

SLA QoS PRODUCTISED

DESCRIPTION OF DJRA1.5 DELIVERABLE

Document Filename:	BG-DJRA1.5-v.1.41
Activity:	JRA1
Partner(s):	KTH, EENET, IMCS UL, PSNC, VU
Lead Partner:	KTH
Document classification:	PUBLIC

Abstract:

The report describes the current state of the SLA QoS product in JRA1. The deliverable DJRA1.5 is a demonstrator and the report is proof of what has been done up to January 2008. It aims to be a help for administrators that are going to install it on their sites.



SLA QOS PRODUCTISED
Description of DJRA1.5 deliverable

Document review and moderation

	Name	Partner	Date	Signature
Released for moderation to	Norbert Meyer	PSNC	March 2008	
Approved for delivery by				

Document Log

Version	Date	Summary of changes	Author
1.0	31/01/2008	Initial version	Kristaps Dzonsons/KTH
1.1	3/02/2008	Second version updated according to the template and discussions at the JRA telecom and e-mail exchange	Kristaps Dzonsons /KTH
1.2	3/03/2008	Updates of the outcome of discussion with HP and further investigations	Kristaps Dzonsons /KTH
1.3	4/03/2008	Editorial changes	Norbert Meyer /PSNC
1.41	14/03/2008	Summary and references updated Editorial changes	Kristaps Dzonsons /KTH Norbert Meyer/PSNC



Contents

1	EXECUTIVE SUMMARY	4
2	OBJECTIVES	5
3	QOS SYSTEM.....	6
4	INSTALLATION AND CONFIGURATION GUIDE	7
5	TEST PROCEDURES	9
6	LIMITATIONS AND KNOWN PROBLEMS	10
7	SUMMARY	12



1 Executive Summary

This report describes the current state of the SLA QoS deployment within the joint research activity (JRA1) of the BalticGrid project. The selected QoS system is *Tycoon*[1], an advanced resource allocation and reserve system. The goals of this activity were to install, configure, and run a pilot installation of the *Tycoon* system. This goal has been met. Unfortunately, during the activity's duration, unforeseen changes in the system's design led it to be inadequate for usage as a QoS system within the BalticGrid.

The objectives of this activity were to research, deploy, and install an advanced reservation system for executing jobs on the BalticGrid. The specific purpose was to deploy a system capable of scheduling jobs with a wide range of requirements; the system would be responsible for choosing the most appropriate execution environment within a set of registered hosts and running the job with whatever resources were initially requested. The system would have to assume a volatile execution environment, provide a means of accounting, and accept environments ranging from simple tasks to interconnected MPI scientific applications. The system would have to run on GNU/Linux operating systems and be fully open source.

The *Tycoon* system is a resource allocation and reserve system. It was chosen by the BalticGrid for use as a QoS system due to its meeting the system requirements.

The general utility of *Tycoon* is to provide a simple means of allocating compute resources on a grid and subsequently executing jobs within the allocated resources. The *Tycoon* system is responsible for enacting the resource reservation, isolating the resources, and executing the job within a closed environment as dictated by those resources; the primary interest in *Tycoon* is that it uses market-driven economics in order to orchestrate this machinery. In using these algorithms, the *Tycoon* system is well-adapted to volatile grid environments with a surplus of users and deficit of resources.

Tycoon is under development at H.P. Labs[5] in the United States. It runs on the GNU/Linux operating system, primarily on Fedora Core[3] and RedHat[4] distributions (or those inheriting the packaging system developed by RedHat, RPM), and primarily uses the Xen multiplicity[6] system in order to orchestrate the scheduling and isolation mechanism for jobs.

Since the initial acceptance of *Tycoon*, the system has undergone considerable change. Unfortunately, changes to the system's availability, and specific questions regarding implementation, have cast significant doubt on the future applicability of the system to the BalticGrid. These changes were not anticipated in the original decision to accept the system for consideration. The initial indication of issues was raised in October, 2007. Since then, no conclusion has been made regarding the applicability of *Tycoon* to the BalticGrid: the matter is still under discussion.

