# Geospatial Virtual Appliances Using Open Source Software

Christian Schwartze, Sven Kralisch, Wolfgang-Albert Flügel

# Geospatial virtual appliances using open source software

Christian Schwartze, Sven Kralisch, and Wolfgang-Albert Flügel

Department of Geoinformatics, Hydrology and Modelling, School of Chemical and Earth Sciences, Friedrich-Schiller-University of Jena, Germany
christian.schwartze@uni-jena.de
http://www.geoinf.uni-jena.de

**Abstract.** The hype on the Cloud is based on promising cost savings if, considering the new service platform concepts (IaaS, PaaS, SaaS) the term comes with, IT resources will be used effectively. Therefore, the trend is moving away from physical systems to more instant and short-term environments and virtualization is increasingly taking on a key role in various system architectures. This is already well accepted by a few business units such as customer relationship management or marketing, operated from *Salesforce.com* for instance [1]. However, earth scientific offers featuring specialized functions and services on demand are still rare but of great benefit in order to overcome the global changes in environmental conditions. Only one task from the field of model preprocessing at the DGHM[1] was picked out for virtualization purposes and the results will be introduced in the following.

**Keywords:** Virtualization, SaaS, Integrated Landscape Management, ILMS, GRASS-GIS, Hydrological Modelling, HRU

## 1 Brief introduction to virtualization

Virtualization in all its forms creates simulated computer environments to meet the requirements of streamlined and economical IT infrastructures, not least due to the increasing awareness of *Green-IT*. In general, software virtualization techniques are aimed at running multiple and various operating systems on one single machine, whether on client or server[2]. In the sections below, the option of system virtualization using a *Virtual Machine Monitor* (VMM) is considered. The fact that the terms VMM and VM (*Virtual Machine*) were already outlined in the remarks by Popek and Goldberg [2] in the mid 1970s does not let it seem very new, contrary to the trend news currently announced. The two definitions stated in [2] are as follows:

[1] Department of Geoinformatics, Hydrology and Modelling, University of Jena

[2] Alternative approaches to realize virtualization use the hardware directly, which is not taken into account in this paper (see e.g. *Xen* or *Microsoft Hyper-V* for such "bare metal" solutions)

> [...], the *Virtual Machine Monitor* provides an environment for programs which is essentially identical with the original machine [...] in complete control of system resources.
>
> A *Virtual Machine* is taken to be an efficient, isolated duplicate of the real machine.

Hence, the VMM, also referred to as *hypervisor*, can be understood as an abstraction layer operating either at hardware or operating system level of a physical system. In the case presented, a so-called *type-2-hypervisor* has to be used which means that the VMM is hosted by the operating system and takes care of hardware emulation (e.g. network) for each VM or rather virtual guest system. Typical representatives of that hypervisor strategy are *Microsoft Virtual PC*, *VMware Workstation* or *Oracle VirtualBox*. The latter one and its features will be described in the next section as this free software is applied in the presented example.

Compared to the container-based virtualization mechanism shown on the right of figure 1, the VMM approach provides the advantage of running guest machines on their own and native kernels with the result that the range of coexistent VM's is wider and less limited. But the freedom of choice of almost any x86 operating system is paid with small performance losses. Using modified and tuned guest kernels as it will be referred to in the next section is only one way to address these lacks if VMM based software virtualization is requested.

