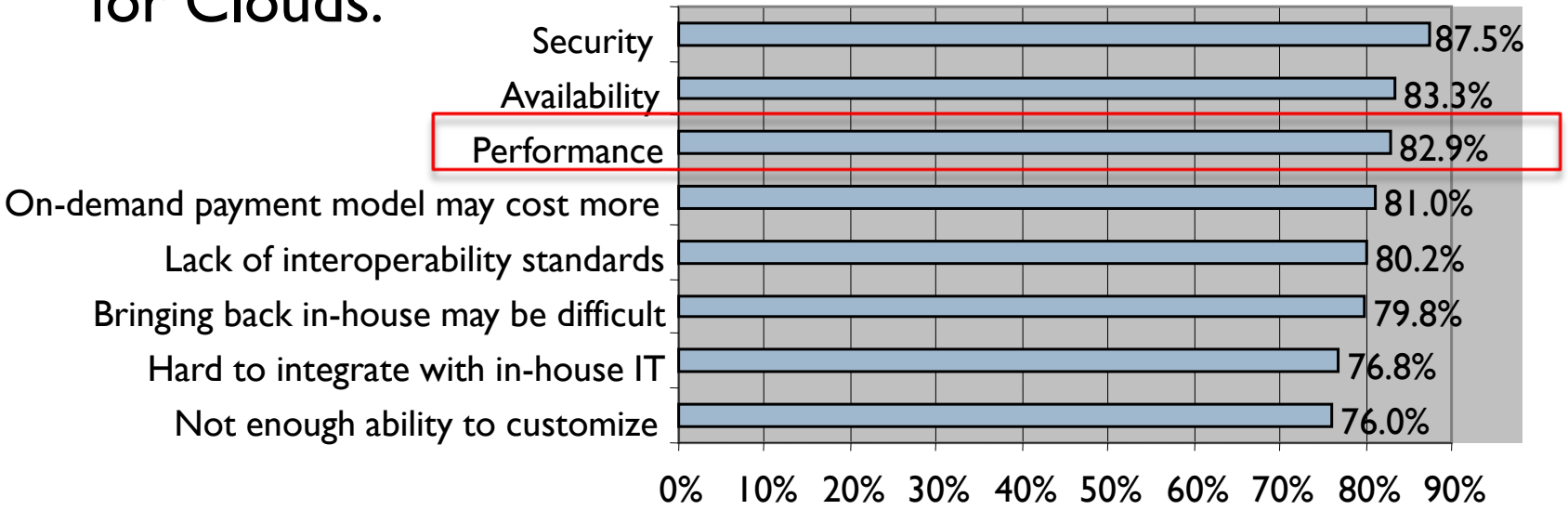
Georgia Tech College of Computing

# An automated Approach to Create, Manage and Analyze Large-scale Experiments for Elastic n-Tier Applications in Computing Clouds

*Deepal Jayasinghe*

# Importance of the Performance

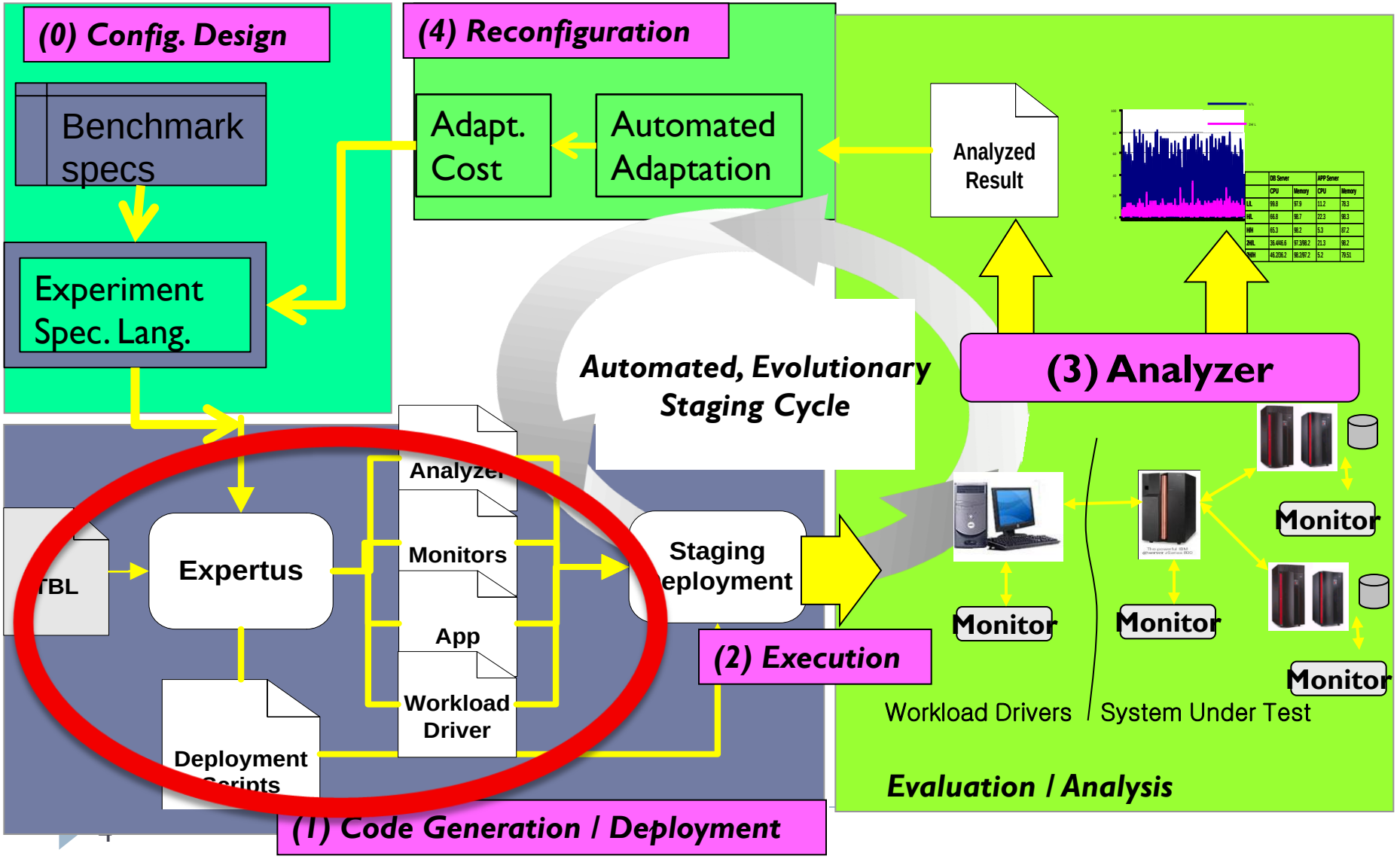
- Extra delay of just 100ms cloud result in roughly 1% loss in sales. amazon.com
- Additional delay of just 500ms could reduce revenues by 20%. Google
- IDC reported performance to be top 3 user considerations for Clouds.



