Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 4 (2011) 589–597

Procedia Computer Science

International Conference on Computational Science, ICCS 2011

SHARE: a web portal for creating and sharing executable research papers

Pieter Van Gorp[a], Steffen Mazanek[b]

[a]p.m.e.v.gorp@tue.nl, Eindhoven University of Technology, The Netherlands
[b]steffen.mazanek@gmail.com, Munich, Germany

Abstract

This paper describes how SHARE (Sharing Hosted Autonomous Research Environments) satisfies the criteria of the Elsevier 2011 Executable Paper Grand Challenge. This challenge aims at disseminating the use of systems that provide reviewers and fellow scientists a convenient way to reproduce computational results of research papers. This can involve among others the calculation of a number, the plotting of a diagram, the automatic proof of a theorem or the interactive transformation of various inputs into a complex output document. Besides reproducing the literate results, readers of an executable paper should also be able to explore the result space by entering different input parameters than the ones reported in the original text.

SHARE is a web portal that enables academics to create, share, and access remote virtual machines that can be cited from research papers. By deploying in SHARE a copy of the required operating system as well as all the relevant software and data, authors can make a conventional paper fully reproducible and interactive. Shared virtual machines can also contain the original paper text — when desirable even with embedded computations.

This paper shows the concrete potential of SHARE-based articles by means of an example virtual machine that is based on a conventional research article published by Elsevier recently. More generally, it demonstrates how SHARE has supported the publication workflow of a journal special issue and various workshop proceedings. Finally, it clarifies how the SHARE architecture supports among others the Elsevier challenge's licensing and scalability requirements without domain specific restrictions.

Keywords: reproducible research, virtualization, web portal, challenge, executable paper, research 2.0

1. Introduction

SHARE [1] has emerged from the organization of the Transformation Tool Contest (TTC, formerly known as GraBaTs), a yearly event aimed at the evaluation and dissemination of advanced transformation techniques and related software[1]. Since TTC is a research contest, it attracts many submissions that rely on software that is still in the prototype phase. This implies, among others, that

1. the software is sometimes not yet publicly released,
2. the software is often difficult to install or configure for proper use with particular inputs,

[1] `http://planet-mde.org/ttc2011/` (all URLs from this paper have last been verified on 2011-03-14)

doi:10.1016/j.procs.2011.04.062

3. the software is often incomplete or only working in combination with other software, which in turn may require a separate download, installation and license,
4. the software version required for the particular execution might not be available anymore for download in the future.

In other scenarios, one is struggling with license issues of the data sets that have been used to come to a particular conclusion. Many papers also rely on very large data sets that, irregardless of license issues, are too tedious to download as part of a paper reviewing task or when performing a literature survey and questioning the validity of the alleged research results.

In all of these cases, it would be very convenient if one could simply click a hyperlink within a research paper to arrive at an environment where all software and data related to the paper would be optimally installed and ready for (temporary and secure) evaluation. Since 2009, we provide SHARE as a free academic service for simplifying as much as possible the workflow for creating such executable papers.

This paper is organized as follows: First, we describe the SHARE system from the perspective of a reader, a volume editor and an author. Then we describe an example machine in order to give an impression of the possibilities provided by SHARE. This machine contains all the executable artifacts of an article previously published by Elsevier as a conventional research article. We continue by briefly distinguishing SHARE from related work. Finally, we conclude by summarizing how SHARE meets the challenge's criteria.

2. Perspective of the Readers of an Executable Paper in SHARE

This section introduces the SHARE system from the perspective of its most casual user, that is from the perspective of the *reader* of an executable paper (reviewers or others).

Figure 1 shows a usage scenario that is typical for readers of a SHARE-supported publication. Via the browser shown at bullet 1, the reader follows a link from a reference in a (conventional) article. This link points to a webpage, where a specific virtual machine image can be instantiated. Assuming that the reader has never used SHARE before, he first follows a registration procedure (shown at bullet 2). Existing users would simply log in, or would jump to the screen from bullet 3 in case they were already logged in. On the screen shown at bullet 3, the user should just click *Request Session* if he wishes to instantiate the hyperlinked virtual machine image immediately. The SHARE website then balances the load between all virtual machine servers that host the requested virtual machine image. Moreover, it enables users to reserve a future timeslot if all virtual machine servers are fully loaded (see field set "*When?*" on the page from bullet 3).

