Control Development Environment Platform

PDEng Software Technology

CERN

TU/e
Presentation Overview

User perspective presentation 25 min

Technical presentation 60 min

Live demo 10 min

Questions 10 min
Presentation Overview

User perspective presentation

- PDEng
- The project
- Demo
- Conclusion
PDEng (Professional Doctorate in Engineering)

TU/e - Software Technology - OOTI
Masters

Post-Master

PDEng (2-years)

PhD (4-years)

- 4 In-house industrial software projects
- On-site 9-months design project: ST project
- Workshops and training

Project #4: CDE Platform (CERN)
The Trainees

16 Trainees

11 Nationalities

0 – 5 Years experience

Computing background
Team Formation

Ronald
Project Manager

Dima
Architect

Backbone team

Language team

Scalability team
<table>
<thead>
<tr>
<th>Team Formation</th>
<th>Team lead</th>
<th>Team architect</th>
<th>Configuration manager</th>
<th>Quality manager</th>
<th>Test manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backbone team</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Language team</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability team</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A Dream?
A Dream?

• Work from anywhere

• Focus on creating, not on environments/tooling

• Use newest technologies to improve productivity
Previous PDEng project

- Eclipse IDE with language plugin
  - Grammar
  - Editing features
  - For Control scripts

- Further development
  - ~2 years

- <demo>
Challenges

• Work from multiple locations
  – Visiting scientists from > 113 countries

• Running/testing on various environments
  – OS
  – Libraries
  – WinCC-OA
  – Patches

• Setup time new developers
Requirements

• Design framework that:
  – Separates **artifacts** from tools (code/logs/etc.)
  – Allows to access **environment** from the **browser**
  – Runs/tests Control code on various **environments**
Requirements

- Design framework that:
  - Contains a **Control code editor** (based on Eclipse IDE with language plugin)
  - Scales up to ~800 users
Our Solution

- **UI**
- **VM Access**
- **VM**
- **Web IDE**
- **Environment Creator**
- **Artifact Storage**

- **OpenStack**
- **Ansible**
- **KVM**
- **Vagrant**
- **Git**
Demo

Control Development Environment Platform

WinCC OA Orion IDE

Environment Chooser
Observations

• Used technologies **new**, but seem **stable**

• Great **potential** with technologies
  – Scalability
  – Speed
  – Usability
  – Functionality

• Configurations can be **complex**
  – **Manage** them in a smart way
Conclusion

“Work from anywhere”

System accessible from browser:

Control Development Environment Platform

WinCC OA Orion IDE
EnvironmentChooser

Web IDE
Conclusion

“Focus on creating, not on environments/tooling”

Providing environments on demand and Control code can be executed from browser:

Control Development Environment Platform
Conclusion

“Use newest technologies to improve productivity”

Extension of Control editor and all systems behind that possible
Presentation Overview

User perspective presentation  25 min

Technical presentation  60 min

Live demo  10 min

Questions  10 min
Presentation Overview

Technical presentation

- Architecture overview
- Language team
- Backbone team
- Scalability team
Presentation Overview

Architecture overview

- Challenges
- Our solution
- Scalability
- OpenStack
Go into details
Control code editor?

Web IDE

Challenges
Challenges

Runs/tests Control code on various environments?

Environment Creator & Virtualization

VAGRANT
ANSIBLE
KVM
CHEF
python
WinC
SIEMENS
Xen
openstack
EMU
Flask
donker
Challenges

Allows to access **environment** from the **browser**?
Separates **artifacts** from tools (code/logs/etc.)?

**Remote Access to VMs & Artifact Storage**

[TeamViewer](#)  [VNC](#)  [SSH](#)  [Java](#)  [git](#)  [rsync](#)  [xrdp](#)  [Subversion](#)
Our Solution

Web IDE
- ORION
- Xtext

Environment Creator
- VAGRANT
- KVM
- ANSIBLE
- openstack

Virtual Machines

Artifact Storage
- git
Our Solution

- Web IDE Language team
- VM Access Scalability team
- Environment Creator Backbone team
- Virtual Machines Scalability team
- Artifact Storage Scalability team
Scalability
Bottlenecks

Web IDE
- Orion
- Xtext

VM Access
- Ansible
- Git

Environment Creator
- Vagrant
- KVM

Virtual Machines
- OpenStack

Artifact Storage
- Git
Scalable Solution

Web IDE

Environment Creator

Virtual Machines

VM Access

Artifact Storage

X 800

Orion

Xtext

Vagrant

Ansible

OpenStack

KVM

Ansible

OpenStack

Cloud Software
Deployment on OpenStack

- Web IDE
- Environment Creator
- Virtual Machines
- VM Access
- Artifact Storage
- Git
Presentation Overview