The two primary issues with the current and legacy *Tycoon* system are of availability and quality. Releases during October and November of 2007 removed core components of the system from the public repository, effectively privatising parts of the system. Although the client source code is available and appropriately licensed, core "server-side" components are unavailable for public enquiry. This raises significant security and logistical concerns: not only may these core components not be audited for correctness and quality, but creating



private, inter-networked pilot installations is no longer possible due to the inability to privately deploy core server components. The BalticGrid pilot installation is able to function and be tested without this networking capability, but the system may not be put into use (i.e., with communication and resource sharing between hosts) without significant security concerns.

The second issue is one of quality and concerns those core components still available in October and November of 2007. Since then, these components have been removed from public access and thus these issues may or may not be resolved. This document describes several issues which have not been demonstrated as fixed despite the relevant core components being re-written and deployed outside the public view.

Preliminary results of this activity were presented at the **Fourth AHM Meeting** [7] in Stockholm, November 2007, although pilot installations had been prepared by the BalticGrid before then. This report describes the state of the software as of the most recent public release in January, 2008. Since the Fourth AHM Meeting, more core components were discovered as removed from the public repository, further highlighting the issue of availability. Communication with H.P. regarding these issues has been largely un-answered.

Despite the limitations of the current software, this activity has resulted in a pilot installation spanning a plurality of BalticGrid sites. The connection of these sites has been demonstrated in the documented tests, although these tests are restricted due to security concerns.

2 Objectives

The Joint Research Activity of BalticGrid project (JRA1) works on SLA QoS and will deploy these services in the BG in co-operation with SA1 and with support of SA2. In this document we focus on QoS (quality of service). The main objective of JRA1 concerning this task is as follows:

- to research, deploy, and fully document a SLA QoS product for use in the BalticGrid.

BalticGrid project focuses on extending EGEE infrastructure to the Baltic states, so that using EGEE software (gLite/LCG) is assumed. Another assumption is reusing existing solutions as much as possible. To fulfil these assumptions, the proposed systems are based on existing tools, previously developed by the project partners and the EGEE middleware.

The main aim of SLA QoS is to formalise request and servicing of compute resources within the BalticGrid. The scope of this activity was finding an appropriate product and deploying it on the BalticGrid. QoS software is an absolutely requirement in distributed computing, especially in grid computing, to manage the volatile pool of resources and resource requests. Although smaller clusters may be manually serviced, the scope of the BalticGrid requires special software to connect grid users with necessary resources. An enumeration of requirements for the BalticGrid QoS system follows:

1. The system must be open source and available to the BalticGrid without restriction
2. The system must run under the GNU/Linux operating system
3. The system must have a demonstrably quality implementation, including source code, documentation, and tests; and
4. The system must be under active development or maintenance.



An enumeration of features for the BalticGrid QoS follows:

1. The system must have a flexible description language for all aspects of task run-time, including but not limited to processor usage, disc space, input and output, and network usage
2. The system must have a simple utility for submission of jobs and a means of monitoring usage and accounting for users
3. Full documentation must be available for operating the system and all of its components, including administration and maintenance
4. The system must operate on a wide variety of hardware with demonstrable effectiveness; and
5. The system must have a provable testing utility to demonstrate the system's correctness in adverse environments.

In general, QoS systems are required for grids to accommodate volatile resources and a high demand for use, a demand most likely outstripping supply. The system must demonstrate all the properties of a well-design and robust utility: documentation, correct operation, etc.

The mile-stones set forth for the QoS activity were to install, configure, and deploy a pilot installation of the system. This pilot installation must be tested and working and demonstrate the system's correctness and effectiveness in preparation for production use. To date, these milestones have been met, given the limitations imposed by the current system design.

3 QoS System

The Tycoon system is a development of H.P. Laboratories. It was chosen for its applicability to those requirements set forth in the Objectives section of this document. Tycoon has been under development for several years, and has a strong set of peer-reviewed scientific publications to bolster its relevance.

