

Compiling Abstract Specifications into Concrete Systems – Bringing Order to the Cloud

ANCOR

- Automated eNterprise network COmpileR -

Ian Unruh, Alexandru G. Bardas,

Rui Zhuang, Xinming Ou, Scott DeLoach



KANSAS STATE
UNIVERSITY



Cloud Users - Desired Features

- **Flexibility**
 - access to the raw resources *e.g.*, compute, storage
- Reliable **automation** capabilities
 - Non-scenario-dependent
 - Automatic deployment and maintenance
 - Dynamic cluster expansion and contraction
- **Migration** between different cloud providers
 - Capturing infrastructure and application requirements in a specification

Current Cloud Computing Offerings

- Allow customers to decide *how much management they want*:

- Infrastructure as a Service (IaaS)

- e.g., Amazon Web Services, OpenStack



- Platform as a Service (PaaS)

- e.g., Heroku, Microsoft Azure



- Software as a Service (SaaS)

- e.g., Salesforce, Google Apps






Current Cloud Computing Offerings

- Allow customers to decide *how much* management they want:

- Infrastructure as a Service (IaaS)

- e.g., Amazon Web Services, OpenStack






	Flexibility (access to the raw resources)
	Automation (non-scenario-dependent)
	Migration (capturing requirements in a specification)

Current Cloud Computing Offerings

- Allow customers to decide *how much* management they want:

- Software as a Service (SaaS)
 - e.g., Salesforce, Google Apps






	Flexibility (access to the raw resources)
	Automation (non-scenario-dependent)
	Migration (capturing requirements in a specification)

Current Cloud Computing Offerings

- Allow customers to decide *how much* management they want:

- Platform as a Service (PaaS)
 - e.g., Heroku, Microsoft Azure



	Flexibility (access to the raw resources)
	Automation (non-scenario-dependent)
	Migration (capturing requirements in a specification)

Proposed Solution

- An **abstraction** that captures *what* a cloud user needs instead of low-level details on *how* to implement those needs



- There must be a process to automatically compile the abstraction into a valid concrete system

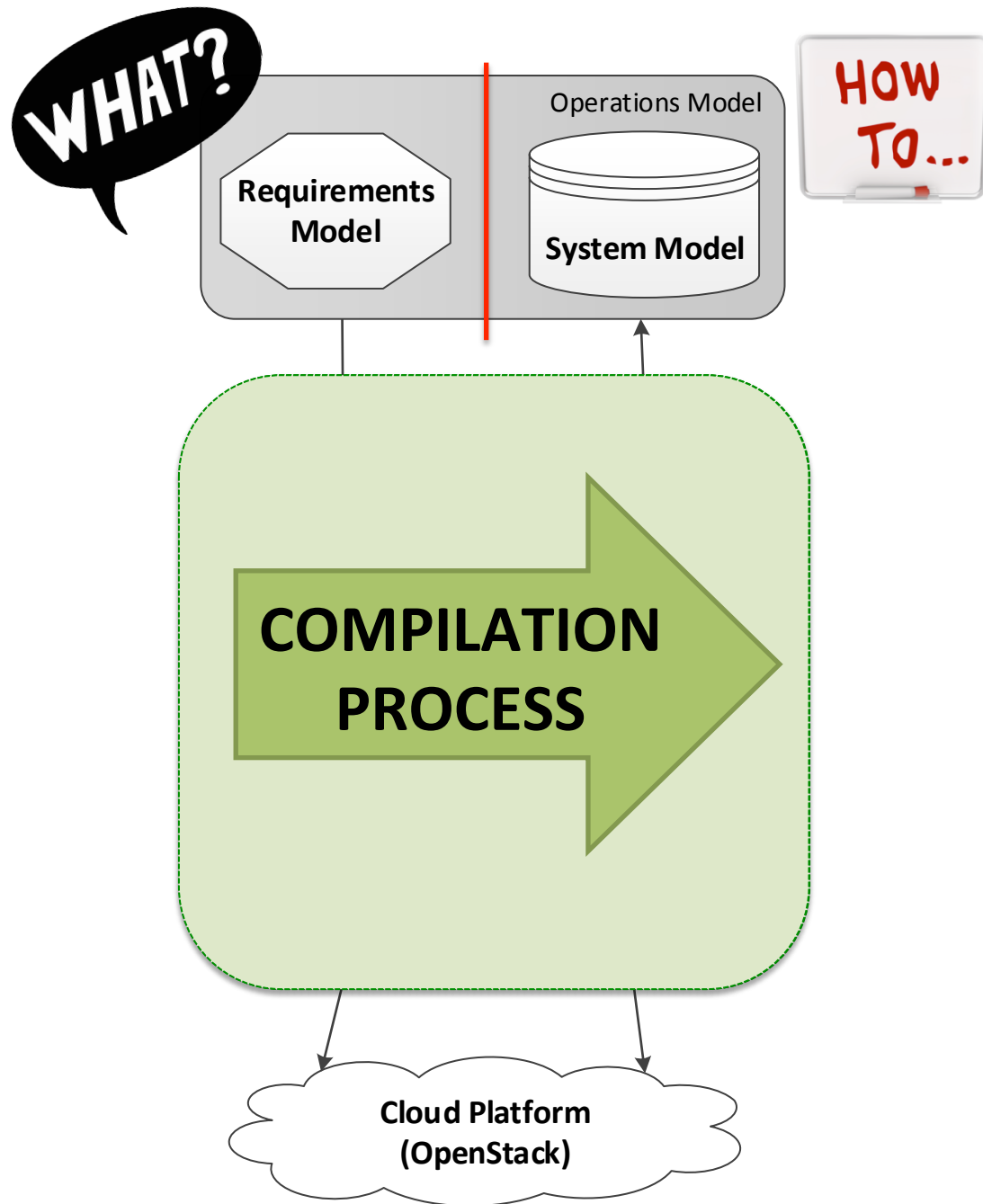
Proposed Solution

- An **abstraction** that captures *what* a cloud user needs instead of low-level details on *how* to implement those needs



- There must be a process to **automatically compile the abstraction into a valid concrete system**

