



HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Building Blockchain as a Service via Hyperledger Cello

Speakers: Henry Zhang, Tong Li

Cello maintainers: Baohua Yang, Haitao Yue, Jiahao Chen

June 26, 2019

About Us

Henry Zhang

- Chief Architect, VMware R&D China
- Founder and evangelist of Project Harbor
- Hyperledger Cello contributor
- Co-author of blockchain book
 - Blockchain Technical Guide (in Chinese)
- Current interest: cloud computing, AI, blockchain etc.

Tong Li

- @email4tong
- Senior Software Engineer at IBM
- Developer advocate
- Focus on cloud, container and blockchain technology development in Open Source community
- Software application deployment and automation enthusiastic supporter
- Hyperledger Fabric Cello committer

Contents

- What
- Key features
- Architecture and Design
- Roadmap
- How to Contribute
- Q&A

Contents

- What
- Key features
- Architecture and Design
- Roadmap
- How to Contribute
- Q&A

What is Cello

Blockchain Operating System

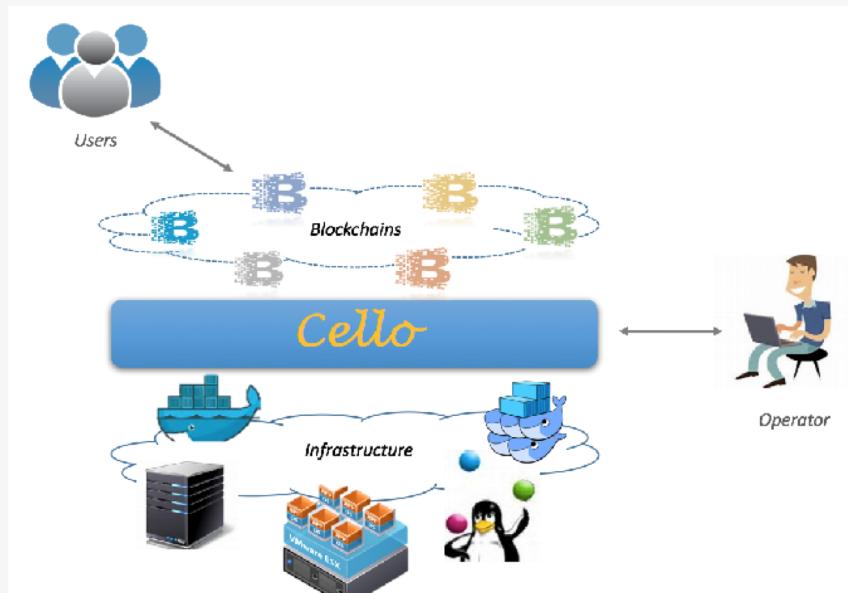
- Resolve challenges to provision and manage blockchain networks
- Allow users to focus on application development
- Monitor and analyze the network activities in multi-levels
- E.g., Hyperledger Fabric is a blockchain kernel



What is Cello

Cello offers

- Quickly provision and manage blockchain networks with dashboard
- Support various computation resource from servers to containers
- Monitor chain health status automatically
- Analyze the running status/log/data for chains



