Contents

Chapter 1

Introduction

This document explains how to use VEP from a user point of view. In the next chapter will be explained how to create a CEE, how to set up your application and how finally tostart and stop it.

Before to starts, it is better to explain some VEP elements and rules. CEE stands for Constrained Execution Environment. It is the document that describes the whole enviroment where the applications run. It contains all the allowed handlers,constraints and rules choosen to deploy the applications. The Deployment Document is a part of an application document and it is sent to the VEP to add VM to the application. The Application Document is the document that describes the complete application, ut contains all the OVF, VMs, reservations and is often send inside the CEE.

A usual operation flow in VEP should follow these steps:

- Creation of a CEE with an Application Document inside **??**
- Retriving of the application id
- Send the Deployment Document
- Start the Application
- Stop the Application

The VEP web user interface is still on development so to act all the operations the user should use a rest client tool. There is the RESTClient tools that is compatible with many browser that can be downloaded at `http://restclient.net`.

Chapter 2

REST requests

2.1 CEE Creation

You can find a description of the fields of a CEE creation request in table 2.1. Here is an example of what such a request could look like.

```
HEADERS:
X-Username:<username>
Content-Type:application/json

BODY:

{   "name": "new test",
    "state": "active",
    "VMHandlers": [
        {
            "href": "http://vep.fr/vmhandlers/1"
        },
        {
            "href": "http://vep.fr/vmhandlers/3"
        }
    ],
    "applications": [
        {
            "name": "ApplicationSLA",
```

```
            "OVFDeployment":false,
            "OVFDescriptors": [
                {
                   "OVFFile":"<Escaped String of the OVF>"
                }
            ]
       }
    ],
    "defaultMapping": [
        {
            "type": "VM",
            "virtualSystem": {
                "href": "#ubu1"
            },
            "handler": {
                "href": "http://vep.fr/vmhandlers/3"
            },
            "corecount": 1,
            "ram":256,
            "cpuFreq":500
        }
    ],
    "constraintsMapping":[
      {
          "constraint":
              {"href":"http://vep.fr/constraints/2"},
          "parameter":"FR",
          "virtualSystem":[
              {"href":"#ubu1"}
          ]
      }
    ]
}
```

You should retrieve the cee id of the new CEE when it is actually registered.

2.2 Application id retrieval

Once the CEE is created, the user needs to retrieve the also newly created application's id, so he can then update the by default empty application with the virtual machines to deploy. To retrieve this id, it is necessary to do a GET on `http://<site>/api/cimi/CEE/{ceeId}/applications`. You can find a description of the fields of the response in table 2.2. A sample response could be as in the following example.

```
HEADERS:
X-Username:<username>
Content-Type:application/json

BODY:
{
  "resourceURI": "VEP/AppCollection",
  "id": "http://<site>/cee/{ceeId}/applications",
  "count": 1,
  "applications": [
    {
        "href": "http:/<site>/cee/{ceeId}/application/{appId}",
        "name": "ApplicationSLA",
        reservations: [{reservationId:<reservationId>}]
        //Only if there are any reservations linked to this app
    }
  ]
}
```

As described in the formal table, appId is the id which will be needed in the following request to upload the deployment document.

2.3 Deployment Document

To actually create virtual machines from the CEE, the user needs to send a deployment document using a POST request with the following formatting at

this address:
POST http://<site>/api/cimi/cee/<ceeid>/application/<applicationid>.
You can find a description of the required fields of the HTTP request in table 2.3. An example follows.

```
HEADERS:
X-Username:<username>
Content-Type:application/json

BODY
{
  "VMs": [
    {
        "name": "AppServer-1",
        "virtualSystem": {"href":"#AppServer"},
        "contextualization": {
                    "key":"value",
                    "key2":"value2"
         }
    },
    {
        "name": "AppServer-2",
        "virtualSystem": { "href": "#AppServer"}
     },
    {
        "name": "DBServer-1",
        "virtualSystem": { "href": "#DBServer"}
    }
  ]
}
```

2.4 Start an application

To start an application a user needs to send a POST request at the following address : http://<site>/api/cimi/CEE/{ceeId}/application/{appId}/action/start. Expected result is HTTP/1.1 200 OK only.