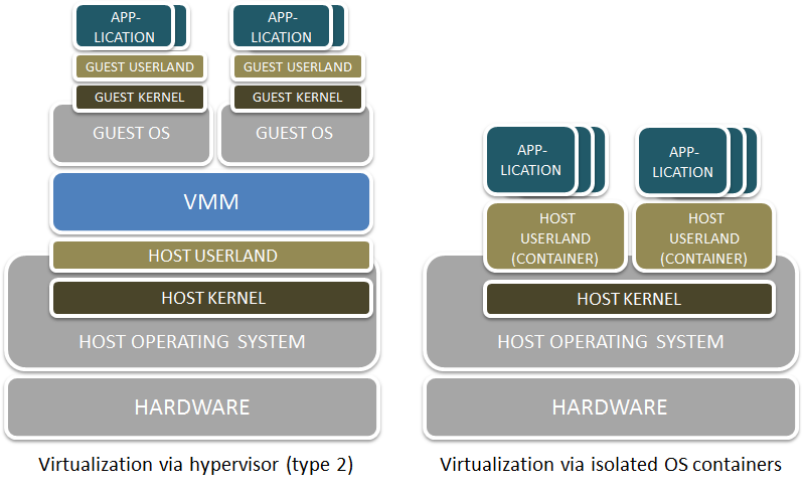


**Fig. 1. Left:** System virtualization using a type-2-hypervisor. **Right:** Isolated runtime environments as provided by *OpenVZ* and *FreeBSD Jails* for instance are using one common kernel which manages all resources (network, memory etc.) needed by the OS containers (host userlands). Other kernels within each container are not allowed.

## 2 VirtualBox

In 2007 *VirtualBox* [3] entered the highly competitive market segment of virtualization solutions and started to become available for end users. Originally developed by the German software company *Innotek* under the influence of previous business relationships (e.g. *Microsoft*), VirtualBox was bought by *Sun Microsystems* one year later. With its acquisition by *Oracle* in 2010, the software's official name is now *Oracle VM VirtualBox*.

Significant motivation for choosing VirtualBox (in its binary edition[3]) was less the large number of supported platforms, but rather the following features from a developers point of view:

- The supplied programming interfaces (in C and Python) open up opportunities to access VMM functionality such as managing or launching virtual machines from 3rd party tools. Thus, the operation of a virtual system using VirtualBox is easy and comfortable to script, which in turn is important for use on servers.
- Virtual machines packaged and distributed as *Virtual Appliances* (see section 2.1) guarantee rapid deployment and due to the industry-standard *Open Virtualization Format* (OVF) they are deployable on various virtualization platforms as long as the hypervisor comes with support for that format. This feature was added in version 2.2 of VirtualBox.
- By providing methods to start virtual machines in a special mode where graphical interfaces are disabled, VirtualBox meets the demands of hosting machines on servers with no GUI frills. This is useful especially when running the machine remotely (over VRDP[4]) or even if it is desired to hide graphical front ends from the user.

### 2.1 The virtual appliance strategy

Extending the definition in [4], a *Virtual Appliance* can be interpreted as a service consisting of a pre-installed, pre-configured operating system and an use-oriented application stack on top of it. The resulting software environment is typically delivered as an OVF archive to be run in virtual machines and enables straightforward distribution and maintenance of software as unique units. Efforts in installation and configuration are consequently minimized at customer site.

The number and kind of software that is included in the appliance depends on the field of use, or to put it more precisely and service-oriented, on the problem which is to be addressed. For example, in order to implement a geospatial appliance not only a GIS and different spatial analysis tools are needed, but also standardized protocols for serving georeferenced data over the network are of high importance. *Web Map Service* (WMS), *Web Feature Service* (WFS) or

[3] An open source edition (VirtualBox OSE) is provided for commercial purposes, but does not include all features

[4] VirtualBox extends the *Remote Desktop Protocol*, but any standard RDP client can be used

other OGC[5] compliant standards may come into consideration according to the requirements and specific application context. In the present case, a *Web Processing Service* (WPS) [5] will be applied as attention is paid to providing an almost automated workflow for preprocessing purposes of distributed hydrological simulation models (section 3).

Since a customized appliance should already bring along the underlying operating system it can be advantageous to choose one with a small footprint tailored to the package of software and drivers actually required (e.g. no window system, no USB, etc.), and even at best with a kernel that is optimized for hypervisors. These characteristics fully apply to *Ubuntu JeOS*[6] - a minimized derivative of Ubuntu Linux which runs smoothly as a guest on virtualization technologies including VirtualBox [6].

