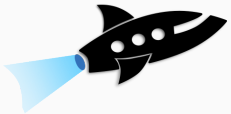


Efficient configuration and reconfiguration of distributed software systems

Hélène Coullon (STACK), *Christian Perez (AVALON)*
Dimitri Pertin (Postdoc), Maverick Chardet (PhD)

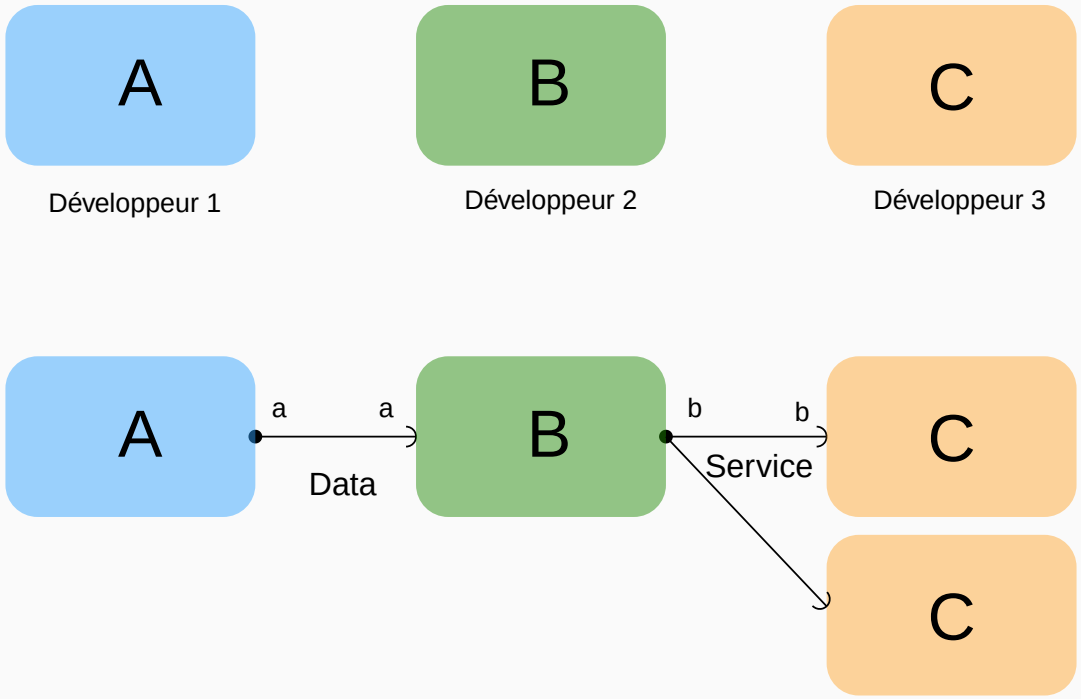
10/07/201

Inria
inventors for the digital world

Outline

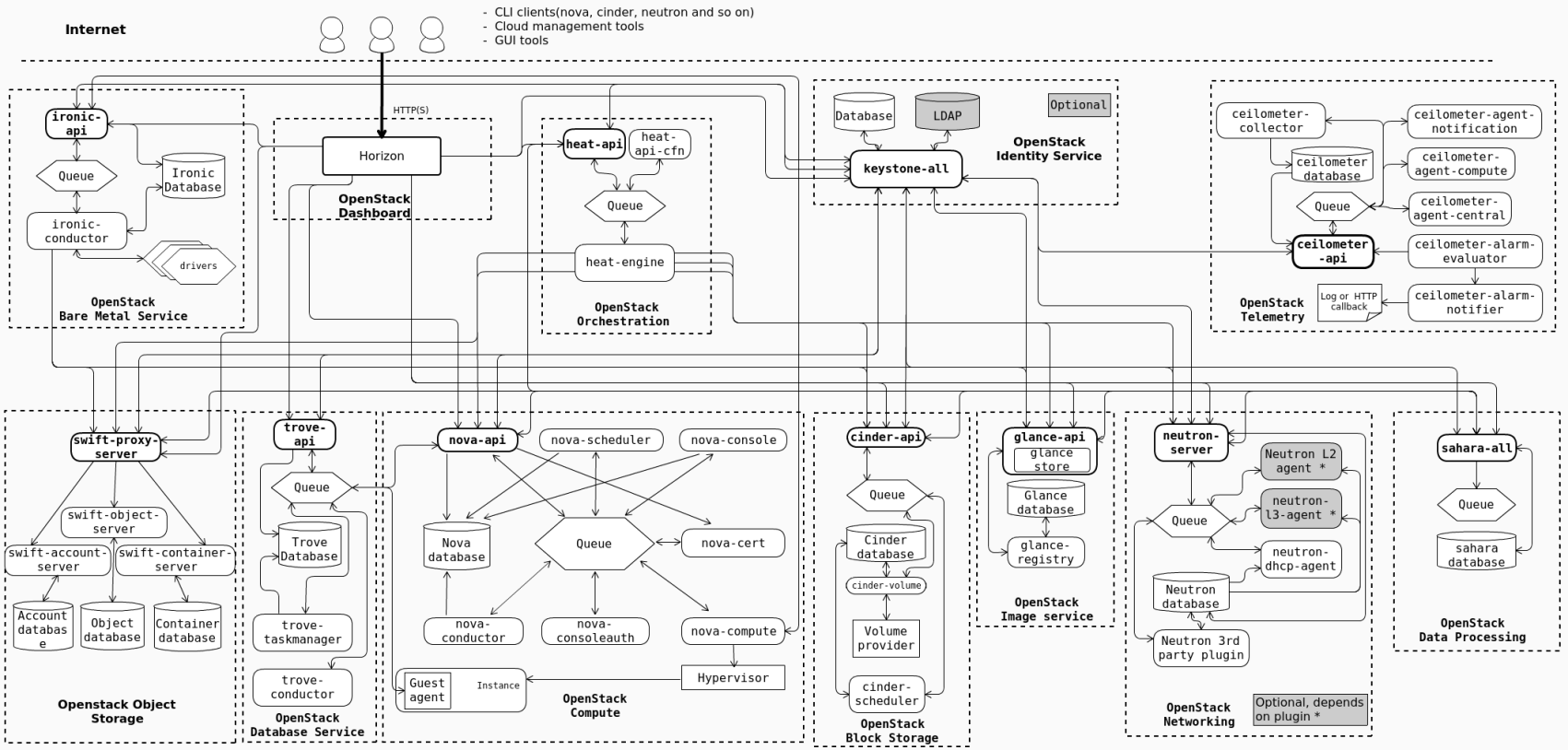
Context

Distributed software



Web applications, microservices, Apache/MySQL, MPI simulations, CORBA applications etc.

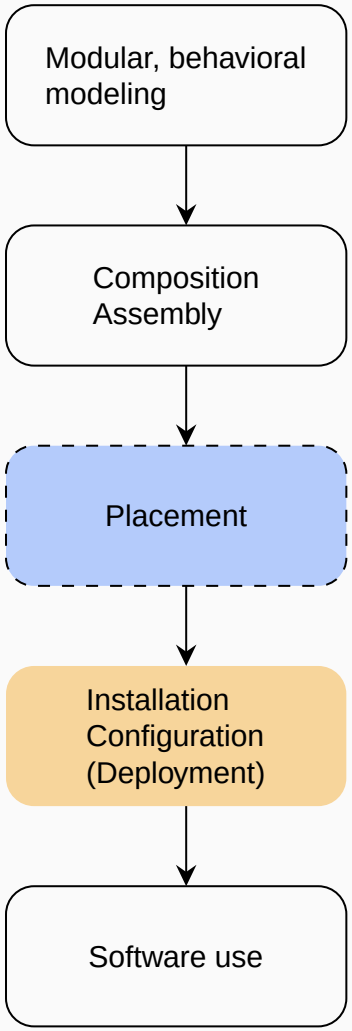
OpenStack (1/2)



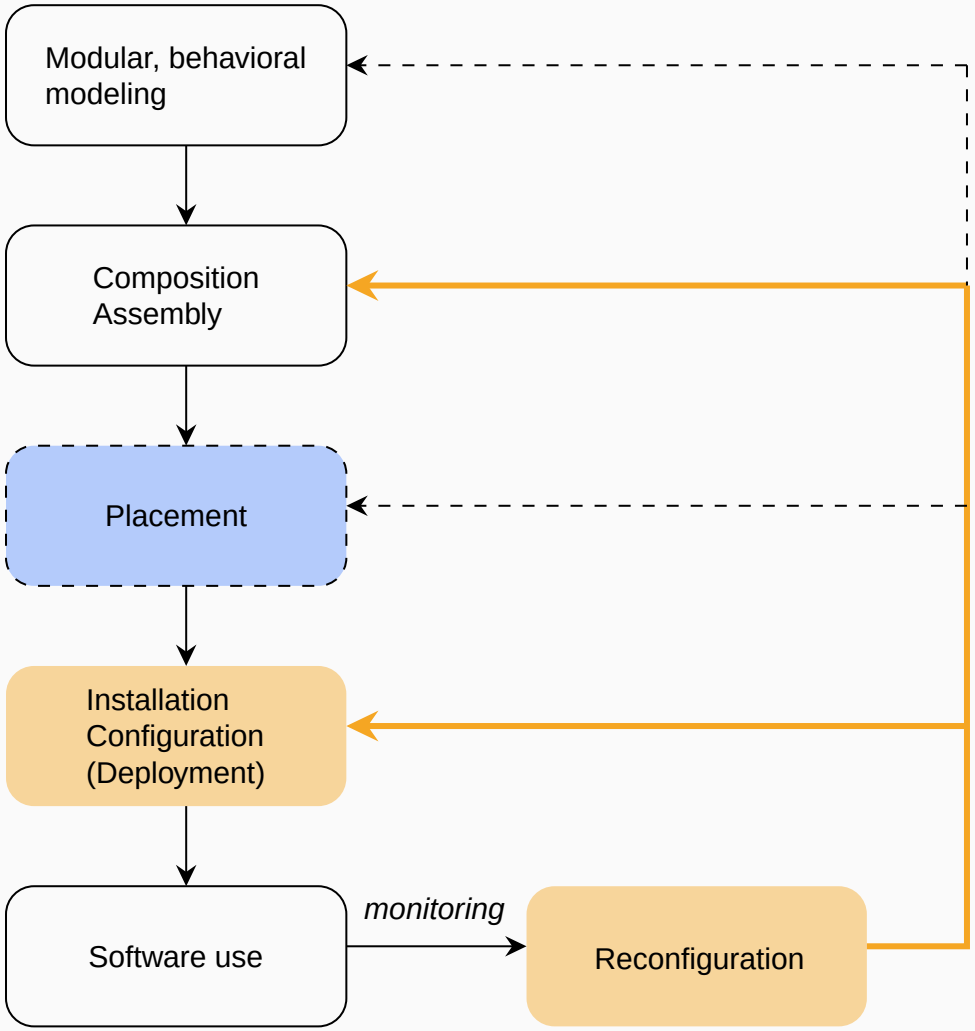
OpenStack (2/2)

- OpenStack is a modular large distributed software composed of more than 30 projects, gathering more than 186 services
- De-facto opensource solution to address the IaaS level of the Cloud paradigm
- 13 million lines of code in six years
- Its community has gathered more than 150 organizations (e.g., Google, IBM, Intel)
- OpenStack is installed and handled by operators of private Clouds, and infrastructure providers (OVH, CERN, RedHat etc.)

From the design to the usage of distributed software



From the design to the usage of distributed software



Challenges

Issues

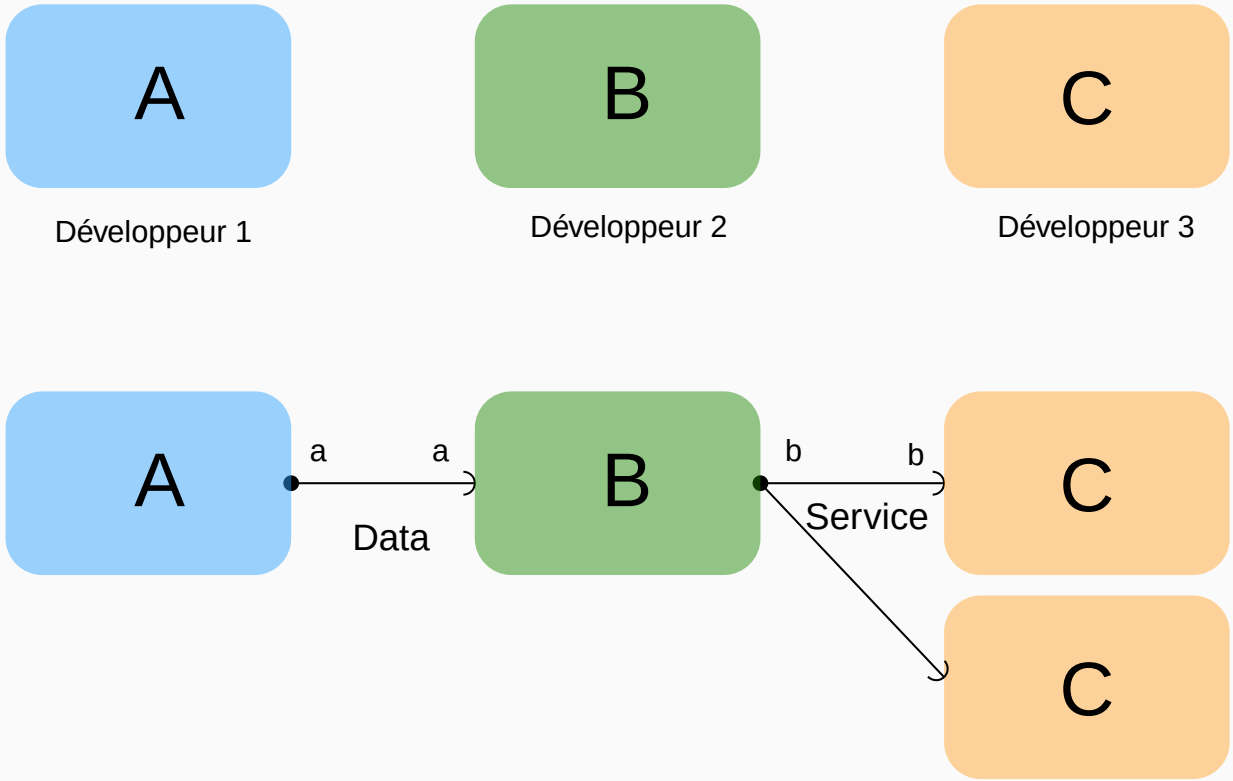
- Lack of formal safe and verified models
- Lack of parallelism and efficiency
- Limited (scaling, retry) or manually written reconfigurations

