Service-oriented Architectures as a Vehicle for ICT in developing Countries: An Awareness Campaign

Agnes F. N. Lumala, Benjamin Kanagwa, José Ghislain Quenum and Jude T. Lubega

Service-oriented architecture (SOA) is one of the ways to build applications today. Indeed current applications of SOA type are an aggregation of several smaller applications. Such applications are synonymous with Web 2.0. The cost of consuming and using existing SOA services is much lower than establishing a new infrastructure to provide the same services and there exists free and open source services. Such a technology is appropriate for developing countries where technical and financial resources to setup and support ICTs are severely constrained. Therefore we organized a contest to create awareness about SOA as alternative application development strategy and to expose university students in Uganda to SOA technologies. In this paper we discuss the SOA contest and we give lessons learned from the SOA awareness as a whole.

1 Introduction

The last decade has witnessed a significant shift from traditional software architectures to service-oriented architecture (SOA). Thanks to the Internet revolution, services developed on one site could be assembled on a different site and could be accessed on a third one. Indeed current applications of SOA type are an aggregation of several smaller applications. Such applications are synonymous with Web 2.0.

Due to the tremendous progress in software development brought in by SOA, one can flexibly and transparently combine a book order and its shipment, a flight ticket to a hotel reservation and even a car reservation at the destination. The list goes on and on.

Moreover, the cost of consuming and using existing SOA services is much lower than establishing a new infrastructure to provide the same services and there exists free and open source services. By using SOA technologies the cost of the service, security, maintenance is greatly reduced. However you need to be aware of the cost and take care of basic technological requirements before you enjoy benefits of the technology. For example the Amazon storage service costs $0.180 per GB for the first 50 TB / month of storage used. We therefore see SOA technologies as an important approach to support ICT use in the developing world.

Given the popularity gained by SOA in recent years, Makerere University Faculty of Computing and Information Technology (FCIT) organized a contest about SOA implementation. This was to create awareness and exposure to SOA among our
students. We offered three applications (see section 2) and requested students to find data from/for any of these applications using Yahoo! Pipes and Amazon S3. The target participants were masters and bachelors students (with a background in information technology) from different universities in Uganda. However, only students from Makerere University and Mukono University expressed interest and participated in the contest. The winner of the contest and the first runner up were to receive a prize of a simple mobile handset. All the contestants would have a chance to attend lectures for two weeks by a SOA specialist who was coming in after the contest. Besides, the contest organizers offered them technical support as they developed the different applications. The contest organizers developed the backend of the applications involved, and the students were expected to develop the front end of the applications involved. Before discussing more about the soa awareness, it is important to introduce some basic technologies that are necessary for building a soa-based application.

1.1 SOA Technologies

There are several tools and technologies available to support the implementation of SOA. They focus on the core tenets of SOA, that is to describe, identify and consume the services. In web services, Web service description language (WSDL) is designed to describe services in way that they can be used by others. WSDL is a web service standard by W3C for describing the syntactical structure of a services. More specifically, the WSDL specifies the operation names and parameters. Details about WSDL can be found in [Chinnici, et. al., 2007, Curbera, et. al., 2002].

To implement SOA there is a need to transfer data across systems. SOAP (Simple Object Oriented Access Protocol) is one of the widely used technologies to package data into envelopes. In web services, an envelope contains information about the service to be invoked and the information required. The SOAP specification can be found in [Box, et. al., 2003] and usage in relation to other technologies is illustrated in [Curbera, et. al., 2002].

Another technology is REST (REpresentational State Transfer). Using REST an application can send requests and responses using HTTP (Hyper Text Transfer Protocol) verbs such as GET, POST, PUT and DELETE. In RESTful terminology a web service is called a resource. These resources can be located using a Uniform Resource Identifier (URI). Refer to [Fielding, 2000] for more details on REST.