# Large-scale Experimental Measurements

- **Idea**: Use large-scale experimental data as "real" (predictive) models for performance and scalability constraints;
  - Findings enhance the **understanding** of application performance in compute clouds.
- **Motivations**:
  - Performance of two identical software configurations can be vastly different in two hardware configurations (Cloud'11).
  - Same application can exhibit non trivial performance characteristics in the same hardware platform (IPDPS'10).

# Elba: Automating Measurements



# Automating Large-scale Experiments

- **Create**
  - Prepare the platform, deploy and configure the application.
- **Manage**
  - Start the application, execute workloads, data collection.
- **Analyze**
  - Data analysis (visualization) and building hypothesis.

# Automation through Code Generation

- **Idea**: Generate resources to automatically create, manage and analyze the experiments from user-friendly specification files.
- Goals:
  - Support heterogeneous clouds and applications.
  - Need flexible customization.
  - Easy to maintain and extend.

# Key Entities

- Artifacts
  - Software packages, Applications, Nodes
- Dependencies
  - JDBC driver, URLs
- Deployment/Start Order
  - "Deploy" before "Configure"
- Constraints
  - JDK 1.7, RHEL 6

# Key Challenges

- Abstract Mapping
  - Should translate high-level design specification into create, manage and analyze scripts + other resources (e.g., property files, configuration files).
- Heterogeneity
  - Should accommodate inputs from multiple specification regimes and output to multiple target platforms.
- Flexible Customization
  - Each application developer can augment scripts and introduce his on application/cloud specific properties.

# Our Approach

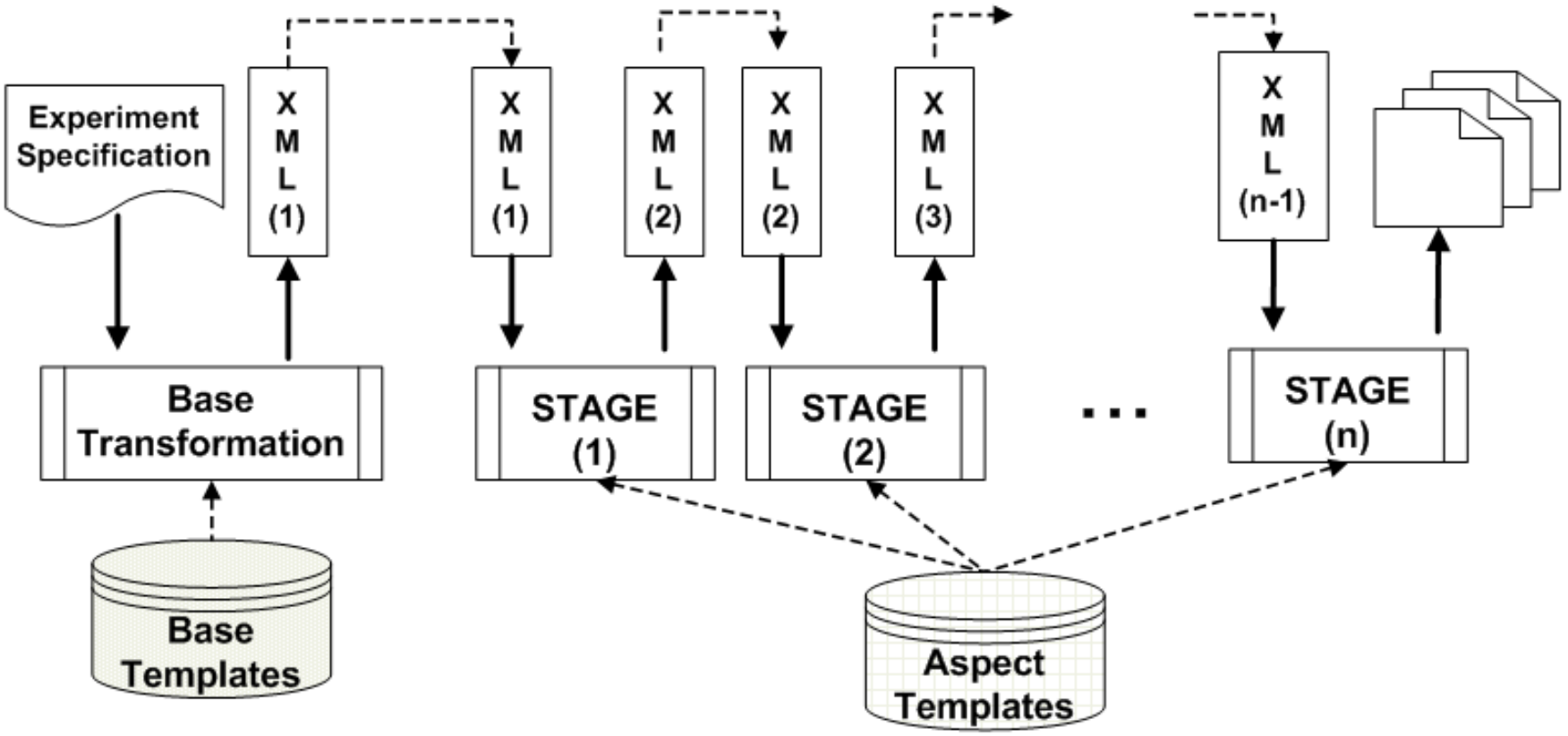
- Generate code through a series of XSL transformation stages.
- The intermediate XML created in the current stage becomes the input to the next stage.
- Flexible customization through AOP (Aspect Oriented Programming) techniques.