2.5 Stop an application

To stop an application a user needs to send a POST request at the following address : `http://<site>/api/cimi/CEE/{ceeId}/application/{appId}/action/stop`. Expected result is HTTP/1.1 200 OK only.

Table 2.1: CEE Creation request description

name	CEE Identification name (mandatory)		
state	'active' or 'check' (mandatory)		
VMHandlers	Array that lists the vmhandlers available in the CEE (mandatory)		
	href	Uri of the VMhandler (mandatory)	
StorageHandlers	As VMHandlers (not mandatory)		
NetworkHandlers	As VMHandlers (not mandatory)		
applications	Array of an application		
	name	Application name (mandatory)	
	OVFDeployment	always false (mandatory)	
	OVFDescriptors	Array of OVF Document (mandatory)	
		OVFFile	Ovf document parsed by an escape function (mandatory)
	reservations	Array listing the reservations	
		virtualSystem	Reference to the VS in the OVF (mandatory)
			href '# + virtualSystem name' (e.g. #vs1) (mandatory)
		count	Number of VMs to reserve
		enddate	End date of the reservation (mandatory)
defaultMapping	Array mapping handlers to virtual systems		
	type	Only "VM" is supported (mandatory)	
	corecount	Specify number of cores for this virtualSystem)	
	ram	Specify amount of memory for this virtualSyste)	
	cpufreq	Specify cpu frequency for this virtualSystem)	
	virtualSystem	Reference to the virtual system in the ovf (mandatory)	
		href	Reference to the VS in the OVF (mandatory)
	handler	**href**	Uri of the handler (mandatory)
constraintsMapping	Array listing the constraints to use in the CEE.		
	parameter	If there is a constraint about country, it must contain the related country code	
	constraint	The constraint to apply (mandatory)	
		href	Uri of the constraint (mandatory)
	virtualSystem	Array of virtual System which have to respect the constraint (mandatory)	
		href	'# + virtualSystem name' (e.g. #vs1) (mandatory)

Table 2.2: Application retrieval response description

resourceURI	VEP/AppCollection		
id	`http://<site>/cee/<ceeId>/applications"`		
applications	Array listing the applications (only one exists by default)		
	href	URL of the application, **needed for next step**	
	name	Name of the application	
	reservations	Array listing the reservations linked to this app, if any	
		reservationId	an integer identifying a reservation

Table 2.3: Deployment Document description

VMs	List of VMs of an application		
	name	Name of the VM	
	virtualSystem	Virtual System this VM refers to	
		href	'# + virtualSystem name' (e.g. #vs1) (mandatory)
	contextualization	Dictionary of key:value to transmit to the VM	
		'key'	'value'(key and value chosen by user)

Chapter 3

Web User Interface

VEP 2.1 introduces new feautures to make the user experience easier; one is the new Web User Interface that let the user to manage completely his CEEs and Application directly from the browser. In the next line this document describes how to use the web user interface, from ”sign in” to ”start application”

3.1 Login

To log into VEP(Fig 3.1) it’s mandatory to have an account. Every user has to sign in, as described in 3.2 and have a certificate released by the VEP’s Administrator. The user has to include his certificate in the browser and then he can log in using username and password.

3.2 New User Registration

As stated in 3.1, to log into VEP it’s mandatory to have an account. To register a new user click on ”Sign In” on the VEP home page and then (Fig 3.2) insert all the required information.

