XNAT Tuning & Monitoring

John Paulett

jpaulett@wustl.edu

June 29, 2010

Overview

Share NRG's experiences running a large XNAT installation, including methods for tuning, testing, and monitoring the application.

Plan

1. Sample XNAT Architecture
2. Hardware “Recommendations”
3. Monitoring XNAT
4. Performance Testing Tools
5. Tuning XNAT

XNAT SCALES!

XNAT Scales

From a single study with dozens of scans

To hundreds of studies, including large, multi-site studies

Central Neuroimaging Data Archive (CNDA)

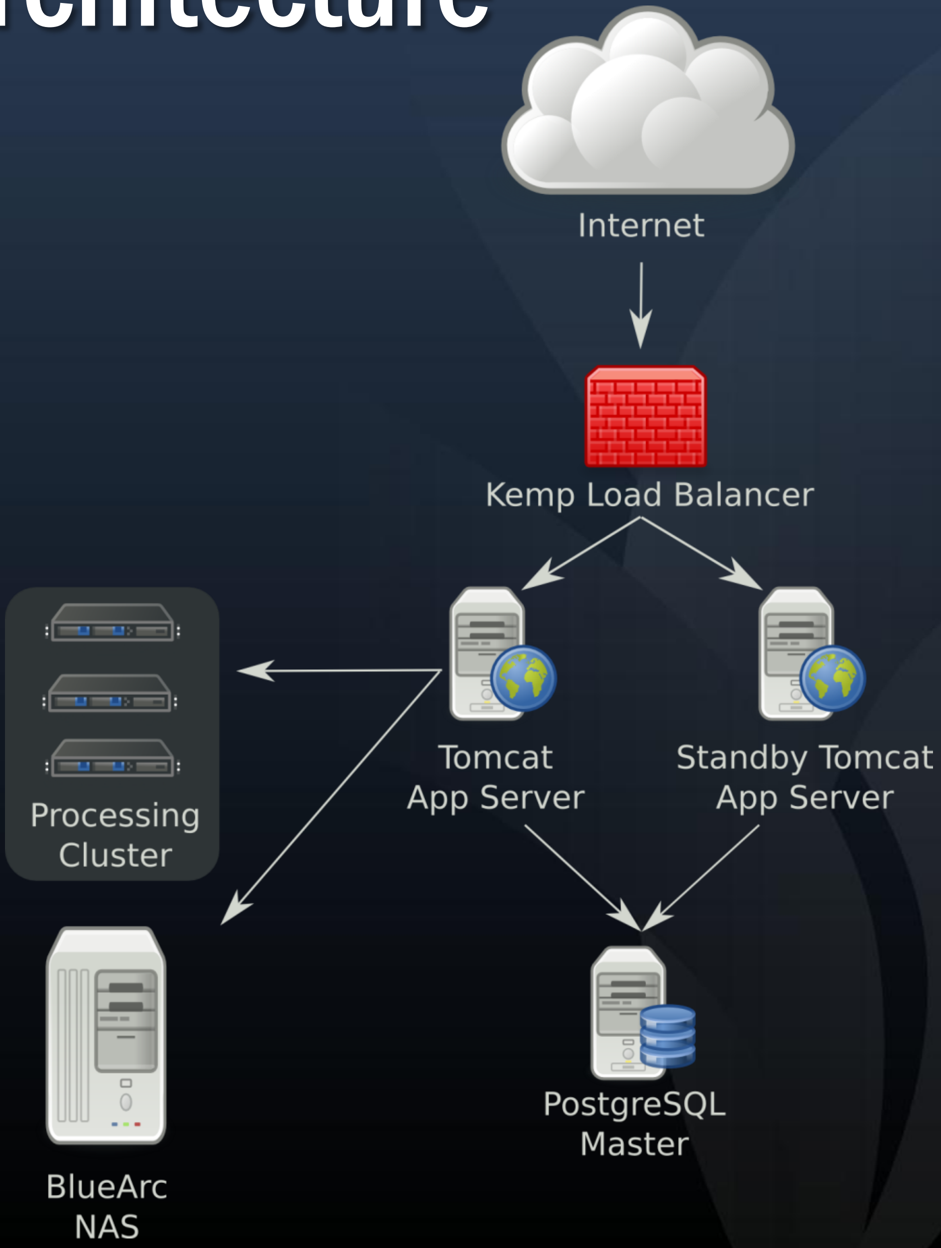
Flagship XNAT installation at Washington University (WUSTL)

As of June 2010:

- 500 studies
- 8000 subjects
- 11k imaging sessions
- 9 TB of data

XNAT

CNDA Architecture



XNAT

CNDA Architecture

- 1x Kemp load balancer
  - SSL acceleration
- 1x Quad-core Xeon, 16GB RAM: PostgreSQL 8.3
- 2x Dual-core Xeon, 4GB RAM (one in standby): Tomcat 5.5 & DicomServer
- BlueArc NAS
- Multiple Sun Grid Engine nodes

