

Clever

@mohitgupta

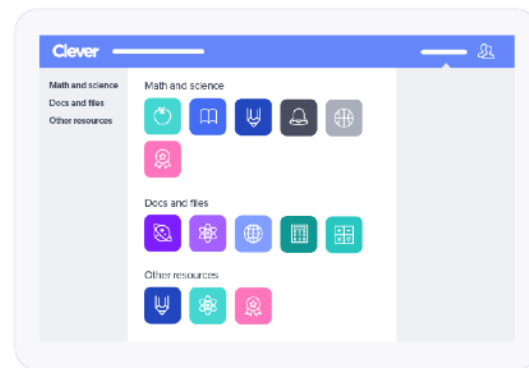
Designing Infrastructure for Rapid Iteration

Help schools get the most out of learning software



Single sign-on

Gives instant access to any app,
on any browser, and any device

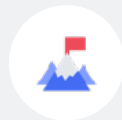


3 million

students log in daily to the Clever
Portal

Infrastructure Engineering

1. Improve resiliency
2. Increase engineering happiness
3. Unblock complex features



>120K

Containers per week



>300

Applications



~40

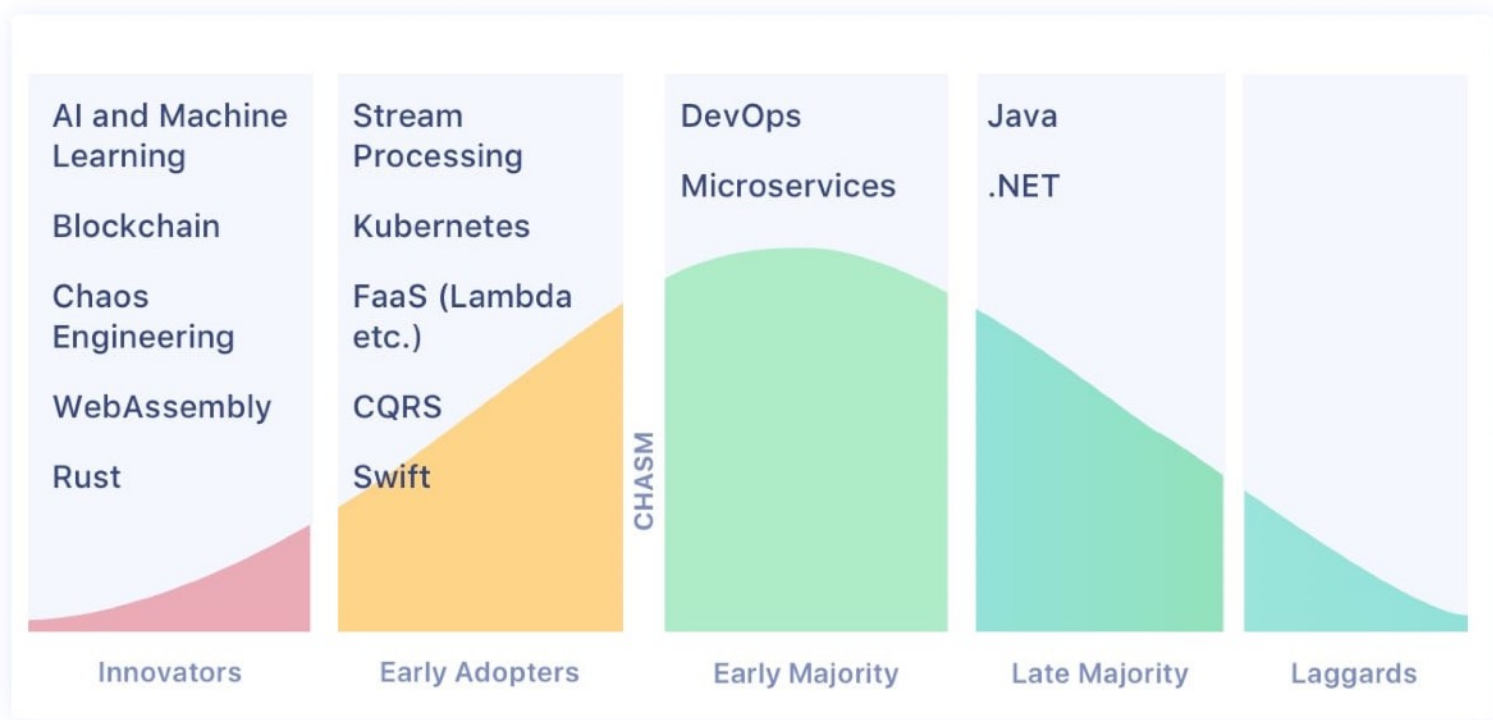
Engineers



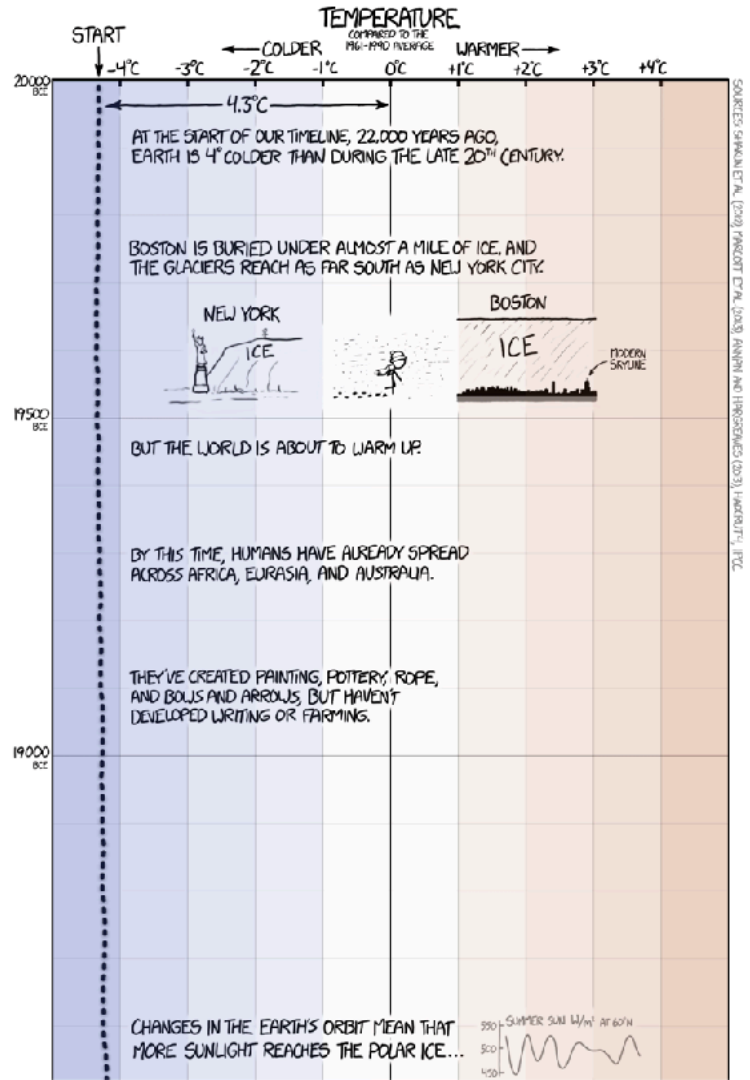
>500

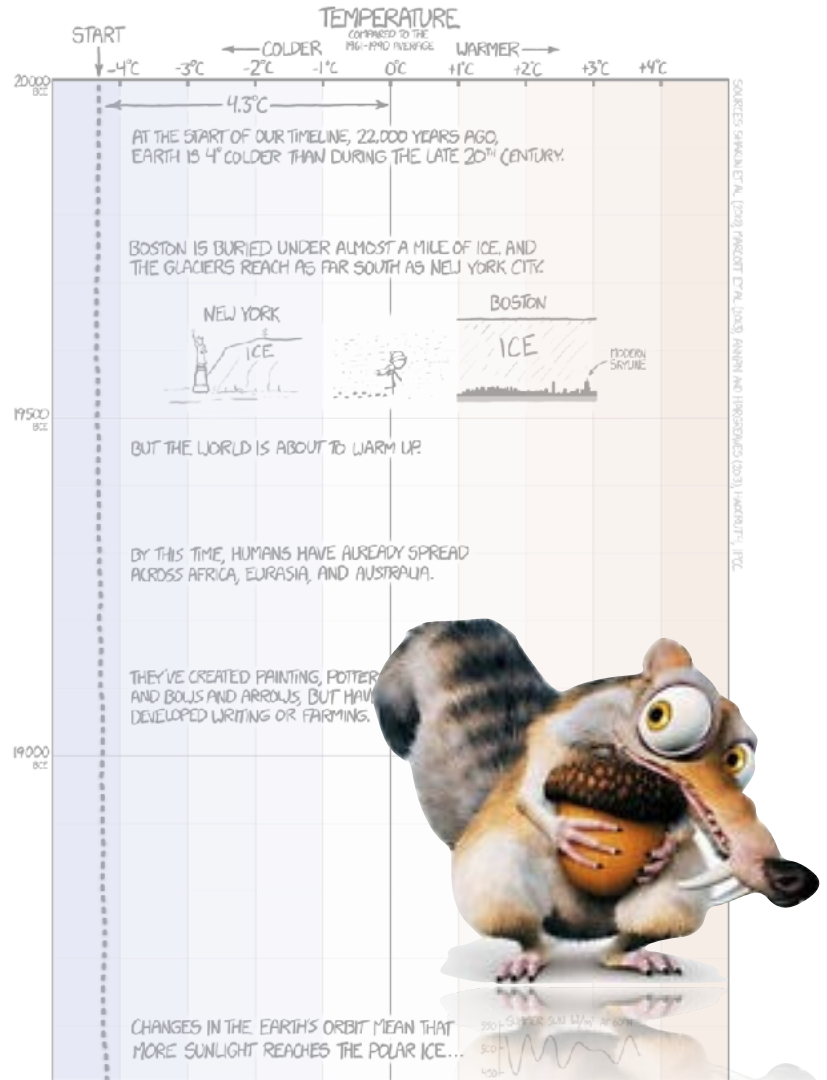
Deployments per week

Technology Adoption Curve - May 2018



This image is adapted from Geoffrey Moore's book Crossing the Chasm





credits: xkcd, 20th Century Fox



Stumbling into Containers



The Containerization Rush



Moar Engineers, Moar Problems



