ArcGIS Server Performance and Scalability – Testing Methodologies
Andrew Sakowicz, Frank Pizzi
Target audience

- Testers
- Administrators (GIS, DBA, System)
- Developers
- Architects

- Level: Intermediate
Outline

Performance testing methodology

• Definitions
• Process
• Tuning
• Testing
  - Hardware
  - Application
• Capacity Planning
• Tools
Definitions

Performance

• The speed at which a given operation occurs
• E.g. Request response time measured in seconds
Definitions

Scalability

• The ability to increase output and maintain acceptable performance

• Examples
  - Capacity 10 maps/sec and response time 1 second
  - Capacity 1000 cars/hrs and speed 55 mph
Definitions

Capacity

- The maximum level of output the system can produce

At capacity

Over capacity
Definitions

Bottleneck

- **Resource(s) limiting the performance or capacity**

Think of:
- lanes - as CPU processor
- toll booths – as ArcGIS Server instances
- cars - as map requests
Definitions

Step Load and Response Time

Response Time (sec)

Step Load (users)

time
Definitions

Throughput (request/hr)

Throughput (req/hr)

Response Time (sec)

Step Load (users)

Throughput (req/hr)

Response Time (sec)

time
Definitions

Resource utilization: CPU, Memory, Network

- Throughput (req/hr)
- CPU Utilization (%)
- Network used (Mbps)
- Response Time (sec)
- Memory used (Mb)
- Step Load (users)

time
Definitions

Capacity

User load
Throughput (req/hr)
CPU Utilization (%)
Network used (Mbps)
Response Time (sec)
Memory used (Mb)
Content length (bytes)

Capacity (~ 85% utilization)
Process
Testing process

- Infrastructure: Hardware and Software
- GIS Services
- Application
Esri Process and Tools

Holistic approach

System Designer
http://www.arcgis.com/home/item.html?id=8ff490eef2794f428bde25b561226bda

For other tools beta evaluation, send email to
SystemTestTool@esri.com or SystemMonitorTool@esri.com
Required skill set
Configuration, Tuning, Testing
Tuning:
A reproducible test cases
Tuning methodology
Profile each tier starting from the top

- Browser
- Web Server
- ArcGIS Server
- ArcSOC
- ArcSDE/DBMS

Total Response Time (t1-t2)
Wait Time
Usage Time
Search & Retrieval Time
Fiddler
Fiddler measurement approximately 5.2 seconds
<table>
<thead>
<tr>
<th>Item</th>
<th>At Scale</th>
<th>Layer Name</th>
<th>Refresh Time (sec)</th>
<th>Recommendations</th>
<th>Features</th>
<th>Vertices</th>
<th>Labeling</th>
<th>Geography Phase (sec)</th>
<th>Graphics Phase (sec)</th>
<th>Cursor Phase (sec)</th>
<th>DBMS CPU</th>
<th>DBMS LIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>167,935,665</td>
<td>SDE.GridPoint</td>
<td>4.75</td>
<td>run DBMS trace; oraCPU=4.74; run DBMS trace, check oracle execution plan: oraLIO=130936; check if index exist for query def attributes;</td>
<td>1,998</td>
<td>False</td>
<td>4.74</td>
<td>.00</td>
<td>4.56</td>
<td>4.74</td>
<td>130,936</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DBMS LIO</th>
<th>DBMS PIO</th>
<th>Source</th>
<th>LayerType</th>
<th>Layer Spatial Reference</th>
<th>LayerQueryDef</th>
</tr>
</thead>
<tbody>
<tr>
<td>130,936</td>
<td></td>
<td>esriDBMS_Oracle, asakowicz, sde: oracle$Sasakowicz:1521/gis2, sde</td>
<td>esriGeometryPoint</td>
<td>GCS_WGS_1984</td>
<td>ID&lt;1000</td>
</tr>
</tbody>
</table>
DBMS Trace

http://www.arcgis.com/home/item.html?id=24c7b251159149848acc9b81ccc8356
**Oracle Trace**

