2012 10th IAPR International Workshop on Document Analysis Systems

The Non-Geek's Guide to the DAE Platform

Bart Lamiroy
Université de Lorraine – LORIA
Nancy, France
Email: Bart.Lamiroy@loria.fr

Daniel Lopresti
Lehigh University – CSE Department
Bethlehem, PA 18055, USA
Email: lopresti@cse.lehigh.edu

***Abstract*—The Document Analysis and Exploitation platform is a sophisticated technical environment that consists of a repository containing document images, implementations of document analysis algorithms, and the results of these algorithms when applied to data in the repository. The use of a web-services model makes it possible to set up document analysis pipelines that form the basis for reproducible protocols. Since the platform keeps track of all intermediate results, it becomes an information resource for the analysis of experimental data.**

This paper provides a tutorial on how to get started using the platform. It covers the technical details needed to overcome the initial hurdles and have a productive experience with DAE.

***Keywords*-benchmark; web services; document analysis; performance evaluation**

I. Introduction and Context

The Document Analysis and Exploitation platform aims to support experimental research through an open and evolving framework that can be extended by community-driven contributions. Previous accounts of the features and the potential uses of the platform [4], [5], [7] focused on its architecture, implementation choices, and its potential to impact experimental research, especially with respect to reproducibility and accountability.

In what follows, we focus on how a potential user can have a productive experience with the DAE platform, giving a step-by-step description and explaining how to avoid technical difficulties. Beyond the basic features of the platform, we also provide insights as to its design philosophy so that interested users can contemplate contributing to its further development. Although our examples show how these features may be used in an integrated way, it should be understood that they can also be employed separately as a task demands. As much as possible, the DAE platform avoids coercing users into changing the way they do their research. Even so, integrating and using all of the platform's features amplifies the benefits they provide.

Since development of the platform is currently ongoing and certain details are likely to change over time, an updated version of this tutorial is maintained at http://dae.cse.lehigh.edu/DAE/Tutorial. The online version contains more detailed illustrations, download links, and step-by-step walkthroughs.

To illustrate our tutorial, we revisit the case of Jane, a young researcher who is working on specific task: given a page image, identify regions that contain handwritten notations (*cf.* [7]). She has developed a method that takes any page as input, making the assumption that the handwriting, if present, covers only a relatively small portion of the page. For the purposes of our fictional story, let us say that Jane has been working in isolation, and is just now discovering that others are examining similar problems elsewhere.

Jane has developed her algorithm using *JavLab++'95* a specialized programming language that runs only under X-WindOS 7.1. Her recent literature review has revealed publications by competing researchers, but their software is not available for her operating system, and she is not proficient in porting source code developed in languages other than the one she knows. As a consequence, Jane is stuck, unable to compare her algorithm to the existing state of the art. That is, until she hears about an international competition [5], [6] and an associated paradigm purporting to address this very issue.

II. Algorithms as Web Services

Jane's main problem now is that the competition paradigm relies on *Web Services*, a concept she is unfamiliar with and is afraid she has no time to learn. But since the description in [5], [6] emphasizes its ease of use, she decides to go and take a look at http://tinyurl.com/DAE-WebServices.

A. Principles of Web Services

Jane quickly discovers that, although web services are a rich, cutting-edge technology [1], she only needs to know that they provide a standardized framework for remote execution of algorithms (called *services*) and that invoking these services, or piping output from one into another, can be done independent of the underlying infrastructure, operating system, and even the data format in certain cases.

B. Web-Service Execution and Taverna Workbench

Consulting the list of published algorithms available as web services on the DAE server at http://dae.cse.lehigh.edu/DAE/services/soap, Jane discovers the NCI Segmentation algorithm she had already heard about [2], and she decides to see how it behaves on the

978-0-7695-4661-2/12 $26.00 © 2012 IEEE
DOI 10.1109/DAS.2012.87

training images she has been using for her own method. Because she has no experience developing code that uses web services, she follows the simple procedure provided by the DAE Web Service Tutorial:

1) Download and install Taverna Workbench [8].
2) Download the execution workflow for her algorithm from http://tinyurl.com/DAE-Workflows.
3) Make a training image available over the Internet by uploading it to a publicly available URL.