Building using Control Planes

Clever

*Stumbling through Containers**

** what are these containers you speak of?*

10,000

Schools use Clever



Clever



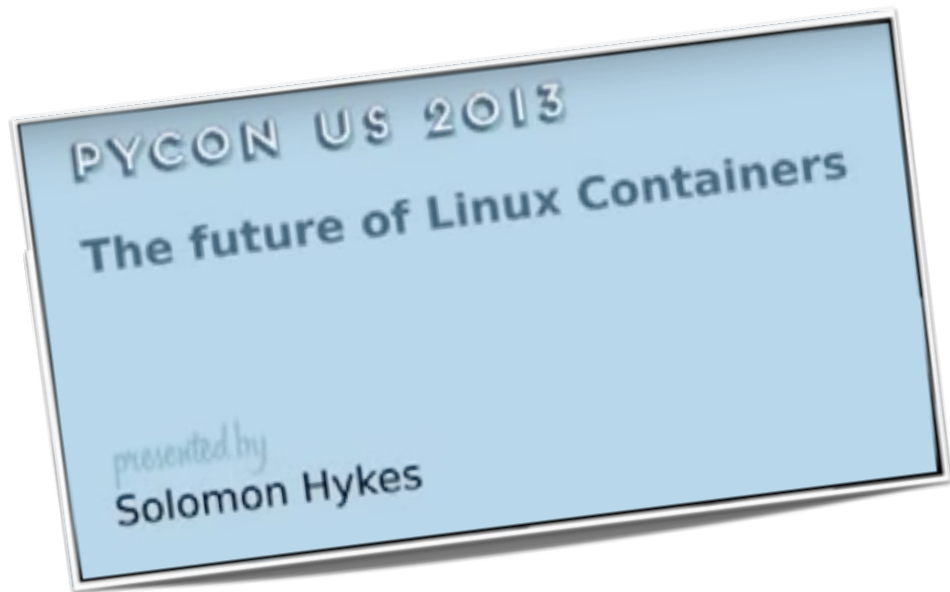
**Zero-Downtime
Rollovers**







**Blue Green
Deployments**



**Immutable Deploys
with History**



 **Shlomo Swidler** @ShlomoSwidler Mar 17, 2013
RT @botchagalupe: Docker .. This is the future of PaaS (IMHO) ow.ly/j6if1 <- of PaaS implementation, maybe. Of features, no way.