# Multi-Stage Code Generation

- Embed a very complex transformation in a linear fashion.
- A transformation at one stage is visible to all the following stages.
- One type of transformation/customization at one stage:
  - Application related processing.
  - Cloud related processing.
  - Operating system related processing.
  - User specific customization.

Georgia Tech College of Computing

# Code Generation Pipeline



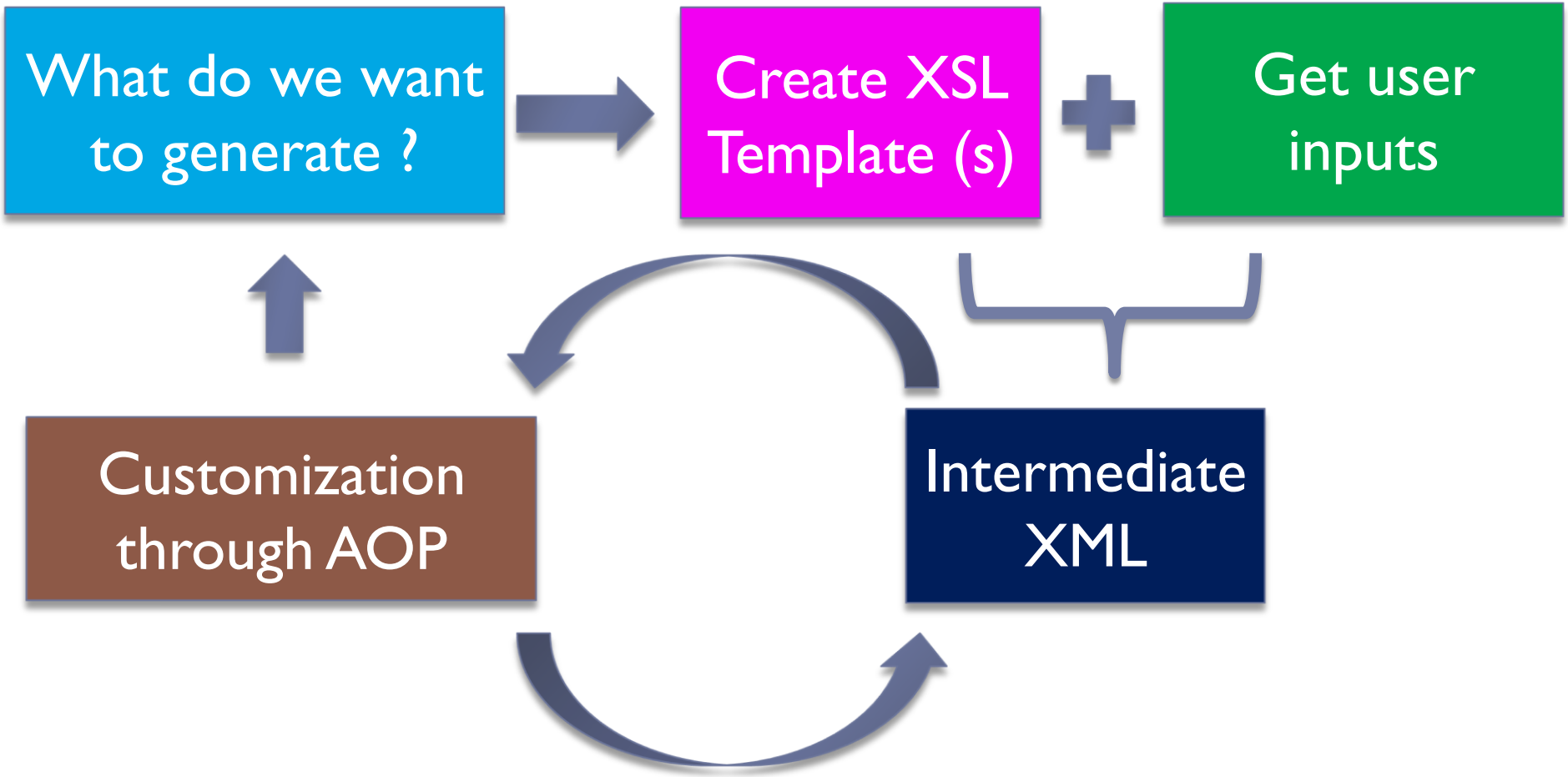
# XML and XSLT ?

- Flexibility, Extensibility, and Maintainability.
- Ability to accept new tags, thereby bypassing the overhead encountered when managing XSL templates.
- XSL helps to transform a single document to different formats.

# Aspect Oriented Programming

- Aims to increase modularity by allowing the separation of cross-cutting concerns.
  - Provide secondary requirements without having to modify the original static model.
- Terminology
  - **Advice –** The additional code that we want to apply to our existing model.
  - **Point-cut -** The term given to the point of execution in the application at which cross-cutting concern needs to be applied.
  - **Aspect -** The combination of the point-cut and the advice.