XNAT

FUTURE ARCHITECTURE

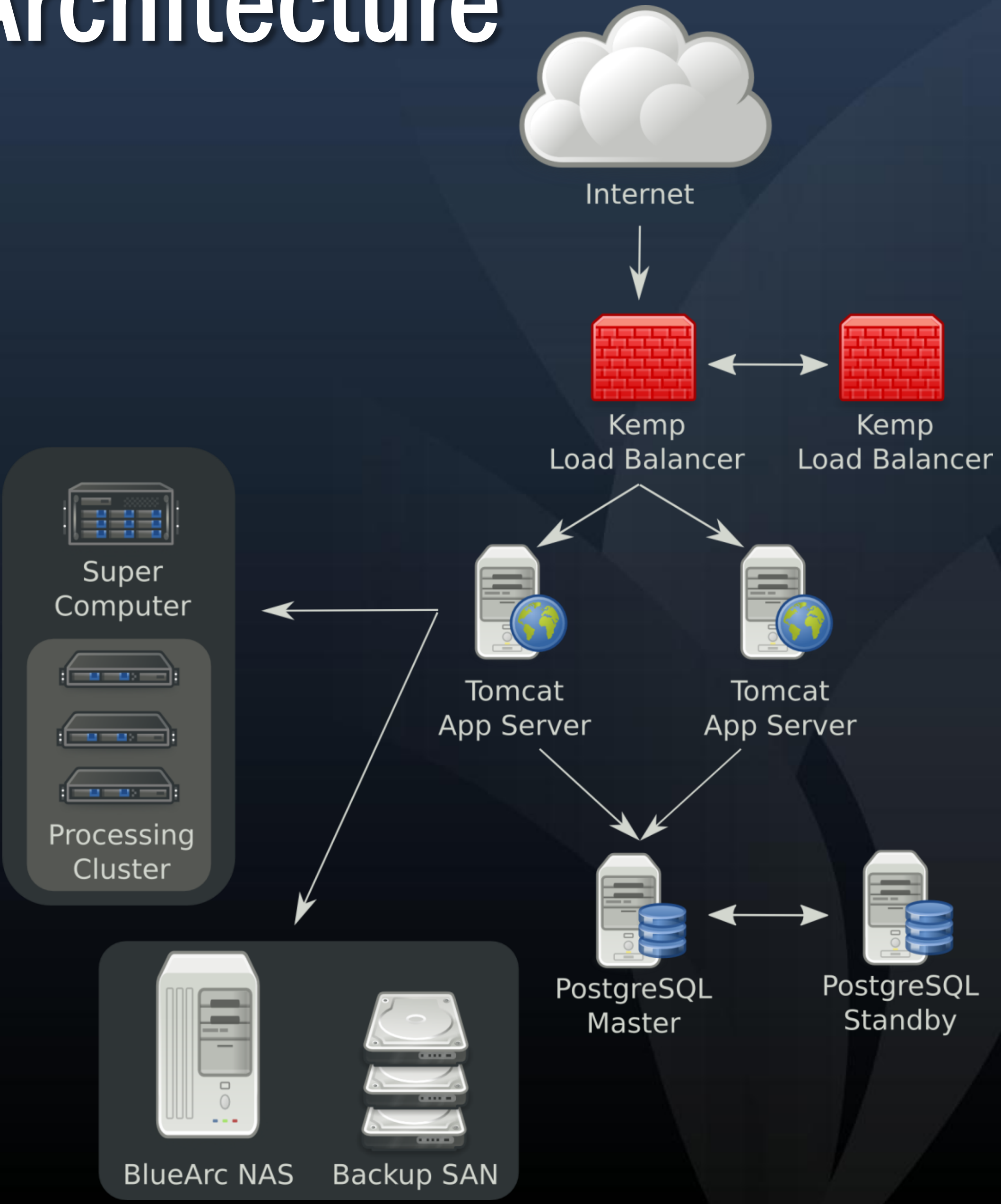
Future Architecture

Reduce single points of failure
- Standby Kemp load balancer
- PostgreSQL Warm Standby
- Actively load balance Tomcat
- Archival storage SAN