Figure 1, bullet 4, displays SHARE's main page for logged in users. In the middle of the page, the details of active sessions are listed. This involves (1) the physical machine at which the virtual machine is running and (2) the port on this server where the Remote Desktop Protocol (RDP) server is listening. In this example, the user has one active session on port *6977* of the machine *jobs.cmi.ua.ac.be*. Bullet 5 shows how the user should enter that information in an RDP client. Users should authenticate using their credentials from the SHARE website. Obviously, the last step (shown at bullet 6) involves working remotely on the virtual machine. To emphasize that users can work concurrently on multiple virtual machines, bullet 6 shows three active RDP sessions. RDP clients are available for most modern operating systems (among others Windows, Linux and Mac). Note that SHARE thus supports multiple operating systems both at the level of the remote virtual machines as well as at the level of the connecting clients running on the user's machine.

3. Perspective of Volume Editors

In SHARE, each virtual machine is part of a so-called bundle. Typically, a SHARE bundle is related to a workshop or a journal issue. Users can subscribe to multiple bundles in order to access the respective machines. Any SHARE user also can apply for *bundle organization* rights. As for other administrative workflows, this would involve submitting a simple form, after which an automated e-mail would be sent to the SHARE users that have the appropriate rights for authorizing the request. For this particular workflow, so-called *bundle administrators* would be notified [1]. Note that SHARE's automated e-mails contain prepared links that minimize the administrative workload.

