Maximo Performance Best Practices

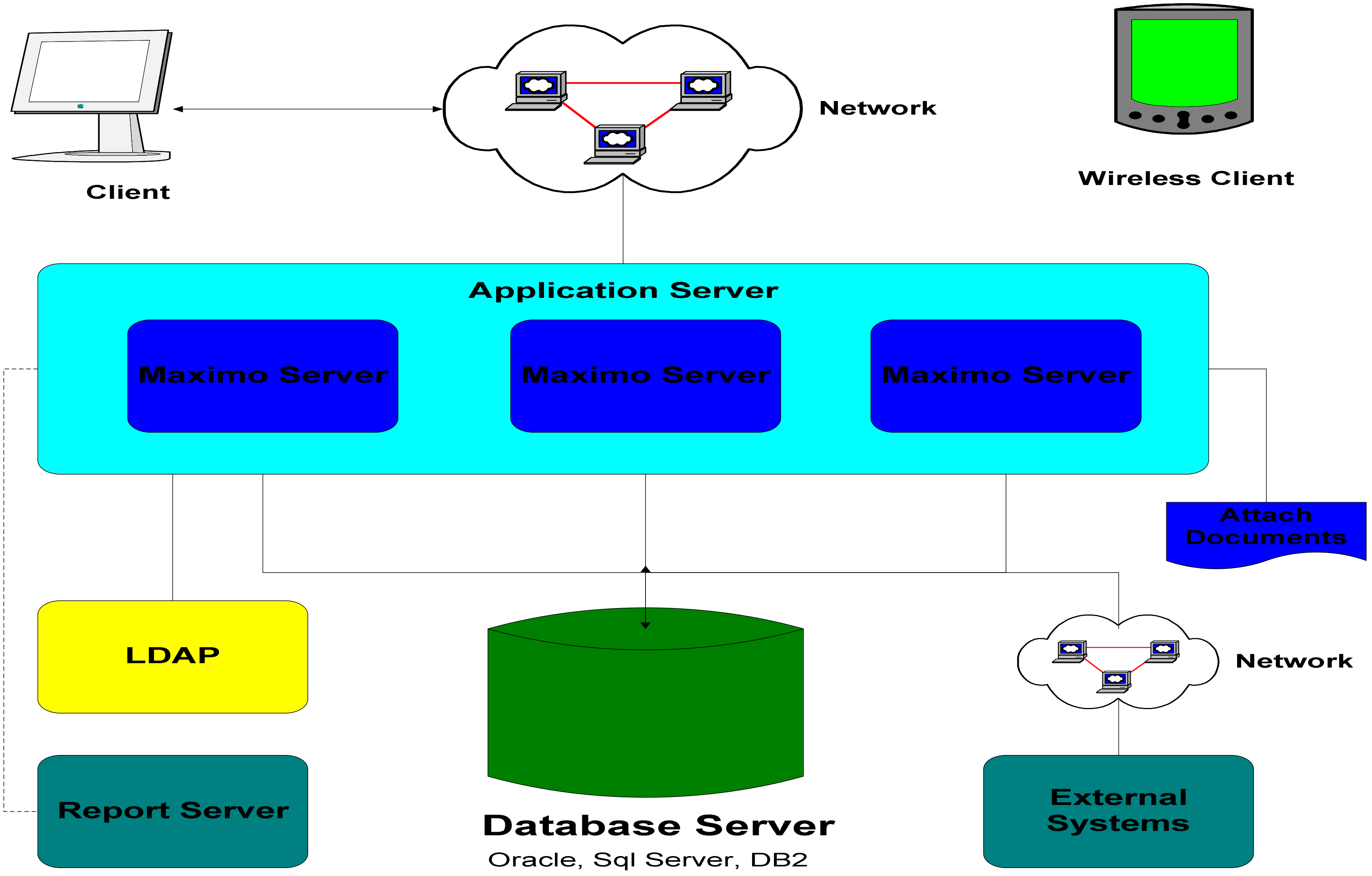
Overview

- Application Architecture
- Approaches to Performance Tuning
  - Sizing and Planning Pre-implementation (proactive)
  - Best Practices
  - Problem Resolution (Reactive – responding to issues)
- Proactive Approach
  - Hardware and Storage Sizing
  - Application Sizing, Software Clustering
  - End to End Tuning
- Best Practices
  - Best Practices Tuning
- Reactive Approach
  - Problem Identification
  - Successful Resolution Identification
- Questions

