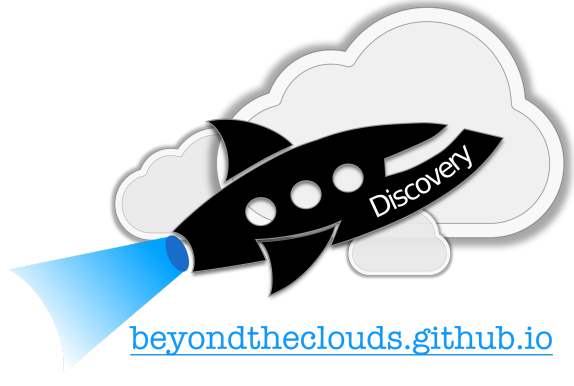
DEPLOYMENT AND RECONFIGURATION CHALLENGES

Hélène Coullon, Christian Perez, Dimitri Pertin

Ascola & Avalon

IPL Discovery



beyondtheclouds.github.io

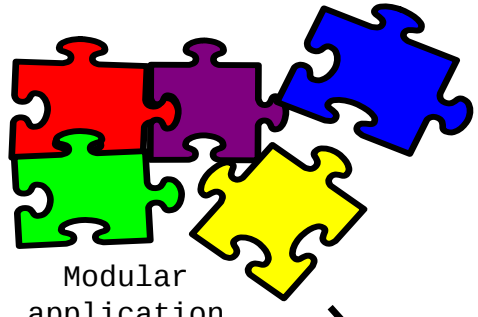
OUTLINE

1. Motivation
2. Survey
3. Initial Deployment
4. Reconfiguration
5. Perspectives
6. IPL and impact

MOTIVATION

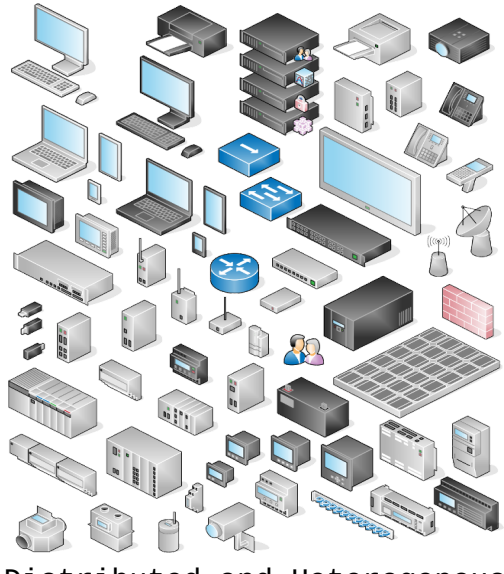
CONTEXT

DEPLOYMENT AUTOMATION



Modular
application

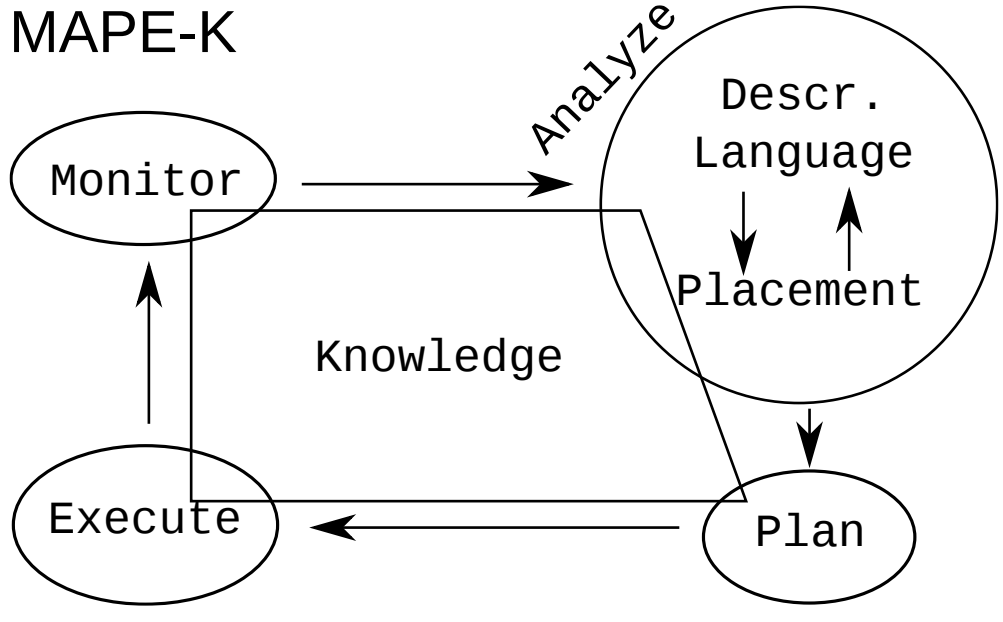
?



