A validation system for data analysis in HEP using virtualization

- motivation
- concepts and design
- walk through the implementation
- summary and outlook

Yves Kemp (DESY IT), Marco Strutz (HTW Berlin)

Fifth Workshop on Data Preservation and Long Term Analysis in HEP
Fermilab, 05/16/2011

HELMHOLTZ ASSOCIATION

DPHEP Study Group for Data Preservation and Long Term Analysis in High Energy Physics

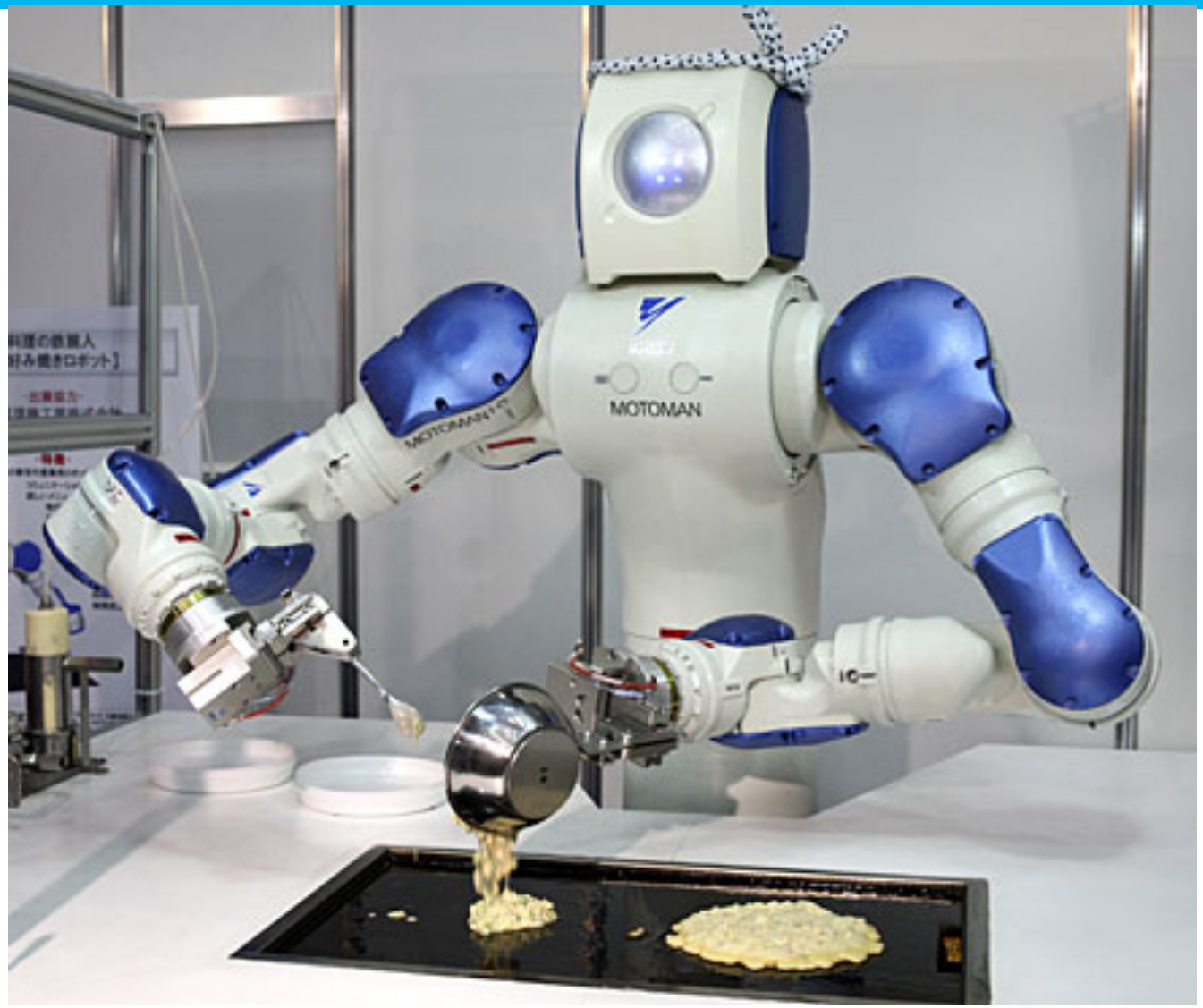
DESY

… but first some thoughts about “Pizza Preservation”

How to preserve a pizza?

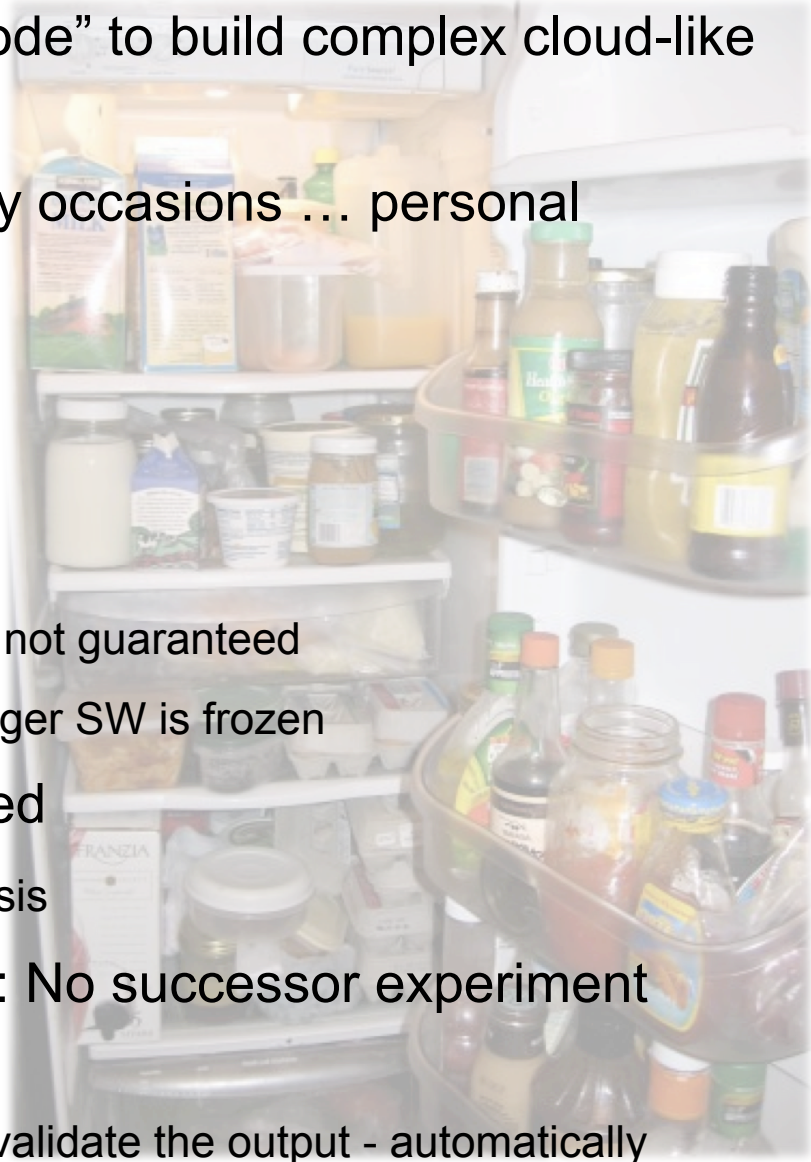
- Couple of days
  - Fridge
- Couple of month
  - Deep freezer
- Couple of years???
  - Preserve the recipe
  - Practice it often: You will not forget the recipe and you can detect variations in external dependencies

… what we have developed:

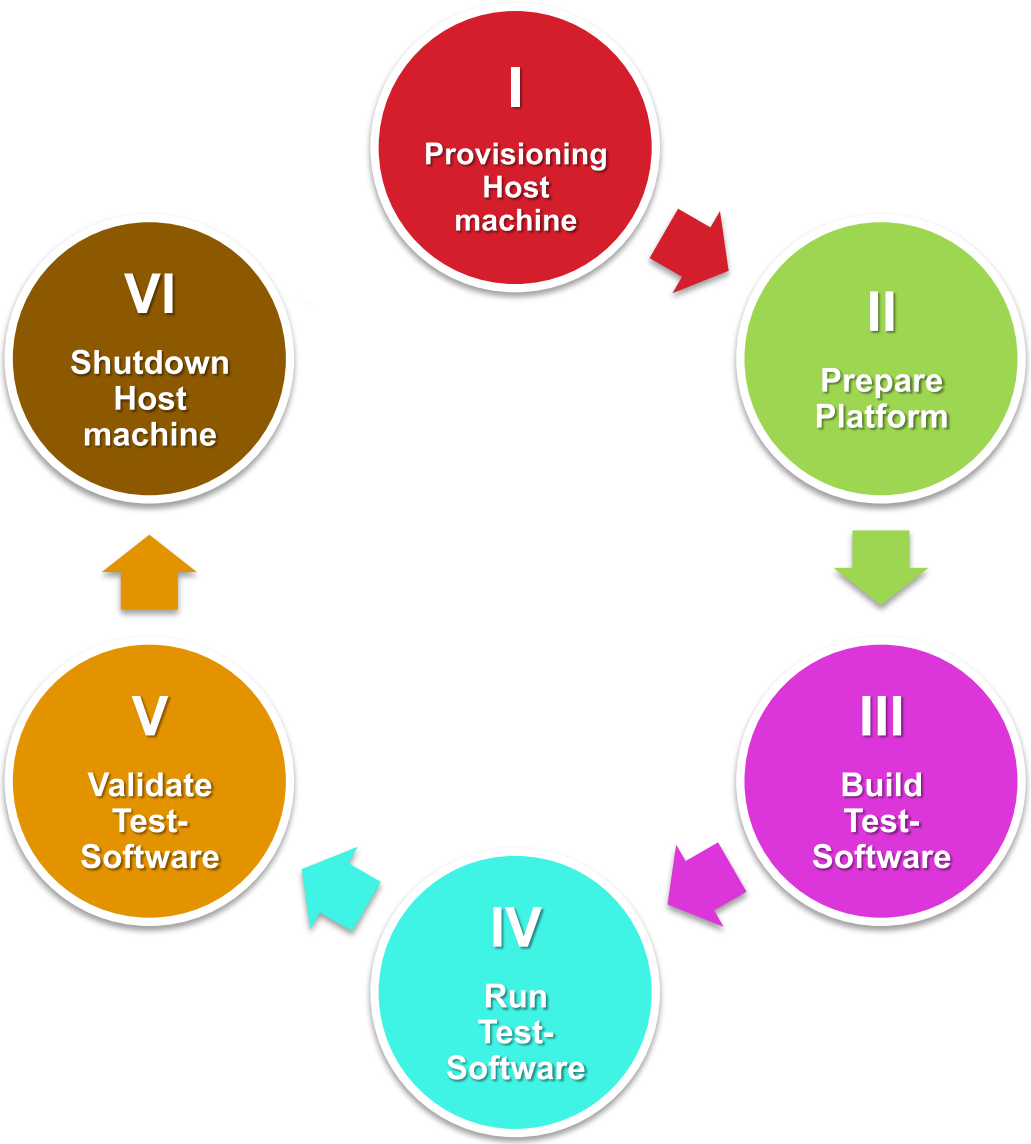


Putting software in the fridge or in the deep freezer

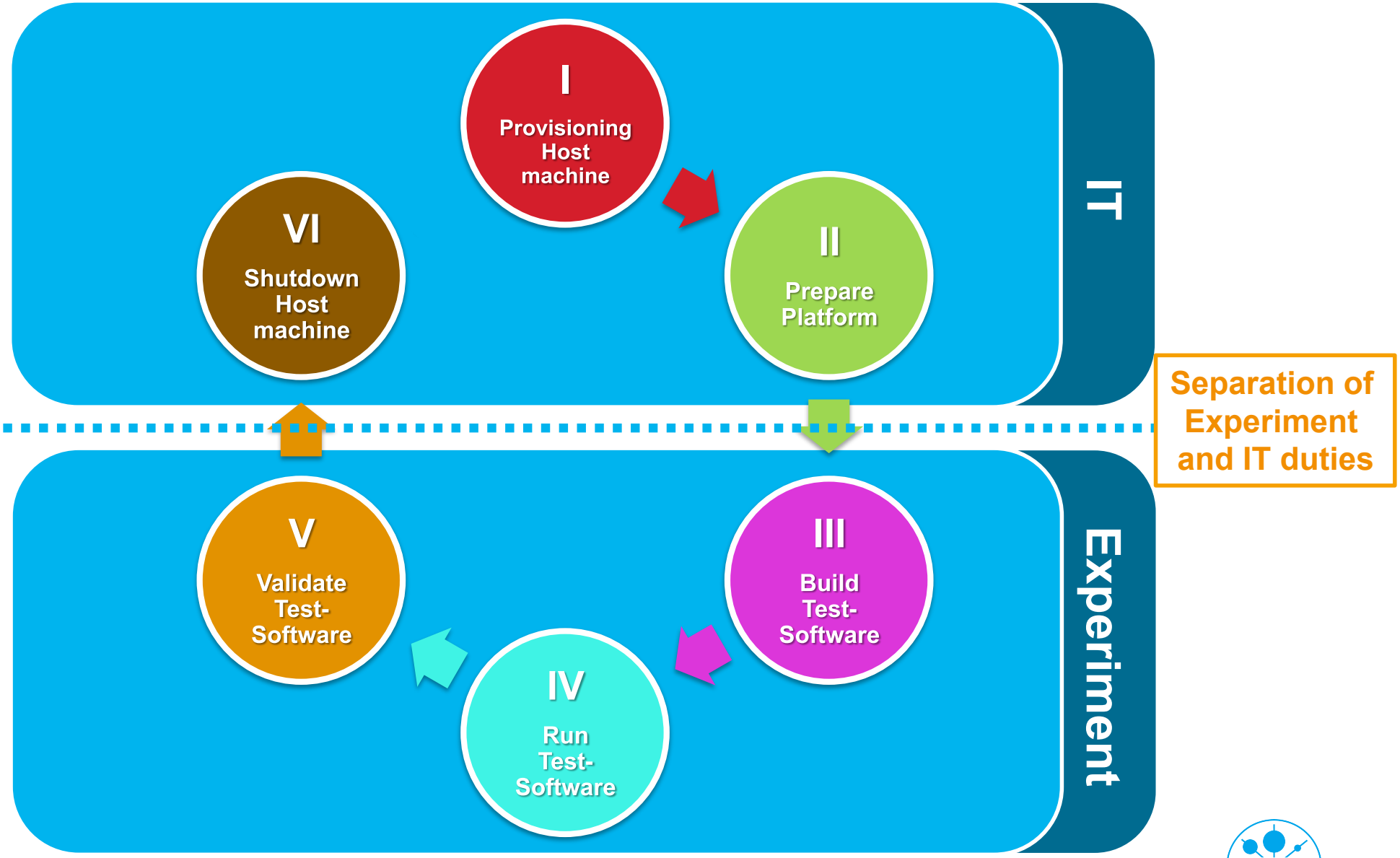
- How? Ranges from just “saving the source code” to build complex cloud-like virtualization production frameworks
- Pro’s and con’s have been discussed at many occasions … personal summary
- Pro’s:
  - Easy to do (manpower), easy to do (time)
- Con’s:
  - Runability of the software and correctness of results not guaranteed
  - Changes if needed will become more difficult the longer SW is frozen
- Freezing SW OK if timeline and scope reduced
  - E.g. makes perfectly sense for BaBar SW and analysis
- … but this is probably not the case for HERA: No successor experiment foreseen
  - So, cook the same recipe ever and ever again, and validate the output - automatically



The Generic Recipe (Atomic Test Life-Cycle)



… and the two cooks



… and then the automation

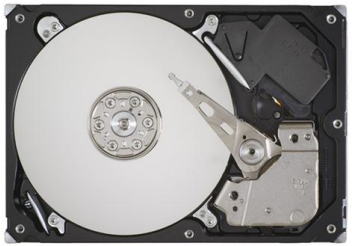
- For each configuration: Run this test cycle often
- You will soon detect when things break e.g.
  - A needed library is no longer available in the distro
  - SW does not compile anymore because of some update
  - SW does not run: Internal error e.g. some API changed
  - SW does not run: External error e.g. Access to mass storage changed
  - SW validation fails: Internal error e.g. compiler optimization behaves different
  - SW validation fails: External error e.g. new chip generation computes different
- You can run daily tests by hand … but easier to use virtualization