**Compare elapsed time**

```
SELECT U_45.st_SHAPE$, U_45.OID, U_45.st_points, U_45.st_numpts,
       U_45.st_entity, U_45.st_minx, U_45.st_miny, U_45.st_maxx, U_45.st_maxy,
       U_45.st_mint, U_45.st_maxt, U_45.st_minz, U_45.st_maxz, U_45.st_mint, U_45.st_maxt, U_45.st_minz, U_45.st_maxz
FROM
  (SELECT b.OID, b.GX, b.GY, b.Z, b.ID, 1 st_SHAPE$, b.SHAPe.points as st_points,
       b.SHAPe.numpts as st_numpts, b.SHAPe.entity as st_entity, b.SHAPe.minx as st_minx, b.SHAPe.miny as st_miny,
       b.SHAPe.maxy as st_maxy, b.SHAPe.minz as st_minz, b.SHAPe.maxz as st_maxz
    FROM SDE.SI_EnIntersect ORS b
    WHERE SDE.SI_EnIntersect(b.SHAPe, 1, 12, 13, 14) = 1 AND b.OID NOT IN (SELECT /*
                                  HASH_AJ */ SDE.DELETES_ROW_ID FROM SDE.D4S WHERE DELETED_AT IN (SELECT
       1.lineage_id FROM SDE.state_lineages 1 WHERE 1.lineage_name = :
       lineage_name) AND 1.lineage_id (= :state_id1) AND SDE_STATE_ID = 0) UNION
      ALL SELECT a.OID, a.GX, a.GY, a.Z, a.ID, 1 st_SHAPE$, a.SHAPe.points as st_points,
       a.SHAPe.numpts as st_numpts, a.SHAPe.entity as st_entity, a.SHAPe.minx as st_minx, a.SHAPe.miny as st_miny,
       a.SHAPe.maxy as st_maxy, a.SHAPe.minz as st_minz, a.SHAPe.maxz as st_maxz
    FROM SDE.SI_EnIntersect(a.SHAPe, 3, 12, 13, 14) = 1 AND a.OID NOT IN (SELECT /*
                                  HASH_AJ */ SDE.DELETES_ROW_ID FROM SDE.D4S WHERE DELETED_AT IN (SELECT
       1.lineage_id FROM SDE.state_lineages 1 WHERE 1.lineage_name = :
       lineage_name) AND 1.lineage_id (= :state_id2) AND SDE_STATE_ID = 0) AND a.SDE_STATE_ID = SL.lineage_id AND SL.lineage_name = :
       lineage_name)
FROM
  (SELECT call, count, cpu, elapsed, disk, query, current, rows
   FROM
     Parse, Execute, Fetch
   GROUP BY call) AS query
ORDER BY elapsed
```

<table>
<thead>
<tr>
<th>call</th>
<th>count</th>
<th>cpu</th>
<th>elapsed</th>
<th>disk</th>
<th>query</th>
<th>current</th>
<th>rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Execute</td>
<td>1</td>
<td>0.03</td>
<td>0.02</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Fetch</td>
<td>20</td>
<td>9.67</td>
<td>9.66</td>
<td>0</td>
<td>129581</td>
<td>0</td>
<td>1998</td>
</tr>
</tbody>
</table>

**Elapsed time slightly changed due to different test runs**
Oracle Execution plan

Inefficient spatial index
Tuning:
No-reproducible test cases
Analyze Web server Logs
IIS log measurement 5.256 seconds
ArcGIS Server logs

Analyzing lots of entries might be challenging
Parse ArcGIS Server Logs

*AGS Log parser tool makes it easy*

- logparser "SELECT top 10 time, target, elapsed, message from 'C:\Program Files (x86)\ArcGIS\Server10.0\server\user\log\*.xml' order by elapsed desc" -i:COM -iprogid:lpx.ags_log -e 1 -q

---

<table>
<thead>
<tr>
<th>Time</th>
<th>Target</th>
<th>methodName</th>
<th>Elapsed</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:07</td>
<td></td>
<td>w</td>
<td>0</td>
<td>Feature count: 36792</td>
</tr>
<tr>
<td>8/19/2011 10:07</td>
<td>EOC.MapServer</td>
<td>UniqueValueRendrerd.Draw</td>
<td>0</td>
<td>End of layer draw: Roadway Cleared</td>
</tr>
<tr>
<td>8/19/2011 10:07</td>
<td>EOC.MapServer</td>
<td>FeatureLayer.Draw</td>
<td>0.00521</td>
<td></td>
</tr>
<tr>
<td>8/19/2011 10:07</td>
<td>EOC.MapServer</td>
<td>UniqueValueRendrerd.Draw</td>
<td>0</td>
<td>Feature count: 37143</td>
</tr>
</tbody>
</table>

