Semantic Modeling for Cloud Computing, Part I

Amit Sheth and Ajith Ranabahu • Wright State University

Cloud computing has lately become the attention grabber in both academia and industry. The promise of seemingly unlimited, readily available utility-type computing has opened many doors previously considered difficult, if not impossible, to open. The cloud-computing landscape, however, is still evolving, and we must overcome many challenges to foster widespread adoption of clouds.

The main challenge is interoperability. Numerous vendors have introduced paradigms and services, making the cloud landscape diverse and heterogeneous. Just as in the computer hardware industry’s early days, when each vendor made and marketed its own version of (incompatible) computer equipment, clouds are diverse and vendor-locked. Although many efforts are under way to standardize clouds’ important technical aspects, notably from the US National Institute of Standards and Technology (NIST), consolidation and standardization are still far from reality. In this two-part article, we discuss how a little bit of semantics can help address clouds’ key interoperability and portability issues.

Cloud-Related Challenges

Figure 1 shows the three main flavors of clouds as outlined by NIST (http://csrc.nist.gov/groups/SNS/cloud-computing). Infrastructure-as-a-Service (IaaS) clouds have the largest gap (a high workload but little automation) in terms of deploying and managing an application. Platform-as-a-Service (PaaS) or software-as-a-service (SaaS) clouds have substantially lower workloads, but at the expense of flexibility and portability.

Given this diverse environment, a cloud service consumer faces four challenges.

First, depending on the application’s requirements, legal issues, and other possible considerations, the consumer must select a cloud to use.

Each cloud vendor exposes these details in different formats and at different granularity levels.

Second, the consumer must learn about the vendor’s technical aspects (service interface, scaling configuration, and so on) and workflow.

Third, the consumer must then develop an application or customize the vendor-provided multitenant application to fulfill his or her requirements. When doing this, the consumer must take into account various technical details such as the choice of programming language and limitations in the application runtime, which will all be vendor-specific.

Finally, after deploying the application, if the consumer must change the service provider (which happens surprisingly often), at least two major considerations arise. First, the consumer might need to rewrite or modify the application code to suit the new provider’s environment. For some clouds (such as IaaS), this is minimal, but porting the code in PaaS and SaaS clouds will likely require more effort.

The second consideration is that data collected for the application might need transformation. Data is the most important asset the application generates over time and is essential for continued functioning. The transformation might even need to carry across different data models. The industry practice is to address such transformations case-by-case.

To overcome these challenges and provide better insight into the aspects requiring attention, proper modeling in this space is essential. Semantic modeling can help with this.

Cloud Interoperability?

First, a word about cloud interoperability is in order. Interoperability requirements in the cloud landscape in Figure 1 arise owing to two types of heterogeneities.
The first is vertical heterogeneity — that is, within a single silo. We can address this by using middleware to homogenize the API and sometimes by enforcing standardization. For example, the Open Virtualization Format (OVF; www.dmtf.org/vman) is an emerging standard that allows migration of virtual-machine snapshots across IaaS clouds.

The second type is horizontal heterogeneity — that is, across silos. Overcoming this is fundamentally more difficult. Each silo provides different abstraction levels and services. High-level modeling pays off, especially when you must move an application and code horizontally across these silos.

Surprisingly, many small and medium businesses make horizontal transitions. PaaS clouds offer faster setup for applications, and many exploit the free hosting opportunities of some platform cloud providers (for example, Google’s App Engine). When the application grows in scope and criticality, however, an IaaS cloud might prove cheaper, more flexible, and more reliable, prompting a transition.

The following discussion on semantic models applies to both vertical and horizontal interoperability. The key in addressing both types is that many of the core data and services causing heterogeneities follow the same semantic concepts. For example, almost all IaaS clouds follow conceptually similar workflows when allocating resources, although the actual service implementations and tools differ significantly. Similarly, the PaaS modeling space is a subset of that for IaaS, from a semantic perspective.

