Vagrant and Ansible

Two software tools to create and manage your custom VMs

Vagrant and Ansible

Highlights

- Overview on the software tools
  - *Why do you should use them*
  - *Install them*
- Details about the configuration files related to two case studies:
  - *Simple Search Engine on Twitter Data (exam CPM)*
    - Configure a VM by using VirtualBox
  - *The MIDAS project (EC Research project)*
    - Configure a VM by using VirtualBox
    - Reply the configuration by changing the VM Hypervisor

Vagrant and Ansible

- **The IT history:**
  - *Mainframe computing*
  - *PC Computing*
  - *Network Computing*
  - *Internet Computing*
  - *Grid Computing*
  - *Cloud Computing*

- **The Virtualization is the enabling technology:**
  - Hardware virtualization or platform virtualization
  - Host machine
  - Guest machine
  - Hypervisor or Virtual Machine Monitor

The Open Source Tools selected

- **Vagrant** https://www.vagrantup.com/downloads.html
  - A free software tool for creating customizable, lightweight, reproducible, and portable development environments made up of Virtual Machine Images (VMIs).
    - *(Windows, Linux, Macintosh)*
- **Ansible** http://docs.ansible.com
  - An IT automation tool to automatically keep different development environments aligned - in our case the ones of the MIDAS project partners.
    - *(installed inside the Guest Virtual Machine)*
- **VirtualBox** https://www.virtualbox.org/wiki/Downloads
  - A powerful x86 and AMD64/Intel64 virtualization product that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2.
    - *(Windows, Linux, Macintosh)*

….more on Vagrant

- **Features**
  - Provide the team developers with a reproducible and portable work environment;
  - Support different VMMs, including VirtualBox, VMware, Amazon Web Services (EC2).
  - Industry-standard provisioning tools, such as:
    - *Chef, Ansible, shell scripts, Puppet*
- **Why you should use them?**
  - Research projects that need of a custom environment
  - Experiments, you can set up an environment with high computing resources (on Cloud) to run a lot of experiments, without spend a lot money to buy new powerful machines

….more on Vagrant

Getting started with Vagrant

- Open a terminal or command prompt and type:

  ```
  $ vagrant -v
  ```

- To initialize a VM configuration starting by an existing Virtual Machine Image, you can type:

  ```
  $ vagrant init precise64 http://files.vagrantup.com/precise64.box
  ```

- To running your VM :

  ```
  $ vagrant up
  ```

- Ok, now a Ubuntu 12.04 LTS 64-bit is running in your local machine! If you want access to it, you can type:

  ```
  $ vagrant ssh
  ```

….more on Vagrant

Suspend, Halt, Destroy your local Virtual Machine

- Suspending the local VM
  - *$ vagrant suspend*

Save the current running state of the VM and stop it. You can resume your VM by typing
$ vagrant up
Pro: only 5 o 10 seconds to stop and start your work
Cons: disk space to store the VM and its status

- Halting the local VM
  - *$ vagrant halt*

Gracefully shutdown the guest OS. You can resume your VM by typing
$ vagrant up
Pro: no disk space to store the status of VM
Cons: extra time to start from a cold boot

- Destroying the local VM
  - *$ vagrant destroy*

Remove all traces of the guest machine from your system. You can reconfigure your VM by typing
$ vagrant up
Pro: the disk space is left clean
Cons: extra time to re-import and re-provision the VM

….more on Ansible

- **Playbooks are expressed in YAML format**
  - They have a minimum of syntax, which intentionally tries to not be a programming language or script, but rather a model of a configuration or a process.
- **Each playbook is composed of one or more ‘plays’ in a list.**
  - The goal of a play is to map a group of hosts to some well defined roles, represented by things ansible calls tasks. At a basic level, a task is nothing more than a call to an ansible module.
  - It is possible to orchestrate multi-machine deployments, running certain steps on all machines in the webservers group, then certain steps on the database server group, then more commands back on the webservers group, etc.
- **Tasks list**
  - Each play contains a list of tasks. Tasks are executed in order, one at a time, against all machines matched by the host pattern, before moving on to the next task. The goal of each task is to execute a module, with very specific arguments. Variables can be used in arguments to modules.
  - Modules are ‘idempotent’, meaning if you run them again, they will make only the changes they must in order to bring the system to the desired state. This makes it very safe to rerun the same playbook multiple times.