Distributed and Heterogenous
Infrastructure

DEPLOYMENT / RECONFIGURATION

MAPE-K

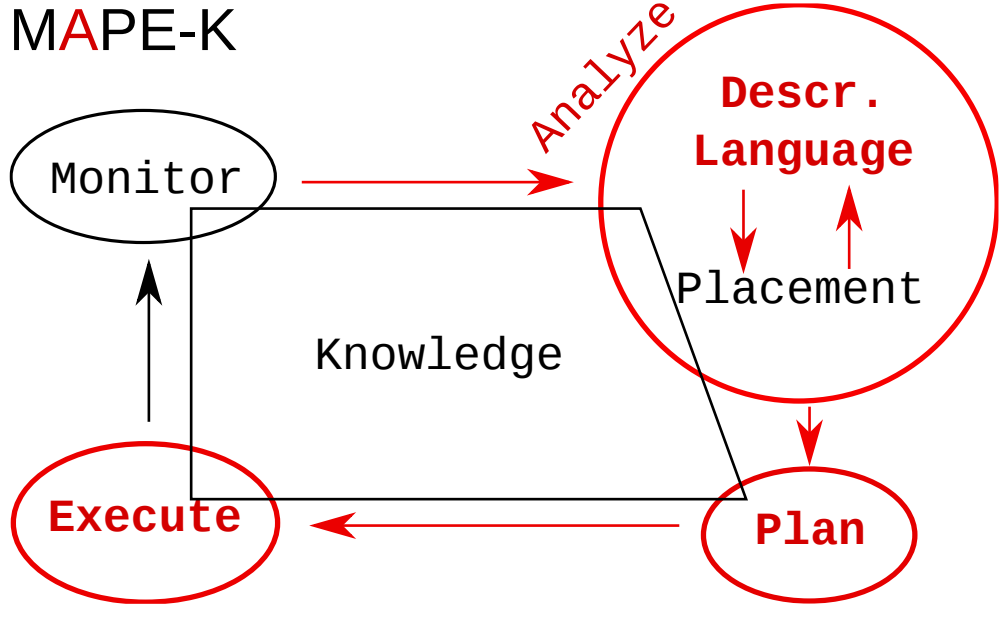


ANALYZE

- Descr. Language
  - *What* to deploy?
  - *How* to deploy?
- Placement
  - *Where* to deploy?
  - Infrastructure/resource description

DEPLOYMENT / RECONFIGURATION

FOCUS



BIG PICTURE

How to deploy/re-deploy systems and applications on infrastructures?

Expected properties:

- low-level flexible generic model
- appropriate level of expressivity
- dynamicity (reconfiguration)
- correctness and attainability
- performances and scalability

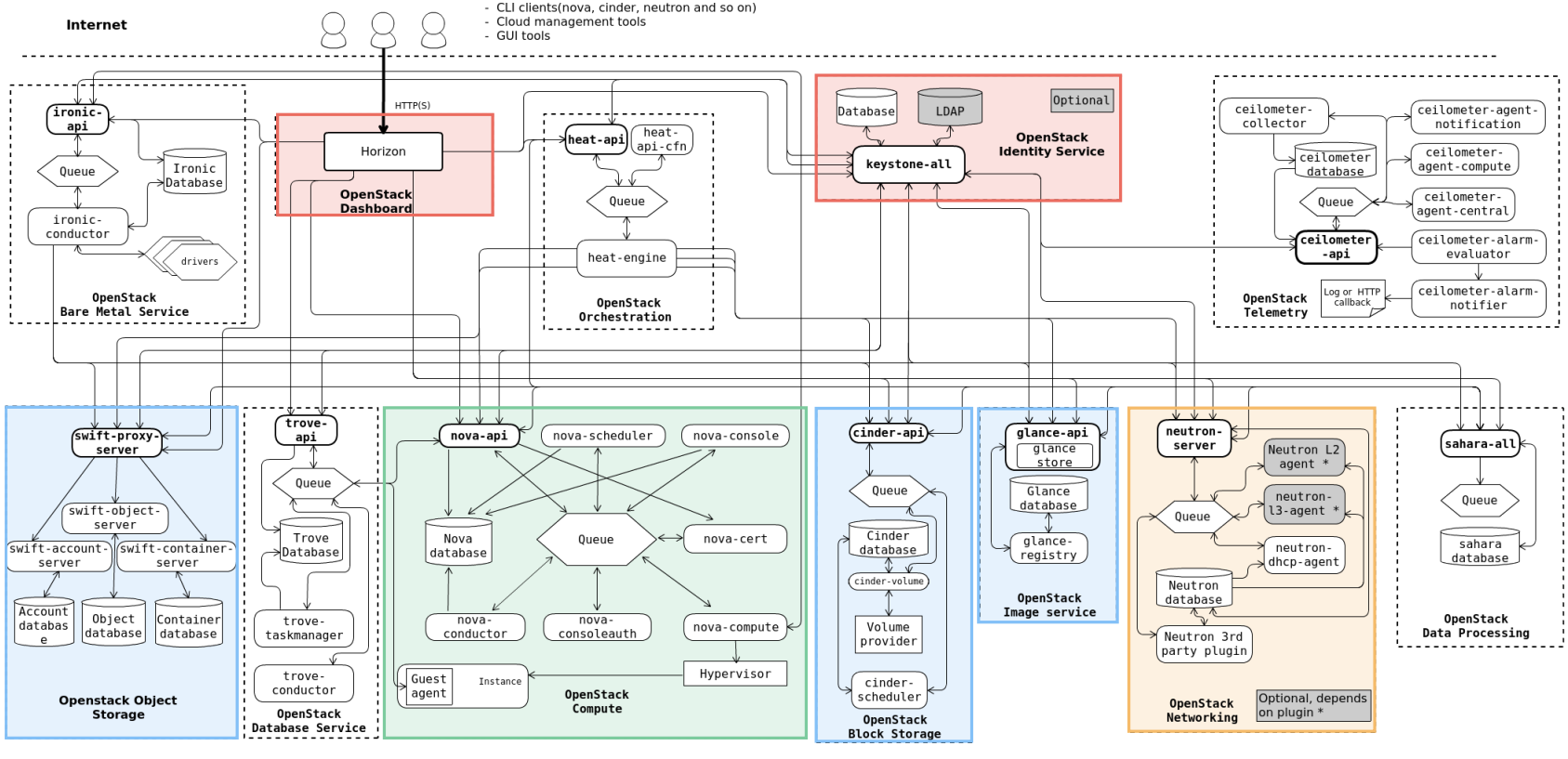
TWO PHASES

1. Research on the initial deployment problem
    - Dimitri Pertin, postdoc researcher
2. Research on the reconfiguration problem
    - Maverick Chardet, PhD student (October 2017)

USE CASES

Any application or system

- OpenStack and its decentralized version



USE CASES

- Smart-* applications composed of hybrid services
  - BigData
  - HPC
  - Stream processing
  - Virtual reality
  - etc.