[http://www.arcgis.com/home/item.html?id=5dfe54f1e9fd48068c4ae0c2c4f459c9](http://www.arcgis.com/home/item.html?id=5dfe54f1e9fd48068c4ae0c2c4f459c9)
Trend Analysis

System Metrics

System Monitor tool. Send email to SystemMonitorTool@esri.com for Beta evaluation.
# Trend Analysis

ArcGIS Server statistics

![ArcGIS Server Statistics](image)

## ArcGIS for Server

<table>
<thead>
<tr>
<th>Chart</th>
<th>Name</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>SampleWorldCities</td>
<td>STARTED</td>
</tr>
<tr>
<td>All</td>
<td>TestCaseFGDB</td>
<td>STARTED</td>
</tr>
<tr>
<td>All</td>
<td>TestCaseOracle</td>
<td>STARTED</td>
</tr>
<tr>
<td>All</td>
<td>TestCaseSQLServer</td>
<td>STOPPED</td>
</tr>
<tr>
<td>All</td>
<td>World_Street_Map</td>
<td>STARTED</td>
</tr>
</tbody>
</table>

- **Chart Options**: Throughput, Busy Time, Free Instances, Max Instances, Busy Instances

This image shows the Trend Analysis section of ArcGIS Server, highlighting statistics for various charts and their statuses.
## Trend Analysis

HTTP custom requests

**HTTP Requests:**

<table>
<thead>
<tr>
<th>Chart</th>
<th>Counter Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>Url-TestCaseOracleHTTPCode</td>
</tr>
<tr>
<td>View</td>
<td>Url-TestCaseOracleContentLength</td>
</tr>
<tr>
<td>View</td>
<td>Url-TestCaseOracleSec</td>
</tr>
</tbody>
</table>
## Trend Analysis

### Geodatabase statistics

<table>
<thead>
<tr>
<th>Chart</th>
<th>DB Host</th>
<th>Counter Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-UsersArcSDESessions</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-ArcSDEVersionCount</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-UsersOracleActiveSession</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-UsersArcSDEEditors</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-ArcSDECount</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-ArcSDEDCount</td>
</tr>
<tr>
<td>View</td>
<td>asakowicz</td>
<td>Query-ArcSDELineageDepth</td>
</tr>
</tbody>
</table>

Examples of versioned geodatabase stats
Demo

System Monitor
Hardware Testing
Network Testing
Important for cloud based solutions

Cloud vendors offer multiple location. Which one is the best for your global solution?

Single data center = lower cost

Performance depends on network: good bandwidth and low latency
Demo

Network Test

http://www.arcgis.com/home/item.html?id=2b8d4ac8b102453bb0437cdd7a07dd27
Network Test
Bandwidth and transport time

- Mbps - Bandwidth
- Mbits / req - Response size
- TH - Throughput (req/hr)

\[
Mbps = \frac{TH \times Mbits / req}{3600}
\]

\[
Transport(\text{sec}) = \frac{Mbits / req}{Mbps - Mbps_{\text{used}}}
\]

No need to calculate it manually, System Designer Tool does it for you:
http://www.arcgis.com/home/item.html?id=8ff490eef2794f428bde25b561226bda
## Network test

### Bandwidth and transport time

- **Factors:**
  - Image compression
  - Content, e.g., Vector vs. Raster
  - Return type, e.g., JPEG vs. PNG

