Orchestration in Cloud Service

Cisco System CCATG Cloud Service

铁威 @fe-rest

Agenda

- Devops in Cloud
- Service Orchestration – why and how
- Practice – CSM Orchestration Engine
- Conclusion

About Us

- Cisco CCATG Cloud Service

  - From outside of Cisco, we are cloud vendor - at SaaS layer
  - From inside Cisco Cloud Services, we look like a cloud provider – private cloud

DevOps in Cloud

- Developers in Cloud era
  - Developers move to AWS, Rackspace …
  - Businesses move to Public/ Private/ Hybrid Cloud

- Key values the Cloud/IaaS offer
  - Abstracted Resources as Pools
  - Standardize the request as API

Issue #1: Cloud is just a VM Generator

- Phenomenon
  - Forget the cloud after VMs created
- Reason
  - Lack of tools connect Infrastructure & Application layers
  - Legacy scripts for legacy applications deployment
- Consequence
  - Using cloud without scaling
  - Learn how to use AWS/ Openstack/ Vmware etc. in computing/ storage/ network

Issue #2: Several Roles in a Cycle

- Phenomenon
  - System admin, operators, developers

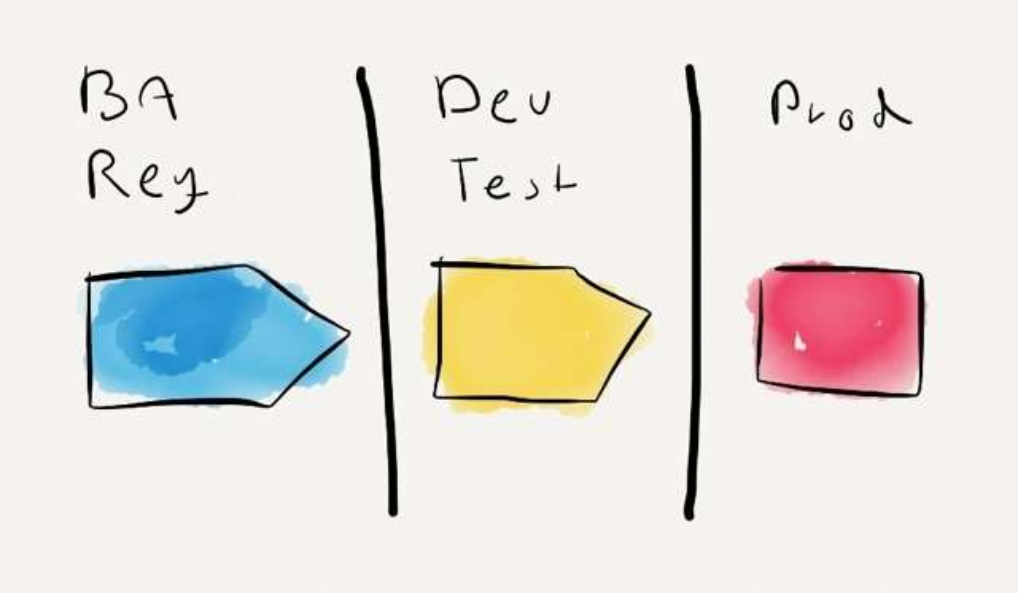
  No one knows the system exactly !
- Reason
  - Lack of tools connect Infrastructure & Application layers
  - Legacy scripts for legacy applications deployment
- Consequence
  - The more roles involved, the more uncertain the environments are.

Issue #3: Environments Inconsistent

- Phenomenon
  - QA, DEV & PRODUCT environments are managed by different teams, with different methods
- Reason
  - The environment of large system is complex.
  - A long period to prepare/build system environment
- Consequence
  - The environments are never the same.
  - The issues in different environments are hard to reproduce.