"Tycoon is a market-based system for managing compute resources in distributed clusters like PlanetLab, the Grid, or a Utility Data Center (UDC). The basic idea is that users have a limited supply of credits. Consuming users pay providing users to use computer resource. Users who provide resources can, in turn, spend their earnings to use resources later.

This automated economic mechanisms allocates resources with more economic efficiency than standard proportional share and priority systems, and orders of magnitude more quickly than going through manual allocation.

Tycoon is currently a prototype implemented using Linux and Xen; a networked economy of hosts running the prototype; and a set of algorithms for doing economic resource allocation."

Although the Tycoon system meets the requirements set forth by the BalticGrid, its implementation is of particular interest. Unlike traditional resource allocation systems, which schedule jobs in a typical batch system or attempt to match requirements with environment



using fitting algorithms, the Tycoon system uses theories of market-driven economics in order to orchestrate reservation

In the Tycoon system, allocation is governed by supply (compute resources) and demand (user jobs). Supply is formalised by resource specifics (bandwidth, disc space, and so on). Users are allocated Tycoon currency with which they may bid on resources in competition with other users. These bids are, in theory, governed by the job requirements. Unlike in a traditional batch system, the currency-based bidding system encourages users to set resource requests only to the limit of their currency. This application of game theory encourages fair play within the system.

The core components of the Tycoon system are the bank, which governs transactions; the client, which acts on behalf of users to bid on resources; the auctioneer, which advertises its resource availability; and the SLS, which collects advertised resource availability and works with the bidder to coordinate resources.

In order for an installation of the Tycoon system to be complete, the SLS and bank must first be installed and available. Auctioneers must be installed on each compute host, and clients must be installed on each host that wishes to bid on resources. Following this, the system may be effectively used.

The process of Tycoon is fairly complex. What follows is an abridged version of the general allocation and bidding protocol between clients and compute hosts.

1. Resources are advertised to the SLS
2. Clients gather relevant compute hosts by querying the SLS given a job descriptor file
3. A client bids on an auctioneer's resources by putting the necessary funds in escrow on the bank
4. If the client's bid is rejected, the client seeks other hosts; or
5. If the client's bid is rejected, funds are transferred from the client into a bank escrow;
6. The client's job is executed on the selected compute hosts; and
7. If the job completes, the funds are transferred from escrow into the auctioneer's host; or
8. If the job does not complete, the funds are transferred back to the client.

The theory of the system is well-described in a variety of publications. The implementation spans approximately four years of development.

4 Installation and Configuration Guide

Introduction

The Tycoon system is responsible for managing compute resource brokerage on a Grid; furthermore, it manages the allocation, scheduling, and processing in order to fulfil resource requests. Tycoon interacts with Xen virtual machines in order to service resource requests. Typically, there are two Tycoon components, a *client* and an *auctioneer*; the remaining elements of Tycoon (bank and SLS) are run privately by H.P. A compute element, in Tycoon parlance, is an *auctioneer*, i.e., it auctions its resources, while a *client* is an entity (user or pre-programmed sequence) that bids on resources.

BalticGrid Tycoon installations use client and auctioneer systems; the remaining systems are not available to the public. This is discussed more fully later in this document.