# The Complete Process



# Bob's script

```
cd /opt/rubbos/output
source set_elba_env.sh
mkdir -p /mnt/rubbos
# install RUBBoS
tar xzf /opt/softwares/RUBBoS.tar.gz
.....
make
sudo make install
```

```
cd <xsl:value-of select="//params[@name='OUTPUT_HOME']/@value"/>
source set_elba_env.sh
mkdir -p <xsl:value-of select="//params[@name='RUBBOS_TOP']/@value"/>
# install RUBBoS
tar xzf <xsl:value-of
select="//params@name='SOFTWARE_HOME']/@value"/><xsl:value-of
select="//params@name='RUBBOS_TARBALL']/@value"/>
make
<axpect id="addsudo"/> make install
```

Bob's Aspect

```
<xsl:template match="//axpect[@id='addsudo']">sudo
</xsl:template>
```

```
<xtbl name="Rubbos" version="0.1">

<params name="OUTPUT_HOME" value="/opt/rubbos/output"/>
<params name="SOFTWARE_HOME" value="/opt/softwares"/>
<params name="RUBBOS_TOP" value="/mnt/rubbos"/>
<params name="RUBBOS_TARBALL " value="RUBBoS.tar.gz"/>
<params name="MYSQL_PORT" value="3313"/>
.....
</xtbl>
```

```
<file>
<name id="Tomcat_deploy.sh" loc="/home/scripts"/>
<content>
cd /opt/rubbos/output
source set_elba_env.sh
mkdir -p /mnt/rubbos
# install RUBBoS
tar xzf /opt/softwares/RUBBoS.tar.gz
.....
make
<axpect id="addsudo"/> make install
</content>
</file>
```

# Alice's Script

```
cd /opt/rubbos/output
source set_elba_env.sh
mkdir -p /mnt/rubbos
# install RUBBoS
tar xzf /opt/softwares/RUBBoS.tar.gz
.....
make
make install
```

```
cd <xsl:value-of select="//params[@name='OUTPUT_HOME']/@value"/>
source set_elba_env.sh
mkdir -p <xsl:value-of select="//params[@name='RUBBOS_TOP']/@value"/>
# install RUBBoS
tar xzf <xsl:value-of
select="//params@name='SOFTWARE_HOME']/@value"/><xsl:value-of
select="//params@name='RUBBOS_TARBALL']/@value"/>
make
<axpect id="addsudo"/> make install
```

Alice's Aspect

```
<xsl:template match="//axpect[@id='addsudo']">
</xsl:template>
```

```
<xtbl name="Rubbos" version="0.1">

<params name="OUTPUT_HOME" value="/opt/rubbos/output"/>
<params name="SOFTWARE_HOME" value="/opt/softwares"/>
<params name="RUBBOS_TOP" value="/mnt/rubbos"/>
<params name="RUBBOS_TARBALL " value="RUBBoS.tar.gz"/>
<params name="MYSQL_PORT" value="3313"/>
.....
</xtbl>
```

```
<file>
<name id="Tomcat_deploy.sh" loc="/home/scripts"/>
<content>
cd /opt/rubbos/output
source set_elba_env.sh
mkdir -p /mnt/rubbos
# install RUBBoS
tar xzf /opt/softwares/RUBBoS.tar.gz
.....
make
<axpect id="addsudo"/> make install
</content>
</file>
```

# Evaluation Metrics

- Richness of the Tool.
  - Magnitude of completed experiments.
  - Amount of different software packages, clouds, and applications it supports.
- Wide Applicability.
  - How quickly a user can change an existing specification to run the same experiment with different settings.
- Generated Script Types and Magnitude.
  - Depends on the application, software packages, deployment platform, number of experiments.
- Extensibility and Flexibility.
  - Supporting new clouds.
  - Supporting new applications.

# A Selected set of Recent Publications

- Empirical analysis of database server scalability using an n-tier benchmark with read-intensive workload – *ACM SAC 2010*
- Variations in Performance and Scalability when Migrating n-Tier Applications to Different Clouds – *IEEE Cloud 2011*
- The Impact of Soft Resource Allocation on n-Tier Application Scalability – *IEEE IPDPS 2011*
- Challenges and Opportunities in Consolidation at High Resource Utilization: Non-monotonic Response Time Variations in n-Tier Applications – *IEEE Cloud 2012*
- Profit-Based Experimental Analysis of IaaS Cloud Performance: Impact of Software Resource Allocation – *IEEE SCC 2012*

