HDT-4470: Automating Automation, Cheaply

InterConnect 2017

Michael Freeman, Sr. Applications Development Analyst
- mlfreeman@gmail.com
- michael.freeman@bkfs.com

Andrew Detandt, Systems Manager
- andy.detandt@bkfs.com

Session Overview

- Who Are We
- History
- Hardware
- Software
- Lessons Learned
- Questions

Who is BKFS?

- Based in Jacksonville, FL
- Formerly known as Lender Processing Services until 2014
- Largest single provider of technology, services, data, and analytics to the U.S. mortgage industry.

History

- 2009
  - Recruited to help start automated testing team for the Servicing Technologies division.
  - Decided to use RFT after evaluating competition
  - Lab would have to run multiple versions of Windows and IE
    - XP + IE 7, XP + IE 8, Win 7 + IE8, Win 7 + IE9, etc.
    - Only viable option: virtual machines.

History

- 2010 – First lab: XenServer on 10 desktops
  - Free, worked, but with limits
    - Performance wasn't the best
    - Thin provisioning of VMs wasn't available
- 2011 – Second lab: VMware ESXi Free Edition – same machines
  - Thin provisioning enabled us to put more VMs in.
  - Free version could only be managed manually.
- 2011/2012 – Third lab: Archipel VM orchestrator + Fedora 15
  - Free, polished UI, but the UI was slow and the backend had issues
  - Gave us ideas for later though

How was all this managed?

- Perl
- More Perl
- Manual Labor
  - Major tasks (e.g. new RFT version)
  - Deciding which scripts run on which machines

Is this painful?

Yes!

Better Way

- Our initial vision was simple: "Automate Automation"
  - Try to automate as many of our tasks as we could so we could just keep coding.
- We saw what you could do with cloud companies.
  - Cost and security restrictions ruled out actually using them.
- Why didn't we use an existing open source VM package?
  - Complexity of setup
  - Automation doesn't need a lot of what they offer

The Present Lab

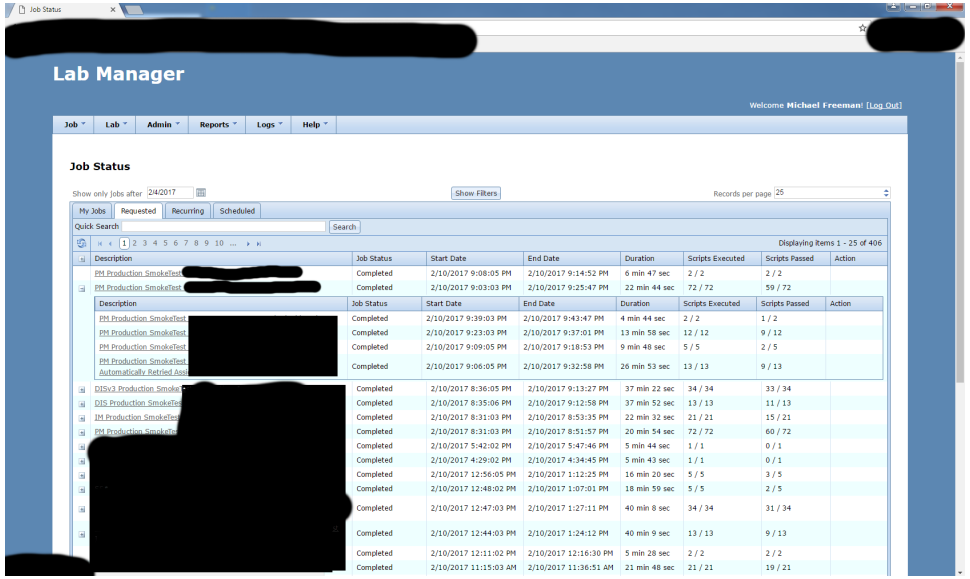
- Multiple levels of labs
  - "Main" lab: 168 possible VMs up across 14 hosts
    - Hosts all have 32 GB RAM and SSDs (256/512 GB)
    - All automated testing for other groups goes on here
  - "Dev" labs
    - Older hardware (previous main-lab machines)
    - Team tests scripts on here before they go live
  - All lab virtual machines use standardized Windows images