All she has to do next is to load the workflow into Taverna, provide her DAE username/password and the URL of the image she wants to process, and press *run*. She has already registered herself (for free) on the DAE platform beforehand. The workflow engine then executes and provides her with a set of URLs on the DAE server containing her results.

Jane is impressed by the simplicity of the approach. She is aware, however, that this was just an easy first test. How would this scale to running such services on a large set of images? She certainly would not want to process them one by one, clicking her way through the Taverna interface each time as she has just done. She also wonders about running more complex combinations of services, and having them interact with her own, local algorithms.

In the following sections, we describe Jane's journey to finding answers to these questions, providing URLs along the way that link to more detailed tutorials, examples, and code fragments that help explain the features of the DAE platform in a hands-on way.

C. Data Exchanges with the DAE Server

In the previous example, Jane provided her own data and was required to upload it to a publicly available URL so that the web service could access it. As we shall see (*cf.* Sections III-B and IV-C), the DAE platform hosts large amounts of pre-existing data that can be accessed through appropriately constructed URLs or queries. These URLs can also be provided by web services, as, for instance, the ones used in the `DatasetExtractor` workflow, available from http://tinyurl.com/DAE-Workflows.

Jane decides to try this approach. Using the Taverna interface, she creates a new, empty workflow, and uses "right-click + Nested Workflow" in the design worksheet of the interface to include the `NCI Segmentation` workflow she had previously tested, as well as the `DatasetExtractor`. The editor's drag and drop features let her easily connect the `ImageURLList` from the latter to the `page_image` input of the former.

Having done so, she has now piped the output of the `DatasetExtractor` to the page segmentation service. The extractor workflow requires two input parameters: a dataset ID and the number of output images the user wants to extract from the dataset. The service then takes care of randomly selecting the documents from the dataset and feeding them, one by one, into the next stage. Jane does not have to indicate that iteration is required. The only information she lacks is how to find the dataset ID – she will learn this later in Section III-B – but for now, let us say she chooses to use dataset ID 360466.

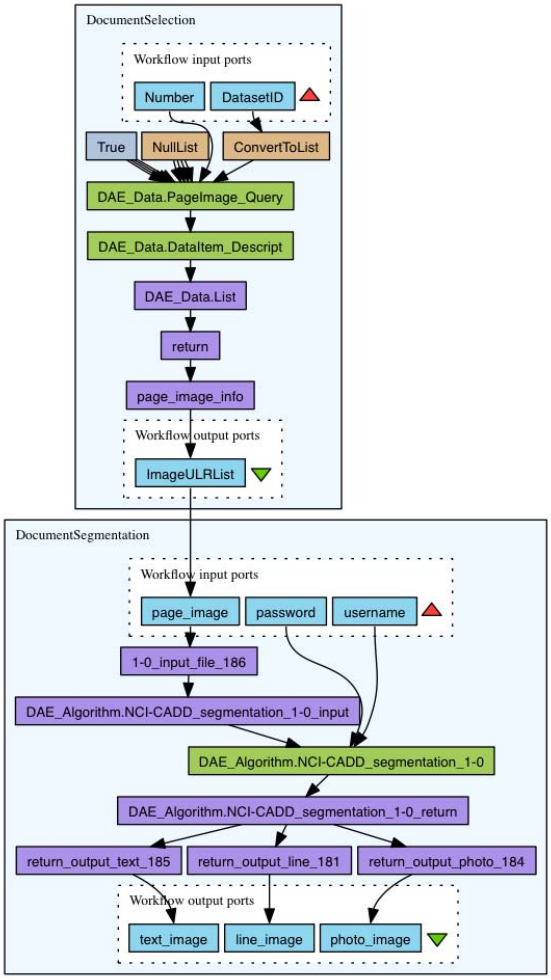

Figure 1. Jane's final worflow (http://tinyurl.com/DAE-JaneFinal).

At this point, we are assuming that all web services are hosted on the DAE server. In Section IV-A, it shall become clear this is not necessary, and it is perfectly reasonable to build workflows using web services outside of the platform. Similarly, DAE web services may be of no interest to users who only wish to access the data hosted on the server.