 **botchagalupe** @botchagalupe 
@ShlomoSwidler IMHO docker is to paas what chef was to laas
4 years ago
3:37 PM - Mar 17, 2013
♥ 2 👤 See botchagalupe's other Tweets 

▲ Docker Considered Harmful (catern.com)

113 points by signa11 3 hours ago | flag | hide | past | web | favorite | 77 comments



```
my-local-changes >>> docker run -H docker-1.internal.clever.com app
```

docker-1

docker-2

docker-3

docker-4



**Zero-Downtime
Rollovers**



**Blue Green
Deployments**



**Immutable Deploys
with History**

“Postpone evaluation of new tools that might fit better. Modify the current toolchain to move us closer to the ideal setup.”



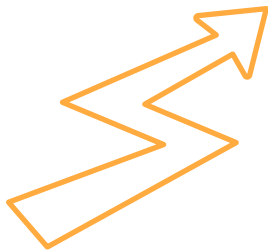
```
run:
  type: process
env:
  - PORT
  - AWS_DYNAMO_PREFIX
  - SESSION_SECRET
resources:
  cpu: 0.2
  max_mem: 0.5
expose:
  - name: http
    port: 80
    health_check:
      type: http
      path: /_healthcheck
```



app definition



```
run:  
type: docker
```



A Year of Docker at Clever

15 October 2014

Rafael Garcia
Co-founder & CTO, Clever



Move fast and *make time to think*

Clever

The Containerization Rush

** here comes everybody*



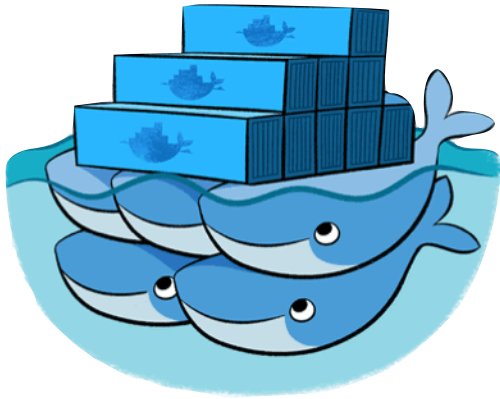
kubernetes



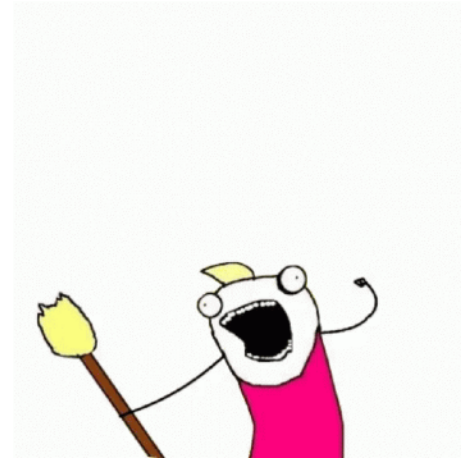
MESOS



MARATHON



Amazon ECS





credits: 20th Century Fox



Twitter used it and they were neighbors



Open Source meant we could change things



AWS was supported and we could run it

The cluster is having issues again!



Twitter had a massive
Mesos team



Nobody was an expert in
the Mesos codebase



Zookeeper!

There was progress in important areas



Fast deployments



**Improved instance
utilization**



UI and API



Move fast and test a leap of faith.

Clever

More engineers, moar problems

** onboarding forever!*

"If my code runs in production then I shouldn't have to change it for you to run it locally"





“more important to have dev stability than production velocity”

“well thought out systems don’t have such problems”

“resiliency, stability, safety !!!!”

Docker Compose: how to run everything on my laptop?

Configuration

Updated values for development in my environment

Dependencies

Current dependencies for walking through a feature

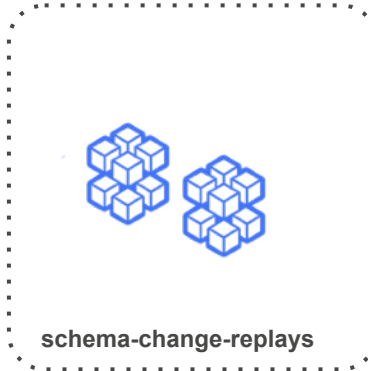
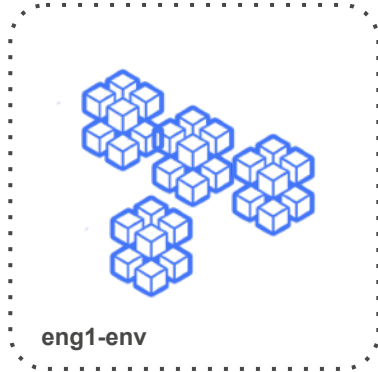
Debugging