Vagrant + Ansible

- **To use Ansible with Vagrant you can type**
  - $ vagrant up

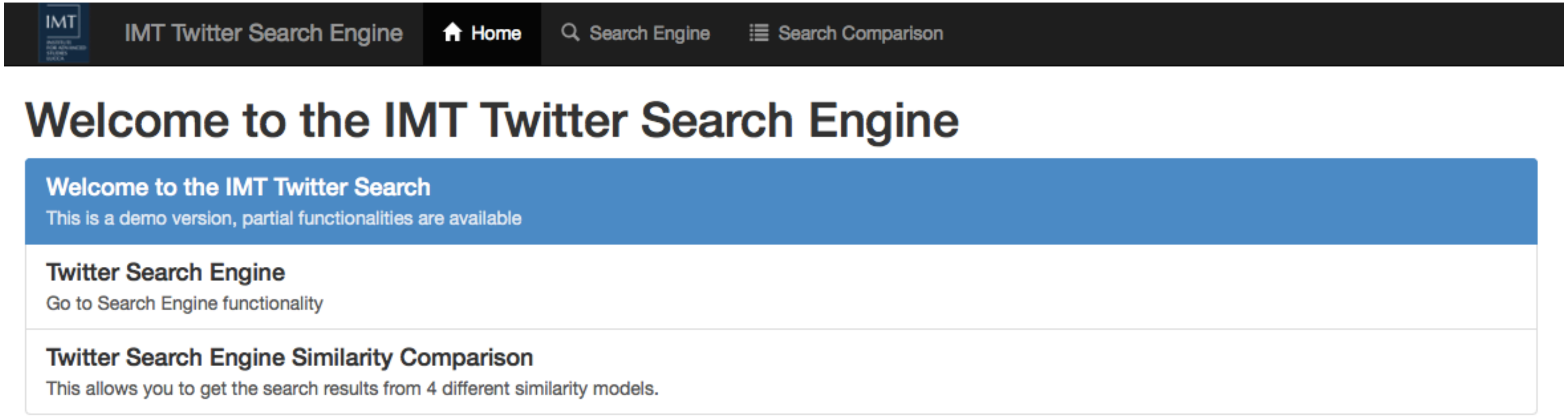
The Ansible playbook is automatically installed (the first time) and run during calls this command. The VM configuration expressed in the YAML format is read and executed.

  - $ vagrant provision

This command allows you to re-configure an already running virtual machine to just run the provisioner. This allows, for example, to run apt-updates on the running VM, or to update the packages behind the running VM