What is Cello

Brief History

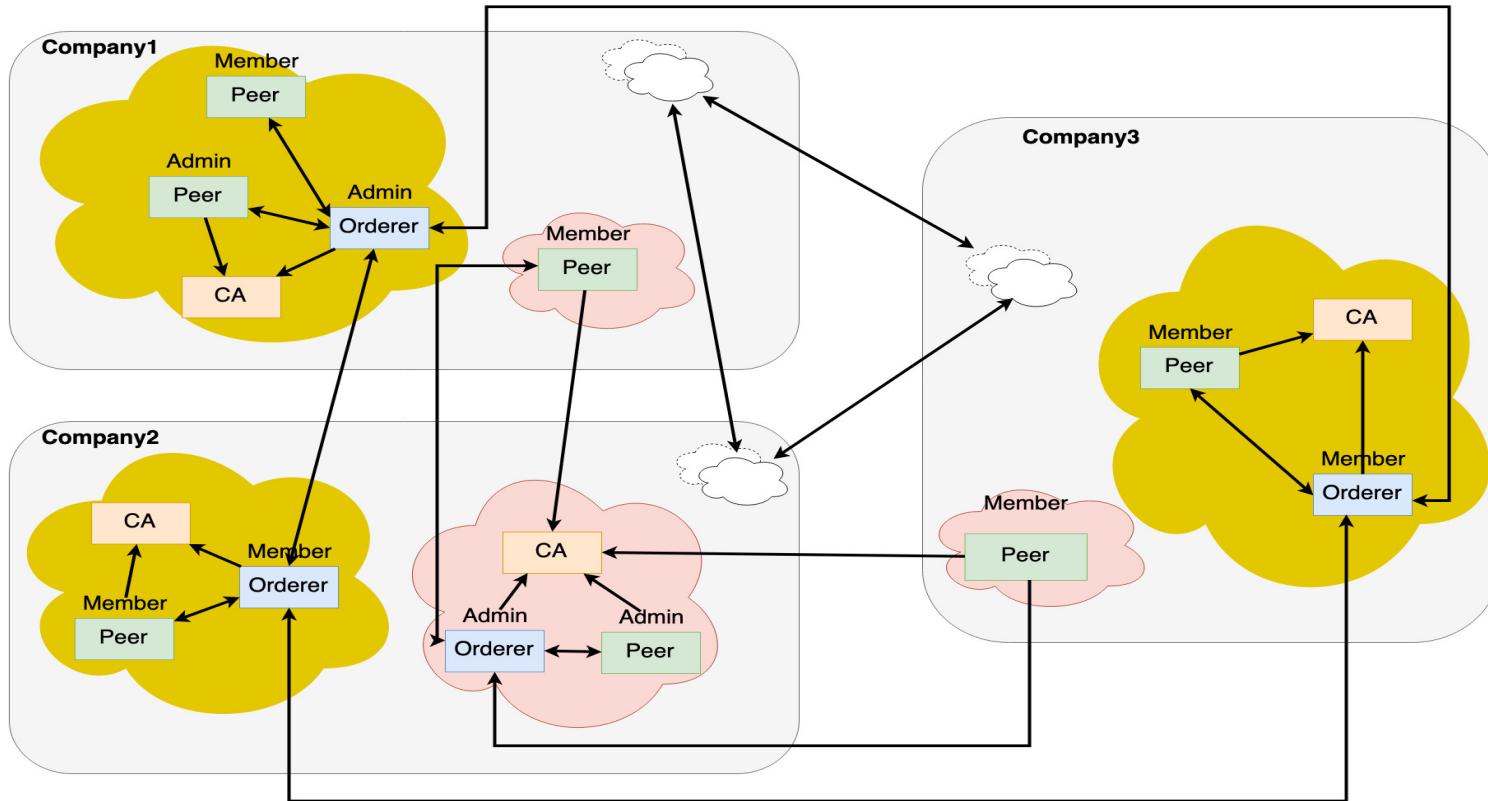
- Accepted as a Hyperledger top project at Jan, 2017
- Supported and contributed by developers from IBM, Oracle, VMware, Cloudsoft, H3C, etc.
- 600+ commits
- Implemented mainly with Python, Golang, Javascript (lightweight!)
- Take advantage of open source tools like Ansible, Kubernetes, Helm Charts

Contents

- What
- Key features
- Architecture and Design
- Roadmap
- How to Contribute
- Q&A

Topology

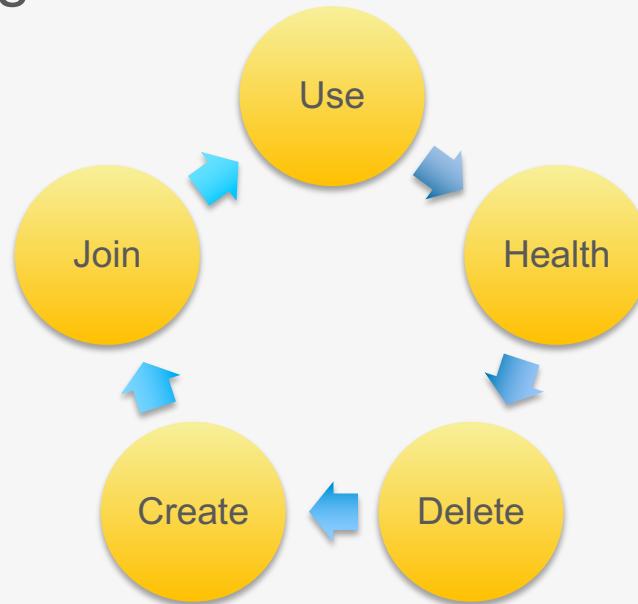
 Network1
 Network2



Key features

Manage the lifecycle of blockchains

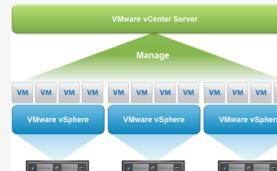
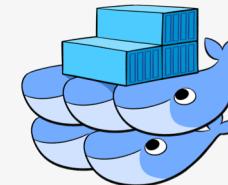
- Create nodes
- Join a network
- Use a network
- Health check
- Delete