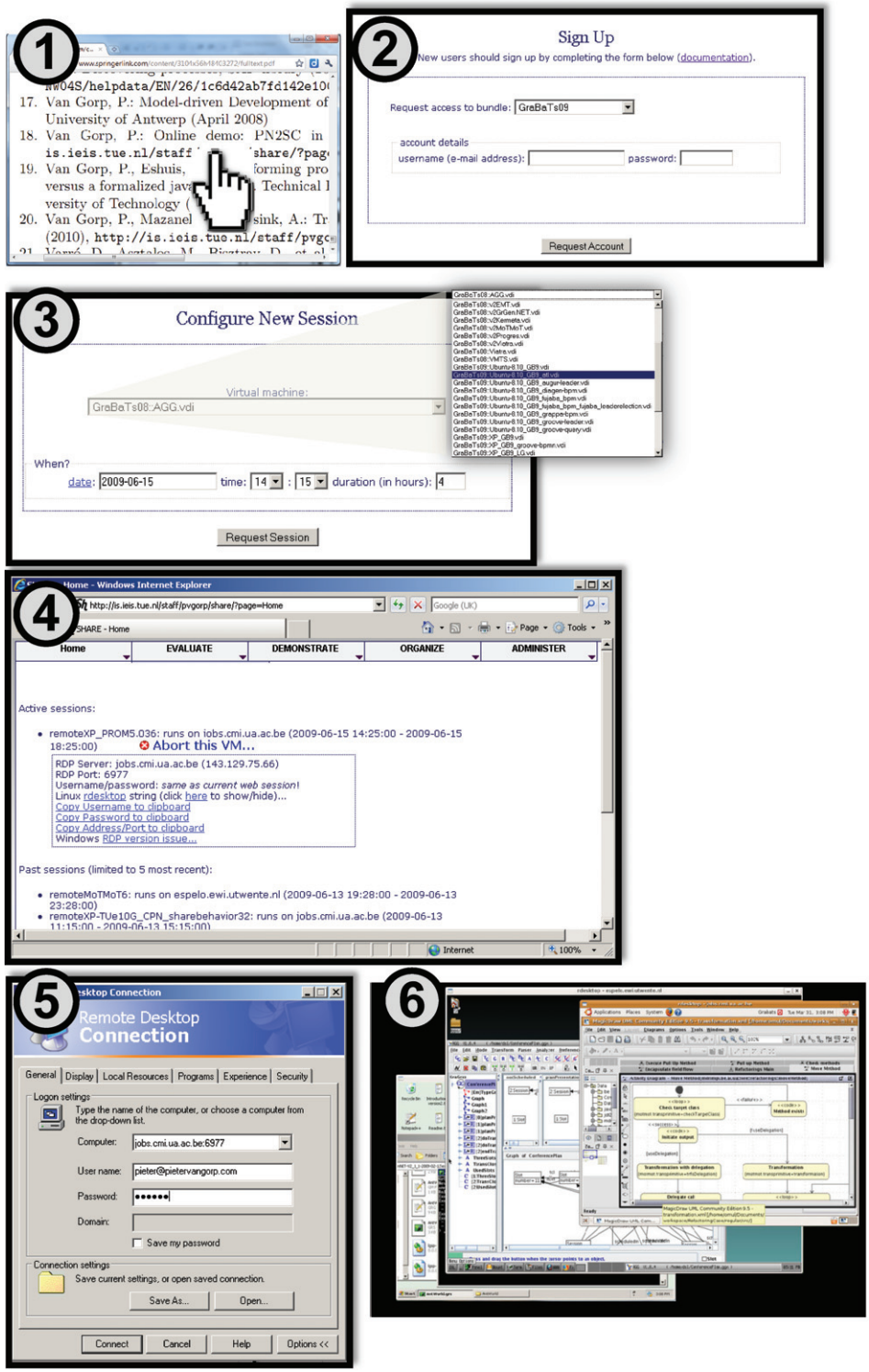

Figure 1: Typical scenario for the reader of a journal special issue or of a workshop proceedings.