III. Data Repository

Jane, encouraged by her first positive experience, browses the DAE server to determine what kinds of datasets it hosts. She finds two interesting sources of information: http://dae.cse.lehigh.edu/DAE/browse lists the major document collections on the server, while http://tinyurl.com/DAE-Data explains how these collections

are structured as individually accessible units.

A. General Overview of Data Model

As already described in [4], the DAE platform stores more than just basic collections of documents. It supports a data model that, from a higher level viewpoint, incorporates concepts and relations summarized by the following set of axioms:

- Stored information relates to *persons*, *algorithms* and *data items*.
- *Persons* run *algorithms* by providing input *data items*.
- *Data items* are either contributed by *persons* or produced by *algorithms*.
- Primary *data items* are *page images* (scanned document images or transformations of those images), *page elements* (regions within images) and *property values* (interpretations of those regions).
- Any *data item* can be given any number user defined type values.

A complete entity-relation model is available at http://dae.cse.lehigh.edu/Design/ER.pdf.

B. Accessing Data – Basic Scenario

Jane starts by exploring the "Browse Data" section on the server (http://dae.cse.lehigh.edu/DAE/browse). Since she is not yet convinced the DAE environment will help her in her work, she decides to browse the system before she considers investing more time learning how to use it.

She is presented with a list of dataset names and document thumbnails, statistics, and some possible interactions:

- A "Download" link allows her to immediately download the entire data set.
- The right-hand side of the listing informs her of a star rating other DAE users have assigned to this dataset, and the number of interactions (views or downloads) the platform has registered for it over the last month.

The upper part of the listing provides her with alternative presentation and sorting options. Jane can choose to display the default main datasets, intermediate sets, individual page images, or a mix of these. She can order items by date of their addition to the repository, their frequency of use, or their star rating. She is also given a search box for finding specific tags or dataset names (more advances querying will be explained in Section IV-C).

Clicking on one of the dataset names takes Jane to a detailed browsing window in which she discovers the underlying dataset hierarchy, the files included in the dataset, and, most importantly, all *page images* it contains. She also notices that, when clicking on the page image thumbnails, she can access even more details: a magnifying glass icon provides access to all *page elements* of a *page image* as well as their *property values* such as OCR transcriptions or ground truth labels.

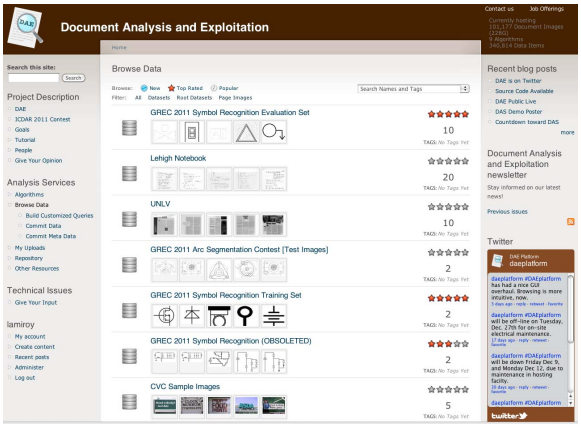

Figure 2. DAE browser (http://dae.cse.lehigh.edu/DAE/?q=browse).

While poking around the various data items, Jane notices that their URLs follow a basic naming scheme. Every *data item* is identified by a unique ID and can be accessed for viewing through http://dae.cse.lehigh.edu/DAE/browse/dataitem/<ID> and downloaded from .../DAE/browse/dataitem/download/<ID>. Jane verifies this by providing some of the page image links for the Tobacco800 datata set (ID 119774) to the web services she had been experimenting with. She now also has a means for applying her algorithm to DAE data on her own local machine through a simple download process.

C. Providing Data – Basic Scenario

Jane finds herself more and more fascinated when she realizes that much of the power of the DAE platform arises from contributions by the research community. She was able to run other people's code without having to go through the trouble of adapting it to her environment, and she could use and download contributed datasets. She is intrigued when she discovers that, while the algorithms she tested obtained good results on the hosted datasets, they behaved less well on her personal training date. She suspects her algorithm may provide a significant improvement over the state of the art, and decides to publish her training set on the server.