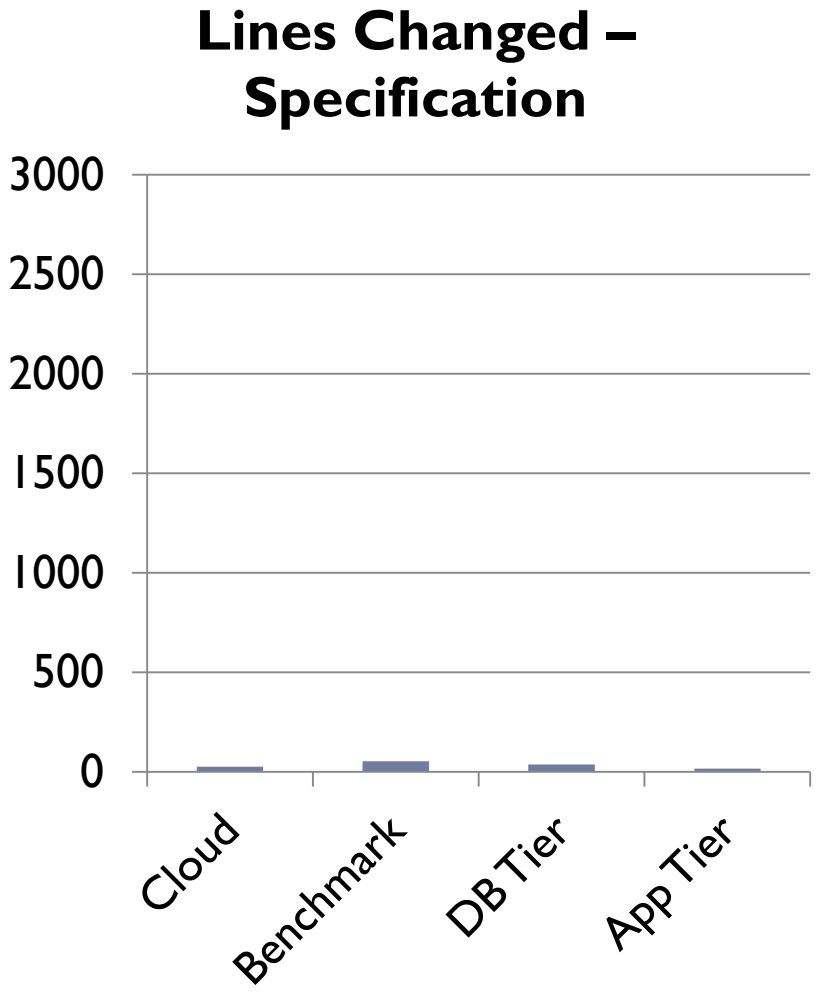
# Wide Applicability

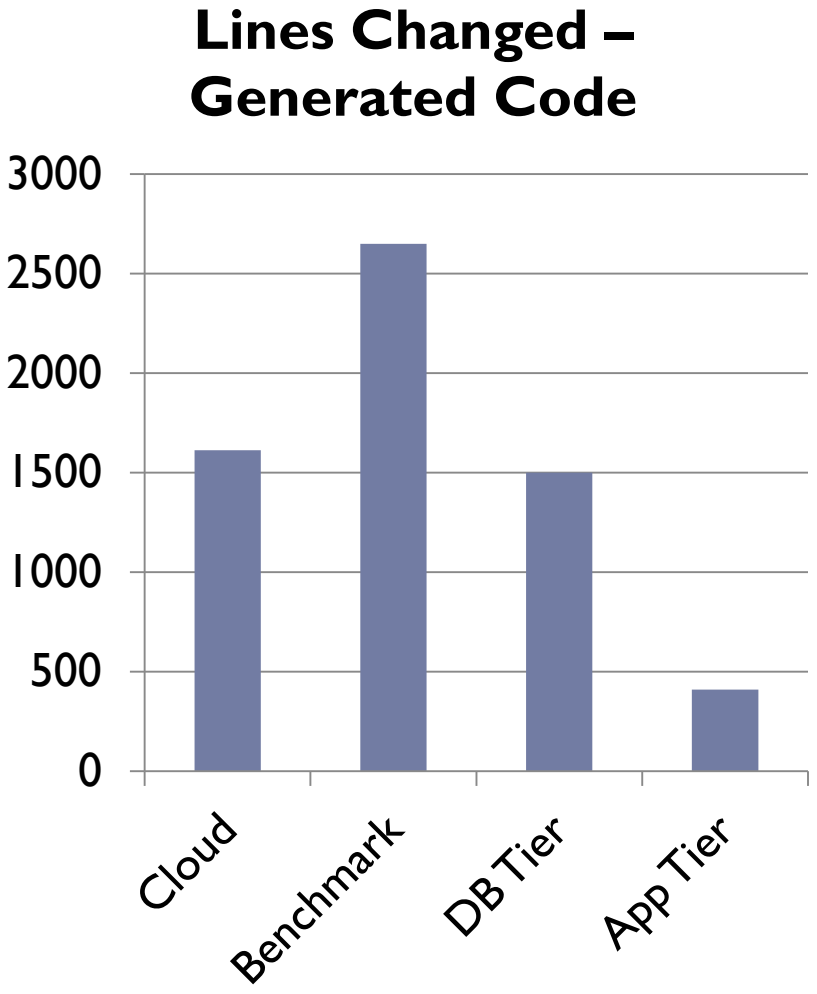
- Over 5,000 different hardware configurations.
- Over 10,000 software configurations.
- Over 100, 000 experiments.
- Many clouds (e.g., Emulab, EC2, OpenCirrus, Georgia Tech cluster, University of Tokyo, and Wipro).
- Many representative applications (e.g., RUBBoS, RUBiS, CloudStone, and over 10 OLTP benchmarks).

# Flexible Extensibility

- 8.21% of template line changes, to support Amazon EC2 once we had support for the Emulab cloud.
- Caused a 25.35% change in the generated code for an application scenario with 18 nodes.
- Switching from the RUBBoS to the RUBiS required only a 5.66% template change.