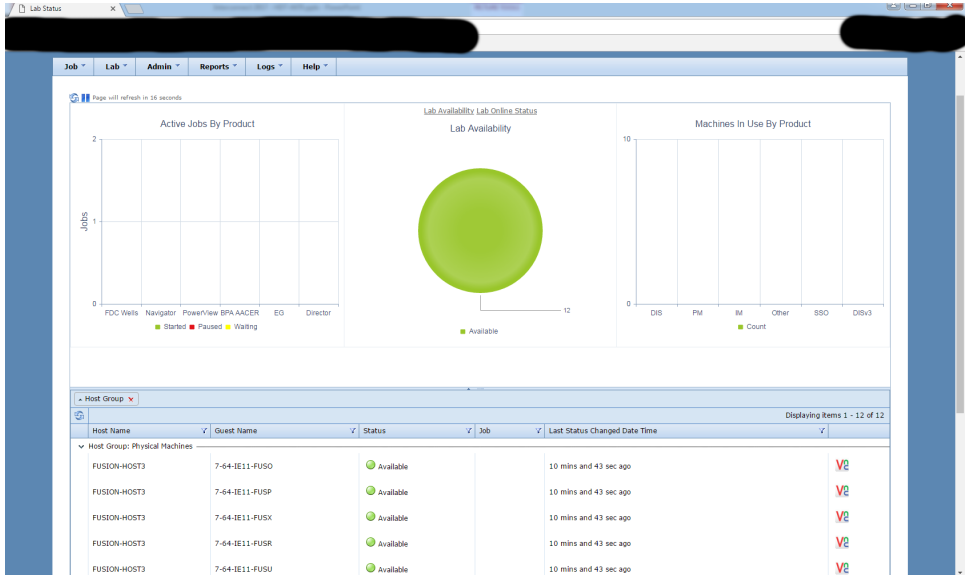
Lab Software

- Created In-House
  - Lab Manager
    - Logging Framework
    - RQM Bridge
  - Virtual Machine Manager
- Not Created In House
  - RFT
  - RQM
  - Linux/QEMU/KVM/libvirtd
  - Windows

Lab Manager

- Started on it in 2011/2012
- First one of our current tools
- Simple goal:
  Automate script execution
- Perl scripts were limited
  - Required manual load balancing of scripts
  - Required coordination among team members



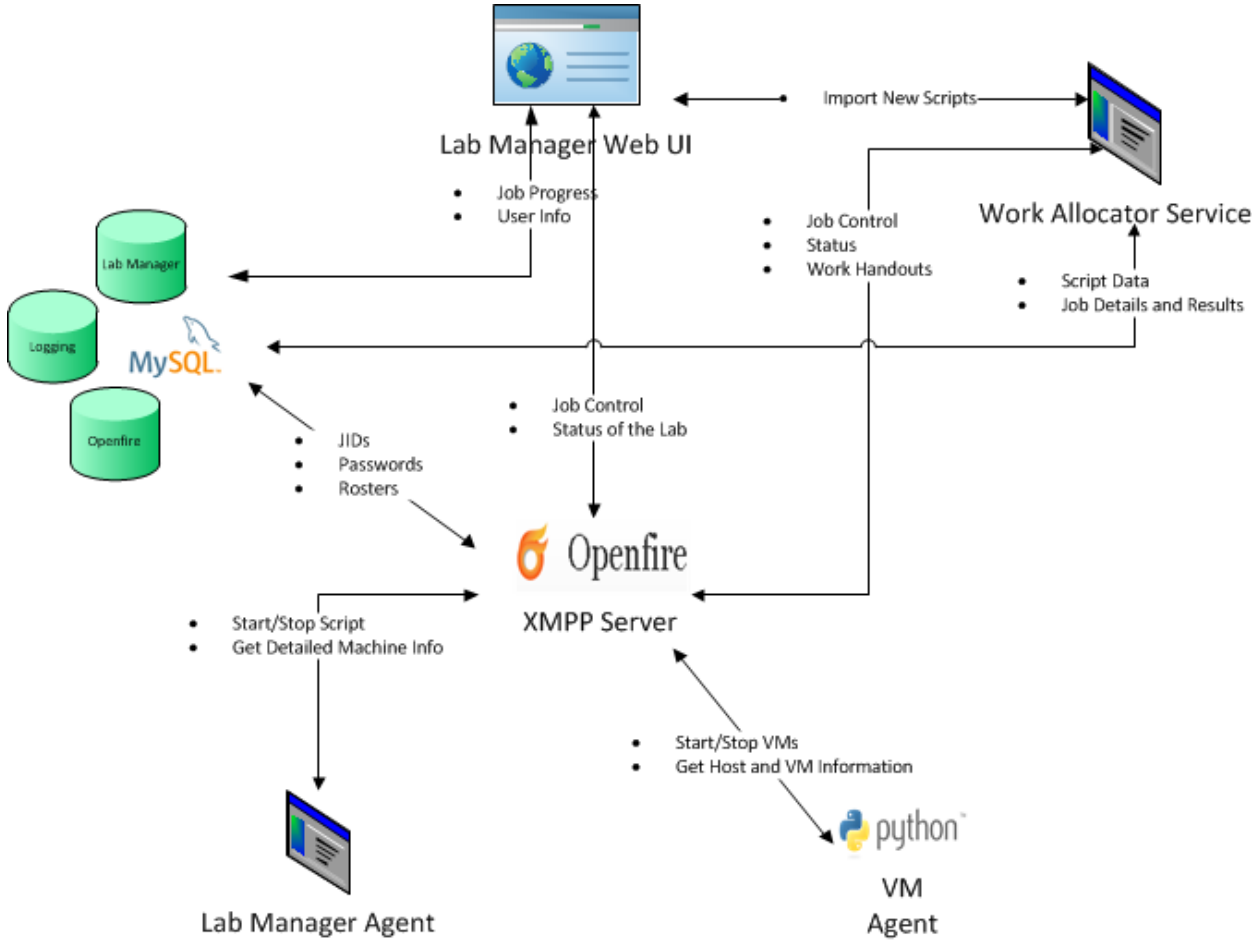


Lab Manager

- We looked at RQM automation support but it was limited too
  - Can't share lab machine across multiple RQM projects
    - Have to manually connect to said machine and point it to other project
  - When we started, Test Automation Cells didn't exist in RQM yet.
    - You had to explicitly specify exactly which machine to send scripts to.