The coffee-mill idea

H1 Software
Zeus Software
$EXP Software

ROOT, GEANT,…
External SW

SL5/SL6/Debian/…
IT provides VMs



Test OK

OS lib missing
➔ IT

Tracking code error
➔ EXP-SW

Data unreadable
➔ IT & EXP-SW

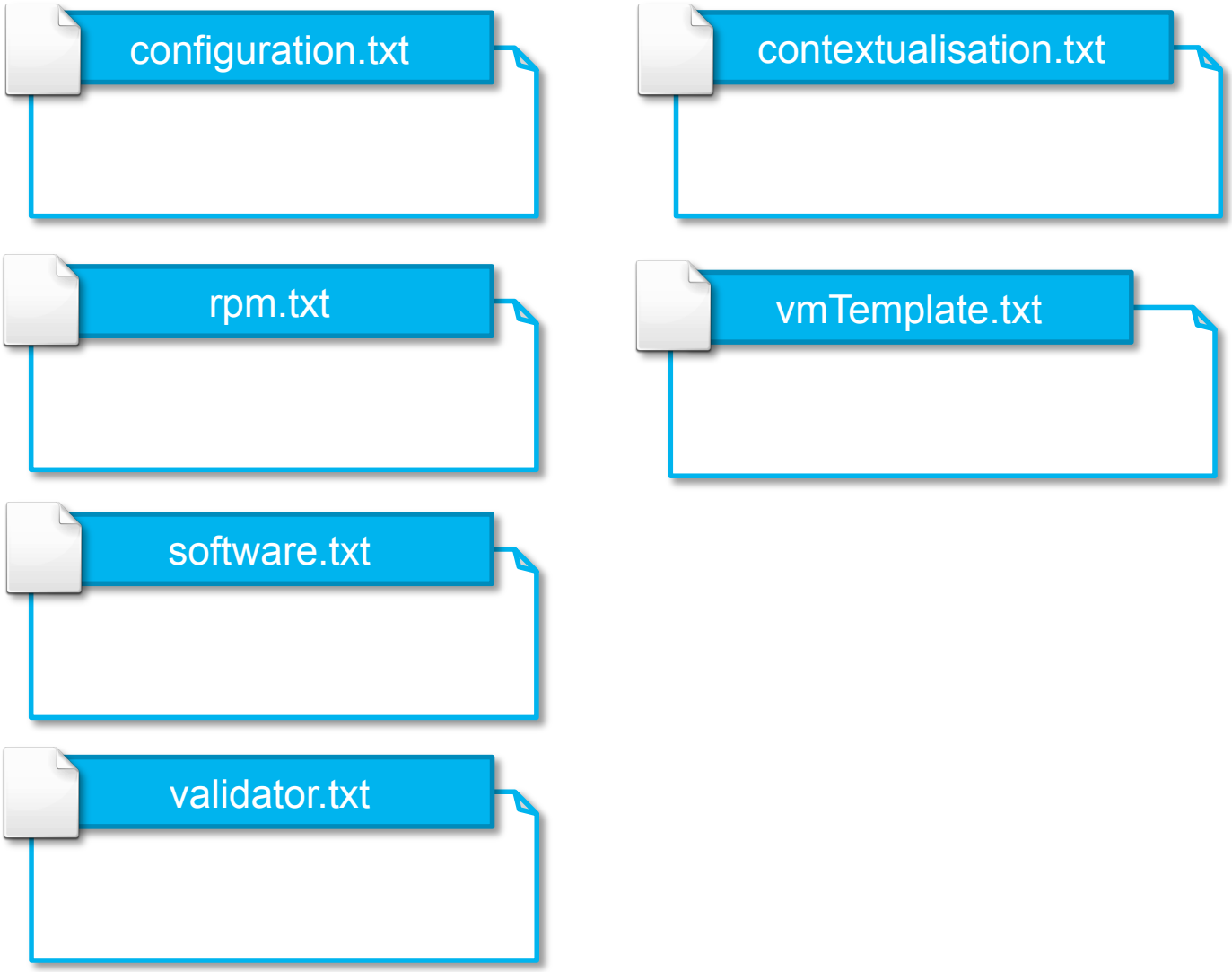
… not just an idea: Marco Strutz (HTW Berlin) implemented a prototype during his master thesis
The prototype is there, and being tested since two weeks by H1, HERMES and ZEUS

… and a walk through the system developed at DESY

- The code:
  - E.g. hello_world.C
- A build.sh script
  - E.g. ./configure && make && make install
- A run.sh script
  - E.g. hello.exe > hello.out 2> hello.err
- A validation.sh script
  - E.g. md5sum hello.out hello.err
- Additional packages in the VM image
  - E.g. gcc in version 4.N.N
- Information about the desired VM image
  - E.g. SL5.N 64bit