Goal

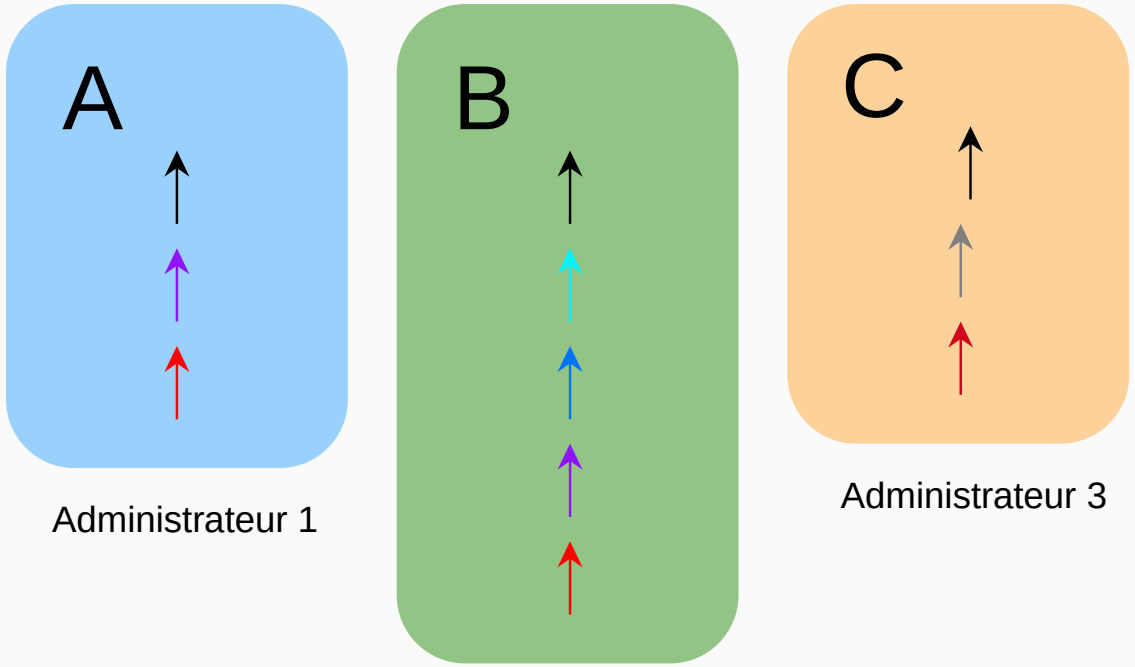
- Generic, safe and efficient language/model for distributed software configuration (deployment) and reconfiguration

Deployment with Madeus

Functional behavior of distributed software



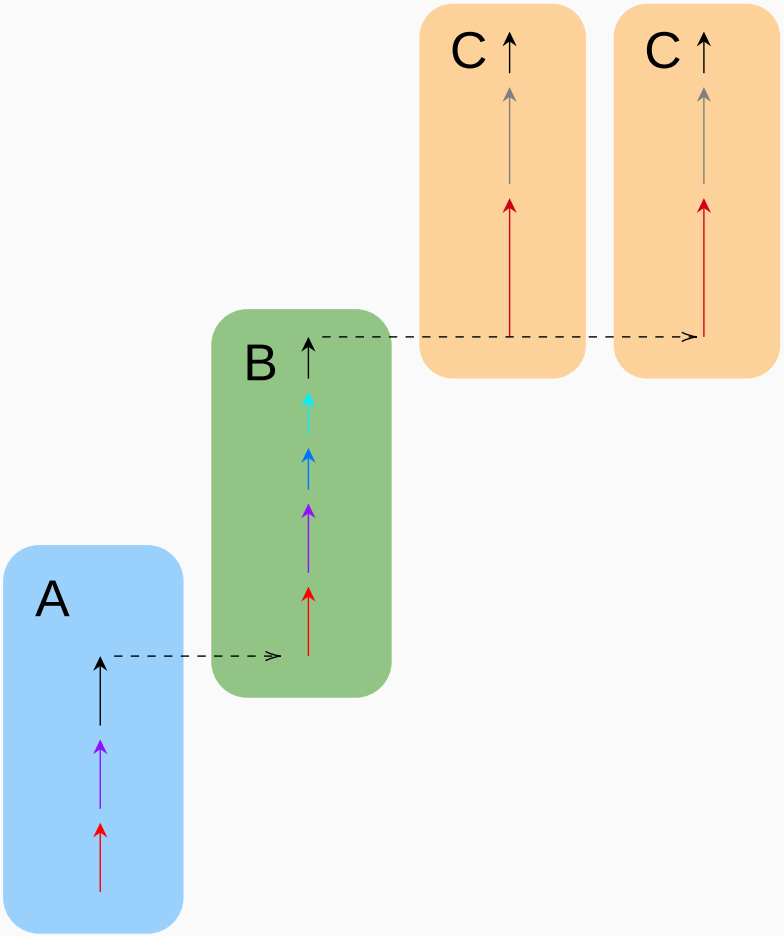
What about the installation/configuration ?



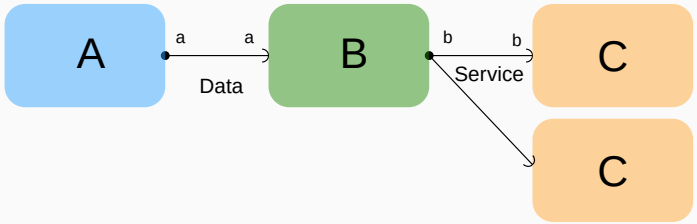
How to orchestrate/coordinate the installation/configuration of the three components?

Configuration execution (1/3)

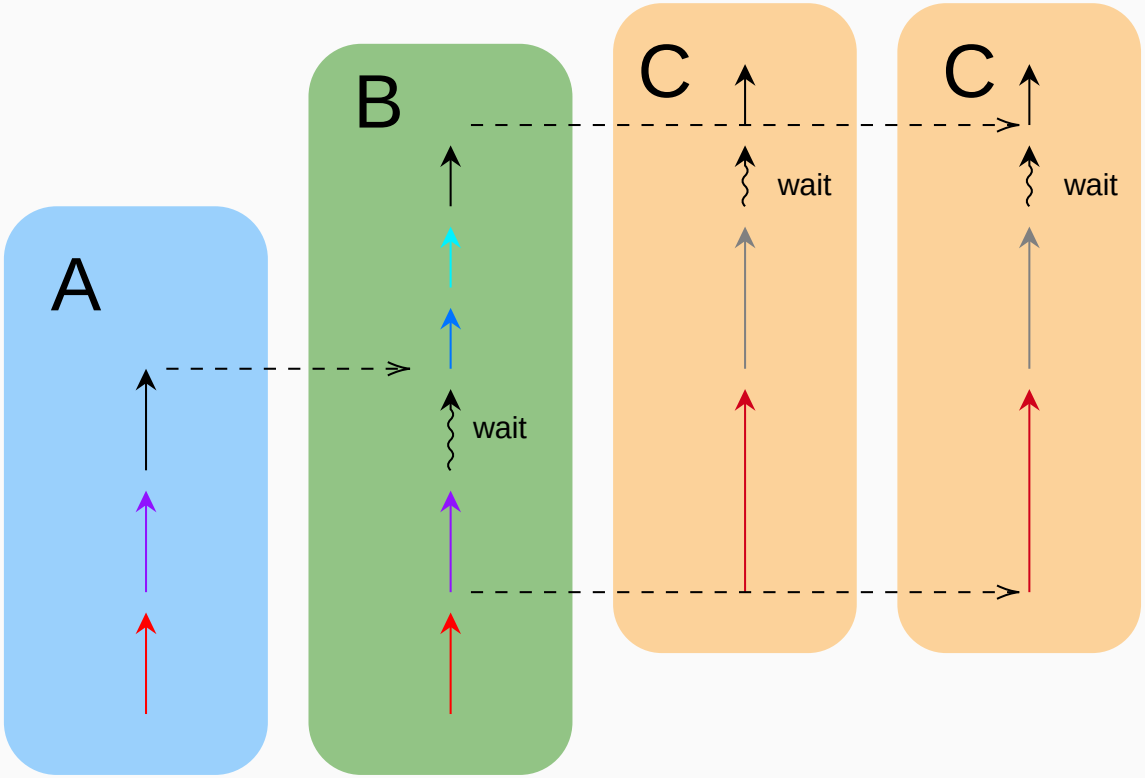
Configuration execution



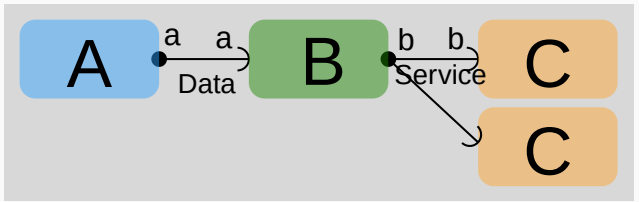
Behavioral model



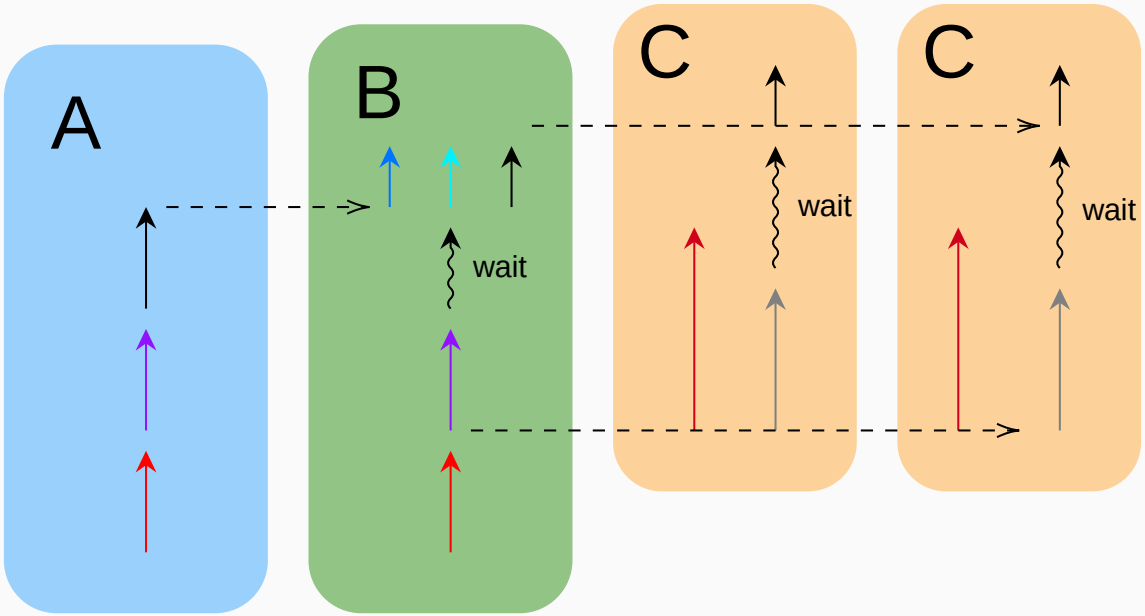
Configuration execution (2/3)



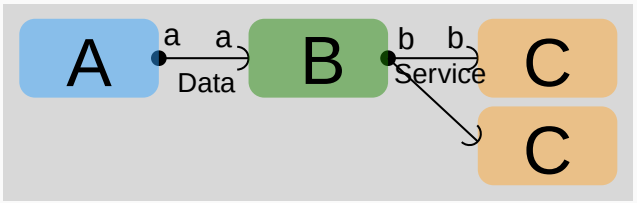
Not enough expressive



Configuration execution (3/3)

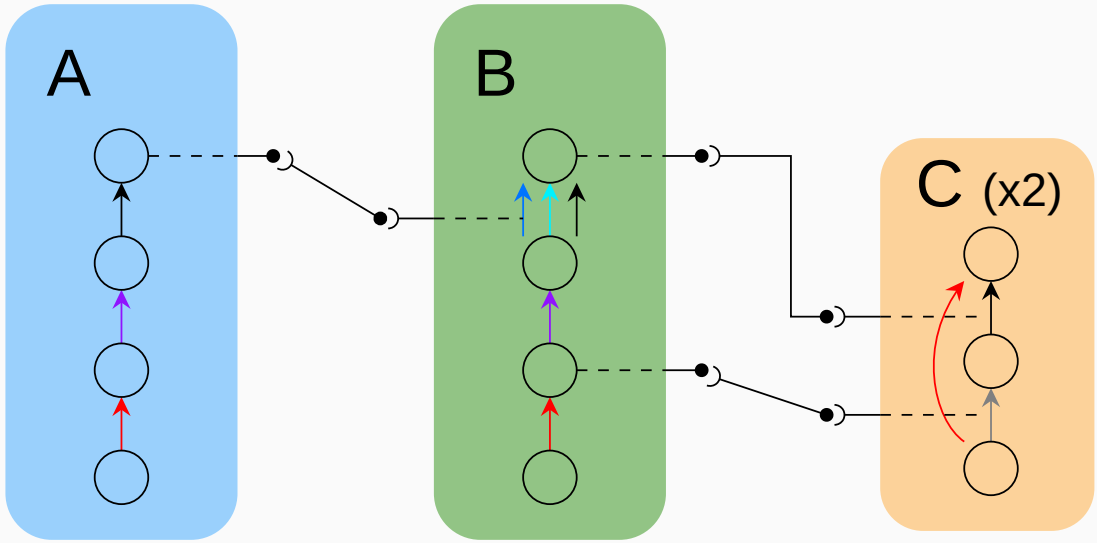


Not enough expressive



Madeus

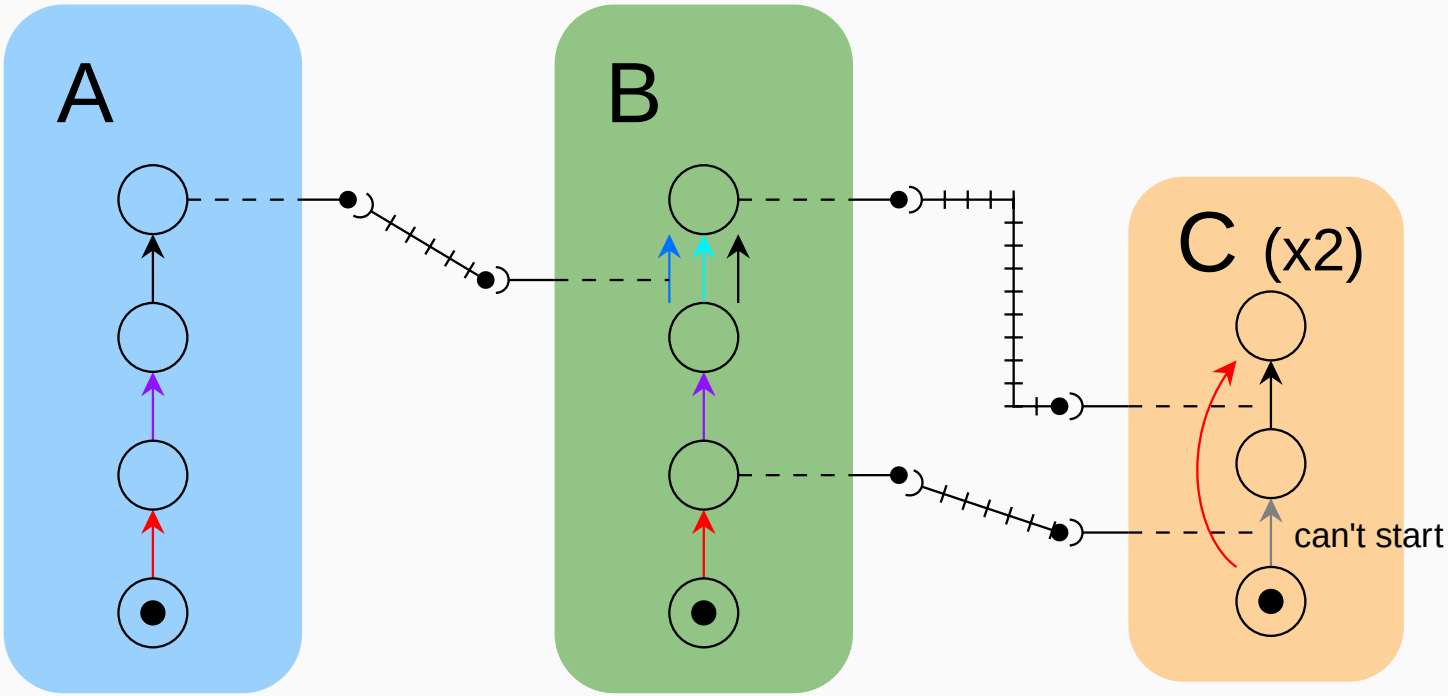
Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



- Publication: Maverick Chardet, Hélène Coullon, Christian Perez and Dimitri Pertin. *Madeus: A formal deployment model*. In 4PAD 2018 (hosted at HPCS 2018), Jul 2018, Orléans, France.