The Key Problems in Cloud DevOps

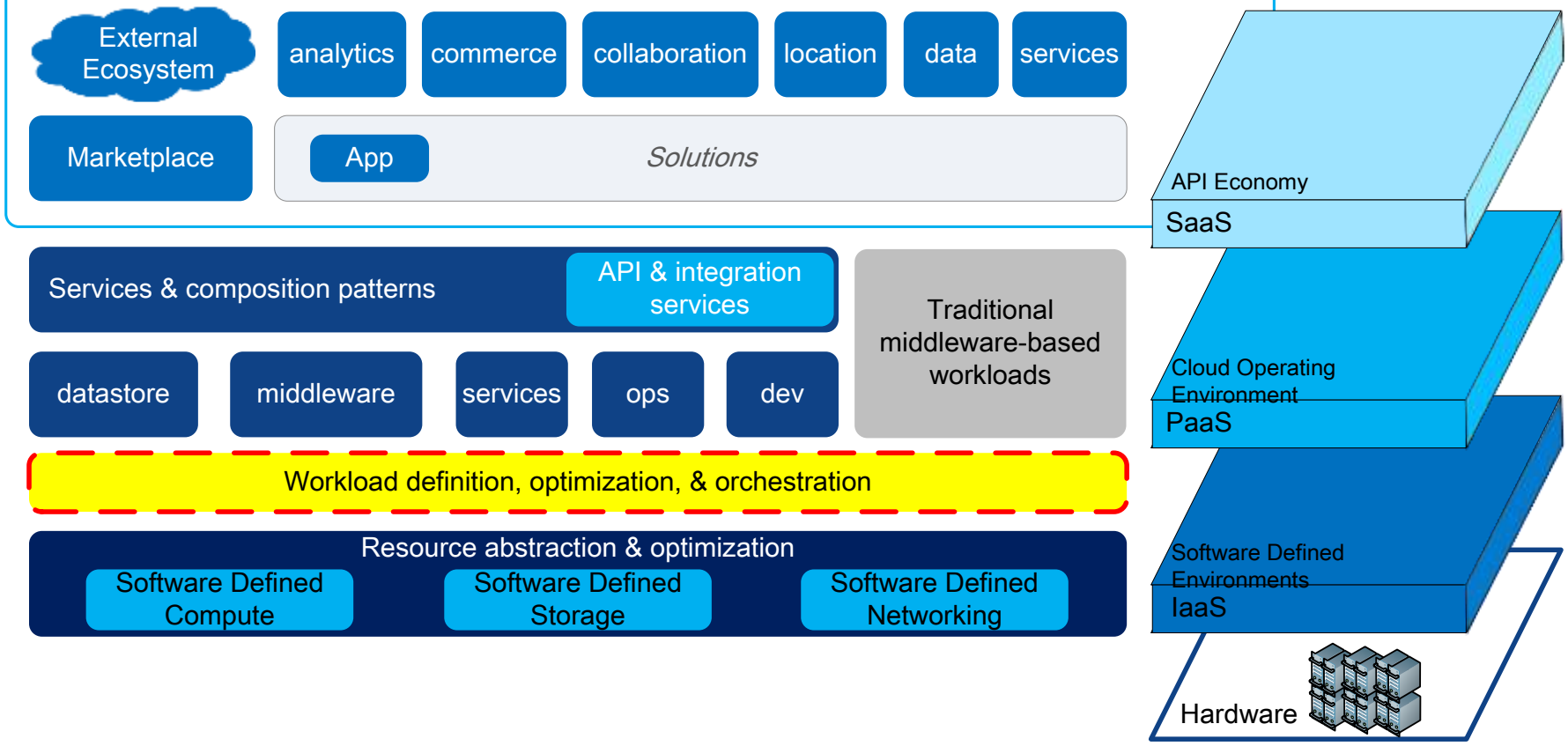
- Lack of a standardized method to describe the system environments
- Lack of a efficient method to build the system environments
- Lack of a uniform method to manage the system environments