Use new super computer at WUSTL

Improve ability to upgrade without downtime

Future Architecture



XNAT

“RECOMMENDED” HARDWARE

Grow into your Architecture

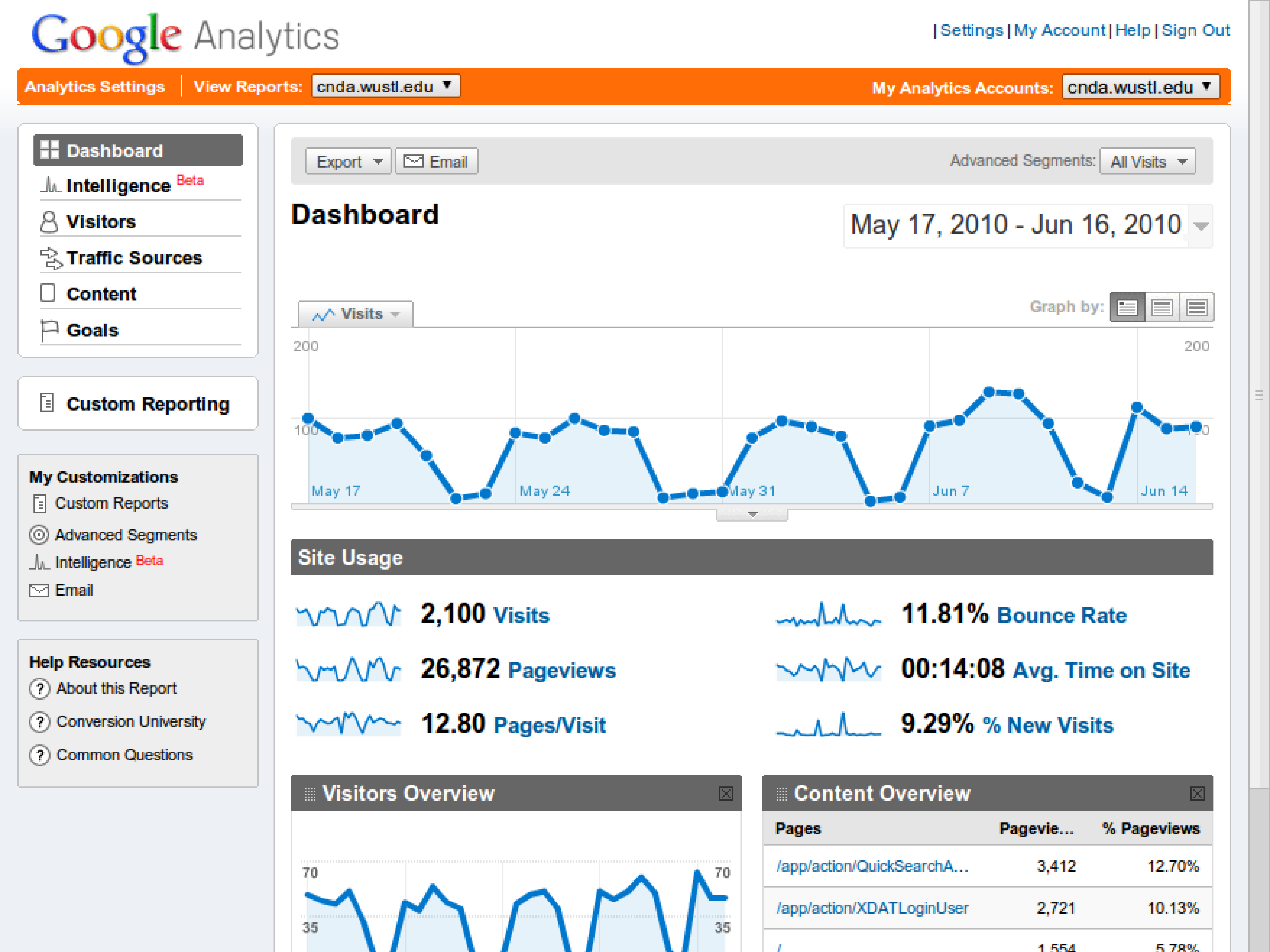
Get single good server

- Multicore with 4-16GB RAM (better than central.xnat.org)
- Consider your archive’s future size & location

When you outgrow:

- Buy a more powerful machine for PostgreSQL
- Leave Tomcat on first server

MONITORING XNAT



Dashboard

May 17, 2010 - Jun 16, 2010

Site Usage

2,100 Visits

26,872 Pageviews

12.80 Pages/Visit

11.81% Bounce Rate

00:14:08 Avg. Time on Site

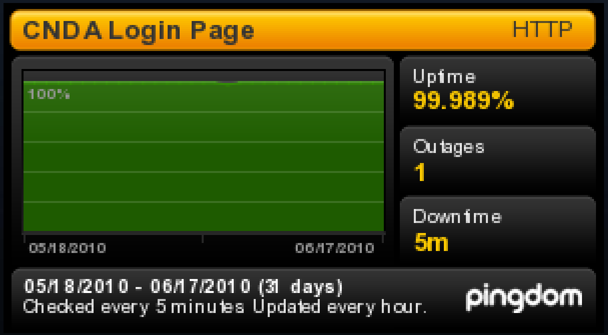
9.29% % New Visits

Content Overview

Pages	Pagevie...	% Pageviews
/app/action/QuickSearchA...	3,412	12.70%
/app/action/XDATLoginUser	2,721	10.13%
/	1,554	5.78%

Pingdom

- World-wide tests for site availability & response time
- SMS & Email alerts when sites are unavailable