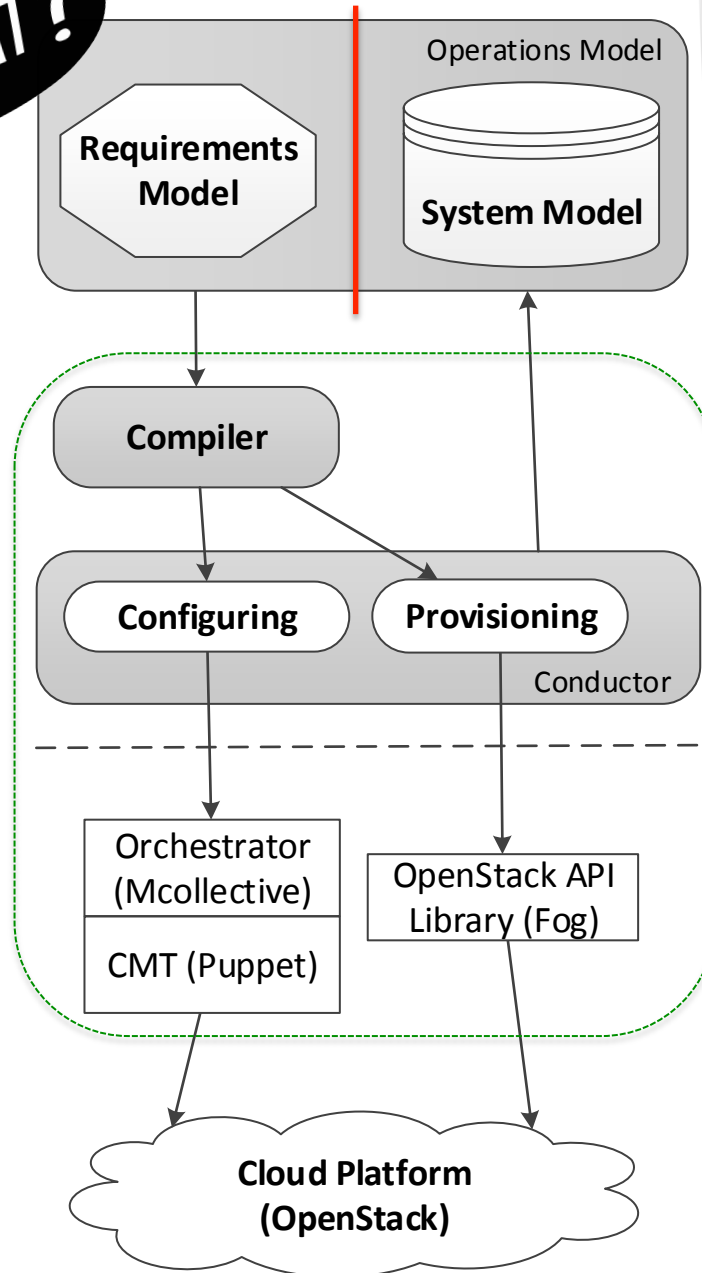
ANCOR



ANCOR

WHAT?

HOW TO...

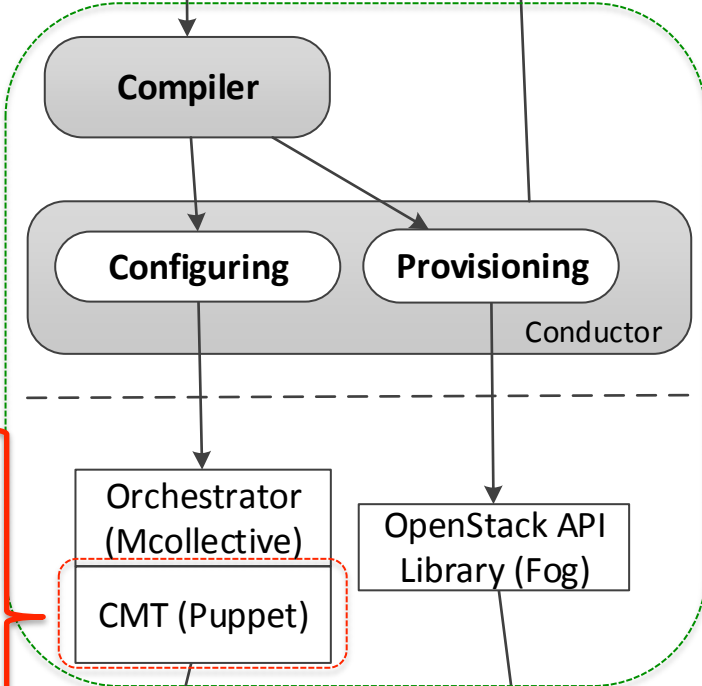
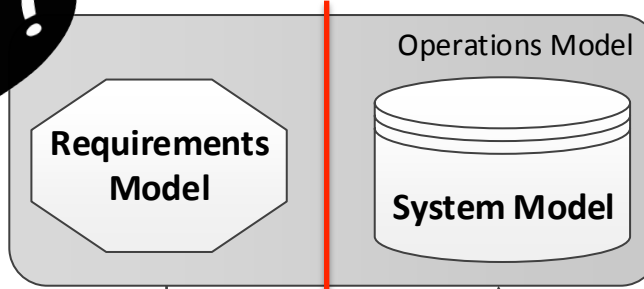


COMPILATION PROCESS

ANCOR

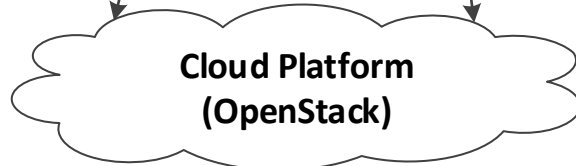
WHAT?

HOW TO...