SURVEY

SURVEY

Hélène Coullon, Christian Perez, Dimitri Pertin.
Production Deployment Tools for IaaSes: an Overall Model and Survey - FiCloud 2017

PRODUCTION TOOLS

1. Kolla: *deploy production-ready OpenStack instances by leveraging Ansible and Docker*
2. Juju: *Canonical project to write your application life cycle and deploy it on major cloud providers*
3. Kubernetes: *a project designed by Google to deploy and maintain containerized applications*
4. TripleO (OpenStack On OpenStack): *an OpenStack project aiming at deploying OpenStack instances using OpenStack's own services*

PRODUCTION TOOLS

GENERICITY

	Kolla	Juju	K8s	TripleO
app. generic	No	Yes	Yes	No
env. generic	No	Yes	No	No

PRODUCTION TOOLS

PLACEMENT ET MONITOR

	Kolla	Juju	K8s	TripleO
Plac.	Ext.	Ext.	Int.	Int.
	Manual	Auto	Auto	Auto
Mntr.	No	Manual	1/2 Auto	1/2 Auto

PRODUCTION TOOLS

DESCRIPTION LANGUAGE

	Kolla	Juju	K8s	TripleO
relations	No	Yes	Yes	Yes
constraints	No	Yes	Yes	Yes
actions	No	Yes	Yes	No
reconfiguration	No	3.	1.	2.

PRODUCTION TOOLS

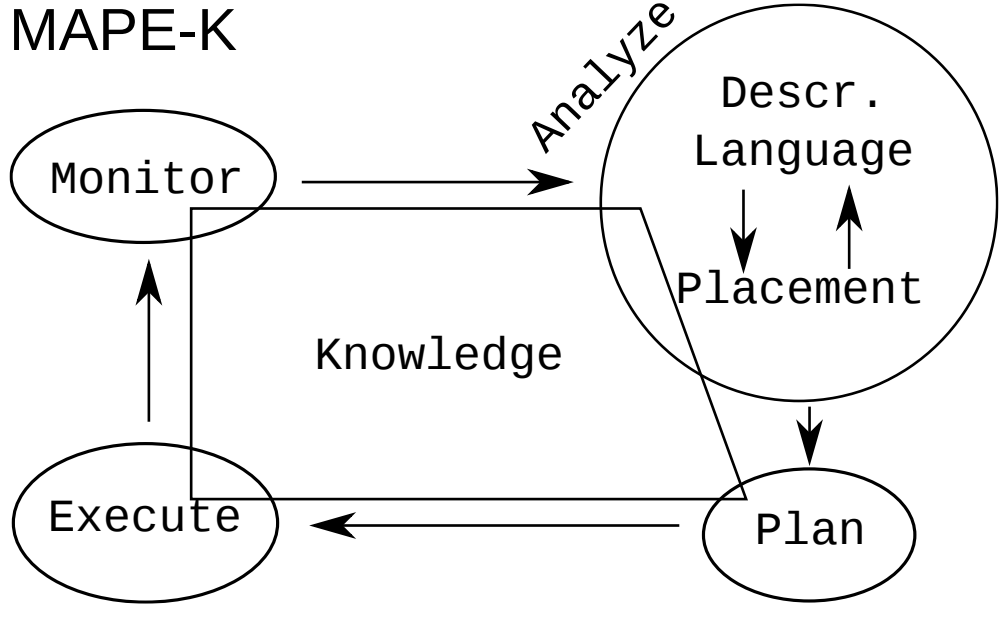
CONCLUSION

Expected properties:

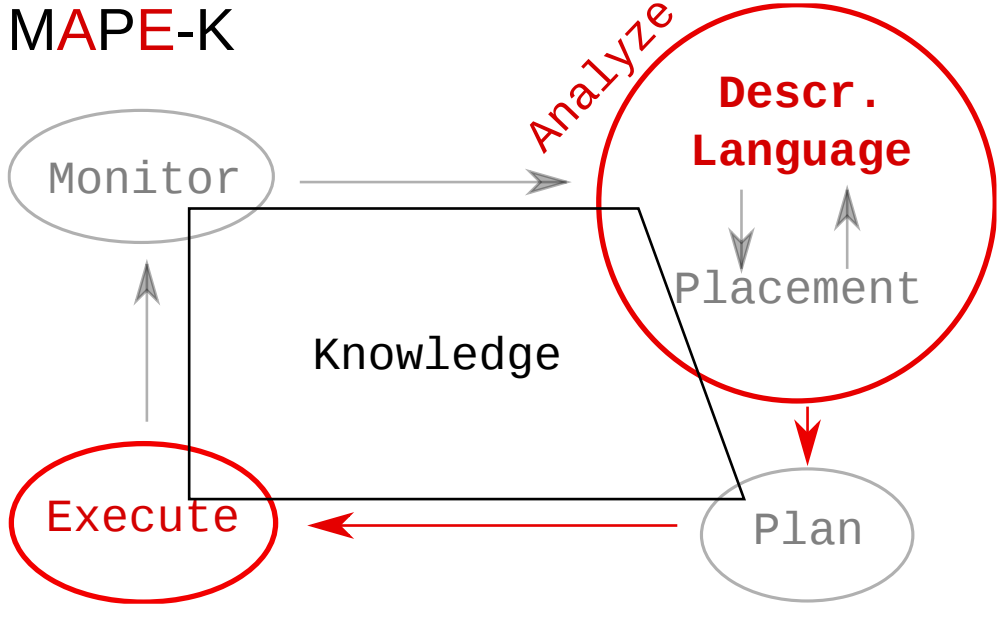
- low-level flexible generic model
- appropriate level of expressivity
- dynamicity (reconfiguration)
- correctness and attainability
- performances and scalability