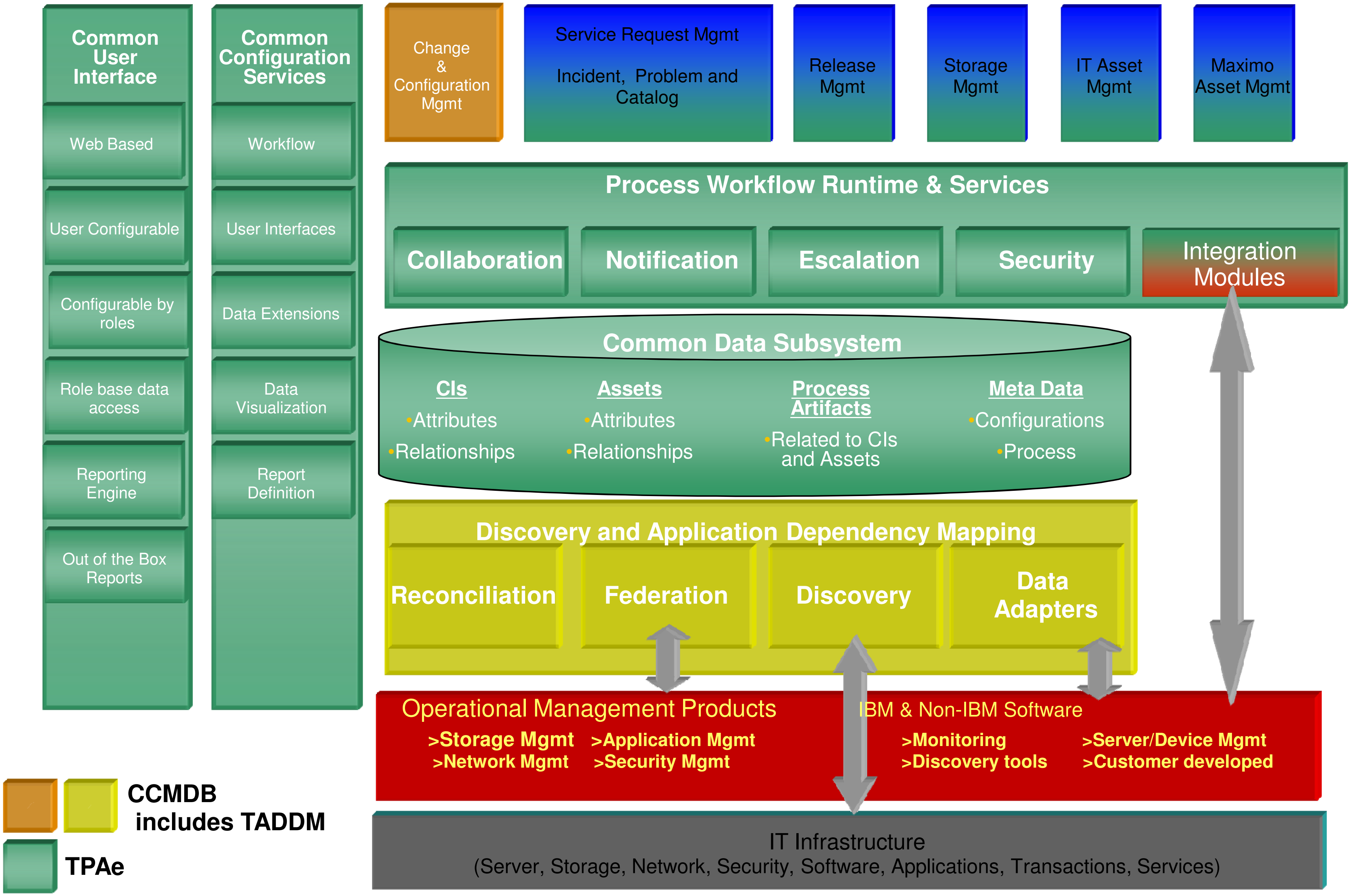
Application Architecture

Tivoli software

Application Architecture



Maximo Based Products



Common User Interface

- Web Based
- User Configurable
- Configurable by roles
- Role base data access
- Reporting Engine
- Out of the Box Reports

Common Configuration Services

- Workflow
- User Interfaces
- Data Extensions
- Data Visualization
- Report Definition

Change & Configuration Mgmt

Service Request Mgmt

Incident, Problem and Catalog

Release Mgmt

Storage Mgmt

IT Asset Mgmt

Maximo Asset Mgmt

Process Workflow Runtime & Services

Collaboration | **Notification** | **Escalation** | **Security** | Integration Modules

Common Data Subsystem

CIs
- Attributes
- Relationships

Assets
- Attributes
- Relationships

Process Artifacts
- Related to CIs and Assets

Meta Data
- Configurations
- Process

Discovery and Application Dependency Mapping

Reconciliation | **Federation** | **Discovery** | **Data Adapters**

Operational Management Products

>Storage Mgmt **>Application Mgmt**
>Network Mgmt **>Security Mgmt**

IBM & Non-IBM Software

>Monitoring **>Server/Device Mgmt**
>Discovery tools **>Customer developed**

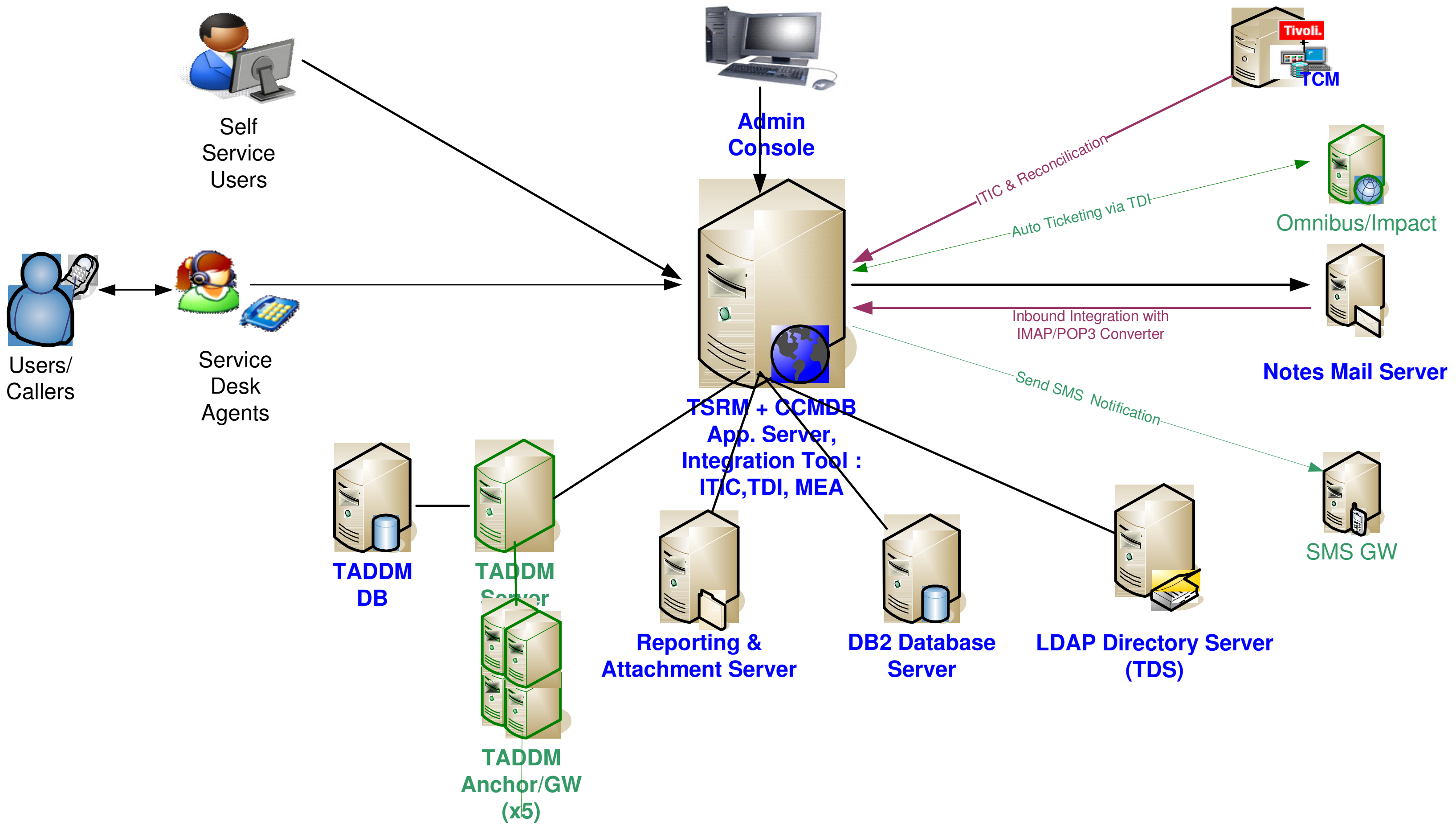
IT Infrastructure
(Server, Storage, Network, Security, Software, Applications, Transactions, Services)