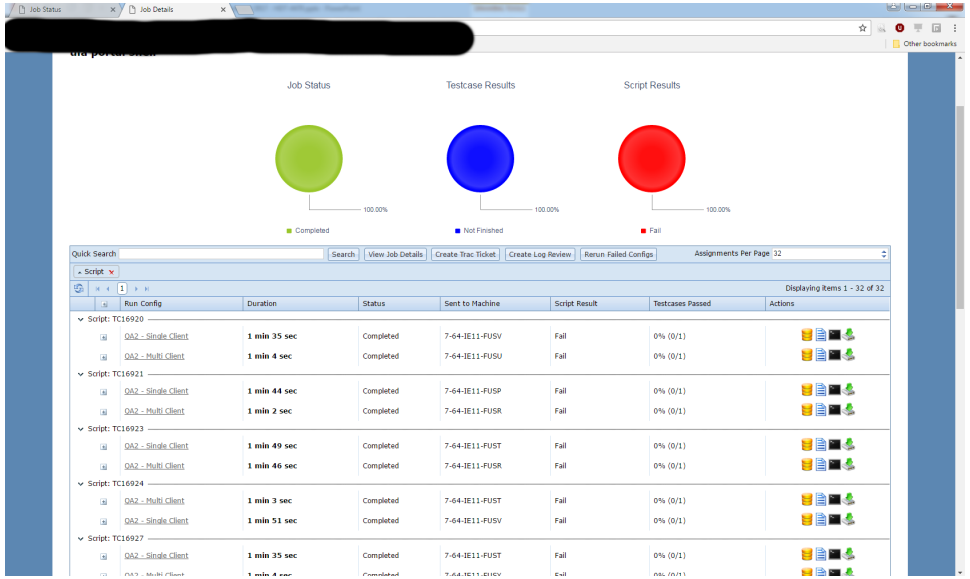
Lab Manager

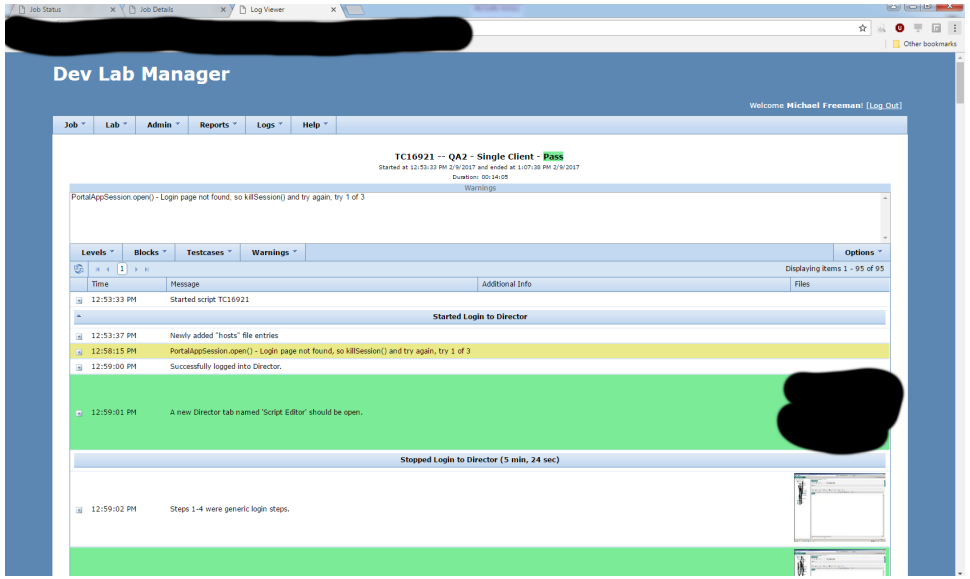
- Multi-tier system
  - Web UI
  - Service API
  - Backend Service
  - Agents on every VM and physical machine
  - Eclipse plugin
- Uses XMPP as a service bus of sorts



Lab Manager

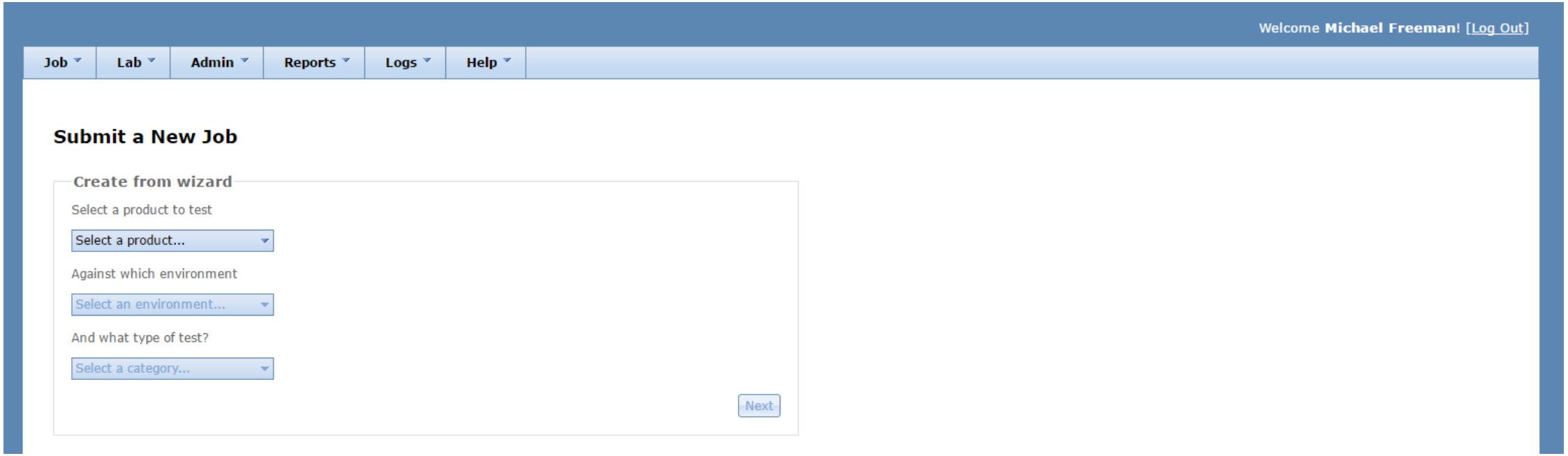
- Multiple execution requests can be submitted at the same time.
    - Even for different Windows/IE versions
- System monitors lab and finds open spaces to run scripts
    - Turns on/off Windows VMs based on demand.
- Common execution scenarios (e.g. regression for product X) can be saved for quick re-use.