Jane creates an archive of her data, together with an appropriate copyright declaration, and uploads everything to the DAE platform via http://dae.cse.lehigh.edu/DAE/browse/upload. The platform automatically uncompresses the archive and creates all necessary entries in its database to make it available, and also generates a webpage to document it. She can now publish her experimental results by simply referencing the URL of her dataset. She can even provide specific copyright information that will be automatically attached to her data when others use it.

At this point we have given an overview of Jane's initial experiences with essential features of the DAE platform:

its hosting of data and algorithms. Their use required no particular technical knowledge, and was accessible via a couple of mouse clicks. In what follows, we follow Jane after this first experience triggers a desire to dig deeper.

IV. More Advanced Uses

After having conduced her first experiments using the basic features of the platform, and reading [5] and http://dae.cse.lehigh.edu/ICDAR2011Contest, Jane concludes she can obtain a significant benefit by modeling her experiments as DAE workflows testing various algorithms on identical datasets. Not only can she make the workflows available for others to run and verify, but the fact that the platform stores all intermediate results makes it possible to backtrack and compare experiments at any point in time.

A. Creating Web-Services

Jane's first step is to convert her *JavLab++'95* code, running on X-WindOS 7.1, into a web service. She is concerned that her lack of knowledge may be a problem, especially since there seem to be no public WSDL libraries or APIs available for JavLab++'95. Fortunately, X-WindOS 7.1 supports running PHP in a web server (having a web server running PHP is convenient, but not absolutely necessary so long as Jane's local machine allows network connections).

The instructions at http://tinyurl.com/DAE-WebServices explain that Jane needs to:

- Download a set of pre-configured PHP files and personalize them to her local environment.
- Make sure her code runs from the command line, without requiring user interaction once started.

Jane's command line code looks like this when run:

```
HWDetect -s5 image.pgm hwimage.pgm
```

It expects a standard pen stroke width to be 5 pixels in `image.pgm`, and produces `hwimage.pgm` as output containing the detected handwriting.

The PHP code she has to fill out in the template files she has downloaded looks like this:

```
$algoname = 'JHWDetector';
$algoversion = '1';
$algopath = 'D:\Jane\HWDetect';

$inputT['input_file'] =
array('name'=>'input_file',
'type'=>'xsd:string');
$inputT['stroke_width'] =
array('name'=>'stroke_width',
'type'=>'xsd:integer');
$outputT['output_file'] =
array('name'=>'output_file',
'type'=>'xsd:string');

$localOutput['output_file'] =
realpath($localdir) .  '/output.pgm';

$execString = "$algopath -s".
$inputValues['stroke_width'].' '.
$inputValues['input_file'].' '.
$localOutput['output_file'];
```

Saved in the appropriate directory, she can import this as a new service in her Taverna interface by providing its location, http://localhost/wsdl/JHWDetector.1.php?soap.

We note here that Jane has just converted her code to a web service for use by herself only. Indeed, she has installed a web server on her local computer and accesses the service on her `localhost` network interface. This is, however, already quite an achievement for her. She may have legitimate reasons for not making her program public (corporate policy, licensing issues, or simply because she feels her code is not yet ready for "prime time"). On the other hand, she can now seamlessly integrate her program with workflows downloaded from the DAE repository, creating experimental setups to test her own method and the published results of others, exactly the same way she had done earlier.

Later, when Jane feels the time is right, she can publish her program and her workflows, either on a public website or on the DAE server.

B. Publishing DAE-Aware Web-Services

When Jane is ready to make her algorithm available through the DAE platform as a web service, she goes to http://dae.cse.lehigh.edu/DAE/algorithms/modify, fills out the associated form (mainly requesting the same basic information as provided in the PHP file), and uploads any of the following three items:

- The public URL where she maintains her web service.
- A binary version of her program, provided it runs on one of the OS environments available on the server.
- A minimalist VirtualBoxed [11] environment running both the OS and her program.

The latter option is the one Jane chooses, since the DAE server does not support X-WindOS 7.1. Her software will then undergo a human review that determines whether it can be hosted on the platform. As mentioned briefly in [5], some DAE-hosted web services run in virtualized Windows XP and Linux Debian environments.

Emphasizing the community-focus of the paradigm, in Section V we show how Jane can even go further by hosting her own DAE platform integrated with others on the Internet.