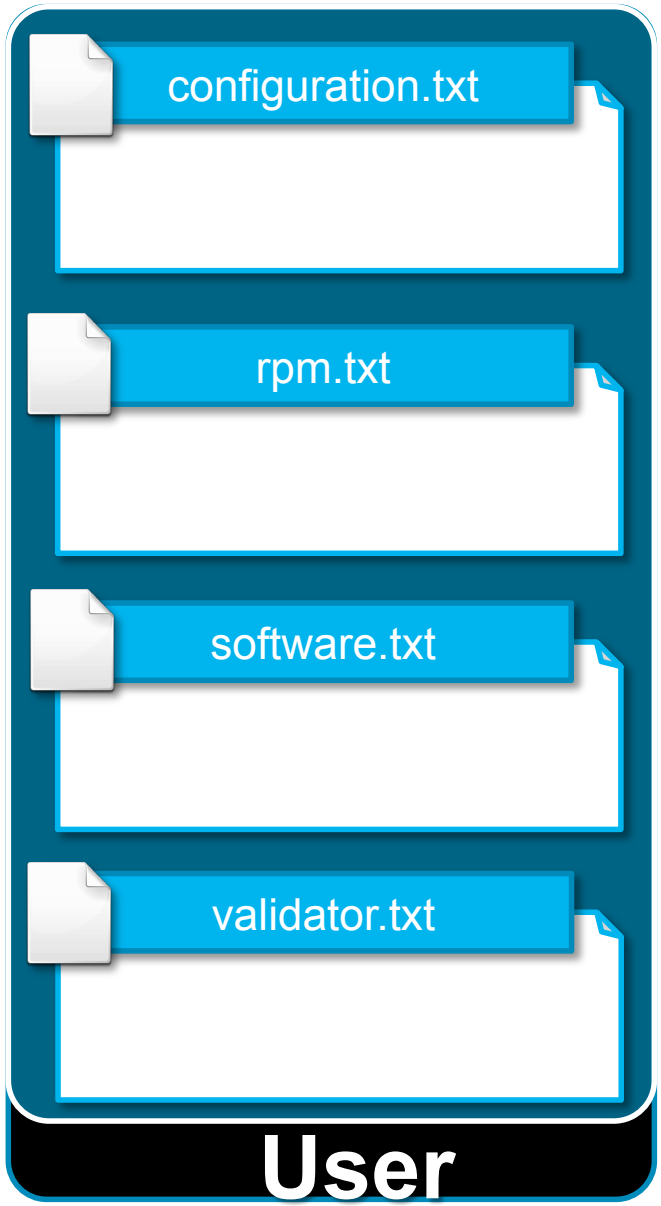
Configuration Example for ROOT

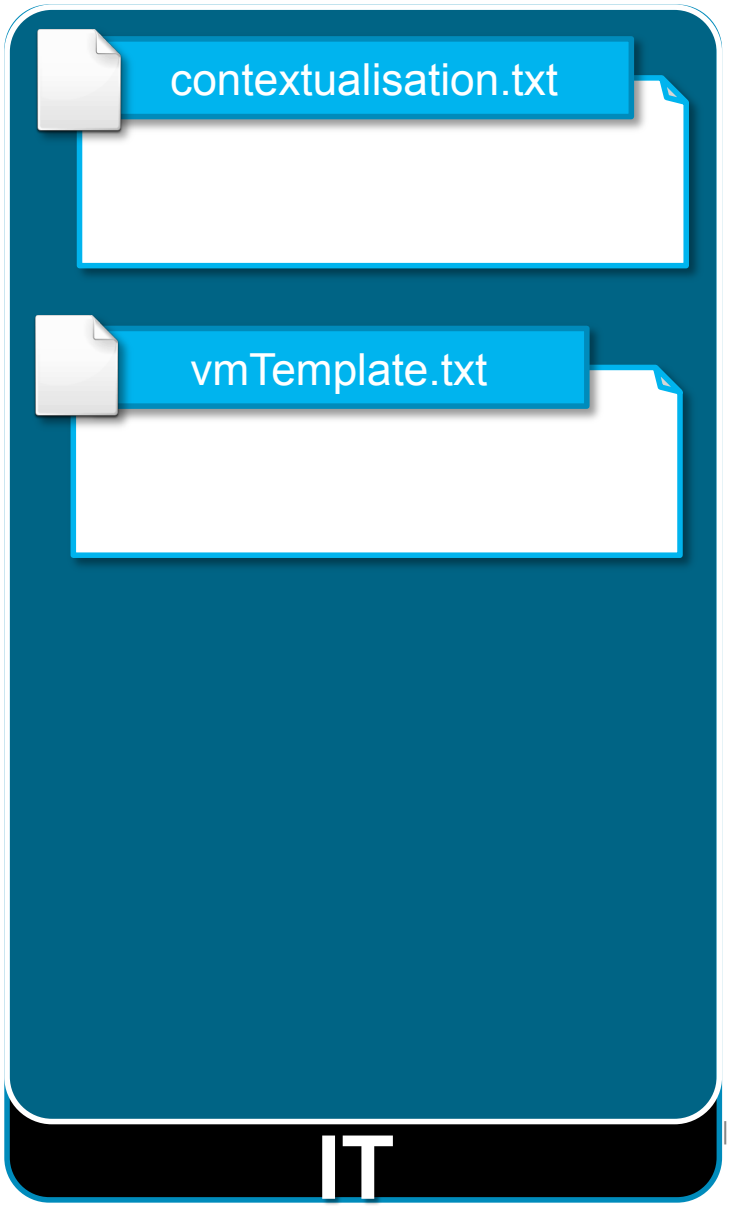
Build Configuration Files



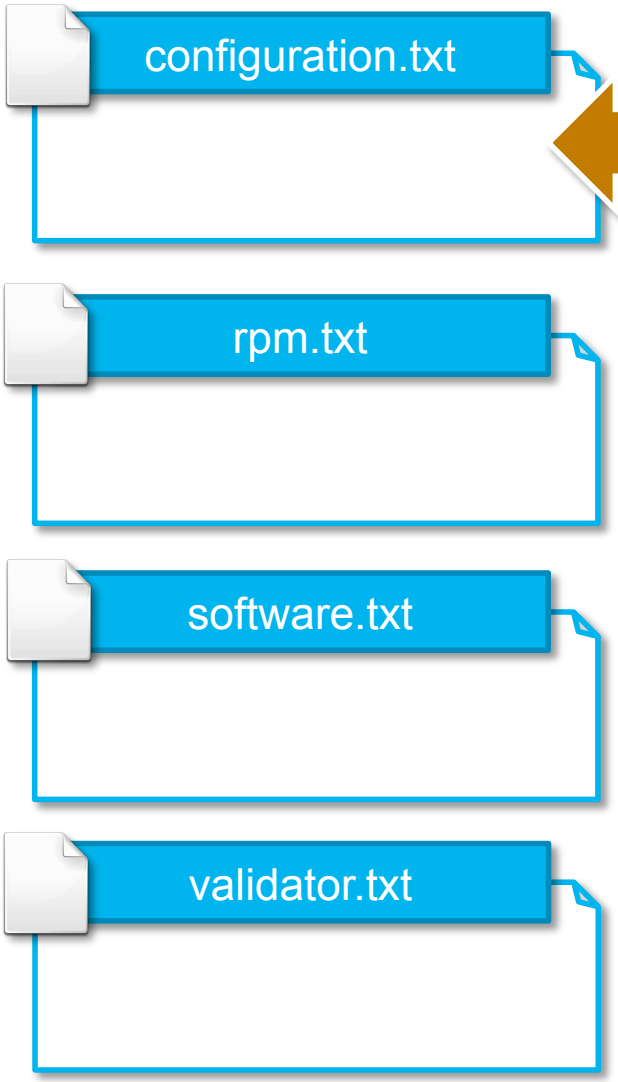
Configuration Example for ROOT

Build Configuration Files



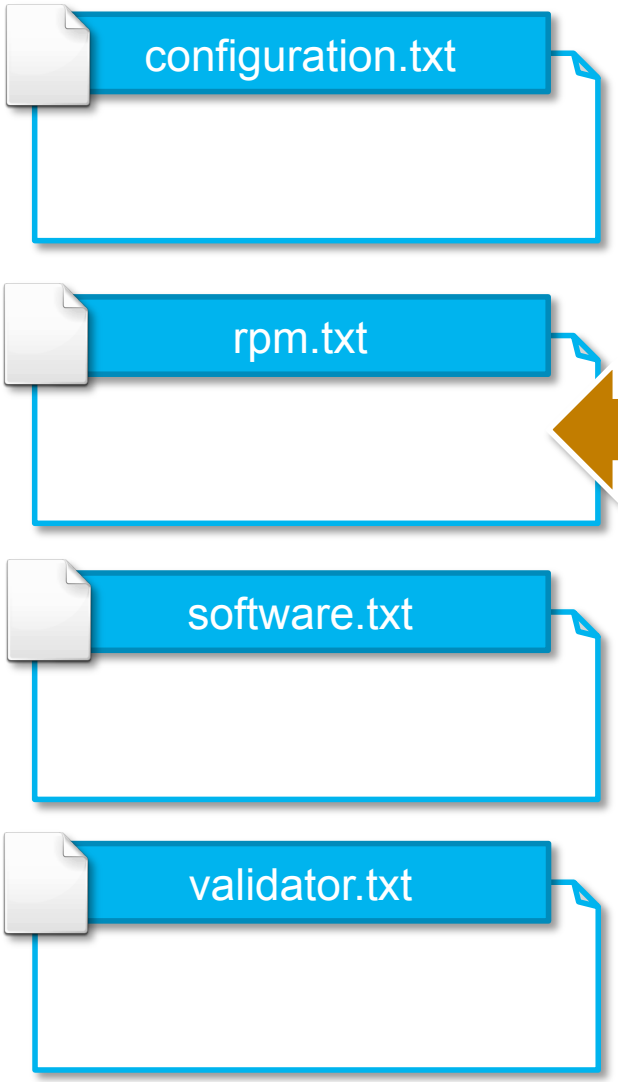


Configuration Example for ROOT
Configuration-Files Content



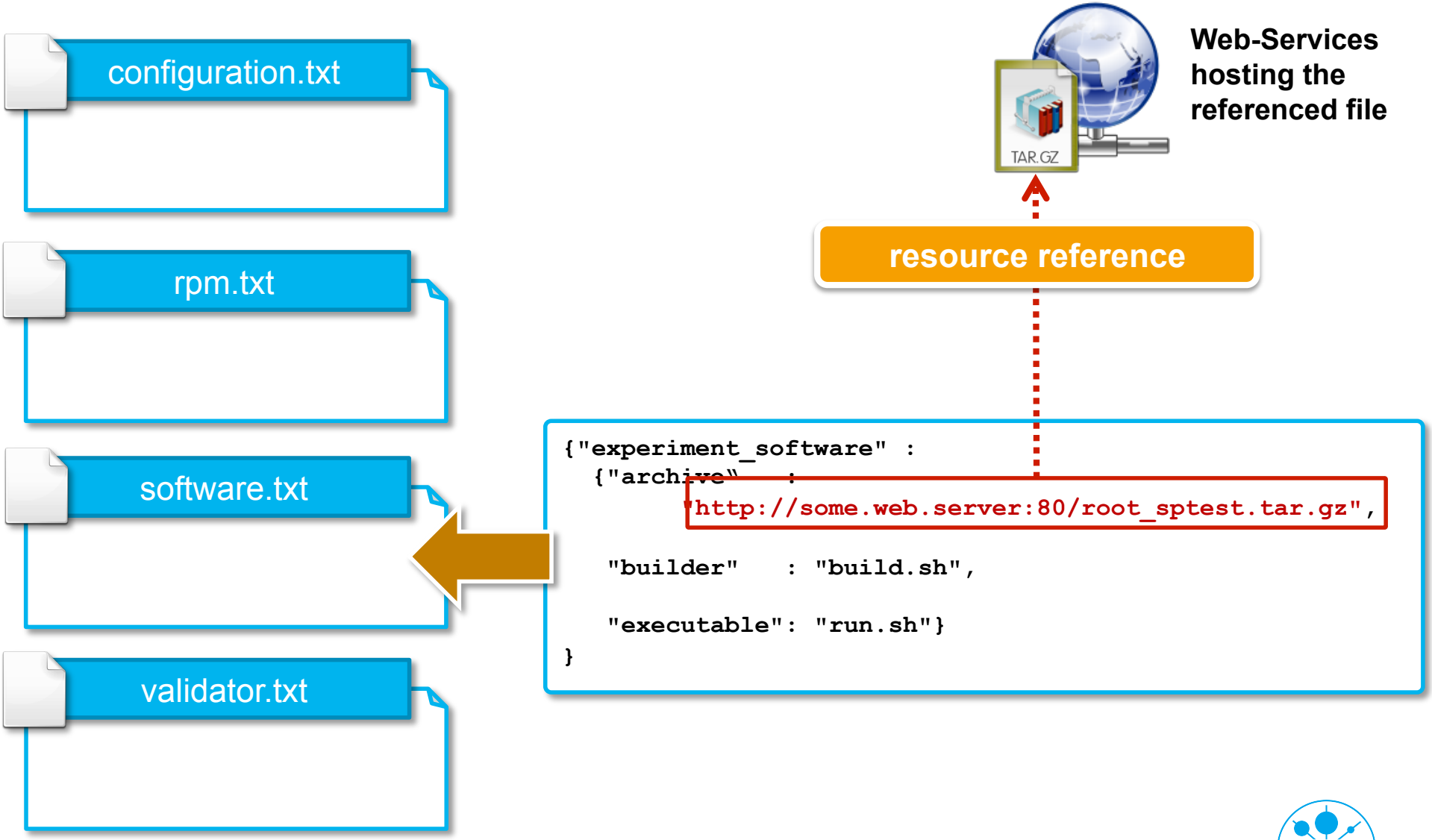

```
{"testcollection":
   {"name"        : "ROOT compiling test",
    "description": "ROOT compiling test",
    "owner"       :
       {"name"   : "marco",
        "email"  : "marco@localhost.com"}
   }
}
```

