



# IaC Compliance Management Framework (IACMF)

# Introduction

- These slides help you get an overview of:
  - The *IaC Runtime Compliance Management Process*.
  - The *IaC Compliance Management Framework (IACMF)* that realizes the process.
- You are kindly asked to have an interview about these concepts as part of the [IAC<sup>2</sup> research project](#).

# What is IT Compliance?

- A state in which an IT system/process adheres to applicable rules.
- What kind of rules?
  - External regulations
    - Laws
    - Standards
    - Guidelines
    - ...
  - Internal policies.

# What is IaC?

- IaC stands for Infrastructure as Code.

“ *Infrastructure as Code is an approach to infrastructure automation based on practices from software development. It emphasizes consistent, repeatable routines for provisioning and changing systems and their configuration.*

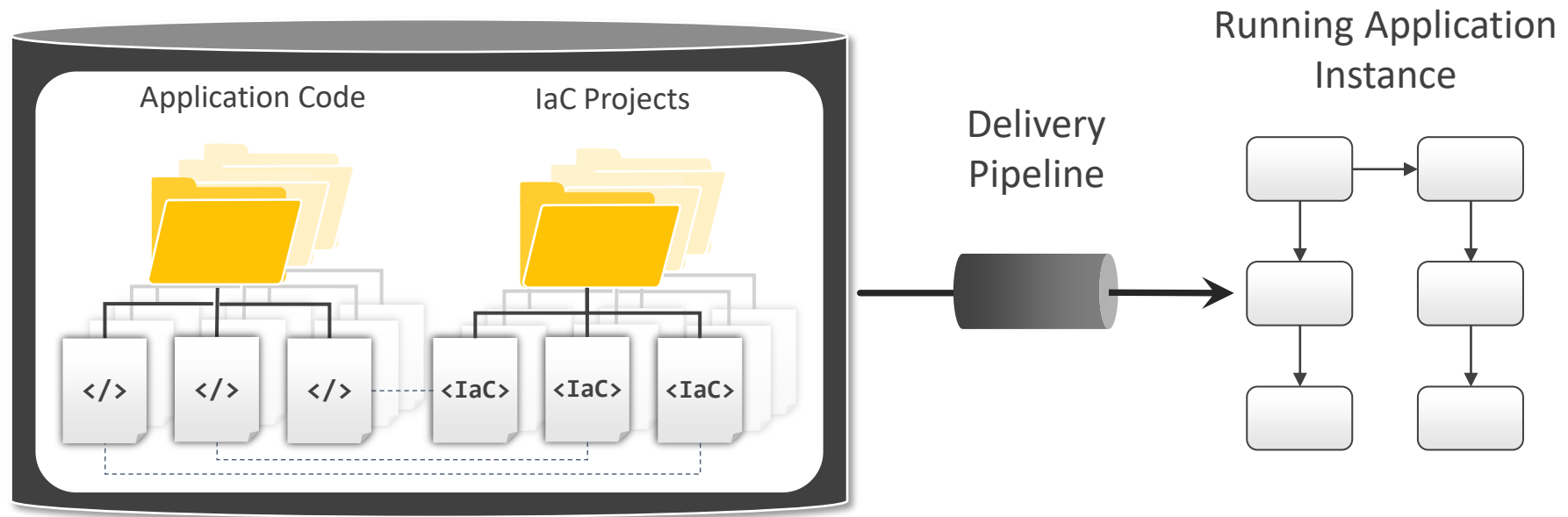
---

Kief Morris, *“Infrastructure as Code – Dynamic Systems for the Cloud Age”*, 2<sup>nd</sup> Edition

”

- System provisioning and management.
- Examples: Terraform, Kubernetes, Puppet, Chef, Ansible,...

# What is IaC?



# What is IaC Compliance?

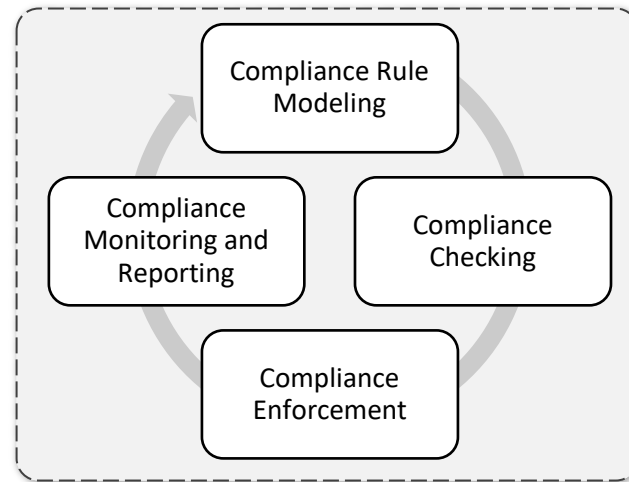
- Compliance of IT systems that use IaC tools.
  - Systems defined and provisioned using IaC tools.
  - Systems managed using IaC tools.
- Can be checked at *design time, deployment time, and/or runtime*.