Key features

Support Various Infrastructure

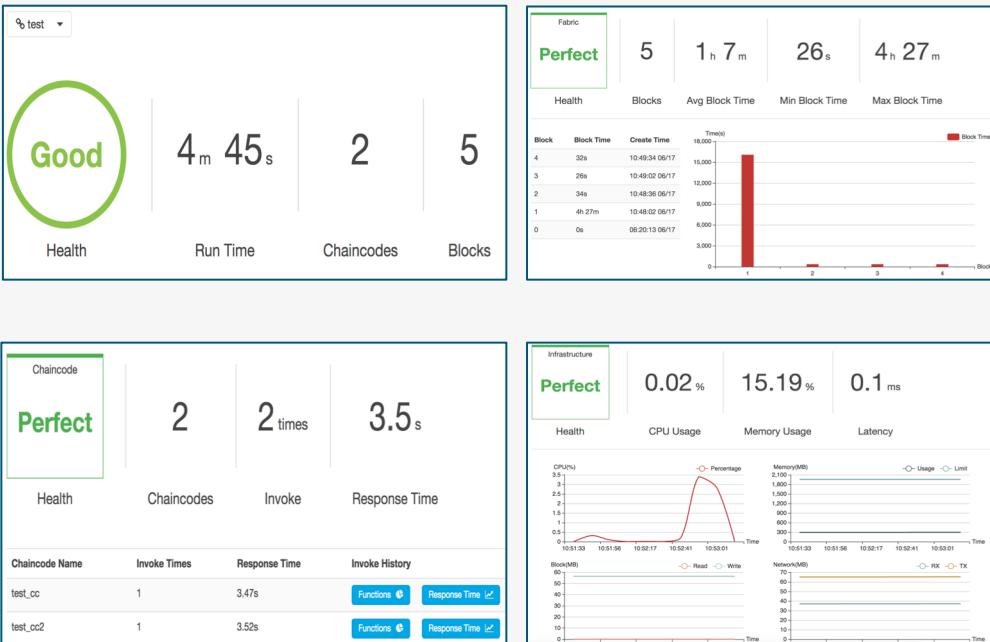
- Servers and VMs (vSphere)
- Docker Host and Swarm
- Kubernetes
- Ansible
- And More...