## 3 Integrated Landscape Management System

The Integrated Landscape Management System (ILMS) developed at DGHM provides an integrated, modular software platform and covers different steps of environmental system analysis and planning in a flexible and user-friendly workflow [7]. As shown on the left of figure 2, this includes ILMS*info* (management, analysis, visualization and presentation of different types of data using the web-based RBIS [8]), ILMS*image* (remote sensing module for object-oriented image analysis [9]), ILMS*gis* (derivation of modelling entities) and ILMS*model* (component-based environmental modelling using the JAMS framework [10]).

By applying the virtualization approach mentioned above to ILMS, a GRASS-GIS based solution has been implemented that aims at subdividing river catchments into *Hydrological Response Units* (HRU) [11]. These designated areas may be considered as spatially distributed entities aggregating common land use, topography or soil type, for instance, and are used in the JAMS/J2000 model suite of ILMS*model*. In general, the developed tools integrated in the process chain use open source software wherever possible.

### 3.1 HRU virtual appliance solution

HRUs are generally delineated by means of GIS overlay analysis and specific methods individually adapted to different extensions on the HRU concept, such as topological flow routing. In order to address this challenge, GRASS-GIS had to be extended by Python scripts and additional software components to cover the following HRU processing steps:

1. Physiographic data input and preparation, i.e. DEM, soils, land use
2. Remove depressions (sinks) in the DEM, derive slope and aspect (D8 flow algorithms)

[5] The *Open Geospatial Consortium* is an international and non-profit organization focusing on geospatial standards

[6] Abbreviation for *Just enough Operating System*

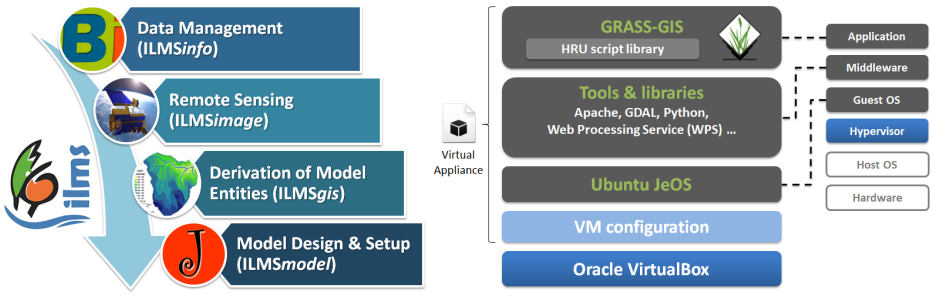


**Fig. 2. Left:** ILMS components for storage and analysis of watershed related data and timeseries, object-based and automated image classification, generation of model entities and environmental model development/application. **Right:** Main application stack of HRU service appliance

3. Reclassification of slope and aspect data
4. Calculate flow accumulation and flow direction
5. Delineate stream network, subbasins and basins based on outlets (D8, recursive upslope)
6. Generate data overlay of all input and calculated maps
7. Dissolve small areas in the overlay outcome (subbasin-by-subbasin)
8. Route water flow, support for N:1 and N:M routing (including flow rates) between HRUs (and HRU/reach)
9. Collect statistics for each HRU, e.g. average slope, soil type, centroid, elevation

As *PyWPS* [12], an implementation of the WPS standard in Python language, comes with native GRASS-GIS support, consequently almost all tasks previously listed are implemented as WPS compliant Python modules (HRU script library, see on the right of figure 2). Thus, the process development profits especially from the temporary GRASS-GIS session management and progress reports available through PyWPS. While each of the above-mentioned processes is performed by HTTP/GET request, the corresponding XML response document either contains literal outputs or more complex results (e.g. GeoTiff maps) respectively given as an URL. The latter is true, for example, at the time when step 9 is completed and will return a final HRU shapefile.

### 3.2 QGIS integration

The free available *QGIS* [13] provides great mapping and data visualization capabilities and the option of writing extensions in C++/Python. For this reason, the platform was chosen to integrate a wizard that not only guides the user through the several delineation steps, but also manages communication between the QGIS plugin (WPS client) and the virtual appliance (see figure 3).