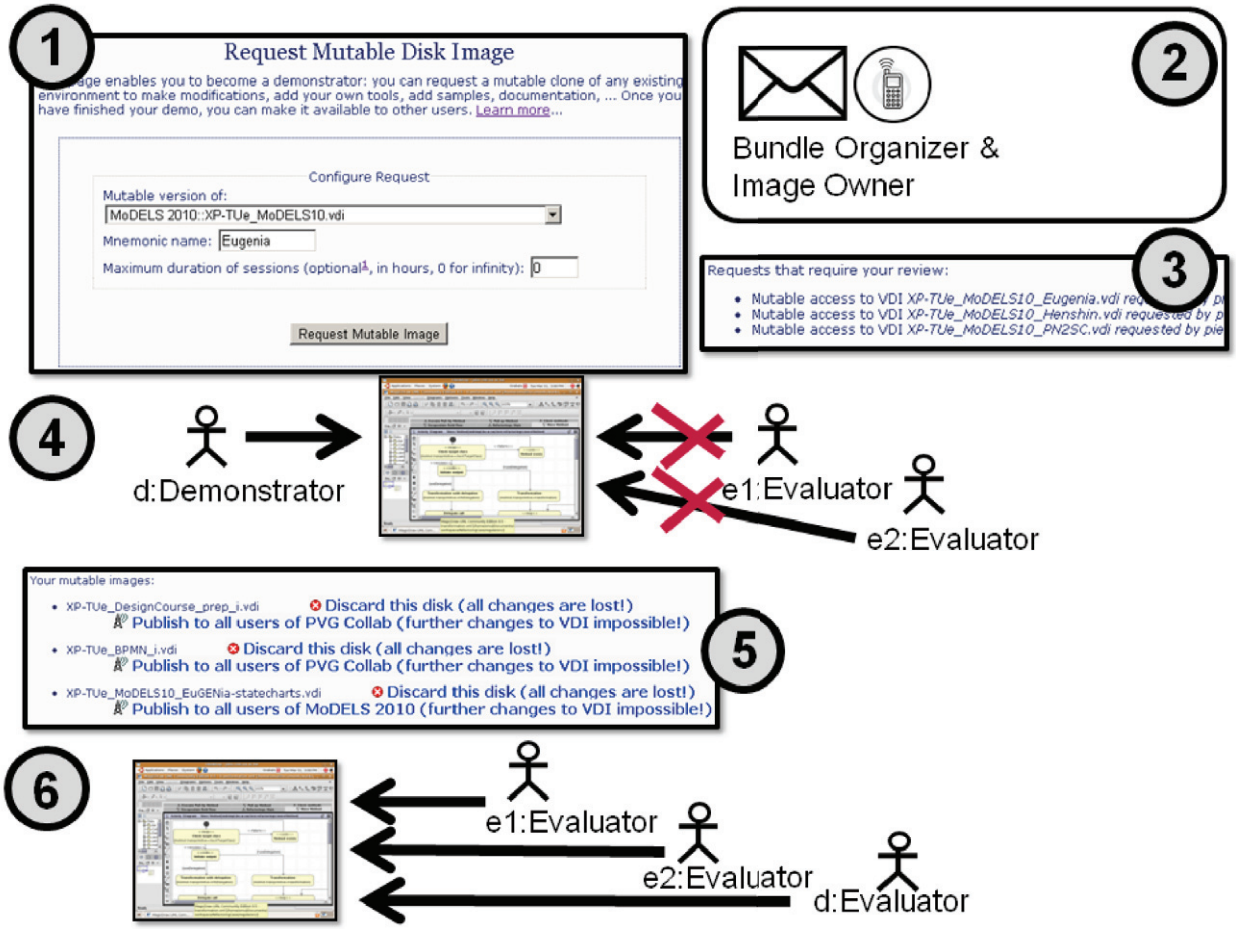

Figure 2: Typical scenario for the author of an executable paper based on SHARE.

In most cases, volume editors want to advertize their (executable) papers as much as possible. SHARE not only provides HTML and BibTeX code that can be conveniently adopted in this context, but also provides index pages that enable anonymous visitors of the SHARE website to browse through the list of available virtual machine images. Bundles that are of no interest to the general public can be hidden by their organizers and will then be excluded from SHARE's index pages.

Once a bundle administrator has approved the request for the new bundle, the volume editor will be notified. At this point, the volume editor needs to populate the empty bundle with one or more "base" virtual machine images. Such base images are technically no different from images that relate to executable papers, but conceptually they are different since (1) they do not relate to a research paper and (2) they typically contain instructions for the authors of the volume.

4. Perspective of the Authors of Executable Papers

Authors can create new SHARE images based on existing ones by means of a simple "clone" operation. SHARE ensures that clones are only created after approval of both the bundle organizer and the owner of the original image. In many cases, authors simply clone one of the base images, as prepared by the bundle organizer. In other cases, authors re-use images that they (or other authors) have previously contributed to SHARE. The latter often saves precious time in practice.

Figure 2 visualizes the typical flow for the author of an executable paper in SHARE: in step 1, he selects the image that he wants to clone, in step 2, the bundle organizer as well as the image owner are notified of this request. Step 3 shows that these stakeholders can decide to postpone the handling of individual requests and handle them in batch via the SHARE website. In this example, we assume that both stakeholders approve the request. In step 4, the author (labeled as *demonstrator d* in the figure) installs any software and data he wishes to share. As displayed by the red crosses, other group members cannot yet launch virtual machines while the image is under preparation. Bullet 5 shows a fragment from the author's view in SHARE. This view provides an overview of all images to which the author has so-called *mutable* (and private) access. As shown in the figure, the author can decide to finalize the image by publishing it as an *immutable* image. Thereafter, it is visible to the peer group members. Alternatively, the image can be discarded and the author can restart the workflow. As shown by bullet 6, we assume here that the author publishes the image. All evaluators (e.g., *e1* and *e2* in the figure) as well as the author him- or herself (*d* in the figure) can now start virtual machines for this image, without changing the shared image or seeing each other's changes. This corresponds to the reader perspective, as discussed in Section 2.

5. Description of a Showcase Machine from the Model Transformation Domain

Several "executable papers" have already been prepared using SHARE. A representative example that highlights potential features of articles provided via SHARE is presented at a companion website for this article:

`http://sites.google.com/site/executablepaper/`

5.1. Background and Motivation

The example SHARE virtual machine[2] contains a solution for the BPMN-to-BPEL case [2, 3, 4] of the Transformation Tool Contest 2009. A discussion of this solution has been submitted after the contest to a special issue of Elsevier's "Journal of Visual Languages and Computing" and has been published in the meanwhile [5]. Therefore, it is possible to directly compare the conventional article with its executable counterpart.

The article under consideration actually is a literate program [6] and, thus, directly executable. The LaTeX files contain the complete program source code already. The text of the article is realized using comments of the respective programming language, here Curry [7, 8]. Unfortunately, the executability property of the literate program has been completely lost in the course of publishing. On Elsevier's sciencedirect website only the resulting PDF file and an HTML version of the article are provided. In addition, the authors of the article have extracted all code in a separate file and submitted it as supplementary material. But the code is not connected with the article anymore. Moreover, it is difficult for a reader to run this code, because a Curry compiler would need to be installed. However, all available Curry compilers require the Linux operating system. So, only reviewers with a Linux system and knowledge of the make installation tool etc. could test the code, explore the result space and verify the performance data provided in the article.