INITIAL DEPLOYMENT

INITIAL DEPLOYMENT



INITIAL DEPLOYMENT



INITIAL DEPLOYMENT

GENERIC

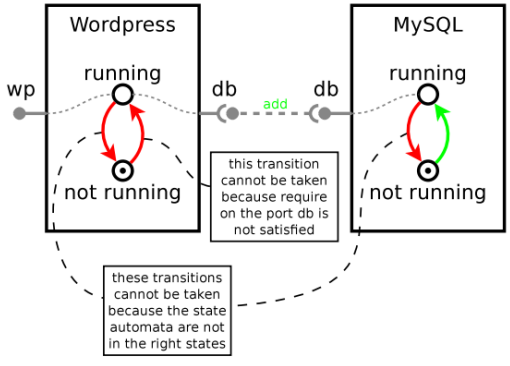
- Production tools: Juju, K8s, ansible
- Academic:
  - *Architecture Description Languages*: Tosca,
  - *Component models*: CCM, SCA, Fractal/Pro-active, L2C/HLCM etc.
  - *Deployment*: Deployware, PaaSage, MuScADeL, Aeolus etc.

INITIAL DEPLOYMENT

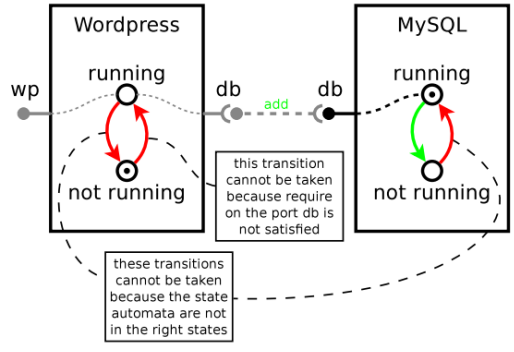
AEOLUS

Di Cosmo et al. 2014, *Information and Computation*

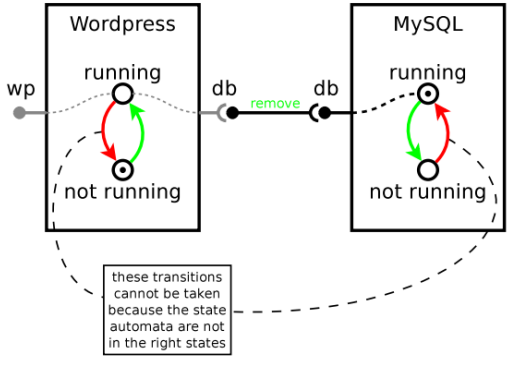
- components = services/modules
- functional dependencies = use-provide
- temporal constraints = states and transitions



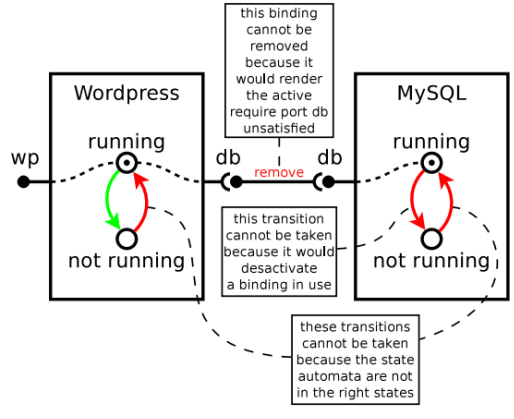
(a) Step 1: both components are not running, there is no binding between them.



(b) Step 2: Only MySQL is running, there is still no binding.



(c) Step 3: Only MySQL is running, a binding has been established.



(d) Step 4: Both components are running, the binding is present.

	provide port	require port	activation relation
inactive	● —	—(	- - - - - - - -
active	●—	—(	- - - - -

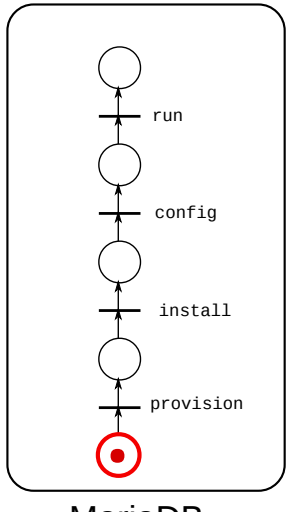
CONTRIBUTIONS

MULTI-TOKEN PETRI NETS, DATAFLOW

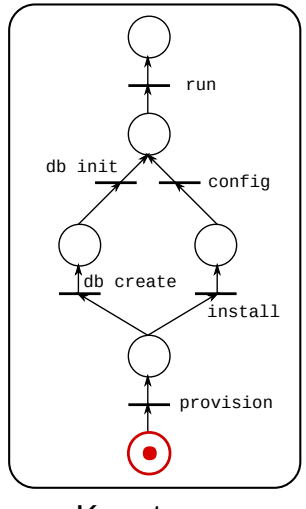
- Multi-token
  - correct internal dependencies
  - expose parallelism
- Petri-net
  - clear semantic between transitions and states
- Dataflow