These observations prompt us to argue for semantic models’ applicability, especially to supplement interoperability in the cloud space. Some parts of the scientific and engineering community weren’t impressed by early semantic-modeling approaches, especially ones that required large up-front investment. This perception, however, is changing rapidly with the influx of new applications and technologies that exploit detailed semantic models to provide improved functionality (for example, biomedical ontologies cataloged at the US National Center for Biomedical Ontology; www.bioontology.org). Semantic models such as ontologies can formalize more details than other traditional modeling techniques. They also enable reasoning, a way to make inferences and gain new knowledge. Such rich models can improve traditional functions far beyond what was once thought possible. For example, Powerset (now part of Microsoft; www.powerset.com), a search service provider, added considerable value to search results by incorporating semantic models and thus enabling fact discovery. These capabilities are being complemented by the ability to more rapidly create domain models, often by mining crowd knowledge represented, for example, in Wikipedia or shared data, exemplified by the linked-object data cloud. The technologies showcased at the Semantic Technology Conference (www.semantic-conference.com) also provide plenty of evidence of semantic-empowered commercial and scientific applications.

**Multidimensional Analysis of Cloud-Modeling Requirements**

We suggest a 3D slicing of the modeling requirements along the following dimensions (see Figure 2).

The types of semantics that are useful for porting or interoperability in cloud computing are similar to those we introduced in 2003 for Web services. This is natural because the primary means of interacting with a cloud environment is through Web services. The four types of semantics — data, functional, nonfunctional, and system — are based on the different semantic aspects a model must cover.

The **language abstraction level** indicates the modeling’s granularity and specificity. Although ontological modeling is preferable at a higher level, developers prefer detailed, concrete syntactic representations. These representations of different granularities might need to be related, often through explicit annotations. For example, although service developers happily use Web Services Description Language (WSDL) descriptions in their Web services, these
descriptions are syntactic and can’t provide useful semantic details. To overcome this deficiency, SAWSDL (Semantic Annotations for WSDL and XML Schema; www.w3.org/TR/sawSDL) attaches semantic-model details to WSDL documents.

The software lifecycle stage is important in determining the modeling requirements. For example, some nonfunctional and system requirements might not be modeled during deployment but will be taken into account only during deployment. A different team handles each of these lifecycle stages; this separation is important so that one team doesn’t step on another’s toes. This separation aims to focus the modeling effort on the correct time and people.

Some cloud models fall under the nonfunctional/system/ontology space in Figure 2. Such models include the Elastic Computing Modeling Language (ECML), Elastic Deployment Modeling Language (EDML), and Elastic Management Modeling Language (EMML), all based on OWL and published by Elastra (www.elastra.com/technology/languages).

However, some aspects of cloud modeling have received little or no attention. For example, there’s no comprehensive higher-level modeling in the data and functional spaces. Lessons learned during large-scale ontological modeling in the Semantic Web and Semantic Web services, biology, and many other domains are readily applicable here and would help address some of the challenges in the cloud space.

In part 2, we’ll look at opportunities for semantic models in cloud computing. Particular areas in which these models can help are functional portability, data modeling, and service enrichment.

References

Amit Sheth is the director of Kno.e.sis – the Ohio Center of Excellence on Knowledge-Enabled Human-Centered Computing at Wright State University. He’s also the university’s LexisNexis Ohio Eminent Scholar and an IEEE Fellow. He’s on the Web at http://knoesis.org/amit.

Ajith Ranabahu is pursuing a PhD in cloud-computing interoperability at Wright State University. He worked with IBM on Sharable Code and its Altocumulus project, and he coordinates the Cirrocumulus project (http://knoesis.wright.edu/research/srl/projects/cirrocumulus). Contact him at ajith.ranabahu@gmail.com.

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.

build your career
IN COMPUTING

www.computer.org/buildyourcareer

MAY/JUNE 2010

83