The Web feeds provide simple means for dynamic update of information on the web. A web feed presents data in format (usually XML) that can be accessed directly by interested users. Through a web feed, you provide a mini database containing headlines and descriptions of what’s new on your site. RSS (Really Simple Syndication or Rich Site Summary) is one of the many formats that belongs to the family of web feeds. Interested users can then link to the URI for the feeds. With this simple Web feed technique, on top of displaying your news on other sites and headline viewers, RSS data can flow into other products and services like PDA’s, cell phones, and many others. Some tools such as Yahoo! pipes provides means of aggregating feeds. Together with
Atom, they are the leading technologies about feeds publishing on the Internet. Despite their widely admitted limitations we adopted RSS here for the sake of simplicity.

Amazon Web Services (AWS): Amazon is well known for a vast experience in offering services especially for their online store. They decided to extend this experience to web application developers and in July 2002 they launched AWS. The aim was to provide a set of infrastructure services that together form a reliable, scalable, and inexpensive computing platform “in the cloud”. AWS is a collection of web services offered by Amazon.com over the internet. These services include:

- **Amazon Elastic Compute Cloud (Amazon EC2)**, a web service that provides resizable compute capacity in the cloud.
- **Amazon SimpleDB**, a web service providing the core database functions of data indexing and querying.
- **Amazon Simple Storage (Amazon S3)** is storage for the Internet.
- **Amazon CloudFront**, a web service for content delivery.
- **Amazon Simple Queue Service (Amazon SQS)** which offers a reliable, highly scalable, hosted queue for storing messages as they travel between computers.
- **AWS Premium Support** which is a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.

They are all designed to make web-scale computing easier for developers. These services can be accessed over **HTTP** using **REST** and **SOAP** interfaces. AWS offered on a per usage billing basis. For example it costs $0.180 per GB for the first 50 TB / month of storage used. In order to utilize these services, you need to sign up with Amazon giving credit card information before you can access the services. For purposes of this contest we demonstrate the use of **Amazon S3** to avail a digital learning repository **DLR** which can be used to build e-learning applications.

**Amazon Simple Storage Service (Amazon S3)**

This service provides unlimited storage on the internet through a simple web service interface. Storage is organized in terms of buckets which contain objects. A user may own up to 100. In any bucket, you may write, read, and delete objects from 1 byte to 5 gigabytes in size with accompanying metadata. There is no fixed limit on the number of objects you can store. Buckets and objects can be created, listed, and retrieved using either a REST-style HTTP interface or a SOAP interface. Currently most ftp clients support manipulation of Amazon S3 buckets. For the DLR we used the GUI-based **bucketexploer** and **S3 Fox** extension of Firefox. The three concepts are buckets, objects and keys.

**Buckets** A bucket is simply a container for objects stored in Amazon S3. Every object is contained within a bucket. For example, if the object named homeindex.html is stored in the courseA bucket, then it is addressable using the URL http://courseA.s3.amazonaws.com/homeindex.xhtml. Buckets serve several purposes: they organize the Amazon S3 namespace at the highest level, they identify the
account responsible for storage and data transfer charges, they play a role in access control, and they serve as the unit of aggregation for usage reporting.

**Objects** Objects consist of object data and metadata. The data portion is opaque to Amazon S3. The metadata is a set of name-value pairs that describe the object. These include some default metadata such as the date last modified, and standard HTTP metadata such as Content-Type. The developer can also specify custom metadata at the time the object is stored.

**Keys** A key is the unique identifier for an object within a bucket. Every object in a bucket has exactly one key. Since a bucket and key together uniquely identify each object, Amazon S3 can be thought of as a basic data map between «bucket + key» and the object itself. Every object in Amazon S3 can be uniquely addressed through the combination of the Service endpoint, bucket name, and key, as in http://courseA.s3.amazonaws.com/homeindex.xhtml, where «courseA» is the name of the bucket, and «homeindex.xhtml» is the key.

Amazon provides technical documentation about the storage service.