CONTRIBUTIONS

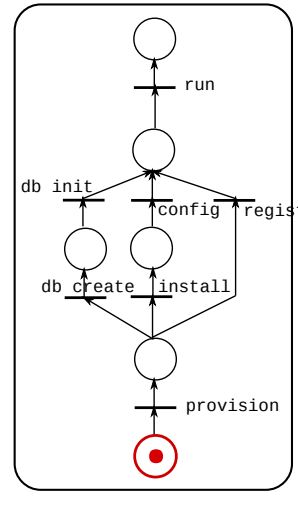
EXAMPLE



MariaDB



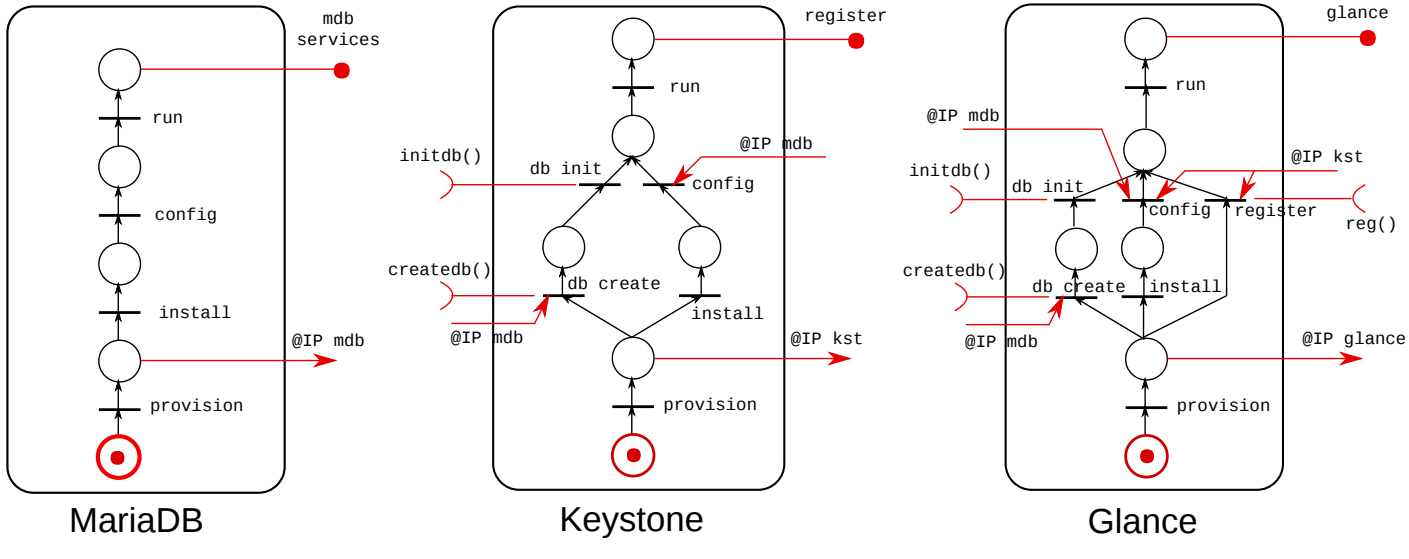
Keystone

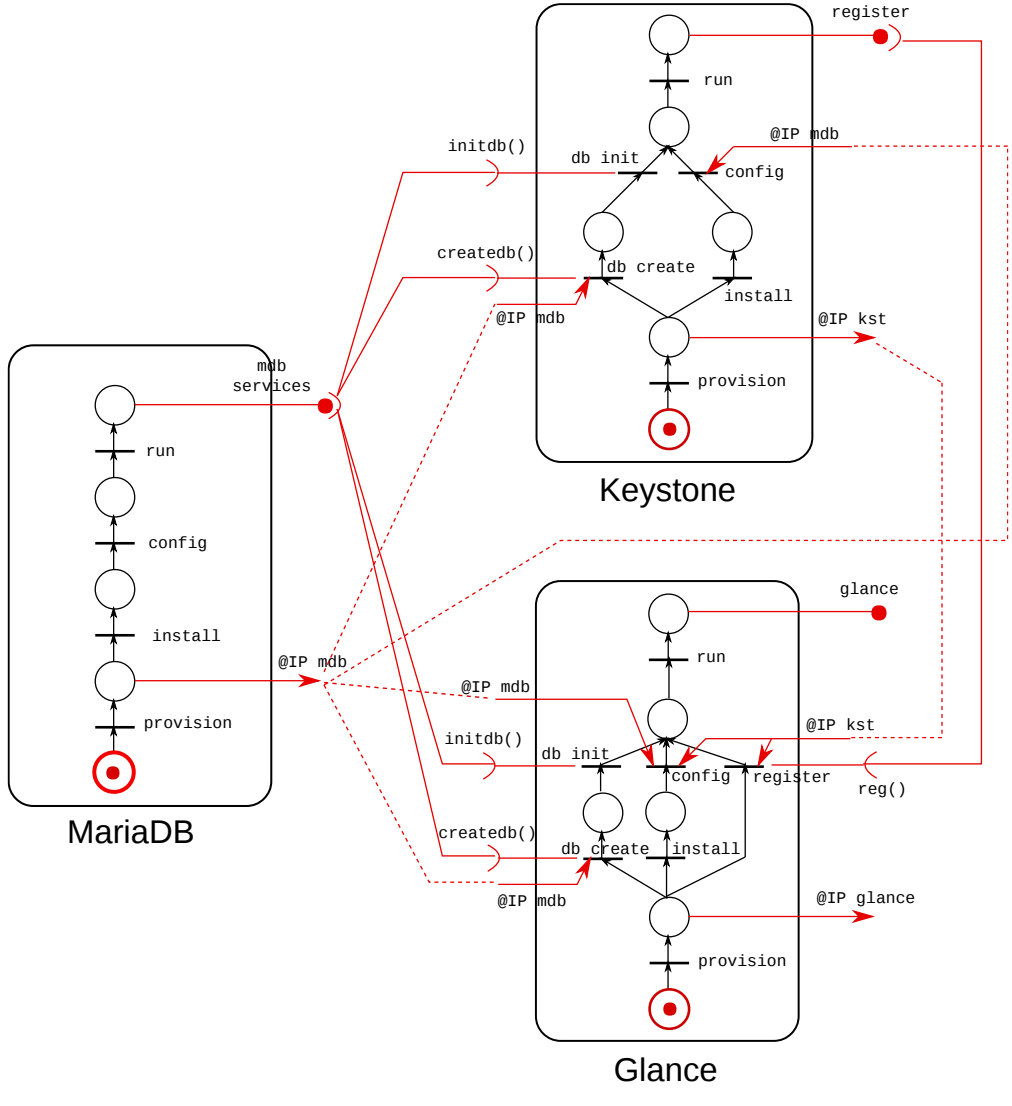


Glance

CONTRIBUTIONS

EXAMPLE



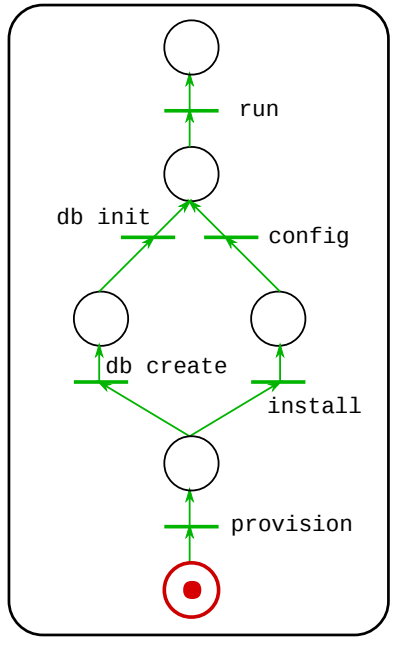

register
run
@IP mdb
initdb()
db init
config
createdb()
db create
install
@IP mdb
@IP kst
