TxMQ MQ Services

Custom Tools, Scripts and Accelerators from TxMQ's Services Team

http://txmq.com/support-options/

Presenters:
Miles Roty – VP of Services & Support
Rob Lee – MQ Architect

TxMQ's Story

Founded in 1979

International footprint

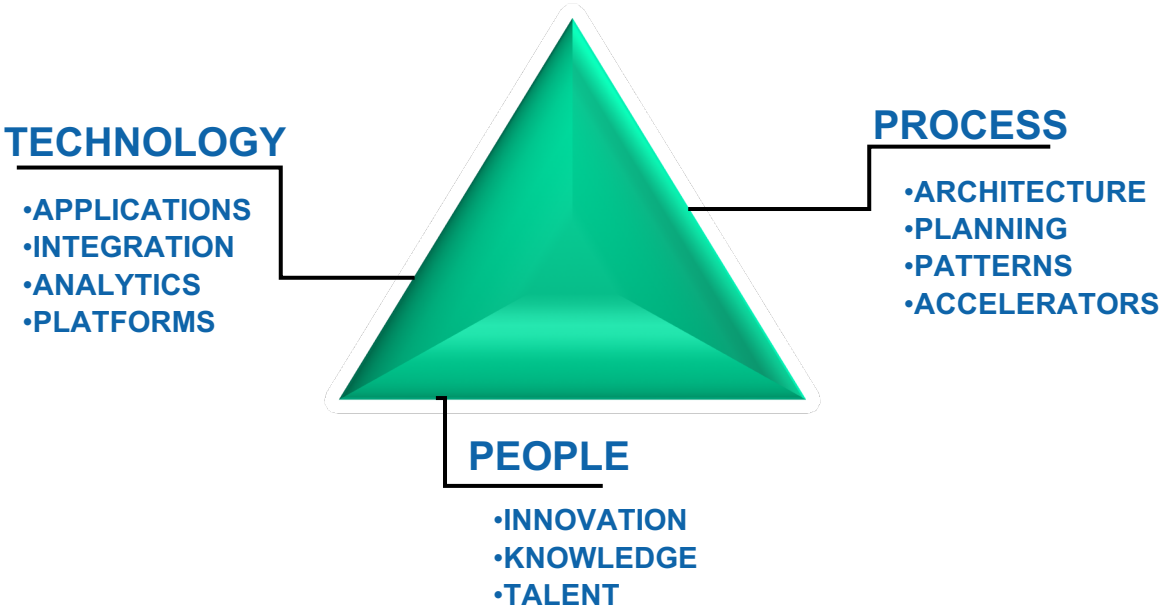
Hundreds of successful client partnerships

Trusted 'go to' partner of record for dozens of large system integrators and software companies

Partnerships & Support for IBM, Oracle, VMWare, Microsoft, Red Hat, and more

TxMQ Solution Approach

At TxMQ, we embrace the balance of *People, Process and Technology* to ensure your projects are successful



Our Background & Capabilities

- **Deep Expertise in IBM Middleware**
  - MQ & MQ Advanced (MFT/FTE, MQTT, AMS–Security)
  - IIB/App Connect
  - DataPower
  - API Connect
- **Strategic and Hands-on Technical Support**
  - Architecture
  - Administration
  - Troubleshooting & Production Support
  - Development Support

Middleware (IBM MQ) Team Tasks

- Build & Deploy
- Monitor & Manage
- Troubleshoot & Recover
- Discover & Document
- Evaluate & Test

* All these common tasks offer opportunity for automation

Middleware (IBM MQ) Automation

- **Build & Deploy –** Queue Manager Build & Deployment Scripts
  - Automation Scripts to create new Queue Managers
  - Deployment Scripts for QMGR's in Prod and lower environments
  - AIX, Windows, z/OS, AS400, Linux
  - On-premise and Cloud
  - VM's/Hypervisors and bare metal
  - Custom scripts and 3rd party/OpenSource automation & deployment tools