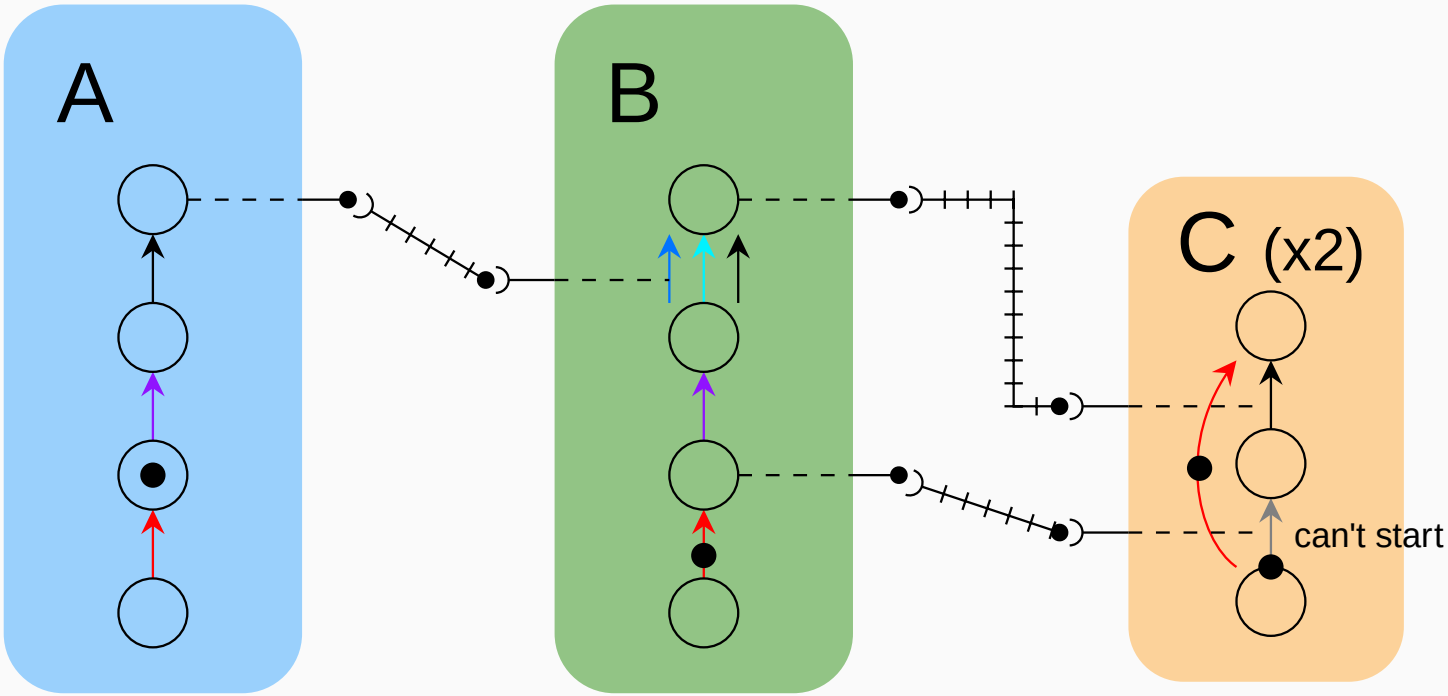
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



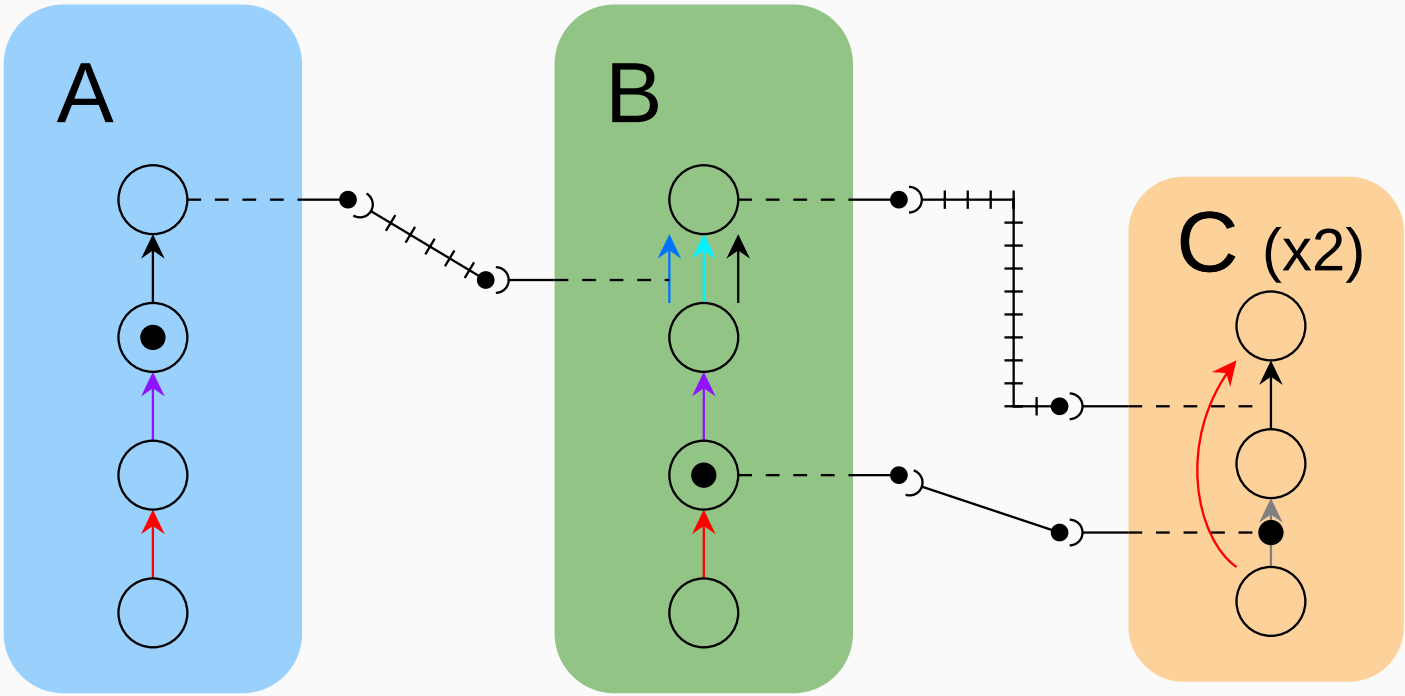
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



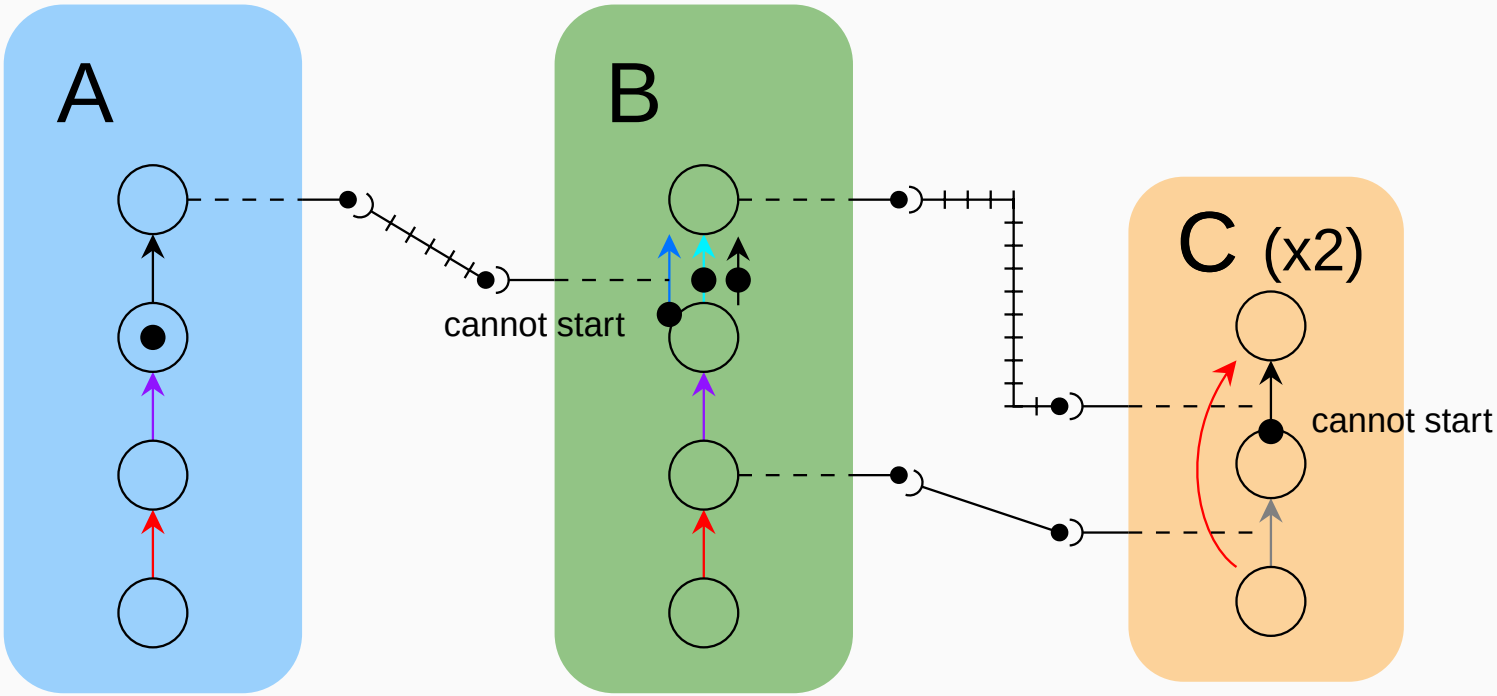
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



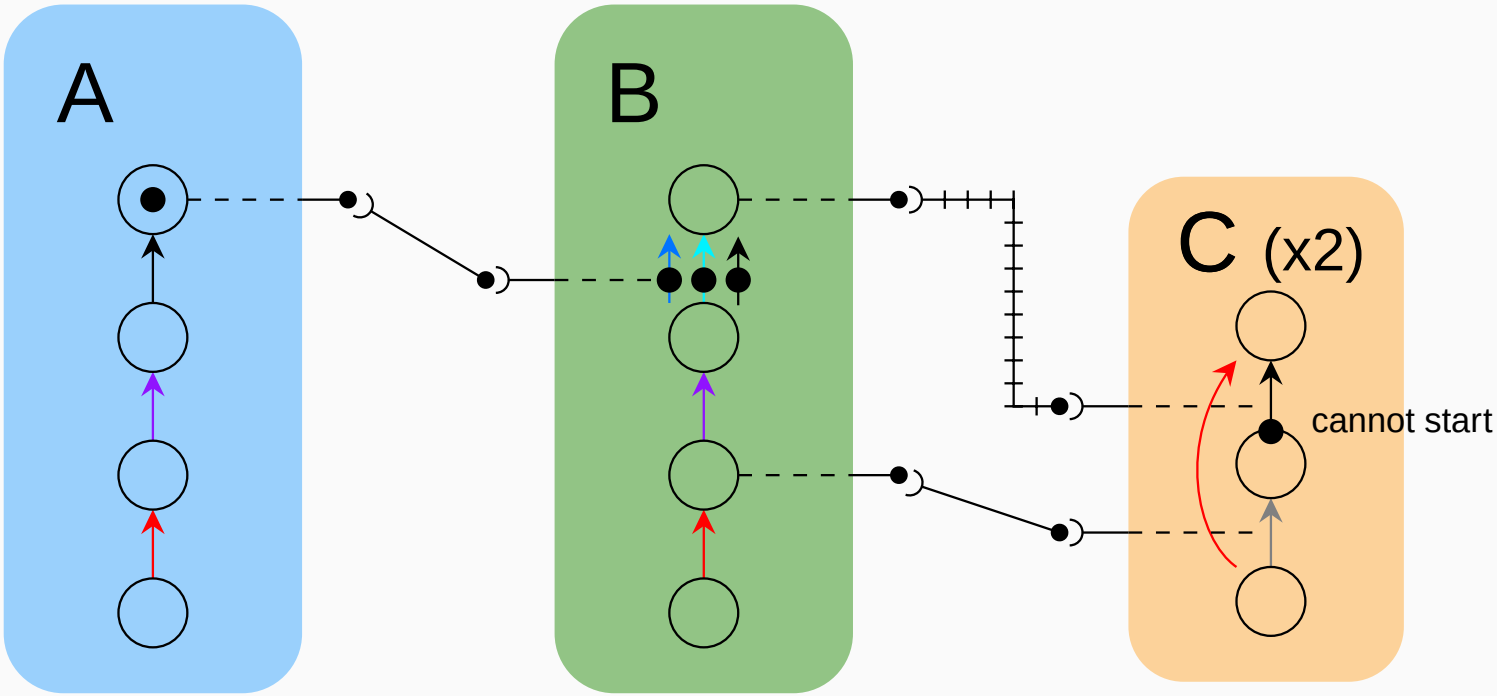
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



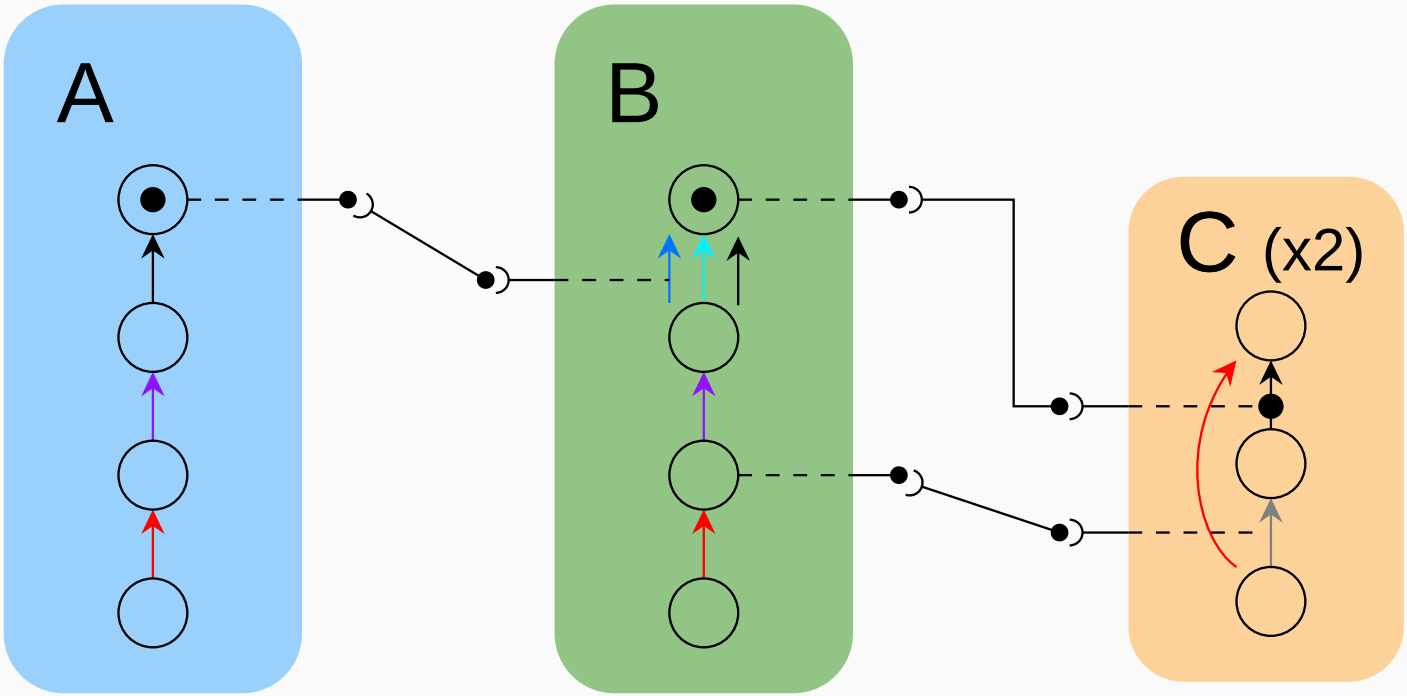
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures



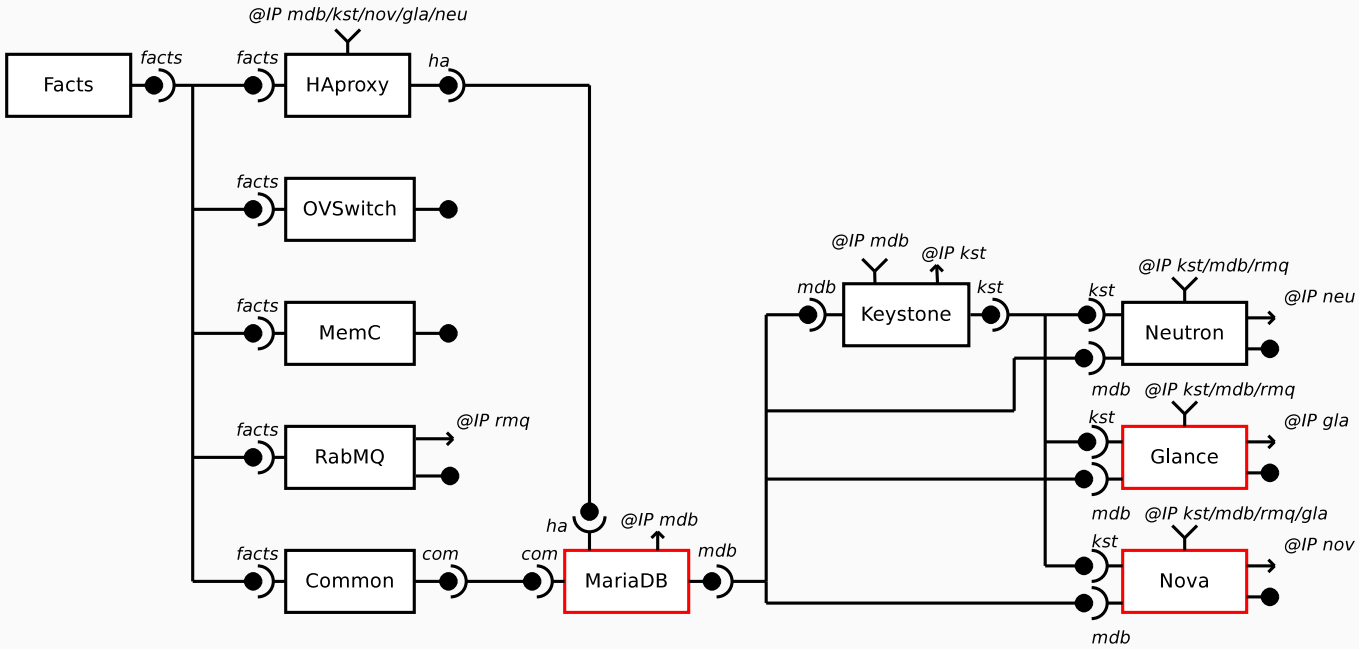
Madeus execution

Madeus is a formal model to adress the safe and efficient coordination of deployment procedures

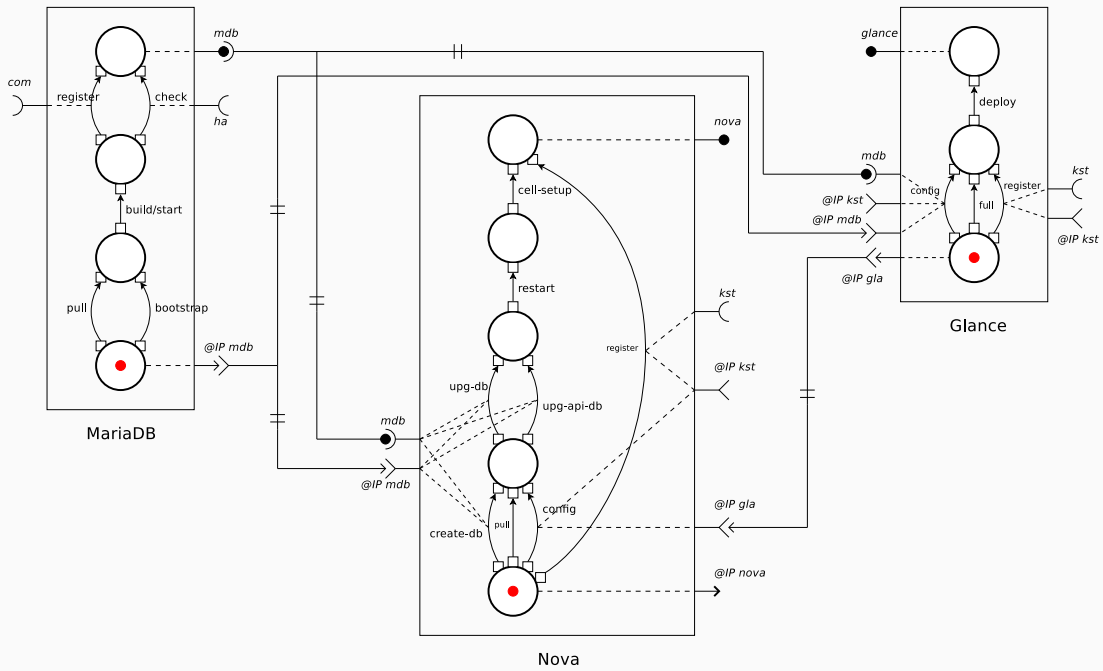


Experiments (1/3)

- Hélène Coullon, Christian Perez and Dimitri Pertin. *Production Deployment Tools for IaaSes: an Overall Model and Survey*. In FiCloud 2017, Aug 2017, Prague, Czech Republic.
- Kolla-ansible deployment of OpenSatck (36 services gathered in 11 components, deployed on three nodes)



Experiments (2/3)



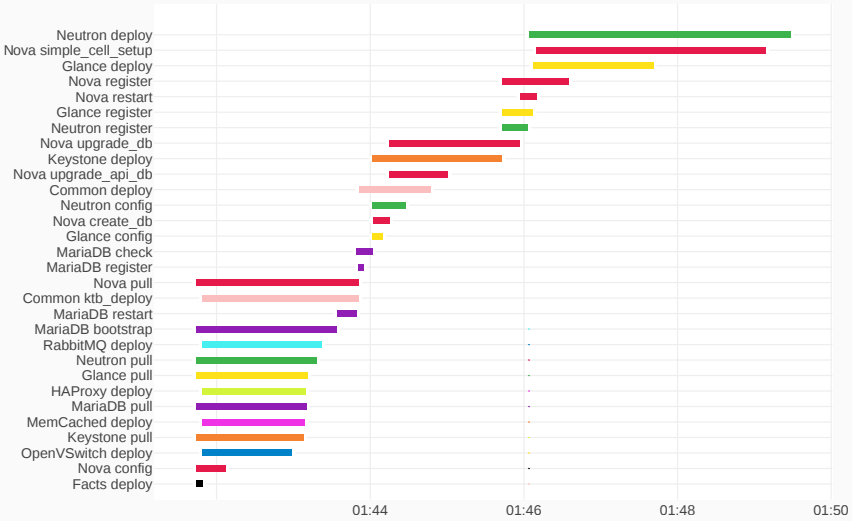
	Places	Trans.	Ports
Facts	2	1	1
Common	3	2	2
HaProxy	2	1	7
MemCached	2	1	2
MariaDB	4	5	4
RabbitMQ	2	1	3
Keystone	3	2	4
Glance	3	4	7
Nova	5	8	8
OpenVSwitch	3	1	2
Neutron	3	4	7
Total	32	30	47

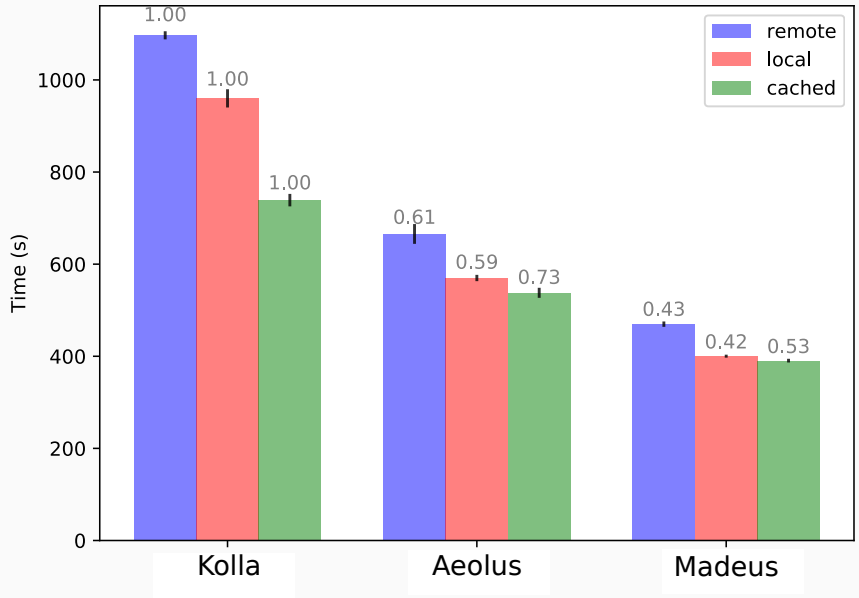
	Compute	Network	Control
# images	9	11	16
Total size (MB)	2767	2705	4916

Cluster	CPU	Memory	Network
Taurus (G5k Lyon)	2 x Intel XeonE5-2630 6cores/CPU	32GB	10Gbps

Experiments (3/3)

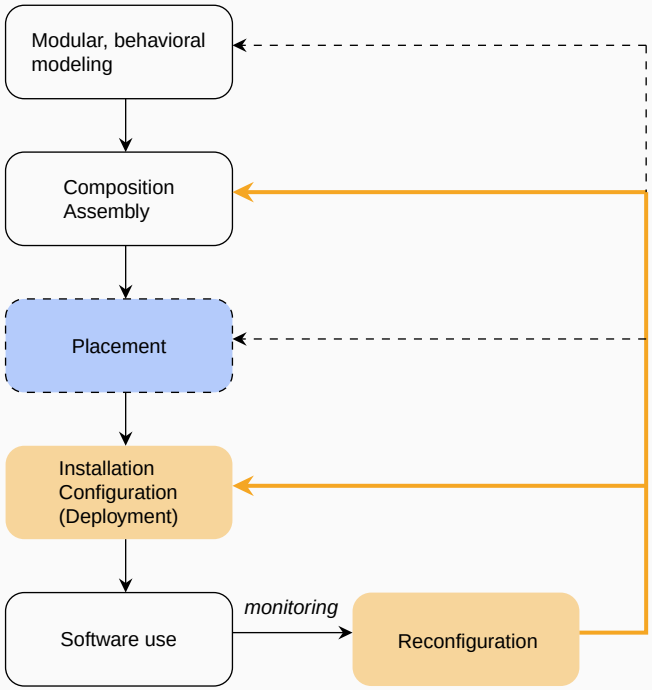
- Madeus prototype:
  `https://mad.readthedocs.io/en/latest/`
- Reproducible lab on OpenStack:
  `https://mad-openstack.readthedocs.io/en/latest/`
- Up to 58% gain compared to Kolla (less than 10 minutes)





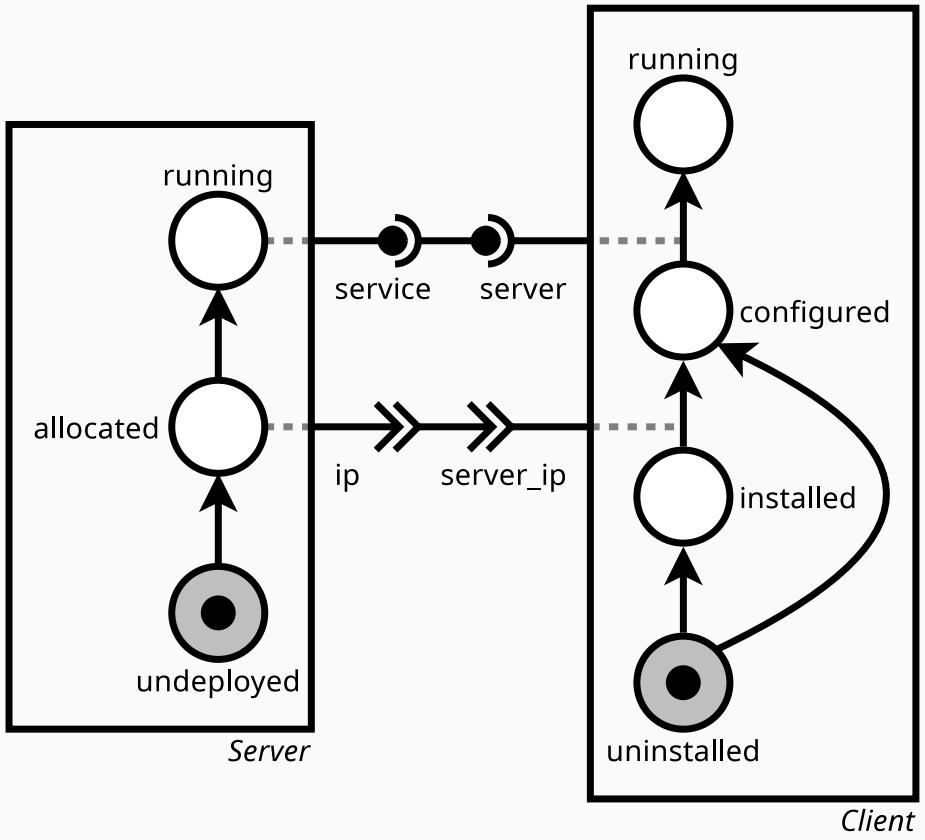
Reconfiguring with Concerto

Reconfiguration



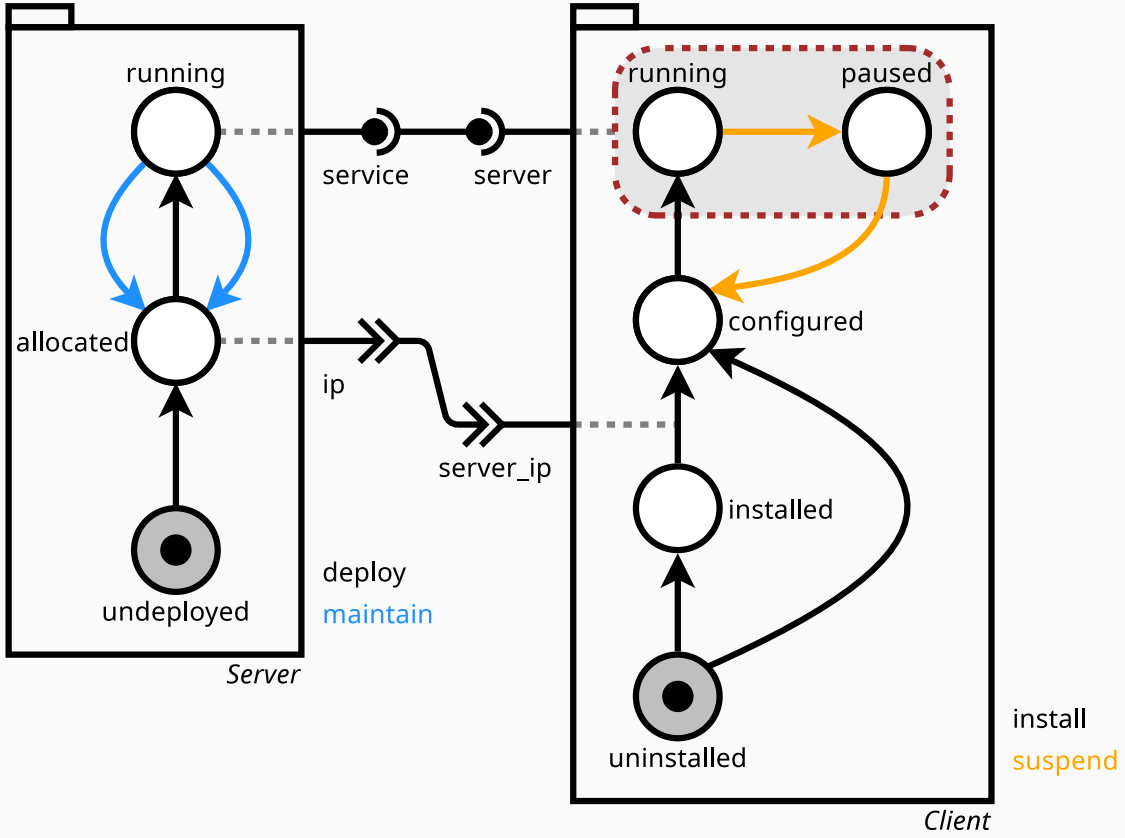
- Resources and placement changes
  - optimization
  - mobility
  - faults
- Software topology changes
  - external events
  - energy, security, sensors etc.
- Software update