**Yahoo! Pipes**

In mid 2006, in order to cope with the growing interest in social networking and Web services, Yahoo! released Yahoo! Pipes. Yahoo! Pipes is a web application that makes it easier to find information in an RSS feed. We use one of the applications for the contest (see section 2) for *Cinnamon*, a virtual reading group web application to demonstrate Yahoo! Pipes.

The remainder of this paper is organized as follows. In section 2 we describe the applications and tools used stating the role of the students for each of the applications. Section 3 discusses how the contest was organized. Section 4 gives the challenges we faced during the SOA awareness and section 5 concludes the paper.

# 2 Applications and tools used

For the purpose of the contest, we introduced two widely adopted tools: *Yahoo! Pipes* and *Amazon S3* (part of the AWS family). The reader should recall an overview of these tools in section 1.1. We developed three applications for purposes of the contest. Other than utilization of SOA technologies for their own sake, we targeted applications that could be developed in a relatively short time, interesting and easy to work with by the students, and extendable into fully fledged applications. In this section we describe each of the applications and we state the expected role of the students for each application.

## 2.1 Virtual Reading Group

In most research centers, a reading group is one where researchers share their recent readings and views about a topic of interest. In its usual setting, it is an active group where members meet periodically. In this application (*Cinnamon*), we intended to create a virtual one where users can post comments and questions about their latest readings. Other users can contribute as well. The challenge here consisted of finding discussion tracks following different criteria.
2.1.1 Basic functionality of Cinnamon

*Cinnamon* is a WebApp to support discussions in a research. Generally, members of a research group have to share their ideas and insight about research topics. *Cinnamon* creates a virtual community to support this kind of activity. *Cinnamon* has been developed using two open source Web development frameworks, *Sproutcore* and *Ruby on Rails* also known as *Rails*. *Sproutcore* (Cocoa) is the Objective-C based programming environment for Mac OS.-inspired javascript framework for creating Web applications that look and feel like Desktop applications (see AppleInsider blog). On the other hand, *Rails* is a fully fledged framework to develop webapp using the Ruby programming language. Both frameworks support MVC (model-view-controller), AJAX, etc.

As depicted in figure 1, a user willing to use application should first log into the system. New users can join by signing up, using the same page. They just need to check the signup button as shown in figure 2. Note that the email address is validated against the usual email address pattern.

**Figure 1: Overview of Cinnamon: Login Phase**
After logging or signing up successfully an interface as depicted in figure 3 appears.

This interface allows to do one of the following:

- Create new topics or delete existing ones. Note that a topic is a concept discussions in a research group revolve around. Topics can be searched for using the search text field and button in the upper right side of the page. When the topic is found it is automatically selected and only post related to that topic will appear.
• For a selected topic, create, modify or delete entries. An entry is a post. It is considered as an entry point in a discussion track. An entry has a title and a content. It can be backed by references (authors, title of the resource, event where it appeared or was delivered, etc.).

• For each entry, participants of a research group can add comments.

• Besides, a user can also update their account information by clicking on the Account button at the bottom of the page. They find out about new version (and any other news about Cinnamon) by clicking the About button. Finally, they can generate RSS feeds, a summary of the activity going on in the research group.

2.1.2 Feeds Generation

To generate an RSS feed, a user simply clicks on the RSS Feeds button. Sproutcore then sends a (RESTful) AJAX request to the Rails server. This triggers the generation of the feeds in “public/feeds/RSS_feed.xml” on the Rails side. Originally, the structure we would like for our feeds is depicted as follows.

```xml
<!- we do not display feeds header here ->
<posts>
  <post title="" author="" topic="">
    <article>entry content</article>
    <comments>
      <comment author="">comment</comment>
      <comment author="">comment</comment>...
    </comments>
  </post>...
</posts>
```

However, because we had to comply with RSS specifications, we flattened the feeds to the following model.

```xml
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <item>
      <title>News for September the Second</title>
      <link>http://example.com/2002/09/01</link>
      <description>other things happened today</description>
    </item>
    <item>
      <title>News for September the First</title>
      <link>http://example.com/2002/09/02</link>
    </item>
  </channel>
</rss>
```
In the following section, we introduce *Yahoo! Pipes* using the simplified version of our feeds.