Prerequisites

1. Installation of Computing Element operating system (Fedora Core x86-32).

Installation of Tycoon

1. Update system.
% yum -y update
2. Install all dependencies:
% yum -y install kernel-xen.i686 ntp.i386 mercurial.i386 \
python-twisted-core.i386 python-crypto.i386 atlas-devel.i386 \
glpk-devel.i386 pyOpenSSL.i386 strace.i386 jdk.i586 \
rssh.i386 vxargs.noarch xen-libs.i386 xen.i386
3. Disable tls library if not disabled.
% mv /lib/tls /lib/tls.disabled
4. Grub must boot xen kernel:
Default you must change file /etc/grub.conf entry from "default=1" to "default=0"
5. Reboot server to boot Xen kernel. To test after reboot is xen enabed
% xm list # It must show Name "Domain-0"
6. Install from tycoon packages
% rpm -ivh <http://tycoon.hpl.hp.com/~tycoon/dl/fedora/5/i386/cvxopt-0.7.1-1.i386.rpm>
7. Download source for tycoon from mercurial repository:
% cd /usr/src/
% hg clone <http://tycoon.hpl.hp.com/cgi-bin/hgwebdir.cgi/tycoon/KL>
% hg clone <http://tycoon.hpl.hp.com/cgi-bin/hgwebdir.cgi/tycoon/tycoon>
% ln -s /usr/src/KL /usr/src/tycoon/src/
8. Environment modifications:
% export PYTHONPATH=/usr/src/tycoon/src
% export JDKDIR=/usr/java/jdk1.6.0_03
9. Compile tycoon:
% cd /usr/src/tycoon/
% make rpm
% make sub_rpms
% make kl_rpm
10. Install tycoon server.
% cd /usr/src/tycoon
% rpm -ivh *.rpm
11. Install tycoon client.
% yum -y -c <http://tycoon.hpl.hp.com/~tycoon/dl/yum/tycoon.repo> \
install tycoon_client



12. To test tycoon:

% tycoon

Initial Configuration

There is no special configuration that must be performed; however, we suggest firewalling the installation hosts to prevent other Tycoon users from accessing the resources broadcast by the auctioneer. Alternately, one can change the SLS address to be the localhost as described in the Testing section of this document.

Further Information and Help

- Look at References section.
- People at BalticGrid with some Tycoon knowledge:
 - Kristaps Dzonsons <kristaps@kth.se>
 - Kaspars Kazarevskis <Kaspars.Kazarevskis@risinajumi.lv>

5 Test Procedures

Local Tycoon installations may be tested if both the auctioneer and client are installed. In order to run the auctioneer, some changes must be made to the installation. First, the SLS (Service Location Server) must be assigned to the local host, or the host's resources will be sent to the main H.P. SLS system.

Modify /etc/tycoon/tycoon.conf to have:

```
SLSHostName = localhost
```

This assumes that "localhost" resolves to the current machine. Next, start the Tycoon auctioneer daemon (N.B., the auctioneer will automatically start upon system start if the installation instructions in this document are followed):

```
% tycoon_aucd &
```

Lastly, use the client to connect to the auctioneer and test that its resources have been registered:

```
% tycoon host get_status localhost
```

This will print out the available resources. If this command emits host data, then it works; if it doesn't, and its invocation returns a non-zero value, then it has failed.

The breadth of these tests is severely limited by the fact that the SLS and bank are no longer available to the public, as described in the Limitations section of this document. By not having a local version of the bank and SLS system, advanced automated testing on the correctness of the system are impossible.

Furthermore, the Tycoon system itself does not have a native testing utility. Without such a utility, all tests are "eye-ball" tests, limited to analysis of the output from the client. This is a sub-standard methodology which may only be improved with availability of all core components (which would allow for "glue" test routines to be written) or close collaboration with the developers, an involvement spanning a potentially-indeterminate amount of time.



6 Limitations and Known Problems

During the activity, several considerable issues were discovered within the Tycoon system. These were not anticipated while selected a QoS system for the BalticGrid, and arose only during the close of the activity. A definitive conclusion as to the relevance of Tycoon is still under consideration due to the recent date of these changes.

In October, 2007, some major issues were discovered in the Tycoon bank system. These ranged from quality of code to theoretic concerns regarding the banking protocol and the integrity of managed capital. The latter issues were produced in a paper [8] for the Fourth AHM in Stockholm, and are described by the following excerpt:

"The Tycoon bank is the central component of Tycoon, a resource management system for distributed computing. Tycoon abstracts compute resources into an economic model of supply and demand; the bank manages the capital with which this model operates. The bank is not appropriated within Tycoon's machinery of market-driven economics: it may not be valued or devalued depending on its perceived integrity. The bank, in other words, must be trusted by all participating members or the entire mechanism is undermined. This in and by itself poses no conflict; however, the current design model of the bank lacks a facility for asserting account and capital integrity, allowing a bank under adversarial control to manipulate capital without detection. This is a serious issue: the bank, not being governed by the market, must have its own governance or the market's equilibrium may be compromised by an adversary. In this document, we propose that the current bank has an intractable protocol for arbitrating an accusation of adversarial agency: an adversary, in the general case, can *always* disprove accusations. We follow this analysis with a system design that provably demonstrates adversarial conditions, and moreover, is able to identify the involved parties."