Configuration Example for ROOT
Configuration-Files Content



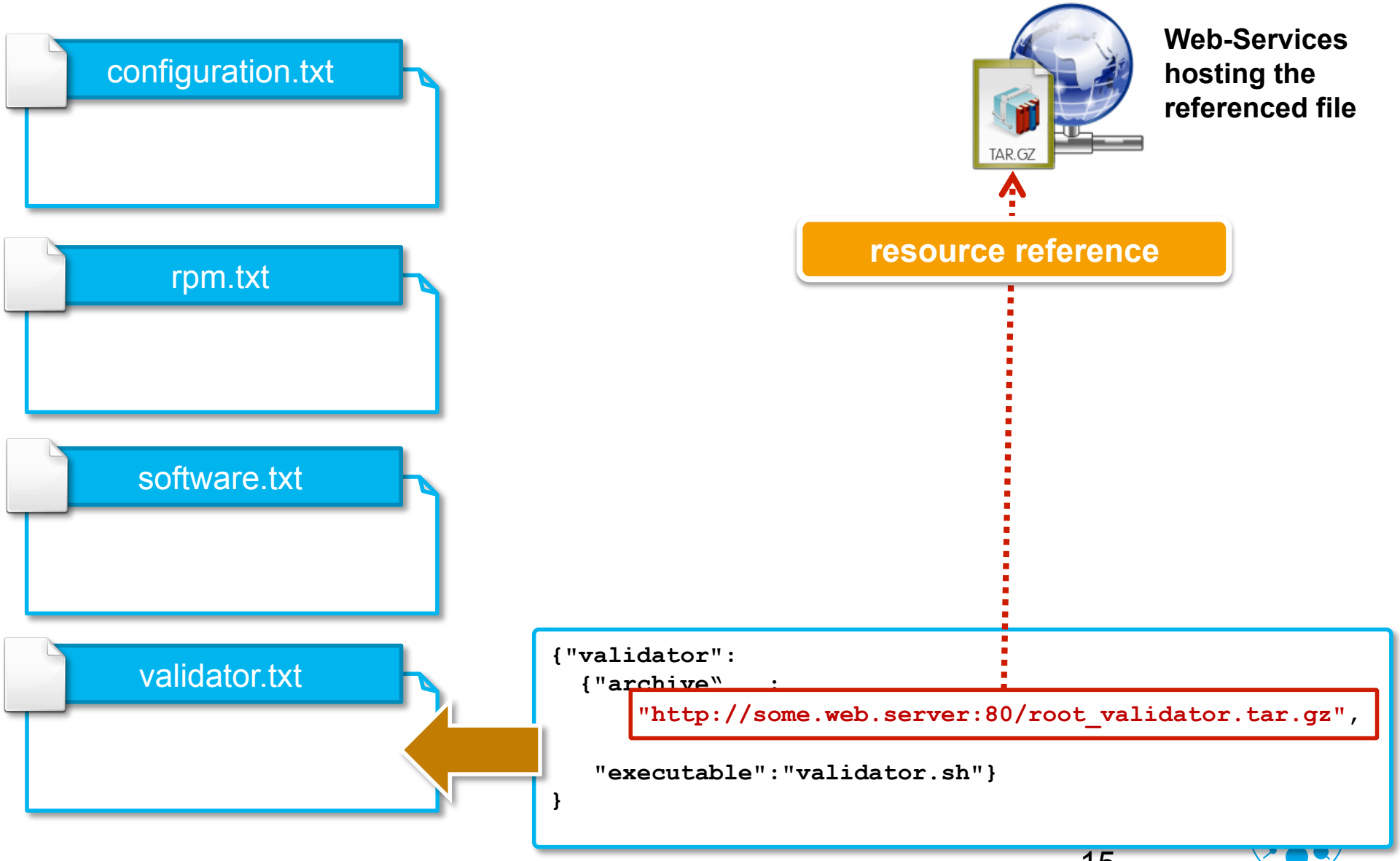

```
{"packages":
  {"gcc-c++" :
     {"version" : "4.1.2",
      "arch"    : "x86_64",
      "summary" : "C++ support for GCC"},
   "libX11-devel" :
     {"version" : "1.0.3",
      "arch"    : "x86_64",
      "summary" : "X.Org X11 libX11 development
                  package"},
   "libXft-devel" :
     {"version" : "2.1.10",
      "arch"    : "x86_64",
      "summary" : "X.Org X11 libXft development
                   package"},
   "libXpm-devel" :
     {"version" : "3.5.5",
      "arch"    : "x86_64",
      "summary" : "X.Org X11 libXpm development
                   package"},
   "libXext-devel" :
     {"version" : "1.0.1",
      "arch"    : "x86_64",
      "summary" : "X.Org X11 libXext development
                   package"}
   }
}
```