# Specification Changes vs. Changes in Generated Code

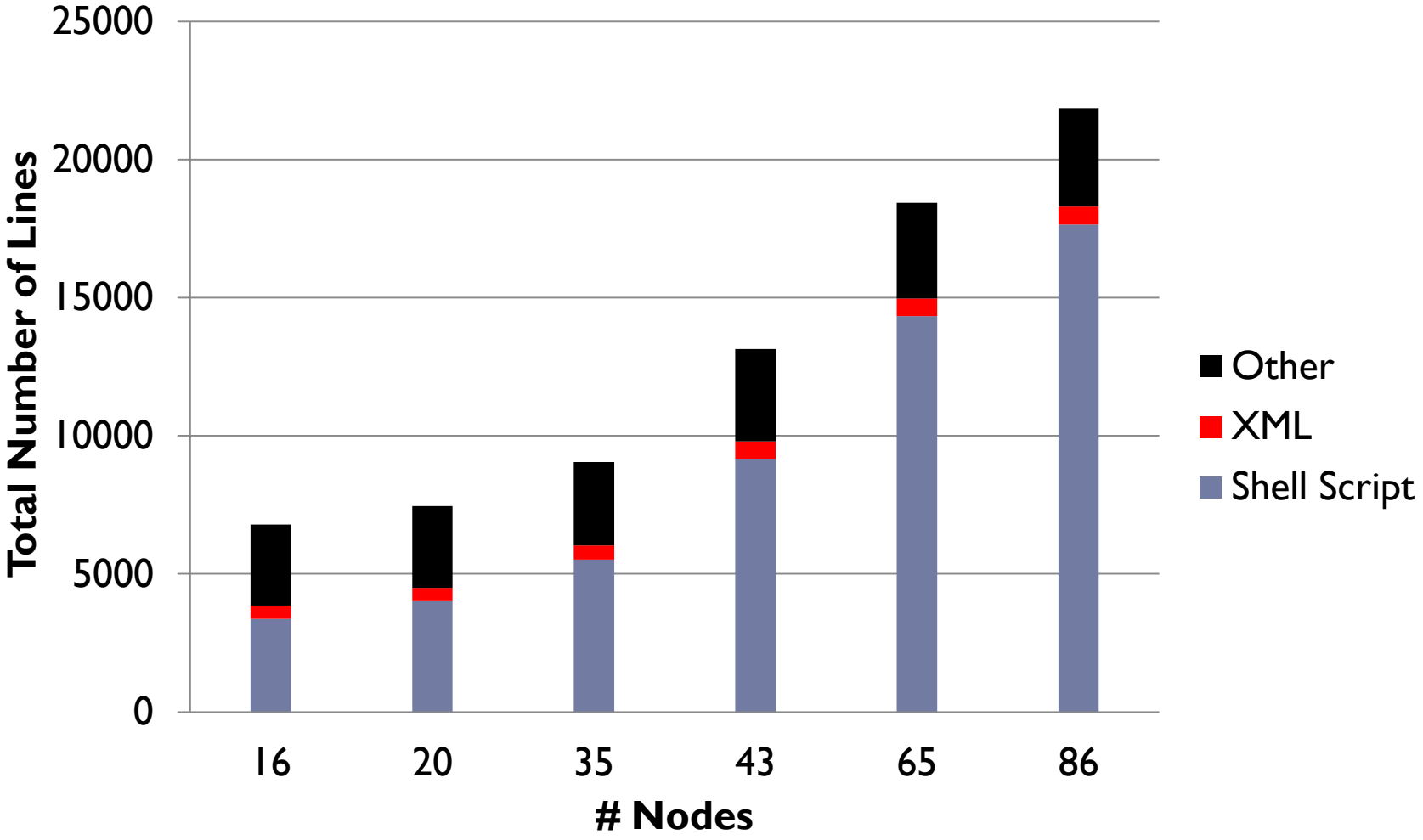


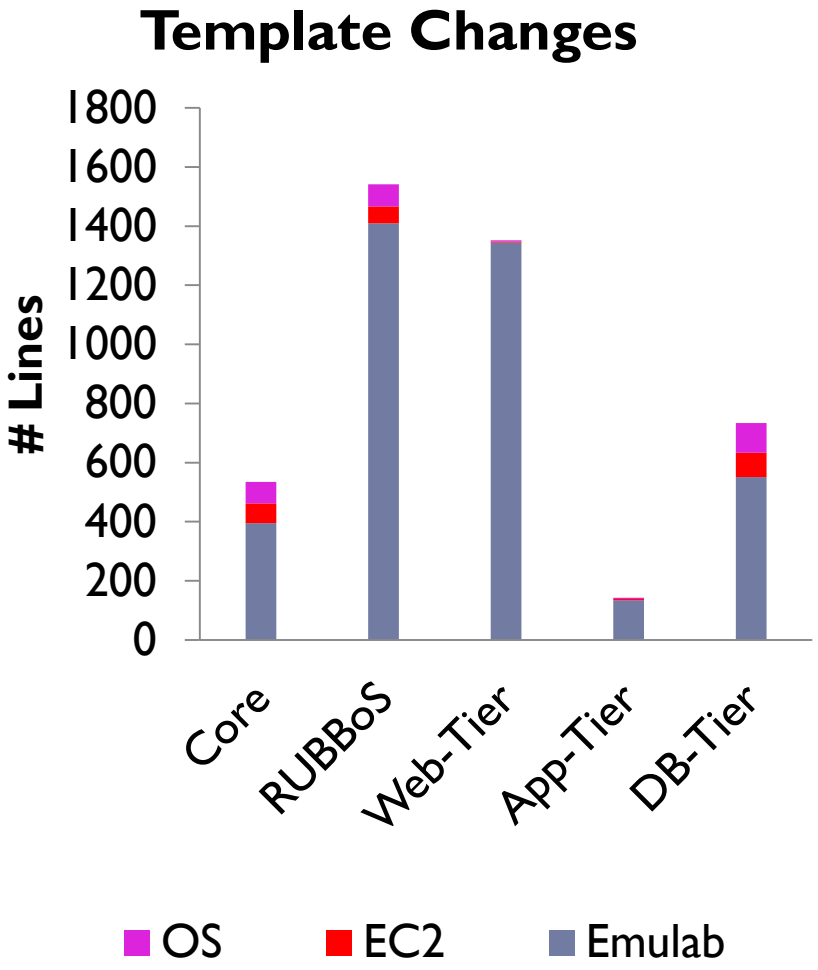

RUBBoS Application with 18 nodes in Emulab

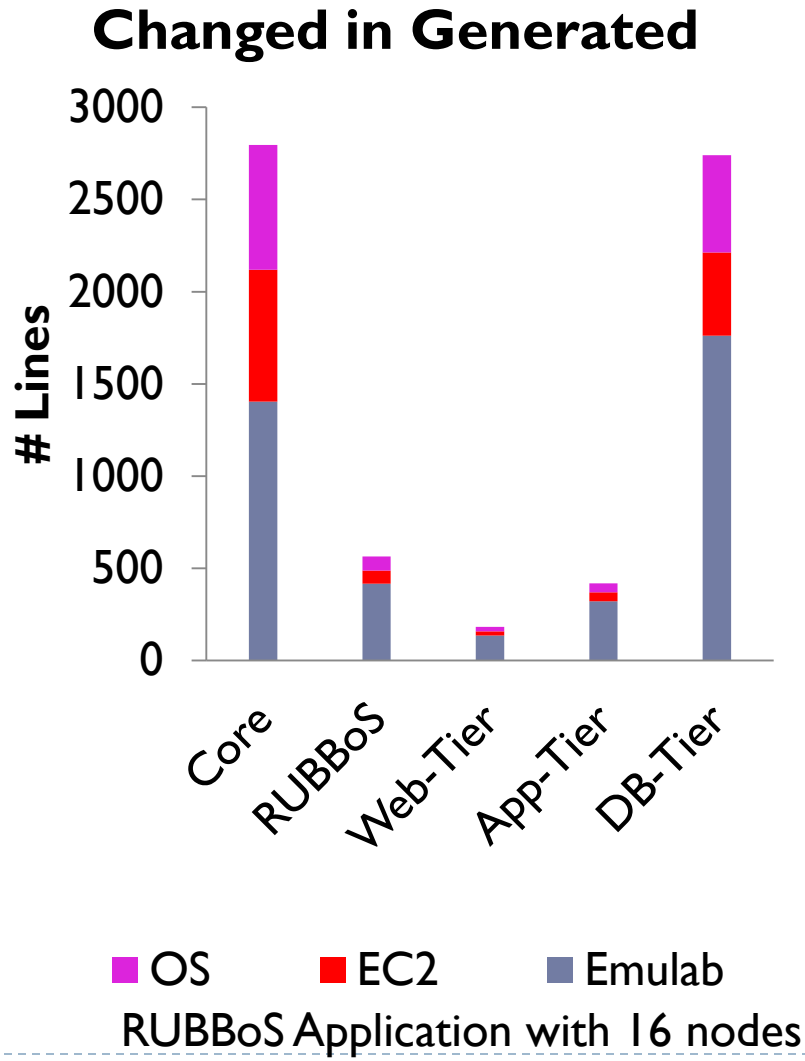
Georgia Tech College of Computing

# Number of Nodes vs. Generated Code



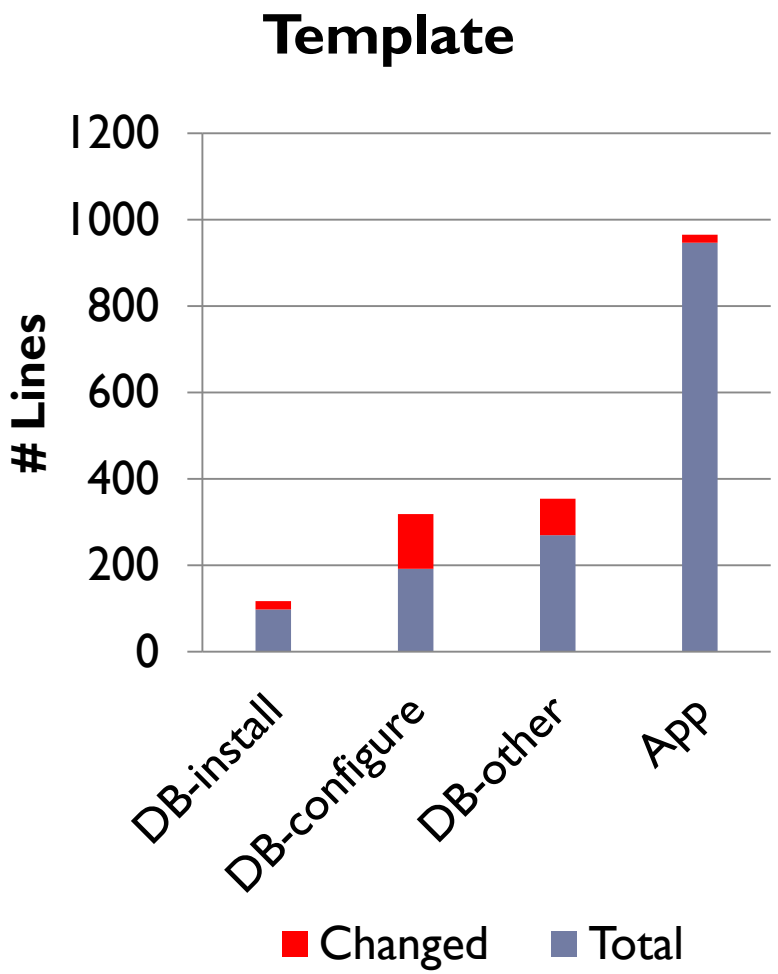