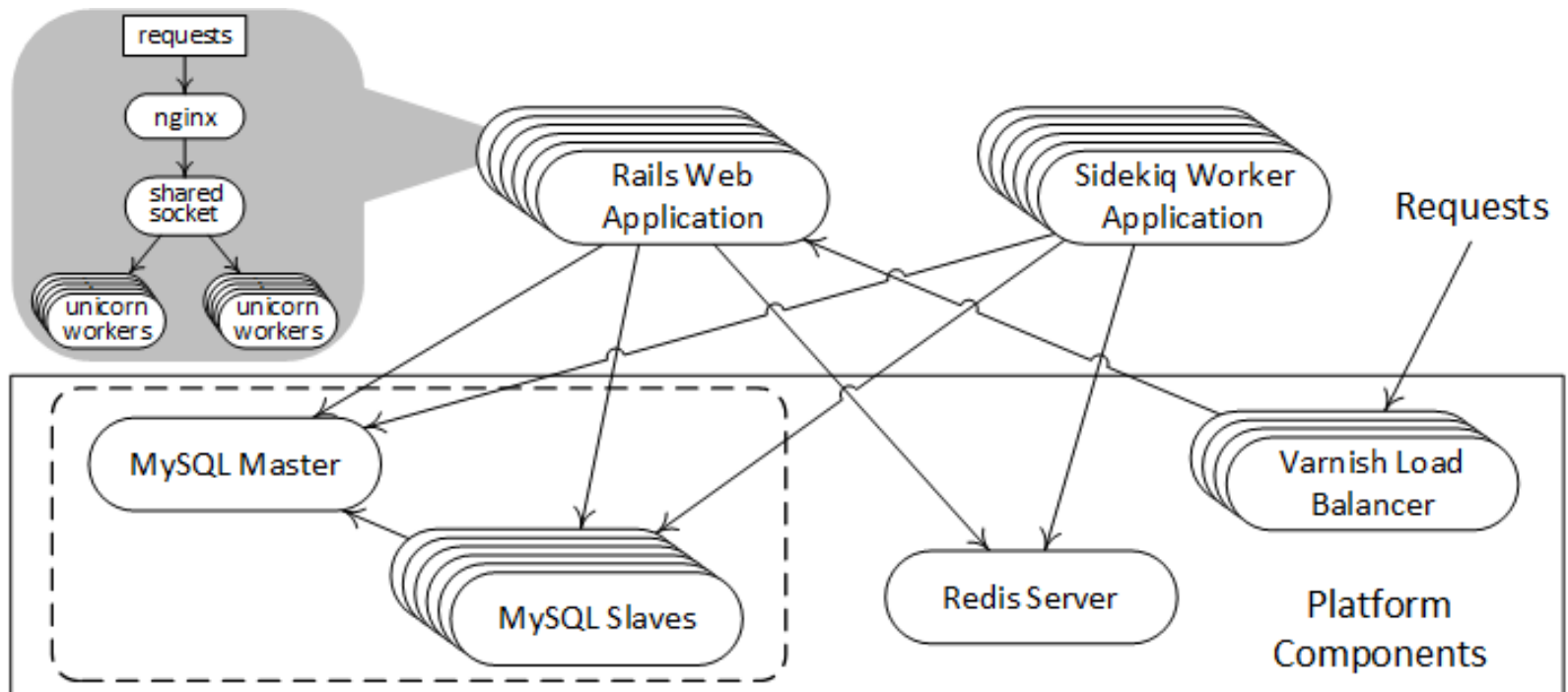


COMPILATION PROCESS

Configuration Management Tools (CMTs)



Deploying an eCommerce Website



Scalable and highly available eCommerce website architecture

ANCOR Requirement Modeling Language (ARML)

```
1. goals:
2.  ecommerce:
3.     name: eCommerce frontend
4.     roles:
5.         - weblb
6.         - webapp
7.         - worker
8.         - work_queue
9.         - db_master
10.        - db_slave

11. roles:
12.  weblb:
13.     name: Web application load balancer
14.     min: 2
15.     is_public: true
16.     implementations:
17.         default:
18.             profile: role::weblb::default
19.     exports:
20.         http: { type: single_port, protocol: tcp, number: 80 }
21.     imports:
22.         webapp: http

23.  webapp:
24.     name: Web application
25.     min: 3
26.     implementations:
27.         default:
28.             profile: role::webapp::default
29.     exports:
30.         http: { type: single_port, protocol: tcp }
31.     imports:
32.         db_master: querying
33.         db_slave: querying
34.         work_queue: redis

35.  worker:
36.     name: Sidekiq worker application
37.     min: 2
38.     implementations:
39.         default:
40.             profile: role::worker::default
41.     imports:
42.         db_master: querying
43.         db_slave: querying
44.         work_queue: redis

45.  work_queue:
46.     name: Redis work queue
47.     implementations:
48.         default:
49.             profile: role::work_queue::default
50.     exports:
51.         redis: { type: single_port, protocol: tcp }

52.  db_master:
53.     name: MySQL master
54.     implementations:
55.         default:
56.             profile: role::db_master::default
57.     exports:
58.         querying: { type: single_port, protocol: tcp }

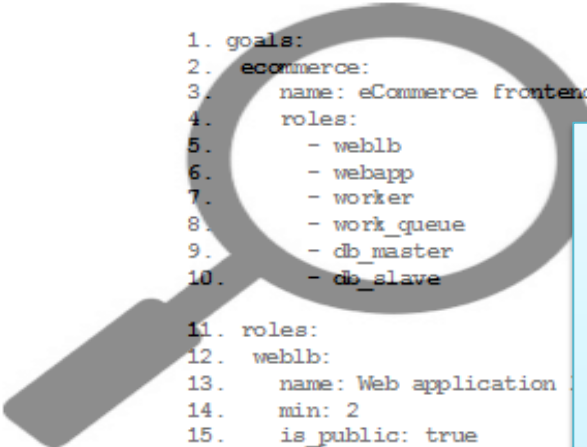
59.  db_slave:
60.     name: MySQL slave
61.     implementations:
62.         default:
63.             profile: role::db_slave::default
64.     min: 2
65.     exports:
66.         querying: { type: single_port, protocol: tcp }
67.     imports:
68.         db_master: querying
```