🟧🟨 **CCMDB includes TADDM**

🟩 **TPAe**

Application Architecture

- Sample Production System



Approaches to Performance Tuning

Tivoli software

Approaches to Performance Tuning

- Proactive
  - Usually done during roll out to prepare for go live loads
  - Planning hardware for required loads and throughput
  - Clustering for required loads and fault tolerance
  - Planning storage requirements
  - Implementation Options

- Best Practices
  - Implementation experiences
  - UAT Testing
  - Load Testing

- Reactive
  - Usually done in response to a problem. (this is the least preferable option as it means that users have already been impacted)
  - Expensive and time consuming
  - Focuses on analysis and most likely cause

Proactive Performance Sizing and Planning

Tivoli software

Proactive Performance Sizing and Planning

- Size your environment
  - Hardware Sizing
    - Application Server
    - Database Server
    - Reporting and LDAP Server
  - Application Server Sizing
    - JVM
    - Memory – Both physical memory and JVM memory
  - Integration Component Sizing
    - Transaction rates
    - Clustering
  - Database Sizing
    - Disk Usage sizing
    - CPU sizing
    - Database memory

Proactive Performance Sizing and Planning

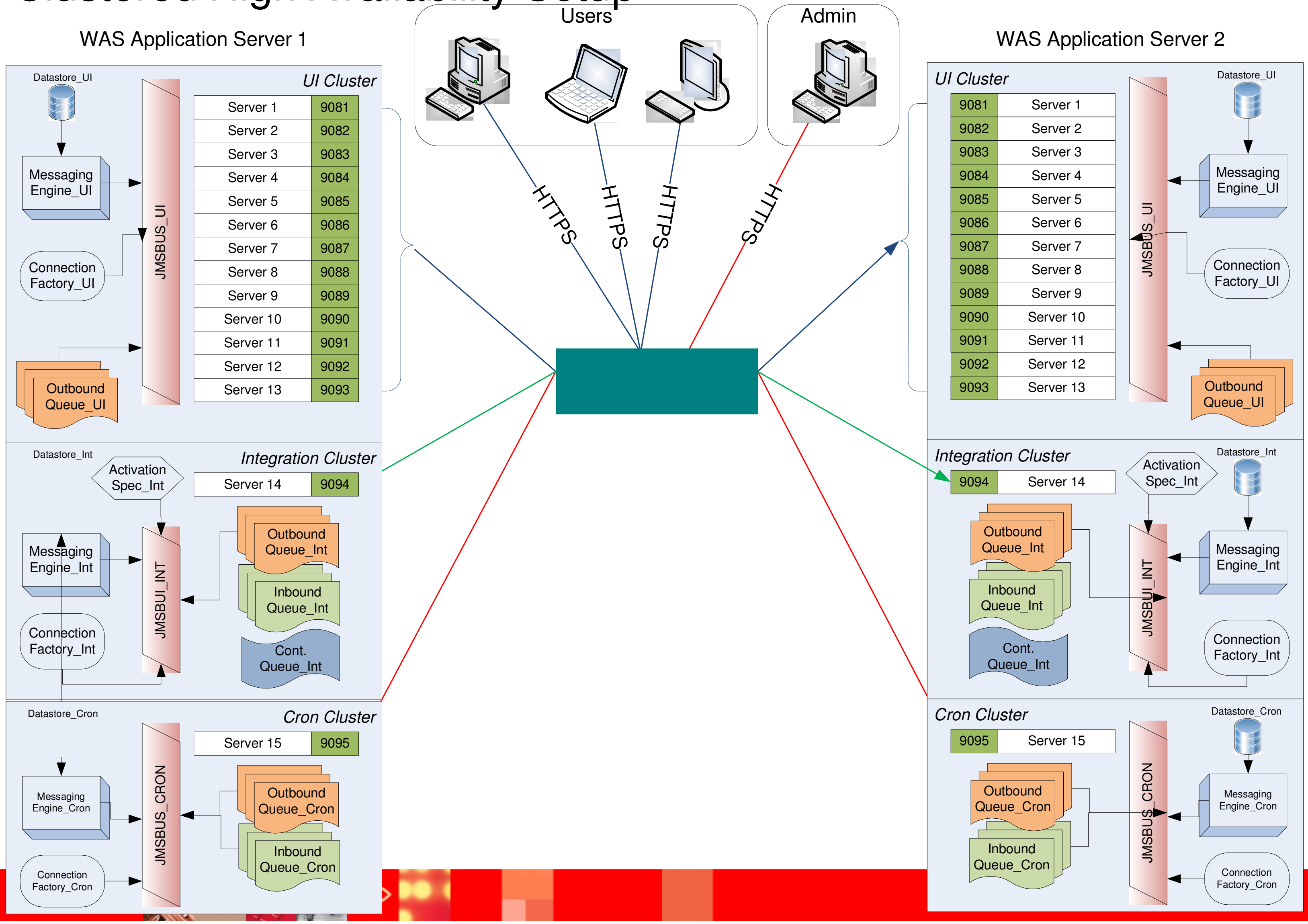
- Application Server Sizing and Setup
  - Application Servers should always be setup in a cluster
    - Cluster member JVM’s should be sized based on the user load
  - Why is clustering so important?
  - Various clusters of the system
    - UI Cluster - Supports end users.
    - Crontask Cluster - Supports background tasks
    - Integration Cluster - Supports integration transactions
    - Reporting Cluster - Supports reporting in a high performance, high availability mode