# IT Compliance Management

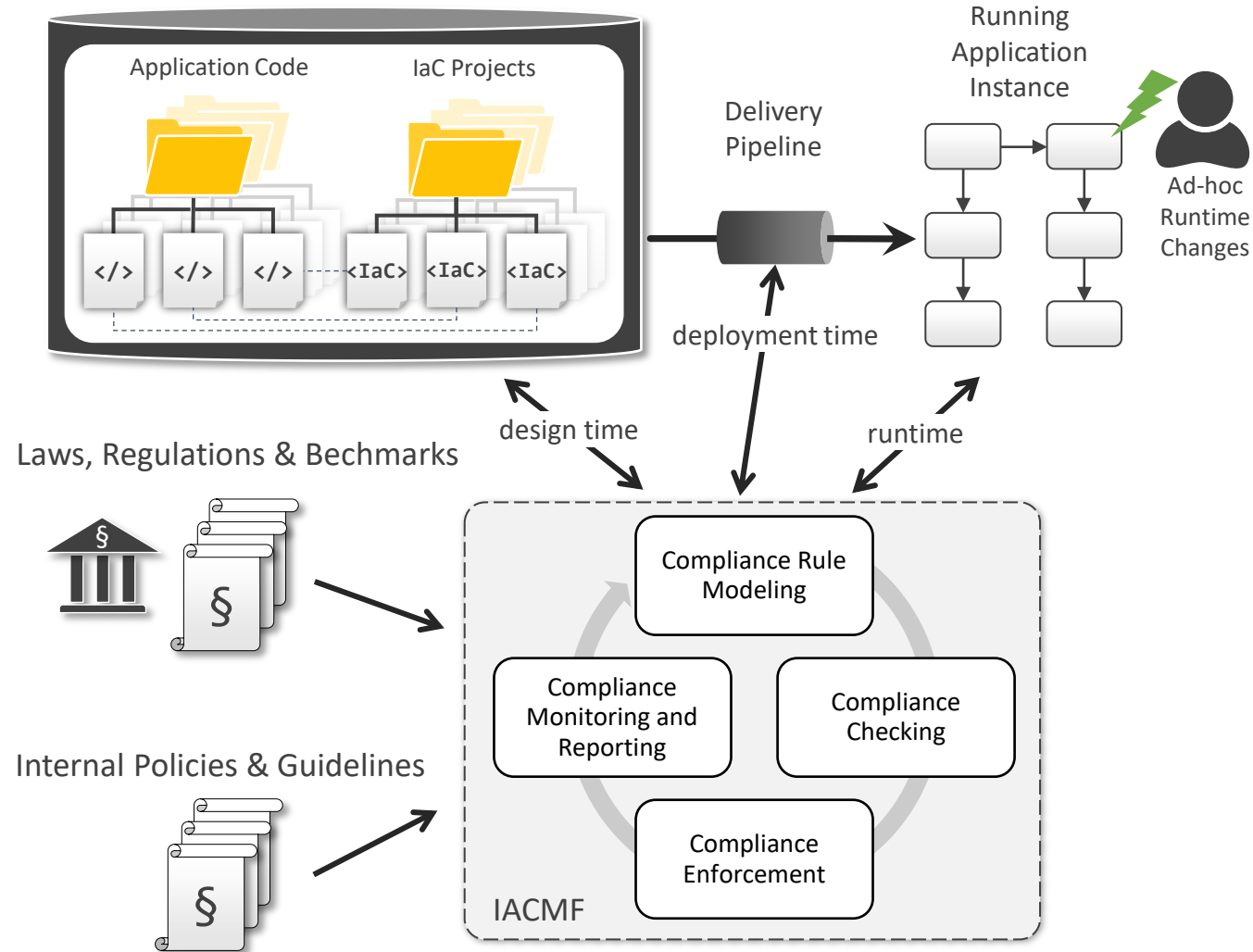
Laws, Regulations & Benchmarks



Internal Policies & Guidelines

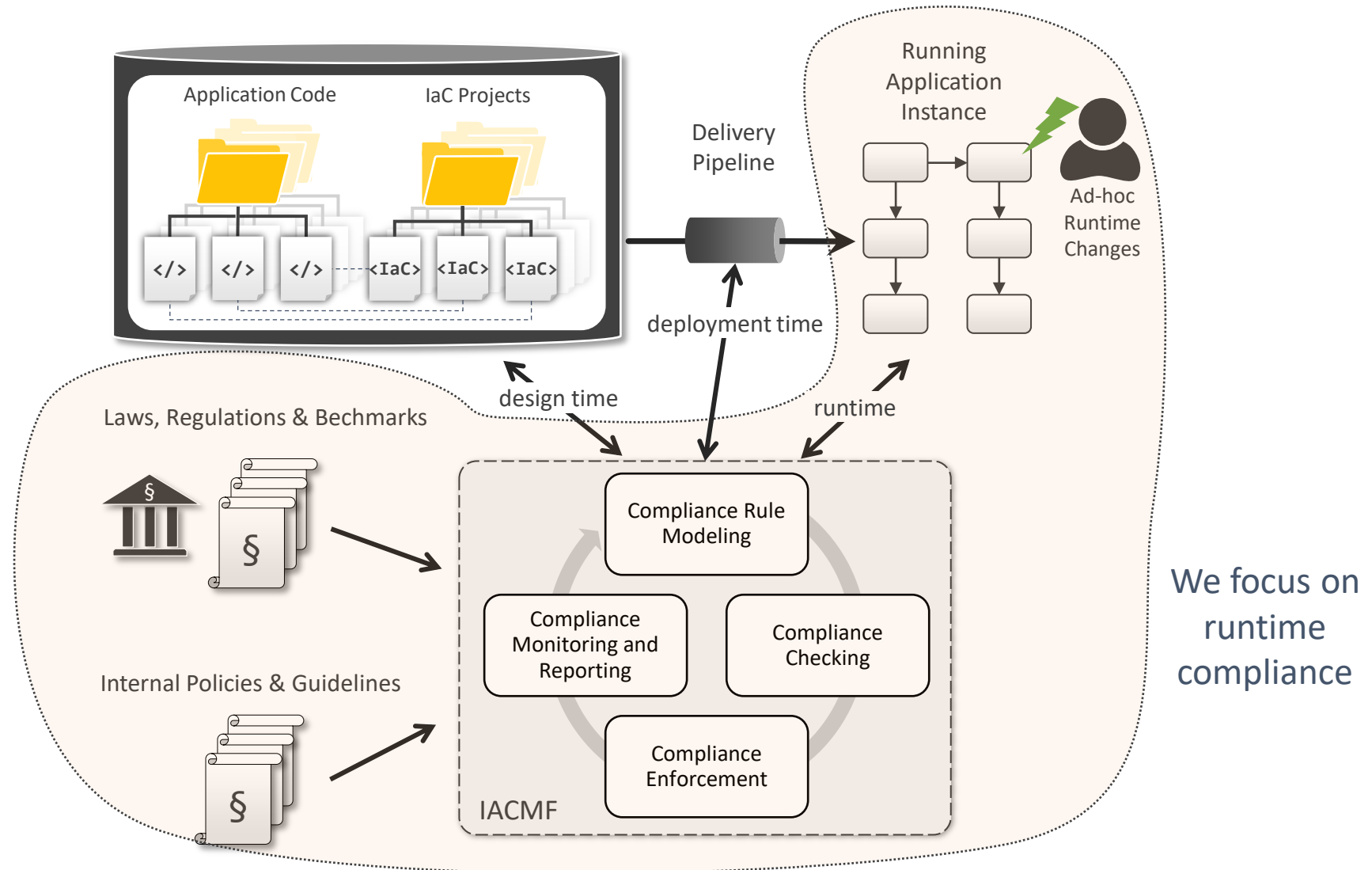


# IaC Compliance Management Framework (IACMF)





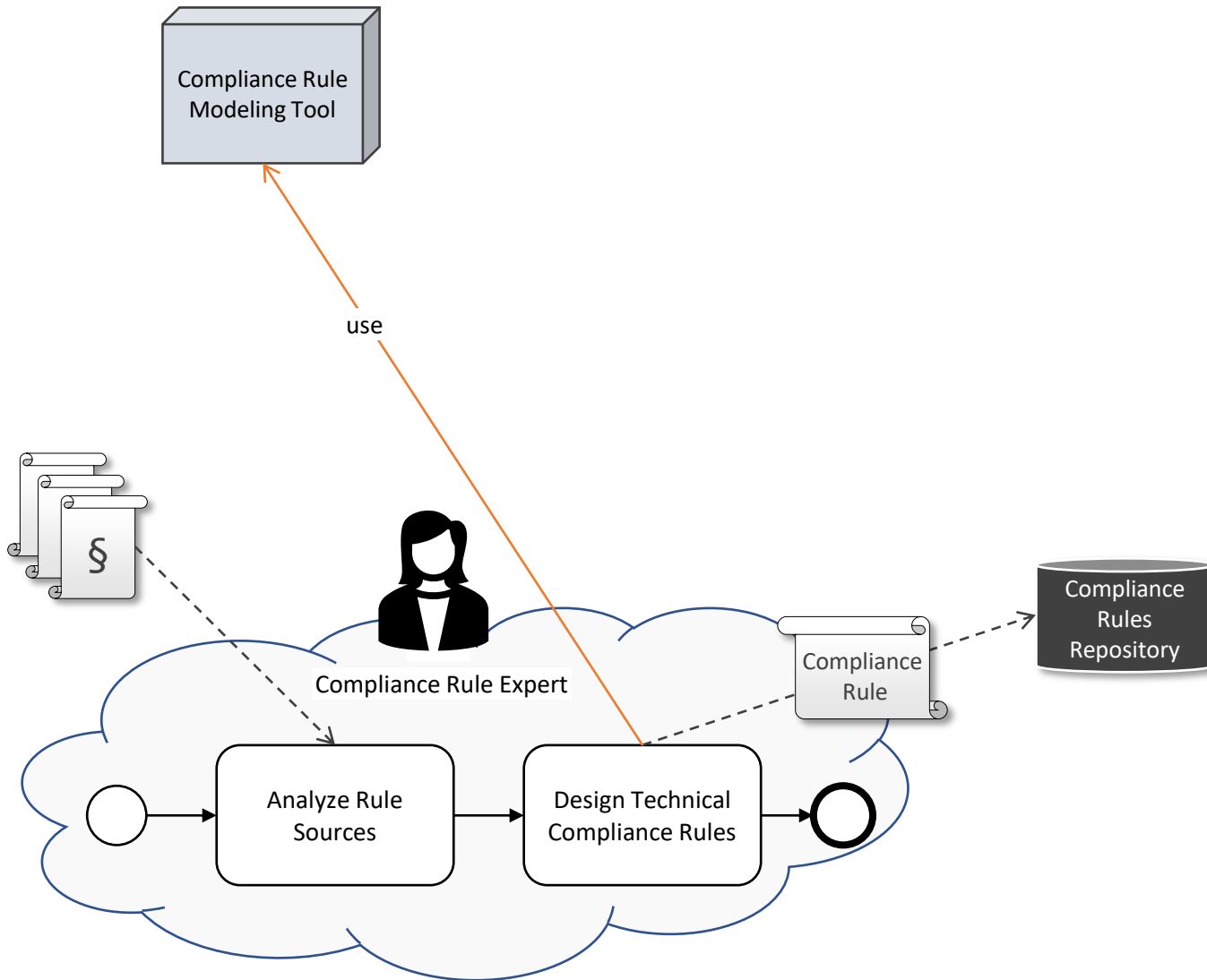
# IaC Compliance Management Framework (IACMF)



# IaC Runtime Compliance Management Process – designing technical compliance rules

- Transform compliance-related regulations and policies into machine-readable models.
- We call these models “*technical compliance rules*”.

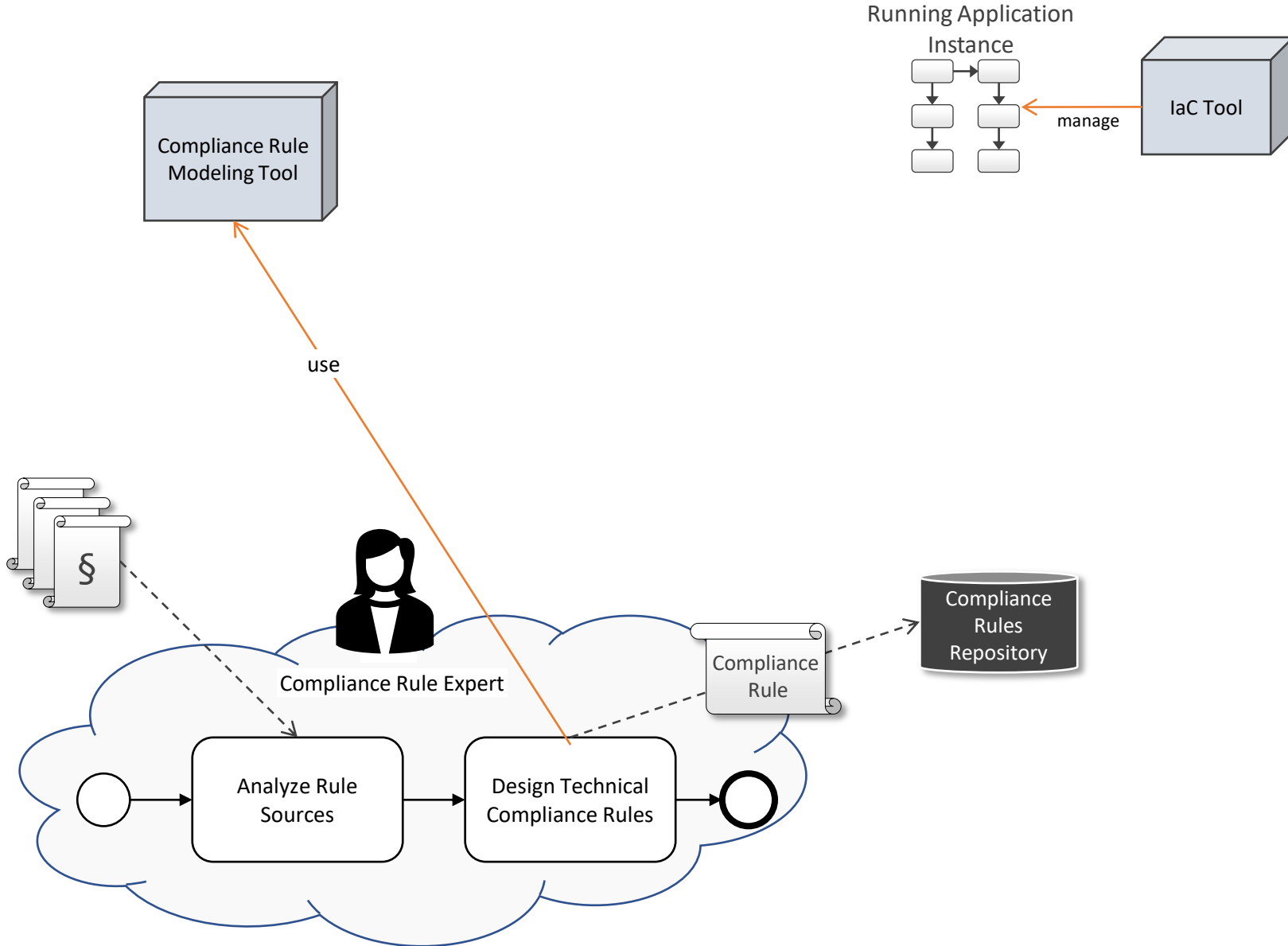
# IaC Runtime Compliance Management Process – designing technical compliance rules