Solution

- Service Orchestration
  - Why?
    - Cloud is all about scale - automated work flows are essential
    - DevOps' goal is deliver services – reduce intermediate steps
    - Cloud make it possible from the bottom
  - How?
    - Standardize everything – The DSL
    - Automate everything – The execution engine

Service Orchestration



Current Solutions

- DSL
  - AWS CloudFormation – Amazon
  - TOSCA – OASIS
  - Non-standard
- Solutions
  - AWS CloudFormation
  - Ubuntu Juju
  - OpenStack HEAT
  - Pivotal BOSH

Current Solutions Comparison

	CloudFormat	Puppet/Chef	Juju	HEAT	BOSH
Provisioning	YES	YES	YES	YES	YES
Package Management	YES	YES	YES	YES	YES
Template	YES	YES	YES	YES	YES
Standard flow	YES	YES	YES	YES	YES
build image	NO	NO	NO	YES	PART
Interdependence	YES	YES	YES	YES	YES
Lifecycle	YES	NO	YES	YES	YES
TOSCA Schema	NO	NO	NO	YES	NO
AWS CloudFormat Schema	YES	NO	NO	YES	NO
Cloud-aware	YES	NO	YES	YES	YES
Corss-cloud	NO	NO	YES	PART	YES
Snapshot	PART	NO	PART	PART	YES
Networking	YES	NO	YES	YES	YES
Cloud Volume	YES	NO	YES	YES	YES
Monitor	YES	PART	YES	YES	PART
Alert	YES	PART	YES	YES	PART
Continuous updating	YES	YES	YES	YES	YES
Auto-Scaling	YES	NO	PART	YES	PART