Lab Manager

- E-mail notifications when work requests finish.
- Service API allows tools such as UrbanCode Deploy to automatically request work
    - (e.g. Run smoke test automatically after build is deployed)



Lab Manager – Overall

- Overall, everyone gets their work done faster.

Product	Type of Test	Execution Times		Time Saved
		Last release w/out lab manager	First release with lab manager	
A	*Smoke*	36 minutes	15 min 46 sec	20 minutes
	Regression	2 hours, 48 minutes	2 hours, 13 minutes and 15 seconds	35 minutes
B	*Smoke*	43 minutes	19 minutes and 18 seconds	24 minutes
	Regression	2 hours, 41 minutes	1 hrs 19 min 2 sec	1 hour, 22 minutes

- Most importantly, **automation scripts can be run and reviewed by other teams on their own.**

Lab Manager – Logging Framework

- As part of all this, we created a custom object-based logging API
  - We need more than just free-form text describing what happened.
    - Screenshots, uploaded/downloaded/generated files, test variables, etc
  - We output to multiple formats
    - 2 forms of HTML, and one PDF layout now
    - Script writers are not allowed to think about what it looks like in the log
  - Structured objects force script writers provide a baseline level of detail
    - Free-form text APIs (e.g. log4j) can let people forget to include info

- Logging DB currently holds 2+ years of data
  - 850,000 script runs
  - 45-50 million log entries
  - 1 TB+ of distinct files (screenshots, etc) based on SHA-512 checksum

RQM Bridge

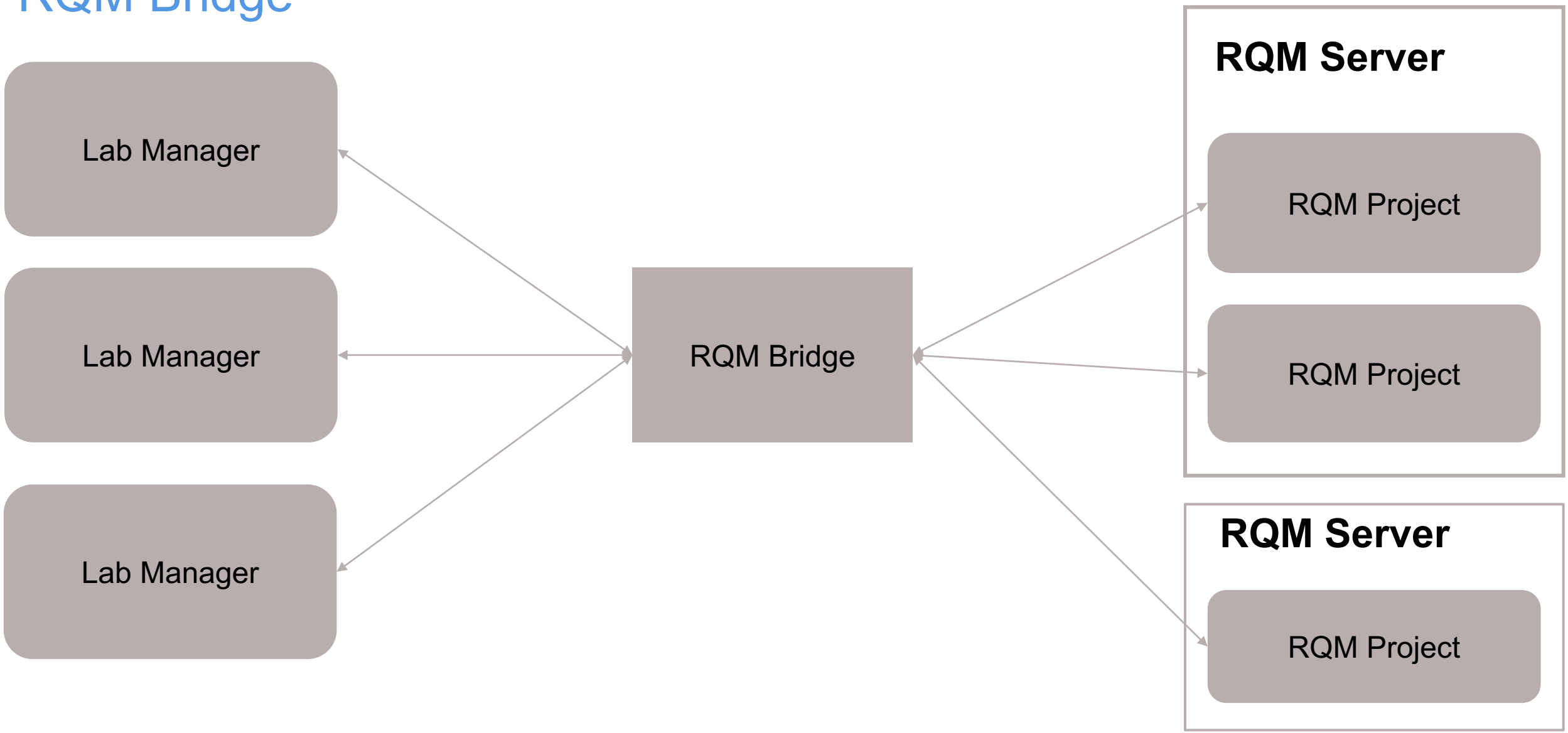
- 2014: Another automation team was merged into ours.
- Previous QA groups working with them were used to the out of the box RQM process.
- We wanted to keep as much of that as we could and still fit it in to our system.