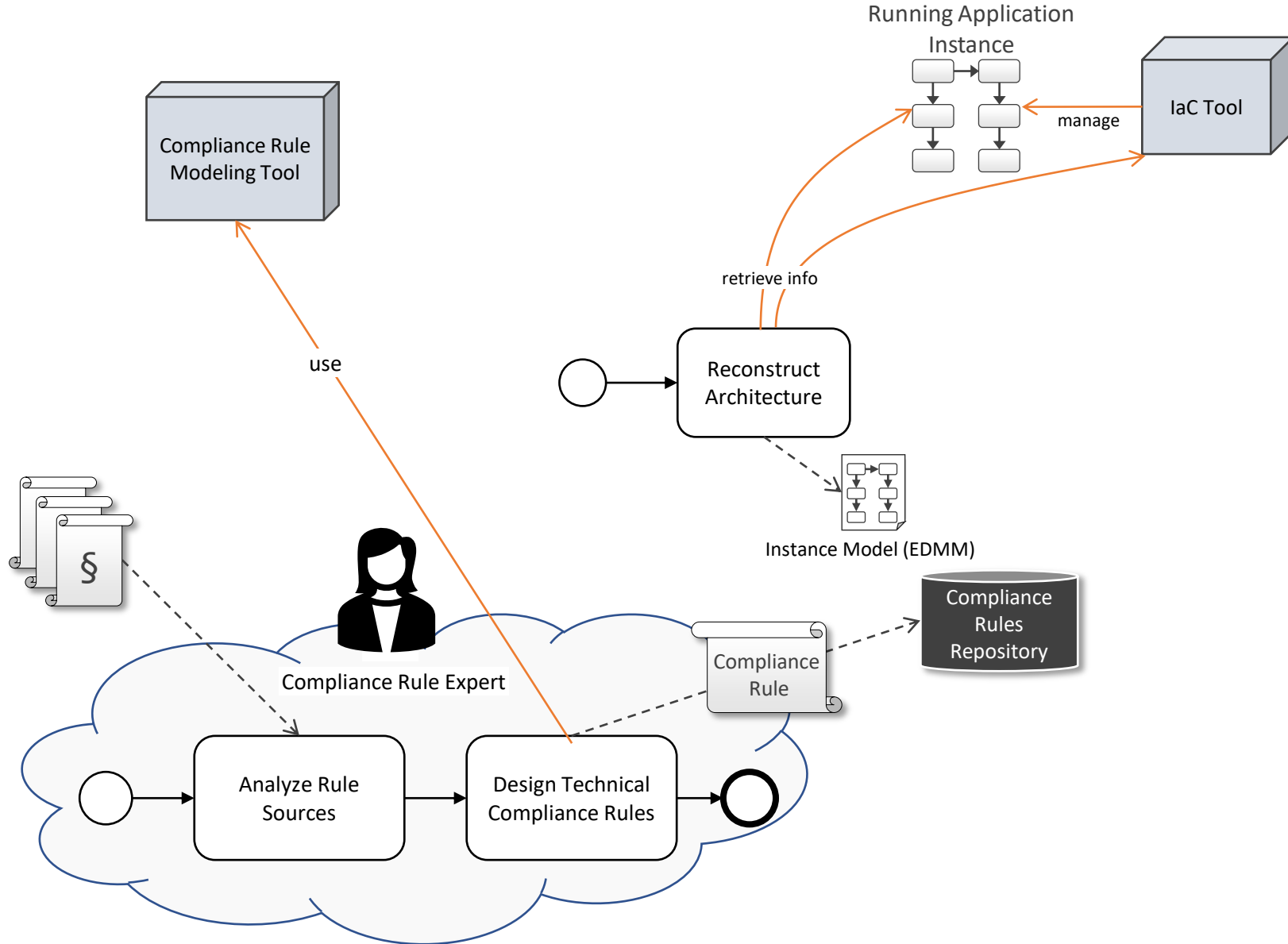
# IaC Runtime Compliance Management Process – reconstruct the architecture

- Create a model that describes the architecture of the application instance
- The model must have enough details to be able to evaluate the technical compliance rules.
- We call this model an *“instance model”*.

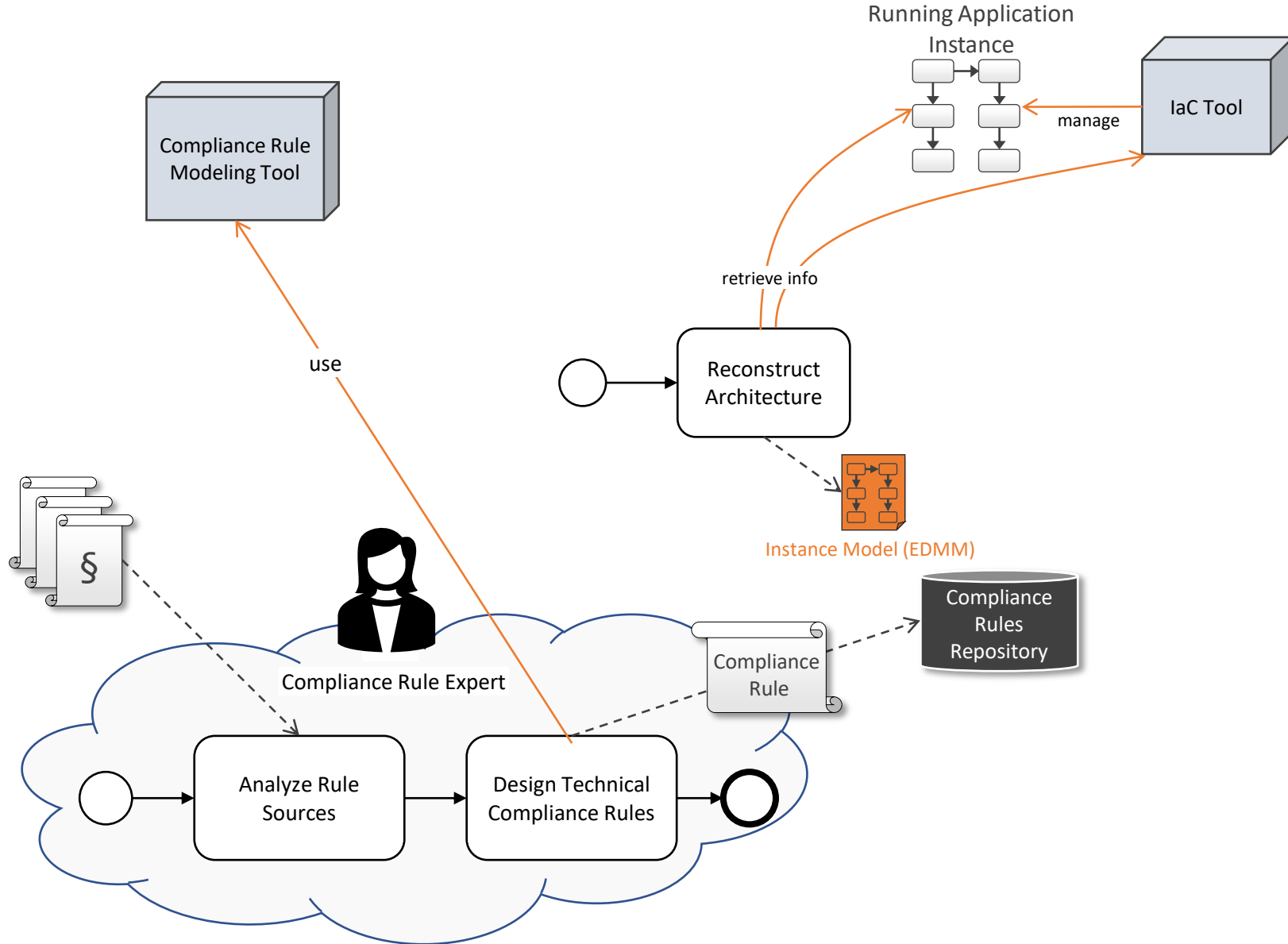
# IaC Runtime Compliance Management Process – reconstruct the architecture



# IaC Runtime Compliance Management Process—reconstruct the architecture



# IaC Runtime Compliance Management Process—reconstruct the architecture

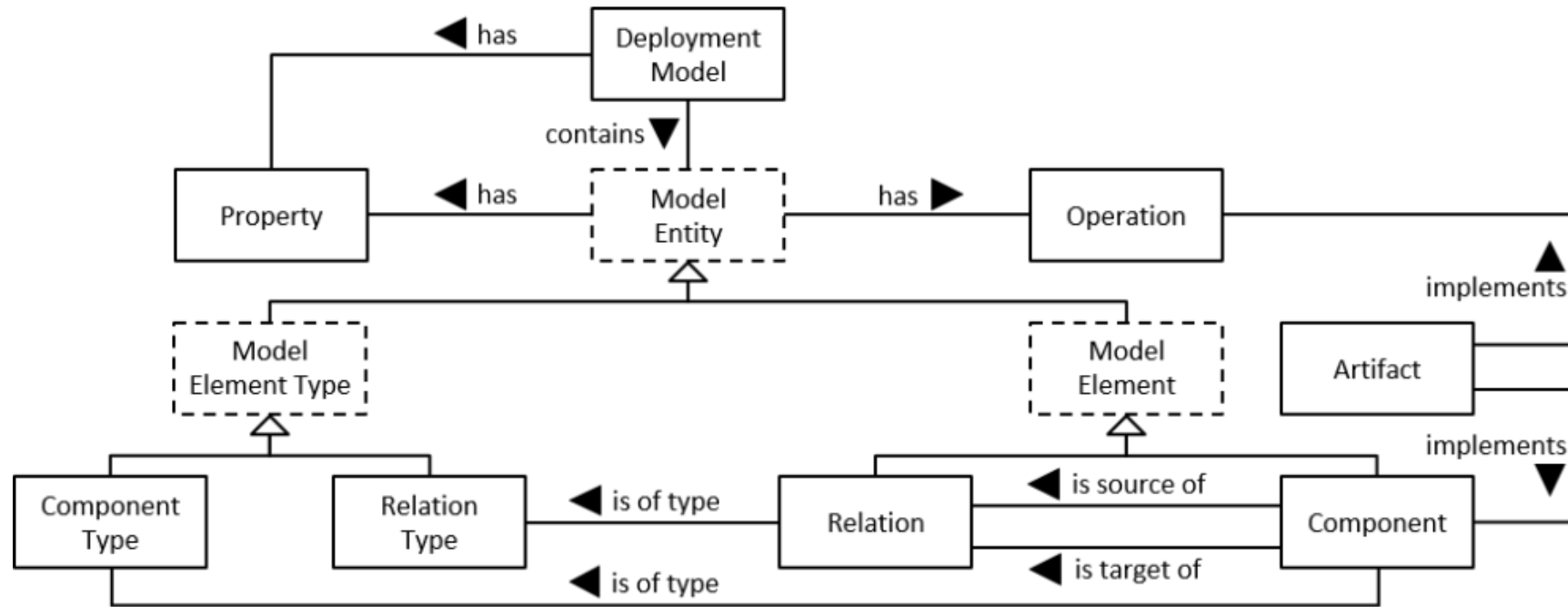


# Instance Model Format: Essential Deployment Metamodel (EDMM)

- Least common denominator among declarative deployment models.
- Result from a systematic review of commonly used IaC technologies.
- Allows us to represent running system instances in a technology-agnostic way.