Ancor Requirement Modeling Language (ARML)



```
1. goals:
2.  ecommerce:
3.     name: eCommerce frontend
4.     roles:
5.         - weblb
6.         - webapp
7.         - worker
8.         - work_queue
9.         - db_master
10.        - db_slave
11. roles:
12.  weblb:
13.     name: Web application load balancer
14.     min: 2
15.     is_public: true
16.     implementations:
17.         default:
18.             profile: role::weblb::default
19.     exports:
20.         http: { type: single_port, protocol: tcp, number: 80 }
21.     imports:
22.         webapp: http
23.  webapp:
24.     name: Web application
25.     min: 3
26.     implementations:
27.         default:
28.             profile: role::webapp::default
29.     exports:
30.         http: { type: single_port, protocol: tcp }
31.     imports:
32.         db_master: querying
33.         db_slave: querying
34.         work_queue: redis
35.  worker:
36.     name: Sidekiq worker application
37.     min: 2
38.     implementations:
39.         default:
40.             profile: role::worker::default
41.     imports:
42.         db_master: querying
43.         db_slave: querying
44.         work_queue: redis
45.  work_queue:
46.     name: Redis work queue
47.     implementations:
48.         default:
49.             profile: role::work_queue::default
50.     exports:
51.         redis: { type: single_port, protocol: tcp }
52.  db_master:
53.     name: MySQL master
54.     implementations:
55.         default:
56.             profile: role::db_master::default
57.     exports:
58.         querying: { type: single_port, protocol: tcp }
59.  db_slave:
60.     name: MySQL slave
61.     implementations:
62.         default:
63.             profile: role::db_slave::default
64.     min: 2
65.     exports:
66.         querying: { type: single_port, protocol: tcp }
67.     imports:
68.         db_master: querying
```

Ancor Requirement Modeling Language (ARML)



```
1. goals:
2.   ecommerce:
3.     name: eCommerce frontend
4.     roles:
5.       - weblb
6.       - webapp
7.       - worker
8.       - work_queue
9.       - db_master
10.      - db_slave
11. roles:
12.   weblb:
13.     name: Web application
14.     min: 2
15.     is_public: true
16.     implementations:
17.       default:
18.         profile: role::web
19.     exports:
20.       http: { type: single
21.     imports:
22.       webapp: http
23.   webapp:
24.     name: Web application
25.     min: 3
26.     implementations:
27.       default:
28.         profile: role::web
29.     exports:
30.       http: { type: single
31.     imports:
32.       db_master: querying
33.       db_slave: querying
34.       work_queue: redis
```

```
35. worker:
36.   name: Sidekiq worker application
37.   min: 2
38.   implementations:
```

```
goals:
  ecommerce:
    name: eCommerce frontend
  roles:
    - weblb
    - webapp
    - worker
    - work_queue
    - db_master
    - db_slave
```

```
67. imports:
68.   db_master: querying
```

Ancor Requirement Modeling Language (ARML)

```
1. goals:
2. ecommerce:
3.   name: eCommerce frontend
4.   roles:
5.     - weblb
6.     - webapp
7.     - worker
8.     - work_queue
9.     - db_master
10.    - db_slave
11. roles:
12.  weblb:
13.    name: Web application load balancer
14.    min: 2
15.    is_public: true
16.    implementations:
17.      default:
18.        profile: role::weblb::default
19.    exports:
20.      http: { type: single_port, protocol: tcp, number: 80 }
21.    imports:
22.      webapp: http
23.  webapp:
24.    name: Web application
25.    min: 3
26.    implementations:
27.      default:
28.        profile: role::webapp::default
29.    exports:
30.      http: { type: single_port, protocol: tcp }
31.    imports:
32.      db_master: querying
33.      db_slave: querying
34.      work_queue: redis
35.  worker:
36.    name: Sidekiq worker application
37.    min: 2
38.    implementations:
39.      default:
40.        profile: role::worker::default
41.    imports:
42.      db_master: querying
43.      db_slave: querying
44.      work_queue: redis
45.  work_queue:
46.    name: Redis work queue
47.    implementations:
48.      default:
49.        profile: role::work_queue::default
50.    exports:
51.      redis: { type: single_port, protocol: tcp }
52.  db_master:
53.    name: MySQL master
54.    implementations:
55.      default:
56.        profile: role::db_master::default
57.    exports:
58.      querying: { type: single_port, protocol: tcp }
59.  db_slave:
60.    name: MySQL slave
61.    implementations:
62.      default:
63.        profile: role::db_slave::default
64.    min: 2
65.    exports:
66.      querying: { type: single_port, protocol: tcp }
67.    imports:
68.      db_master: querying
```


Ancor Requirement Modeling Language (ARML)