Key features

Support Monitor & Analysis

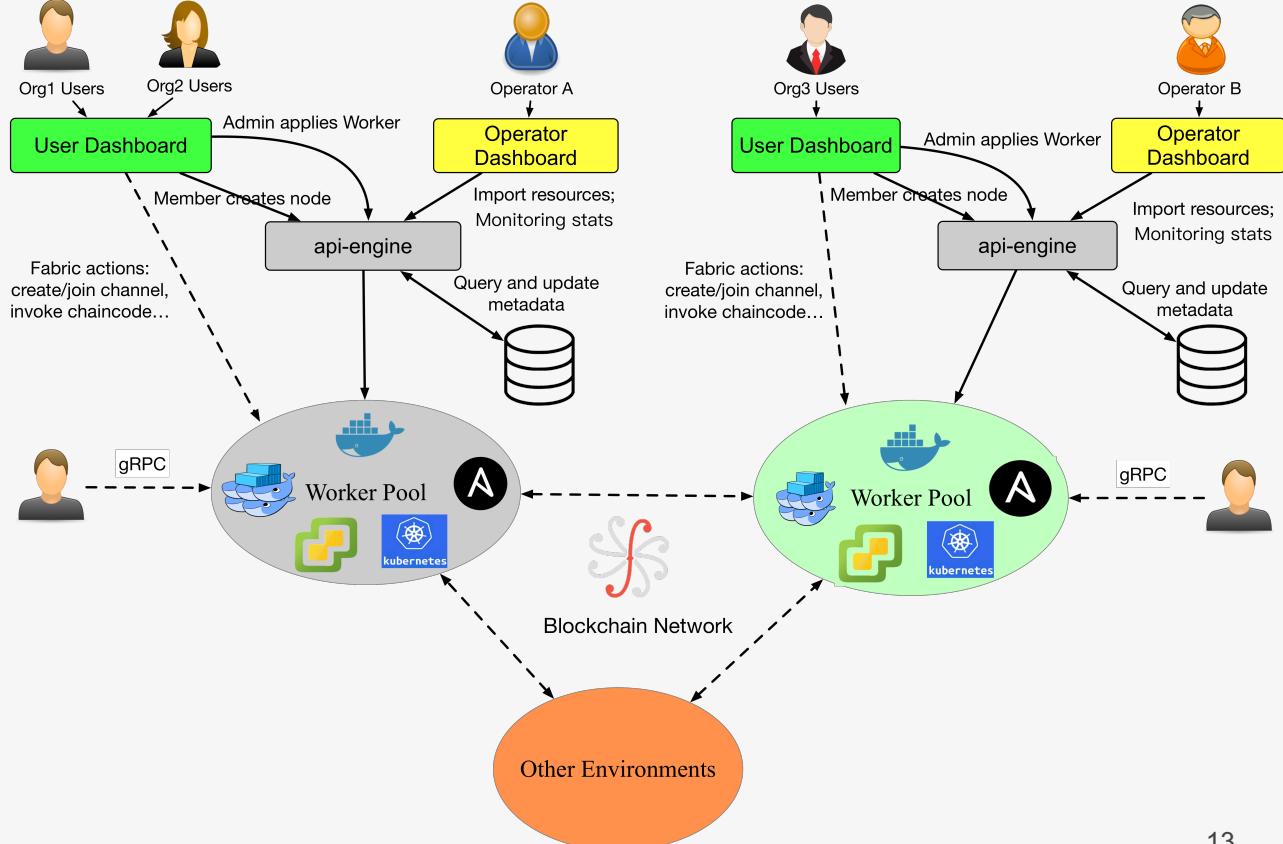
- Chain Status
- System Utilization
- Network Latency
- Log messages
- Chaincode Operation Analysis



Key features

Distributed and Dedicated Deployment

- Consortium with other users
- Manage local blockchain networks



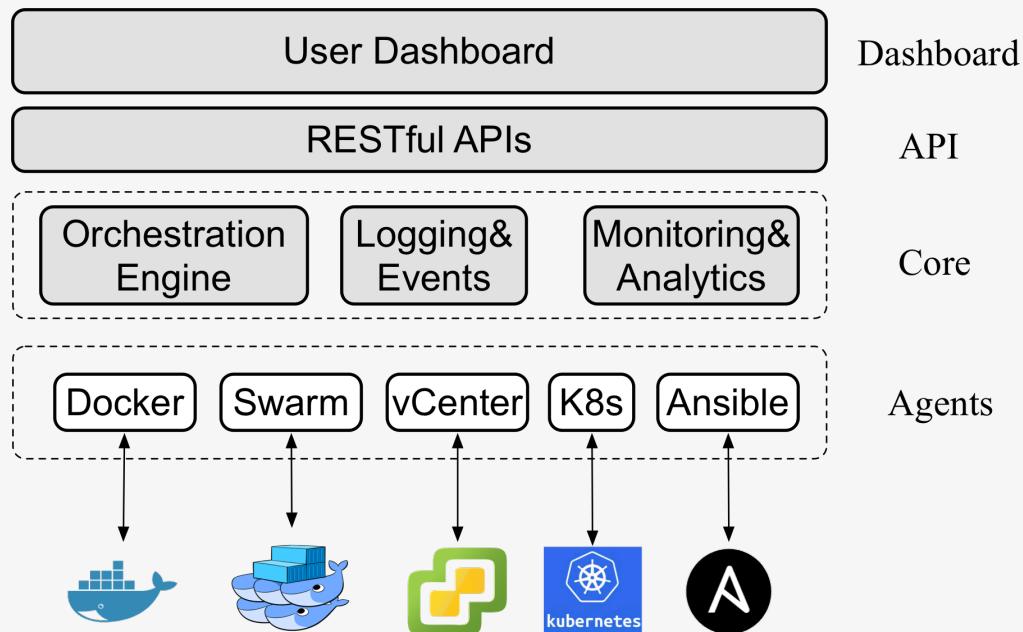
Contents

- What
- Key features
- **Architecture and Design**
- Roadmap
- How to Contribute
- Q&A