Proactive Performance Sizing and Planning

- Keep JVM and Application Server fix packs updated
  - Within the same version, updated to latest fix packs to take advantage of application server fixes
- 32-bit environment versus 64-bit environment
  - 32-bit environment has memory limitations for JVM
  - Due to memory limitations, server stability becomes an issue
  - Convert 32-bit JVM environment to 64-bit environment

Proactive Performance Sizing and Planning

- Clustered High Availability Setup



WAS Application Server 1 – UI Cluster

Server 1	9081
Server 2	9082
Server 3	9083
Server 4	9084
Server 5	9085
Server 6	9086
Server 7	9087
Server 8	9088
Server 9	9089
Server 10	9090
Server 11	9091
Server 12	9092
Server 13	9093

Integration Cluster

Server 14	9094

Cron Cluster

Server 15	9095

WAS Application Server 2 – UI Cluster

9081	Server 1
9082	Server 2
9083	Server 3
9084	Server 4
9085	Server 5
9086	Server 6
9087	Server 7
9088	Server 8
9089	Server 9
9090	Server 10
9091	Server 11
9092	Server 12
9093	Server 13

Integration Cluster

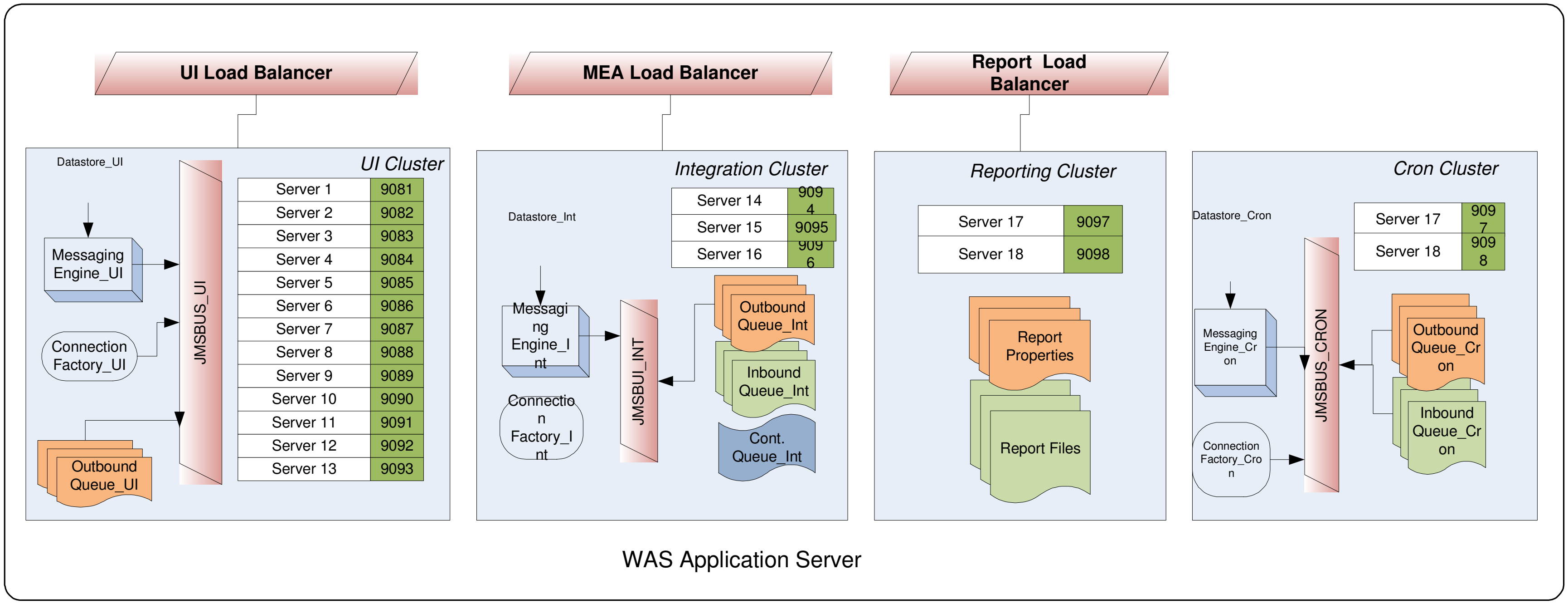
9094	Server 14

Cron Cluster

9095	Server 15

Proactive Performance Sizing and Planning

- High Performance Clustering
  - Create separate clusters to gain high throughput
    - UI, MEA, Reporting, Cron task
    - Transactions get load balanced across multiple servers