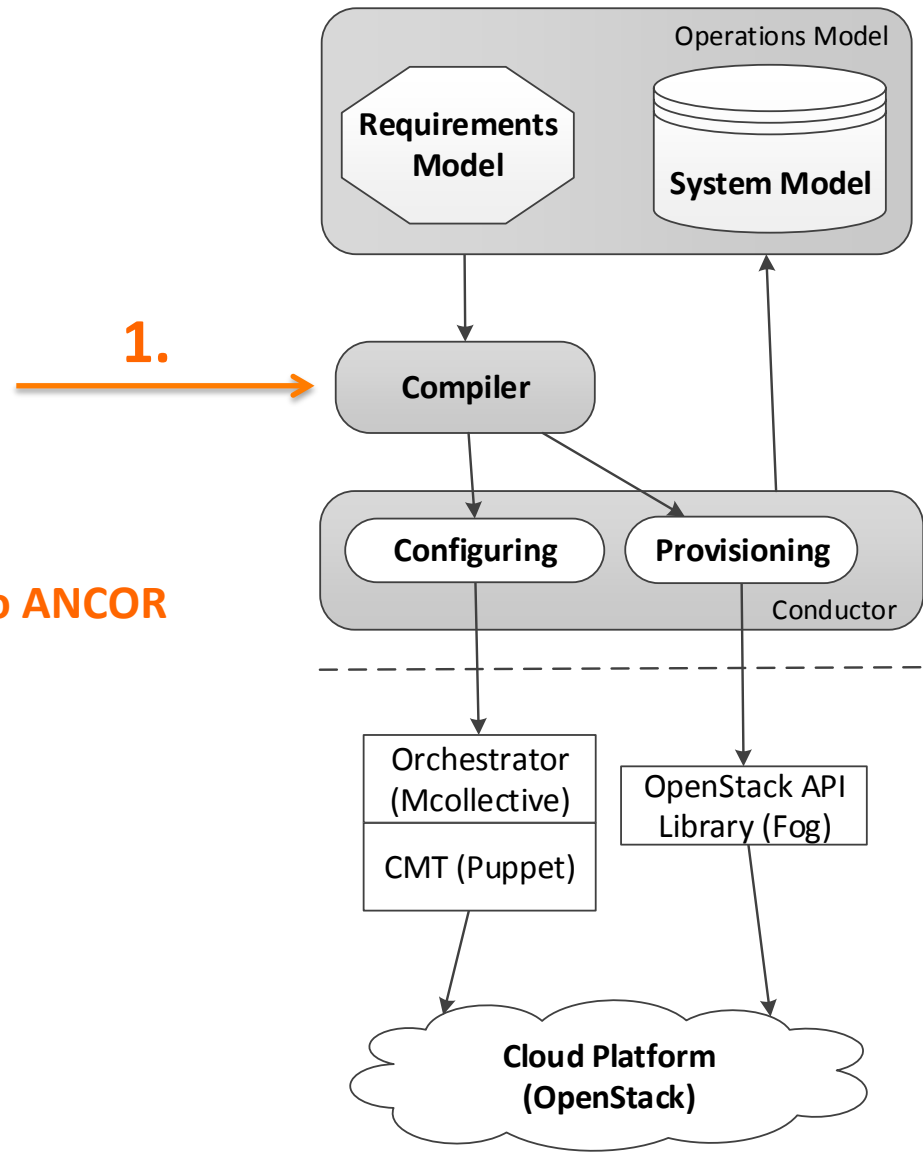
```
1. goals:
2.  ecommerce:
3.     name: eComm
4.     roles:
5.     - weblb
6.     - webapp
7.     - worker
8.     - work_qu
9.     - db_mast
10.    - db_slav
11. roles:
12.  weblb:
13.     name: Web a
14.     min: 2
15.     is_public:
16.     implementat
17.     default: {
18.     profile
19.     exports:
20.     http: { ty
21.     imports:
22.     webapp: h
23.  webapp:
24.     name: Web a
25.     min: 3
26.     implementat
27.     default: {
28.     profile
29.     exports:
30.     http: { t
31.     imports:
32.     db_master
33.     db_slave:
34.     work_queu
```

```
roles:
  weblb:
    name: web application load balancer
    min: 2
    is_public: true
    implementations:
      default: { profile:
        "role::ecommerce::
        weblb::default" }
    exports:
      http: { type: single_port,
        protocol: tcp, number: 80 }
    imports:
      webapp: http
```

ANCOR Workflow

```
1: node
2:   name: cCommerce frontend
3:   role: web
4:   profile: role:web:default
5:   _role: web
6:   _profile: role:web:default
7:   _role: web
8:   _profile: role:web:default
9:   _role: web
10:  _profile: role:web:default
11:  _role: web
12:  _profile: role:web:default
13:  _role: web
14:  _profile: role:web:default
15:  _role: web
16:  _profile: role:web:default
17:  _role: web
18:  _profile: role:web:default
19:  _role: web
20:  _profile: role:web:default
21:  _role: web
22:  _profile: role:web:default
23:  _role: web
24:  _profile: role:web:default
25:  _role: web
26:  _profile: role:web:default
27:  _role: web
28:  _profile: role:web:default
29:  _role: web
30:  _profile: role:web:default
31:  _role: web
32:  _profile: role:web:default
33:  _role: web
34:  _profile: role:web:default
35:  _role: web
36:  _profile: role:web:default
37:  _role: web
38:  _profile: role:web:default
39:  _role: web
40:  _profile: role:web:default
41:  _role: web
42:  _profile: role:web:default
43:  _role: web
44:  _profile: role:web:default
45:  _role: web
46:  _profile: role:web:default
47:  _role: web
48:  _profile: role:web:default
49:  _role: web
50:  _profile: role:web:default
51:  _role: web
52:  _profile: role:web:default
53:  _role: web
54:  _profile: role:web:default
55:  _role: web
56:  _profile: role:web:default
57:  _role: web
58:  _profile: role:web:default
59:  _role: web
60:  _profile: role:web:default
61:  _role: web
62:  _profile: role:web:default
63:  _role: web
64:  _profile: role:web:default
65:  _role: web
66:  _profile: role:web:default
67:  _role: web
68:  _profile: role:web:default
69:  _role: web
70:  _profile: role:web:default
71:  _role: web
72:  _profile: role:web:default
73:  _role: web
74:  _profile: role:web:default
75:  _role: web
76:  _profile: role:web:default
77:  _role: web
78:  _profile: role:web:default
79:  _role: web
80:  _profile: role:web:default
81:  _role: web
82:  _profile: role:web:default
83:  _role: web
84:  _profile: role:web:default
85:  _role: web
86:  _profile: role:web:default
87:  _role: web
88:  _profile: role:web:default
89:  _role: web
90:  _profile: role:web:default
91:  _role: web
92:  _profile: role:web:default
93:  _role: web
94:  _profile: role:web:default
95:  _role: web
96:  _profile: role:web:default
97:  _role: web
98:  _profile: role:web:default
99:  _role: web
100: _profile: role:web:default
```