Configuration Example for ROOT
Configuration-Files Content



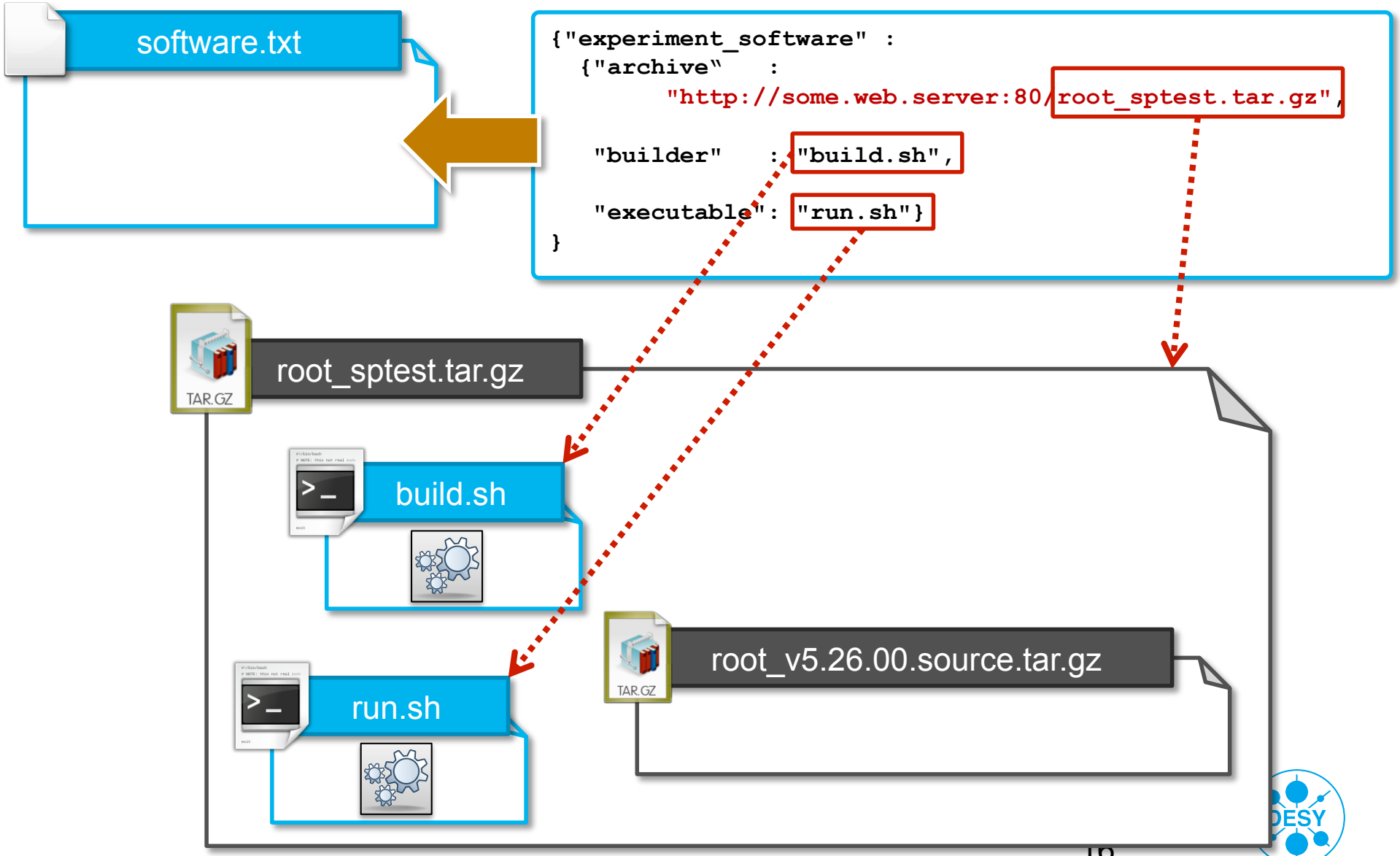
Configuration Example for ROOT

Configuration-Files Content

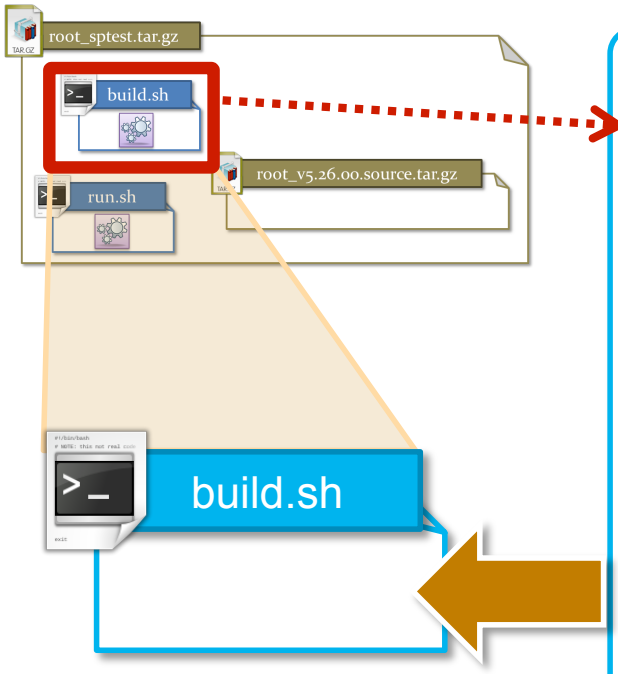


Configuration Example for ROOT

Test-Logic Reference

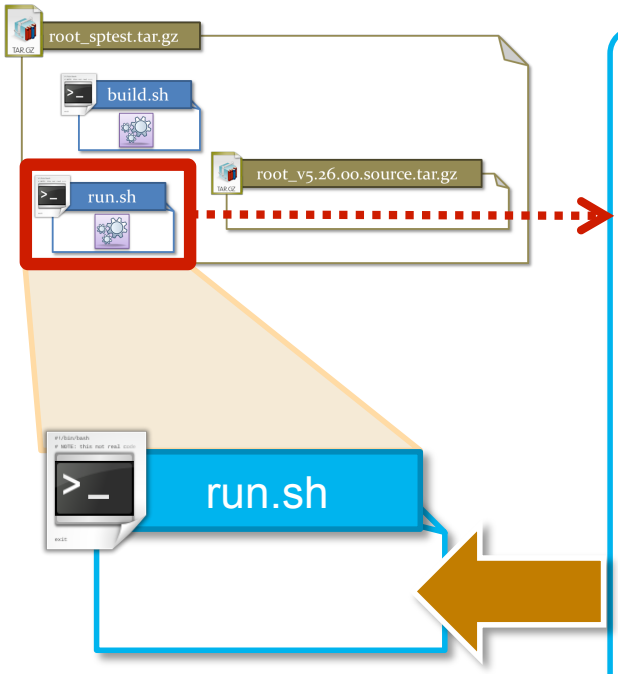


Configuration Example for ROOT
Script Payload




```
#!/bin/sh
PACKAGE="root_v5.26.00.source.tar.gz"
TARGET=./rootSrc

mkdir $TARGET
cd $TARGET

echo "extracting '$PACKAGE'..."
tar xvzf ../$PACKAGE"

#set env
export ROOTSYS=$(pwd)/root
ROOTSYS=$(pwd)/root

#configure ROOT
cd $ROOTSYS
./configure linuxx8664gcc

#make ROOT
make
```

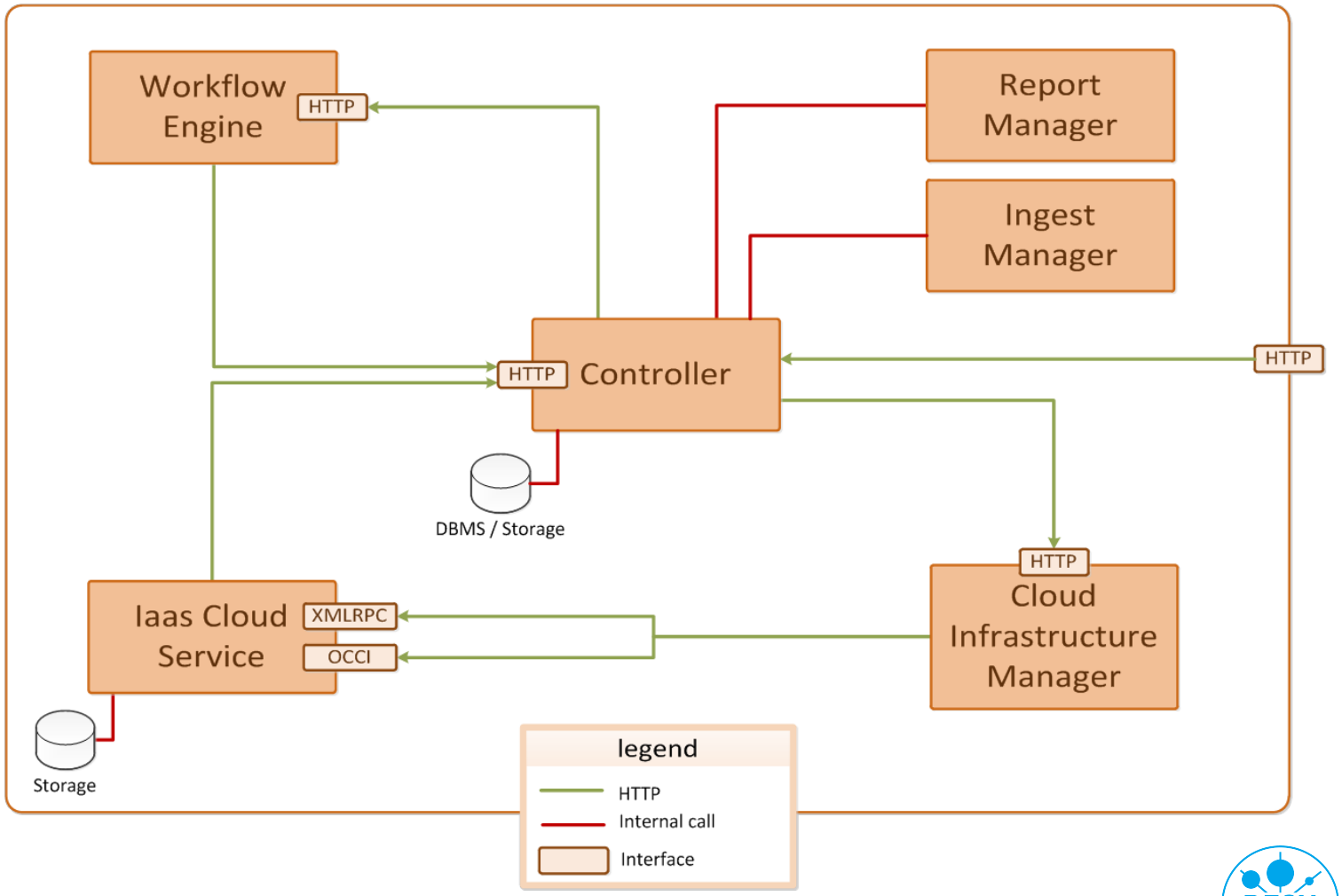
Configuration Example for ROOT
Script Payload