I wrote a bridge to connect the two.

RQM Bridge

- Java application
- Powered by OSLC4J
- Based on guide on jazz.net https://jazz.net/wiki/bin/view/Main/RQMTestAutomationAdapterAPI
- RQM Bridge can dispatch work from multiple RQM projects to one Lab Manager and intelligently track everything.

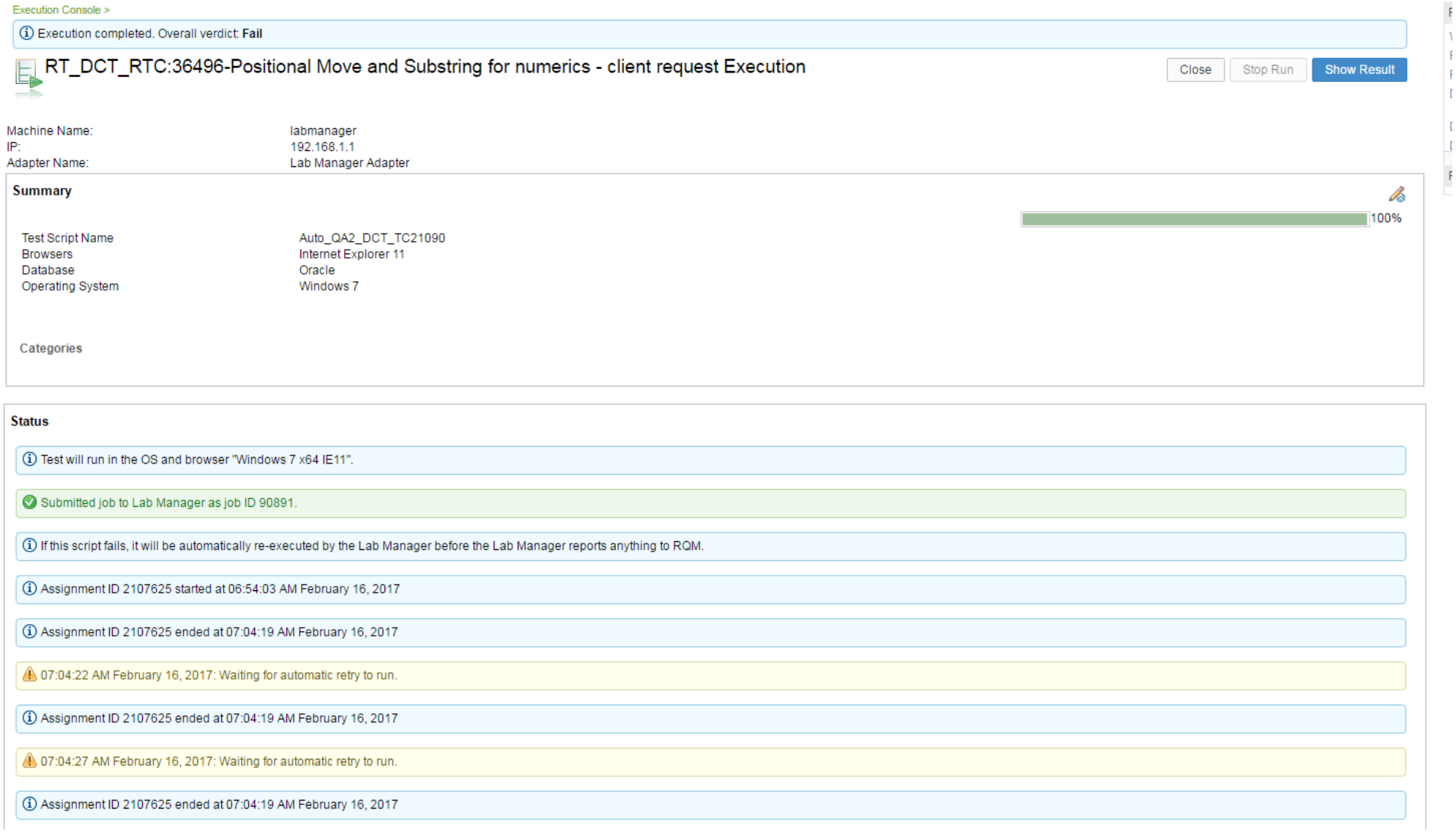
RQM Bridge


Lab Manager
Lab Manager
Lab Manager
RQM Bridge
RQM Server
RQM Project
RQM Project
RQM Server
RQM Project