# Supporting new Clouds: Changes to Template vs. Generated

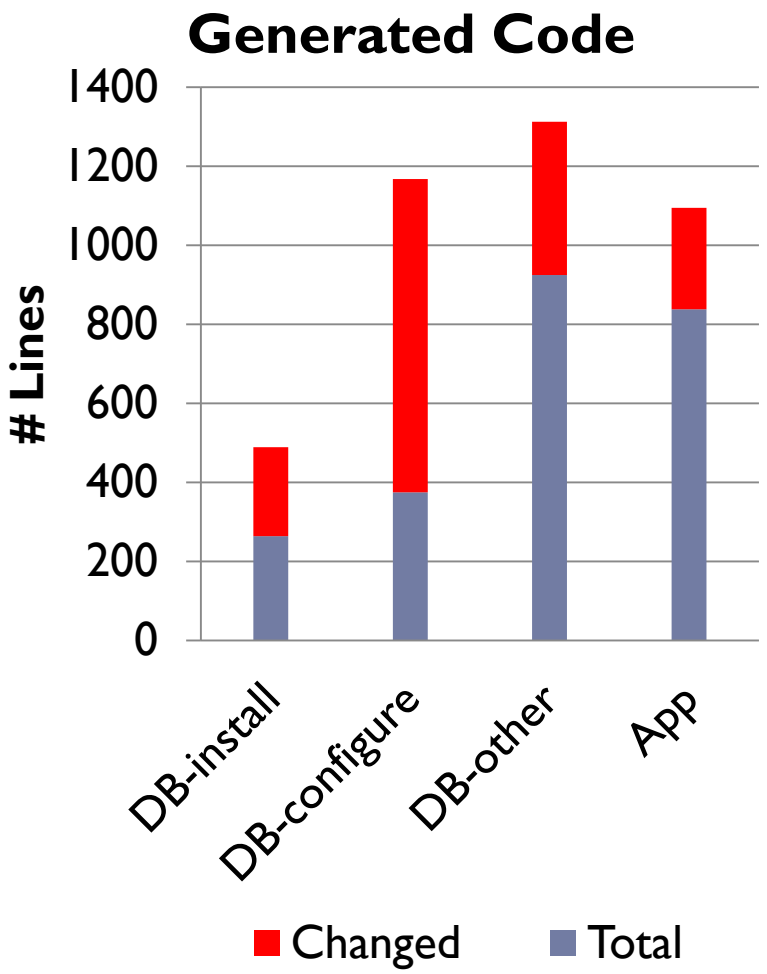



RUBBoS Application with 16 nodes

# Supporting new DBMS: Changes to Template vs. Generated




Adding MySQL Cluster support (Application with 4 data nodes)