Server 1	9081
Server 2	9082
Server 3	9083
Server 4	9084
Server 5	9085
Server 6	9086
Server 7	9087
Server 8	9088
Server 9	9089
Server 10	9090
Server 11	9091
Server 12	9092
Server 13	9093

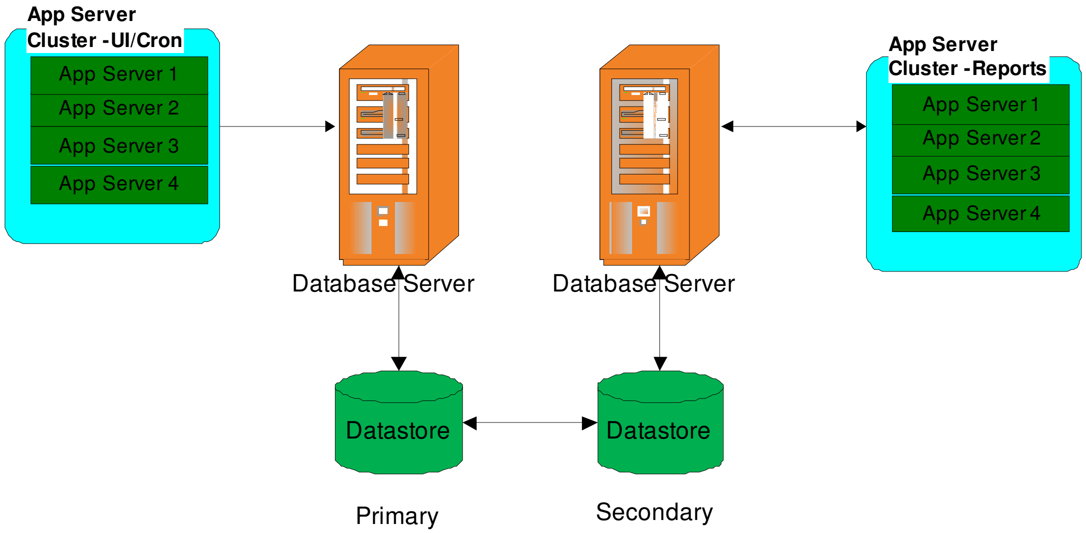
Server 14	9094
Server 15	9095
Server 16	9096

Server 17	9097
Server 18	9098

Server 17	9097
Server 18	9098

Proactive Performance Sizing and Planning

- High Availability Database Setup
  - Platform Specific
    - DB2 provides HACMP and HADR
    - Oracle has RAC
    - SQL Server High Availability
  - High Availability helps in failover
  - Separate reporting database helps reduce load on the primary database.



HADR Setup

Best Practices

Tivoli software

Best Practices

- JVM clusters for like functionality (UI, Integration, Cron, Reports, etc.)
- Application Server/JVM settings
- User queries and search types
- Reporting database
- IBM HTTP Server settings
- Database connection pooling properties
- Database parameters/indexes/statistics
- Network – acceleration/compression/cache
- Load test before going live
  - If adding new sites or increasing user load, repeat load test
- Maximo 7 Performance Best Practices
  - http://www-01.ibm.com/support/docview.wss?uid=swg21440192
- Best practice tip
  - Keep a spreadsheet of your entire setup information

Application Server/JVM Best Practices

- JVM clusters for like functionality (UI, Integration, Cron, Report, etc.)
- JVM heap memory settings are very important and play a big role in application server performance
  - Heap memory setting varies between 32 bit application servers and 64 bit application servers
  - 32 bit – Maximum is 3GB based on the platform. Windows 32 but machine can only handle 1.8GB of heap for a JVM
  - 64 bit – You can increase the heap memory.
  - Initial heap size and Max heap size
  - What is the best setting?
- -Dsun.rmi.dgc.ackTimeout=10000
  - Extremely important setting for Maximo due to heavy use of java rmi objects
  - Without this setting, maximo objects are held in memory for longer duration, which can lead to
    - Out of Memory situations
    - Too many database connections

Database Best Practices

- Tune database as per install and best practices documents
- Database tuning gives the best ROI for performance
  - Database is the most critical part of Maximo application
- Database specific settings are documented in install guide and best practices document
  - Oracle – cursor sharing, increase SGA
  - DB2 – lock timeout, memory settings
  - SQL Server – turn off page level locking, Maximo properties for SQL Server
- Analyze database memory and user I/O
  - Maximo fetches lot of data into application server
  - Increase system memory. Helps reduce user I/O
- Separate tablespaces and mount points to optimize I/O
- Educate users on ad-hoc queries
  - Check for newly created user queries and optimize them
- Reduce start center data retrieval
  - Keep the default start center to be a simple minimal data fetch
    - There can be multiple start center, but the first one displayed should be simple to avoid excessive data fetch

Database Best Practices

- Setup an index statistics update schedule
  - Weekly update helps in better performance
- Purge / Archive data
  - Transactional data needs archiving plus purging
  - Non-transaction data should be purged on a regular basis
  - E.g. Workflow transaction data, login tracking, start center
- Add indexes as they are needed
  - Do not be scared to add indexes
  - In Maximo, 80% is data fetch, 20% is data insert/update
  - Lack of right index causes more performance problem than slowing down on insert/update
- If possible, change WILD card searches to EXACT and educate users to use % wild card option in search fields
  - This reduces un-necessary like searches and greatly improves database query timings

Database Best Practices

- Use Maximo tool – Integrity Checker to find Maximo schema problems that can affect applications as well as performance
  - Integrity Checker can be found in the Maximo tools folder
  - Integrity Checker should be done in development phase to production phase

Report Performance Best Practice

- Decide whether to have separate report server cluster or use the UI cluster for reporting
- Each one has its own pros and cons. Both are fine as long as they are configured right
- UI Cluster for reporting
  - Pros
    - Able to utilize the UI JVM's for reporting
    - Less complex setup
    - Less Infrastructure needs
  - Cons
    - May experience user response degradation due to very large / complex reports taking up JVM resources
- BROS cluster for reporting
  - Pros
    - Separate JVM's provide better response
  - Cons
    - More complex setup
    - More infrastructure needs