RQM Bridge

- Bridged the two service APIs
    - Takes requests from RQM.
    - Translates them to Lab Manager requests
    - Monitors progress in Lab Manager, updating RQM
    - Gets final log from Lab Manager Logging DB.
    - Sends result and log back to RQM as RQM-friendly HTML
- Requires us to use our own logging overlay framework in scripts.
- Could not make it work anywhere near as well with RFT's default logging API / pipeline.

RQM Bridge


Execution Console >
Execution completed. Overall verdict: Fail
RT_DCT_RTC:36496-Positional Move and Substring for numerics - client request Execution
Close
Stop Run
Show Result
Machine Name: labmanager
IP: 192.168.1.1
Adapter Name: Lab Manager Adapter
Summary
100%
Test Script Name Auto_QA2_DCT_TC21090
Browsers Internet Explorer 11
Database Oracle
Operating System Windows 7
Categories
Status
Test will run in the OS and browser "Windows 7 x64 IE11".
Submitted job to Lab Manager as job ID 90891.
If this script fails, it will be automatically re-executed by the Lab Manager before the Lab Manager reports anything to RQM.
Assignment ID 2107625 started at 06:54:03 AM February 16, 2017
Assignment ID 2107625 ended at 07:04:19 AM February 16, 2017
07:04:22 AM February 16, 2017: Waiting for automatic retry to run.
Assignment ID 2107625 ended at 07:04:19 AM February 16, 2017
07:04:27 AM February 16, 2017: Waiting for automatic retry to run.
Assignment ID 2107625 ended at 07:04:19 AM February 16, 2017

RQM Bridge

Sections
- Weight Distribution
- Test Environment
- Defects
- Notes
- Categories
- Execution Variables
- Custom Properties
- Previous Results
- Result Details
- Formal Review
- State of Test Artifacts
- Associated E-Signatures
- Show All Sections

History

Build: 311: DCT 1.12 & FC 1.12

Weight: 100

Result Details

07:10:27 AM February 16, 2017 Started script DCT_TC21090

Description: Setup PositionalMove and Compute functions, run Data Conversion, and verify query returns correct data

Arguments

Index	Name	Required	Validation Passed	Value Present	Default Value Present	Default Value Used	Value	Comment
0	$site	No	Yes	Yes	Yes	No	[redacted]	Optional argument $site (Environment to run on) was supplied in at launch ("QA2") and will be used for this script.
1	$userName	No	Yes	Yes	Yes	No	[redacted]	Optional argument $userName (The user name of user) was supplied in at launch ("lbwuser1") and will be used for this script.
2	$company	No	Yes	Yes	Yes	No	[redacted]	Optional argument $company (The company name) was supplied in at launch ("loanboard") and will be used for this script.

07:10:44 AM February 16, 2017 Opened Internet Explorer 11.0.9600.18314

- *URL* = [redacted]
- *Session Process ID* = 2692
- *Tab Process ID* = 700
- *Window Handle* = 524924

07:11:04 AM February 16, 2017 Desktop Login Attempt

- *User* = [redacted]
- *Password* = [redacted]
- *Company* = [redacted]

RQM Bridge - Surprises

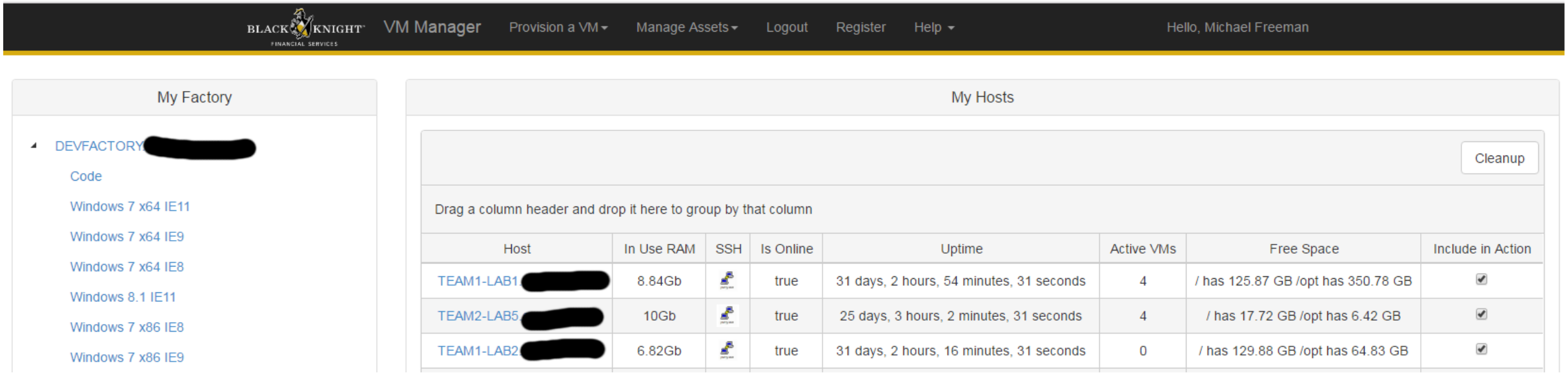
- Custom Test Automation Script Types don't allow you to have variable arguments from instance to instance.
    - Our scripts don't always take the exact same number of parameters.
    - We have pre-defined "run configurations" in Lab Manager.
        - Each one of our custom test automation script instances just points to one of those.
        - The "run configuration" is a combination of the script and a specific set of args for it.
- The bridge had to be ready for connections to RQM being dropped a lot.
- Log returned to RQM must be simple HTML but you can include screenshots.
- RQM does *not* de-duplicate file attachments (e.g. screenshots)
    - An infinite loop accidentally filled up one of our RQM servers until I made the bridge do de-duplication on its end.