Technical presentation

- Architecture overview
- Language team
- Backbone team
- Scalability team
Presentation Overview

Language team

Web IDE

Editing Control Code
Our Solution

- VM Access
- Web IDE
- UI
- VM
- Environment Creator
- Artifact Storage

- OpenStack
- Ansible
- Vagrant
- KVM
- Git
Our Solution

- **Web IDE (Orion)**
- **VM Access**
- **VM**
- **UI**

**Environment Creator**
- openstack
- Ansible
- Vagrant
- KVM

**Artifact Storage**
- git
What is our Web IDE?

**Browser**
- Web IDE
- Editor
- Version Control

**Server**
- Web IDE
- IDE
- APIs
- REST

**Deployment Env.**
- Virtual Machines
Web IDE

Why a web-based IDE?

load

request

Run code
Web IDE

Eclipse WinCC-OA evolution

Eclipse IDE with language plugin

Reuse functionalities
  - Language grammar
  - Static code analysis

Extend functionalities
  - Bookmarks

Web IDE
## Foundation

<table>
<thead>
<tr>
<th>Eclipse Che</th>
<th>Orion</th>
<th>Xtext</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Migrating existing features</td>
<td>- Easy to integrate with Xtext DSL services</td>
<td>- Deploy Xtext grammar language</td>
</tr>
<tr>
<td>- Integrating with GIT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| | | |
| | - High risk of deploying Xtext grammar language | - Low risk of deploying Xtext grammar |
| | - Limited documentation | |

- Migrating existing features
- Integrating with GIT
Web IDE

Support for editing and running Control code

Edit  View  Run
Presentation Overview

Language team

Web IDE

Editing Control Code
Editing Control Code

- Syntax highlighting
- Code auto completion
- Code validation

- Control Language Grammar
- Code auto completion
- Code validation

- Control Language Grammar
- Syntax highlighting

Web IDE

Xtext Server

Orion Server
Syntax Highlighting

```java
void main()
{
    int replyCode;
    string dpeName;

    // dpConnect() our 3 HV channels, each to its own simulator code instance, running in its own thread.
    // When you have completed the basic exercise, you can look at the extension to this exercise
    // for a more intelligent, and more general, piece of code to do this!

    dpeName = "muonHvChannel1.settings.onOff";
    replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName);
    
    dpeName = "muonHvChannel2.settings.onOff";
    replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName);
    if (replyCode != 0)
    {
        DebugTN("hvSimulator.cpp: Error=" + replyCode + " from dpConnect() to dpeName = " + dpeName);
    }

    dpeName = "muonHvChannel3.settings.onOff";
    replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName);
    if (replyCode != 0)
    {
        DebugTN("hvSimulator.cpp: Error=" + replyCode + " from dpConnect() to dpeName = " + dpeName);
    }

    // END main
```
Code Validation

Control code file

Errors and warnings

Web IDE

Xtext Server

```
main()
{
    int replyCode;
    dyn_string exceptionIn

    replyCode = dpConnect()
    if (replyCode != 0)
    { DebugTH("ftcGasSystem
        return;
    }
}
```

dpName = "muonHvChannel1.settings.onOff"
replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName );

```
if (replyCode != 0 )
{ DebugTH("hvSimulator.ctl: Error=" + replyCode + " from dpConnect() to dpeName = " + dpeName );
}
```

dpName = "muonHvChannel2.settings.onOff"
replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName );

```
if (replyCode != 0 )
{ DebugTH("hvSimulator.ctl: Error=" + replyCode + " from dpConnect() to dpeName = " + dpeName );
}
```

dpName = "muonHvChannel3.settings.onOff"
replyCode = dpConnect("CoursePVSSProjectBluePeterComponent_hvSimulatorCB", FALSE, dpeName );

```
if (replyCode != 0 )
{ DebugTH("hvSimulator.ctl: Error=" + replyCode + " from dpConnect() to dpeName = " + dpeName );
}
```

```
} // END main
```
Code Auto Completion

Web IDE

Control code file & Caret offset

Proposals

Xtext Server

Xtext
Re-using Existing Functionalities

Eclipse IDE with language plugin

Web IDE

Static code analysis

Bookmarks
Static Code Analysis

Xtext Server

Invoke

Request execution

Web IDE

Read

Return result

LOG

Eclipse IDE with language plugin

LOG

Invoke

Request execution

Read

Return result

LOG

Eclipse IDE with language plugin
Bookmarks

Web IDE

ORION

Reload

Store

Orion Server

```java
//Demonstration
void main()
{
    string hello = "Hello ";
    string space = " ";
    string world = "World ";
    string explicite = " ! ";
    DebugTN(hello + space + world + explicite);
}
```
Running the Control Code