Report Performance Best Practice

- BROS option
  - Helpful when using a separate reporting database
- Identify long running reports from report usage and tune them
  - Can identify poor performing ad hoc reports
- Scheduled report cron tasks
  - Separate these out of UI servers
  - mxe.crontask.donotrun=REPORTSCHEDULE
  - No need to explicitly mention the instance name, just put REPORTSCHEDULE in the do not run list to separate them out of UI server
- BIRT Reporting Performance document link
  - https://www-304.ibm.com/support/entdocview.wss?uid=swg21305031

Logging Best Practices

- Keep Maximo as well as application server logging to minimal level unless investigating a problem
  - Set everything except Maximo root logger to ERROR
  - Maximo logging adds 5% + overhead
- Clean up log folders from filling up
  - Watch out for heap dumps as these files are very large
- Since logging changes can be applied dynamically, turn logging on when needed and turn it off when done
- For SQL performance analysis
  - Use logSQLTimeLimit , extremely useful
  - Collect them in separate log files
- If analyzing memory usage
  - Use verbose GC logging. Clean up old log files
  - Use Garbage Collection and Memory Visualizer to analyze

Maximo Built in Tools to Help Troubleshoot

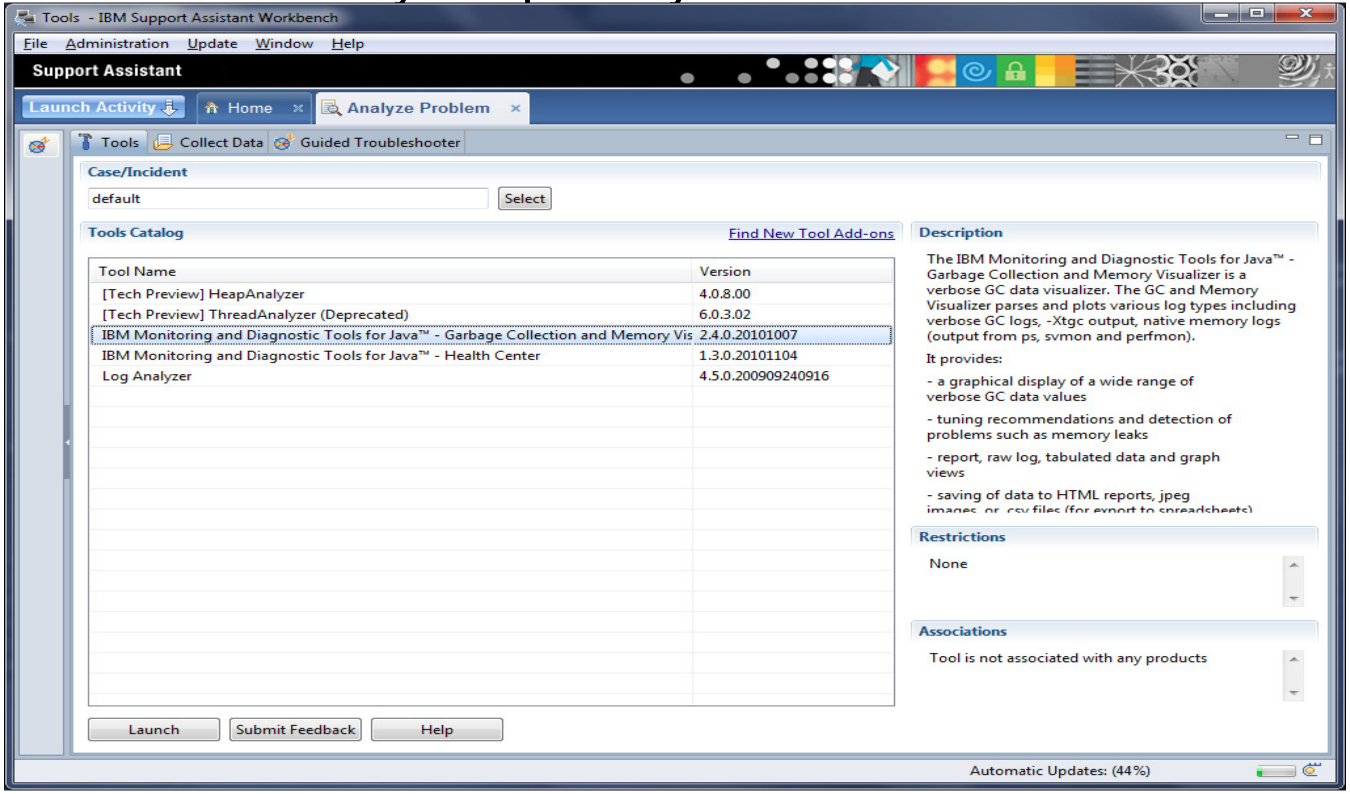
- FetchStopLimit
  - Enable and use until you reach a stable state. Returns an error to the end user. Can collect the information from the log and contact support
    - mxe.db.fetchStopLimitEnabled
    - mxe.db.fetchResultStopLimit
    - mxe.db.fetchResultStopLimit.OBJECTNAME
    - mxe.db.fetchStopExclusion
    - Using fetch stop limits to prevent out-of-memory errors
    - https://www-304.ibm.com/support/docview.wss?uid=swg21412865
- FetchResultLogLimit
  - This greatly helps in identifying excessive data fetch
  - Use this to identify the code that is fetching excessive data
  - Depending on data or code fix, adjust the data or get a code fix