This document, after formalising errors with the existing bank, continues to suggest an implementation of the Tycoon bank that supports provable authenticity and integrity of managed currency.

Upon notifying the Tycoon developers of these issues (in October, 2007), the fact of the bank's withdrawal from public access was first discovered. Subsequent communication to the Tycoon developers demonstrate that, although the bank itself may have changed its internal structure, the underlying issues as demonstrated remained unaddressed. These were verified by examination of the available client source code, which continues to have no facility for authenticity or integrity.

Before this paper and the research required to write it, many other issues were discovered in the existing Tycoon bank (as of October, this bank has since been deprecated in favour of the privately-developed H.P. bank). A summary of these issues follows:

- The system was vulnerable to timing attacks, where a user of appropriate privilege could trick the bank into cancelling or accepting selected escrow values



- The system was very vulnerable to corruption, being at least vulnerable to simple data corruption and at worst selective corruption of public key and account files, leading to adversarial control over the bank's assets
- The system had no defined path of trust regarding administration and maintenance; and
- Files were stored in plain text and passwords were stored and passed on the command-line to the tool itself.

These flaws were found before the bank was announced as no longer available for public access.

Both of these issues aren't as considerable as the issue of the bank availability itself. Since the bank defines a site's capital environment, having private banks is a necessity for closed pilot installations. By participating in a single bank environment, such as is the current design, capital is managed by H.P. without any means of integrity management or resolution of conflict. At the time, this was considered a limitation but not a serious one, as bank accounts are not available between users regardless the bank deployment location.

At the time, the BalticGrid continued with deploying a pilot installation after carefully analysing the effected situation and determining it acceptable.

After a pilot installation using the H.P. bank and all other local components failed, it was discovered that the SLS had also been revoked from public access. The Tycoon web-site does not reflect either of these changes. The SLS, in managing all grid resources, is an absolutely critical element of the system. In making the SLS global, sites lose the ability to run private Tycoon networks.

The closure of the SLS from public access has severely limited the effectiveness of Tycoon tests, and due to the security ramifications of exporting BalticGrid site data to the public at large, with the potential, as the system has poor account oversight, of BalticGrid resources being used by the general public, the future of the Tycoon system and the BalticGrid is under serious question.

A short enumeration of remaining issues with the Tycoon system follow.

- Lack of documentation. The Tycoon product has a complete lack of documentation distributed with the system, i.e., manuals. There are some manuals available on the system's web-site, but these are not distributed in the standard, acceptable format of a manual (e.g., a Unix roff-formatted manual page). The on-line documentation, furthermore, is severely lacking quality. Many sections are marked as "XXX", i.e., not at all done. Only the most cursory usage of the *Tycoon* system may be learned by reading the on-line documentation.
- Accountability. Given that both SLS and bank components are no longer viewable by the BalticGrid, both in the sense of source code and running systems, there is no way of accounting for the integrity of these systems. In other words, flaws in the system, or adversarial agency, could influence the contents of the SLS and bank without oversight by the BalticGrid.
- Working relationship. Although the Tycoon developers may be contacted with questions, they are under no obligation to respond to questions. Furthermore, previous questions have resulted in significant delays in response; some questions were never answered or avoided (mostly regarding the availability of now-private components).



This is an unacceptable scenario: the complexity of the system requires intricate source-level knowledge, which may only be met by internal developers. Lastly, the closure of the bank and SLS components was not reflected on the Tycoon web-site, which does not reflect well on the forthrightness of the development team regarding system quality.