5.2. Benefits of using SHARE

Using the SHARE platform, a really executable version of this article has been created. Here it serves as an example of the general usage of SHARE for the creation of reproducible and executable research papers. The following features have been realized in the provided machine:

- The Münster Curry Compiler MCC[3] has been installed, so that the program code can be executed.
- LaTeX has been installed to compile the article into a PDF file.
- Gnuplot[4] has been installed in order to freshly generate the performance graph.
- Help files that describe the possibilities of the machine, e.g. how the transformation can be invoked or how further input data can be constructed.

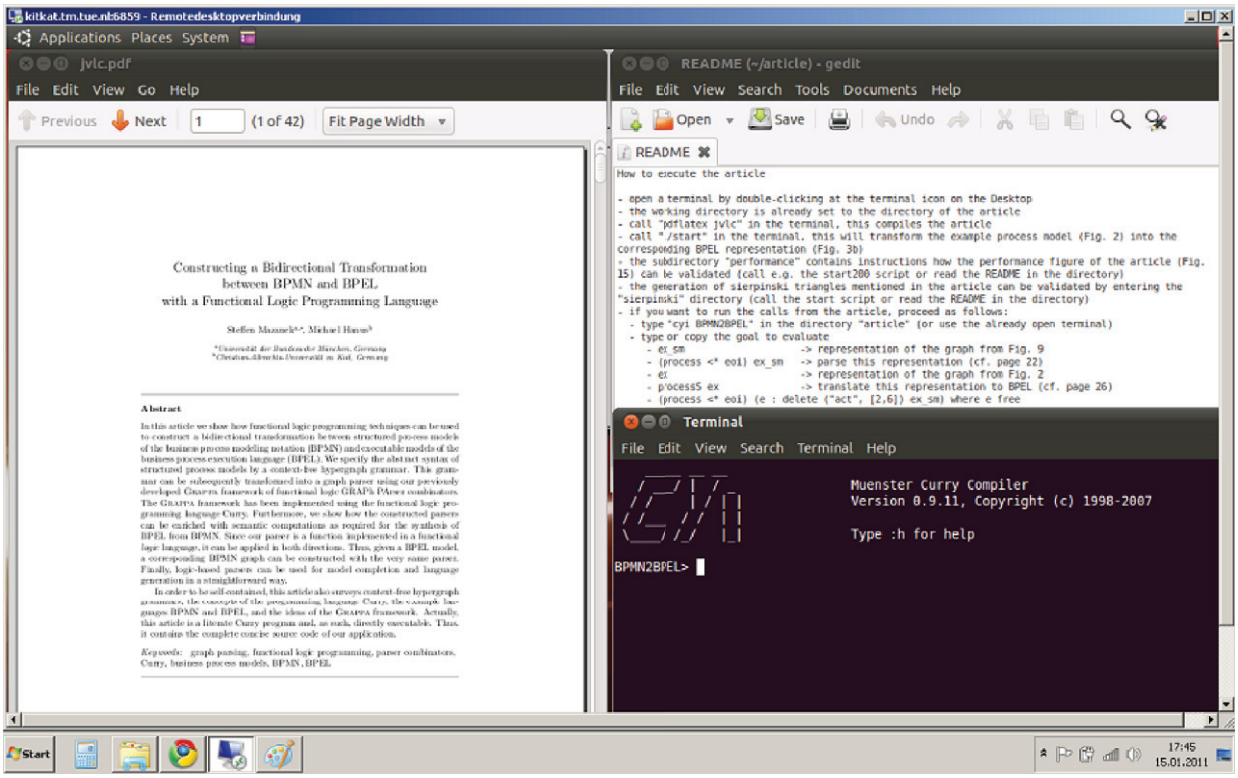

Figure 3: Screenshot of example machine

It would also be possible to install a BPMN editor in order to create input for the transformation in a more convenient way or even a BPEL platform for the direct execution of the output processes. The possibilities are nearly unlimited.

Figure 3 shows a screenshot of the example machine. It can be seen that the client operating system is Windows, in contrast to the virtual machine, which runs on Linux. As already mentioned, SHARE also supports the other way around. Several startup scripts have been created that open and arrange the required windows automatically after startup. It can be seen that the article document, a readme file and the Curry interpreter are started that way. Figure 4 shows the generation and plot of live performance data. The graph generated within the SHARE machine obviously will be different from the original one given in the paper. Actually, the SHARE machine is much more powerful than the author's PC — another benefit of moving such computations into the cloud.