Maximo Tools to Help Troubleshoot

- System Properties used as part of logging
  - Enabled through System Properties application. Dynamic, can be enabled without re-building application ear file
  - SQL logging to collect all SQL statements – Logging Application
  - Use the following system properties to collect useful log information
    - mxe.mbocount (5.2 P05 and later)
    - mxe.db.fetchResultLogLimit (5.2 P02A and later)
    - mxe.db.logSQLTimeLimit (5.2 P02A and later)
    - mxe.db.logSQLPlan (6.0 P01 and later; Oracle only)
    - mxe.db.sqlTableScanExclude (6.0 P01 and later; Oracle only)
- Use Maximo tool – Integrity Checker to find Maximo schema problems that can affect applications as well as performance
  - Integrity Checker can be found in the Maximo tools folder
  - Integrity Check should be done in development phase to production phase

IBM Support Assistant

- Available from IBM Support Site
- Contains many add-on tools
- Useful for memory heap analysis and JVM thread



Case/Incident

default

Tools Catalog

Tool Name	Version
[Tech Preview] HeapAnalyzer	4.0.8.00
[Tech Preview] ThreadAnalyzer (Deprecated)	6.0.3.02
IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Vis	2.4.0.20101007
IBM Monitoring and Diagnostic Tools for Java™ - Health Center	1.3.0.20101104
Log Analyzer	4.5.0.200909240916

Description

The IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Visualizer is a verbose GC data visualizer. The GC and Memory Visualizer parses and plots various log types including verbose GC logs, -Xtgc output, native memory logs (output from ps, svmon and perfmon).

It provides:

- a graphical display of a wide range of verbose GC data values

- tuning recommendations and detection of problems such as memory leaks

- report, raw log, tabulated data and graph views

- saving of data to HTML reports, jpeg images, or .csv files (for export to spreadsheets)

Restrictions

None

Associations

Tool is not associated with any products

Automatic Updates: (44%)

Best Practices

- Implementation Options
  - RMI Server Deployment – Deploy RMIREG.WAR
  - Search Methodologies – Default to “EXACT”
  - User Training – Best practices and procedures for tasks
  - Network Appliances – Riverbed, Juniper, Citrix
  - Task Focused Screen Design – Create small screen
  - Off Hours WO Generation, Reorder, Crons, and Reporting
  - Client Browser Configuration – Page refresh to Automatic

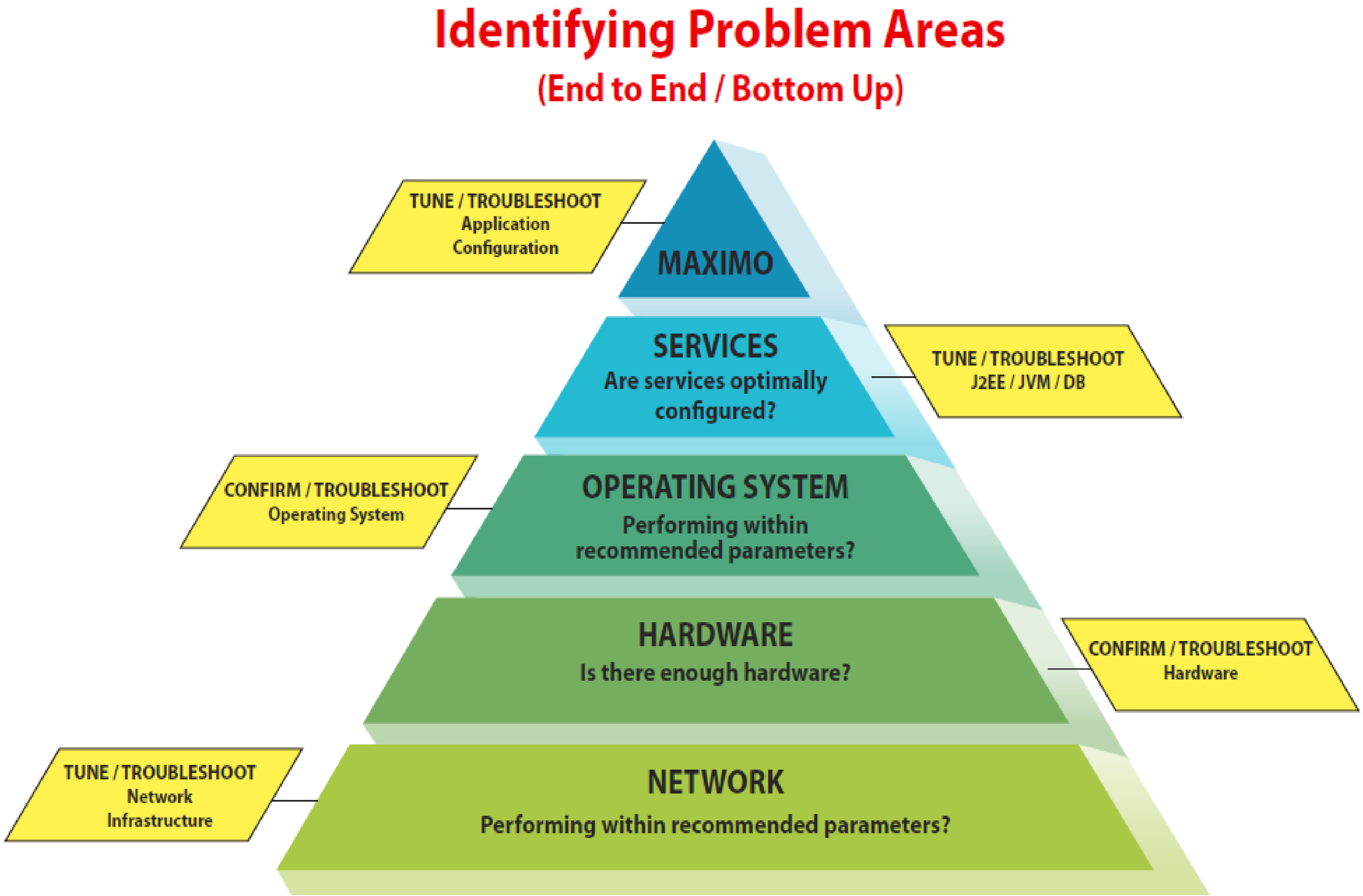
Reactive Performance Tuning

Tivoli software

Reactive Performance Tuning

- Define problem, resolution, and focus
  - Analyze and clearly define reported problem
    - Reported problems are often not clear. Understand what is being reported. "Application is slow" is not a definition. Find out what is slow. See it for yourself if possible.
  - Determine/define what is "successful resolution"
    - Don't allow client perception to interfere with a solution. Get a clear definition of what is acceptable so you have a measure for success.
  - Focus on technologies that might impact issue
    - If users at one plant are affected but users at another plant are not affected, it is likely not a server problem. Focus on items that can impact the affected area.