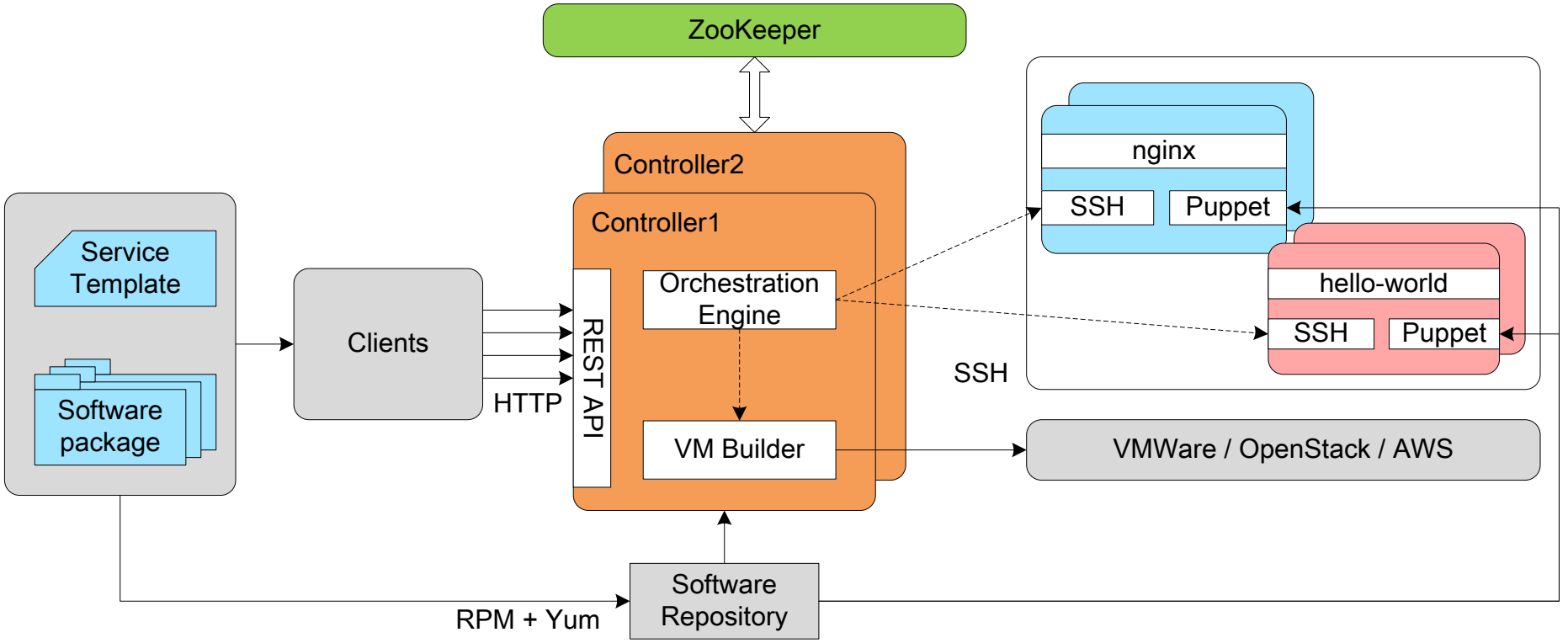
Background of CSM

- Background
  - Hybrid Cloud based on VMware, Openstack and AWS
  - CentOS
  - Applications delivered as RPM packages
  - Already have puppet scripts for configuration
- Why not open source ?
  - Most of the OS solutions focus on ONE platform
  - No succeed user story on CentOS
  - Too heavy for us

CSM Goal

- Orchestration in multi-cloud environment
  - Functional
    - Support rapid deployment and upgrades for CentOS-based application clusters
    - Support AWS, Openstack, VMWARE three cloud platforms
    - Standardize and automate the whole management process
  - Non-functional
    - Easy to expansion
    - Easy to maintain
    - High availability
- Principle – Minimum Viable Products

CSM Architecture



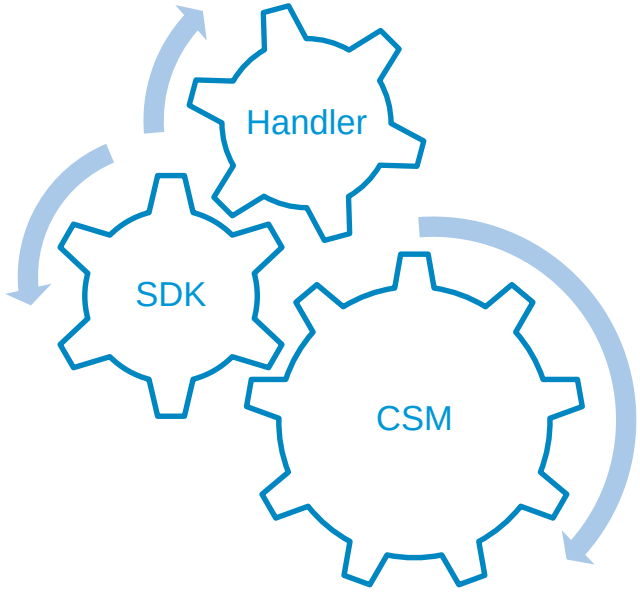
Template vs Instance

- Template
  - Describe the whole information about the deployment standardized
  - Describe all the information related to specific environment abstractly
- Instance