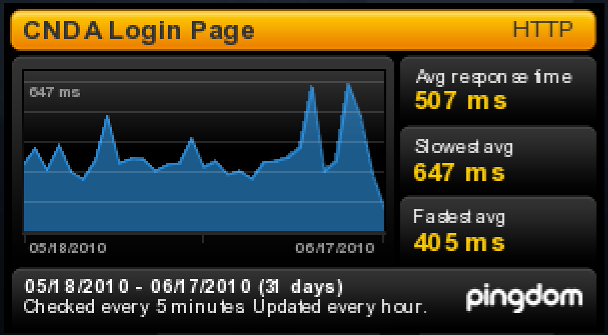
CNDA Login Page HTTP

Uptime 99.989%

Outages 1

Downtime 5m

05/18/2010 - 06/17/2010 (31 days)
Checked every 5 minutes Updated every hour.



CNDA Login Page HTTP

Avg response time 507 ms

Slowest avg 647 ms

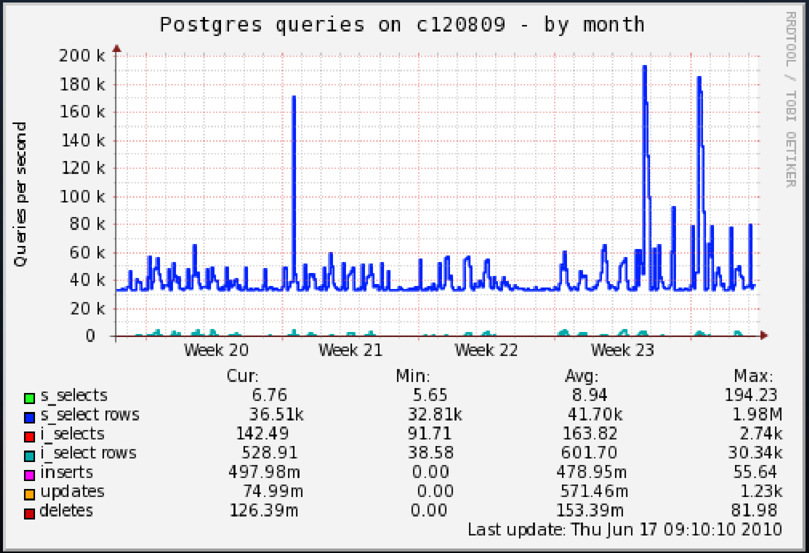
Fastest avg 405 ms

05/18/2010 - 06/17/2010 (31 days)
Checked every 5 minutes Updated every hour.

Munin

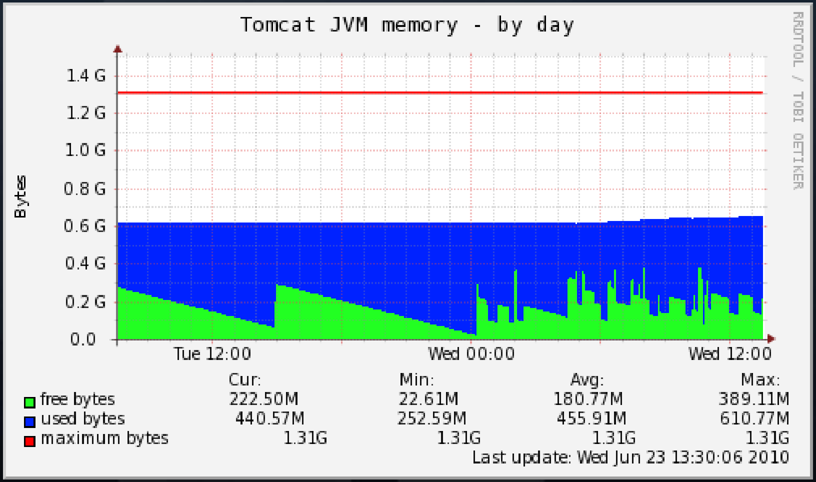
PostgreSQL, Tomcat, & Linux metrics over time

– Memory, CPU, queries, requests, etc.



	Cur:	Min:	Avg:	Max:
🟩 s_selects	6.76	5.65	8.94	194.23
🟦 s_select rows	36.51k	32.81k	41.70k	1.98M
🟥 i_selects	142.49	91.71	163.82	2.74k
🟦 i_select rows	528.91	38.58	601.70	30.34k
🟪 inserts	497.98m	0.00	478.95m	55.64
🟧 updates	74.99m	0.00	571.46m	1.23k
🟥 deletes	126.39m	0.00	153.39m	81.98

Last update: Thu Jun 17 09:10:10 2010



	Cur:	Min:	Avg:	Max:
🟩 free bytes	222.50M	22.61M	180.77M	389.11M
🟦 used bytes	440.57M	252.59M	455.91M	610.77M
🟥 maximum bytes	1.31G	1.31G	1.31G	1.31G