provision
Keystone
mdb
services
run
config
install
@IP mdb
provision
MariaDB
glance
run
@IP mdb
@IP kst
initdb()
db init
config
register
reg()
createdb()
db create
install
@IP mdb
@IP glance
provision
Glance

OTHER CONTRIBUTIONS

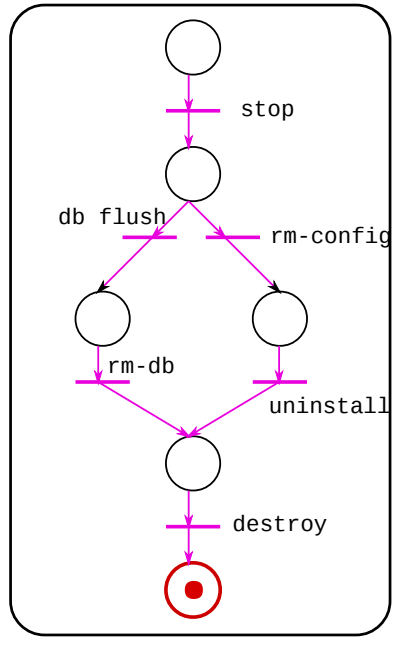
- Conditions (Workflow?)
- Colored petri-nets
- Hierarchy and cardinality