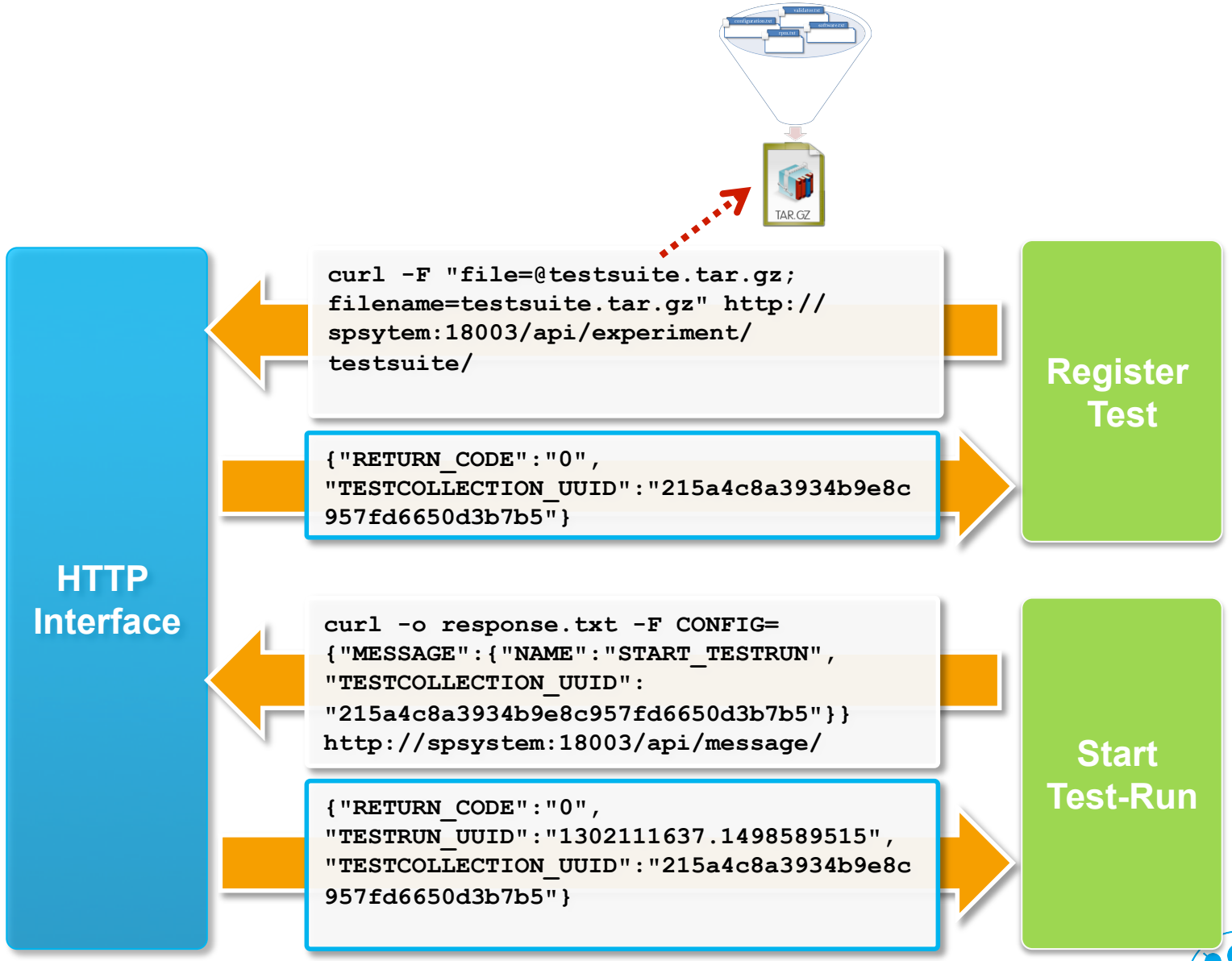

```
#!/bin/sh

export ROOTSYS=$(pwd)/rootSrc/root
export PATH=$ROOTSYS/bin:$PATH
export LD_LIBRARY_PATH=$ROOTSYS/lib:
$LD_LIBRARY_PATH

cd ${ROOTSYS}/test/

#part of the run-step can also be a build-
call
make

echo "running 'stressHepix' test..."
./stressHepix

echo "running 'bench' test..."
./bench
```

A Parenthesis: Components and Communication

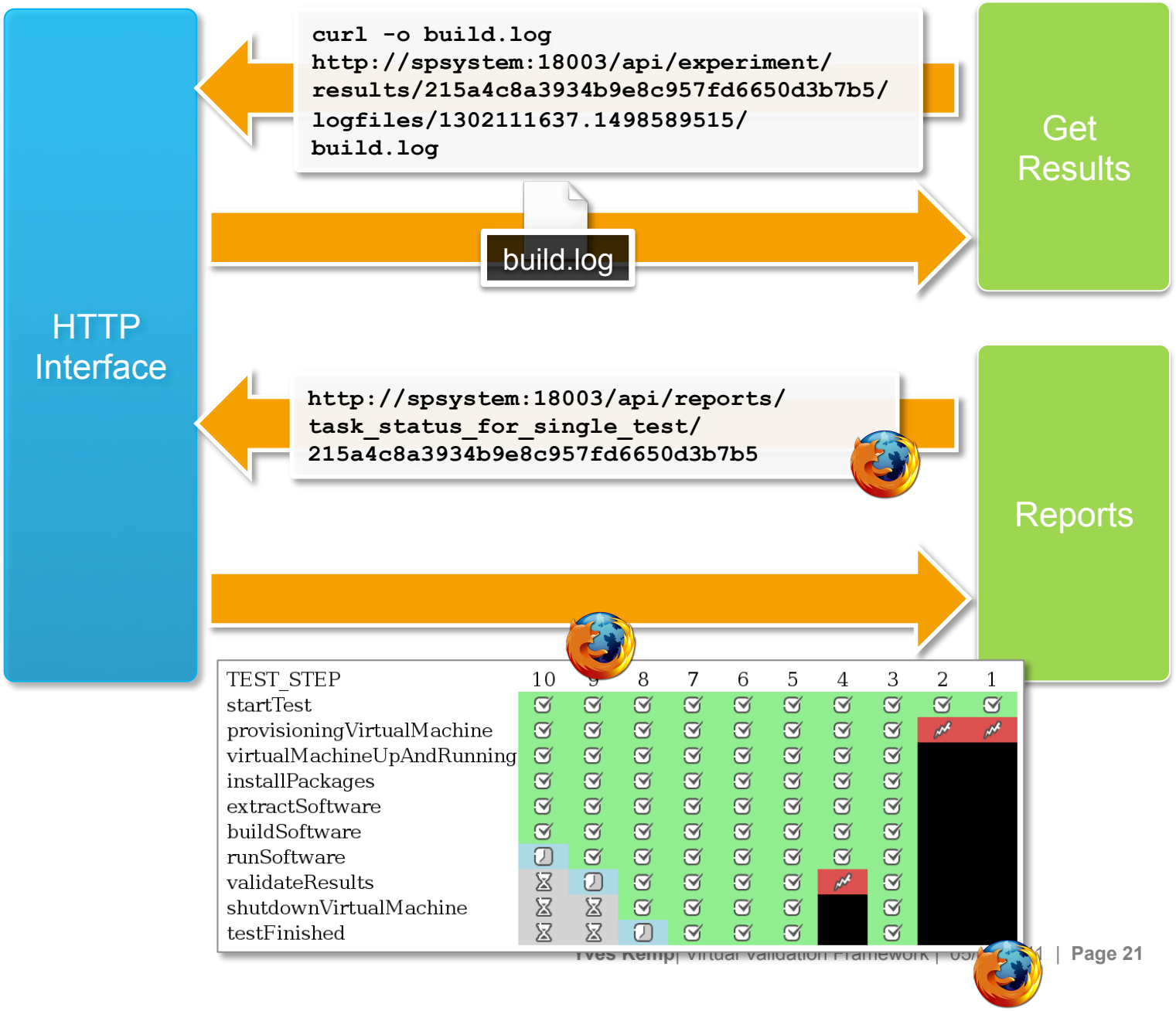
- Modular design of the Validation Framework
- Communication based mostly on HTTP



Launch a test in the Validation Framework



Get results from the test run



Get results from the test run

```
curl -o build.log
http://spsystem:18003/api/experiment/
```

```
+ wget -q http://www.desy.de/~johndoe/virt/sw-test.tgz
+ tar xvzf sw-test.tgz
tar: sw-test.tgz: Cannot open: No such file or directory
tar: Error is not recoverable: exiting now
tar: Child returned status 2
tar: Exiting with failure status due to previous errors
+ chmod 755 swmc_run
chmod: cannot access `swmc_run': No such file or directory
+ ./swmc_run config1234 seed 9876
/home/dphep/application.sh: line 9: ./swmc_run: No such file or directory
+ ls -ltra
total 32
-rw-r--r--. 1 dphep dphep  124 Mar 31  2010 .bashrc
-rw-r--r--. 1 dphep dphep  176 Mar 31  2010 .bash_profile
-rw-r--r--. 1 dphep dphep   18 Mar 31  2010 .bash_logout
drwxr-xr-x. 2 dphep dphep 4096 Mar 31  2010 .gnome2
drwxr-xr-x. 4 dphep dphep 4096 May 13  2010 .mozilla
drwxr-xr-x. 3 root  root  4096 Jun  7  2010 ..
drwx------. 4 dphep dphep 4096 Mar 19 19:04 .
-rwxr-xr-x. 1 dphep dphep  167 Mar 19 19:04 application.sh
```

“Stress tests”

- “Stress Test” new buzz word in Germany: For banks, nuclear power plant, Stuttgart railway station, …
- Also “stress test” your experiment software!
- What to test? We (IT) do not know – but we know some things that failed in the past, and which one should write tests for:
  - Is the access to the mass storage working? E.g. dCap library, door name and port, …
  - Are external services still running and working? (Databases, CVS,…)
  - If you compile SW: Does it compile at all with the current update of your compiler?
  - If you compile SW: Is the compiler optimization doing the same things that before?
  - Do OS tools behave the same way?
  - If there is a change in underlying HW architecture: Are computation results the same?
  - …
- Basically, these are just tests that already exist for e.g. validating nightly builds or validating a Grid SW installation or …