Architecture and Design

4 Layers

- Dashboard: Web frontend
- API: RESTful APIs
- Core: Orchestration implementation
- Agent: Drivers for various infrastructures



Architecture and Design

Dashboard layer

- Role Based Access Control (RBAC)
- View of user role : Blockchain usage
- View of operator or admin role : Resource management

Architecture and Design

API layer

- User management: register, login, create users...
- Organization management: create, update, generate credentials ...
- Resource management: add nodes, add agents, monitor usage...
- Blockchain lifecycle management: create network, start nodes, join channels, deploy chaincode, send transactions ...

Architecture and Design

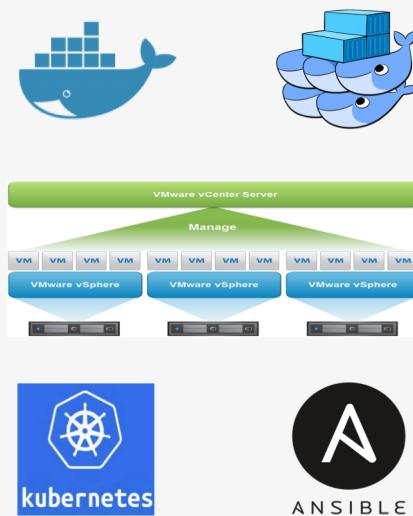
Core layer

- Orchestrate the blockchain networks
- Collect log, event, running status data
- Resource and agent management
- Cello-analytics will do analytics on the data

Architecture and Design

Agent layer (pluggable)

- In Cello, each type of resource cluster is a “Worker”
- Docker/Swarm: standard Docker/Swarm API
- vSphere: standard vSphere API
- Kubernetes: Kubernetes API
- Ansible: Ansible API



Ansible agent of Cello

Use Cello, one can deploy fabric in many different ways. If you like to stand up fabric on Kubernetes, Cello ansible agent is the tool to help you stand up a fabric network within few minutes.

The following few slides explain the two steps to stand up a fabric network onto K8S. We will be using cello ansible agent to deploy Hyperledger Fabric 1.4.1 using Raft onto Kubernetes

Deploy a fabric network onto K8S

How do I prepare?

```
ubuntu@u1710:~/ansible-agent$ pwd
/home/ubuntu/ansible-agent
ubuntu@u1710:~/ansible-agent$ tree
.
└── vars
    ├── ca-dal13-tong-hfrd.pem
    ├── kubeconfig
    ├── networkspec.yml
    └── resource.yml
```

Deploy a fabric network onto K8S What do I do?

```
docker run --rm \
    -v $(pwd)/vars:/opt/agent/vars \
    hyperledger/cello-ansible-agent \
    ansible-playbook \
    -e "mode=apply env=networkspec" \
    setupfabric.yml
```

Deploy a fabric network onto K8S

What is the magic in network spec file?

```
network:
  fabricnet:
    orderers: [orderer0.ordererorg, orderer1.ordererorg, orderer2.ordererorg]
    peers: [worker@peer1.org0, worker@peer2.org0, worker@peer1.org1,
            worker@peer2.org1, worker@peer1.org2, worker@peer2.org2]

  baseimage_tag: "1.4.1"
  helper_tag: "1.4.1"
  ca:
    admin: "admin"
    adminpw: "adminpw"
    image_tag: "1.4.1"

repo:
  url: "hyperledger/"
  username: ""
  password: ""
```

Deploy a fabric network onto K8S

Magic in network spec file continued

```
  fabric:  
    peer_db: "goleveldb"  
    tls: true  
    consensus_type: "etcdraft"  
    generate_certs: True  
    logging_level: "ERROR"  
    metrics: false  
    k8s:  
      cpu_limit: '6'  
      cpu_req: '1'  
      exposeserviceport: true  
      mem_limit: 8Gi  
      mem_req: 1Gi  
      shared_storage: false  
      storagecapacity: 20Gi  
      storageclass: ibmc-file-gold
```

Deploy a fabric network onto K8S

What just happened in K8S cluster?