Virtual Machine Manager

- Second major tool we created
- Started life as part of Lab Manager
- Pulled it out into a separate tool so we could offer a copy to another group
    - They use it to manage their dev environment VMs.
- Currently, it shares/reuses the Linux agent from Lab Manager.
- Provides a web front end for lower-level tasks that used to be restricted to trained team members running Perl scripts

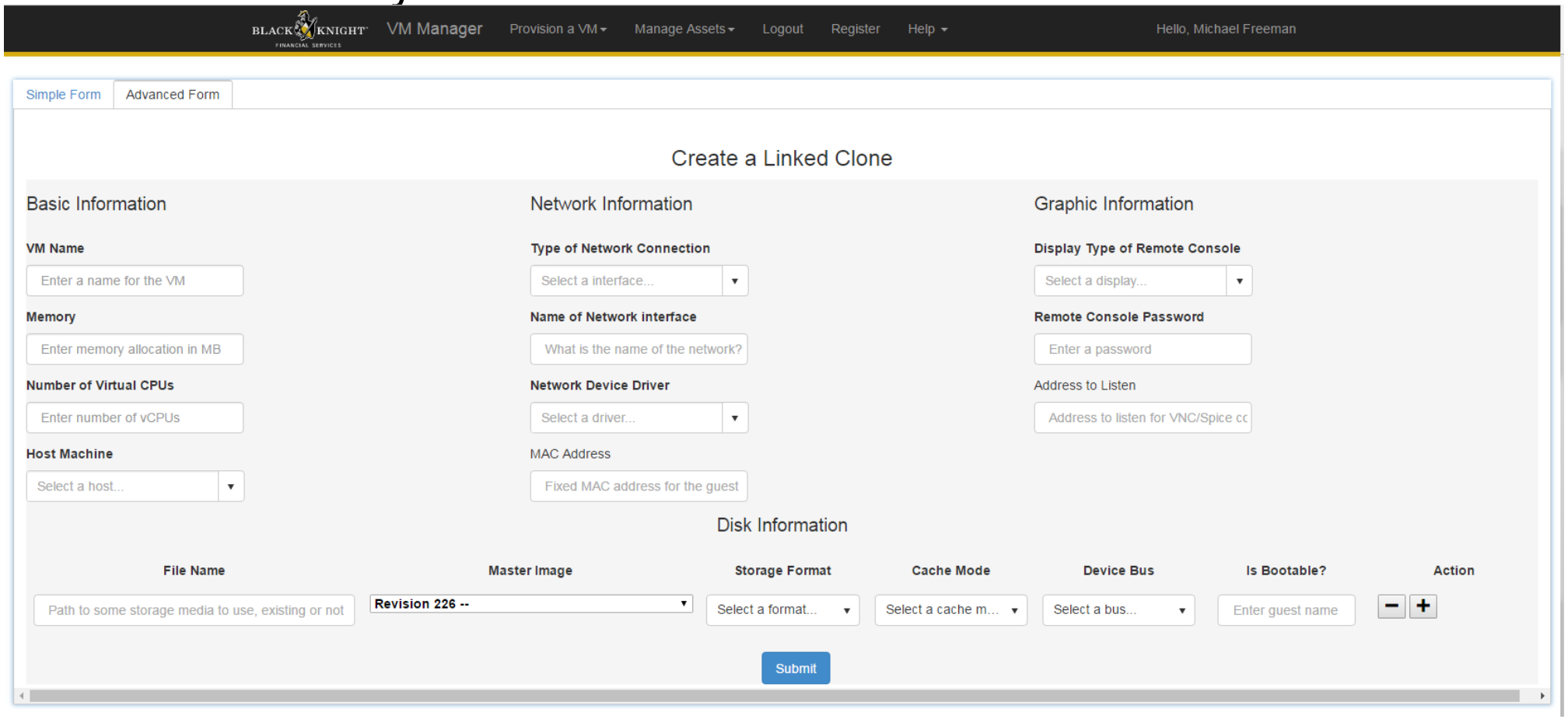
Virtual Machine Manager

- Offers "infrastructure as a service" type capabilities
- Start/stop VMs individually or in bulk
- Monitor lab health (physical machine status, free space, etc)



