I. SHORT USE CASE DESCRIPTION

In order for one to test an adaptive admission control and resource allocation algorithm, it is necessary to set up an appropriate testbed of a distributed web application like RUBiS benchmark [1], an auction site prototype modelled after eBay.com. It provides a virtualized distributed application that consists of three components, a web server, an application server, a database and its workload generator, which produces the appropriate requests. Furthermore it can be deployed in a virtualized environment using Xen server technology, which allows regulating system resources such as CPU usage and memory, and provides also a monitoring tool, Ganglia, that measures network metrics, such as round trip time and other statistics, and resource usage in virtual machines.

The adaptive admission control and resource allocation algorithm is applied to succeed in specific target of network metrics, like round trip time and throughput. This will be done by deploying a proxy-like control component for admission control and using Xen server technology to regulate CPU usage. During this scenario the adaptive admission control and resource allocation algorithm is tested against network metrics, like round trip time and throughput. RUBiS clients will produce requests so that push RUBiS components to their limits, so that resource like CPU usage and network throughout get high values.

During an example setup, a researcher wants to test an http proxy software written in C programming language that implements an admission algorithm. To carry out the experiment the researcher requests some http traffic generators and some http web servers as resources. The scenario presented can be easily scaled up with many clients and web applications. Also, the proxy under test can be replaced by one or more load balancers, making the case suitable for both academic and industrial usages. The scenario is depicted in Figure 1 and requires as resources three traffic generators (GenA, GenB and GenC), two web servers (WebA and WebB) and a virtual machine (VM) that will host the proxy algorithm.

A. Technical environment

Figure 2 displays the equipment and the connections used during the scenario. The equipment used is as follows:

- Linux machines for the RUBiS based work load generators
- A Linux machine for the hosting the algorithm unit, capable of compiling C and Java software
- Linux machines for running XEN server where on top will run the RUBiS Web app and database

The final user needs to provide the algorithm under test. He will just login to the Proxy Unit, compile the software and execute it.
The tools to deploy, monitor and run the experiments are those offered by the Panlab architecture [2]. A Web Portal is available where customers and providers can access services, a visual Creation Environment which is called “Virtual Customer Testbed (VCT) tool” where a customer can define requested services, a repository which keeps all persistent information like resources, partners, defined federation scenarios, etc. Experimenters can browse through the resource registry content and can select, configure, deploy and access reserved resources. The above components interact with each other in order to offer a service called “Teagle”. Part of Teagle is also the Teagle Gateway, the component that is responsible for transferring provisioning and configuration commands to selected resources lying in various administrative domains.

Some additional components for integrating testbeds that belong to various administrative domains, are the following: the Panlab Testbed Manager (PTM) which is responsible for configuring the domain’s resources. PTM implements the so called Resource Adaptation Layer where Panlab partners “plug-in” their Resource Adapters (RA). A Resource Adapter (a concept similar to device drivers) wraps a domain’s resource API in order to create a homogeneous API defined by Panlab.

Federation Computing Interface (FCI) is an API for accessing resources of the federation. It is an SDK for developing applications that access VCT requested resources through the Panlab office services during operation of testing. It is quite easy to embed it into your application/SUT in order to gain control of the requested resources during testing. The FCI is delivered to the customer after the generation of the SLA and not only does it contain the necessary libraries but also the alias of the resources that are used in the VCT scenario. This allows the User-Application/SUT to access the testbed resources during execution of the experiment in order to manage and configure various environment parameters or to get status of the resources.

Details and specifications of Panlab’s components can be found at [2].

B. Results

First results of running such an experiment although not comparable currently with similar approaches are really encouraging in terms of moving the designed algorithms from simulating environments to near production environments.

Using the existing deployed RUBiS facility makes the setup and scaling up of such a testbed much easier. What is really attractive is that such algorithms can be tested in a best-effort environment with real connectivity issues that cannot be easily performed in simulation environments.

II. TEST BED AVAILABILITY

The resources for creating similar scenarios are going to be available under the Panlab Office offerings. Currently there are a limited amount of resources that are capable of hosting the RUBiS environment. We expect to make more resources available as demand increases.