# Expertus – Web Portal

- User can upload an experiment specification.
- Specify metadata (e.g., cloud, date, user name).
- Provide a custom aspect specification.
- Generate resources to automate experiment measurements.

http://elbacontrol.cc.gatech.edu:8080/elba/

# Elba

**Automated Design/Configuration Evaluation and Tuning**

Georgia Tech College of Computing

Home | Project | People | Publications | Contact

## Introduction to Elba

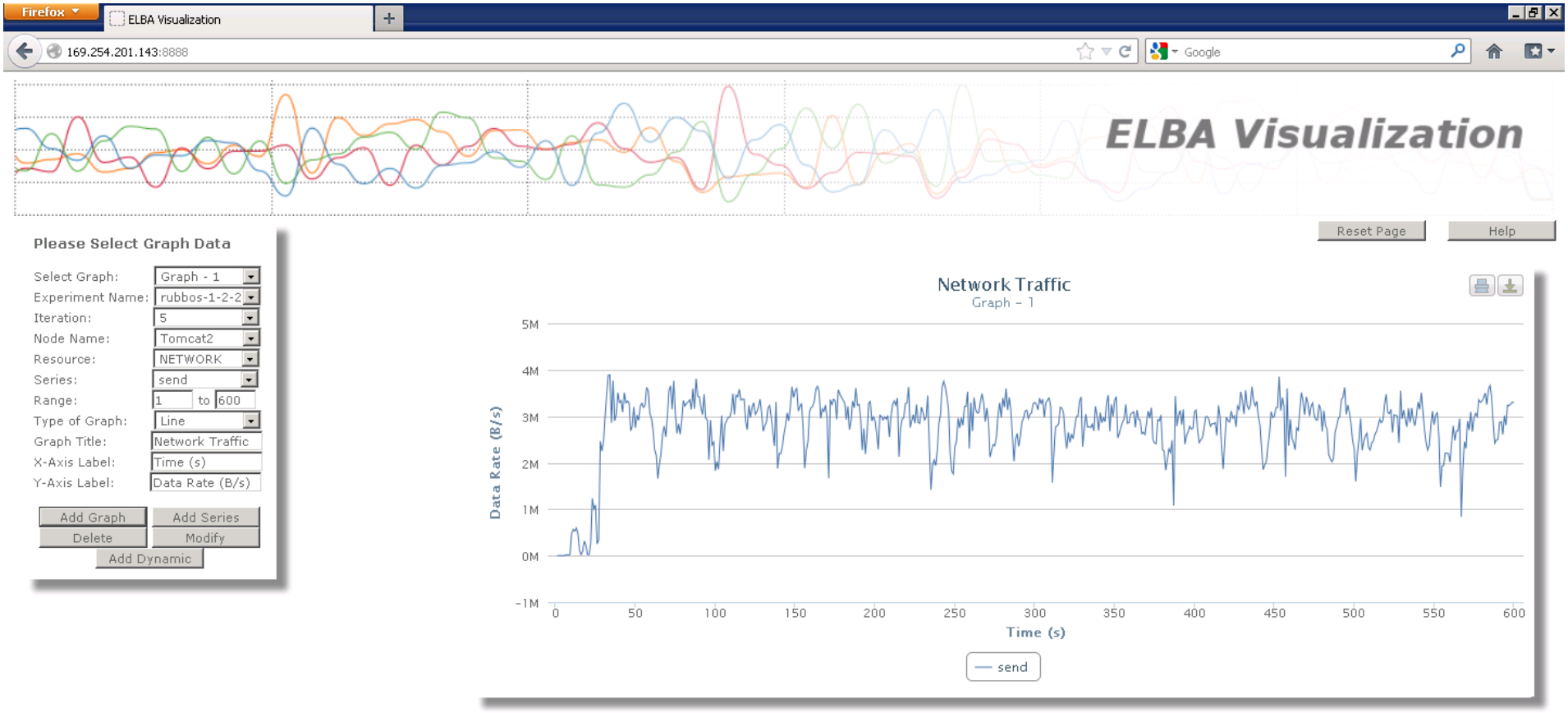
The main goal of the project is to provide flexible and extensible toolkit to generate scripts and other resources to automate application deployment in cloud. You can upload your experiment configuration file, and the our tool will generate all the scripts which are needed to get the application working. It will automatically create the tar.gz and allow you to download. Simply copy that and use. The online code generation feature still under experiment phase and will add more features soon.

## Upload an Experiment Configuration File

| | |
|---|---|
| **Experiment Name** | |
| **Experiment Description** | |
| **Experiment platform (cloud)** | ec2 |
| **Benchmark application** | rubbos |
| **User name** | |
| **Experiment Date** | select |
| **Experiment Configuration File** | Browse... |
| **Aspect Configuration File** | Browse... |
| | Upload |

If you have comments or questions, please contact us at Elba Project. Thanks!

# Experstore – Web Portal




http://expertus.cc.gatech.edu:8888/

# Experstore – Web Portal

Georgia Tech College of Computing

Georgia Tech College of Computing

Deepal Jayasinghe

School of Computer Science