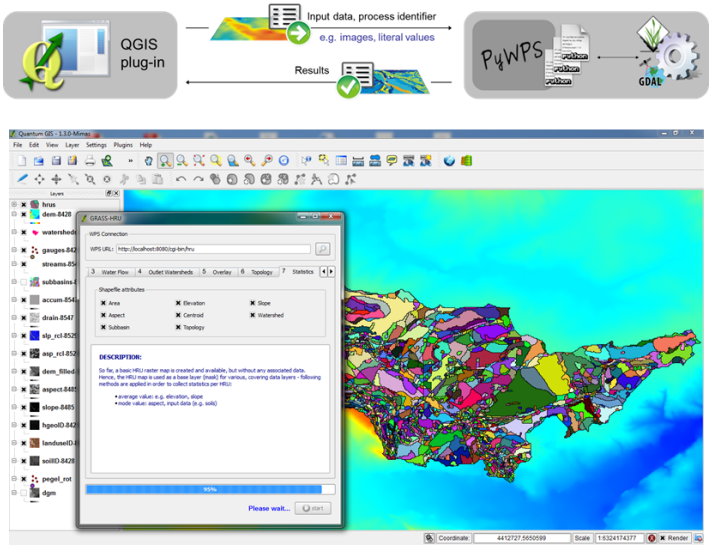


**Fig. 3.** The delineation wizard in QGIS acts as a client and sends requests to the geospatial appliance that includes all components for data processing

## 4 Summary

Using the delineation of spatially distributed model entities as an example, this paper illustrated the technical background and implementation of a geospatial appliance with focus on the deployment under VirtualBox. Due to the standardization of software environments designed for hypervisors, applications can be easily packaged and distributed to users. It is not just that the encapsulated solution stacks are installable without large installation costs and knowledge of included software, but also the additional security they provide since application errors have no impact on other hosted virtual machines.

Beyond the technical details, it was demonstrated how the VMM approach can be used in order to set up a GRASS-GIS based service which is applicable for sustainable land and water management in accordance with the ILMS workflow. During the implementation of required HRU processes, particular attention was paid to usability and the opportunity to easily extend the process chain and this is why a modular structured QGIS plugin was developed.

## References

1. Salesforce.com, `http://www.salesforce.com/`
2. Popek, G.J., Goldberg, R.P.: Formal requirements for virtualizable third generation architectures. Communications of the ACM, Vol. 17, 412–414 (1974)
3. Oracle VM VirtualBox `http://www.virtualbox.org`

4. Distributed Management Task Force, Inc. (DMTF), DMTF Standard DSP0243, Open Virtualization Format Specification 1.1.0, January 2010. http://www.dmtf.org/sites/default/files/standards/documents/DSP0243_1.1.0.pdf
5. Open Geospatial Consortium, Inc. (OGC), Web Processing Service (WPS) 1.0.0, June 2007. http://www.opengeospatial.org/standards/wps
6. Ubuntu server virtualization, http://www.ubuntu.com/business/server/virtualisation
7. Integrated Landscape Management System (ILMS) http://ilms.uni-jena.de
8. Kralisch, S., Zander F., Krause P.: Coupling the RBIS Environmental Information System and the JAMS Modelling Framework. In: 18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation, edited by R. Anderssen, R. Braddock, and L. Newham, 902–908. Cairns, Australia (2009)
9. Matejka, E., Reinhold M., Selsam P.: IMALYS - an automated and database-integrated object-oriented classification system. In: GEOBIA 2008 - Pixels, Objects, Intelligence: Geographic Object Based Image Analysis for the 21st Century. Calgary, Canada (2008)
10. Jena Adaptable Modelling System (JAMS) http://jams.uni-jena.de
11. Flügel, W.-A.: Delineating Hydrological Response Units (HRU's) by GIS analysis for regional hydrological modelling using PRMS/MMS in the drainage basin of the River Bröl, Germany. Hydrological Processes, Vol. 9, 423–436 (1995)
12. PyWPS Development Team: Python Web Processing Service (PyWPS) http://pywps.wald.intevation.org
13. Quantum GIS project (QGIS) http://www.qgis.org