Web IDE

Request execution

Select an environment

Request execution

VM

LOG

Return result
Presentation Overview

Technical presentation

- Architecture overview
- Language team
- Backbone team
- Scalability team
Presentation Overview

Backbone team

- Environment creation
- Provisioning
Our Solution

- VM Access
- VM
- Web IDE
- UI
- Environment Creator
- Artifact Storage
Requesting a Machine: Overview

• Desktop as a Service

• OS
• Software

VM Request
Process

Interpret request

Install software

Create a virtual machine

Available VM
Hypervisor

VM

APP

OS

VMM

VM

APP

OS

VMM

VM

APP

OS

VMM

Kernel

Scheduler

PHYSICAL HARDWARE

CPU

Storage

Memory

Networking

OpenVZ

Xen

vmware

KVM

Koding

Microsoft Virtual PC

proxmox

EMU

VMware vSphere
### Hypervisor Decision

<table>
<thead>
<tr>
<th></th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OpenStack</strong></td>
<td>- Xen integration&lt;br&gt;- Widely used</td>
<td>- No browser interface</td>
</tr>
<tr>
<td><strong>KVM</strong></td>
<td>- Browser interface&lt;br&gt;- Open source&lt;br&gt;- Good performance&lt;br&gt;- Widely used</td>
<td>- Only Linux-based hosts</td>
</tr>
<tr>
<td><strong>Proxmox</strong></td>
<td>- Using KVM and QEMU&lt;br&gt;- Web browser access</td>
<td>- Not broadly used</td>
</tr>
</tbody>
</table>
Creating and Configuring VMs

Configuring and handling VMs directly with the Hypervisor

Configuration management tool (wrapper).
Creating VMs with Vagrant

1. Request VM creation
2. Request SW installation

Vagrant Boxes

VM with SW

Installation steps
Vagrant Box

- Boxes are the package format for Vagrant environments:
  - OS already installed
  - Vagrant access settings (SSH)
  - Other SW defined by the user
The primary function is to describe the type of machine required.
Vagrant.configure(2) do |config|
  config.vm.box = "slc"

  config.vm.network "forwarded_port", guest: 80, host: 8080
  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.synced_folder "../data", "/vagrant_data"

  config.vm.provider "libvirt" do |v|
    v.memory = 4096
    v.cpus = 2
  end

  config.vm.provision "ansible" do |ansible|
    ansible.playbook = "cern.yml"
  end
end
Creating VMs with Vagrant

1. Request VM creation

KVM

Create VM

Vagrant file

VAGRANT

Vagrant Boxes
Creating VMs with Vagrant

1. Request VM creation

2. Request SW installation

3TU School for Technological Design

STAN ACKERMANS INSTITUTE

KVM

Create VM

Vagrant Boxes

Vagrantfile

VAGRANT

Execute installation

Installation steps

VM with SW

Provisioning tool
Presentation Overview

Backbone team

- Environment creation
- Provisioning
• Automation installation/configuration
## Provisioning

<table>
<thead>
<tr>
<th>CheF</th>
<th>Puppet Labs</th>
<th>Ansible</th>
</tr>
</thead>
<tbody>
<tr>
<td>![CheF Logo]</td>
<td>![Puppet Labs Logo]</td>
<td>![Ansible Logo]</td>
</tr>
</tbody>
</table>
| • Mature solutions  
• All major OS support  
• Popularity among Technical companies | • Complex  
• Slow learning curve | • New (since 2012)  
• No support for Windows |
| • Low overhead *(Playbook-based)*  
• Fast and simple  
• Python-based | | |
| ![Thumb Up] | ![Thumb Down]  
Slowish (not agile)  
Puppet DSL required | |
Playbooks

- Ansible’s configuration, deployment, and orchestration language
- YAML format (more readable & writable)

```yaml
# A list of tasty fruits
fruits:
  - Apple
  - Orange
  - Strawberry
    price: 10
    color: red
```
- name: Install GIT to Ubuntu guest
  hosts: ubuntuNode
  tasks:
  - name: Install Git
    apt: pkg=git state=latest
How a Playbook Works

Management Node

- Hosts Inventory
- Playbook

SSH Trust

VM
VM
VM

Web Interfaces

• Python
  – Official support for Ansible
  – Support for Vagrant

• Flask
  – Lightweight web framework
  – Quick to start
Web Interfaces

HTTP GET
JSON

VM Creator
VM Provisioner
Code Executor
Presentation Overview

Technical presentation

- Architecture overview
- Language team
- Backbone team
- Scalability team
Presentation Overview