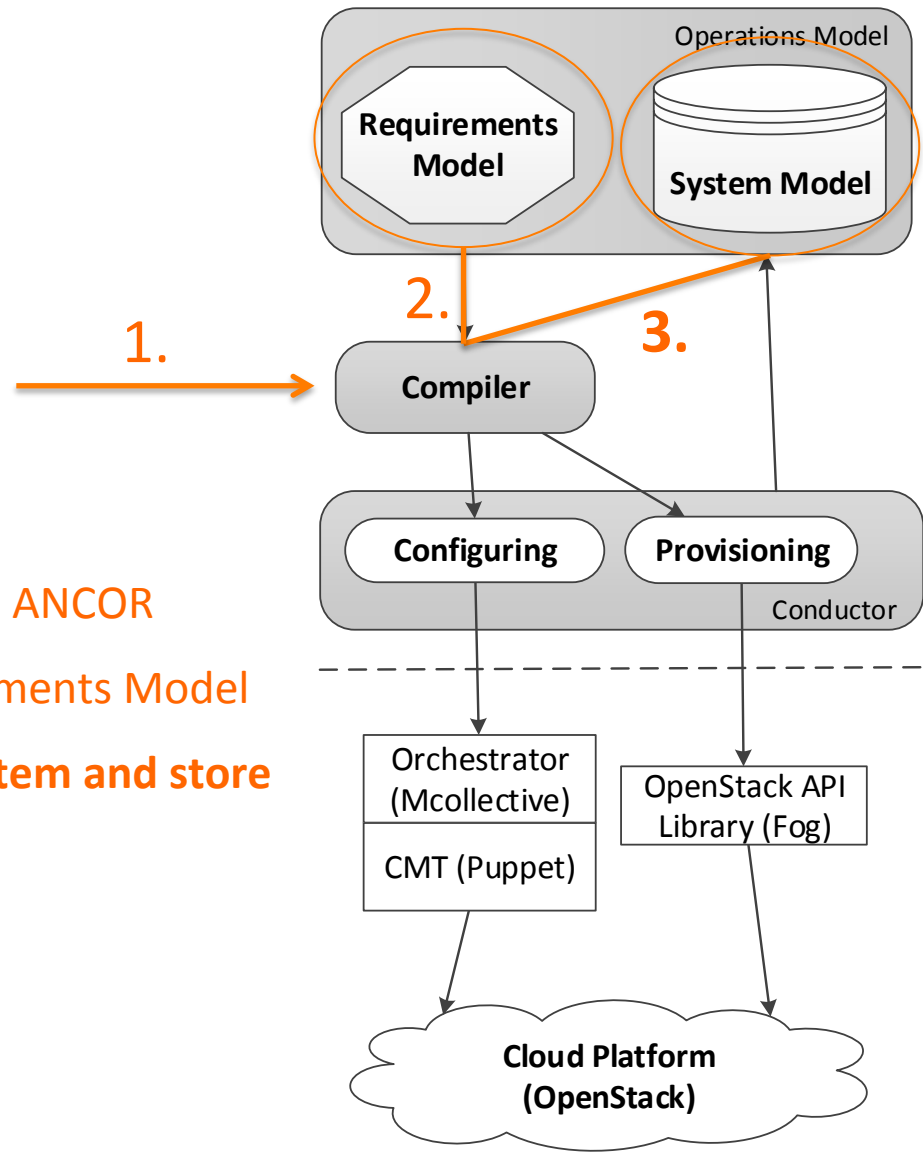
1. Passing the IT system specification to ANCOR



ANCOR Workflow

```
1. ymla
2. namespace:
3.   name: cCommerce frontend
4.   role:
5.     - webapp
6.     - webapp
7.     - webapp
8.     - webapp
9.     - webapp
10.    - @_kname
11.
12. role:
13.   name: web application load balancer
14.   role:
15.     - webapp
16.     - webapp
17.     - webapp
18.     - webapp
19.     - webapp
20.     - webapp
21.     - webapp
22.     - webapp
23.     - webapp
24.     - webapp
25.     - webapp
26.     - webapp
27.     - webapp
28.     - webapp
29.     - webapp
30.     - webapp
31.     - webapp
32.     - webapp
33.     - webapp
34.     - webapp
35.     - webapp
36.     - webapp
37.     - webapp
38.     - webapp
39.     - webapp
40.     - webapp
41.     - webapp
42.     - webapp
43.     - webapp
44.     - webapp
45.     - webapp
46.     - webapp
47.     - webapp
48.     - webapp
49.     - webapp
50.     - webapp
51.     - webapp
52.     - webapp
53.     - webapp
54.     - webapp
55.     - webapp
56.     - webapp
57.     - webapp
58.     - webapp
59.     - webapp
60.     - webapp
61.     - webapp
62.     - webapp
63.     - webapp
64.     - webapp
65.     - webapp
66.     - webapp
67.     - webapp
68.     - webapp
69.     - webapp
70.     - webapp
71.     - webapp
72.     - webapp
73.     - webapp
74.     - webapp
75.     - webapp
76.     - webapp
77.     - webapp
78.     - webapp
79.     - webapp
80.     - webapp
81.     - webapp
82.     - webapp
83.     - webapp
84.     - webapp
85.     - webapp
86.     - webapp
87.     - webapp
88.     - webapp
89.     - webapp
90.     - webapp
91.     - webapp
92.     - webapp
93.     - webapp
94.     - webapp
95.     - webapp
96.     - webapp
97.     - webapp
98.     - webapp
99.     - webapp
100.    - webapp
```

