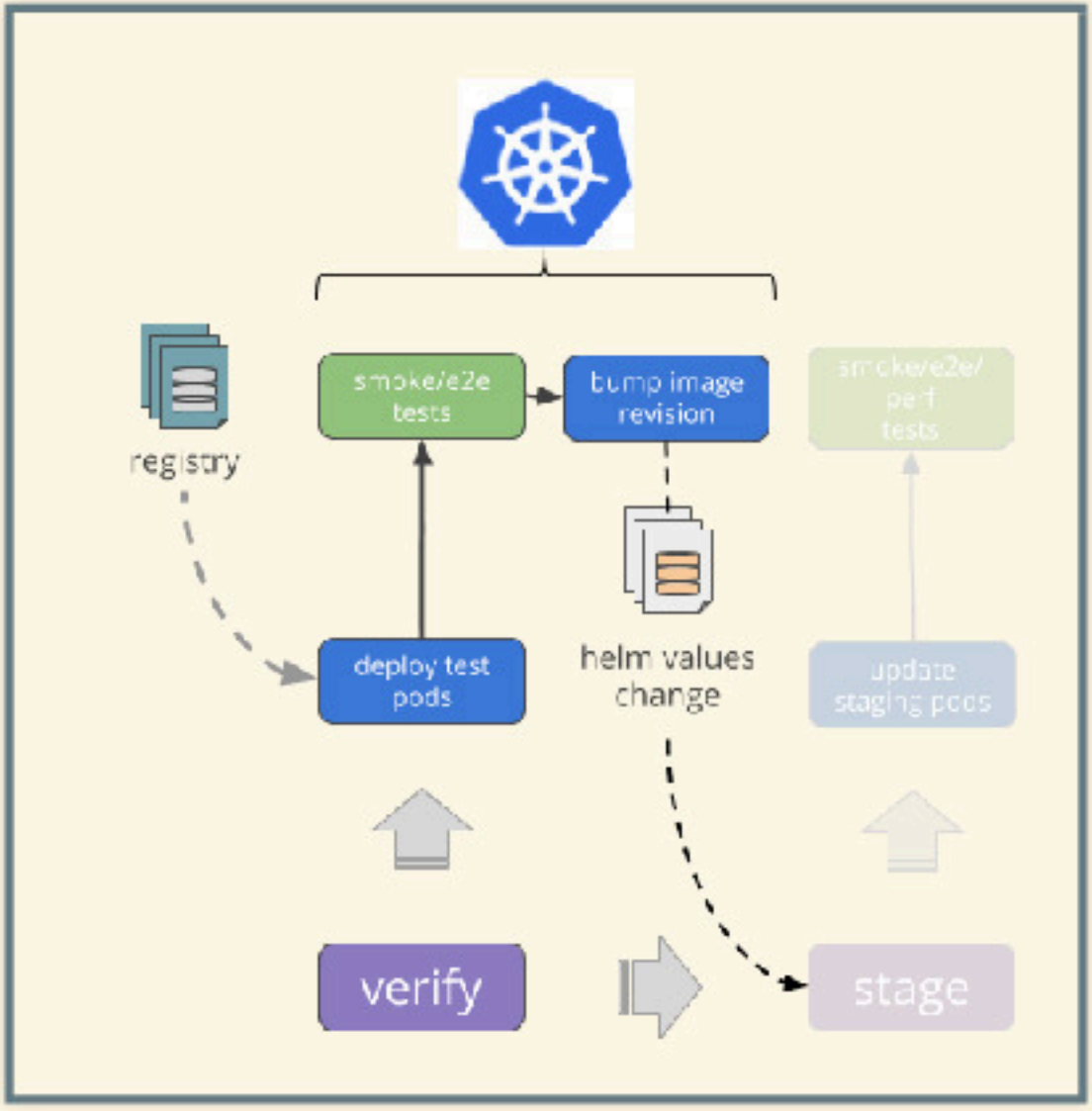
Jenkins Pipeline

JENKINS PIPELINE STEPS

1. Checkout patch from zuul and build test variant
2. Run test variant
3. Build production variant
4. Deploy production variant to minikube in CI
5. Helm test
    - service-checker-swagger in a separate pod
    - verifies endpoints in deployed application pod
6. Push production image to the registry

JENKINS PIPELINE STEPS

1. Checkout patch from zuul and build test variant
2. Build test variant and run entry point
3. Build production variant
4. **Deploy production variant to minikube in CI**
5. **Helm test**
    - **service-checker-swagger in a separate pod**
    - **verifies endpoints in deployed application pod**
6. Push production image to the registry



Pipeline Verify Stage

DOCKER PRODUCTION REGISTRY

- docker-registry.wikimedia.org
- Mathoid image currently running in production:

```
docker-registry.wikimedia.org/
wikimedia/mediawiki-services-mathoid:build-44
```

HELM

- Glues together configuration and containers
- Manages deployments as a unit
- Chart repo: releases.wikimedia.org/charts/

DEPLOYMENT TO PRODUCTION

`helm upgrade`

SLIDES AVAILABLE AT

people.wikimedia.org/~thcipriani/pipeline

BLUBBER AVAILABLE AT

people.wikimedia.org/~thcipriani/blubber