A significant benefit for Jane and the research community in registering and running her algorithm on the DAE platform is the ability to conduct *a posteriori* analyses of her experiments at any point in the future. This raises new research opportunities in traceable and reproducible performance evaluation, as was demonstrated in practice during competitions at ICDAR 2011 [5].

C. Accessing Data Using Queries

The previous arguments in favor of publishing web services on the platform brings us to the topic of more

flexible access to stored data than the simple approach described in Section III-B. Given the generality of the DAE data model, exploiting its full power requires SQL technical skills and knowledge of the entity-relation schema supporting the platform. Although DAE allows direct SQL querying through http://dae.cse.lehigh.edu/DAE/Repository and an associated `Repository` web service, this level of database sophistication is at odds with the kind of user we are imagining Jane to be.

Users like Jane can instead get started with simplified queries that directly map to families of URLs. For instance, .../DAE/Repository/query/<X>/<Y>/<Z> is read as `select id from X where Y like Z;` and is restricted to the retrieval of data item IDs.

The platform provides an interactive query constructor at http://dae.cse.lehigh.edu/DAE/browse/query that is more flexible, but this still only permits the retrieval of page images. Its main advantage is that the constructed queries map directly to parametrized URLs that can be used in existing code. A screen snapshot is shown in Fig. 3.

Figure 3. Query Generator showing a query for images in the Tobacco800 dataset containing an identified signature (here: *Gravely, L.E*).

D. Providing Data Interpretations

Space constraints prevent us from demonstrating how Jane can add her own interpretations (so-called "ground truth") to page images and regions, but the same general philosophy espoused by DAE applies here as well; the server provides an easy-to-use interactive interface for creating manual markup, as well as an automated approach for uploading more complex interpretations. More information can be found by browsing through the UNLV or Tobacco800 datasets which contain such annotations, and by reading http://tinyurl.com/DAE-MetaData. Fig. 4 shows an example of the interface. Details on how to convert standard ground-truth formats can be found at http://tinyurl.com/DAE-Convert.

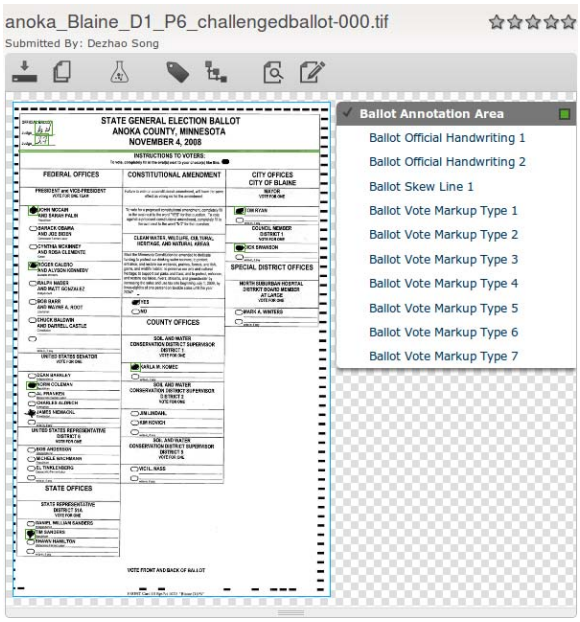

Figure 4. Example of page image markup showing interpretations for an op-scan ballot from the Minnesota Challenged Ballots dataset at http://dae.cse.lehigh.edu/DAE/browse/dataitem/45981.

V. Power User Scenarios

We have demonstrated throughout this paper that the DAE platform offers multiple levels of interaction. The examples provided are backed by active links to actual implementations and readers can follow Jane step-by-step to acquire hands-on experience with DAE. In doing so, we have hopefully made the case that the platform is conceived to handle a level of complexity and completeness on the one hand, while offering a relatively smooth learning curve on the other.

Jane's story does not have to end here, however. She has had the opportunity to create her own web service and upload her experimental data to the platform, and, in return, benefit from community-contributed data and resources. She has become a believer in the DAE paradigm, but ...

A. The Current DAE Server Does Not Suit Me

Jane finds that some of the existing DAE features do not fit her needs. Perhaps she stumbled upon a bug, or key information that is missing, or maybe she sees an

opportunity to add something completely new to the system, *e.g.*, a tool for creating context-constrained crowd-sourced annotations.