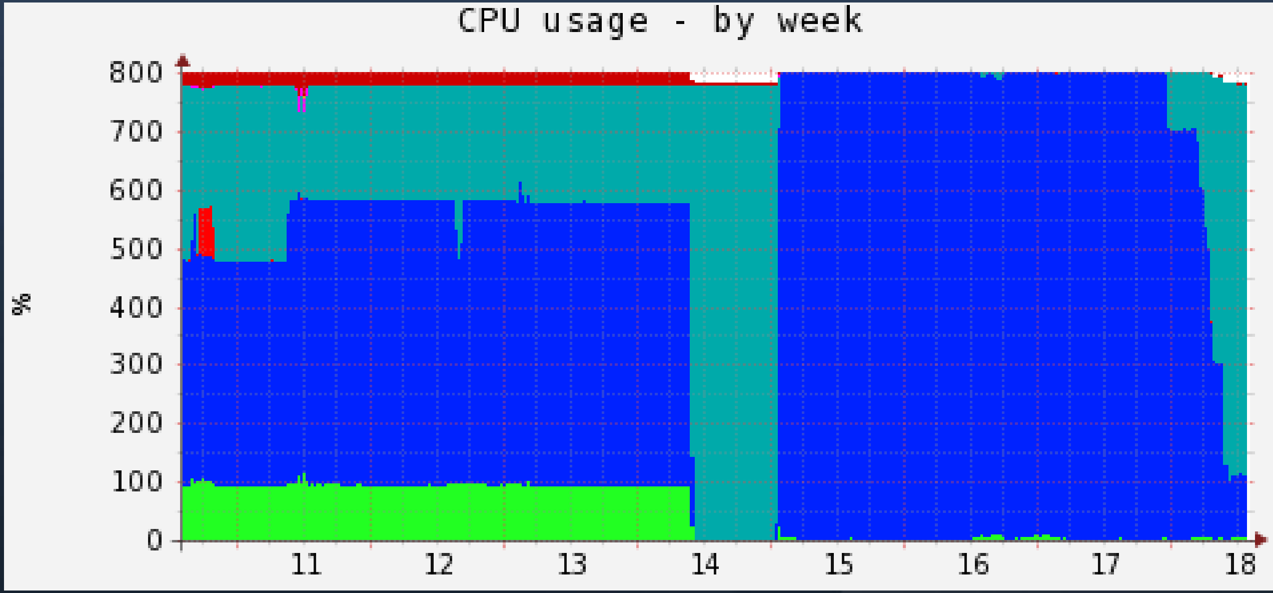
Last update: Wed Jun 23 13:30:06 2010

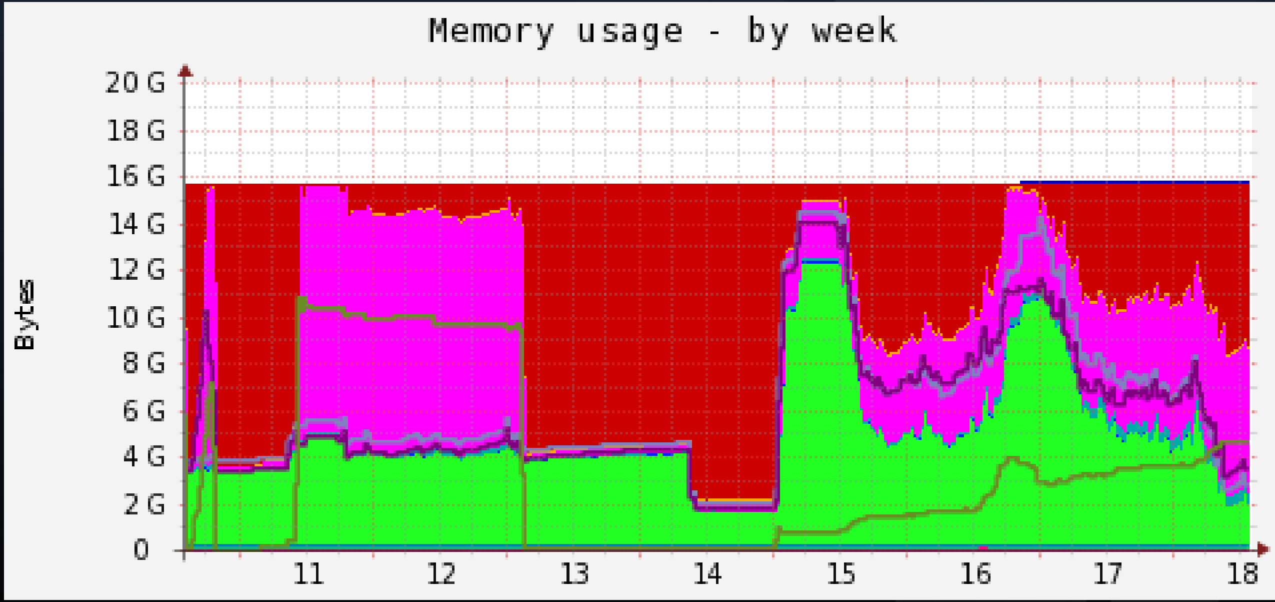
XNAT

Overview

- localhost
  - **localhost** :: [Disk Network Nfs Processes Sendmail System]
- neuroimage.wustl.edu
  - **nrglin4.neuroimage.wustl.edu** :: [Disk Network Printing Processes Sendmail System Tomcat]
- nrglin10
  - **nrglin10** :: [Disk Network Printing Processes Sendmail System]
- nrglin5
  - **nrglin5** :: [Disk Network Nfs Postgresql Processes Sendmail System]
- nrglin6
  - **nrglin6** :: [Disk Network Nfs Printing Processes Sendmail System]
- nrglin7
  - **nrglin7** :: [Disk Network Printing Processes Sendmail System]
- nrglin8
  - **nrglin8** :: [Disk Network Printing Processes Sendmail System]

This page was generated by Munin version 1.2.6 at 2010-06-23 T 13:30:24