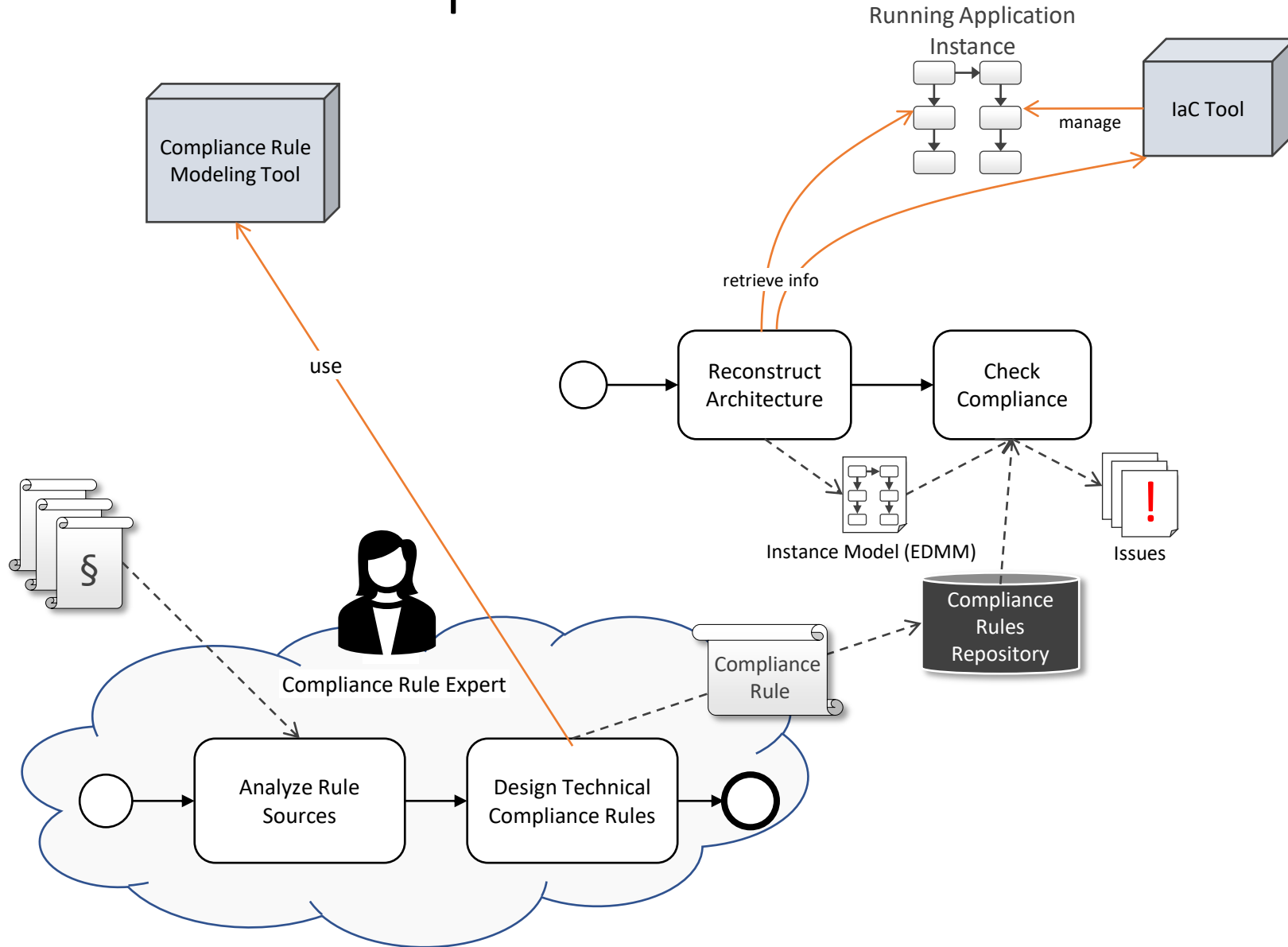
# Instance Model Format: Essential Deployment Metamodel (EDMM)



# laC Runtime Compliance Management Process – check compliance rules

- Automatically evaluate the fulfillment of the technical compliance rules by the reconstructed instance model.

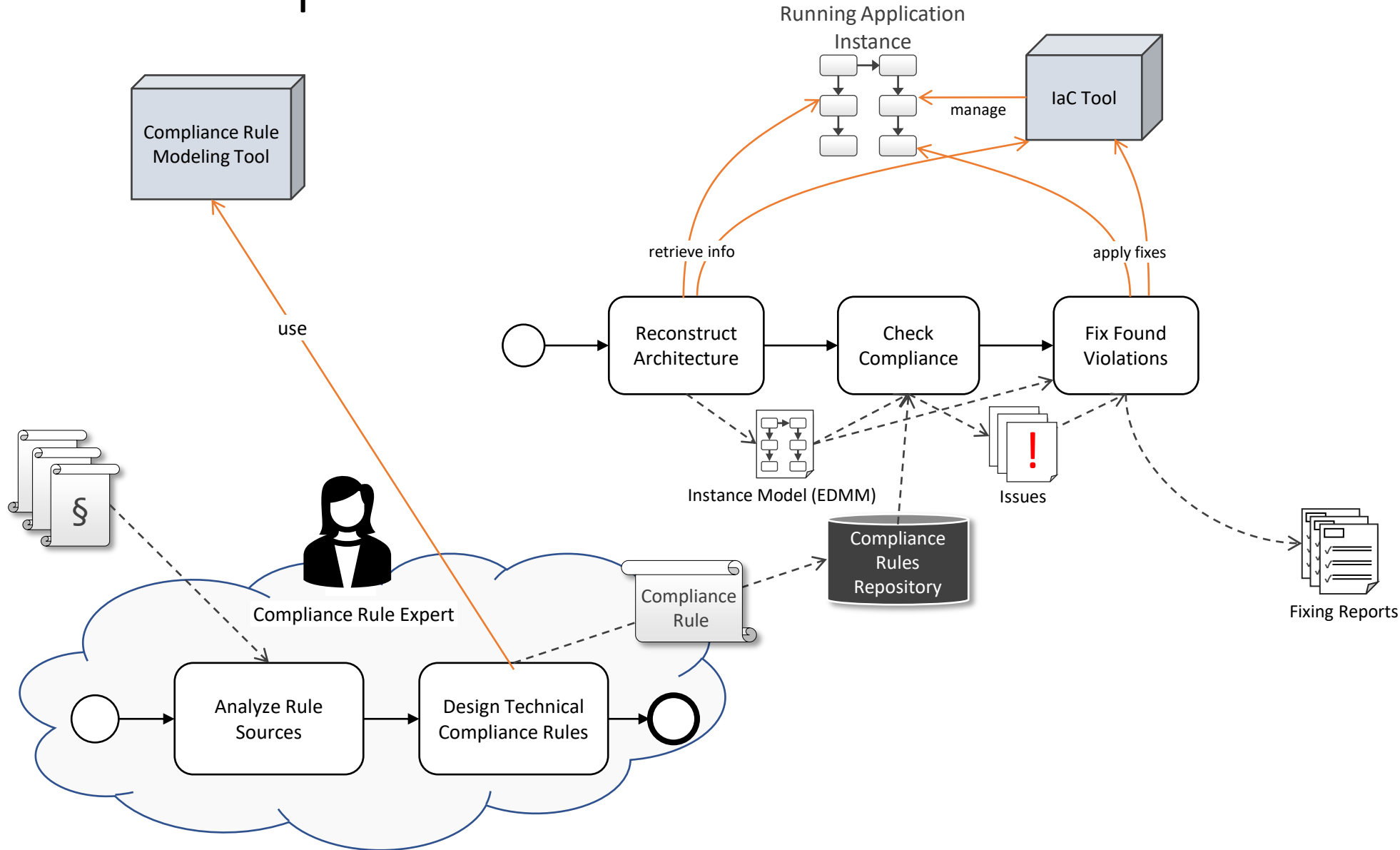
# IaC Runtime Compliance Management Process – check compliance rules



# laC Runtime Compliance Management Process – fix compliance violations

- Fix the compliance violations that might have been found during checking.

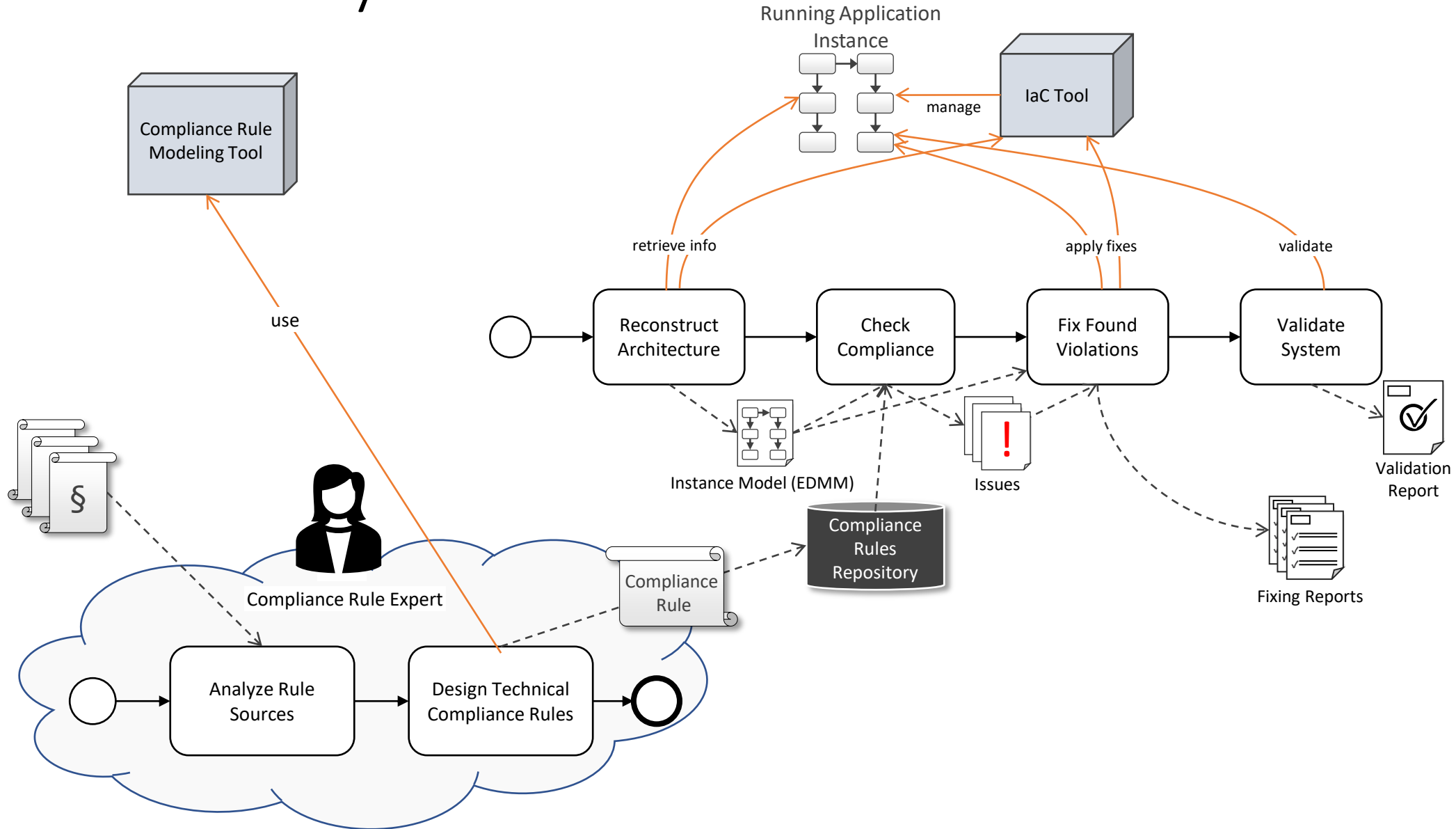
# IaC Runtime Compliance Management Process – fix compliance violations



# laC Runtime Compliance Management Process – validate system

- Ensure that the potential fixes conducted in the previous step did not affect the integrity of the application instance.