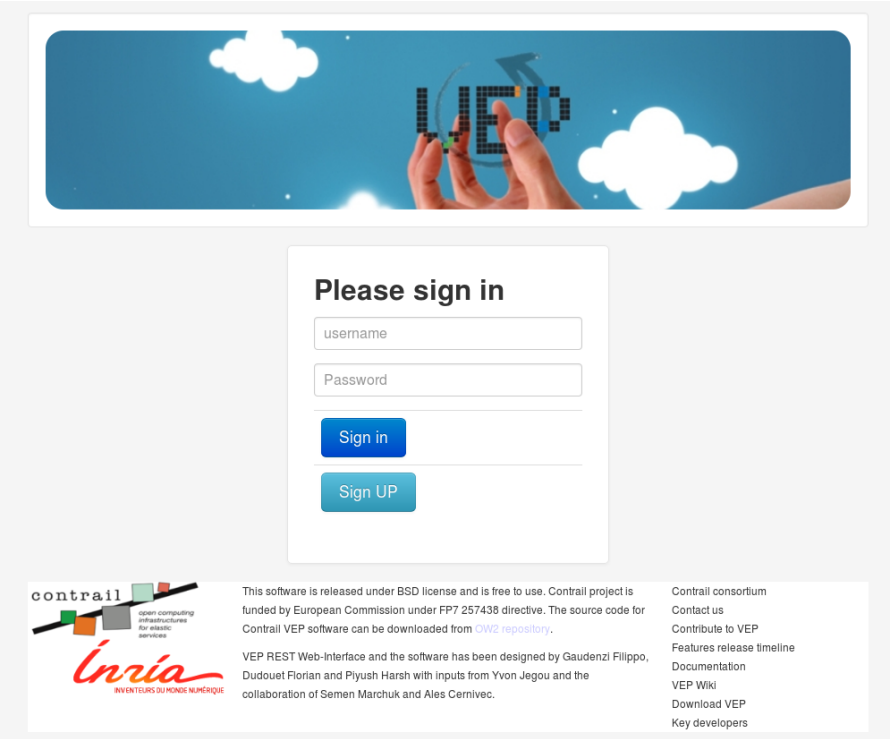

Figure 3.1: Login Page

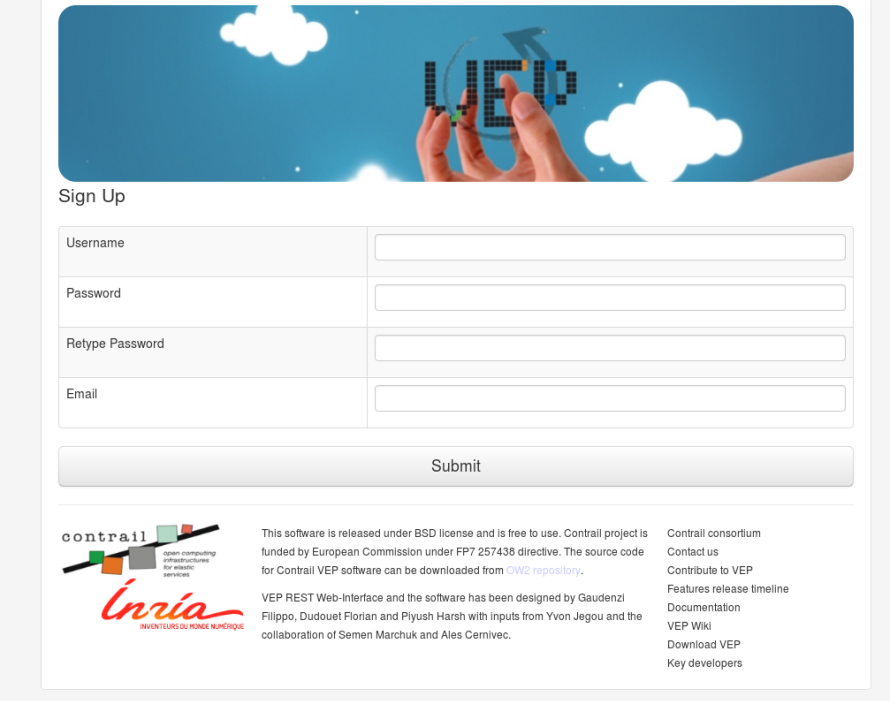

Figure 3.2: Sign In Page

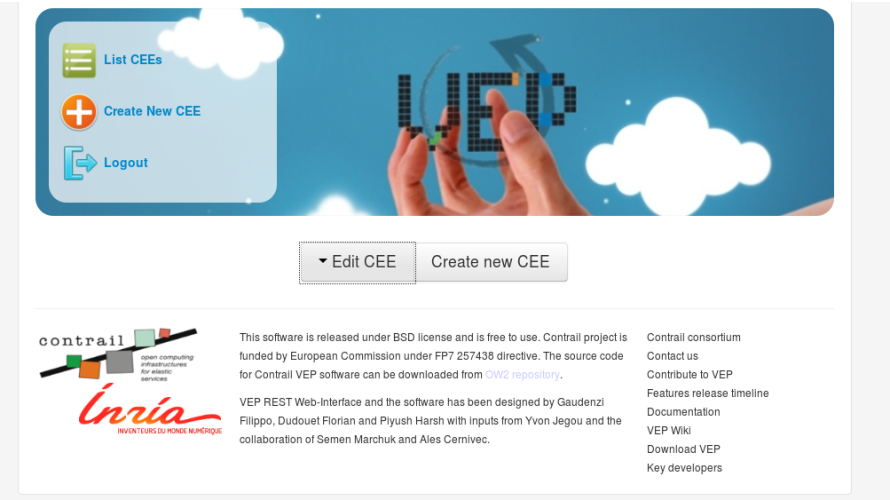

Figure 3.3: List CEEs Page

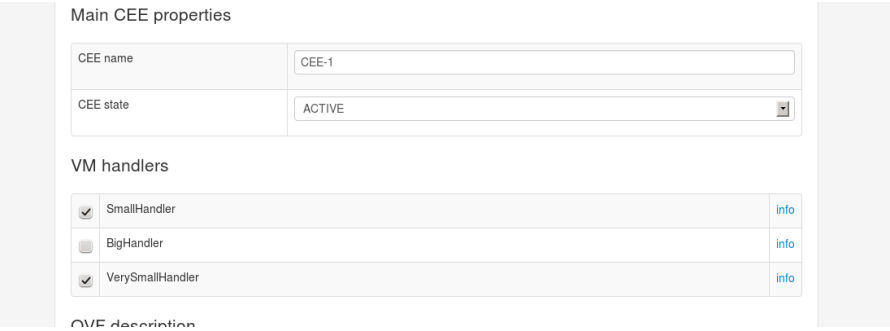

Figure 3.4: Create new CEE Page

3.3 List CEEs Page

As you are logged, you will see this page (Fig 3.3). If you have already some CEEs you will see them clicking on "Edit CEE". Choosing one then you can add VMs to it (3.5). As well clicking on "Create New CEE", you can create a new CEE.

3.4 New CEE Creation

This page (Fig 3.4) is made to create a new CEE. You can easy follow the instructions in the form and fill all the required box. Please remember that before to parse the ovf you have tho choose at least one VMHandler. After you have parsed the OVF, you can specify the amount of resource for each Virtual System. Remeber that VEP follows these rules:

- In case you leave the resource fields empty, VEP will use the low-bounds of the choosen VMHandler. Be carefull that in the case that the values specified in the OVF respect the VMhandler's ranges, VEP will use these values.
- If there are values in the resource fields thay will be alway preferred over OVF and VMHandler values but they have to be higher than the OVF values

Reservations and Constrains have to be filled also after you have parsed the OVF, but they are not mandatory to succeed in creating a CEE.