CPU usage - by week
%
800
700
600
500
400
300
200
100
0
11
12
13
14
15
16
17
18


Memory usage - by week
Bytes
20 G
18 G
16 G
14 G
12 G
10 G
8 G
6 G
4 G
2 G
0
11
12
13
14
15
16
17
18

XNAT

Monit

Active process monitoring & management

Define criteria for emailing alerts & restarting processes

- CPU, memory thresholds
- Connection failures (check web services)

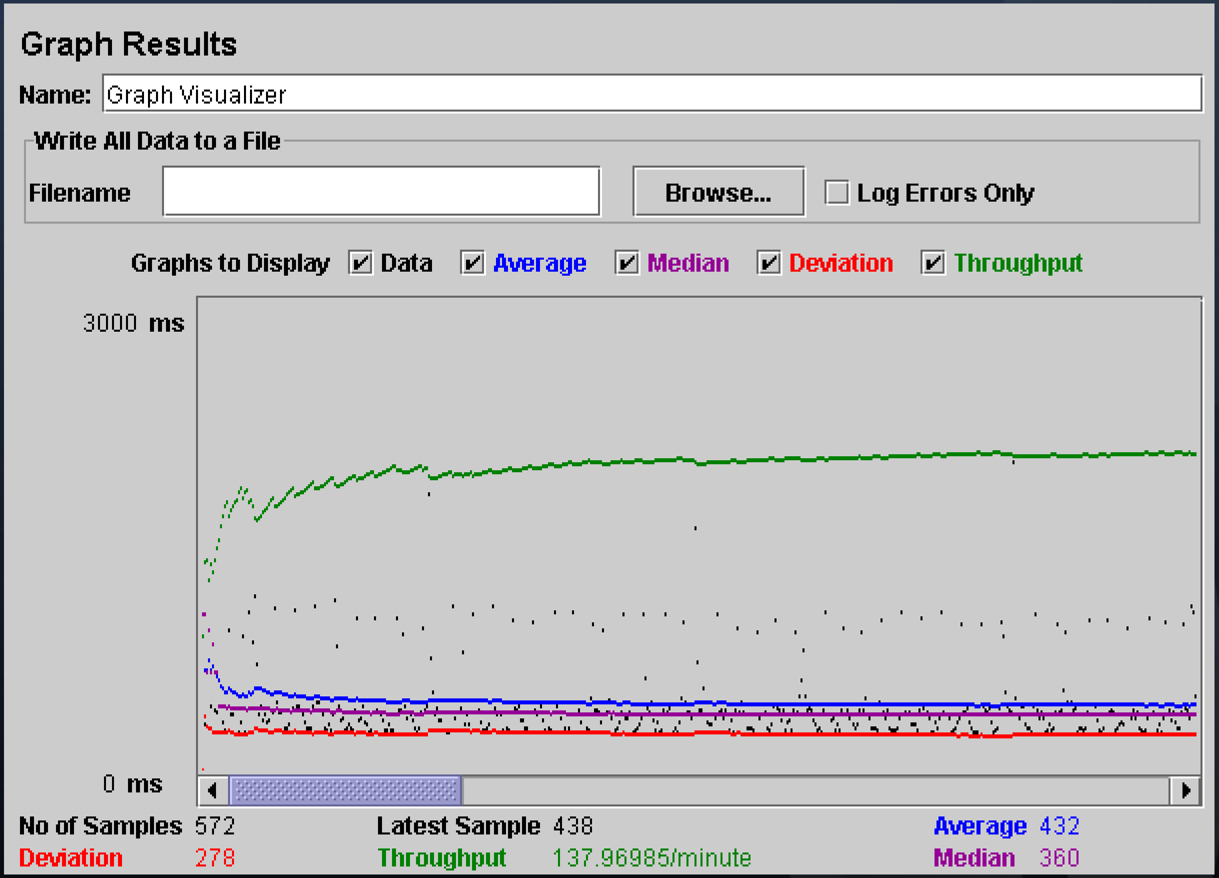
PERFORMANCE TESTING TOOLS

JMeter

Generate load & analyze throughput

Complex HTTP transactions

JMeter



Graph Results

Name: Graph Visualizer

Write All Data to a File

Filename [Browse...] ☐ Log Errors Only

Graphs to Display ✓ Data ✓ Average ✓ Median ✓ Deviation ✓ Throughput

No of Samples	572	Latest Sample	438	Average	432
Deviation	278	Throughput	137.96985/minute	Median	360