# IaC Runtime Compliance Management Process – validate system

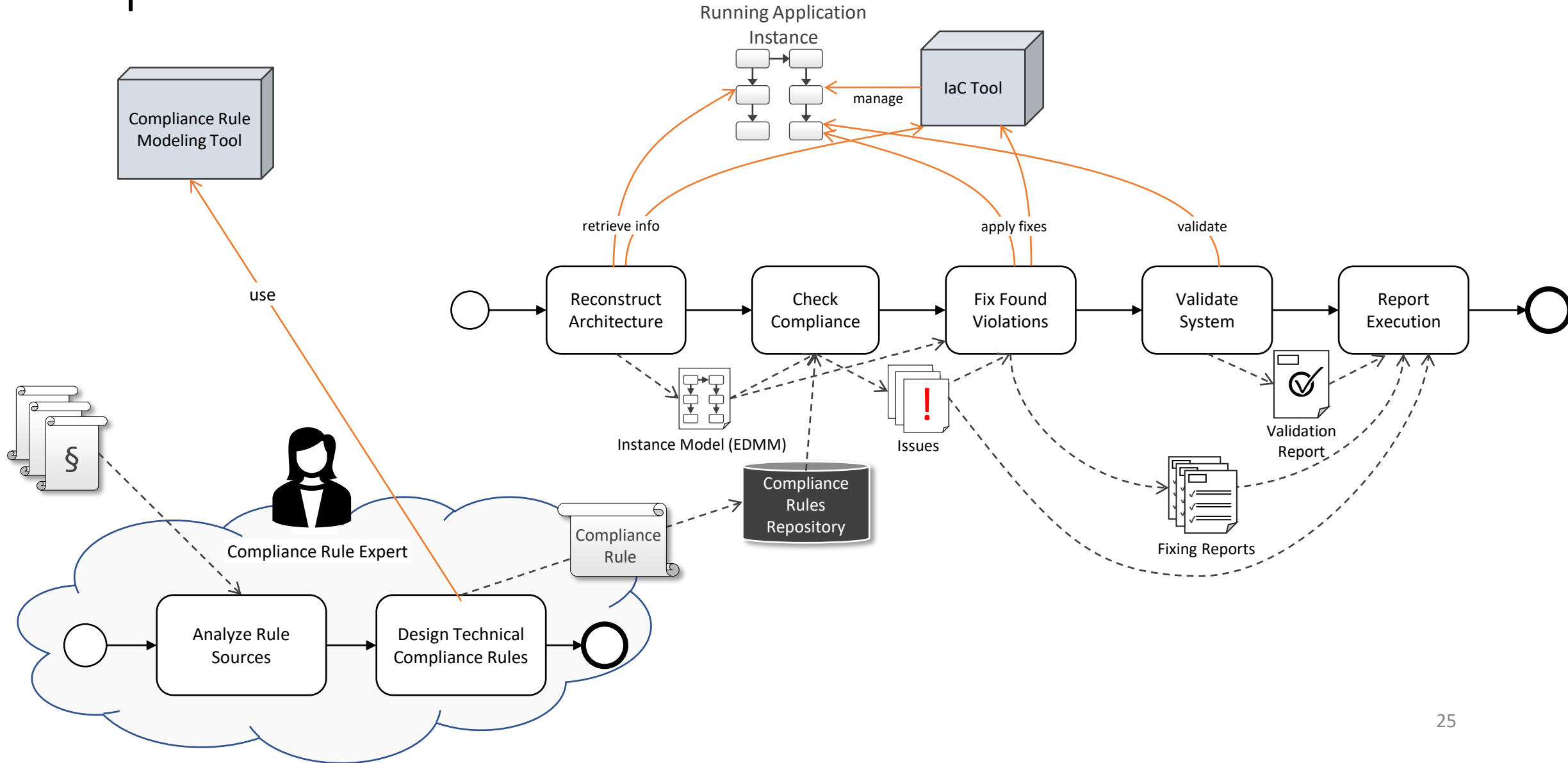


# laC Runtime Compliance Management Process – report execution

- Report a summary of the execution to external parties, e.g.,:
  - An administrator that needs to take manual fixing steps.
  - A manager.
  - An external system that extends the process with further steps.



# IaC Runtime Compliance Management Process – report execution



# IACMF Extensibility

- Many IaC tools exist.
- Many reconstruction approaches exist.
- Many checking approaches possible.
- ...
- IACMF is plugin-based
  - Every step in the IACMF workflow is extensible.
  - Plugin interfaces are predefined.
  - Suitable plugins for different scenarios can be chosen and customized.
  - Currently 11 reusable plugins are implemented.

# IACMF Extensibility – Plugins Overview

#	Plugin Name	Plugin Type	Functionality
1	opentosca-container-model-creation-plugin	Initial instance model creation	Allows creating an instance model for applications deployed using <a href="#">OpenTOSCA Container</a> .
2	kubernetes-model-creation-plugin	Initial instance model creation	allows creating an instance model for applications managed by the <a href="#">Kubernetes container orchestrator</a> .
3	manula-model-creation-plugin	Initial instance model creation	allows retrieving an existing EDMM-based instance model that is created using an external tool, such as a text editor or <a href="#">Eclipse Winery</a> .
4	docker-refinement-plugin	Instance model refinement	updates the instance model with information about all the Docker containers hosted on any of the Docker engine components that exist in the input instance model.
5	mysql-db-refinement-plugin	Instance model refinement	updates the instance model with information about the users of each of the MySQL databases in the input instance model.
6	bash-refinement-plugin	Instance model refinement	Allows running a customizable <i>bash</i> script over <i>ssh</i> on any Linux-based VM component in the input instance model to retrieve and process information from the component and store them in the instance model.
7	subgraph-matching-checking-plugin	Compliance rule checking	Checks the fulfillment of compliance rules using <a href="#">the approach proposed by Krieger et al.</a>
8	docker-fixing-plugin	Compliance violation fixing	Allows fixing violations related to having unexpected Docker containers running in the application instance.
9	remove-mysql-db-users-fixing-plugin	Compliance violation fixing	Allows fixing violations related to having unauthorized users for MySQL databases.
10	bash-fixing-plugin	Compliance violation fixing	Allows running a customizable <i>bash</i> script over <i>ssh</i> on a Linux-based virtual machine to fix violations related to it.
11	smtp-email-sending-plugin	Execution reporting	Sends a human-readable compliance job execution report to an email address.

# Use Cases

- We present three use cases that demonstrate the usage of IACMF.
- Use cases 1&2 are also part of a video demonstration you will watch before the interview.

# Use Case 1 – operating system STIGs

- STIGs: security technical implementation guides.
- Cybersecurity requirements for certain products (e.g., operating systems).
- Published by the US department of defence (DoD).
- Basis for many security guidelines/baselines.

# Use Case 1 – operating system STIGs

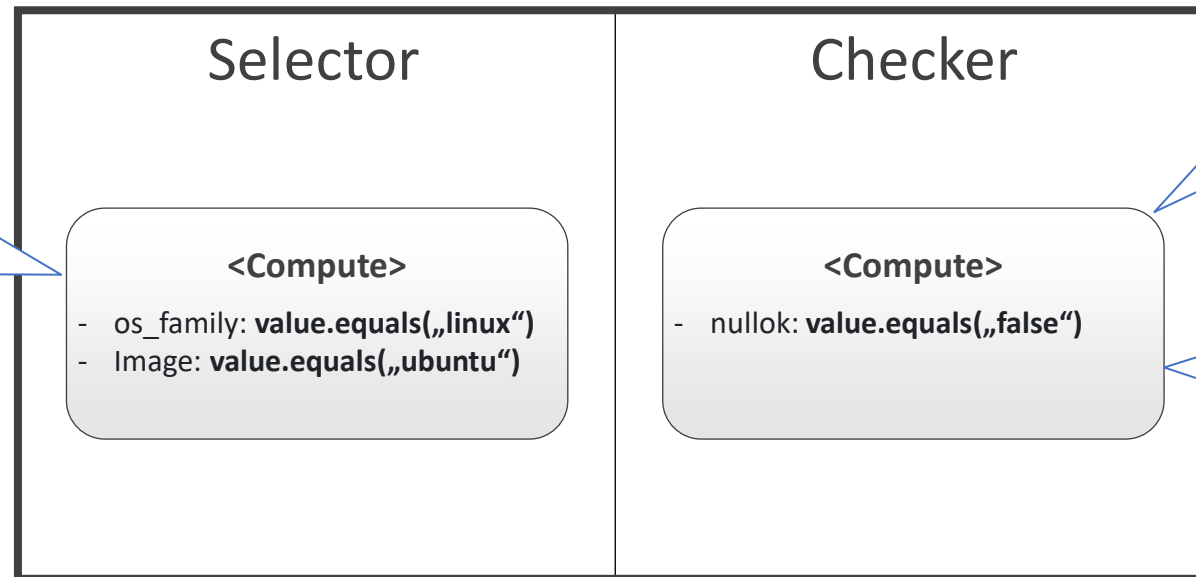
- Example:
  - Canonical Ubuntu 20.04 LTS STIGs (**169 rules**)
  - **STIG ID:** UBTU-20-010463
  - The Ubuntu operating system must not allow accounts configured with blank or null passwords.
- How can we implement and check this rule using IACMF?

# Use Case 1 – operating system STIGs

- Technical compliance rule design
  - Plugin: subgraph-matching-checking-plugin
  - Rule Modeling Tool: *Eclipse Winery*

Match all components that represent ubuntu virtual machines in the instance model.