### 2.1.3 Yahoo! Pipes and Cinnamon

Yahoo! Pipes offers the possibility to fetch information from different sources, compose them in a way suitable to one’s own needs and publish the output. The WebApp comes with an editor. You need a Yahoo! account to use the editor.

To create a pipe, you will use the pipe editor offered by Yahoo!. The editor consists of three elements: *Library*, *Canvas* and *Debugger*. The library contains a set of modules. A module performs a specific task, e.g., treating data sources, translating items from one language to another, etc. The canvas is the working area for assembling and testing the pipes. Finally, the debugger offers you the possibility to inspect the output of a pipe.

Figure 4 depicts an excerpt of the feeds generated in Cinnamon.

**Figure 4: Cinnamon: RSS Feeds**
To create a pipe displaying information about the discussion tracks in Cinnamon, you can follow the steps as discussed below.

- select the *Fetch* module and enter the URL of Cinnamon RSS feeds. As you may have guessed, this module reads RSS feeds from the URL provided.
- to filter the information gathered from Cinnamon, you can add a filter. The latter defines rules to block or permit feeds related to keywords entered in the appropriate text field.
- finally, you can connect the output of the filter to the input of the output pipe. The connections are similar to composing services: input, output and constraints on the possible composition.
- you can now save the pipe and add it to your Yahoo! page. Note that search engines can search your pipes. Therefore, we recommend you add properties to your pipe that facilitate the search. These are name, description and tags.

### 2.2 E-Notice Board

In this application, there is an electronic board, where notices are displayed for some time. A notice can be a textual information (pdf, doc, etc), a sound or a video. A user can register for some type of notice and be notified once they are available on the board. The challenge in this contest is to find different notices, which are interrelated given a topic-name or keyword and which are still temporarily valid. The task of the participants was limited to extracting the RSS feeds to combine them into a single notice board like application. The fundamental principle of service-orientation is seen in terms of the autonomous nature of the feeds and ability to compose the feeds. The composition of simple RSS feeds provides a good basis to explain two concepts. First, it explains the concept of composing services into more complex services. Second, it explains the concept of exposing a service as a resource for use.

Further, the fact that a single feed can be used by different notice board applications puts emphasis on the concept and advantages of reuse. Reusability is one of the fundamental concepts in service-orientation. Moreover, the subscribe-push concept of feeds is good basis to introduce the need for advanced forms of registry and discovery techniques including semantic capabilities to enable full automation of services. More details on the need for semantic capabilities can be found in [McIlraith and Son, 2002, Fensel et al., 2002]. In figure 5, we show the e-notice board resources to be extracted and composed into a notice board application with form factor as in the bottom right part of the figure.
2.3 Digital Learning Repository (DLR)

DLR is a digital repository which provides services to users seeking specific learning contents of their choice. It accomplishes the latter by acquiring contents deposited into a repository built on an ontological understanding of the learning object (LO) concept [Hamel and Ryan-Jones, 2002, Weitl et al., 2004]. Content developers therefore design content to suit the LO with varying sizes, formats, quality and educational values. The learning objects used for the contest are XHTML files. We used Amazon S3 to store the different learning objects. A bucket contains all the learning objects related to a single subject. These learning objects can be accessed using URLs. The challenge was to find the exact and appropriate content for e-learning for both a tutor and a student.

3 SOA Contest Organization

We started by developing the backend of all the applications. The students role would then be to come up with the front end of a chosen application. The students were allowed to work in groups.