Scalability team

Artifact storage

Running Control Code

Accessing Environment
Our Solution

- UI
- VM
- VM Access
- Web IDE
- Environment Creator
- Artifact Storage

- OpenStack
- Ansible
- KVM
- Vagrant
- Git
Running Control Code

- **UI**: Control Development Environment Platforms
- **VM**: Virtual Machine
- **Web IDE**: ORION
- **VM Access**: Request execution
- **Return result**: Send code
- **Artifact Storage**: git, Ansible, Vagrant, KVM
- **Environment Creator**: OpenStack
Artifact Storage

UI

Environment Creator

VM Access

VM

Web IDE

Send code

Request execution

Return result
Artifact Storage: User Data Hub

- Web IDE
- VM
- Local access
- git

Todo: link with others
Artifact Storage: Versioned?
Running Control Code

- **VM Access**
  - Request execution
  - Return result

- **UI**

- **Web IDE**

- **Environment Creator**
  - OpenStack
  - Ansible
  - Vagrant
  - KVM

- **Artifact Storage**
  - Git
Running Control Code: Getting the Code?

```java
main()
{
    int replyCode;
    dyn_string exceptionIn
    replyCode = dpConnect(
    if ( replyCode != 0 )
    { DebugTN("fcGasSystem
        return;
    }
}
// .................
```
Running Control Code

1. Request execution from Web IDE (ORION).
2. Send code to VM.
3. VM executes the code.
4. Return result back to Web IDE.
5. Log is generated.
6. Artifact Storage (git) is updated with the code.
Running Control Code: Run Command
Running Control Code

1. Request execution
2. Send code
3. VM
   - Return result
4. Request execution
5. Send code
6. Artifact Storage
7. Web IDE
   - Send code
8. LOG
9. Artifact Storage
10. git
Running Control Code: Getting the Result?
Running Control Code

1. Request execution
2. Send code
3. VM
4. Request execution
5. Return result
6. Send code
7. Web IDE
8. Artifact Storage
Running Control Code

- Request execution
- Web IDE: ORION
- Push code: git
- Return JSON
- Execute as CTRL node
- VM
- Pull code: git
- Artifact Storage
- Request execution
- http://
- SSH
- python
- http://
Scalability of Running Control Code

X 800

Or... any other form of load balancer or central access
Running Control Code

1. Request execution
2. Send code to VM
3. VM Access
4. VM
5. Return result
6. Send code to UI
7. UI
8. Environment Creator
9. Artifact Storage

Tools:
- Web IDE: Orion
- VM Access
- VM
- UI
- Environment Creator: Ansible, OpenStack
- Artifact Storage: Git
- Control Development Environment Platforms: Vagrant, KVM
Our Solution

- Web IDE
- VM Access
- UI
- Environment Creator
- Artifacts Storage
  - openstack
  - ANSIBLE
  - VAGRANT
  - KVM
  - git
VM Access
How to Access the Remote Desktop?

- **Non-browser solutions**
  - **Existing tools**
    - **Known to be performant**
    - **Installation necessary**
  - **Manually keep track of VM's**

- **Qemudo**
  - **No installation on client**
  - **Poorly documented**
  - **Configuration needed**

- **Guacamole**
  - **No installation on client**
  - **Well documented**
  - **Configuration needed**
Access Environment

Request environment -> Create

Create -> Update info

Update info -> http://

http:// -> VM Access

VM Access -> Guacamole

Guacamole -> Access

Access -> http://

http:// -> VM Access

VM Access -> Access
Scalability of Environment Access

X 800 → Load balancer → X (?) http://
VM Access
Presentation Overview

**User perspective presentation**
- 25 min

**Technical presentation**
- 60 min

**Live demo**
- 10 min

**Questions**
- 10 min
Live Demo

Control Development Environment Platform

WinCC OA Orion IDE

Environment Chooser
Recommendations

• Use OpenStack in combination with our system
  – Some part might be re-useable

• Investigate performance concerns
  – Might make the system more usable

• Used technologies seem worth exploring further
Conclusion

• The dream is real:
  – “Work from anywhere”
  – “Focus on creating, not on environments/tooling”
  – “Use newest technologies to improve productivity”

• Migrating current Eclipse IDE features to Web IDE takes effort, but can be done

• Technologies seem usable
Thank You
Presentation Overview

User perspective presentation
25 min

Technical presentation
60 min

Live demo
10 min

Questions
10 min