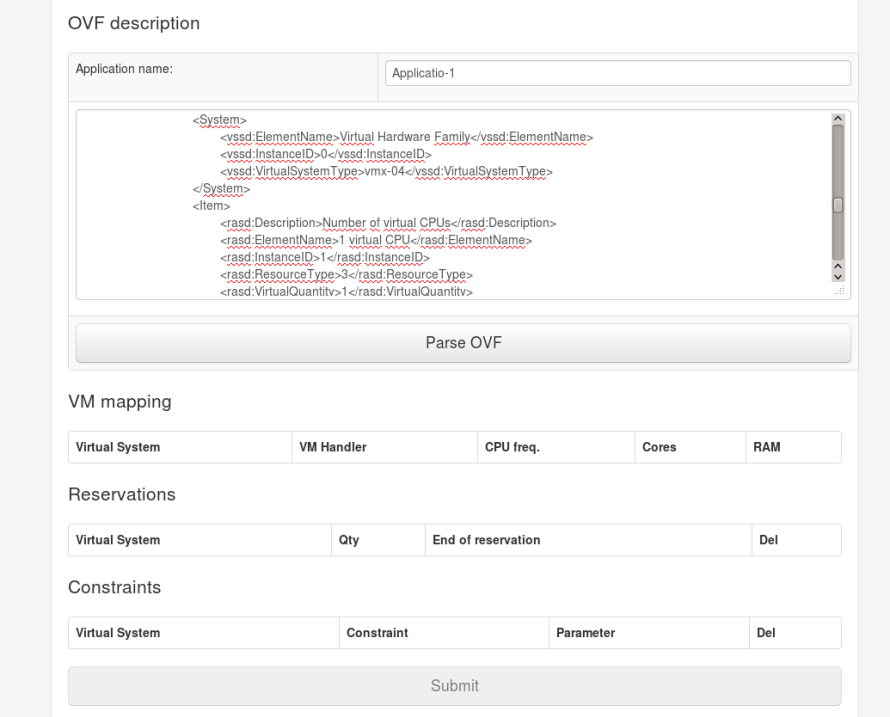

Figure 3.5: Create new CEE Page - OVF Parsing

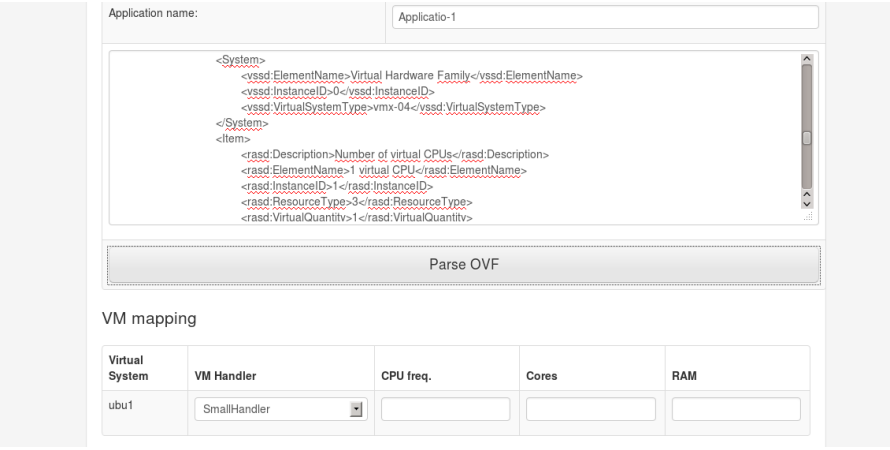

Figure 3.6: Create new CEE Page - Binding VM to VMhandler

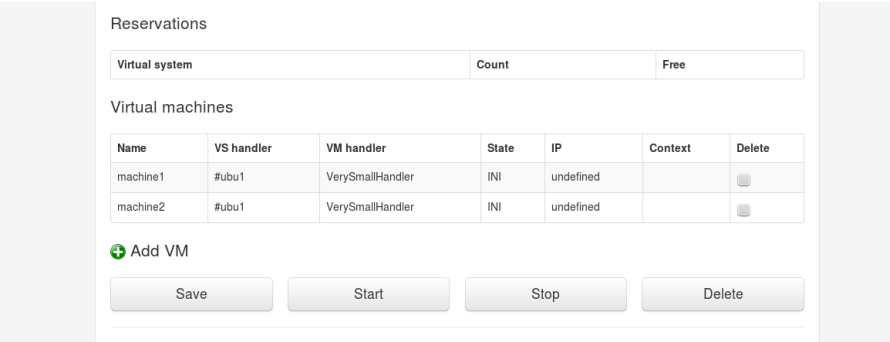

Figure 3.7: Edit CEE Page

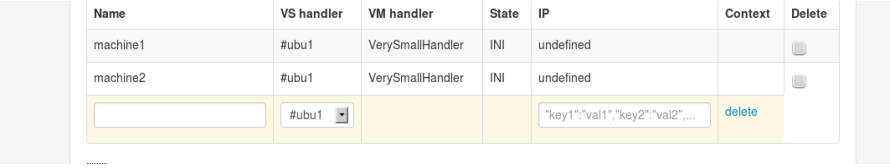

Figure 3.8: Edit CEE Page - adding VMs

3.5 Edit CEE

From page (Fig 3.3), choosing a CEE is possible to make operations on the application related to it. The page (3.7 shows a summury of the CEE and its application. From this page you can start the application, stop the application, and add/delete VMs. As you add or delete one or more VMs remember to click on the button ”SAVE” to send the command to VEP.