The reader should remember that Cinnamon was developed using Rails and Sproutcore web application development frameworks. For the e-Notice Board, the resources were provided statically using standard RSS format. For DLR, we created the learning objects as XHTML files and we employed the Amazon S3 service for storage. The Amazon S3 service provides a way of accessing services using technologies and techniques that are commonly used in web services.

We created a website www.cit.ac.ug/soato provide detailed information about the contest. It is through this website that contestants registered for the contest, got
documentation about the different tools, technologies and applications related to the contest. Here they also got technical support from the organizers.

The contestants were given a duration of 1 month to work with one of the applications. The contestants were allowed to upload the most current versions of their applications. This allowed us to monitor the progress of the contestants, thereby providing necessary technical support.

4 Challenges

In this section we highlight challenges faced during the SOA awareness contest. We divide the challenges into technical challenges, social and economic challenges.

4.1 Technical Challenges

The following challenges that we met are common in developing countries. However, in order to benefit fully from the use of SOA technologies for application development, it is important that they are resolved.

i. **Internet Services**: It was difficult to work with technologies that employ the internet with very low bandwidth available in most universities. For example the amazon infrastructure is a purely web-based infrastructure which requires good internet. The internet services were intermittent. Therefore completion of long internet transactions was almost impossible. In order to carry out the contest we had to pay up in internet cafes to complete some of the online transactions.

ii. **Connectivity**: Remote connection to and through the local network was difficult. This restricted participants from working off the university site. For example it was not possible to connect using Secure Shell (SSH). We only managed to resolve this with Makerere university which allowed temporary remote connection for the sake of the contest.

iii. **Physical infrastructure**: Some of the applications required the use of a server and it was challenging accessing one. A server allocated for use during the contest but it was done late and this lengthened the time set for the contest. 

   **Note**: The use of some SOA technologies requires skilled manpower. Therefore it is important that skilled manpower is available before planning for use of SOA technologies for application development.

4.2 Social and Economic Challenges

i. **Choice of applications**: The first challenge was to craft suitable applications for students with less exposure to Information Technology (IT). For instance students from the School of Industrial Fine Art expressed interest, but it was not easy to explain an advanced IT concept (SOA) due to low IT essential skills. This led us to restrict the awareness to students with a background in IT.

ii. **Standardization**: For some technologies the way to consume and use services differed from popular techniques in web services. Therefore more time was used to understand the same concepts.
iii. **Collaborative communication:** The organization was mainly virtual because of the holiday season. However in a developing environment, working virtually is met with unpredictable situations for example power outages, low bandwidth, lack of remote access and so forth. Therefore it was a challenge to utilize virtual communication to coordinate project activities.

iv. **Publicity:** SOA being a new concept in the contest environment, it was challenging to encourage students to participate in the contest. We used prizes to attract participants to the contest.

5 Conclusion

There are several services that are offered via the web (SOA) that can be appropriately utilized for ICT in a developing world. The services are both available as proprietary and open source for application development. In this paper we discussed a contest conducted to serve as an awareness campaign for SOA technologies as an alternative to application development. The contest targeted students from all universities in Uganda. However, only students from Makerere university and Mukono University showed interest and participated. In future we would like to find out why the other students did not respond at all to such a campaign. At the end of the contest, the participants were more knowledgeable about SOA in general and SOA technologies in particular. This they demonstrated when they presented their work for evaluation as part of the contest. From the contest, it was clear that despite the various challenges (intermittent internet especially) that are inherently faced by Uganda as a developing country, training in SOA technologies can be done to build enough skill for future use for application development.

The reader should note that a development environment for SOA-based applications requires various components for service request, service discovery and selection, service composition and deployment. However, the discussion of these components is beyond the scope of this paper.

Acknowledgements

We wish to acknowledge Amazon.com which allowed us to use AmazonS3 for the contest without paying. We also acknowledge Mr. Maxwell Omwenga who developed and maintained the SOA awareness website.

References