CONTRIBUTIONS

COLORS



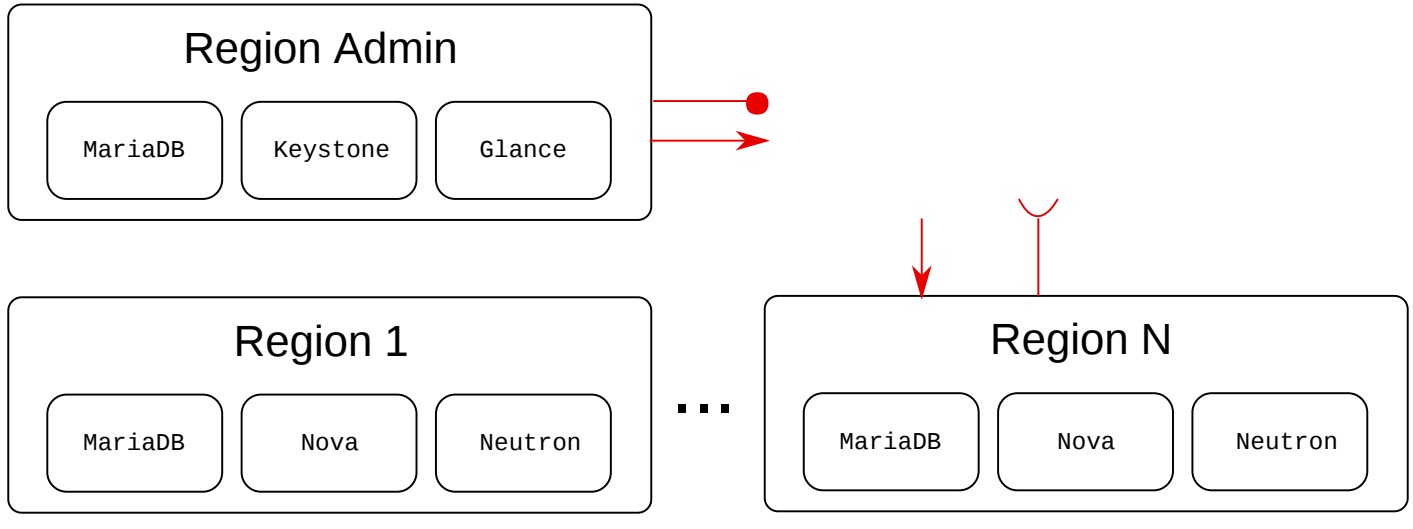
Keystone



Keystone

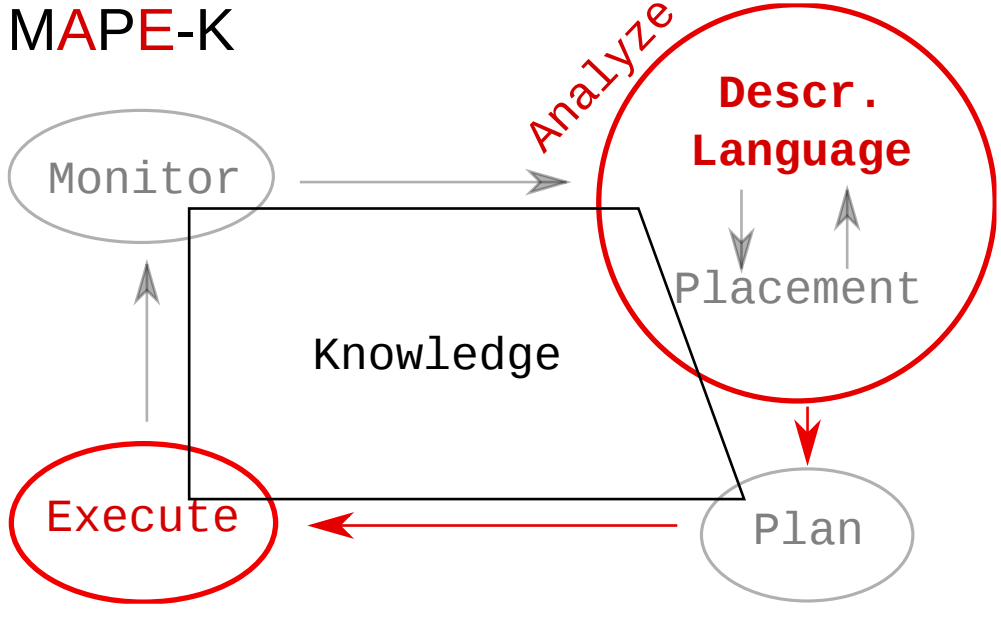
CONTRIBUTIONS

HIERARCHY

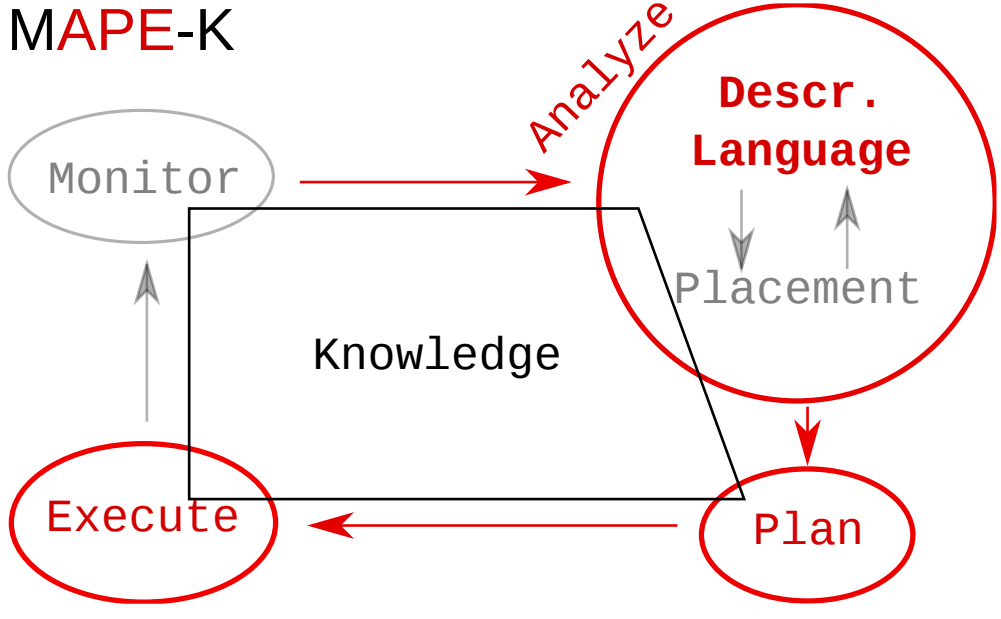


RECONFIGURATION

RECONFIGURATION



RECONFIGURATION



RECONFIGURATION

APPLICATION SIDE

- Fault Tolerance
- Automatic scaling (scale out, scale up)
- Updates (maintainability)
- Automatic software changes for various external reasons