Concerto



Concerto = Madeus + Behaviors + ScoreL language

Concerto

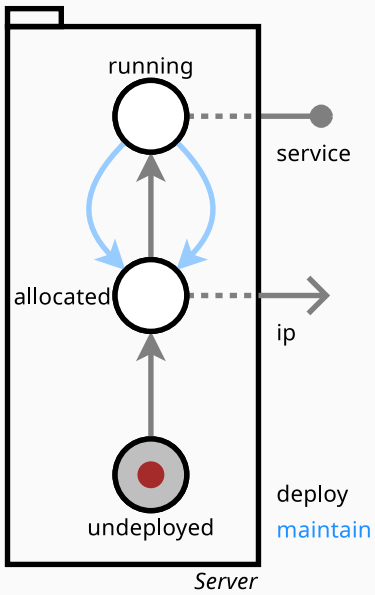


Concerto = Madeus + Behaviors + ScoreL language

Concerto - Deployment example

Example (Deployment)

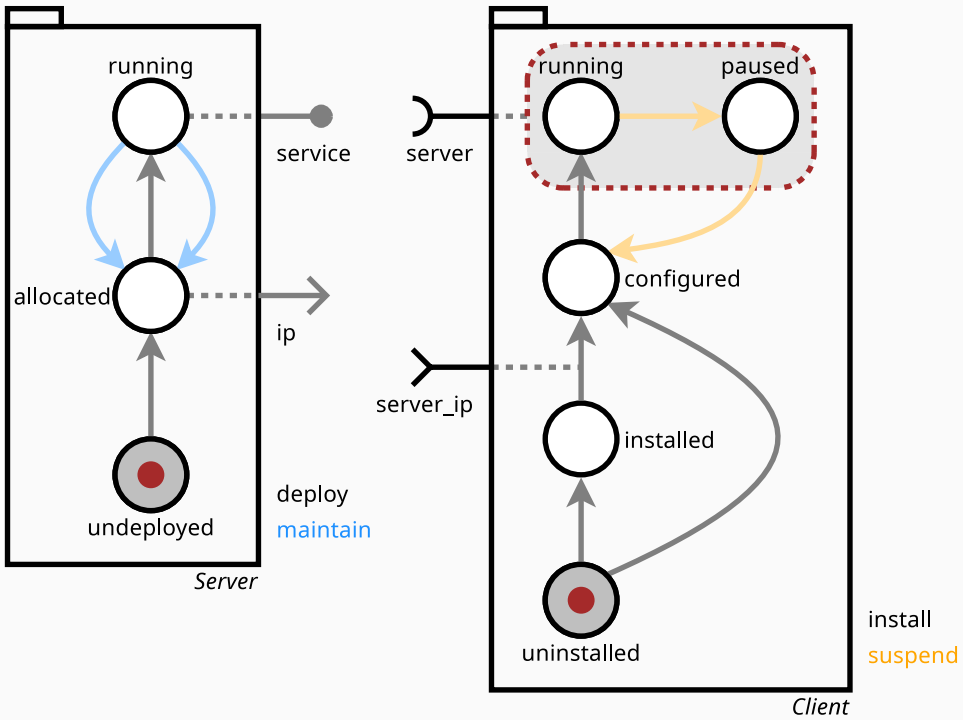
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

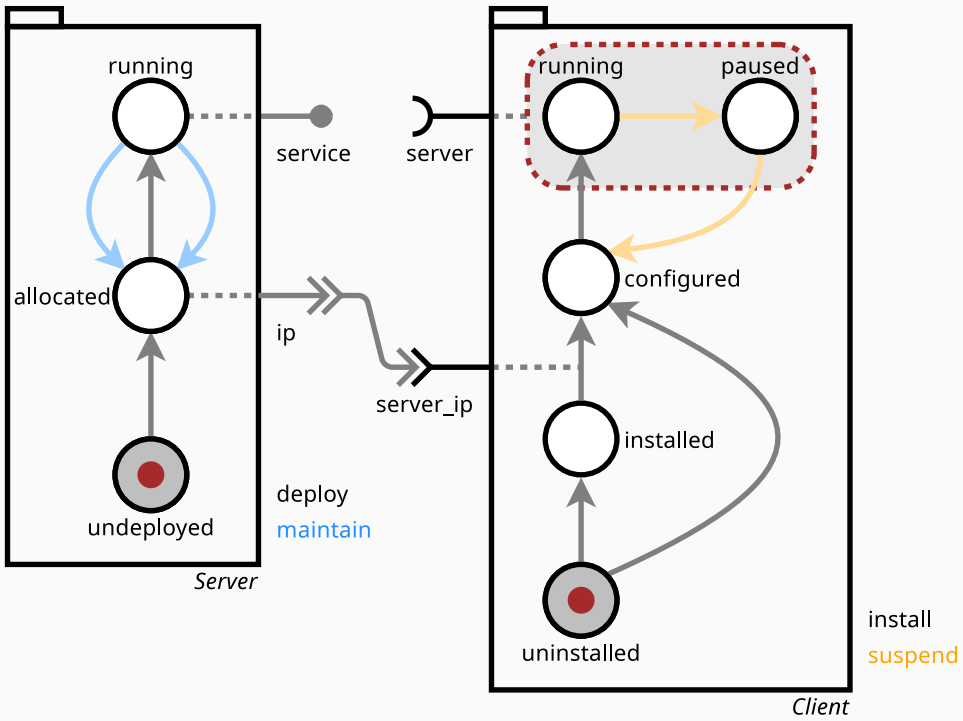
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

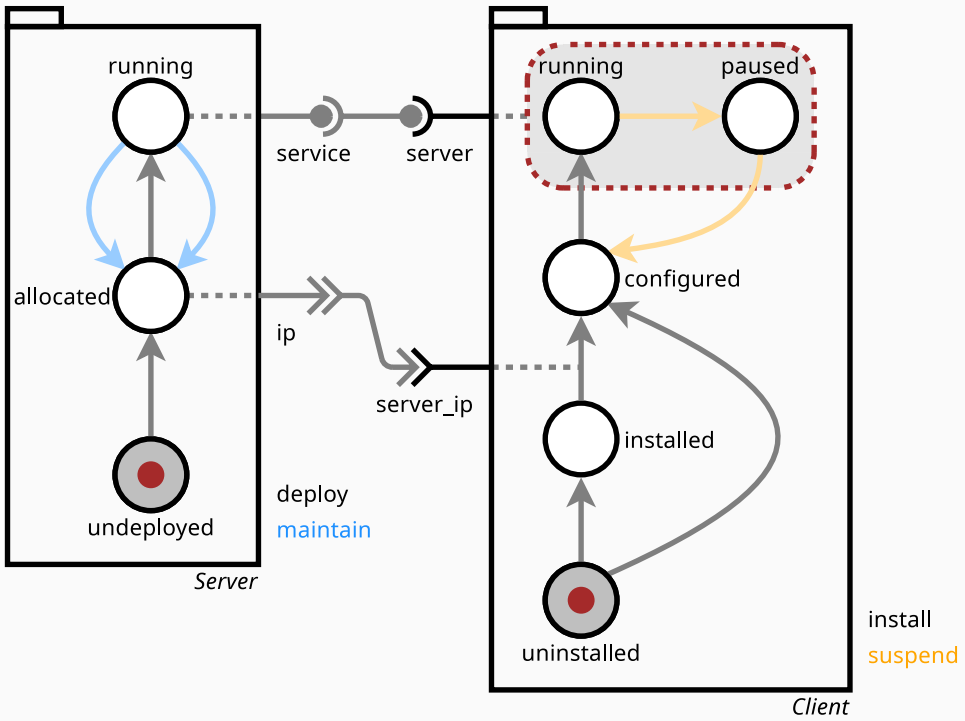
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

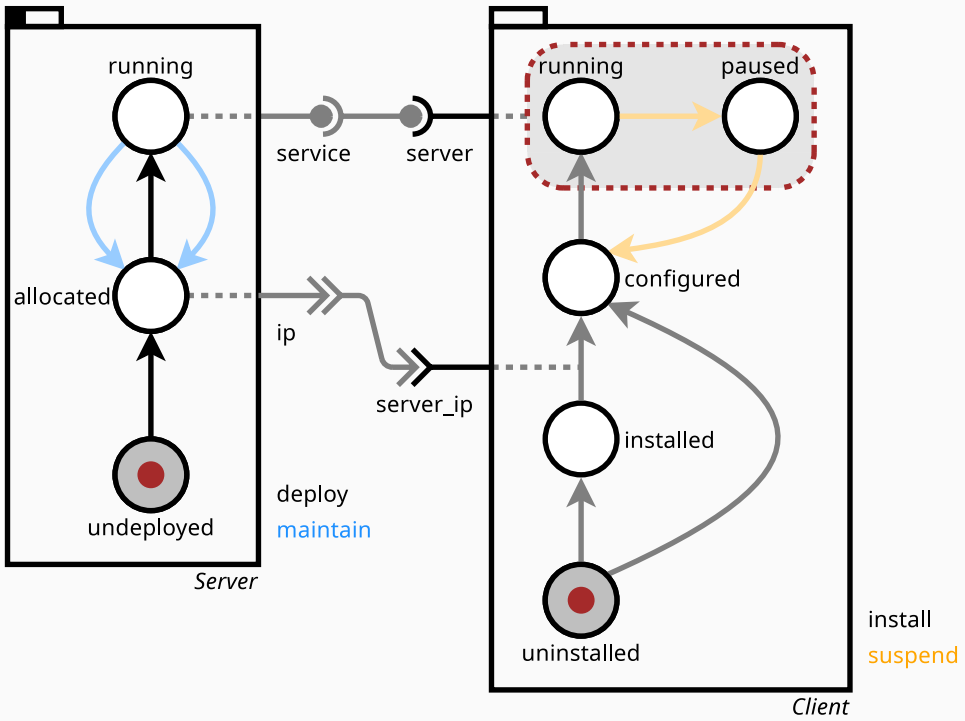
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

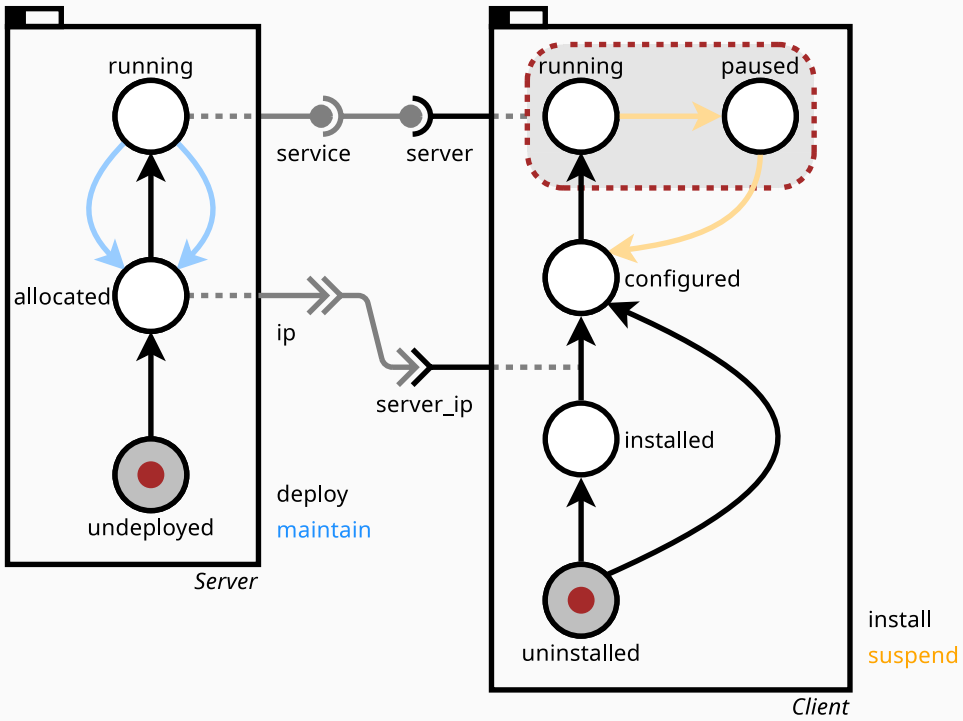
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

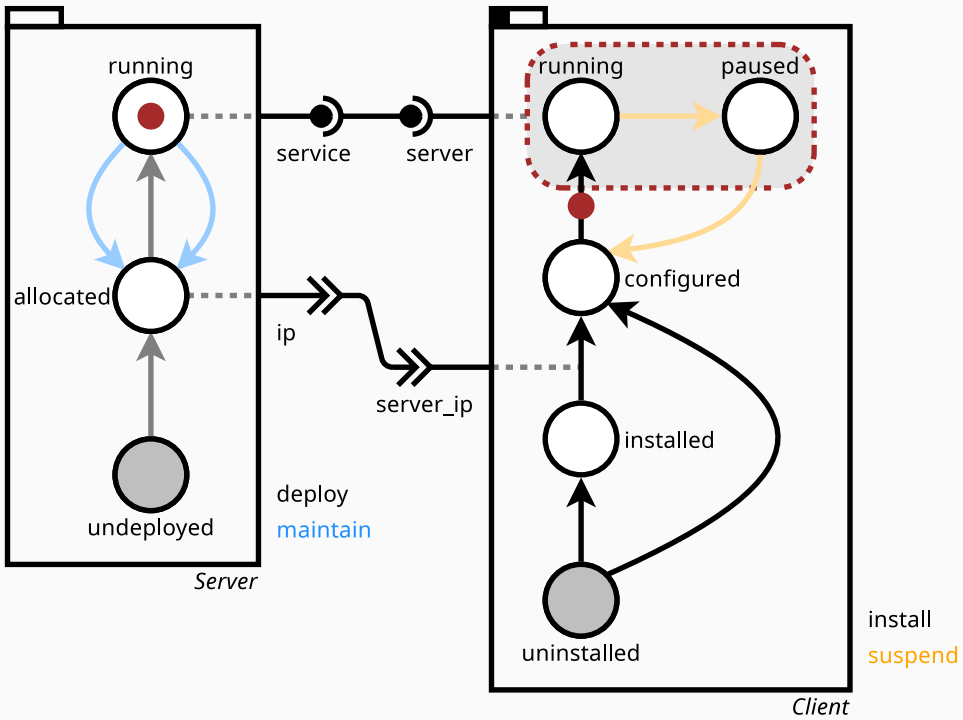
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