The website accompanying this article also contains screencasts that show the invocation of the machine as well as the machine in action. At the end, the SHARE version of this article turns out to be much more useful than the published conventional article, because it is interactive and all results can readily be reproduced. We recommend to investigate this machine in order to get an impression of the power of the SHARE approach.

6. Adoptance and Related Work (Innovation over Current Options)

As discussed in Section 1, SHARE has emerged from reproducibility problems in the context of one specific research workshop/contest (TCC). In 2008, various alternatives have been evaluated, including the direct (i.e., local) use of virtualization software (i.e., VirtualBox[5]) by all stakeholders. Instead of accessing hosted virtual machines re-

[2] http://is.tm.tue.nl/staff/pvgorp/share/?page=ConfigureNewSession&vdi=Ubuntu-11.04_bpmn2bpel-grappa.vdi
[3] http://danae.uni-muenster.de/~lux/curry/
[4] http://www.gnuplot.info/
[5] http://www.virtualbox.org/

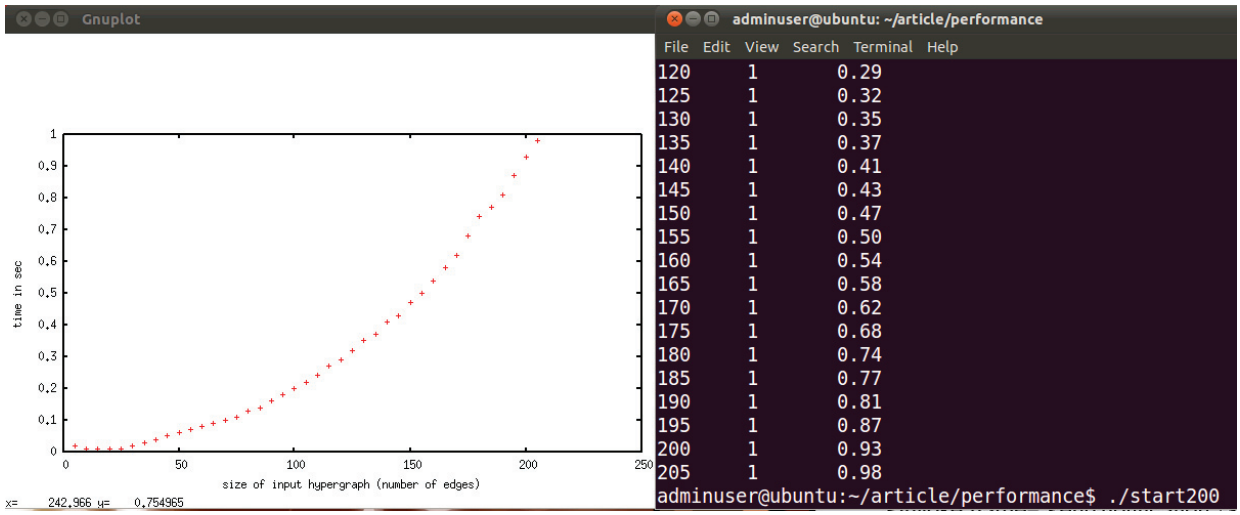

Figure 4: Generation and visualization of live performance data

motely, workshop participants were producing and sharing virtual machine images directly. This has lead to numerous interoperability problems, since participants often applied the virtualization software in different ways unconsciously. Another disadvantage of directly sharing virtual machine images is the clear conflict with license restrictions. For example, it is evident that one should not offer a virtualized Windows machine for download, even in a research setting. Finally, basic virtualization software does not offer SHARE's load balancing support, which has lead to unfortunate problems during a workshop in 2008.

Since then, various related cloud computing platforms have been applied in academia, but mainly in *e-Science*. For example, Keahey et al. use *Nimbus* to create virtual clusters dynamically. As a running example, the authors dynamically aggregate computational resources from three different universities to satisfy the service level agreements (SLAs) for heavyweight biomedical computations [9]. SHARE is quite different from Nimbus, as it does not focus on executing one particular type of computations efficiently on a cluster. While Keahey et al. aim at configuring one set of machines optimally, SHARE balances the load of running all kinds of virtual machines across four university networks in a transparent way. To reduce the dependency on hypervisor specific virtual network configuration details, each SHARE virtual machine is based on just one image. Moreover, all images are based on the same file format. This simplifies the migration procedure for the case that all hypervisor vendors might discontinue support for SHARE's image format (*.VDI* at the time of writing).