- Quality of code. This section may not reflect (but likely does) the current version of the source base. Previous examination of critical elements of the sources revealed numerous obvious vulnerabilities and errors. In general, errors in one section of a source-base suggest that other areas are similarly affected.

These are all minor issues compared to the closure of the core SLS and bank components, or more specifically, just the SLS component. Although a private bank would have been temporary acceptable, a closed SLS is a major security concern.

7 Summary

Although a Tycoon pilot installation was successful and the SLA milestones were all fully met, the issues discovered during the activity have brought into question usage of the Tycoon system in future activities. Although most of the issues are recoverable, the lack of availability of some components severely limit the applicability of Tycoon to the BalticGrid in the short- and medium-term.

In terms of milestones, the specific goals were met, as set forth in the deliverables: research, deploy and fully document a QoS system for the BalticGrid

Specifically, a system was researched and decided upon, and a pilot installation was deployed within the BalticGrid. The sequence of the research and deployment were fully documented.

The system chosen for this activity was Tycoon, developed by H.P. Laboratories. Tycoon is a compute resource reservation and scheduling system with a focus on market-driven economics as the underlying premise of the system. Tycoon has several years of development time, and available parts of the code-base are licensed under the GPL (General Public License). The system runs on specific releases of Fedora Core or Redhat GNU/Linux.

Since the decision to use the Tycoon systems, design changes to the system and availability of the system have discouraged usage of the Tycoon system for the BalticGrid. These events weren't anticipated in originally deciding upon the system. A summary of these changes follows, in order of importance:

- Availability of core components of the system, where central components aren't accessible for review and deployment
- Quality of the available source, where a thorough review of the original bank code revealed many design- and implementation-level issues, some of critical severity (discussed at the BalticGrid Fourth AHM)
- The poor quality of system documentation made available with the system; and
- The lack of tools available to test the correctness and general functionality of the system.

Although in recent communications, H.P. has expressed an interest in making available the missing components, these aren't anticipated to be available with enough time to thoroughly review the available code, which was the primary goal of this activity. Assuming these



releases remedy the system's history of low quality, the matter of documentation and test suites remains. Furthermore, theoretic issues with the Tycoon bank remain unaddressed.

A final solution has not yet been put forth: there are several several proposed solutions to these issues. These include, but are not limited to, negotiation with H.P. regarding the availability of their system; forking the Tycoon sources at the last known point of full-system availability (estimated to be September or October, 2007); or completely re-writing the utility from those points listed in the Objectives section of this document.

Since the poor quality of some parts of the system were only fully explored in October, and the unavailability of core components discovered and verified in late November, appropriate measures have not yet been determined. This decision will be re-raised and resolved by the final report.



REFERENCES

1. H.P. Labs Research: Tycoon: Market-based Resource Allocation. <http://tycoon-wiki.hpl.hp.com>
2. Installation and Configuration of Tycoon. <http://tycoon.grid.lumii.lv>
3. Fedora Project. <http://fedoraproject.org>
4. RedHat GNU/Linux. <http://www.redhat.com>
5. H.P. Labs: Advanced Research at H.P. <http://www.hpl.hp.com>
6. Xen: Open Source Virtualisation. <http://www.xensource.com>
7. BalticGrid Fourth All-Hands Meeting: <http://balticgrid.org/Dissemination/News/ahm4>
8. Kristaps Dzonsons. *Capital Integrity in the Tycoon System*. BalticGrid internal report. November, 2007.



GLOSSARY

CE	Computing Element
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Service
EGEE	Enabling Grids for E-science
FQAN	Fully Qualified Attribute Name
GPL	General Public License
IP	Internet Protocol. Also IP address.
QoS	Quality of Service
RPM	Red Hat Package Manager
SSH	Secure Shell
SLA	Service Level Agreement
SLS	Service Location Server
TCP	Transmission Control Protocol
UID	User Identifier
VM	Virtual Machine