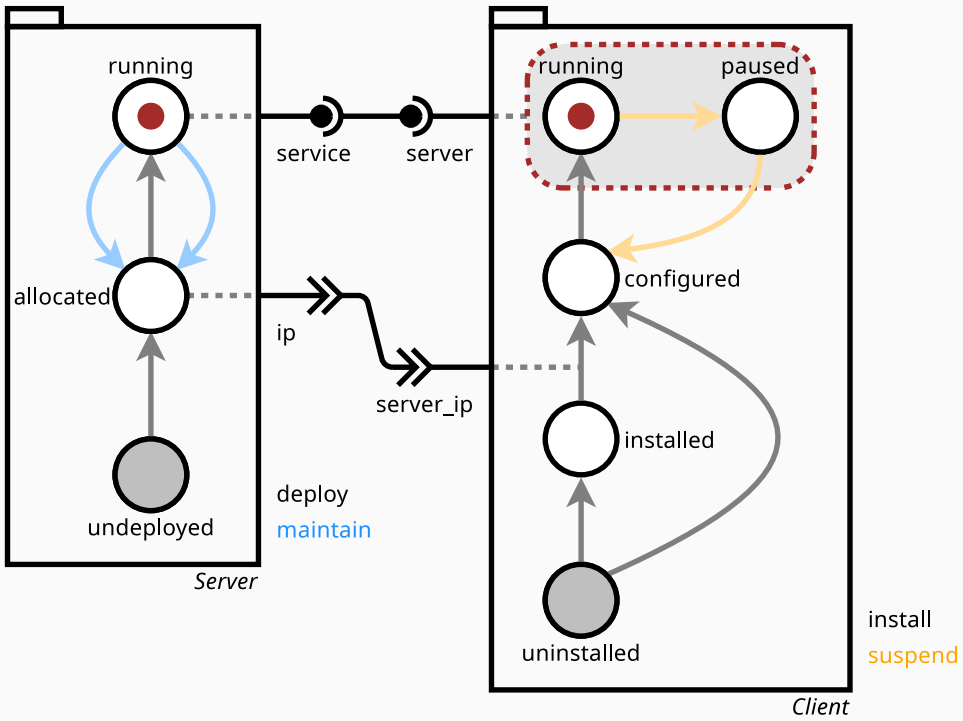
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Deployment example

Example (Deployment)

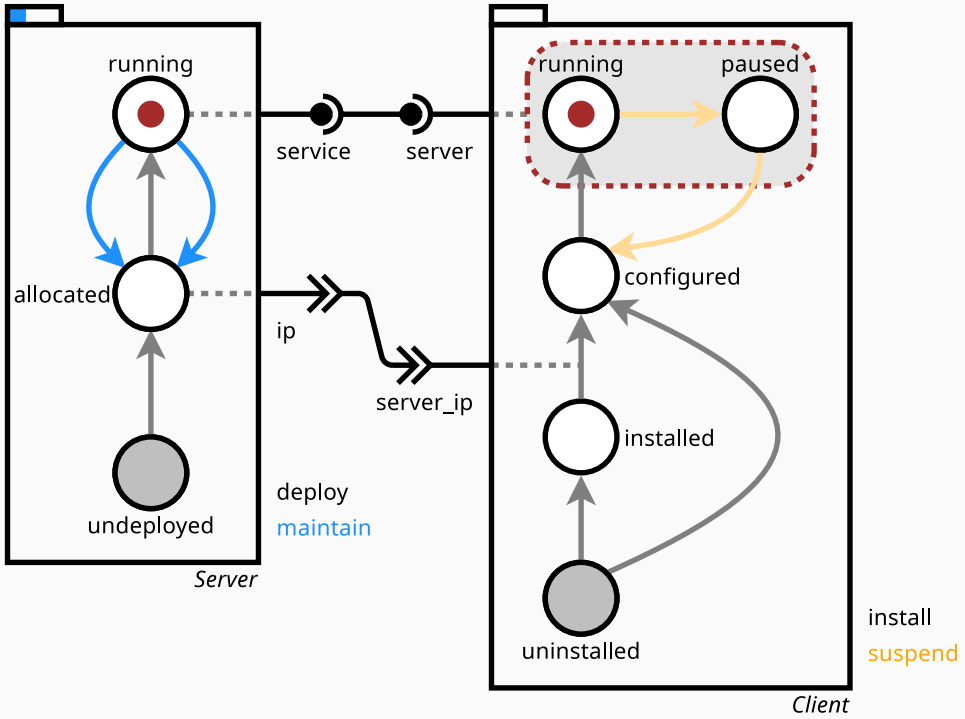
```
add(s : Server)
add(c : Client)
con(s.ip, c.server_ip)
con(s.service, c.server)
pushB(s, deploy)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

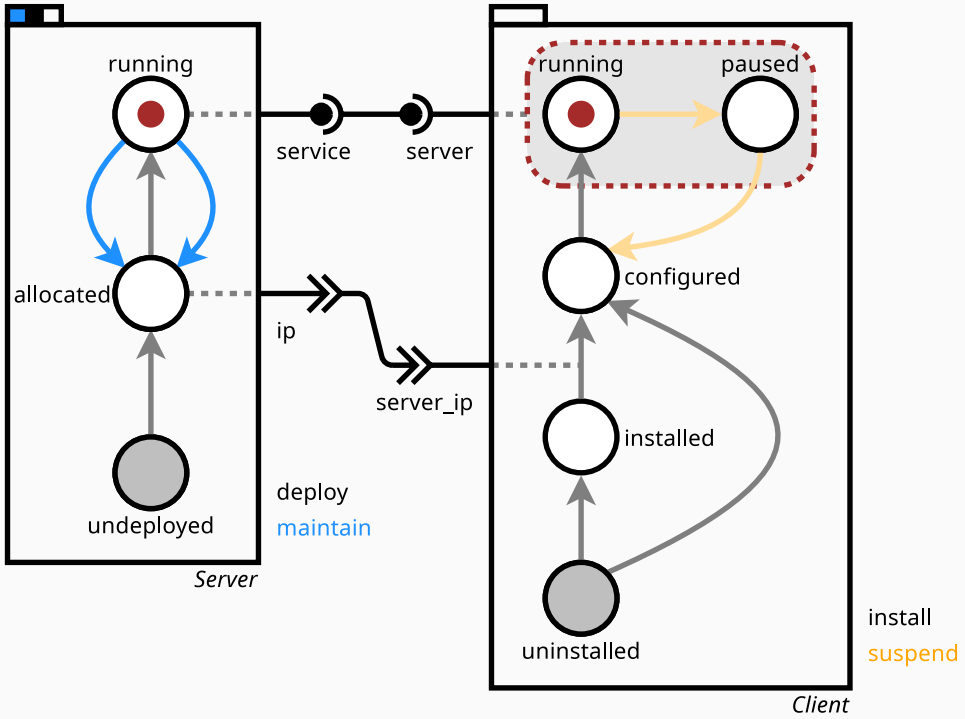
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

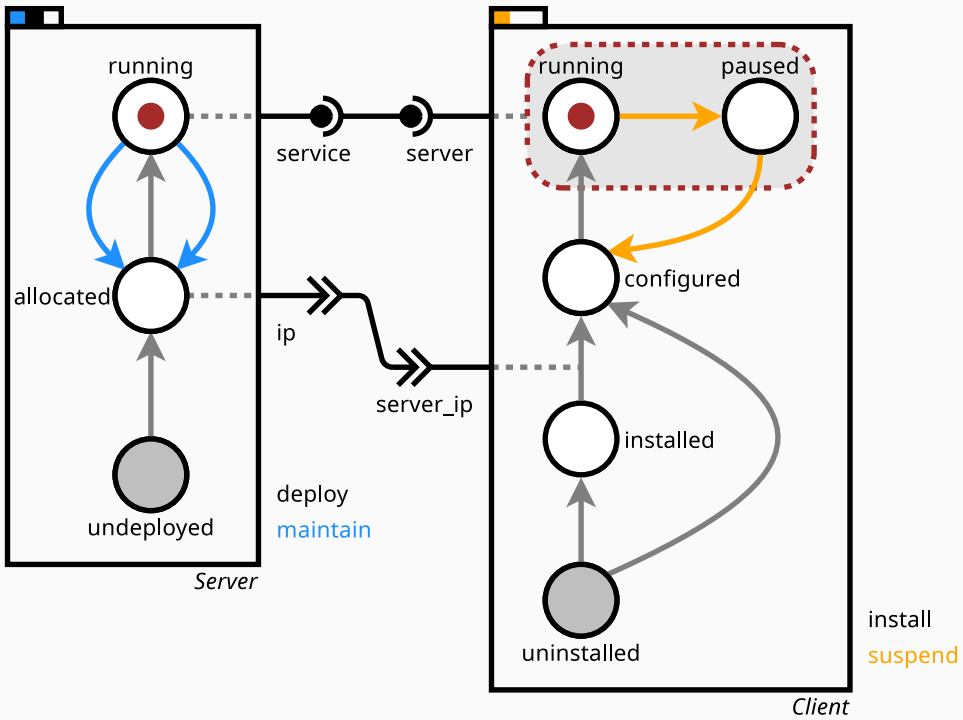
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

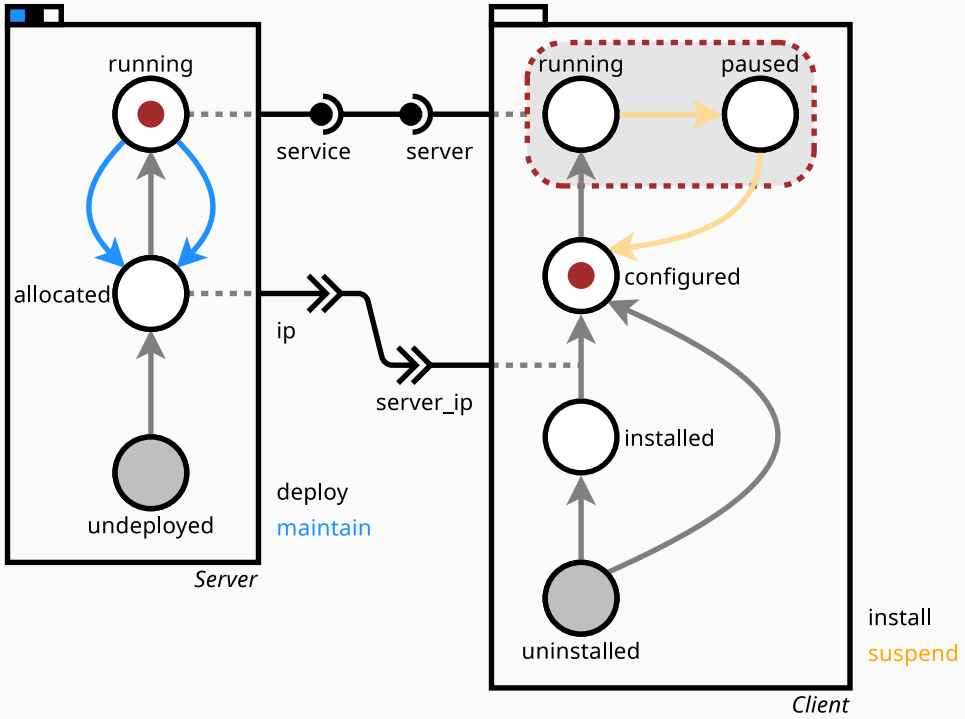
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

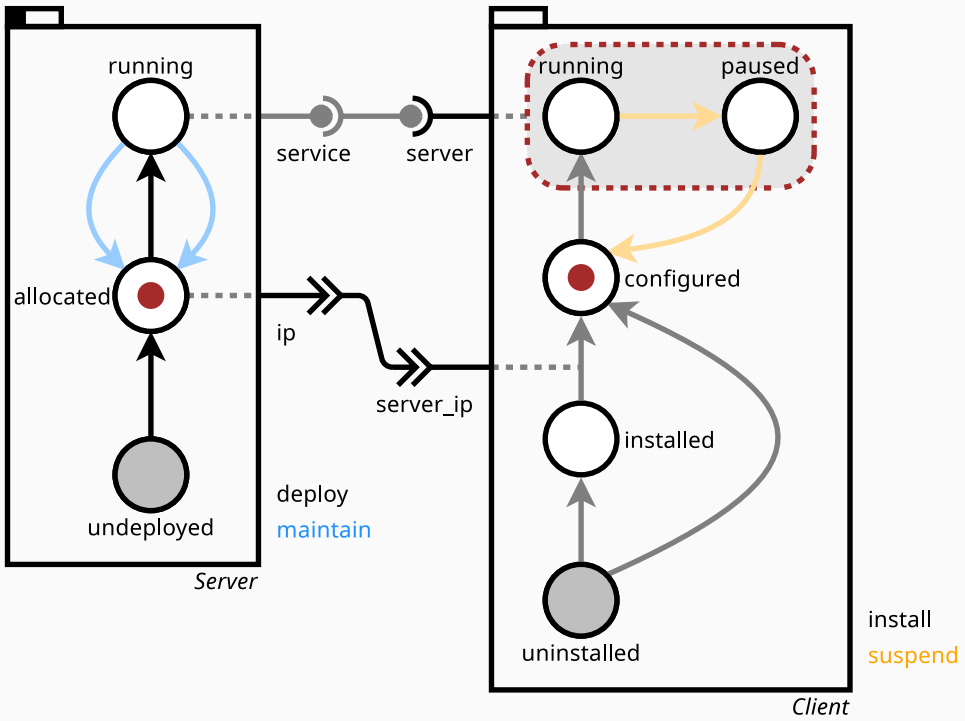
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

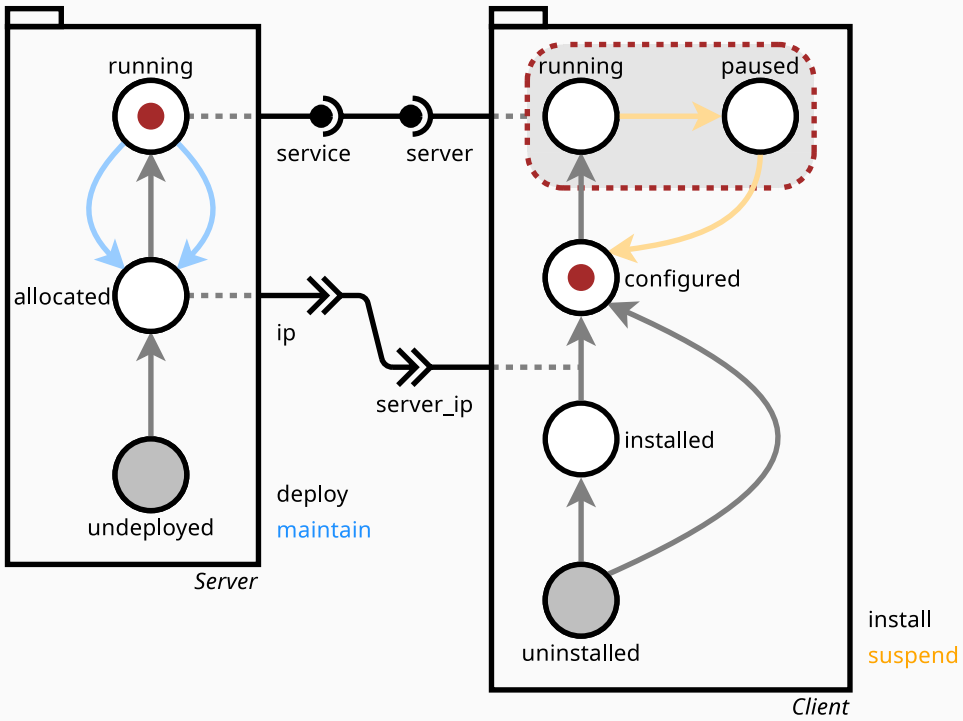
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

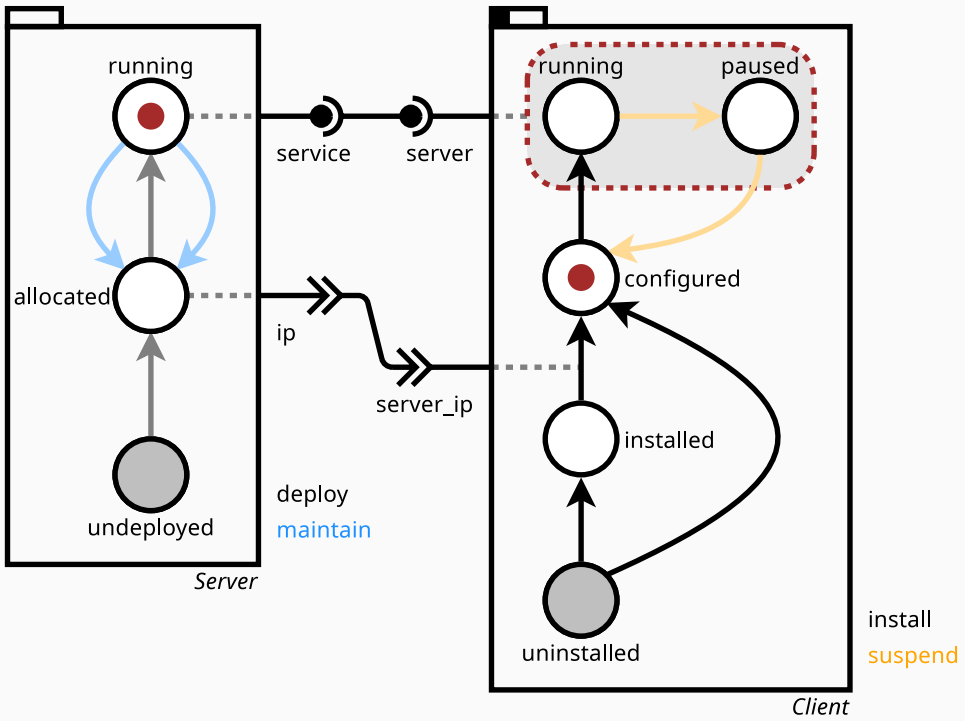
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

Example (Maintenance)

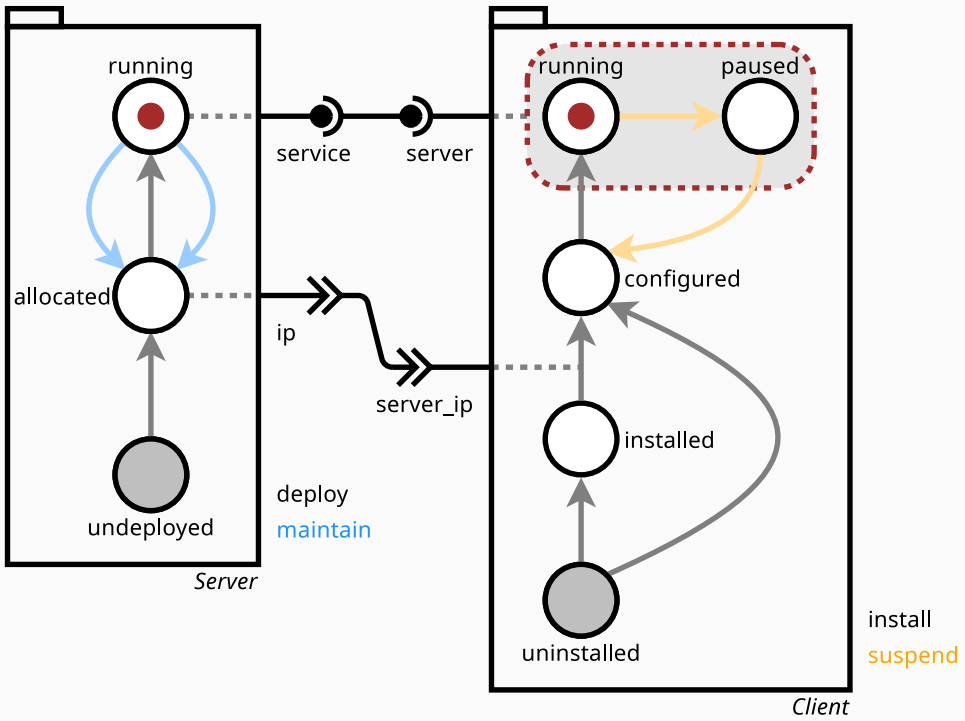
```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Concerto - Maintenance example