Logs and metrics for testing outside production

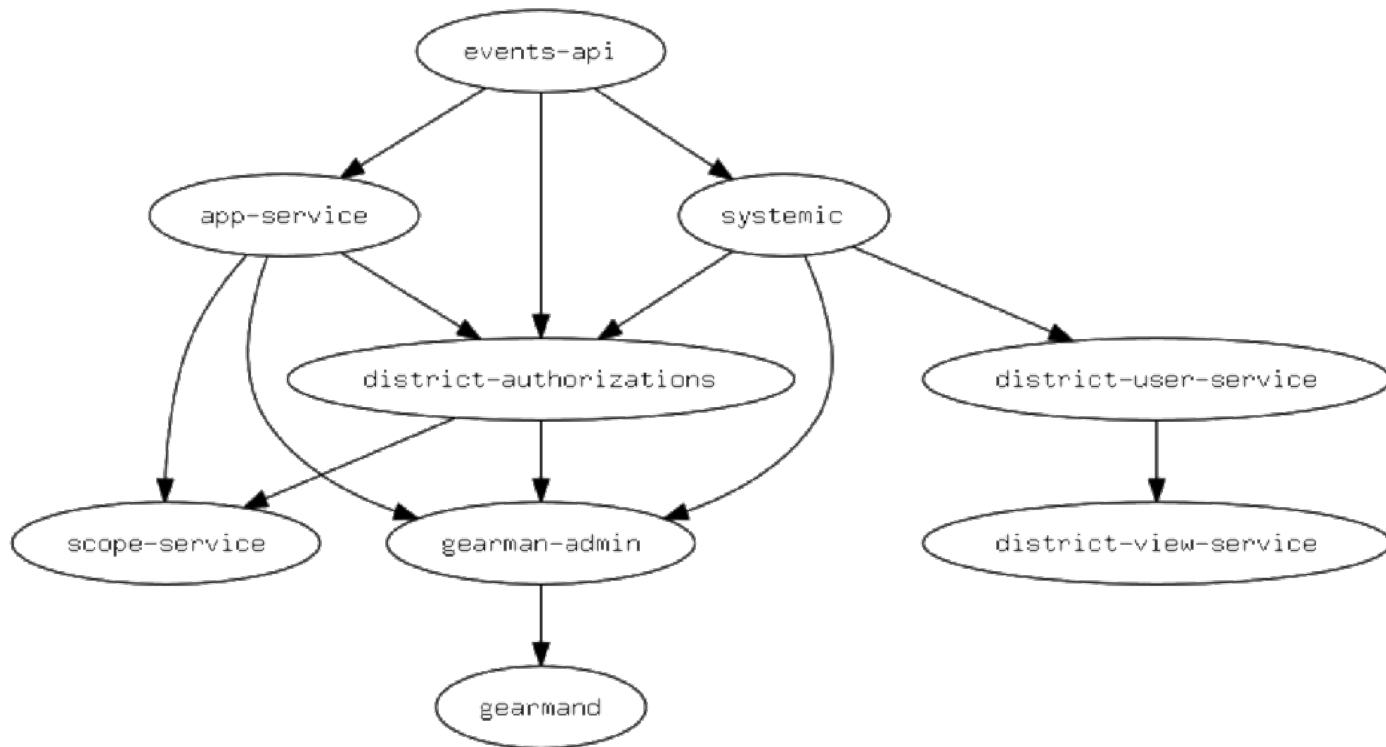


a tool to supercharge development that can
eventually be used in production

On-Demand Development Environments



```
appService, err := appServiceClient.NewFromDiscovery()
if err != nil {
    log.Fatal(fmt.Sprintf("app-service initialization failed: %s", err))
}
```