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Service/Op</th>
<th>Content</th>
<th>Return Type</th>
<th>Mb/Tr</th>
<th>56 kbps</th>
<th>1.54 Mbps</th>
<th>10 Mbps</th>
<th>45 Mbps</th>
<th>100 Mbps</th>
<th>1 Gbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArcGIS Desktop</td>
<td>Map</td>
<td>Vector</td>
<td></td>
<td>10</td>
<td>0.056</td>
<td>1.540</td>
<td>10.000</td>
<td>45.000</td>
<td>100.000</td>
<td>1000.000</td>
</tr>
<tr>
<td>Citrix/ArcGIS</td>
<td>Map</td>
<td>Vector+Image</td>
<td>ICA Comp</td>
<td>1</td>
<td>178.571</td>
<td>6.494</td>
<td>1.000</td>
<td>0.222</td>
<td>0.100</td>
<td>0.010</td>
</tr>
<tr>
<td>Citrix/ArcGIS</td>
<td>Map</td>
<td>Vector</td>
<td>ICA Comp</td>
<td>0.3</td>
<td>5.357</td>
<td>0.195</td>
<td>0.030</td>
<td>0.007</td>
<td>0.003</td>
<td>0.000</td>
</tr>
<tr>
<td>ArcGIS Server</td>
<td>Map</td>
<td>Vector</td>
<td>PNG</td>
<td>1.5</td>
<td>26.786</td>
<td>0.974</td>
<td>0.150</td>
<td>0.033</td>
<td>0.015</td>
<td>0.002</td>
</tr>
<tr>
<td>ArcGIS Server</td>
<td>Image</td>
<td>JPG</td>
<td></td>
<td>0.3</td>
<td>5.357</td>
<td>0.195</td>
<td>0.030</td>
<td>0.007</td>
<td>0.003</td>
<td>0.000</td>
</tr>
<tr>
<td>ArcGIS Server</td>
<td>Map Cache</td>
<td>Vector</td>
<td>PNG</td>
<td>0.1</td>
<td>1.786</td>
<td>0.065</td>
<td>0.010</td>
<td>0.002</td>
<td>0.001</td>
<td>0.000</td>
</tr>
<tr>
<td>ArcGIS Server</td>
<td>Map Cache</td>
<td>Vector+Image</td>
<td>JPG</td>
<td>0.3</td>
<td>5.357</td>
<td>0.195</td>
<td>0.030</td>
<td>0.007</td>
<td>0.003</td>
<td>0.000</td>
</tr>
</tbody>
</table>
Hardware resources

- CPU processor
- Network bandwidth and latency
- Memory
- Disk

Most well-configured and tuned GIS systems are CPU processor bound.
Hardware resources
CPU Processor Speed

http://www.spec.org/cgi-bin/osgresults?conf=rint2006

http://www.cpubenchmark.net/cpu_list.php
Demo

Hardware Test

Application Testing
Testing Objectives

- Meet Service-Level Agreement (SLA)
- Bottlenecks analysis
- Capacity planning
- Benchmarking different alternatives
# Testing Tools

<table>
<thead>
<tr>
<th>Test Tools</th>
<th>Open Source</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
</table>
| LoadRunner             | No          | • Industry Leader  
• Automatic negative correlations identified with service-level agreements  
• HTTP web testing  
• Click and script  
• Very good tools for testing SOA  
• Test results stored in database  
• Thick client testing  
• Can be used for bottleneck analysis | • High cost  
• Test development in C programming language  
• Test metrics difficult to manage and correlate  
• Poor user community with few available examples                                                                                                                                                                   |
| Silk Performer         | No          | • Good solution for testing Citrix  
• Wizard-driven interface guides the user  
• Can be used for bottleneck analysis | • Moderate to high cost  
• Test metrics are poor.  
• Test development uses proprietary language.  
• Test metrics difficult to manage and correlate  
• Poor user community with few available examples                                                                                                                                                                   |
| Visual Studio Test Team| No          | • Low to moderate cost  
• Excellent test metric reporting  
• Test scripting in C# or VB .NET  
• Unit and web testing available  
• Blog support with good examples  
• Very good for bottleneck analysis | • No built-in support for AMF  
• No thick-client options  
• Moderate user community                                                                                                                                                                                                 |
| JMeter                 | Yes         | • Free  
• Tool | • Provides only response times  
• Poor user community with few available examples                                                                                                                                                                                                                             |
Testing steps

1. Validate environment
2. Design test
3. Validate results
Designing test

1. Profile requests
2. Prepare test data
3. Define transactions and requests
4. Bind test data to requests
5. Define a load test
Test data