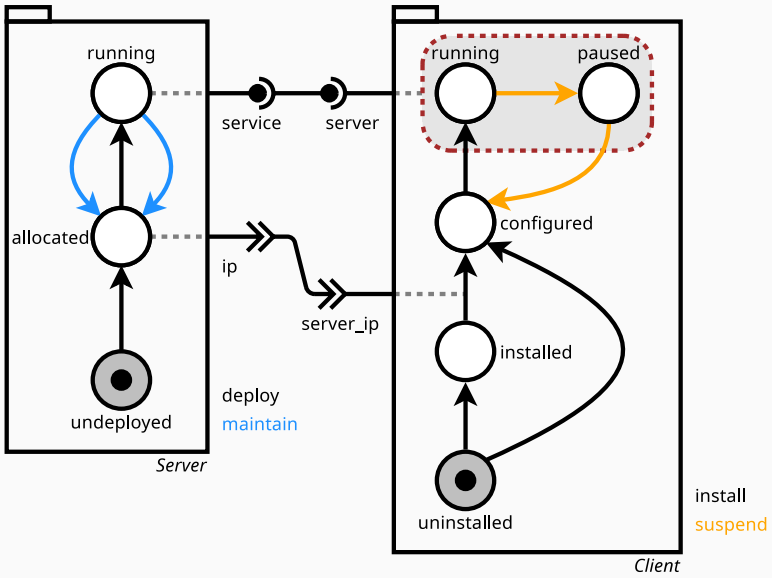
Example (Maintenance)

```
pushB(s, maintain)
pushB(s, deploy)
pushB(c, suspend)
wait(s)
pushB(c, install)
wait(c)
```



Experiments - client/server (1/2)

- Prototype : `https://gitlab.inria.fr/mchardet/madpp`



- Scenario A: Deployment
- Scenario B: Maintenance
- Simulated transitions
  - uniformly randomly chosen
  - between 0 and 10 seconds
- 300 runs

Experiments - client/server (2/2)

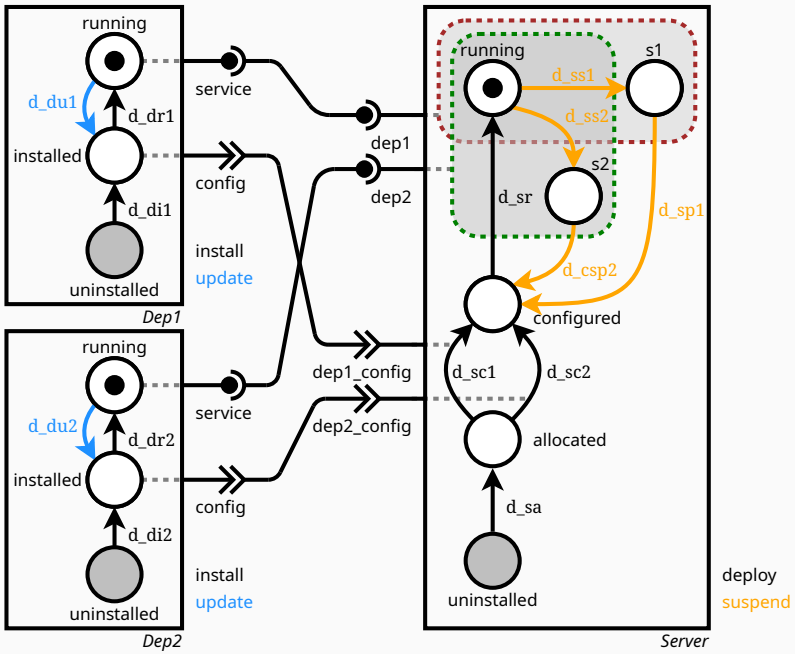
- Overhead = % difference between
  - theoretical performances
  - obtained performances with the Concerto prototype

	Average execution time	Average overhead
A	17.1 s	0.3 %
B	18.3 s	0.25 %

- Theoretical gain compared to Aeolus and Ansible

	Aeolus			Ansible		
	Min	Max	Average	Min	Max	Average
A	0 %	45.83 %	16.1 %	10.28 %	62.65 %	42.35 %
B	0 %	38.25 %	11.27 %	11.69 %	57.76 %	38.75 %

Experiments - server/deps (1/2)



- Scenario A: Deployment
  - 1 dependence
  - 5 dependencies
  - 10 dependenices
- Scenario B: Update
  - 1 dependence
  - 5 dependencies
  - 10 dependencies
- Simulated transitions
  - uniformly randomly chosen
  - between 0 and 10 seconds
- 160 runs

Experiments - server/deps (2/2)

- Overhead = % difference between
  - theoretical performances
  - obtained performances with the Concerto prototype

	Average execution time	Average overhead
A	19.62 s	0.27 %
B	21.16 s	0.23 %

- Theoretical gain compared to Aeolus and Ansible

	Aeolus		Ansible	
	Max	Average	Max	Average
A				
1 dep	0 %	0 %	49.84 %	10.65 %
5 deps	59.52 %	12.75 %	72.19 %	19.79 %
10 deps	75.44 %	20.87 %	80.14 %	23.89 %
B				
1 dep	0 %	0 %	49.3 %	10.64 %
5 deps	70.97 %	19.8 %	76.82 %	23.06 %
10 deps	84.44 %	25.64 %	86.42 %	26.8 %

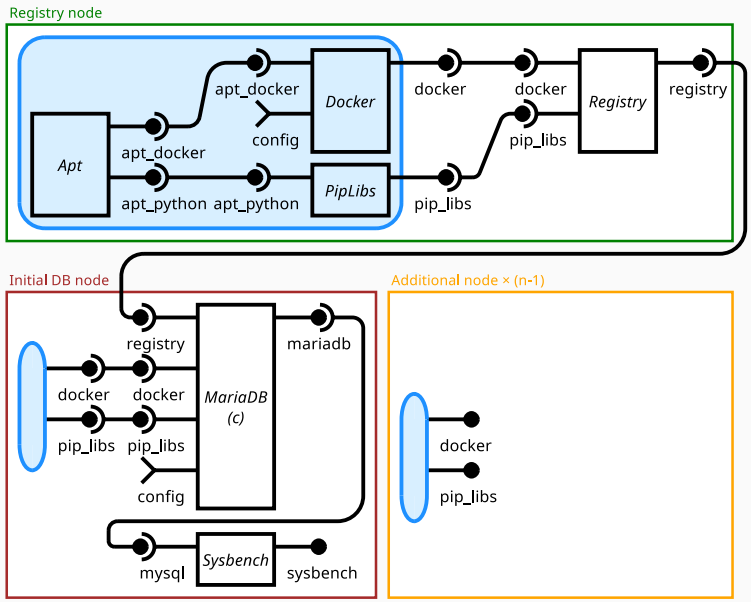
Experiments - decentralized DB (1/4)

Use-case

- From centralized MariaDB to decentralized MariaDB
- Galera cluster of MariaDBs (see the OpenStack summit)
- Reference code written with Ansible (see Juice)

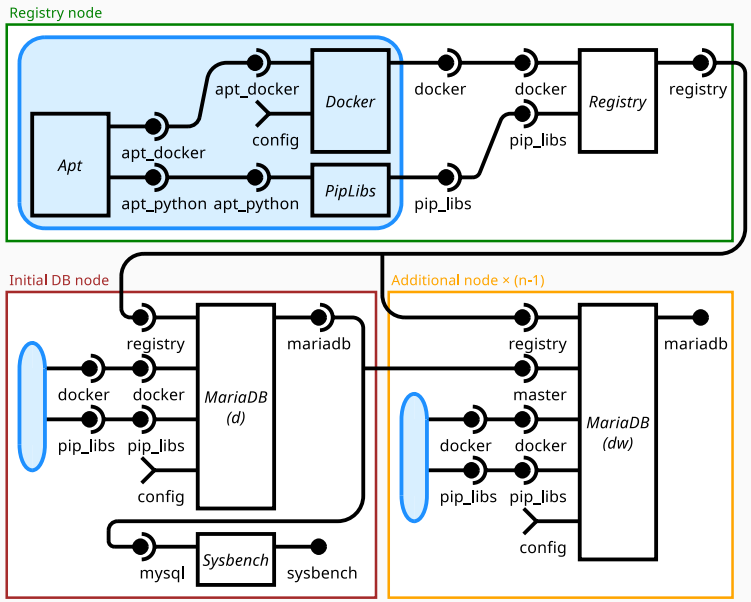
Experiments - decentralized DB (2/4)

- Deployment: Docker registry, MariaDB
- Reconf1: Docker registry, 1 MariaDB master, 1 MariaDB worker
- Reconf2: Docker registry, 1 MariaDB master, N MariaDB worker



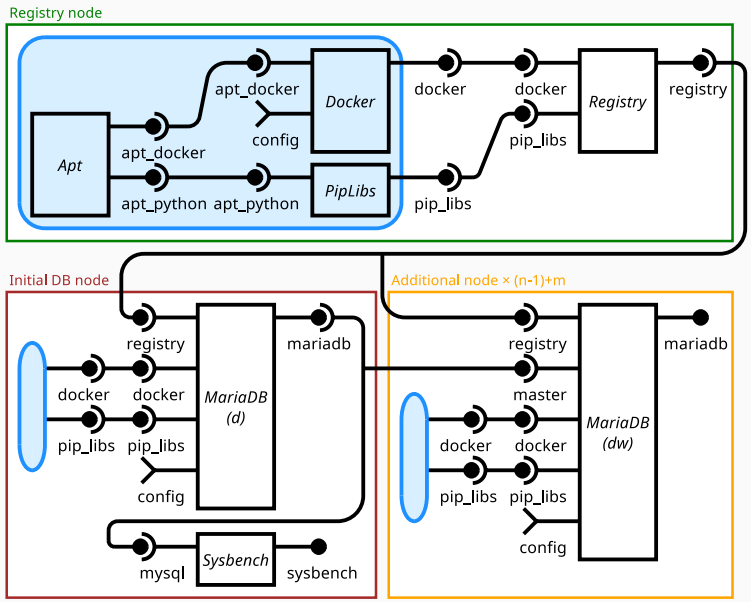
Experiments - decentralized DB (2/4)

- Deployment: Docker registry, MariaDB
- Reconf1: Docker registry, 1 MariaDB master, 1 MariaDB worker
- Reconf2: Docker registry, 1 MariaDB master, N MariaDB worker



Experiments - decentralized DB (2/4)

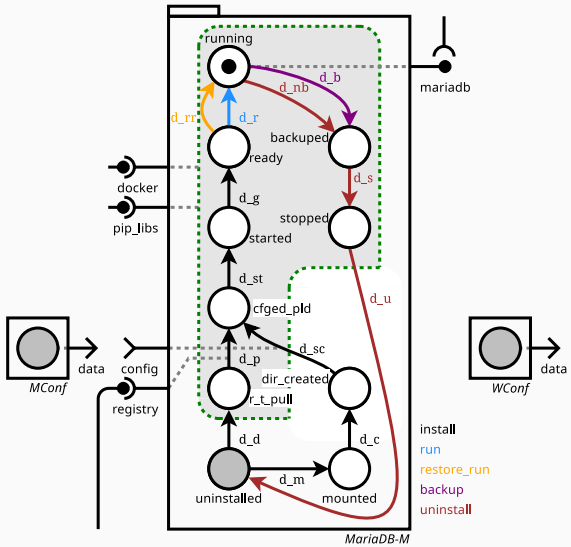
- Deployment: Docker registry, MariaDB
- Reconf1: Docker registry, 1 MariaDB master, 1 MariaDB worker
- Reconf2: Docker registry, 1 MariaDB master, N MariaDB worker