ark start —environment my-env my-app

```
run:
  type: docker
env:
  - PORT
  - AWS_DYNAMO_PREFIX
  - SESSION_SECRET
resources:
  cpu: 0.2
  max_mem: 0.5
expose:
  - name: http
    port: 80
    health_check:
      type: http
      path: /_healthcheck
external:
  host: schools.clever.com
dependencies:
  - api-router
  - district-authorizations
  - beam
team: eng-districts
shepherds:
  - cosmo.wolfe@clever.com
```

```
x ~ ark start -e c-soup events-api
```

```
Preparing for deploy...
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% CREATING c-soup environment right now! (ctrl-c to exit) %
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
□ gearmand WAITING
```

```
? district-view-service QUEUED
```

```
? gearman-admin QUEUED
```

```
? scope-service QUEUED
```

```
? district-user-service QUEUED
```

```
? district-authorizations QUEUED
```

```
? systemic QUEUED
```

```
? app-service QUEUED
```

```
? events-api QUEUED
```

```
Deploying dependencies for events-api...
```

```
x ~ ark start -e c-soup events-api
```

```
Preparing for deploy...
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% CREATING c-soup environment right now! (ctrl-c to exit) %
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

- gearmand WAITING
- ? district-view-service QUEUED
- ? gearman-admin QUEUED
- ? scope-service QUEUED
- ? district-user-service QUEUED
- ? district-authorizations QUEUED
- ? systemic QUEUED
- ? app-service QUEUED
- ? events-api QUEUED

engineers are users too.

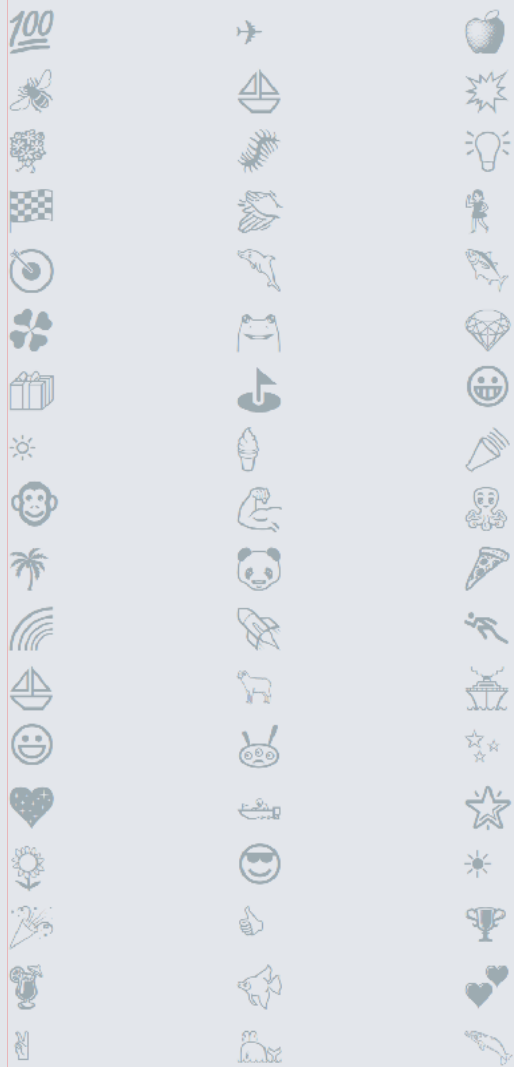
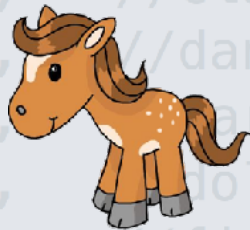
```
Deploying dependencies for events-api...
```





start with thin wrappers


find your own defaults

highlight your quirks




 **Compute**
EC2
EC2 Container Service
Lightsail [↗](#)
Elastic Beanstalk
Lambda
Batch

 **Developer Tools**
CodeStar
CodeCommit
CodeBuild
CodeDeploy
CodePipeline
X-Ray


 **Analytics**
Athena
EMR
CloudSearch
Elasticsearch Service
Kinesis
Data Pipeline
QuickSight [↗](#)


 **Application Services**
Step Functions
SWF
API Gateway
Elastic Transcoder

 **Storage**
S3
EFS
Glacier
Storage Gateway


 **Management Tools**
CloudWatch
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Trusted Advisor
Managed Services


 **Artificial Intelligence**
Lex
Polly
Rekognition
Machine Learning


 **Messaging**
Simple Queue Service
Simple Notification Service
SES

 **Database**
RDS
DynamoDB
ElastiCache
Redshift

 **Internet Of Things**
AWS IoT
AWS Greengrass


 **Business Productivity**
WorkDocs
WorkMail
Amazon Chime [↗](#)

 **Networking & Content Delivery**
VPC
CloudFront
Direct Connect
Route 53


 **Security, Identity, Compliance**
IAM
Inspector
Certificate Manager
Directory Service
WAF & Shield