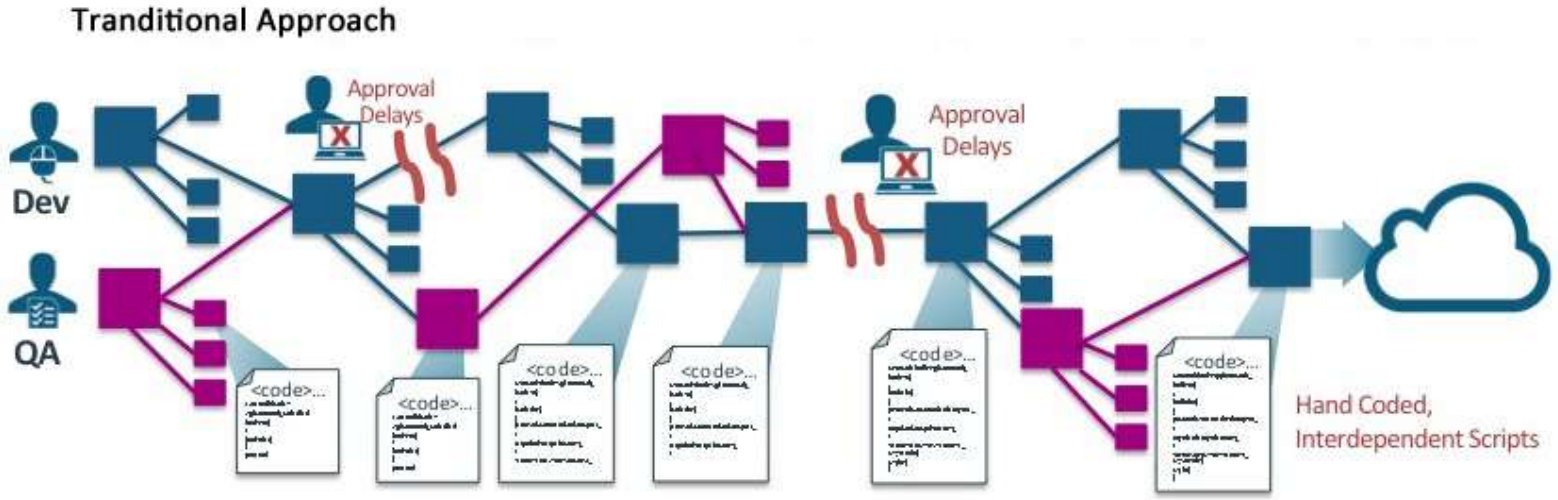
  An instance of the template related to a specific environment.
- Mapping
  - Assign values for the abstract properties in template
  - Assign specific parameters of target environment

Handler

- Using SDK to inject operations during the process
  - Predefined handlers – deploy, destroy, upgrade
  - Customizable handler – verify, notify



CSM Value

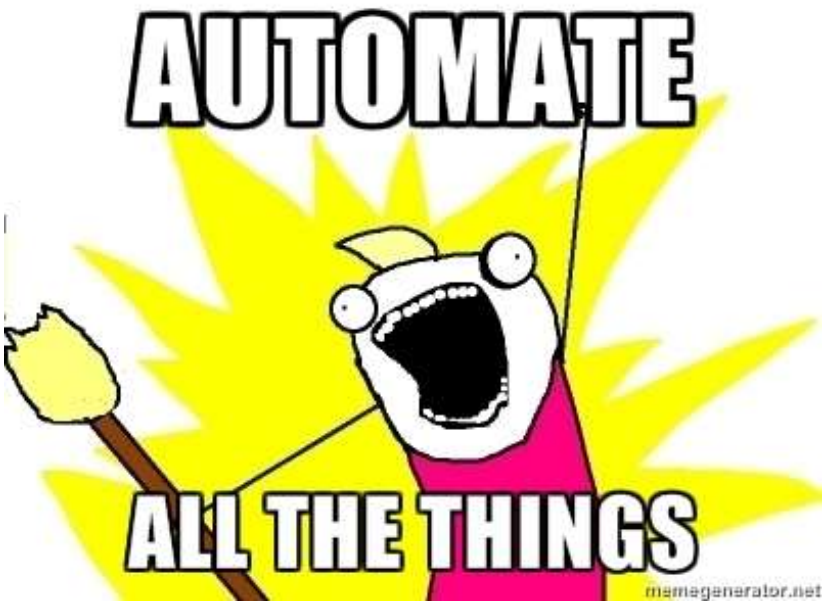


What's Next

- CI Integration
- More monitor
- Auto scaling

Conclusion

- DevOps could do more in cloud era
- Cloud service orchestration in a programmer's perspective
  - Programmer do think, machine do work
  - Automation everything
  - Streamline everything



Thank you

Q & A

Share your story

苏州软件开发者社区