Experiments - decentralized DB (3/4)

Example (Deployment)

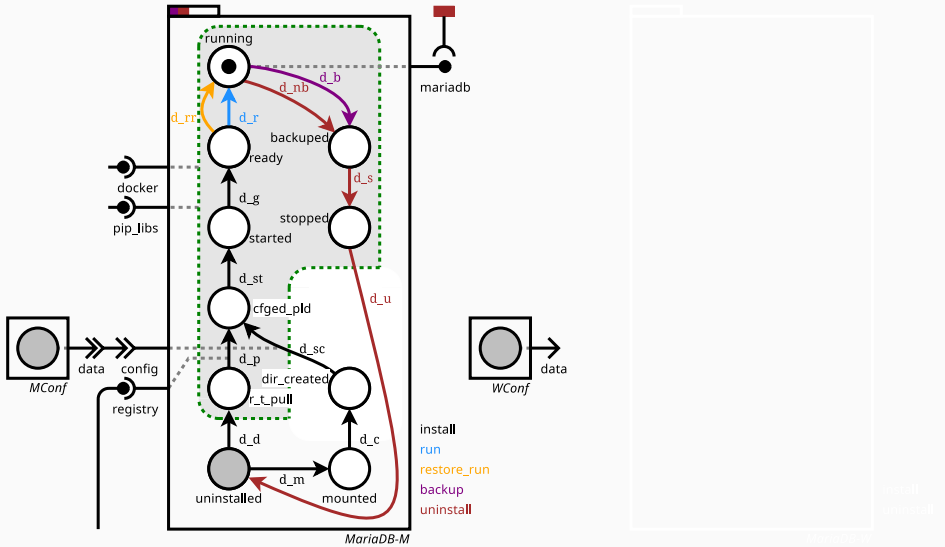
```
pushB (sysbenchm, suspend)
pushB (mariadb1, backup)
pushB (mariadb1, uninstall)
con (mconf.data, mariadb1.config)
for i in 2..n: [n+1..n+m]
  add (mariadb{i} : MariaDBw)
  con (wconf.data, mariadb{i}.config)
  con (mariadb1.mariadb, mariadb{i}.master)
  con (docker{i}.docker, mariadb{i}.docker)
  con (piplibs{i}.pip_libs,mariadb{i}.pip_libs)
  con (r_registry.registry,mariadb{i}.registry)
for i in 1..n:
  pushB (mariadb{i}, install)
pushB (mariadb1, restore_run)
pushB (sysbenchm, install)
```



Experiments - decentralized DB (3/4)

Example (Deployment)

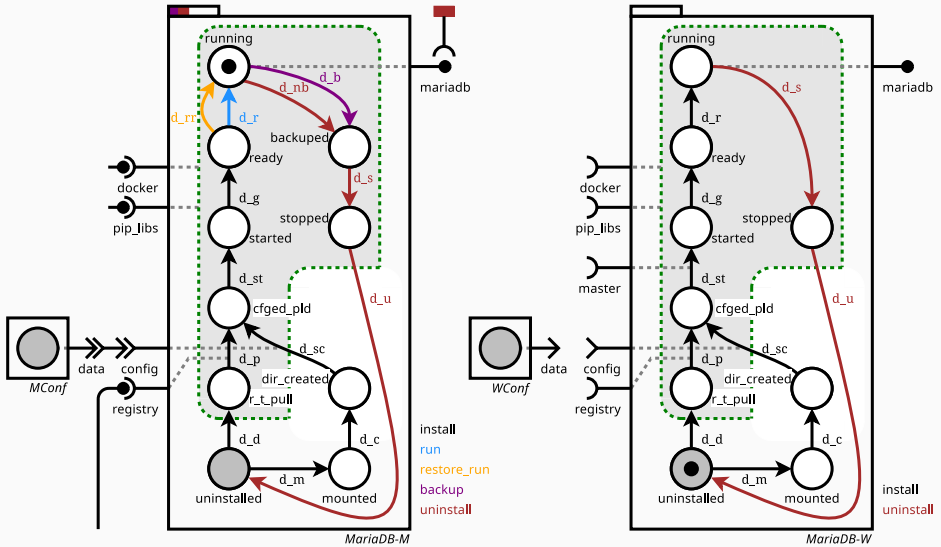
```
pushB (sysbenchm, suspend)
pushB (mariadb1, backup)
pushB (mariadb1, uninstall)
con (mconf.data, mariadb1.config)
for i in 2..n: [n+1..n+m]
  add (mariadb{i} : MariaDBw)
  con (wconf.data, mariadb{i}.config)
  con (mariadb1.mariadb, mariadb{i}.master)
  con (docker{i}.docker, mariadb{i}.docker)
  con (piplibs{i}.pip_libs,mariadb{i}.pip_libs)
  con (r_registry.registry,mariadb{i}.registry)
for i in 1..n:
  pushB (mariadb{i}, install)
pushB (mariadb1, restore_run)
pushB (sysbenchm, install)
```



Experiments - decentralized DB (3/4)

Example (Deployment)

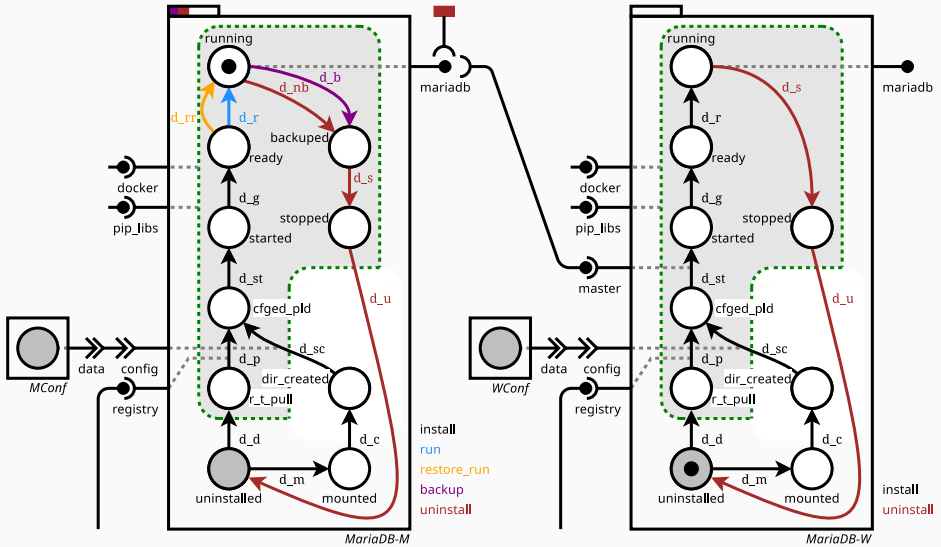
```
pushB (sysbenchm, suspend)
pushB (mariadb1, backup)
pushB (mariadb1, uninstall)
con (mconf.data, mariadb1.config)
for i in 2..n: [n+1..n+m]
  add (mariadb{i} : MariaDBw)
  con (wconf.data, mariadb{i}.config)
  con (mariadb1.mariadb, mariadb{i}.master)
  con (docker{i}.docker, mariadb{i}.docker)
  con (piplibs{i}.pip_libs,mariadb{i}.pip_libs)
  con (r_registry.registry,mariadb{i}.registry)
for i in 1..n:
  pushB (mariadb{i}, install)
pushB (mariadb1, restore_run)
pushB (sysbenchm, install)
```



Experiments - decentralized DB (3/4)

Example (Deployment)

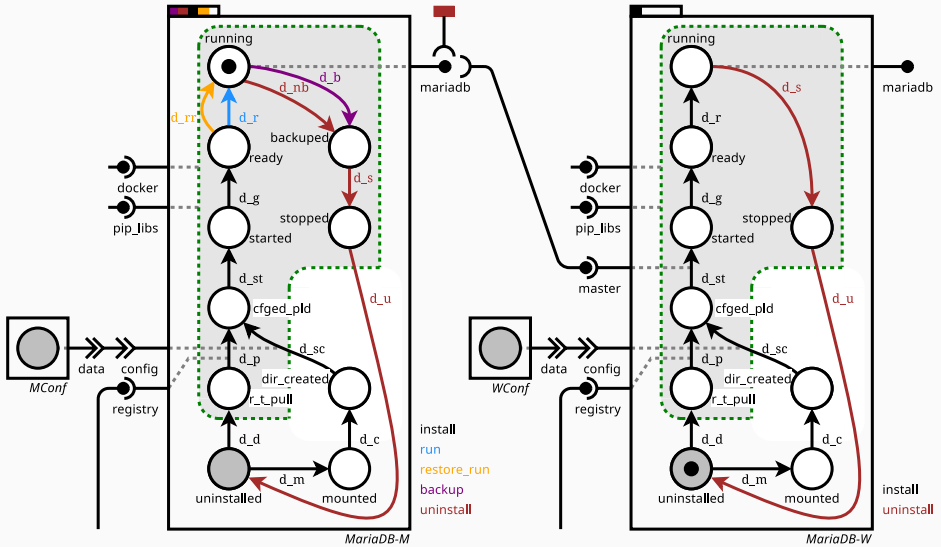
```
pushB (sysbenchm, suspend)
pushB (mariadb1, backup)
pushB (mariadb1, uninstall)
con (mconf.data, mariadb1.config)
for i in 2..n: [n+1..n+m]
  add (mariadb{i} : MariaDBw)
  con (wconf.data, mariadb{i}.config)
  con (mariadb1.mariadb, mariadb{i}.master)
  con (docker{i}.docker, mariadb{i}.docker)
  con (piplibs{i}.pip_libs,mariadb{i}.pip_libs)
  con (r_registry.registry,mariadb{i}.registry)
for i in 1..n:
  pushB (mariadb{i}, install)
pushB (mariadb1, restore_run)
pushB (sysbenchm, install)
```



Experiments - decentralized DB (3/4)

Example (Deployment)

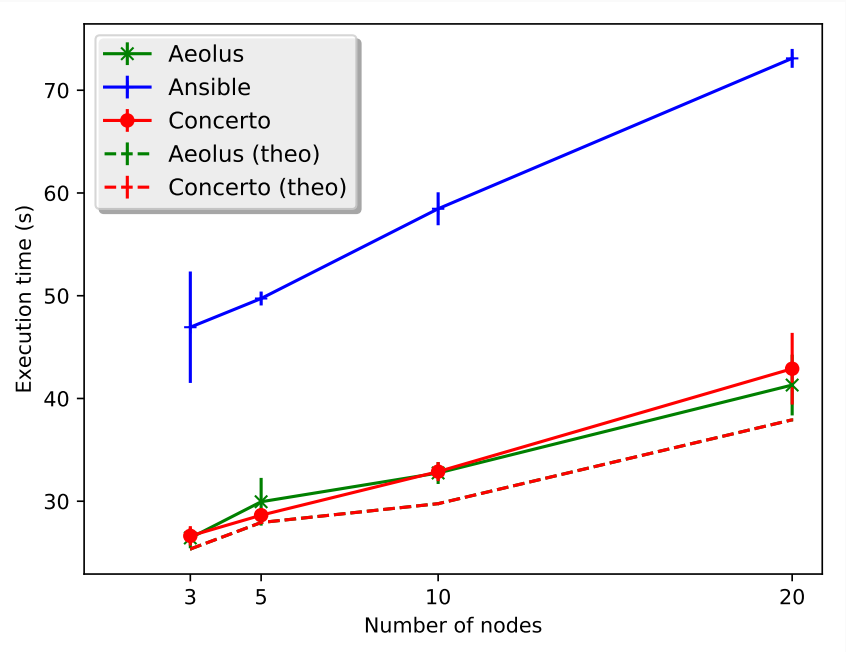
```
pushB (sysbenchm, suspend)
pushB (mariadb1, backup)
pushB (mariadb1, uninstall)
con (mconf.data, mariadb1.config)
for i in 2..n: [n+1..n+m]
  add (mariadb{i} : MariaDBw)
  con (wconf.data, mariadb{i}.config)
  con (mariadb1.mariadb, mariadb{i}.master)
  con (docker{i}.docker, mariadb{i}.docker)
  con (piplibs{i}.pip_libs,mariadb{i}.pip_libs)
  con (r_registry.registry,mariadb{i}.registry)
for i in 1..n:
  pushB (mariadb{i}, install)
pushB (mariadb1, restore_run)
pushB (sysbenchm, install)
```

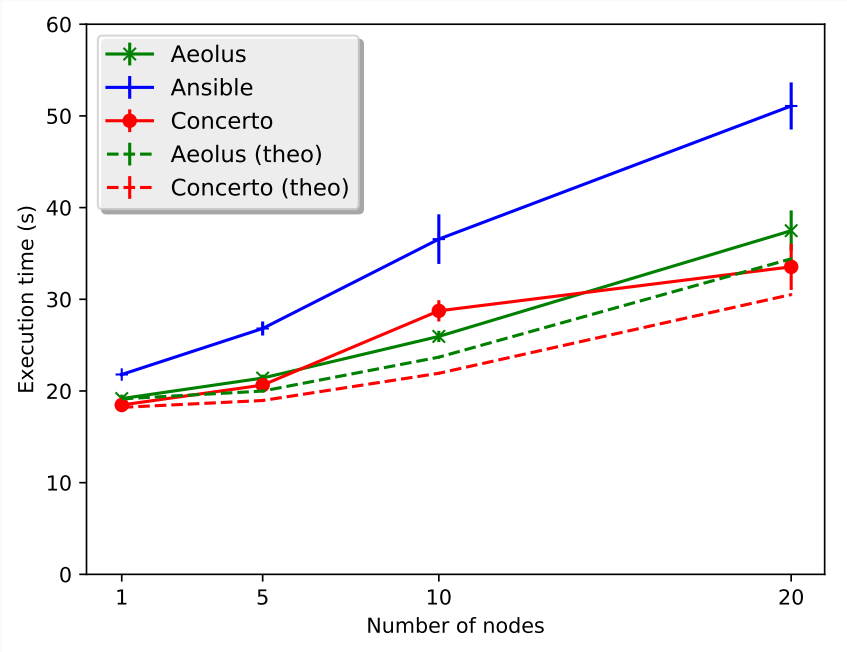


Experiments - decentralized DB (4/4)

Example (Reconfiguration 1)

- Concerto: 15 lines, no technical code, internal states abstraction
- Aeolus: 34 lines, no technical code
- Ansible: 110 lines, technical code to write





Conclusion and perspectives

Conclusion

Madeus

- Formal operational semantics
- Theoretical performance model
- Madeus prototype
- Reproducible lab on OpenStack

Concerto

- Formal operational semantics
- Theoretical performance model
- Concerto prototype
- Synthetic benchmarks
- Real use-case (in progress)

Perspectives

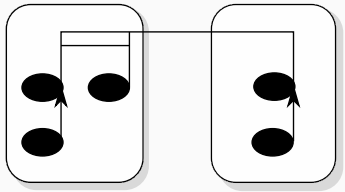
Madeus

- Spark use-case
- Automatic generation of Madeus assemblies from Puppet or Ansible scripts

Concerto

- Explore further the decentralized DB use-case
- Build another use-case on the reconfiguration of OpenStack
- Study concurrent reconfigurations

VeRDi



Verified Reconfiguration Driven by execution

Project

- 18 months postdoc (Simon Robillard)
- 18 months engineer (Charlène Servantie)

Work

- Static and dynamic verifications reconfiguration
- Formally study concurrent reconfigurations
- Formally study decentralized reconfigurations

Questions?