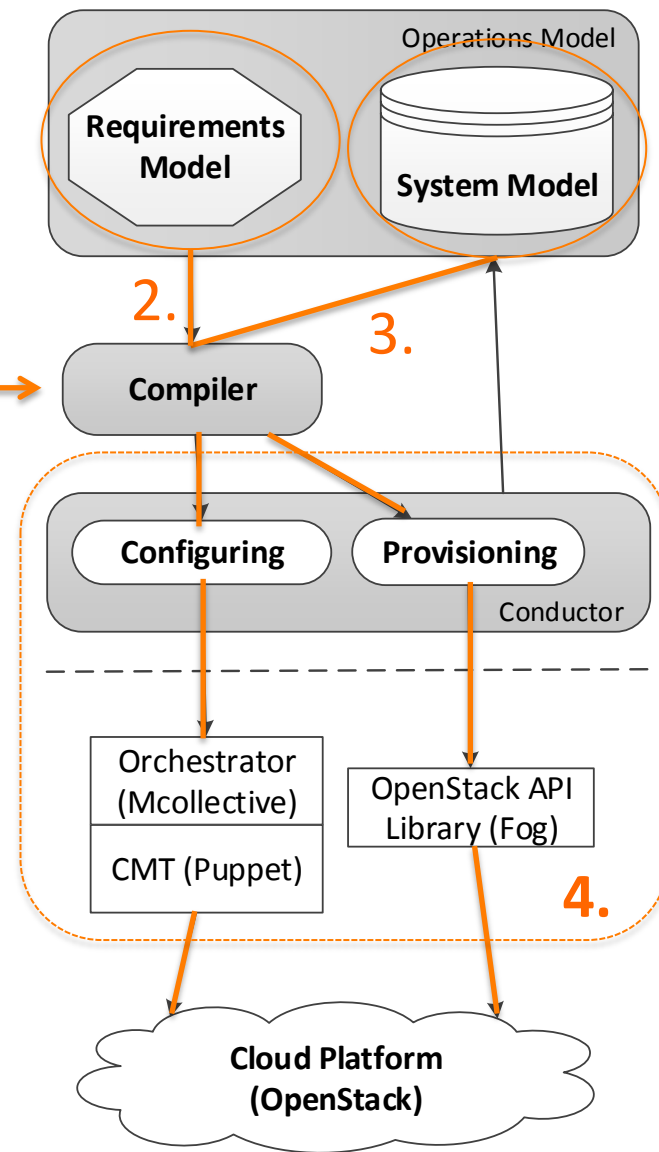
1. Passing the IT system specification to ANCOR
2. Specification is stored in the Requirements Model
3. Compute low-level details of the system and store them in the System Model



ANCOR Workflow

```
1. yolo
2. namespace:
3.   name: cCommerce_frontend
4.   namespace:
5.     - webapp
6.     - webapp
7.     - webapp
8.     - webapp
9.     - webapp
10.    - @_kubernetes
11.    - @_kubernetes
12.  roles:
13.  - role: web_application_load_balancer
14.  - role: web_application_load_balancer
15.  - role: web_application_load_balancer
16.  - role: web_application_load_balancer
17.  - role: web_application_load_balancer
18.  - role: web_application_load_balancer
19.  - role: web_application_load_balancer
20.  - role: web_application_load_balancer
21.  - role: web_application_load_balancer
22.  - role: web_application_load_balancer
23.  - role: web_application_load_balancer
24.  - role: web_application_load_balancer
25.  - role: web_application_load_balancer
26.  - role: web_application_load_balancer
27.  - role: web_application_load_balancer
28.  - role: web_application_load_balancer
29.  - role: web_application_load_balancer
30.  - role: web_application_load_balancer
31.  - role: web_application_load_balancer
32.  - role: web_application_load_balancer
33.  - role: web_application_load_balancer
34.  - role: web_application_load_balancer
35.  - role: web_application_load_balancer
36.  - role: web_application_load_balancer
37.  - role: web_application_load_balancer
38.  - role: web_application_load_balancer
39.  - role: web_application_load_balancer
40.  - role: web_application_load_balancer
41.  - role: web_application_load_balancer
42.  - role: web_application_load_balancer
43.  - role: web_application_load_balancer
44.  - role: web_application_load_balancer
45.  - role: web_application_load_balancer
46.  - role: web_application_load_balancer
47.  - role: web_application_load_balancer
48.  - role: web_application_load_balancer
49.  - role: web_application_load_balancer
50.  - role: web_application_load_balancer
51.  - role: web_application_load_balancer
52.  - role: web_application_load_balancer
53.  - role: web_application_load_balancer
54.  - role: web_application_load_balancer
55.  - role: web_application_load_balancer
56.  - role: web_application_load_balancer
57.  - role: web_application_load_balancer
58.  - role: web_application_load_balancer
59.  - role: web_application_load_balancer
60.  - role: web_application_load_balancer
61.  - role: web_application_load_balancer
62.  - role: web_application_load_balancer
63.  - role: web_application_load_balancer
64.  - role: web_application_load_balancer
65.  - role: web_application_load_balancer
66.  - role: web_application_load_balancer
67.  - role: web_application_load_balancer
68.  - role: web_application_load_balancer
69.  - role: web_application_load_balancer
70.  - role: web_application_load_balancer
71.  - role: web_application_load_balancer
72.  - role: web_application_load_balancer
73.  - role: web_application_load_balancer
74.  - role: web_application_load_balancer
75.  - role: web_application_load_balancer
76.  - role: web_application_load_balancer
77.  - role: web_application_load_balancer
78.  - role: web_application_load_balancer
79.  - role: web_application_load_balancer
80.  - role: web_application_load_balancer
81.  - role: web_application_load_balancer
82.  - role: web_application_load_balancer
83.  - role: web_application_load_balancer
84.  - role: web_application_load_balancer
85.  - role: web_application_load_balancer
86.  - role: web_application_load_balancer
87.  - role: web_application_load_balancer
88.  - role: web_application_load_balancer
89.  - role: web_application_load_balancer
90.  - role: web_application_load_balancer
91.  - role: web_application_load_balancer
92.  - role: web_application_load_balancer
93.  - role: web_application_load_balancer
94.  - role: web_application_load_balancer
95.  - role: web_application_load_balancer
96.  - role: web_application_load_balancer
97.  - role: web_application_load_balancer
98.  - role: web_application_load_balancer
99.  - role: web_application_load_balancer
100. - role: web_application_load_balancer
```