A pair of EDMM-models



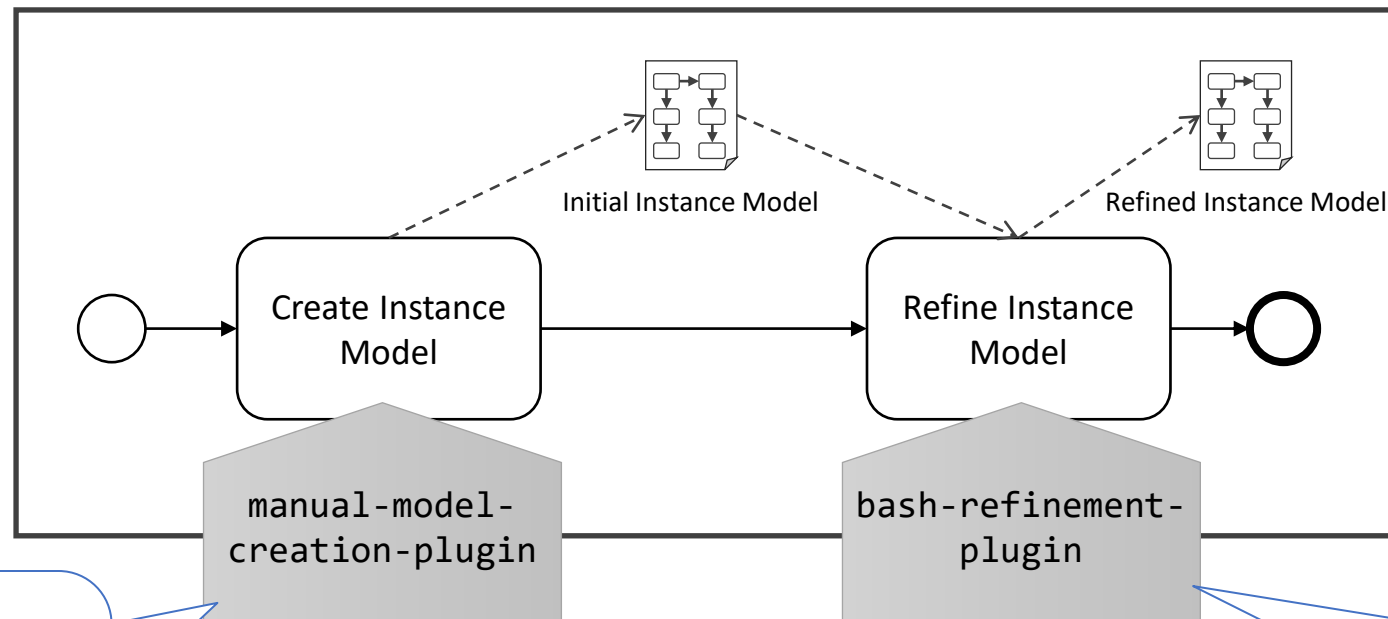
Compliance Rule

Every matched component must have a property “nullok” with a value “false”.

But where do we get this information from?  
- Architectural reconstruction!

# Use Case 1 – operating system STIGs

- Architectural reconstruction



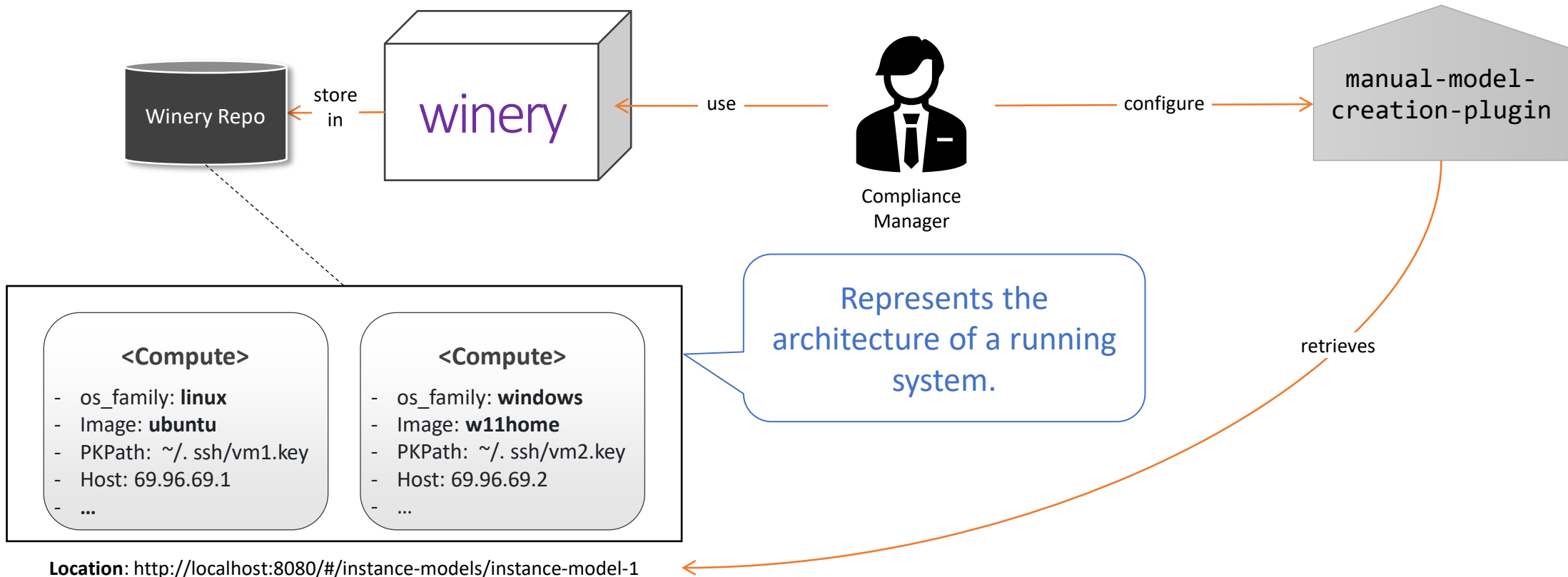
Imports an existing instance model that describes the production system.

Retrieves information about running OS components by running a **bash** script over **ssh**.



# Use Case 1 – operating system STIGs

- Architectural reconstruction – initial instance model creation



# Use Case 1 – operating system STIGs

- Architectural reconstruction – instance model refinement

Finds out if the option „nullok“ is used in the config file:  
/etc/pam.d/common-password



Compliance Manager

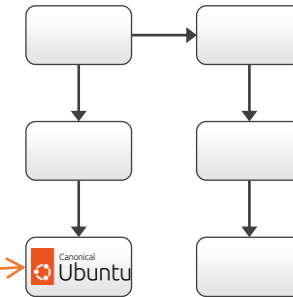
```
[[ ! -z $(sudo grep nullok /etc/pam.d/common-password) ]]  
&& echo 'true' || echo 'false'
```

configure

bash-refinement-  
plugin

execute command  
via ssh on ubuntu  
VMs

Running Application  
Instance



<Compute>

- os\_family: **linux**
- Image: **ubuntu**
- PKPath: ~/. ssh/vm1.key
- Host: 69.96.69.1
- ...

<Compute>

- os\_family: **windows**
- Image: **w11home**
- PKPath: ~/. ssh/vm2.key
- Host: 69.96.69.2
- ...

Initial instance model

find ubuntu VMs

generate

<Compute>

- os\_family: **linux**
- Image: **ubuntu**
- PKPath: ~/. ssh/vm1.key
- Host: 69.96.69.1
- **nullok: true**
- ...

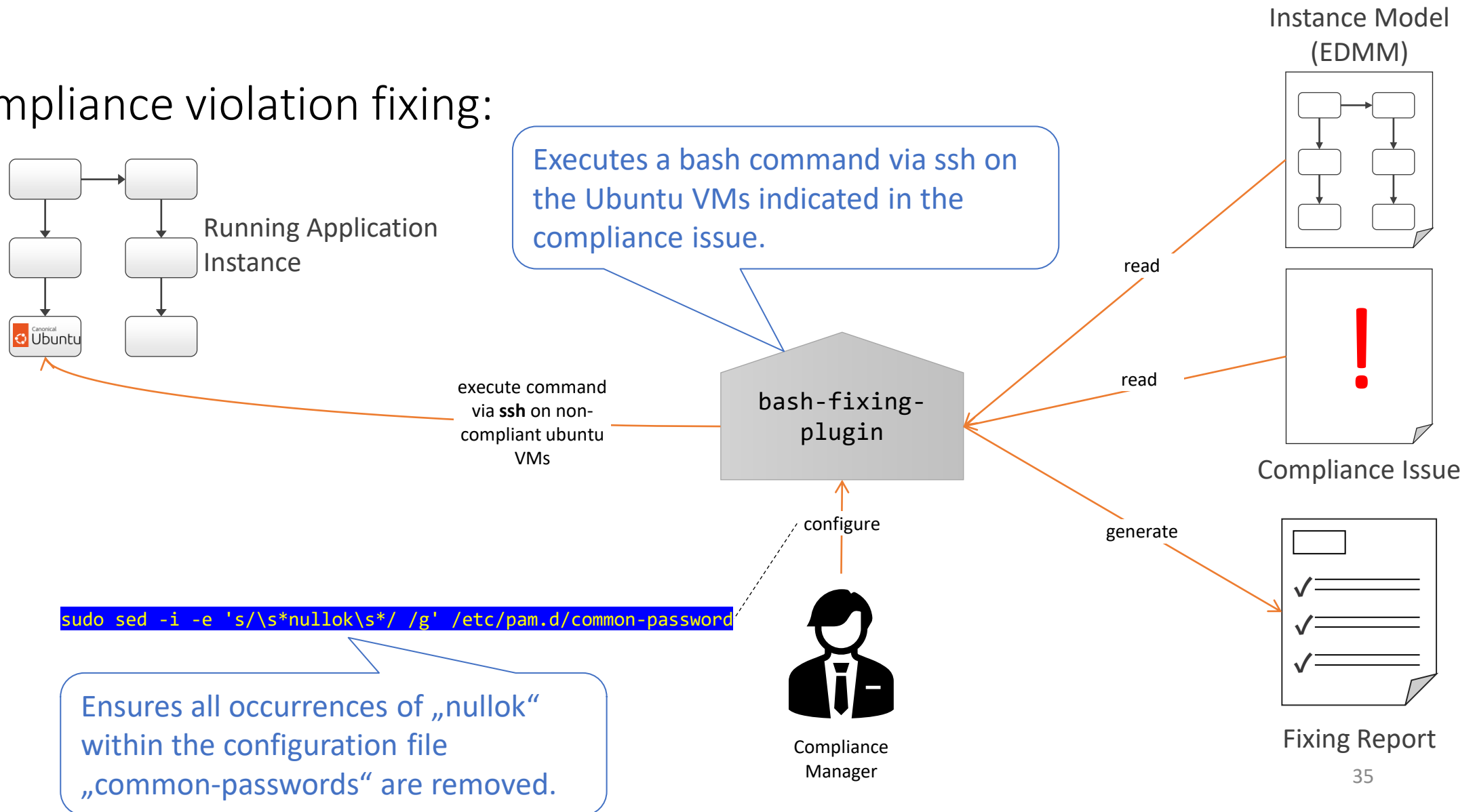
<Compute>

- os\_family: **windows**
- Image: **w11home**
- PKPath: ~/. ssh/vm2.key
- Host: 69.96.69.2
- ...