CASE STUDY 1: SIMPLE SEARCH ENGINE on twitter data

http://friulano.isti.cnr.it:5001



Welcome to the IMT Twitter Search Engine

Welcome to the IMT Twitter Search

This is a demo version, partial functionalities are available

Twitter Search Engine

Go to Search Engine functionality

Twitter Search Engine Similarity Comparison

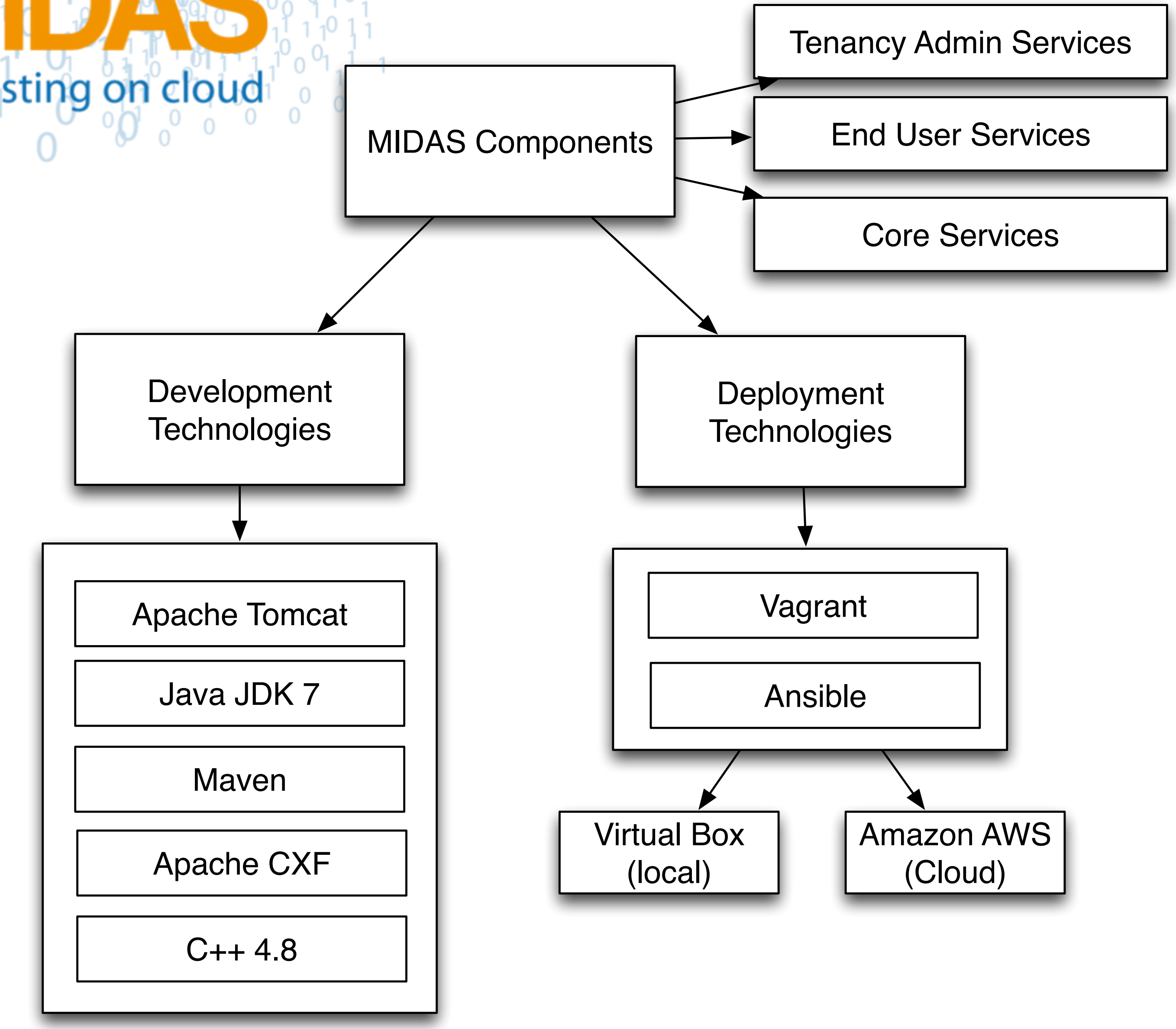
This allows you to get the search results from 4 different similarity models.

CASE STUDY 2: The MIDAS project

- **Model and Inference Driven – Automated testing on Services architectures**
  - Goal: We want to make it easier for SMEs to benefit from SOA testing
  - Developers and researchers will have access to a MIDAS Platform as a Service that allows deploying on the MIDAS SaaS innovative testing methods and tools in a controlled manner. MIDAS project aims to build a strong community of testing tool developers beyond the MIDAS partners.
  - The MIDAS project defines and evaluates business models for providing and distributing accessible, affordable, automated SOA testing facilities through innovative channels such as SaaS, that allow actors such as SME, traditionally outside the SOA testing market, to enter. SMEs could play a very important role in European economy by accessing to this kind of innovations.