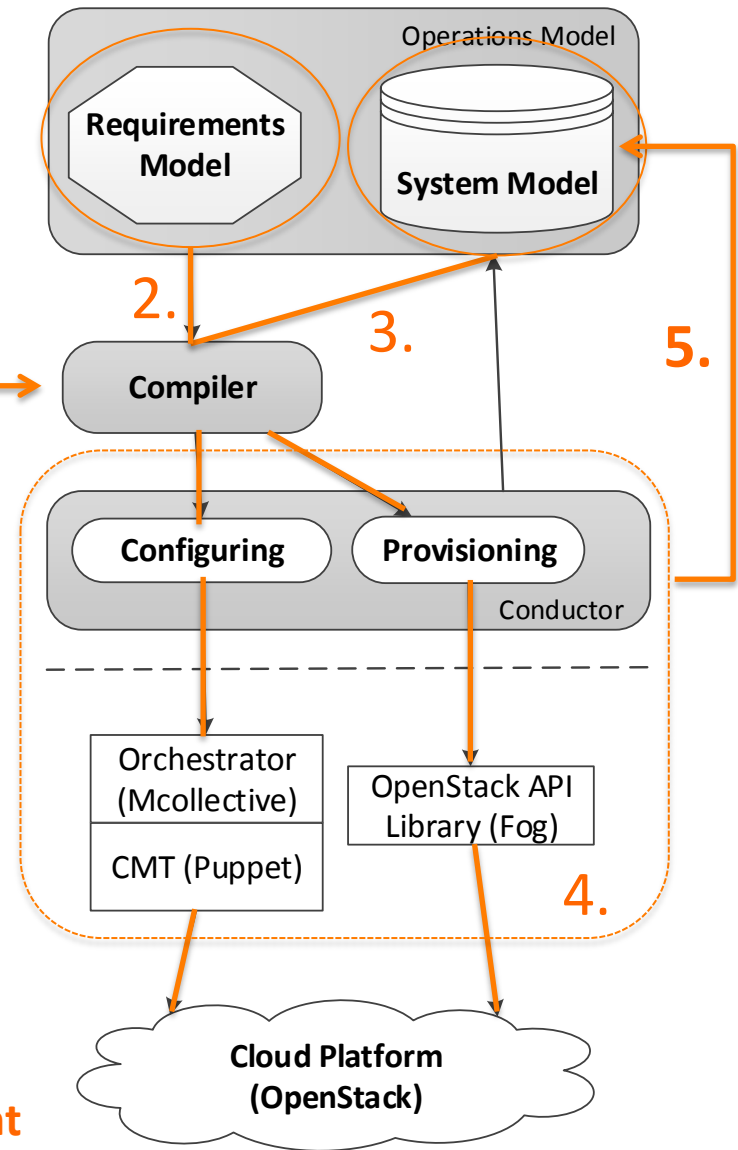
1. Passing the IT system specification to ANCOR
2. Specification is stored in the Requirements Model
3. Compute low-level details of the system and store them in the System Model
4. Start deploying the system on the cloud infrastructure using cloud platform and CMT APIs



ANCOR Workflow

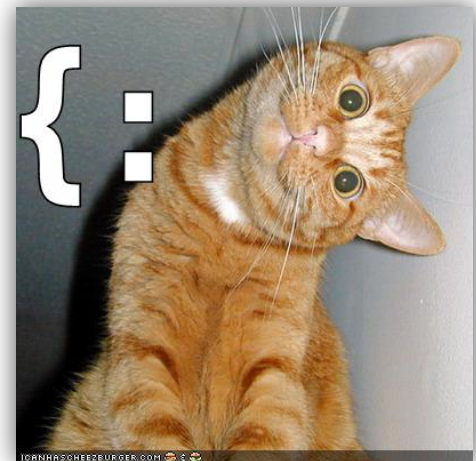
```
1: #role
2: #namespace
3: name: cCommerce frontend
4: #role
5: #namespace
6: name: webapp
7: #role
8: #namespace
9: name: work_queue
10: #role
11: #namespace
12: name: redis
13: #role
14: #namespace
15: name: web_application load_balancer
16: #role
17: #namespace
18: name: redis
19: #role
20: #namespace
21: name: redis
22: #role
23: #namespace
24: name: web_application
25: #role
26: #namespace
27: name: redis
28: #role
29: #namespace
30: name: redis
31: #role
32: #namespace
33: name: redis
34: #role
35: #namespace
36: name: redis
37: #role
38: #namespace
39: name: redis
40: #role
41: #namespace
42: name: redis
43: #role
44: #namespace
45: name: redis
46: #role
47: #namespace
48: name: redis
49: #role
50: #namespace
51: name: redis
52: #role
53: #namespace
54: name: redis
55: #role
56: #namespace
57: name: redis
58: #role
59: #namespace
60: name: redis
```

1. Passing the IT system specification to ANCOR
2. Specification is stored in the Requirements Model
3. Compute low-level details of the system and store them in the System Model
4. Start deploying the system on the cloud infrastructure using cloud platform and CMT APIs
5. Update the System Model so it is ***always*** consistent with the deployed system



ANCOR Benefits

- ✓ Infrastructure and application “portability”
- ✓ Up-to-date application dependencies
- ✓ Building highly dynamic systems
- ✓ Automated fine-grained firewall configuration
- ✓ Security assessments
- ✓ Performance evaluations
- ✓ Creating customized PaaS



ANCOR

- Current implementation and more information:

<http://arguslab.github.io/ancor/>



Conclusion

- Separating user requirements from the implementation details has the potential of changing the way IT systems are deployed and managed in the cloud
- ANCOR – framework that captures the high-level requirements and translates them into a working IT system on a cloud infrastructure

<http://arguslab.github.io/ancor/>

LISA Labs



Today 4:00PM – 5:30PM

Alex Bardas: bardasag@ksu.edu

Related Work

- ***Automation Solutions***
 - Automating instance management *e.g.*, AWS OpsWorks
 - Deploying/migrating applications on different cloud providers *e.g.*, Cliqr, Cloud Velocity, CloudSwitch
 - Managing and automating instances deployment *e.g.*, Right-Scale, Service-Now
- ***Abstraction Approaches*** (PaaS specific)
 - Windows Azure Service Definition Schema (.csdef), Google AppEngine YAML-based specification
- ***Managing Infrastructure*** (support CMT integration)
 - OpenStack Heat, AWS CloudFormation, Terraform

More Related Work

- Docker container-based solutions:
 - Maestro-NG, Flynn, Deis, OpenShift, *etc.*



- Ubuntu Juju:
 - Works at a similar abstraction level





ANCOR vs. JUJU

- Similarities:
 - Work at a similar abstraction level
 - Have a way of capturing the dependencies between software applications (services)
- Differences:
 - Using existent CMT modules and workflow:
 - ANCOR: minimal changes
 - Juju: usually, significant changes (integration with Juju Tools)
 - Dependent services
 - ANCOR: more “centralized” management scheme
 - Juju: negotiation scheme
 - Current feature sets *e.g.*, OS support