Observe correlation between feature density and performance.
Demo

Testing GIS Service
System Test. For Beta evaluation, send email to SystemTestTool@esri.com
Analyze results

- Compare and correlate key measurements
  - Response Time Vs. Throughput
  - CPU, Network, Disk, and Memory on all tiers
  - Passed and Failed tests

- Validation
  - Lack of errors does not validate a test
  - Spot check request response content size
Analyze results
Valid

- *Expected* CPU and Response time correlation
Analyze results

Invalid

- Validation Example
  - Test failure – memory bottleneck in w3wp process
Capacity planning: using test results
Throughput (Req/sec)
CPU utilization

<table>
<thead>
<tr>
<th>Counter</th>
<th>Instance</th>
<th>Category</th>
<th>Computer</th>
<th>Color</th>
<th>Range</th>
<th>Min</th>
<th>Max</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>% Processor Time</td>
<td>Total</td>
<td>Processor</td>
<td>FPE227</td>
<td></td>
<td>100</td>
<td>18.9</td>
<td>26.7</td>
<td>20.8</td>
</tr>
<tr>
<td>Available MBytes</td>
<td>Memory</td>
<td>Memory</td>
<td>FPE227</td>
<td></td>
<td>1,000</td>
<td>740</td>
<td>904</td>
<td>816</td>
</tr>
</tbody>
</table>
Test Results as Input into Capacity Planning

• Throughput = 3.89 request/sec (14,004 request/hour)
• Response time = 0.25 seconds
• Average CPU Utilization = 20.8%
• Mb/request = 1.25 Mb
Test Results as Input into Capacity Planning

- **Input from testing**
  - #CPUs = 4 cores
  - %CPU = 20.8
  - TH = 14,004 requests/hour
  - SPEC per Core of machine tested = 35

- **ST** = \( \frac{4 \times 3600 \times 20.8}{14,004 \times 100} \) = 0.2138 sec
  - Note Service Time is very close to Average response time of 0.25

\[
ST = \frac{\#CPU \times 3600 \times \%CPU}{TH \times 100}
\]
Target values

1. Server SpecRate/core=10.1

![SPEC® CINT2006 Result](image)

2. User load=30,000 req/hr
3. Network=45 Mbps
Target values

Target CPU cores calculation

• Input to Capacity Planning:
  - ST = Service Time = .2138 sec
  - TH = Throughput desired = 30,000 request/hour
  - %CPU = Max CPU Utilization = 80%
  - SpecRatePerCpuBase = 35
  - SpecRatePerCpuTarget = 10.1

• Output
  - #CPU required = ( [.2138*30,000*100]/3600*80) *[35/10.1]
  - #CPU required = 7.7 cores ~ 8 cores

\[
#CPU_t = \frac{ST_b \times TH_t \times 100}{3600 \times %CPU_t} \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}
\]

No need to calculate it manually, System Designer Tool does it for you:
http://www.arcgis.com/home/item.html?id=8ff490eef2794f428bde25b561226bda
Target values
Target network calculation

• Input to Capacity Planning:
  - Mb/req=1.25
  - TH = 30,000 request/hour

• Output
  - Network bandwidth required = $30,000 \times 1.25 / 3600$
  - =10.4 Mbps < 45 Mbps available
  - Transport = $1.25 / (45 - 10.4) = 0.036$ sec

\[
Mbps = \frac{TH \times Mbits / req}{3600}
\]

\[
Transport (sec) = \frac{Mbits / req}{Mbps - Mbps_{used}}
\]
Test Results as Input into Capacity Planning

System Designer

• Input:
  - Throughput=30000
  - ST=0.21
  - Mb/tr=1.25
  - Hardware=80.9 Spec
Test Results as Input into Capacity Planning

System Designer

- **Input**
  - Hardware=80.9 Spec
Summary

Testing

- Process
- Skills
- Tools

Performance Testing

Sizing (Hardware and Software)

Performance Tuning
Questions?

Andrew Sakowicz, asakowicz@esri.com
Frank Pizzi, fpizzi@esri.com

Tools beta evaluation:
SystemTestTool@esri.com
SystemMonitor@esri.com

Please fill out your session surveys
http://esriurl.com/survey

Offering ID: 229