Compliance Reports

 **Contact Center**
Amazon Connect

 **Desktop & App Streaming**
WorkSpaces
AppStream 2.0

 **Migration**
Application Discovery Service
DMS
Server Migration
Snowball

 **Game Development**
Amazon GameLift

 **Mobile Services**
Mobile Hub
Cognito
Device Farm
Mobile Analytics
Pinpoint

outsource complexity when possible

[↔ Code](#)

find documentation about everything that Infra provides: deploying, monitoring, alerting, analytics.

 **412** commits

 **30** contributors

invest in training and documentation

Clever

Changing all the time

* how we learnt to stop worrying and embrace new tools

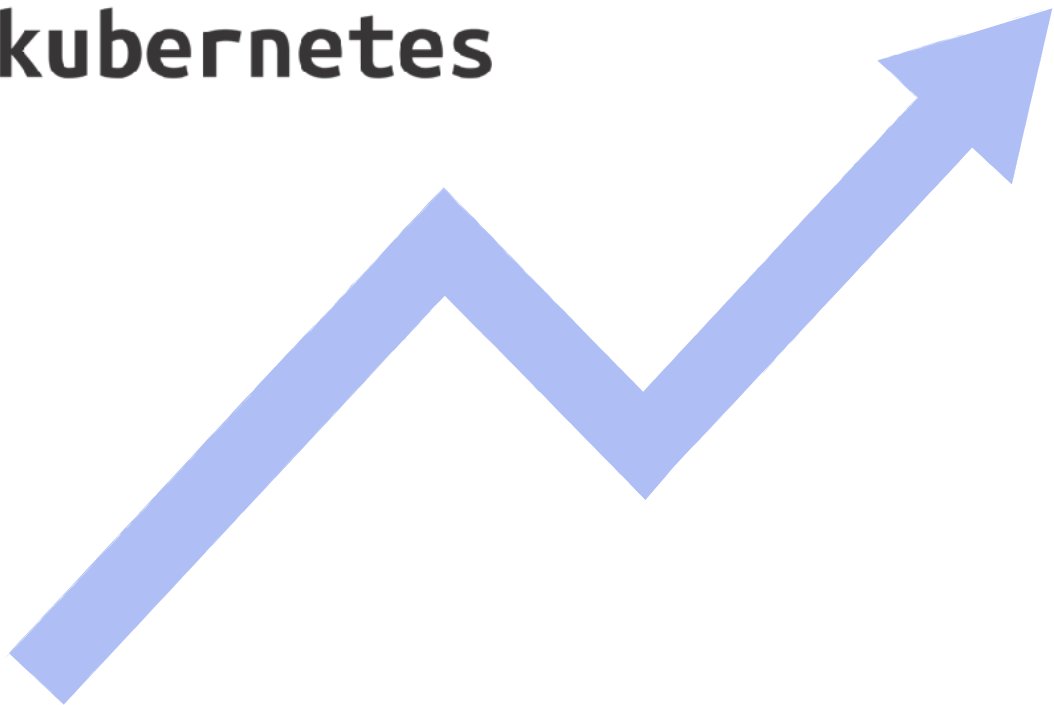
Clever



kubernetes



Amazon ECS





**Integration with existing
tools we used**



**Completely managed by
AWS**



Met all our requirements!



Canary Deployments



**Cross-cluster
Environments**



ARK



Daemon Containers



Cluster Autoscaling

Asynchronous Workflows at Clever

- REST API and Web UI for managing workflows
- More than a million executions a week
- Over 100 asynchronous workers

The screenshot displays the 'Workflows' management interface. On the left, a sidebar contains navigation options: 'Workflows' (selected), 'Manual Runs', 'Repairs', and 'Resources'. The main content area is titled 'Workflows' and shows a summary of 'Active' workflows: 1,514 total, with a small number of failed ones. Below this is a filter section with a dropdown menu set to 'FILTER BY STATUS' and a 'DATE RANGE' of 'Nov 17, 11:54:21 AM — Nov 17, 12:25:59 PM'. The main part of the interface is a table listing individual workflow runs. The table has four columns: 'Created +', 'Workflow ID', 'App ID', and 'District ID'. Each row represents a workflow run, including its creation time (all from Nov 17, 2017, at 12:25:53 PM or 12:25:54 PM), a unique Workflow ID, the App ID (Benchmark Education Company), and the District ID. The table also includes status icons (red circles with exclamation marks) and a pagination bar at the bottom showing 'Prev', '1', and 'Next'.

Created +	Workflow ID	App ID	District ID
Nov 17, 2017, 12:25:53 PM 7 days ago	3a5894c9-22c0-4648-bf63-a32313814e1c	Benchmark Education Company	57ee8dc6-2998010000002f
Nov 17, 2017, 12:25:53 PM 7 days ago	e2244706-d23e-477a-87a8-5846462851aa	Benchmark Education Company	53f64e2c71a6f3340d000b45
Nov 17, 2017, 12:25:54 PM 7 days ago	affca9c4-3e06-4ded-9f98-674c96a7a8b0	Benchmark Education Company	58cc61ee2234d0008e12fc
Nov 17, 2017, 12:25:54 PM 7 days ago	074f38e4-86f3-4d27-869b-041df3e99c04	Benchmark Education Company	5363fb123b39631290000cc
Nov 17, 2017, 12:25:54 PM 7 days ago	33d9b278-9222-471c-832a-a0265ea70969	Benchmark Education Company	51a27f954f4682280d00458f

```
// WorkflowManager is the interface for creating, stopping and checking status for Workflows
type WorkflowManager interface {
    CreateWorkflow(ctx context.Context, def models.WorkflowDefinition, input string, name
    RetryWorkflow(ctx context.Context, workflow models.Workflow, startAt, input string) (
    CancelWorkflow(ctx context.Context, workflow *models.Workflow, reason string) error
    UpdateWorkflowSummary(ctx context.Context, workflow *models.Workflow) error
    UpdateWorkflowHistory(ctx context.Context, workflow *models.Workflow) error
}
```


Built using AWS Batch



Fully Managed

No software to install or servers to manage. AWS Batch provisions, manages, and scales your infrastructure



Integrated with AWS

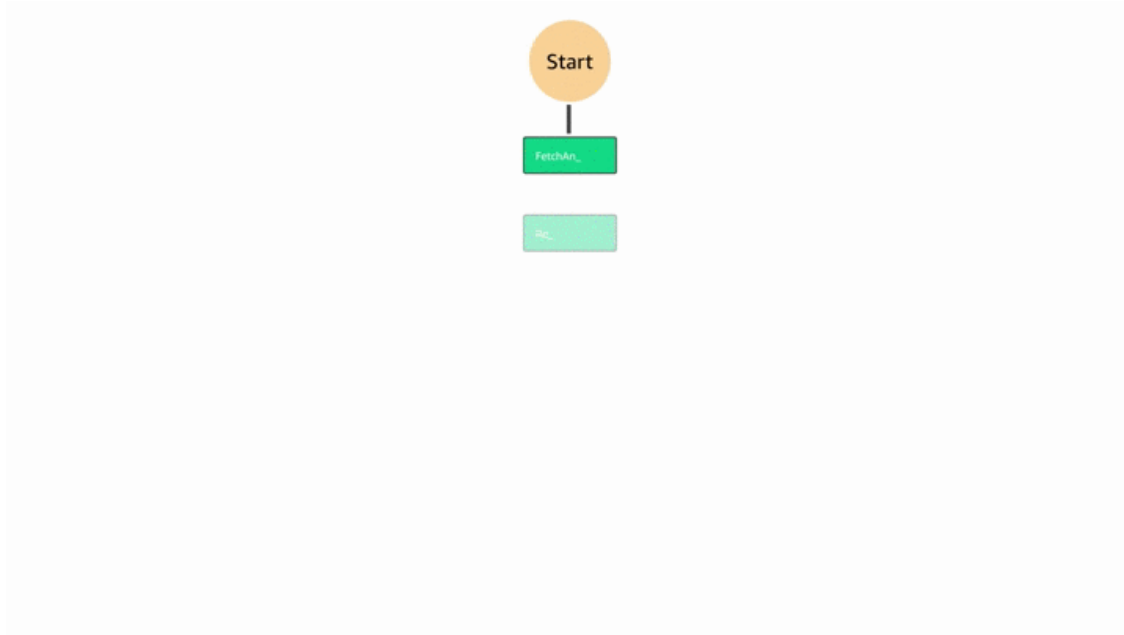
Natively integrated with the AWS Platform, AWS Batch jobs can easily and securely interact with services such as Amazon S3, DynamoDB, and Rekognition



Cost-optimized Resource Provisioning

AWS Batch automatically provisions compute resources tailored to the needs of your jobs using Amazon EC2 and EC2 Spot

Built using AWS Step Functions



Control planes allow for rapid infrastructure experimentation

Clever

Questions



@mohitgupta



github.com/Clever