At the time of writing, more than 100 heterogenous images have been contributed by different research communities. Most images originate from the model and graph transformation communities but SHARE is also used in the Business Process Management (BPM) community and agreements have been made for evaluating SHARE in the Geosciences. The latter cooperation is supported by the shared data center of the federation of the three Dutch technical universities[6]. Also, other competitive workshops start adopting SHARE. For instance, the new language workbench competition[7] will rely on the SHARE infrastructure.

Eucalyptus [10] and the Grid Virtualization Engine (GVE [11]) are open source cloud computing platforms that are similar to Nimbus but that put more emphasis on the use of *standard* web service technologies such as WSDL and BPEL [1]. Amazon provides a commercial cloud computing platform, called *EC2*[8]. EC2 virtual machines are called *elastic*, since Amazon customers can dynamically change the amount of physical server resources that are allocated to their running virtual machines. Nimbus, Eucalyptus and EC2 are similar to SHARE in that these platforms deal with the metadata of virtual machine images. These projects have a different background though: Nimbus and Eucalyptus

[6] http://www.3tu.nl/
[7] http://www.languageworkbenches.net/
[8] http://aws.amazon.com/ec2/

have an e-Science background and EC2 has a hosting background. Therefore, these platforms have been designed for executing computational jobs on a pool of long-running virtual machines. Typically, multiple machines are started automatically (behind the scenes) to support scientific workflows (or production web servers, etc.). The user (a researcher or an e-commerce site visitor) typically does not have direct access to the underlying virtual machine sessions since these sessions tend to be stateful and tend to serve other users too.

In contrast, the SHARE platform has a *Reproducible Research* background and has thus been designed for (1) direct, and properly isolated, access to short-running virtual machine sessions that have been started explicitly by the user and (2) for image sharing. As a result, Nimbus, Eucalyptus and EC2 have a more sophisticated API for *monitoring* runtime virtual machine performance (e.g., to check computational SLAs) whereas SHARE has a more specialized user interface for ad-hoc image *evaluation*, *cloning* and *dissemination*.

Seminal work related to executable papers has been contributed by Claerbout et al. in the early nineties [12]. They already applied automatic build tools to produce CD-ROM images that contained the research article, the corresponding TeX source, related code and data, scripts to rebuild certain figures from the article automatically, and even a special purpose TeX viewer to trigger these scripts while reading the article. Actually, SHARE makes it possible to create a virtual machine with the full content of these CD-ROMs. All of the Claerbout-based executable papers, thus, can be made permanently reproducible inside SHARE. [1] also surveys more recent work that implements Claerbout's ideas[9]. All in all, the most important advantage of SHARE over many other approaches that try to make research reproducible is its flexibility.

7. Conclusion

In this article we have proposed SHARE as a means to make research articles executable. Following the SHARE approach, all artifacts of an article, programs as well as data, are installed into a virtual machine, which can be made immutable afterwards and published that way. On reader's demand and triggered by a single mouse click, a clone of the machine is created that waits for inspection. The reader can connect to that machine using a remote desktop tool as delivered with most modern operating systems.

The article also has described the different roles and workflows how bundles of machines can be created that are related to a scientific workshop or conference. The potential and concrete benefits of SHARE machines have been described along an example machine corresponding to a conventional research article.

Fulfillment of the Challenge's Criteria

- Executability: SHARE machines can be used very flexibly, so that, among others, interactive equations, tables and graphs are possible and the particular experiment can be repeated and manipulated.
- Short and long-term compatibility: SHARE's only bottleneck with regards to durability is the hypervisor of its underlying virtualization software. Currently, SHARE is built on top of Oracle's (previously Sun Microsystems's) academic version of VirtualBox. But even in the event of discontinuation of that software, SHARE's layered architecture supports long-term availability.
- Validation by reviewers: Opening the environment is just one click, provided the user is logged into the system already. This simplifies the reviewing and the validation of the data and the code.
- Copyright/licensing: SHARE enables authors already to restrict the time that their contributed virtual machine image is used per session. Readers can upload files (e.g., test data) to remote virtual machines. However, they can never download artifacts to their local computers. So far, SHARE has only been used in an academic setting. If Elsevier aims to make SHARE virtual machines also available to industrial readers, then a special purpose license needs to be developed and agreements with large software vendors have to be made [1].
- Systems: Images are replicated across virtual machine servers. Changes in the infrastructure generally are hidden from end-users. In the domain of high-performance computing, one can make the hardware available as SHARE virtual machine server(s) and restrict the number of concurrent sessions on such servers.