https://svn.apache.org/repos/asf/jakarta/jmeter/

XNAT

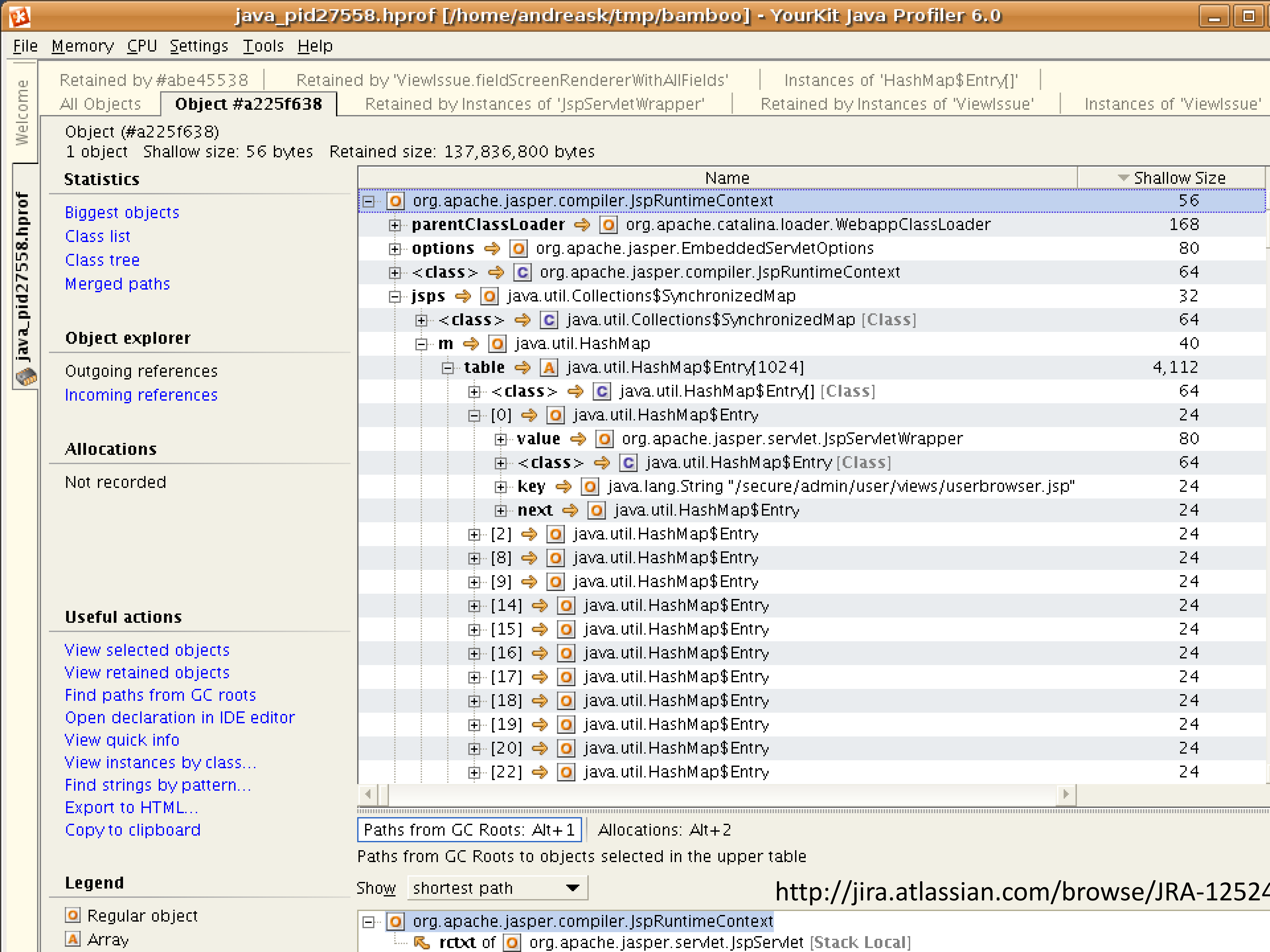
YourKit Profiling

Lower level debugging tool

Memory & CPU profiling

Hunt down memory leaks & code hot spots

Can instrument in production server



java_pid27558.hprof [/home/andreask/tmp/bamboo] - YourKit Java Profiler 6.0

File Memory CPU Settings Tools Help

Retained by #abe45538 | Retained by 'ViewIssue.fieldScreenRendererWithAllFields' | Instances of 'HashMap$Entry[]'
All Objects | Object #a225f638 | Retained by Instances of 'JspServletWrapper' | Retained by Instances of 'ViewIssue' | Instances of 'ViewIssue'