Since the entire DAE framework is freely available on http://sourceforge.net/projects/daeplatform/, nothing prevents Jane from enhancing the code and offering her contributions back for the benefit of other researchers like herself. In doing so, she becomes a productive and valued member of the international research community.

B. I Want My Own Personal DAE Server

Jane may wish to maintain a greater degree of control over her data than would be allowed by uploading it to a third party platform, even though she still wants to make it available to the larger community. She may also feel that her institution deserves credit beyond which she would receive through a mention on an existing server. In this case, she can set up and host her own instance of a DAE server. (At the time of this writing, the authors are aware of two other DAE installations now running which are not public.) Ongoing work is looking at ways to seamlessly handle provenance queries across multiple instances of DAE.

VI. Related Work

There has long been interest in performance evaluation and shared datasets in the field of pattern recognition. In some sense, DAE is the culmination of decades of efforts by individuals as well as organizations such as the IAPR's TC-10 ("Graphics Recognition") and TC-11 ("Reading Systems") [3]. There are far too many to mention individually, but for an excellent survey we refer the interested reader to the wonderful keynote delivered by George Nagy at the 2010 DAS workshop [9]. Indeed, it was this talk that prompted us to note how little things have changed over the past 50 years and provided one of the "sparks" for DAE.

In terms of recent integrated activities that are similar in philosophy to DAE, we can point to the growth in formal competitions at international conferences, as noted previously, as well as a number of efforts to build and promote websites for networking scientific communities, *e.g.*, the VIVO effort [10].

VII. Discussion

The platform reported here is currently still under development; its value to the community hinges on the degree to which it is adopted, used, and extended. Looking beyond the current DAE, future work could attempt to leverage interoperability with other initiatives such as MyExperiment (http://www.myexperiment.org) and Impact (http://www.impact-project.eu/), as well as other large document repositories. This also means that the scalability of the approach, its distribution over multiple sites, and its reliability and redundancy would need to be studied. Future research might also examine means for efficiently querying and filtering the available data with respect to specific experimental contexts.

These and other currently open questions present opportunities for investigation by the document analysis community as we strive to enhance the interconnections between our shared research goals.

Acknowledgment

Daniel Lopresti acknowledges support from a DARPA IPTO grant administered by Raytheon BBN Technologies.

References

[1] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[2] I. V. Filippov and M. C. Nicklaus. Optical structure recognition software to recover chemical information: Osra, an open source solution. *Journal of Chemical Information and Modeling*, 49(3):740–743, 2009.

[3] International Association for Pattern Recognition. http://http://www.iapr.org/.

[4] B. Lamiroy and D. Lopresti. A platform for storing, visualizing, and interpreting collections of noisy documents. In *Proceedings of the Fourth Workshop on Analytics for Noisy Unstructured Text Data*, pages 11–18, Toronto, Canada, October 2010.

[5] B. Lamiroy and D. Lopresti. An open architecture for end-to-end document analysis benchmarking. In *Proceedings of the Eleventh International Conference on Document Analysis and Recognition*, pages 42–47, Beijing, China, September 2011.

[6] B. Lamiroy, D. Lopresti, and T. Sun. Document analysis algorithm contributions in end-to-end applications: Report on the ICDAR 2011 contest. In *Proceedings of the Eleventh International Conference on Document Analysis and Recognition*, Beijing, China, September 2011.

[7] D. Lopresti and B. Lamiroy. Document analysis research in the Year 2021. In *Proceedings of the Twenty-Fourth International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 264–274, Syracuse, NY, USA, July 2011.

[8] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. A. Goble. Taverna, reloaded. In M. Gertz and B. Ludäscher, editors, *Proceedings of the Twenty-Second International Conference on Scientific and Statistical Database Management*, pages 471–481, Heidelberg, Germany, June-July 2010.

[9] G. Nagy. Document systems analysis: Testing, testing, testing. In *Proceedings of the Ninth IAPR International Workshop on Document Analysis Systems*, page 1, Boston, MA, USA, June 2010. http://cubs.buffalo.edu/DAS2010/GN_testing_DAS_10.pdf.

[10] VIVO: an interdisciplinary national network. http://vivoweb.org/.

[11] J. Watson. Virtualbox: bits and bytes masquerading as machines. *Linux Journal*, February 2008.