[9] See for example `http://www.reproducibility.org`.

- Size: The SHARE approach saves reviewers/readers from downloading huge files. Moreover, disk usage can be optimized on the server side: large data sets are typically mounted on special network drives that can be read by multiple virtual machines.
- Provenance: SHARE stores information about virtual machine sessions as well as the clone relations between virtual machine images. Such information can be used to perform impact analysis. As discussed in [1], one could install in SHARE virtual machines existing software for tracking events (keybord and mouse actions) to provide more detailed provenance functionality.
- Project quality: SHARE has been stress-tested by the participants of numerous transformation tool contests. At these events dozens of machines run in parallel. The machines containing the submitted solutions are reviewed before as well as during the contest. New machines are created for all solutions of the workshop's live contest and evaluated afterwards. The feedback of the participants regarding SHARE has been very good so far.
- Scope: The aim of the SHARE project is to provide a mature portal for making papers *executable*. Advanced metadata functionality is out of scope, but integration with specialized existing solutions is not. SHARE already provides integration with *LiquidJournal*[10], a platform that enables authors as well as volume editors to combine all artifacts related to a research paper into one integrated online publication that can be analyzed for citations, etc. A similar integration can be built for other bibliographic tools in a straightforward way.
- Feasibility of integration in publishing workflow and scalability: SHARE's distribution of administrative tasks across multiple organizers is key to the scalability from a publisher's perspective.

8. References

[1] P. Van Gorp, P. Grefen, Supporting the internet-based evaluation of research software with cloud infrastructure, Software and Systems Modeling (2010) 1–18doi:10.1007/s10270-010-0163-y.
[2] Object Management Group, Business Process Modeling Notation (BPMN), http://www.omg.org/spec/BPMN/1.2/ (2009).
[3] OASIS, Web Services Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf (2007).
[4] M. Dumas, Case study: BPMN to BPEL model transformation, http://is.ieis.tue.nl/staff/pvgorp/events/grabats2009/cases/grabats2009synthesis.pdf (2009).
[5] S. Mazanek, M. Hanus, Constructing a bidirectional transformation between BPMN and BPEL with a functional logic programming language, Journal of Visual Languages & Computing 22 (1) (2011) 66 – 89, special Issue on Visual Languages and Logic. doi:DOI:10.1016/j.jvlc.2010.11.005.
[6] D. E. Knuth, Literate programming, The Computer Journal 27 (2) (1984) 97–111.
[7] M. Hanus, Curry: An Integrated Functional Logic Language (Version 0.8.2), http://www.curry-language.org/ (2006).
[8] S. Antoy, M. Hanus, Functional logic programming, Communications of the ACM 53 (4) (2010) 74–85.
[9] K. Keahey, M. Tsugawa, A. Matsunaga, J. Fortes, Sky computing, IEEE Internet Computing 13 (5) (2009) 43–51. doi:http://doi.ieeecomputersociety.org/10.1109/MIC.2009.94.
[10] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov, The eucalyptus open-source cloud-computing system, in: Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid, CCGRID'09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 124–131. doi:http://dx.doi.org/10.1109/CCGRID.2009.93.
URL http://open.eucalyptus.com/documents/ccgrid2009.pdf
[11] L. Wang, G. von Laszewski, J. Tao, M. Kunze, Grid virtualization engine: design, implementation and evaluation, IEEE Systems Journal 3 (4) (2009) 477–488. doi:http://10.1109/JSYST.2009.2028589.
[12] J. Claerbout, Electronic documents give reproducible research a new meaning, in: Proc. Ann. Int. Mtg Soc. of Expl. Geophys., 1992, pp. 601–604.

[10] http://project.liquidpub.org/research-areas/liquid-journal.