RECONFIGURATION

INFRASTRUCTURE SIDE

- Massively distributed
- Failures
- Enter/leave
- Heterogeneity

RECONFIGURATION

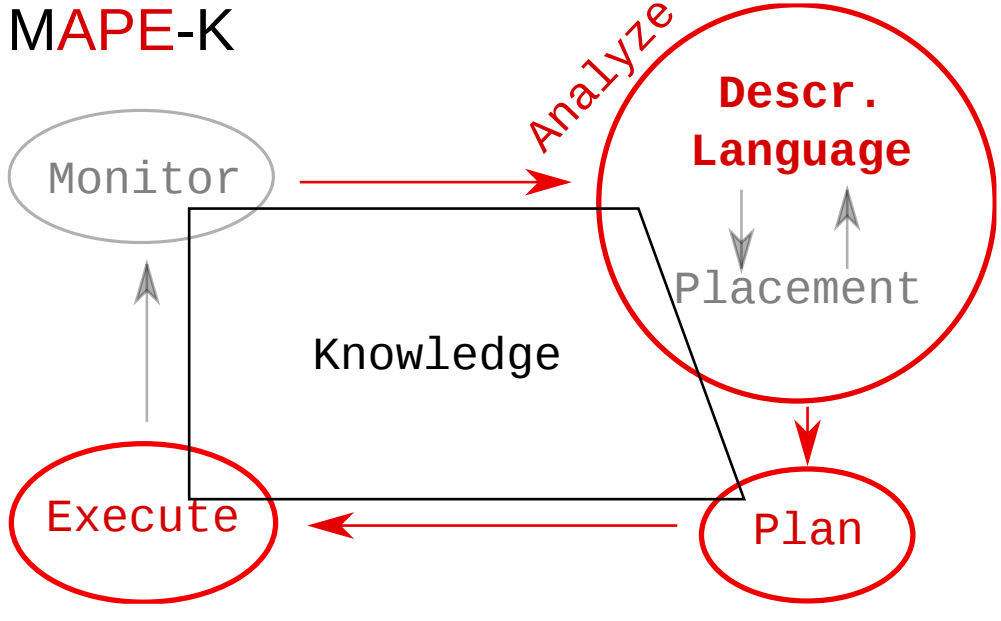
CHALLENGES

- Reconfiguration expressivity
- How to perform the reconfiguration? (Plan)
- Performances of the reconfiguration

PERSPECTIVES

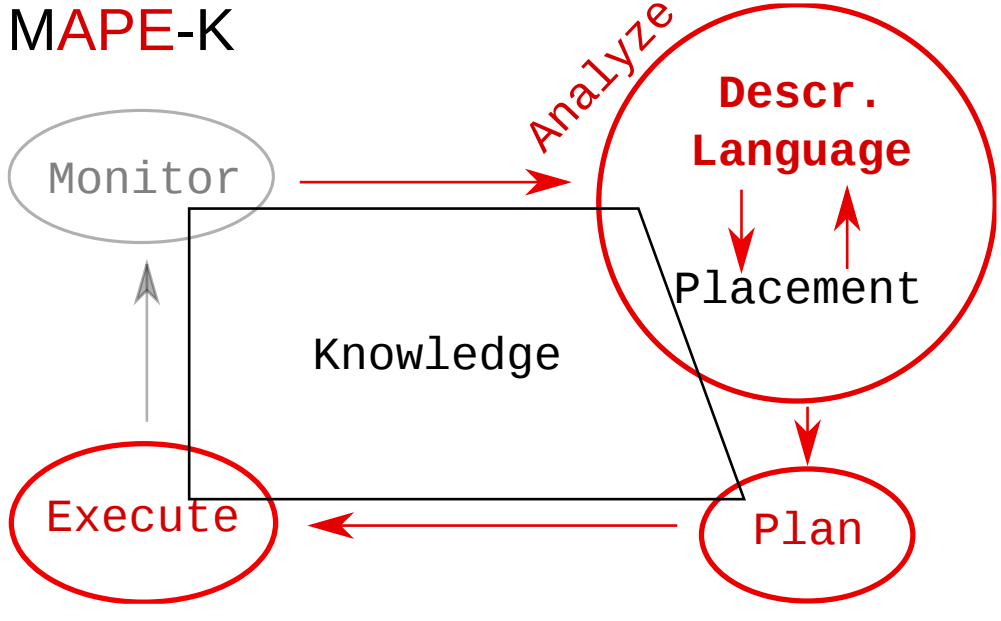
PERSPECTIVES

RESEARCH



PERSPECTIVES

RESEARCH



PERSPECTIVES

RESEARCH

- Formalism and proofs
- Higher abstraction level models:
  - Generic model as a back-end
  - Being more specific to applications and systems
  - DSLs
  - Compilation

PERSPECTIVES

COLLABORATIONS

- Roberto Di Cosmo: Formalism and proofs
- Marcos Dias: User-centric use-cases (Avalon)
- Mario Studholt: Security policies (Ascola)
- Erik Elmroth and Elastisys

IPL ROLE AND IMPACT

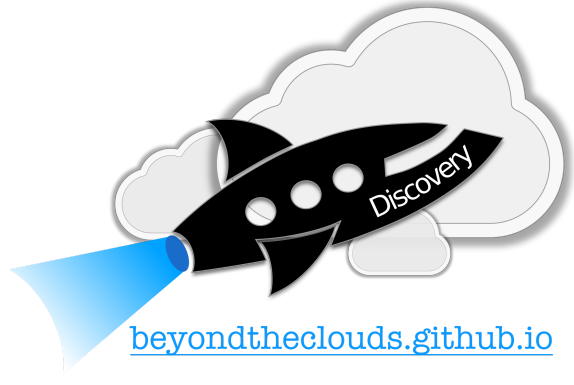
IPL ROLE

- EPC Avalon, Lyon: Component models
- EPC Ascola, Nantes: Distributed Systems, OpenStack
- Use-case: Discovery ! (decentralized OpenStack)

IMPACT

- To deploy OpenStack and automatically manage its behavior
- Within OpenStack to improve application deployment on VMs

THANK YOU !



beyondtheclouds.github.io