DESY

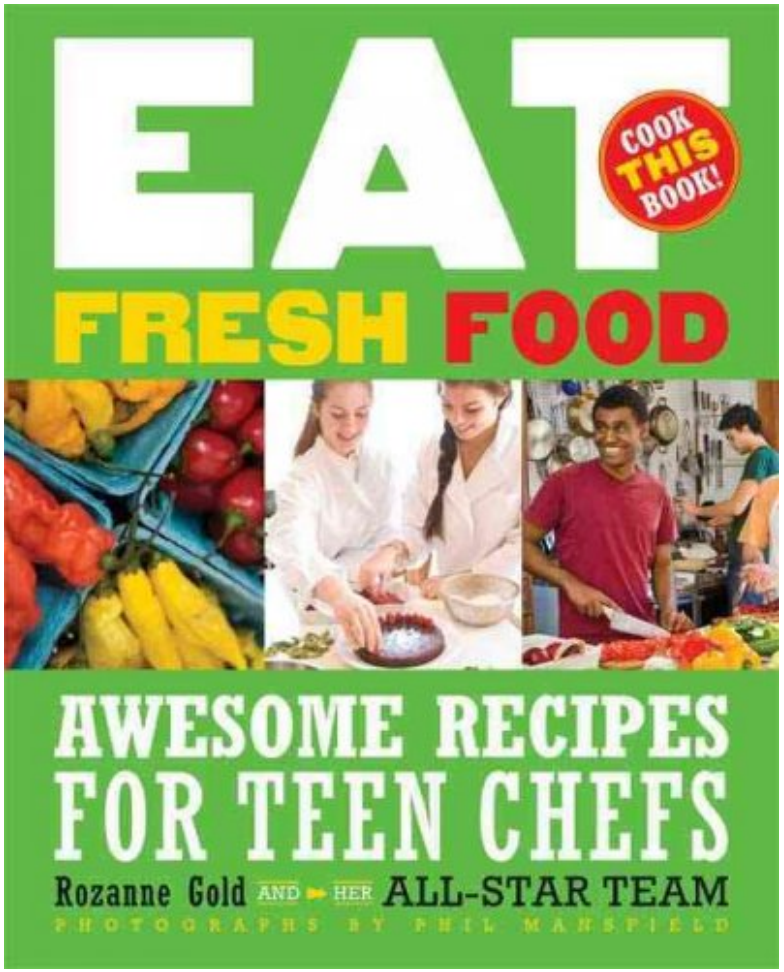
Things we cannot do in the validation framework

- The framework is designed for software verification, validation and migration support only.
- It is not designed for mass production or large scale analysis
  - Both in HW resources and in the interface

- The framework tells you whether you ***could run production today*** – and it can tell you how to prepare your system ("the pizza recipe")
- If you ***want to run production today***, use something like the BaBar system (or any other infrastructure that fits – Grid, EC2, …)

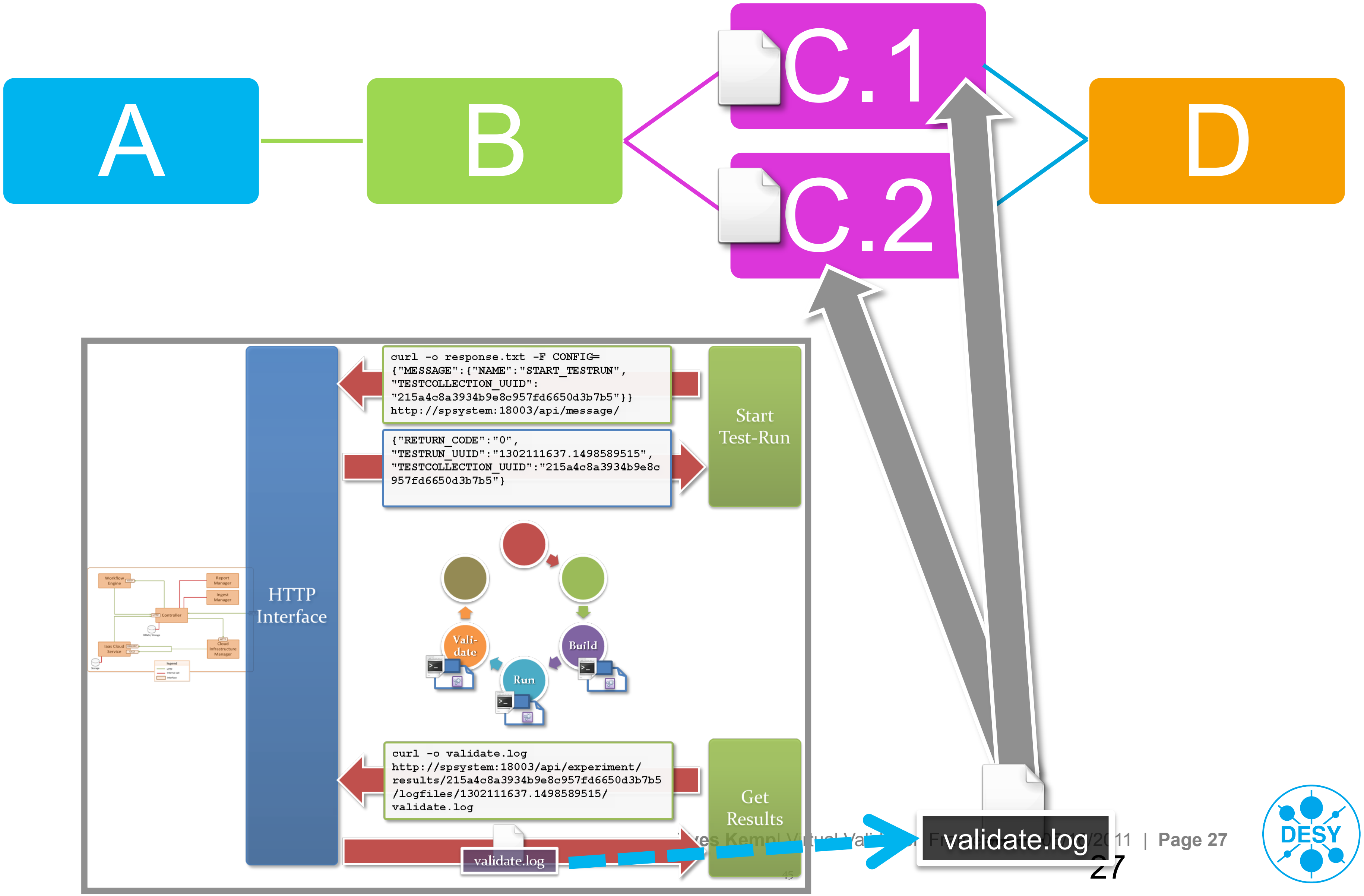
Conclusion and outlook

- DESY IT developed a prototype for automated validation of software in DPHEP context
  - Master thesis of Marco Strutz
- Just a prototype, but already being tested by HERA experts
  - Already some new features/changes identified
  - We will consolidate HW in the next weeks, then tests from non-HERA enthusiast welcome
  - Having a “stress test” in June/July, discuss about production phase in August.
- Validation means running tests
  - These must be provided by experiments … the more the better!



Backup material – technical details

Integration into complex or external workflows possible



Software Components

- Controller, [Cloud Infrastructure | Ingest | Report] Manager
  - Logic : Python v2.4.6 (compatible to v2.6.5)
    http://www.python.org/download/releases/2.4.6/
  - Database : SQLite v2.8.17
    http://www.sqlite.org/
  - Web-Services : web.py 0.34 (Python Framework)
    http://webpy.org/
- Workflow Engine
  - Hudson CI v1.386 (2010/11/19)
    http://hudson-ci.org/changelog.html
- IaaS Cloud Service
  - OpenNebula v2.0.1 (Tue Dec 21, 21:23:35, 2010 +0100) (mostly written in Ruby)
    http://opennebula.org/
  - Hypervisor: KVM (qemu-kvm-0.12.3)

Hardware Components

- OpenNebula [1xFront-End | 2xCluster Nodes]
  - Dell PowerEdge 1950 (Rack Mount Chassis)
  - two Intel(R) Xeon(R) CPU 5160, each 2 cores @ 3.00GHz, 64 bit, VT-x
  - Broadcom Corporation NetXtreme II BCM5708 Gigabit Ethernet
  - 8 GB system memory, FB-DIMM DDR2 FB-DIMM Synchronous 667 MHz
  - 80 GB WDC WD800JD-75MS, (Front-End only: 500GB SG ST3500630NS)
- Controller packages
  - One machine similar as the previous one
- Future (in procurement)
  - 3 machines for controller packages (in virtual machines)
  - Cloud engine: 1x frontend, 2x cluster nodes (but current hardware, have one AMD machine)

Communication Protocols

- Validation Framework Interface
  - JSON
    http://www.json.org/
  - RESTful WebService
  - Linux Shell (/bin/sh)
- Cloud Infrastructure Manager → OpenNebula
  - OCCi (Open Cloud Computing Interface)
    http://occi-wg.org/
  - XMLRPC
    http://www.xmlrpc.com/
- OpenNebula
  - Control Plane: SSH + libvirt (http://libvirt.org/)
  - VM Image Access: NFS