Refined instance model

# Use Case 1 – operating system STIGs

- Compliance violation fixing:

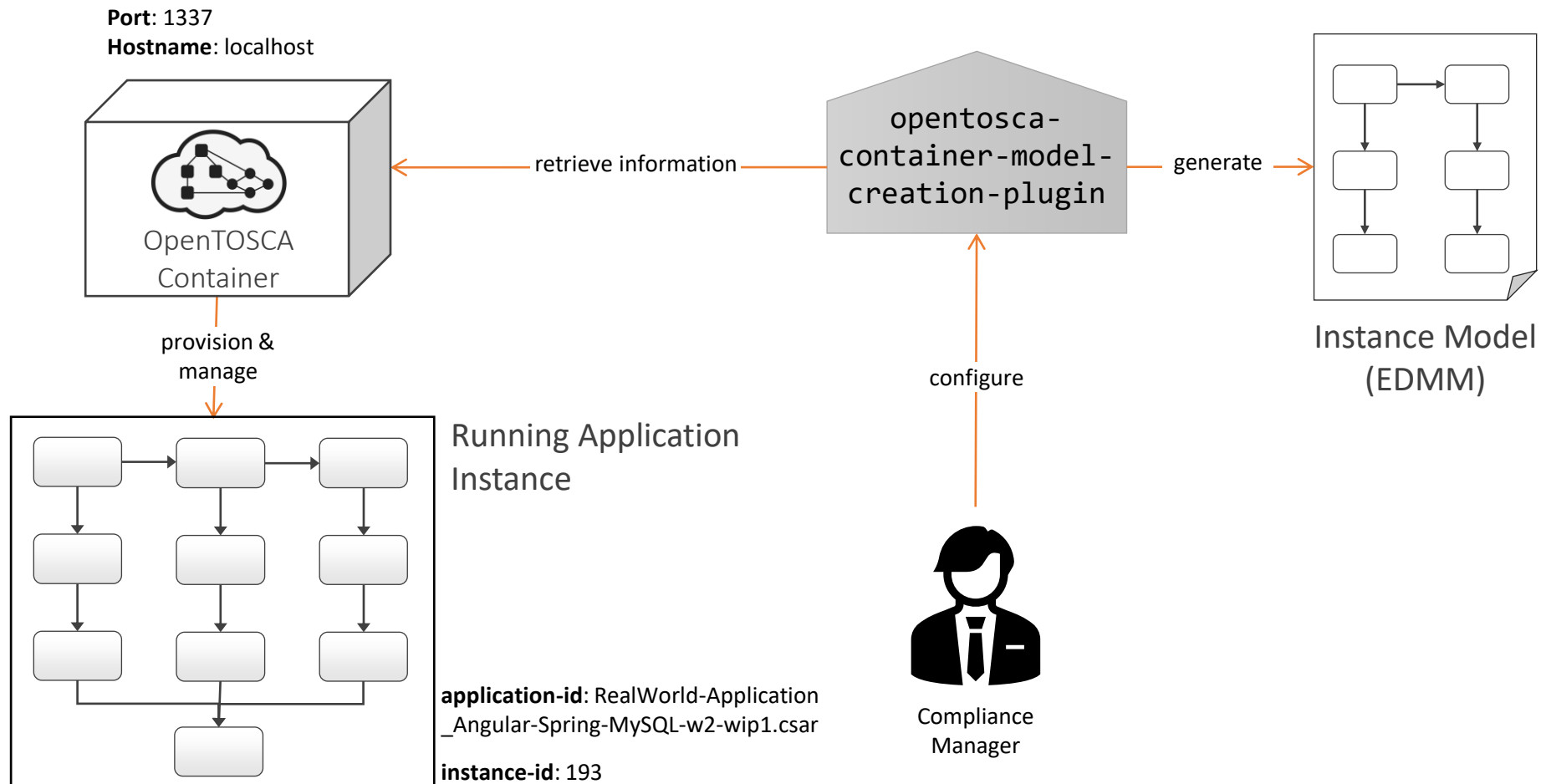


# Use Case 2 – unauthorized DB users

- Production system:
  - Sample cloud application
    - Modeled using TOSCA
    - Provisioned using the OpenTOSCA Container
    - Three stacks using the same docker engine.

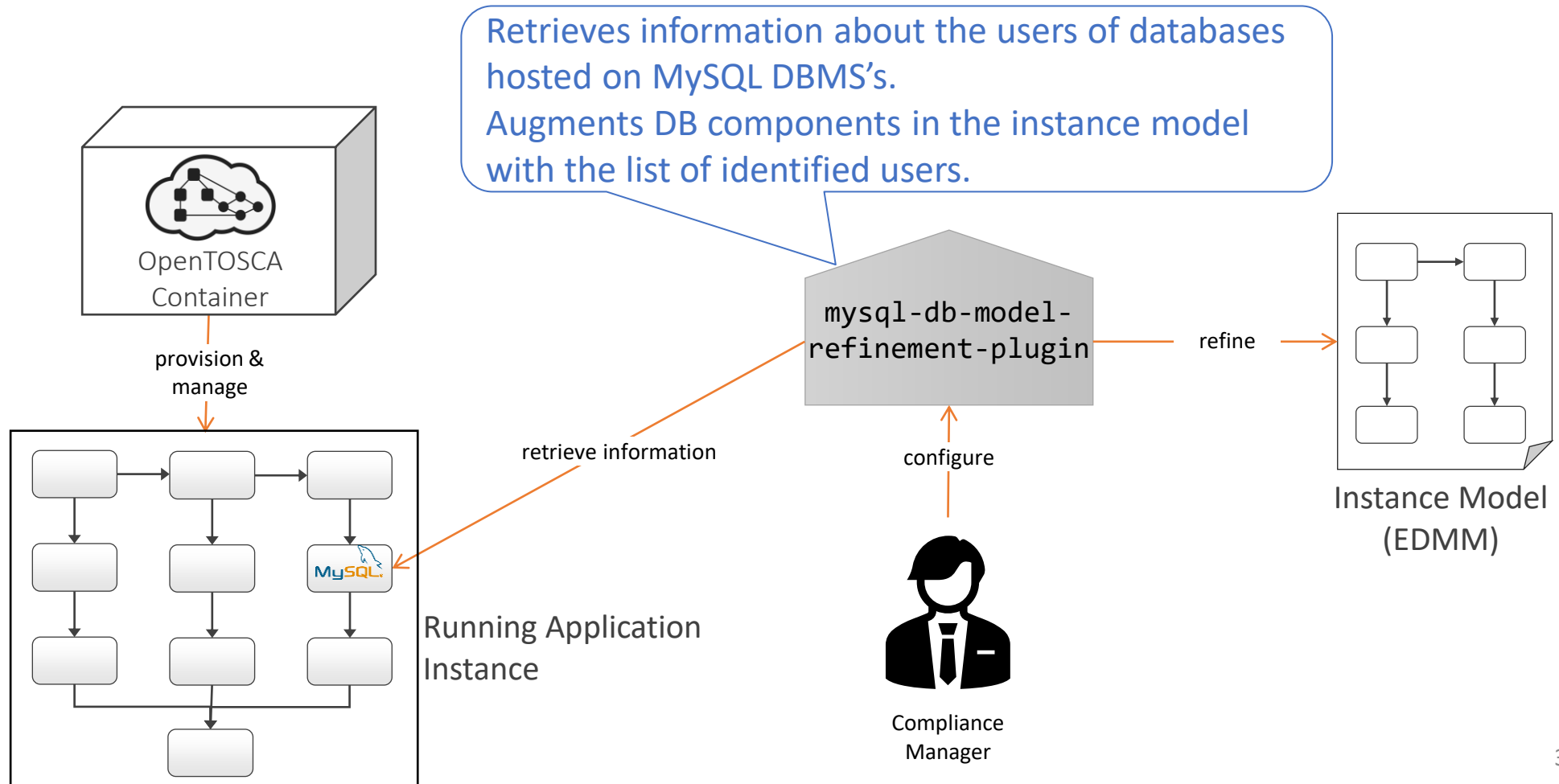
# Use Case 2 – unauthorized DB users

- Architectural reconstruction – initial instance model creation:



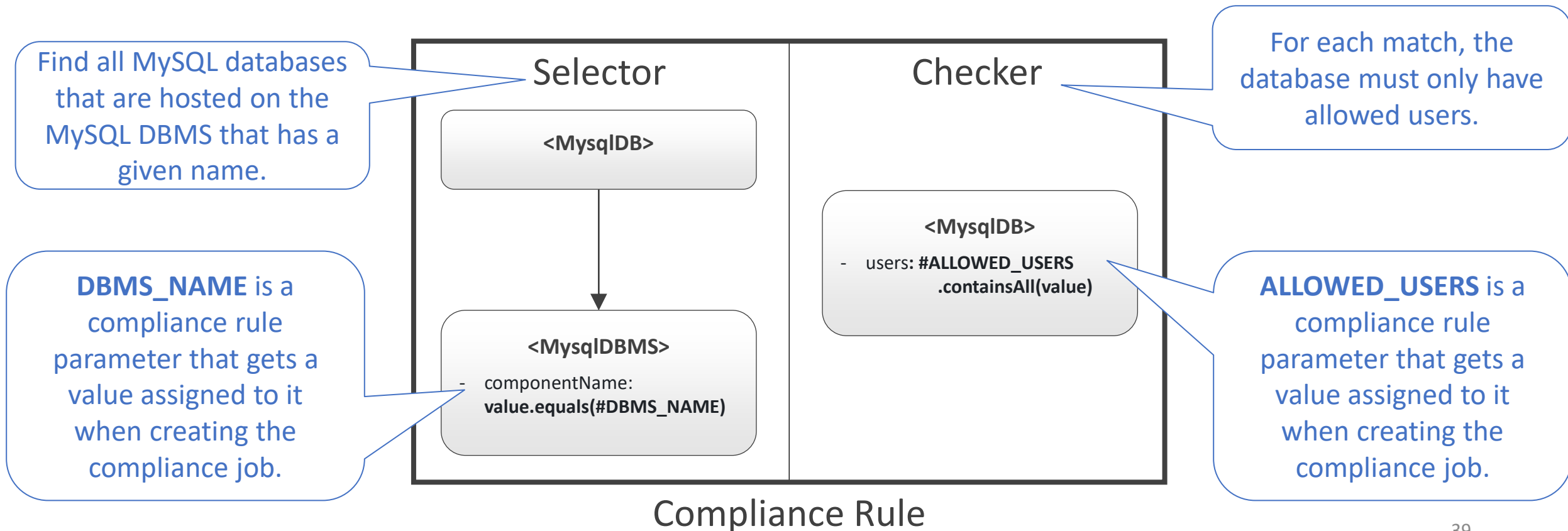
# Use Case 2 – unauthorized DB users

- Architectural reconstruction – instance model refinement:



# Use Case 2 – unauthorized DB users

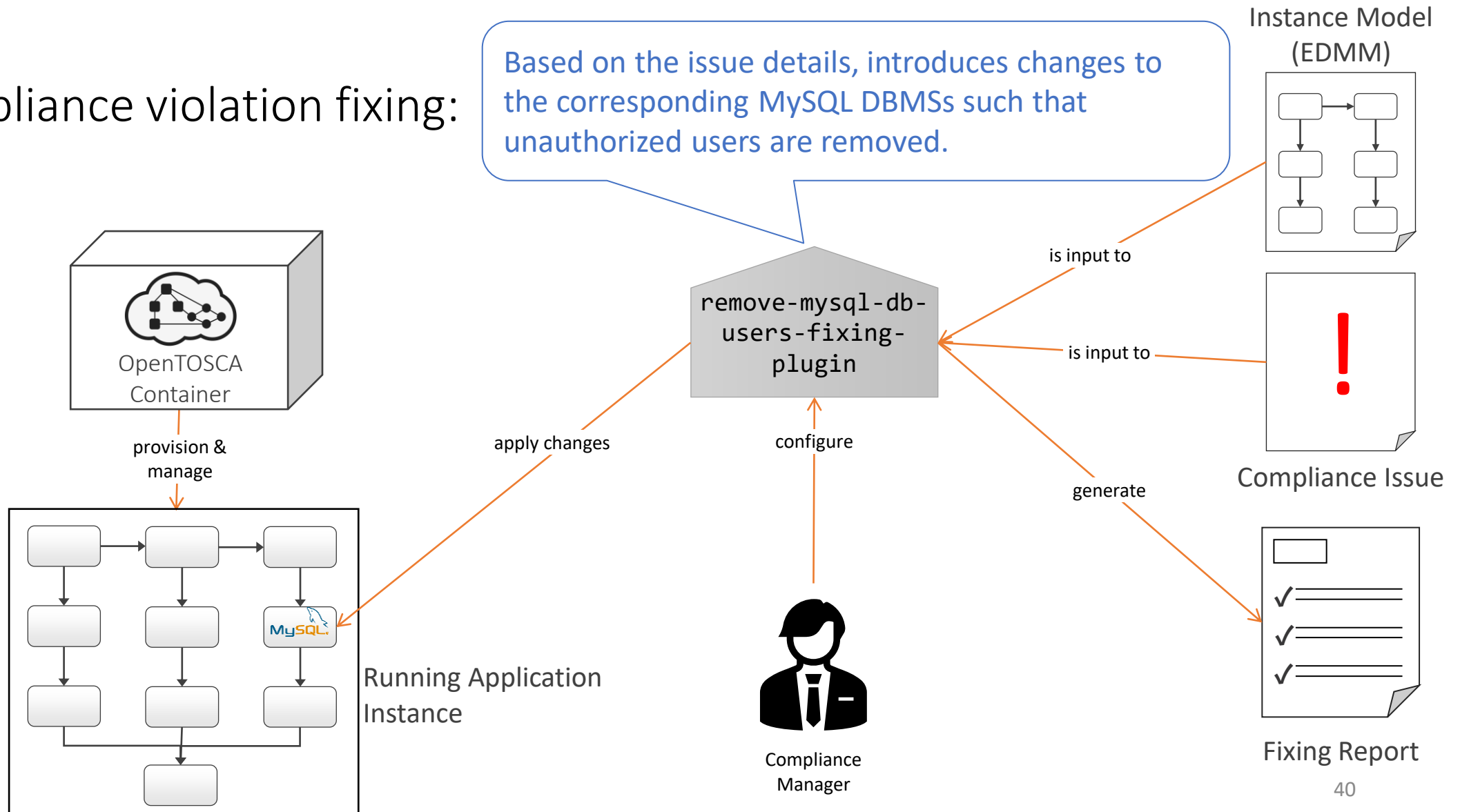
- Compliance rule checking



# Use Case 2 – unauthorized DB users

- Compliance violation fixing:

Based on the issue details, introduces changes to the corresponding MySQL DBMSs such that unauthorized users are removed.

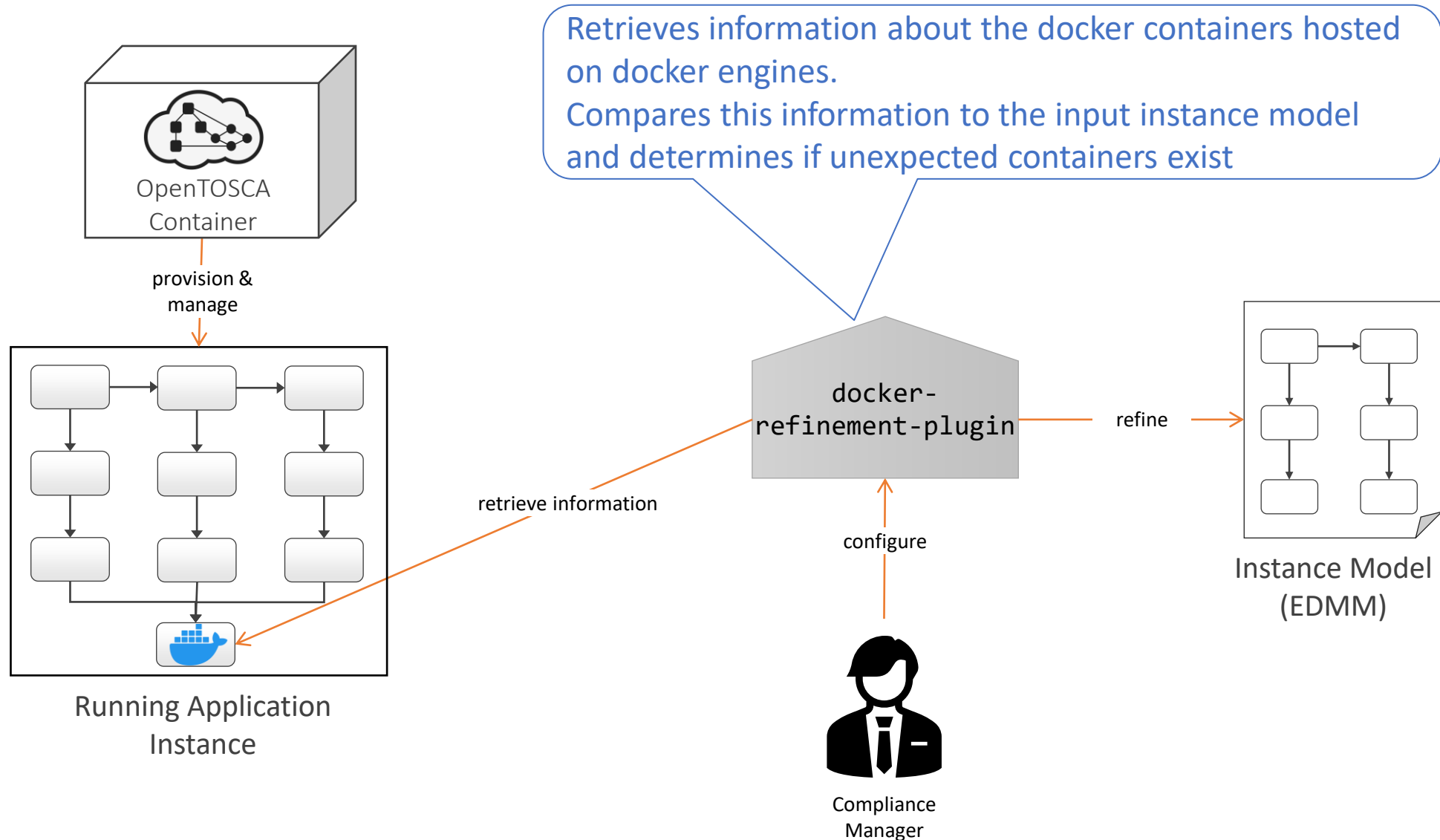




# Use Case 3 – unexpected Docker containers

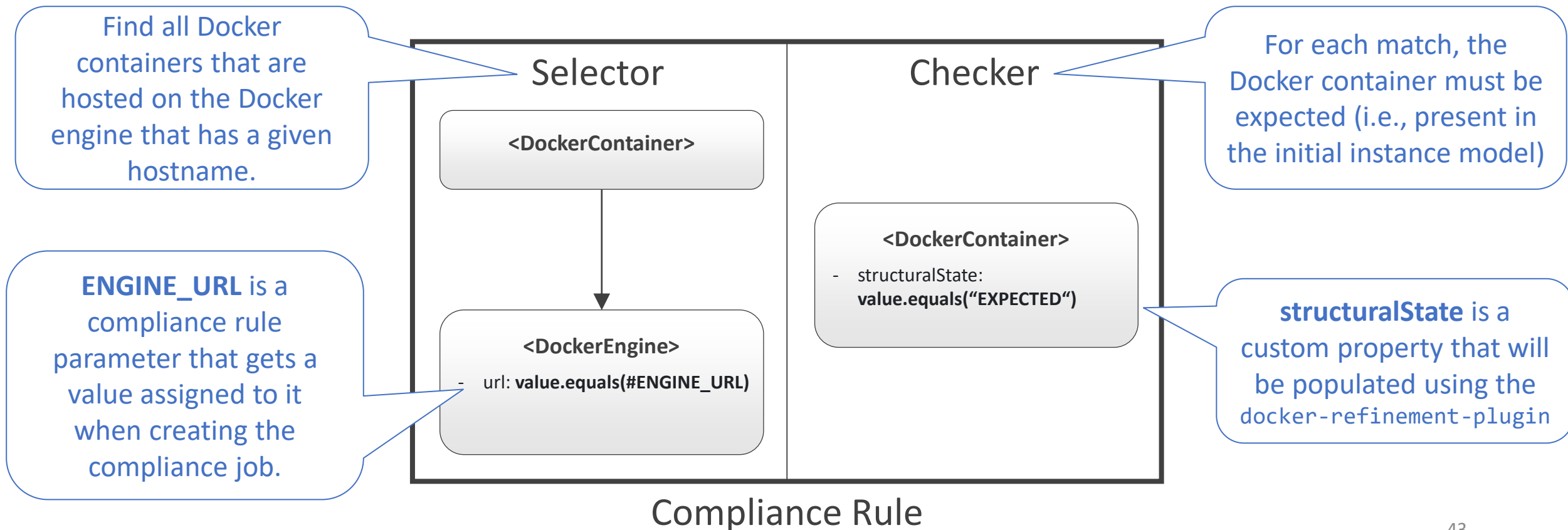
- Same production system as use case 2.
- Same initial instance model creation plugin (`opentosca-container-model-creation-plugin`).
- Different model refinement plugin.

# Use Case 3 – unexpected Docker containers



# Use Case 3 – unexpected Docker containers

- Compliance rule checking



# Use Case 3 – unexpected Docker containers

- Compliance violation fixing:

