The Path to Virtual Machine Images as First Class Provenance

Sotiria Lampoudi

Department of Computer Science
University of California
Santa Barbara, CA 93106-5110, USA

Abstract. The scientific community's increased exposure to cloud computing has led to increased familiarity with the machine virtualization technology that underpins the cloud. Efforts to define and implement provenance for the cloud are under way. In the meantime, however, an orthogonal idea, aimed at quickly facilitating repeatability and curation, has taken shape. This is the idea of using virtual machine images (VMIs) as authoritative, encapsulated and executable records of computations, especially computations whose results are destined for publication and/or re-use. In this paper we trace the origins of this idea, discuss its strengths and limitations, and outline some of the future steps and potential pitfalls on the path to its realization. We also analyze how this approach differs from, yet composes with traditionally defined provenance.

1 Introduction

Modern machine virtualization technology, as typified by the open source Xen hypervisor[1], on the software level, and Intel VT[2] and AMD-V[3], on the hardware level, has reached maturity. It is possible for the working computational scientist to work entirely on a virtualized workstation. However, this is not typical. Instead, most computational scientists exposure (conceptual or hands-on) to machine virtualization appears to be via exposure to cloud computing (CC). Virtualization is only one of the fundamental technologies underpinning only one of the flavors of cloud computing (specifically, Infrastructure as a Service, abbreviated IaaS). However, the correlation between the concepts of, on the one hand, virtualization and, on the other, the cloud has led to some confusion between the two, especially in publications originating outside computer science. Specifically, the issue appears to be confusion about which attributes of modern virtual machines are consequences of virtualization itself and which are consequences of virtualization being used in the cloud setting.

One body of discourse where the this confusion is consequential concerns the use of virtual machine images (VMIs) to facilitate scientific repeatability and

[1] `http://www.xen.org/`

[2] `http://ark.intel.com/VTList.aspx`

[3] `http://sites.amd.com/us/business/it-solutions/virtualization/Pages/virtualization.aspx`

curation. The overarching idea or strategy is this: **to use VMIs as authoritative, encapsulated and executable records of computations, especially computations whose results are destined for publication and/or reuse**. (Whenever, in subsequent parts of this paper, we refer to "this idea" or "this strategy" without a tighter scope, the referent should be understood to be the idea above, in bold.)

The timeliness of this idea becomes clear when one notes three factors: a) the buzz surrounding the cloud and, consequently, virtualization, b) the proliferation of calls in the popular scientific press [1, 2], and even the NSF[4] for code publication, Open Science [3], etc., and c) the rise of provenance, the field that is formally concerned with repeatability and curation.

We, like the people who have published and spoken about this idea before us, argue that it has merit. However, our focus is on clarifying the context in which the idea has merit, at present, and identifying possible paths to the expansion of that context in the future, based on emerging technologies and standards.

The aims of this paper are many-fold: first, to trace the origins and shape of this idea, by reviewing some of the discourse (literature and talks) in which it has appeared; second, to discuss its feasibility, strengths and weaknesses, especially in light of the crystal clear distinction between virtualization in general and virtualization for the cloud; third, to point out some of the challenges and opportunities along the path to implementing this idea; and fourth, to point out how this idea is orthogonal to, yet composes with, current research directions on provenance for the cloud.

The paper is organized as follows: section 2 summarizes the relevant technologies; section 3 reviews the discourse on the subject; section 4 discusses the merits, drawbacks and limitations of the strategy and some technology gaps; section 5 discusses what provenance has to contribute to this idea.

2 Background

A Virtual Machine Monitor (VMM) or hypervisor (e.g. Xen, KVM, HyperV, VMware) is a software layer that multiplexes access by many VMs to a machine's physical hardware. A VMM can present the hardware to VMs in one of two ways: a) para-virtualized, or b) fully virtualized. If the hardware is para-virtualized it means that a layer of emulation or abstraction is placed between the physical hardware and the virtual hardware. Para-virtualization can sometimes lead to better-than-native performance by simplifying or slimming down the drivers required to access devices [4]. In full virtualization, direct, multiplexed access by VMs to the hardware is mediated by the VMM. Full virtualization most often requires hardware support.

[4] `http://www.nsf.gov/pubs/2011/oise11003/`"Changing the Conduct of Science: Summary Report of Workshop Held on November 12, 2010," at the National Science Foundation Workshop Changing the Conduct of Science in the Information Age, June, 2011.

Hypervisors differ in the precise details of their organization, which we will not discuss here. However, the abstract notion of a virtual machine image (VMI), that is, a collection of whatever is needed for a hypervisor to boot a VM, is useful across the board. For a Linux installation, a VMI consists of a Linux kernel/RAMdisk and a root filesystem. Many of the existing VMI packaging formats are inter-convertible, by separating the source format into its components (kernel/RAMdisk, filesystem) and re-packaging it according to the destination format.

Notice that up to this point everything we have described concerns machine virtualization in general, and we have said nothing about cloud computing.

Cloud computing provides available (”when you need it”), dynamic (”pay for what you use”) and elastic (”use as much as you need”) provisioning of computational resources. It comes in three flavors: Infrastructure, Platform and Software as a Service (IaaS, PaaS and SaaS, respectively), of which we will be concerned with the first, IaaS. The ”infrastructure” of IaaS refers to computational, network and storage resources. The computational resources are presented to the user as virtual machines (VMs).

In this paper we take the position that IaaS cloud computing is currently exemplified by Amazon’s Elastic Compute Cloud (EC2), the first and currently dominant player in the field, and other software stacks (e.g. Eucalyptus [5]) that are compatible with it. It appears that, at the moment, Amazon’s APIs are the *de facto* standard in IaaS, although it remains to be seen whether that will continue to be the case, as later arrivals to the industry are attempting to craft and force the adoption of competing standards.

Most VMMs, Xen included, are capable of presenting a configurable mix of virtualized and para-virtualized resources. In order to achieve availability and elasticity at scale, Amazon’s EC2 severely restricts the number of VM configurations among which their customers can choose. The hypervisor of choice at EC2 is Xen, the VMs all have one network interface, and they come in only a handful of VM configurations, parameterized by the number of CPU ”core equivalents” and RAM. As of the writing of this paper, one instance type provides access to additional Graphical Processing Unit (GPU) hardware. This standardization of VMs allows EC2 to account for its resources in a crude but effective fashion that makes it possible to deploy VMIs very rapidly.

There is an important distinction between what happens to a VMI if it is booted on a run-of-the-mill workstation with virtualization enabled vs on the cloud. In the former case, any changes to the filesystem that occur while the VM is running become permanent by virtue of the fact that the filesystem is persistent. In the original EC2 usage model, the VMI’s filesystem was treated as an immutable object from which multiple, initially identical, but independently evolving VMs could be booted. A VM that has booted from a VMI, and whose running state can be different from other VMs that have booted from the same VMI, is called an *instance*. Nowadays there are two ways to configure an EC2 instance: under the original model, the filesystem of an instance is volatile, so that, if the user wants any changes that occur to it to become persistent, he or she must explicitly save (”bundle”) the image; under the newer model, the filesystem is stored in a persistent, rather than a volatile, store (Elastic Block Store, EBS). The newer model is similar to the behavior of a run-of-the-mill virtualized workstation that has booted from a network attached filesystem.

3 Literature Overview

The use of VMIs for repeatability and curation of scientific results is a strategy, rather than a piece of software or infrastructure or a standard. Thus, it is difficult to definitively trace when and by whom it may have first been used. Here we first survey some foundational literature on the use of (para-)virtualization and the cloud in scientific computing, and then we present some of the published literature and scientific talks in which the idea of using virtualization for repeatability and curation appears.

Virtualization has a long history in the area of operating systems, dating back to IBM’s PDP-11 in the 1970s. Machine virtualization was historically associated with a performance penalty due to the additional software layers required to achieve isolation from the hardware. Thus, an important question is whether modern machine (para-)virtualization technology suffers from the same drawback, which would disqualify it from being an ideal platform for scientific computing. This question is studied in [4, 6] in the context of para-virtualization in Xen.

The question of whether cloud computing achieves performance worthy of scientific computing (presumably at decreased cost) is altogether different. As we described in the previous section, CC severely restricts the VM configurations available to the user, can be subject to dramatic fluctuations in demand, and uses CPU core equivalents, to circumvent the fact that the underlying hardware can be heterogeneous, depending on when it was deployed. Thus, while to assess the impact of para-virtualization it is possible to compare performance of the virtual vs the native hardware, measured performance in EC2 must be compared to either the performance of an unrelated computational resource, e.g. a cluster, or to the theoretical peak CPU core equivalent performance that Amazon describes. Other performance metric comparisons can be similarly confused. Nevertheless, this is an important question, and it is studied in [7, 8] and [9].

[10] proposes the use of VMs to encapsulate large, complicated stacks of scientific software so they can be deployed across supercomputing centers without the need to install each of the software packages individually in every new environment. The paper describes a proof-of-concept of this strategy developed for the High-Energy and Nuclear Physics application STAR, deployed on a five node cluster. While this paper does not address repeatability or curation, it proposes virtualization as a solution for achieving encapsulation and portability of the software environment of a scientific application, one of the big challenges on the way to repeatability and curation.

In 2011 concerns over computational repeatability reached a crescendo, mostly in the form of calls for source code publication. The idea of using VMIs to achieve computational repeatability and curation appeared in two venues. In early 2011 the scientific publishing house Elsevier announced a call for papers and proofs-of-concept called the "Executable Paper Grand Challenge"[5]. The statement of the problem they posed was: "how can data intensive research be made repeatable and workable within the context of a scholarly journal article?" The second place prize was won by [11] in which a cloud-based environment for authoring, sharing, modifying and executing computational results was described. SHARE includes a web portal for accessing its functionality, and the VMIs on which it is based are deployed in an on-premise Eucalyptus cloud hosted at Eindhoven University of Technology, Netherlands.

In July 2011 a workshop entitled "Reproducible Research: Tools and Strategies for Scientific Computing" took place[6]. Three talks made reference to the strategy of using VMIs to facilitate repeatability. In the first Tiffani Williams introduced Paper Mache, which was also one of the finalists of the Executable Paper Grand Challenge, and was later demoed at ICCS [12]. It consists of a workbench, a modified hypervisor and a tool for creating (visual) images, and it creates custom VMIs intended for use throughout the submission, review, modification and curation process of a paper. Sorin Mitran gave a talk entitled "Archiving Computational Research in Virtual Machines"[7], and Bill Howe gave a talk entitled "Virtual Appliances, Cloud Computing, and Reproducible Research"[8].

4 The Approach

The strategy of using VMIs to achieve repeatability and curation of scientific computing results is just that: a strategy. As such, it permits implementations that span a wide range of abstraction and/or automation, SHARE, Paper Mache and Mitran's hand-crafted approach being only some of the possibilities.

At its core, the strategy simply requires a scientist to – somehow – generate a VMI of a computational environment capable of reproducing his or her results, and to make that VMI publicly accessible. The strategy hinges on the assumptions that the VMI will be an *authoritative*, *encapsulated* and *executable* record of computations. To say that a VMI is an authoritative record of a computation is trivially true. Encapsulation, on the other hand, can be achieved by proper design, a subject we discuss briefly in the last section. But the claim that a VMI will be generally executable bears some scrutiny.

At present there is no unqualified answer to the question: if a VMI is published, will it be executable in the short-term (repeatability) and in the long-term (curation)?

[5] `http://www.executablepapers.com/`
[6] `http://www.stodden.net/AMP2011/index.html`
[7] `http://www.stodden.net/AMP2011/slides/ReproducibleResearchMitran.pdf`
[8] `http://www.stodden.net/AMP2011/slides/reproducibility_in_the_cloud-HOWE.pdf`

The issue here is that not all VMIs are created equal when it comes to the likelihood that they will be able to be booted in the near and far future. A VMI that is cloud-compatible, in the sense of being compatible with EC2, is guaranteed to boot under most versions of Xen, to conform to a simplistic device model, and to have one of only a handful of CPU and RAM configuration expectations. In the short term, it will be easy to boot it on any Xen installation, whether on the cloud or not. In the long term, it is reasonable to argue that, because it is one of many thousands of VMIs with similar hypervisor/VM expectations, it has a better chance of being supported.

The same cannot be said of an arbitrary VMI generated on a virtualized workstation, a virtualized HPC cluster or any other arbitrary virtualized environment. Indeed, even a VMI that is compatible with an on-premise cloud is not guaranteed to be cloud-compatible, in the sense of being EC2-compatible, since on-premise cloud software, like Eucalyptus, can be configured to work with hypervisors (e.g. KVM) and VM configurations other than those specified by EC2.

In order to improve the odds that an arbitrary VMI will be able to boot on today's foreign and tomorrow's alien resources, it seems reasonable to propose that there should exist a metadata format for specifying a VMI's hypervisor and VM configuration expectations. Indeed such a standard now exists: the Open Virtualization Format[9]. On the one hand, this is a welcome development for the scientific community, and any community interested in harnessing the power of virtualization without adopting the cloud model and its inherent simplicity-for-availability trade-off. On the other hand, because the flexibility of being able to spin up an arbitrarily large assortment of VM types comes with a penalty in availability, it is unlikely that Amazon's EC2 and EC2-compatible software stacks that care about scale will adopt OVF. This leaves the scientific community and anyone else who wants to adopt arbitrary OVF-wrapped VMIs with a difficult decision, based on speculation about the future. Will OVF thrive (once the standard converges) and become a new classic, like Portable Document Format (PDF) before it, which also had difficulties at first? Or will arbitrary VMIs will be relegated to un-executability and OVF to the cemetary for useless metadata formats?

Thus, it appears that cloud-compatible VMIs have an excellent chance of being executable in the near future, and an arguably good chance at being executable in the further future. Arbitrary VMIs, however, should at a minimum come with a description of their hypervisor and VM expectations, preferably in a recognized format, like OVF, and may not, even then, have a good chance of being executable as their date of creation fades into the past.

A question to which the answer is not dictated by the strategy of using VMIs for repeatability and curation, and which is, therefore, open to discussion, is whether a published VMI needs to be the scientist's original working environment, or whether it can be a re-creation of it.

Articles calling for source code publication for science [1, 2] cite the burden of cleaning up and polishing code and data in order to produce a final publish-

[9] `http://dmtf.org/standards/ovf`

able product as being an important deterrent from code publication. One might imagine, then, that this deterrent would extend to re-creating a computational environment on a VMI for publication. If the community's objective is to promote the rapid and wide adoption of the VMI publication strategy, then perhaps it makes sense to express a preference for the VMI environment to be the original environment in which the results were obtained, rather than a re-creation of it.

Bill Howe, in his AMP2011 talk, made a related argument for conducting the entire life cycle of computational science on the cloud. He pointed out that the cloud is an excellent environment for cheap experimentation with software. He further explained that often there is no clear distinction between writing, testing, debugging and final experimentation with scientific software. Thus, carrying out the entire lifecycle of computational science on the cloud has the advantage of allowing publication of a VMI whenever the cycle happens to be complete, however unpredictable, messy and heterogeneous that final VMI may be.

Sorin Mitran brought up several drawbacks to the VMI publication for repeatability and curation approach in his AMP2011 talk. According to him, those are: a) that VMIs contain a large amount of data, and so pose a storage challenge, b) that there is no guarantee that a VM emulator will exist in the far future, and c) that this approach does not support the special-purpose hardware and multi-machine configurations that are often used in computational science. He countered his own arguments with the observations that, a) storage is getting cheaper, b) there is strong vendor support for VMs, and c) CUDA, MPI and OpenMP are now all supported. To his counter-argument for (a) we would add that, irrespective of whether storage gets cheaper, the problem of storage bloat is one that is being encountered in many settings, including provenance, and thus is being independently studied and addressed. We believe that concern (b) is different for cloud-compatible vs arbitrary VMIs, as we explained above. Finally, we believe that concern (c) actually encompasses two separate issues. On the one hand there is the issue of special hardware availability (e.g. GPUs and FPGAs). Amazon is currently addressing at least GPU availability, but one has to wonder whether that support, which appears to be a deviation from EC2's core business model, will last. In any case, a metadata model such as OVF, that allows the specification of virtual hardware requirements, theoretically addresses how to make sure VMIs appropriately ask for the resources they require. On the other hand is the issue of Multiple Program Multiple Data software architectures. OVF, again theoretically contains the ability to specify a "federation" of VMIs that are intended to be deployed in a particular order.

While complete solutions based on the strategy of using VMIs for repeatability and curation, such as SHARE and Paper Mache, are appealing in settings where a GUI-based, high-level, end-to-end "executable paper" approach is desirable, they are only one way to implement the strategy. We would like to propose that the strategy will have a better chance at wide adoption if an ecosystem of lower-level software, libraries and conventions addressing small, encapsulated aspects of it emerges. Such individual components could be mashed-up and remixed as necessary to form entire solutions. They would also be individually easier to replace as the relevant technologies change. The components would each stand a better chance of being adopted individually, not only in the scientific computing setting, but also in enterprise. Some ideas for such components follow.

One of the things that must be specified in order to make a published VMI useful for reproducibility and curation purposes is what it is capable of. In the case of SHARE and Paper Mache, the GUI or the author(s) generate scripts that perform the tasks that the VMI is supposed to demonstrate. Analogously, it would be useful if there was a general purpose convention addressing where one should look to understand what a VMI is about. It would be convenient if the capabilities of a VM could be specified in a human readable and in an executable format in some standard location. Possibilities include that the home directory of some default user should contain a file called README, that `make` should be invoked at some standard location to demonstrate some of the functions of the VMI, or that the last init script (in the case of UNIX variants) launch the VMI's functionality.

Further, we would like to propose two types of systems-level improvements that would render Linux an easier operating system from which to generate VMIs for repeatability and curation. Both of these concern the handover of Linux-based VMIs from the author to end-users.

The first class of improvements concern de-authentication and de- personalization of a running VM. A VMI normally contains files that link back to its author. Depending on how the VMI was generated there may be more of fewer of these. These files can range from password files, to public and private keys, to, simply, settings. In general, for the purposes of implementing the VMI publication strategy, it would be useful to have stand-alone tools for "undressing" a VMI of the authentication and personalization of the author, and "re-dressing" it, if necessary, with the authentication and personalization of the person instantiating it.

A second class of improvements concerns what Chris Read calls "transient runtime pollution"[10]. Because Linux uses the filesystem for process ID locks, logs and other transient files, these can bundled into a VMI. Aside from their potentially revealing nature, e.g. in the case of logs, they can present technical difficulties when they represent a state of the VM that is no longer valid. It would be useful to have ways of specifying where and how the filesystem may be polluted with such transient runtime artifacts, and how to clean them up during the process of packaging a VMI for publication.

5 Discussion

The strategy of using VMIs for repeatability and curation is an attempt to quickly and cheaply (in terms of time investment, and also commitment) achieve a subset of the objectives of provenance. In a sense, this strategy is a case of "low hanging fruit". But, as we explained in the previous section, it appears that

[10] `http://blog.chris-read.net/2009/04/08/ec2-ami-creation-tips-part2/`

the strategy is currently useful in only a fraction of the cases in which it could be applied.

Research on provenance for the cloud is largely orthogonal to the strategy of using VMIs for repeatability, as it focuses on the extension of existing provenance methodologies and tools to the cloud [13–18]. Current provenance-for-the-cloud approaches do not impede publication of VMIs, and, conversely, publication of VMIs does not detract from the impact of provenance-for-the-cloud approaches.

What the strategy of using VMIs for repeatability and curation does stand to benefit from is exposure to the list of concerns and capabilities within the provenance community. Specifically, there are four issues that the provenance community has thought about that also arise when considering this strategy. The first is the issue of transforming and packaging VMIs in such a way that they achieve the degree of encapsulation necessary for repeatability and/or curation. Indeed, it can be a provenance problem to identify and resolve data dependencies, e.g. internet URL references to data used as inputs, in a VMI.

The second issue, which the provenance community is also faced with, is that of bloat in storage requirements. In the same way that provenance metadata can exceed in volume the data that it annotates, so can bundled VMIs can be much larger than, e.g., the datasets or figures whose computation they document.

The third issue with which the provenance community has a great deal of familiarity is that of constructing a naming and location scheme for VMIs. Whether such a scheme will be like URI, DOI, ARK or something else is wide open. One possibility that we can suggest is an extension to ARK[11]that provides for an inflection for re-creating, via VMI execution (as opposed to retrieving from storage), data for which that makes sense.

Finally, we believe that the development of a standard for specifying the hypervisor and VM expectations of individual VMIs and collections of VMIs would benefit from the involvement of the provenance community. Among current customers of the cloud, the demand for configurable VMs, virtualized special-purpose hardware and VM federations is low, whereas among the scientific computing community, and especially in the high performance computing (HPC) community, demand for these features is high. If VMIs are to become first class provenance artifacts, then the provenance community should be involved in the specification of the relevant metadata formats. Aside from this being an opportunity to have significant real-world impact, it is also a unique academic opportunity. Since VMIs are (at least in some cases, as we hope we have convinced you) executable, the metadata surrounding them is, at once, both a *descriptive* record of their origin, and a *prescriptive* recipe for their reconstruction.

References

1. McCafferty, D.: Should code be released? Commun. ACM **53**(10) (October 2010) 16–17

[11] `https://confluence.ucop.edu/display/Curation/ARK`

2. Ince, D.C., Hatton, L., Graham-Cumming, J.: The case for open computer programs. Nature **482**(7386) (February 2012) 485–488
3. Nielsen, M.: Reinventing Discovery: The New Era of Networked Science. Princeton University Press (2011)
4. Youseff, L., Wolski, R., Gorda, B., Krintz, R.: Paravirtualization for hpc systems. In: In Proc. Workshop on Xen in High-Performance Cluster and Grid Computing, Springer (2006) 474–486
5. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: The eucalyptus open-source cloud-computing system. In: Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on. (May 2009) 124 –131
6. Youseff, L., Seymour, K., You, H., Dongarra, J., Wolski, R.: The impact of paravirtualized memory hierarchy on linear algebra computational kernels and software. In: Proceedings of the 17th international symposium on High performance distributed computing. HPDC '08, New York, NY, USA, ACM (2008) 141–152
7. Deelman, E., Singh, G., Livny, M., Berriman, B., Good, J.: The cost of doing science on the cloud: the montage example. In: Proceedings of the 2008 ACM/IEEE conference on Supercomputing. SC '08, Piscataway, NJ, USA, IEEE Press (2008) 50:1–50:12
8. Ostermann, S., Iosup, A., Yigitbasi, N., Prodan, R., Fahringer, T., Epema, D.: A performance analysis of ec2 cloud computing services for scientific computing. In Avresky, D.R., Diaz, M., Bode, A., Ciciani, B., Dekel, E., Akan, O., Bellavista, P., Cao, J., Dressler, F., Ferrari, D., Gerla, M., Kobayashi, H., Palazzo, S., Sahni, S., Shen, X.S., Stan, M., Xiaohua, J., Zomaya, A., Coulson, G., eds.: Cloud Computing. Volume 34 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer Berlin Heidelberg (2010) 115–131
9. Jackson, K., Ramakrishnan, L., Muriki, K., Canon, S., Cholia, S., Shalf, J., Wasserman, H., Wright, N.: Performance analysis of high performance computing applications on the amazon web services cloud. In: Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. (December 2010) 159 –168
10. Keahey, K., Freeman, T., Lauret, J., Olson, D.: Virtual workspaces for scientific applications. Journal of Physics: Conference Series **78**(1) (2007) 012038
11. Gorp, P.V., Mazanek, S.: Share: a web portal for creating and sharing executable research papers. Procedia Computer Science **4**(0) (2011) 589 – 597 Proceedings of the International Conference on Computational Science, ICCS 2011.
12. Brammer, G.R., Crosby, R.W., Matthews, S.J., Williams, T.L.: Paper mch: Creating dynamic reproducible science. Procedia Computer Science **4**(0) (2011) 658 – 667 Proceedings of the International Conference on Computational Science, ICCS 2011.
13. Muniswamy-Reddy, K.K., Macko, P., Seltzer, M.: Making a cloud provenance-aware. In: First workshop on on Theory and practice of provenance. TAPP'09, Berkeley, CA, USA, USENIX Association (2009) 12:1–12:10
14. Muniswamy-Reddy, K.K., Macko, P., Seltzer, M.: Provenance for the cloud. In: Proceedings of the 8th USENIX conference on File and storage technologies. FAST'10, Berkeley, CA, USA, USENIX Association (2010) 15–14
15. Muniswamy-Reddy, K.K., Seltzer, M.: Provenance as first class cloud data. SIGOPS Oper. Syst. Rev. **43**(4) (January 2010) 11–16

16. da Cruz, S.M.S., Paulino, C.E., de Oliveira, D., Campos, M.L.M., Mattoso, M.: Capturing distributed provenance metadata from cloud-based scientific workflows. Journal of Information and Data Management **2**(1) (2011)
17. de Oliveira, D., Ocana, K., Ogasawara, E., Dias, J., Baiao, F., Mattoso, M.: A performance evaluation of x-ray crystallography scientific workflow using scicumulus. In: Cloud Computing (CLOUD), 2011 IEEE International Conference on. (July 2011) 708 –715
18. Slominski, A.: Flexible creation and adaptive execution of scientific workflows in cloud and grid environments by using web 2.0-based electronic lab notebook metaphor. In: Proceedings of the 2010 6th World Congress on Services. SERVICES '10, Washington, DC, USA, IEEE Computer Society (2010) 326–327