Middleware (IBM MQ) Automation

- **Monitor & Manage –** Tools to view and monitor MQ traffic and related system health
  - Is MQ Explorer sufficient?
  - Do we really *need* end-to-end transaction monitoring?
  - What are right tools for your environment?
  - Is it time for Tool Consolidation?
  - Custom Dashboards
  - Custom Administration Utilities
  - Log file management & queries

Middleware (IBM MQ) Automation

- **Troubleshoot & Recover –** Scripts to diagnose system issues, and for recovery of MQ Servers and messages
  - MQ is the problem…or is it?
  - Root cause analysis
  - Recovering Queue Managers
  - Recovering messages after outage
  - Log file recovery & review

Middleware (IBM MQ) Automation

- **Discover & Document –** Scripts to identify MQ Objects, utilities to capture statistics, and documentation
  - Objects
  - Queues
  - Queue Managers
  - Usage & Performance statistics
  - MQ Network Topology Diagrams, Graphs and other MQ Documentation

Middleware (IBM MQ) Automation

- **Discover & Document –** Scripts to identify MQ Objects, utilities to capture statistics, and documentation
  - Objects
  - Queues
  - Queue Managers
  - Usage & Performance statistics
  - MQ Network Topology Diagrams, Graphs and other MQ Documentation

Middleware (IBM MQ) Automation

- **Evaluate & Test –** TxMQ's QPacity™ and test scripts
  - QPacity™ enables administrators to measure usage and capacity of your MQ infrastructure
  - QPacity™ scripts support troubleshooting efforts to identify root cause of performance degradation and outages
  - TxMQ supports customers with development of custom test harnesses for new or poorly performing applications

Benefits from Automation

- Repeatability
- Expedites rollouts
- Ensures consistency
- Meets Standards
  - IBM Best Practices
  - Corporate/Industry
  - Security
- Supports DevOps

Custom Tools, Scripts and Accelerators

- BatchSudo
- buildMQ z/OS
- MQ Healthcheck Scorecard

Custom Tools, Scripts and Accelerators

BatchSudo:

This is a java program created as a result of not having access to Blade Logic or similar DevOps tools.

Using it, we can specify an xml structure containing a list of commands that need to be invoked on N-number of servers. It uses ssh/sftp under the covers to run commands as “root” and captures their stderr/stdout to a local file, named per server name. It’s also multi-threaded, and can chunk through a large list of servers efficiently.

Locally, the commands a placed in temp script file, this file is then sftp’d to targeted servers, it’s permission’s changed and made executable.
In principle it’s similar to “Ansible”, but less complicated.

Custom Tools, Scripts and Accelerators

BatchSudo:

This is a simple illustration, in the following input XML, we issue three commands (whoami, df, dspmq) on two servers (*.137 and *.138):

```
<?xml version="1.0"?>
<Config>
        <sudoCmds>
          whoami
          df
        </sudoCmds>
        <servers>
         192.168.27.137
         192.168.27.138
        </servers>
</Config>
```

Custom Tools, Scripts and Accelerators

BatchSudo:

This would result in the following two log files:

192.168.27.137.log

```
09/10/2018 16:20:44.942 192.168.27.137 - /----------------------------------------------\
09/10/2018 16:20:44.943 192.168.27.137 - pool-1-thread-1: begin
09/10/2018 16:20:45.813 192.168.27.137 - Command(s) to execute:
09/10/2018 16:20:45.813 192.168.27.137 - 1) #!/bin/bash
09/10/2018 16:20:45.813 192.168.27.137 - 2) ## ------------------------------------------------------------
09/10/2018 16:20:45.814 192.168.27.137 - 3) ## - mwscript3125373176329139146.sh
09/10/2018 16:20:45.814 192.168.27.137 - 4) ## - support: rob.lee@hcahealthcare.com
09/10/2018 16:20:45.814 192.168.27.137 - 5) ## - executed by: admin
09/10/2018 16:20:45.814 192.168.27.137 - 6) ## ------------------------------------------------------------
09/10/2018 16:20:45.814 192.168.27.137 - 7)
09/10/2018 16:20:45.814 192.168.27.137 - 8) whoami
09/10/2018 16:20:45.814 192.168.27.137 - 9) rc=$?; if [[ $rc != 0 ]]; then rm mwscript3125373176329139146.sh &&
exit $rc; fi
09/10/2018 16:20:45.815 192.168.27.137 - 10) df
09/10/2018 16:20:45.815 192.168.27.137 - 11) rc=$?; if [[ $rc != 0 ]]; then rm mwscript3125373176329139146.sh &&
exit $rc; fi
09/10/2018 16:20:45.815 192.168.27.137 - 12) rm mwscript3125373176329139146.sh
09/10/2018 16:20:45.815 192.168.27.137 -

Connecting to [192.168.27.137]:
09/10/2018 16:20:47.126 192.168.27.137 - root
09/10/2018 16:20:47.126 192.168.27.137 - Filesystem     1K-blocks      Used Available Use% Mounted on
09/10/2018 16:20:47.126 192.168.27.137 - /dev/sda3       18555904  13889144   4666760  75% /
09/10/2018 16:20:47.126 192.168.27.137 - devtmpfs         1917996         0   1917996   0% /dev
09/10/2018 16:20:47.127 192.168.27.137 - tmpfs            1932652         0   1932652   0% /dev/shm
09/10/2018 16:20:47.127 192.168.27.137 - tmpfs            1932652      9020   1923632   1% /run
09/10/2018 16:20:47.127 192.168.27.137 - tmpfs            1932652         0   1932652   0% /sys/fs/cgroup
09/10/2018 16:20:47.127 192.168.27.137 - /dev/sda1         303780    179120    124660  59% /boot
09/10/2018 16:20:47.127 192.168.27.137 - .host:/        976759804 775558988 201200816  80% /mnt/hgfs
09/10/2018 16:20:47.127 192.168.27.137 - tmpfs             386532         0    386532   0% /run/user/996
09/10/2018 16:20:47.128 192.168.27.137 - tmpfs             386532        24    386508   1% /run/user/1000
09/10/2018 16:20:47.128 192.168.27.137 - exit-status: 0

09/10/2018 16:20:47.129 192.168.27.137 - Elapsed Time: 0 min, 2 sec, 186 ms
09/10/2018 16:20:47.129 192.168.27.137 - pool-1-thread-1: end
09/10/2018 16:20:47.129 192.168.27.137 - \----------------------------------------------/
```