Object (#a225f638)
1 object Shallow size: 56 bytes Retained size: 137,836,800 bytes

Statistics

Biggest objects
Class list
Class tree
Merged paths

Object explorer

Outgoing references
Incoming references

Allocations

Not recorded

Useful actions

View selected objects
View retained objects
Find paths from GC roots
Open declaration in IDE editor
View quick info
View instances by class...
Find strings by pattern...
Export to HTML...
Copy to clipboard

Legend

O Regular object
A Array

```
Name                                                                                      Shallow Size
⊟ O org.apache.jasper.compiler.JspRuntimeContext                                                    56
  ⊞ parentClassLoader ⇨ O org.apache.catalina.loader.WebappClassLoader                             168
  ⊞ options ⇨ O org.apache.jasper.EmbeddedServletOptions                                            80
  ⊞ <class> ⇨ C org.apache.jasper.compiler.JspRuntimeContext                                        64
  ⊟ jsps ⇨ O java.util.Collections$SynchronizedMap                                                  32
    ⊞ <class> ⇨ C java.util.Collections$SynchronizedMap [Class]                                     64
    ⊟ m ⇨ O java.util.HashMap                                                                       40
      ⊟ table ⇨ A java.util.HashMap$Entry[1024]                                                  4,112
        ⊞ <class> ⇨ C java.util.HashMap$Entry[] [Class]                                             64
        ⊟ [0] ⇨ O java.util.HashMap$Entry                                                           24
          ⊞ value ⇨ O org.apache.jasper.servlet.JspServletWrapper                                   80
          ⊞ <class> ⇨ C java.util.HashMap$Entry [Class]                                             64
          ⊞ key ⇨ O java.lang.String "/secure/admin/user/views/userbrowser.jsp"                     24
          ⊞ next ⇨ O java.util.HashMap$Entry                                                        24
        ⊞ [2] ⇨ O java.util.HashMap$Entry                                                           24
        ⊞ [8] ⇨ O java.util.HashMap$Entry                                                           24
        ⊞ [9] ⇨ O java.util.HashMap$Entry                                                           24
        ⊞ [14] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [15] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [16] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [17] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [18] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [19] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [20] ⇨ O java.util.HashMap$Entry                                                          24
        ⊞ [22] ⇨ O java.util.HashMap$Entry                                                          24
```

Paths from GC Roots: Alt+1 | Allocations: Alt+2

Paths from GC Roots to objects selected in the upper table

Show: shortest path

```
⊟ O org.apache.jasper.compiler.JspRuntimeContext
    rctxt of O org.apache.jasper.servlet.JspServlet [Stack Local]
```

http://jira.atlassian.com/browse/JRA-12524

TUNING XNAT

On Tuning

Tuning results dependent on many variables, what worked in one case may not work universally

XNAT is a complex system! Some parts are CPU bound, others are memory bound, and others are bandwidth bound

General rule:

faster CPUs + more RAM + bigger network pipe = faster XNAT

On Tuning

1. Find something that is “slow”
2. Quantify slowness
3. Tune
4. Quantify improvement
5. Go to #1

PostgreSQL Tuning

PostgreSQL 8.3 has serious performance improvements

Put PostgreSQL and Tomcat on separate machines

- Get powerful database machine
- PostgreSQL can take advantage of multiple cores & lots of RAM

postgresql.conf

Default settings designed for legacy machines

Increase available memory. Allows query planner to do more work in RAM and less on disk

Increase max connections

Tweak kernel settings to allow access to more memory

Tomcat Tuning

Increase available memory & use “server VM”

catalina.sh:

```
JAVA_OPTS="$JAVA_OPTS "-XX:MaxPermSize=256m" "-
  XX:PermSize=256m" "-mx1512m" "-server"
```

Tomcat Tuning

Increase connections & threads in *`server.xml`*

- At this point, consider load balancing between multiple Tomcat servers

XNAT Tuning

Upgrade to XNAT 1.4!

Increase `MaxConnections` to database in *`WEB-INF/conf/InstanceSettings.xml`*

– Set in line with PostgreSQL's `max_connections`

Tools Mentioned

Google Analytics: Free

Pingdom: Monthly subscription. One site free

Munin: Open Source

Monit: Open Source

JMeter: Open Source

YourKit: Commercial. Time-limited trial & free for open source

Questions & Your Experiences?

http://www.xnat.org/XNAT+2010+Workshop+-+Tuning,+Optimization,+Monitoring

XNAT