MIDAS testing on cloud

CASE STUDY 2: The MIDAS project



MIDAS Components

Tenancy Admin Services

End User Services

Core Services

Development Technologies

Apache Tomcat

Java JDK 7

Maven

Apache CXF

C++ 4.8

Deployment Technologies

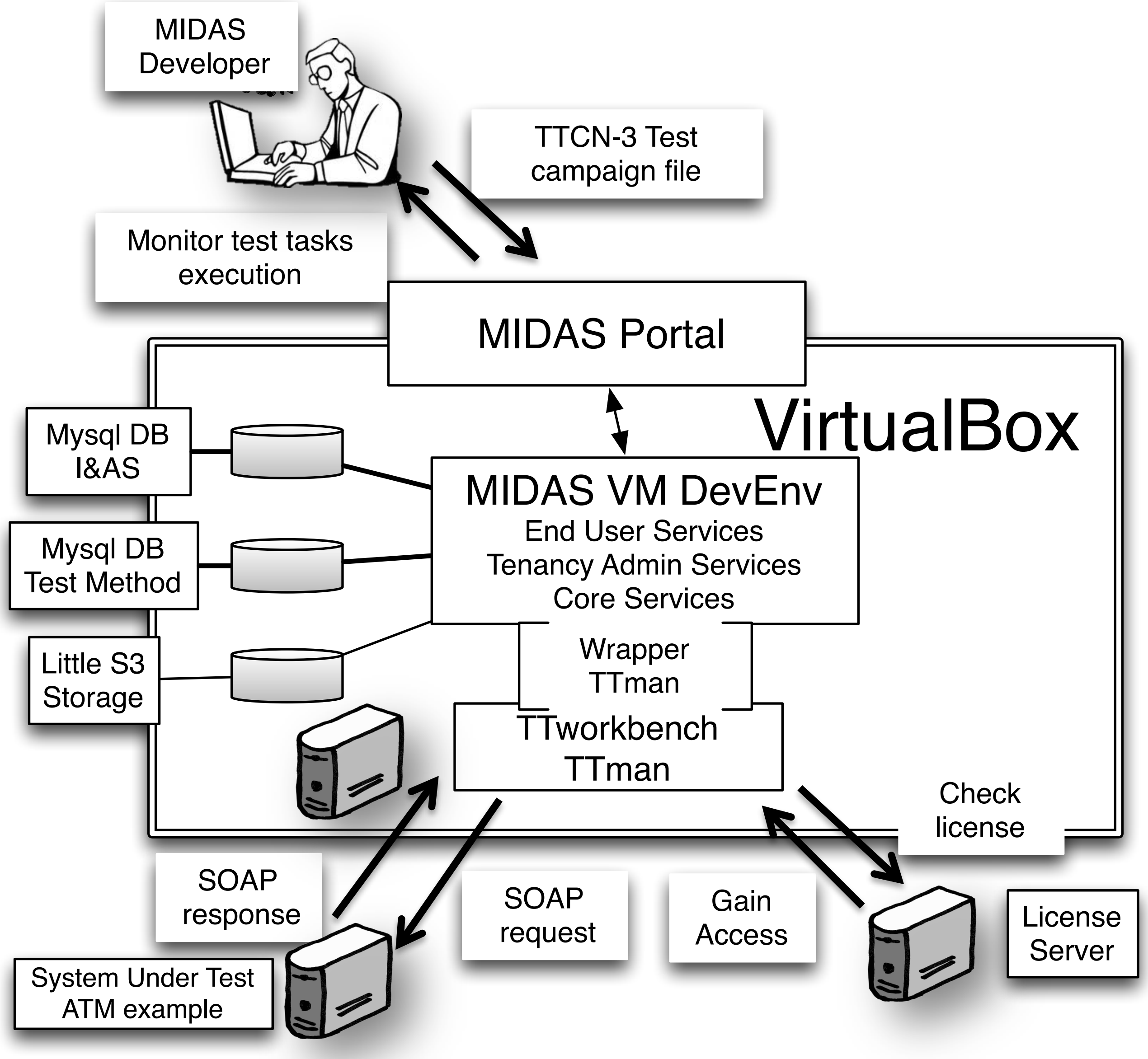
Vagrant

Ansible

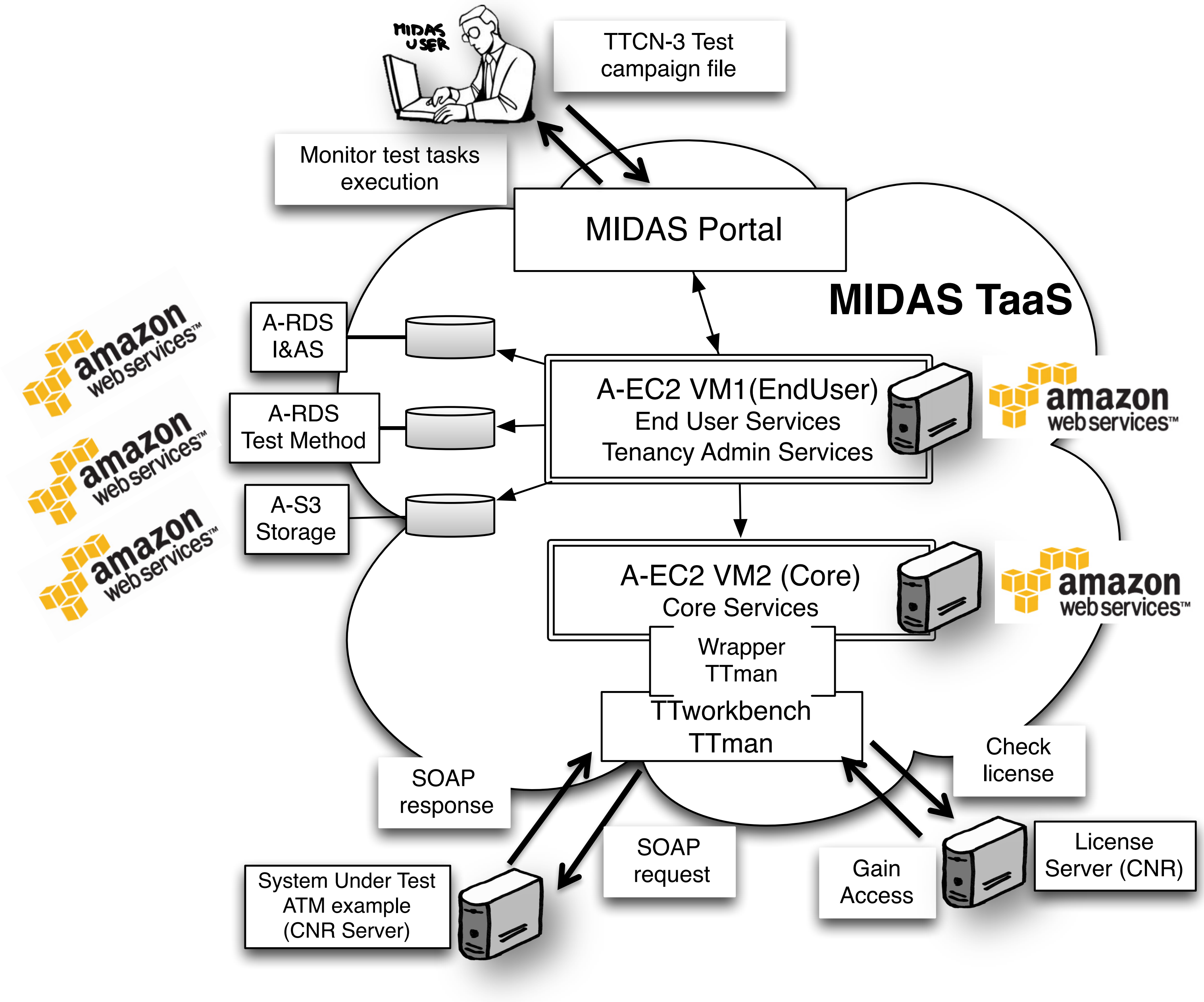
Virtual Box (local)

Amazon AWS (Cloud)

The MIDAS Development Environment



MIDAS TaaS “The First Prototype on cloud”



TTCN-3 Test campaign file

Monitor test tasks execution

MIDAS Portal

MIDAS TaaS

A-RDS I&AS

A-RDS Test Method

A-S3 Storage

A-EC2 VM1(EndUser)
End User Services
Tenancy Admin Services

A-EC2 VM2 (Core)
Core Services

Wrapper TTman

TTworkbench TTman

SOAP response

SOAP request

System Under Test
ATM example
(CNR Server)

Check license

Gain Access

License Server (CNR)

THANK YOU!

Alberto De Francesco

alberto.defrancesco@isti.cnr.it