- K8S services
 - One service per peer, per orderer
- K8S StatefulSet
 - One stateful set per peer, per orderer node
- K8S persistent volume claims
 - One persistent volume claim shared across all the pods
- Fabric network
 - A test channel named “firstchannel” created
 - All peers join that test channel
 - A sample chaincode installed and instantiated.

Deploy a fabric network onto K8S

What do I do after I am done with it?

```
docker run --rm \
-v $(pwd)/vars:/opt/agent/vars \
hyperledger/cello-ansible-agent \
ansible-playbook \
-e "mode=destroy env=networkspec" \
setupfabric.yml
```

Contents

- What
- Key features
- Architecture and Design
- **Roadmap**
- How to Contribute
- Q&A

Roadmap

v0.6.0

Support Fabric v0.6
 Support Docker Host
 Support Swarm
 Operational Dashboard
 System Monitoring

v0.7.0

Support Fabric v1.0
 Support Ansible agent
 User Dashboard and API
 Start vSphere & Kubernetes Agent support

v0.8.0

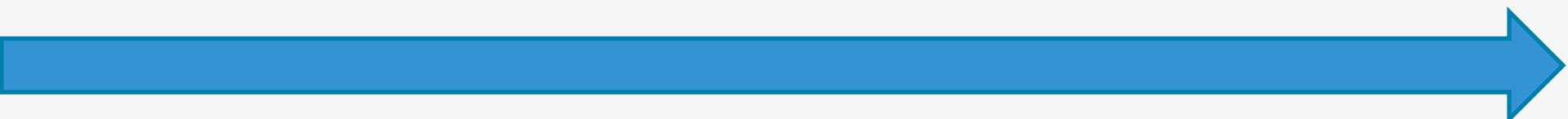
Integrate blockchain-explorer
 Enhance user-dashboard
 Support fabric kafka mode
 Create images at Dockerhub
 Support x86, ppc64le, s390x

v0.9.0

Add kubernetes agent
 Enable dynamic credential
 Redesign operational dashboard

v1.0.0

Support new governing model
 Support importing existing network



Q2 2017

Q4 2017

Q1 2018

Q4 2018

Q3 2019

Contents

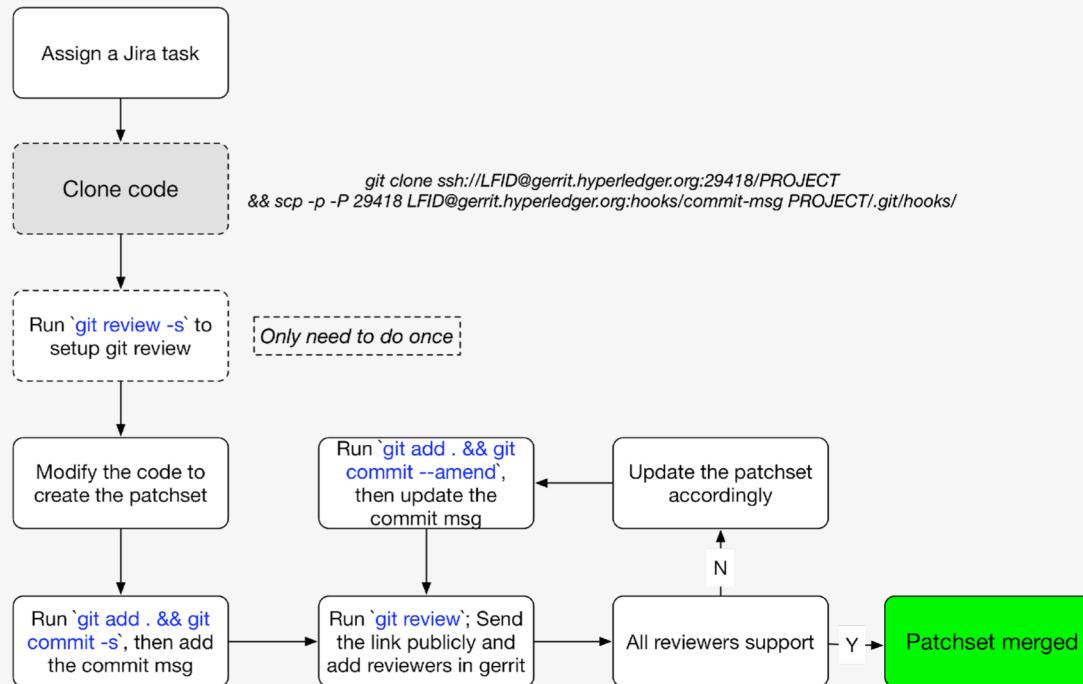
- What
- Key features
- Architecture and Design
- Roadmap
- **How to Contribute**
- Q&A

How to Contribute

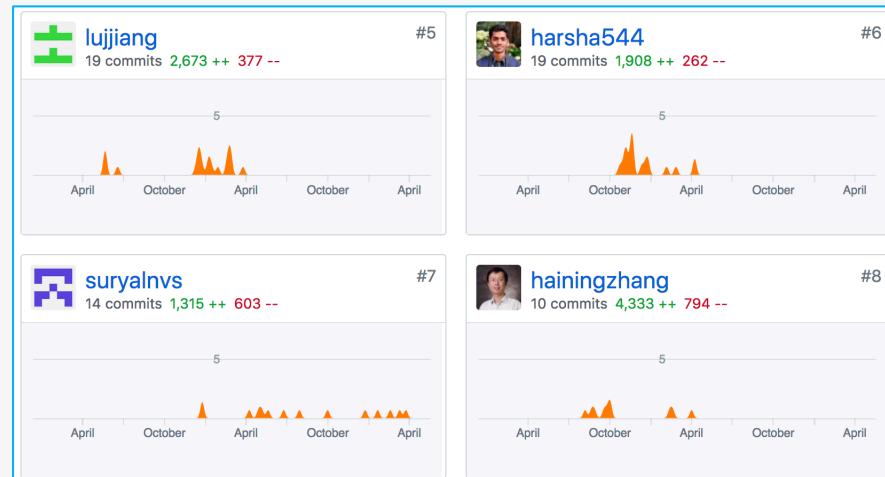
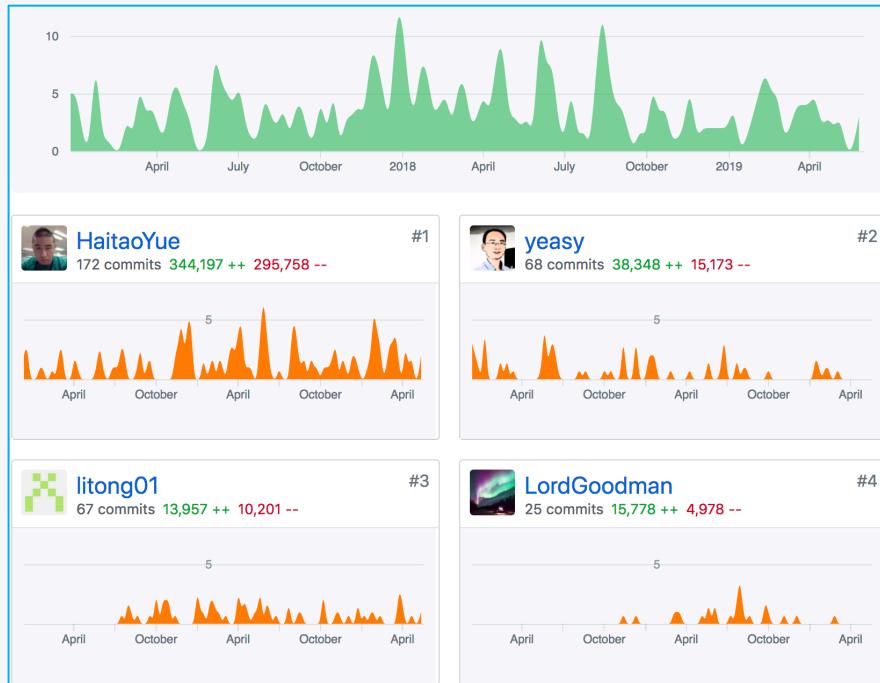
Process

- Assign a task from Jira
- Clone the code and create a new branch naming with the task ID
- Create patchsets, git commit and git review
- Send patchset link to rocketchat or mail-list for review

How to Contribute



Get Involved!



Useful channels

- Wiki: <https://wiki.hyperledger.org/display/cello/>
- Mail-list: cello@lists.hyperledger.org
- RocketChat: <https://chat.hyperledger.org/channel/cello>
- Jira Task Board: <https://jira.hyperledger.org/projects/CE>
- Gerrit: <https://gerrit.hyperledger.org/r/#/admin/projects/cello>

Question?