192.168.27.138.log

```
09/10/2018 16:20:44.943 192.168.27.138 - /----------------------------------------------\
09/10/2018 16:20:44.943 192.168.27.138 - pool-1-thread-2: begin
09/10/2018 16:20:45.813 192.168.27.138 - Command(s) to execute:
09/10/2018 16:20:45.813 192.168.27.138 - 1) #!/bin/bash
09/10/2018 16:20:45.813 192.168.27.138 - 2) ## ------------------------------------------------------------
09/10/2018 16:20:45.814 192.168.27.138 - 3) ## - mwscript3125373176329139146.sh
09/10/2018 16:20:45.814 192.168.27.138 - 4) ## - support: rob.lee@hcahealthcare.com
09/10/2018 16:20:45.814 192.168.27.138 - 5) ## - executed by: admin
09/10/2018 16:20:45.814 192.168.27.138 - 6) ## ------------------------------------------------------------
09/10/2018 16:20:45.814 192.168.27.138 - 7)
09/10/2018 16:20:45.814 192.168.27.138 - 8) whoami
09/10/2018 16:20:45.814 192.168.27.138 - 9) rc=$?; if [[ $rc != 0 ]]; then rm
mwscript3125373176329139146.sh && exit $rc; fi
09/10/2018 16:20:45.815 192.168.27.138 - 10) df
09/10/2018 16:20:45.815 192.168.27.138 - 11) rc=$?; if [[ $rc != 0 ]]; then rm
mwscript3125373176329139146.sh && exit $rc; fi
09/10/2018 16:20:45.815 192.168.27.138 - 12) rm mwscript3125373176329139146.sh
09/10/2018 16:20:45.815 192.168.27.138 -

Connecting to [192.168.27.138]:
09/10/2018 16:20:47.140 192.168.27.138 - root
09/10/2018 16:20:47.140 192.168.27.138 - Filesystem     1K-blocks    Used Available Use% Mounted on
09/10/2018 16:20:47.140 192.168.27.138 - /dev/sda3       18555904 8929064   9626840  49% /
09/10/2018 16:20:47.140 192.168.27.138 - devtmpfs         1917952       0   1917952   0% /dev
09/10/2018 16:20:47.140 192.168.27.138 - tmpfs            1932652       0   1932652   0% /dev/shm
09/10/2018 16:20:47.141 192.168.27.138 - tmpfs            1932652    8932   1923720   1% /run
09/10/2018 16:20:47.141 192.168.27.138 - tmpfs            1932652       0   1932652   0% /sys/fs/cgroup
09/10/2018 16:20:47.141 192.168.27.138 - /dev/sda1         303780  179288    124492  60% /boot
09/10/2018 16:20:47.141 192.168.27.138 - tmpfs             386532      32    386500   1% /run/user/1000
09/10/2018 16:20:47.141 192.168.27.138 - exit-status: 0

09/10/2018 16:20:47.142 192.168.27.138 - Elapsed Time: 0 min, 2 sec, 199 ms
09/10/2018 16:20:47.142 192.168.27.138 - pool-1-thread-2: end
09/10/2018 16:20:47.142 192.168.27.138 - \----------------------------------------------/
```

Custom Tools, Scripts and Accelerators

buildMQzOS:

Utility to create Qmgrs on z/OS by processes a single parameter file, build jobs and procs, and submit them into FTP's JES.

- Eliminates mis-typing of parms and debugging efforts involved
- Accurate, Efficient and Fast
- Could have easily been written in any number of languages or scripts.
- I chose vbscript to because it's most non-native to MVS

Custom Tools, Scripts and Accelerators

buildMQzOS:

1. Customize INI Parm file
2. Run buildMQzOS script

buildMQzOS processes a sub-directory of template procs and jobs,
substituting custom parms into a new set of procs/jobs and submits them in order.

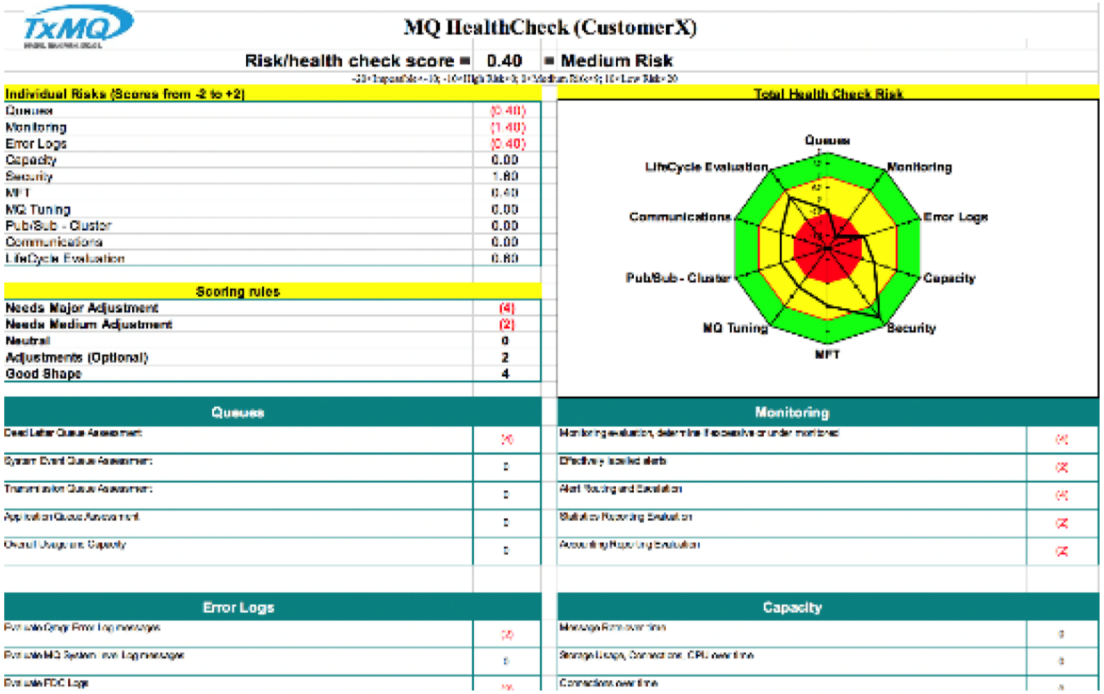
Jobs Created:

- xxxxBSDS - Creates Bootstrap and log Data Sets
- xxxxTXMQPAGE - Allocate the page set data sets Defines you pagesets:
- xxxxDGDG - If Exist Delete Generation Data Group / Re-create
- xxxxZPRM - Assemble and link a new system parameter module

Custom Tools, Scripts and Accelerators

MQHealthcheck Scorecard:

This is a template (excel spreadsheet) organized and designed to help facilitate thorough evaluations of a client's QMgr Systems.



TxMQ

MQ HealthCheck (CustomerX)

Risk/health check score = 0.40 = Medium Risk

Individual Risks (Scores from -2 to +2)	
Queues	(0.40)
Monitoring	(1.40)
Error Logs	(0.40)
Capacity	0.00
Security	1.60
MFT	0.40
MQ Tuning	0.00
Pub/Sub - Cluster	0.00
Communications	0.00
LifeCycle Evaluation	0.80

Scoring rules	
Needs Major Adjustment	(4)
Needs Medium Adjustment	(2)
Neutral	0
Adjustments (Optional)	2
Good Shape	4

Total Health Check Risk

Queues	
Dead Letter Queue Assessment:	(4)
System Event Queue Assessment:	0
Transmission Queue Assessment:	0
Application Queue Assessment	0
Overall Usage vs. Capacity	0

Monitoring	
Monitoring evaluation, determine if excessive or under monitored	(4)
Effectively scaled alerts	(2)
Alert Routing and Escalation	(4)
Statistics Recording Evaluation	(2)
Accounting Recording Evaluation	(2)

Error Logs	
Evaluate Qmgr Error Log messages	(2)
Evaluate MQ System Error Log messages	0
Evaluate FDC Logs	(4)

Capacity	
Message Retention	0
Storage Usage, Connection, CPU, overtime	0
Connection overtime	0

Digital Business Transformation

We help companies **evolve** their **technologies** to meet the demands of the **digital economy.**

- **Application Integration**
- **API Economy**
- **Business Process Management**
- **Automated Decision Management**
- **Robotic Process Automation**
- **Mobility**

Decades of Experience

TxMQ's deep experience in Enterprise Architecture provides the foundation for our solutions

- **Business Impact**
  - Improve efficiency
  - New revenue opportunities
  - Time to value
- **Enterprise Grade**
  - Security & Stability
  - Performance & Scalability
  - Integration

TxMQ Account & Services Team

Miles Roty
VP, Services and Support

Rob Lee
Middleware Specialist

Cindy Gregoire
Practice Manager, Middleware

Contact Us

TxMQ Inc.
501 John James Audubon Parkway, Suite 103
Amherst, NY 14228
http://txmq.com

Miles Roty – VP of Services & Support
p| 716-636-0070 x228
m| 415-535-3960
miles@txmq.com