Reactive Performance Tuning



Identifying Problem Areas

(End to End / Bottom Up)

MAXIMO

TUNE / TROUBLESHOOT
Application
Configuration

SERVICES
Are services optimally configured?

TUNE / TROUBLESHOOT
J2EE / JVM / DB

OPERATING SYSTEM
Performing within recommended parameters?

CONFIRM / TROUBLESHOOT
Operating System

HARDWARE
Is there enough hardware?

CONFIRM / TROUBLESHOOT
Hardware

NETWORK
Performing within recommended parameters?

TUNE / TROUBLESHOOT
Network
Infrastructure

Reactive Performance Tuning

- Indentifying Problem Areas
  - Application Physical Server
  - Operating System (AIX,HP-UX,Linux,Solaris,Windows)
  - **Java Application Server (WebSphere/WebLogic)**
  - Java Virtual Machine
  - **Application (Maximo, CCMDB, SRM, TAMIT, etc)**
  - Database Physical Server
  - Operating System
  - **Database Server (DB2, Oracle, SQL Server)**
  - **Network Connections (DB and Client)**
  - Client Physical Equipment
  - **Client Browser Configuration**

- *** Bold items most common with highest impact**

Maximo Index Analysis

- Use database tools to find long running queries and recommended indexes
- In Maximo, use Maximo logging to collect long running queries and analyze index usage
  - If needed alter the query to better utilize existing indexes
  - Enable mxe.db.logSQLTimeLimit = 1000 to collect all SQL statements that are longer than 1 second
  - Extract from logs and order by longest time consuming queries
  - Run them through SQL development tools and add required indexes
  - Always make sure the indexes are added through Maximo DB Configure to retain during fixpack application and upgrades
  - Add non-standard indexes through scripts and run the script after each database configuration (configdb) operation

Search Methodology Change

- Converting WILDCARD search to EXACT match has resulted in better performance
- Application users need to be educated
  - Change impact on them
  - How it improves the performance
  - Resulting benefits
- This change improves
  - Query retrieval times
  - Reduces data volume
  - Improves the overall database response
- End users can still use %value% to achieve WILDCARD searches
  - The default behavior is set to EXACT
- Change LIKE functionality in stored queries and custom relationships

Reactive Performance Tuning

- **Maximo Logging and Activity Dashboard**
  - **Activity Dashboard - Used to be called as performance monitor**
    - Not a monitor, but a debugging tool
  - **Available by default from Maximo 7.1.1.6 onwards**
    - It was introduced during Maximo 5.2, but is part of the product now
  - **Enable in development / test / breakfix area to identify the problem**
    - Enabled through web.xml and requires a ear file rebuild and re-deployment
    - Do not use in production environment, uses lot of memory and will bring down the server
  - **You can identify the SQL statements for a transaction and the time taken**
    - Can also identify for the origin of the sql statement
    - SQL timings are noted
    - Object counts are noted

Maximo Performance Monitor Configuration

https://www-304.ibm.com/support/docview.wss?uid=swg21448706

Support Documents

Application Server

- IBM WebSphere Tuning (can cause application to hold objects too long if not set correctly)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21261874
- Oracle WebLogic Tuning
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21261853

Network Performance

- Network caching and compression properties
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21262009
- Browser Caching (Helps with Wide Area Network -WAN- performance and high latency)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21292557

Support Documents

Database Server Performance

- All SQL Server Environments (applies to all SQL Server 7, 2000,2005)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21261979
- SQL Server 2000 with Maximo 6.2 and above (required additional parameters)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21313428
- SQL Server 2000 with Maximo 6.1 and below (correct parameter settings)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21295983
- SQL Server 2005 (correct parameter settings)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21296072
- SQL Server 2005 (row versioning)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21440598

Support Documents

Database Server Performance

- Oracle Cursor Sharing (All Oracle Releases)
  - http://www.ibm.com/support/docview.wss?rs=3214&uid=swg21262959
- DB2 Performance
  - https://www-304.ibm.com/support/docview.wss?uid=swg21421645
- DB2 Parameter Tuning
  - https://www-304.ibm.com/support/docview.wss?uid=swg21451593

Support Documents

Troubleshooting

- Debug properties
  - http://www-01.ibm.com/support/docview.wss?uid=swg21291250
- Must Gather – General Information
  - http://www-01.ibm.com/support/docview.wss?uid=swg21313647
- Must Gather – Performance
  - http://www-01.ibm.com/support/docview.wss?uid=swg21313341
- Logging
  - http://www-01.ibm.com/support/docview.wss?uid=swg21264064
- Logging Appenders
  - http://www-01.ibm.com/support/docview.wss?uid=swg21446599

Support Documents

Best Practices

- Maximo 5 and 6 Performance Best Practices
- http://www-01.ibm.com/support/docview.wss?uid=swg21395387
- Maximo 7.1 Performance Best Practices
  - http://www-01.ibm.com/support/docview.wss?uid=swg21440192
- Maximo 7.5 Performance Best Practices
  - http://www.ibm.com/support/docview.wss?uid=swg21591070
- Maximo Wiki
  - http://www.ibm.com/developerworks/wikis/display/maximo/Home