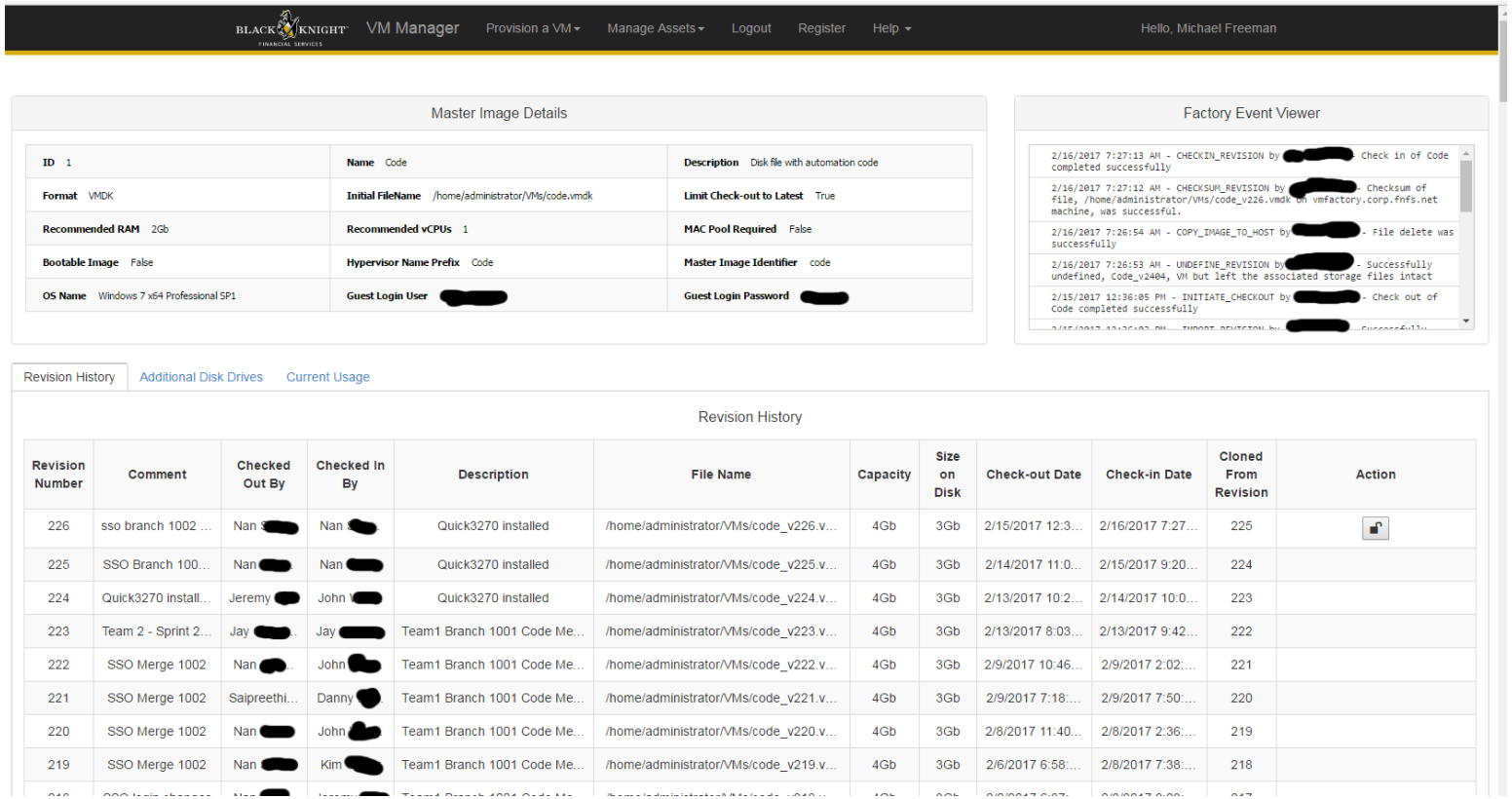
Virtual Machine Manager

- Create/delete VMs on demand
  - Individually or in bulk



Virtual Machine Manager

- Manage lab images (virtual disks)
  - Track/store multiple revisions of the same image, for easy deployment and rollback



Rational Functional Tester

- I personally have been using it since 2007.
- The ability to treat RFT's testMain() as if it were basically a regular Java main() function made a lot of what we do possible.
    - Other automation tools aren't so easy to drive programmatically.
- It works well enough for our lab now.
    - 90% of our applications under test are web apps.
    - Other technologies:
        - .NET full applications: 2
        - 3270 terminal apps: 1
        - Java applets: 1
        - Silverlight applications: 1
        - Delphi 2010 applications: 1

Rational Functional Tester

- Had to build a framework that worked inside it but could bypass or extend it for some cases
    - Logging (mentioned earlier)
    - Apps that needed to use MS UIAutomation underneath for testing
        - Silverlight
        - Delphi
        - Windows Dialogs (Open, Save As, Print, etc)
        - Shared not-under-test apps (e.g. Notepad++, Notepad, etc)
    - Database connectivity
    - Webservice connectivity
    - File IO (Excel, PDF, HTML parsing, CSV, ZIP, etc)

Linux

- Lab machines use Ubuntu Linux 14.04 LTS
- We install QEMU/KVM to provide the actual hypervisor for Windows VMs.
- We use libvirtd as the thin layer between our own agent and the virtual machines on each host.
- If you use boxes with solid state drives, the base Ubuntu OS layer can work basically out of the box with no real changes needed.

Linux

- We can comfortably get about 3 VMs a core, before RFT slows down
- We never let guest memory exceed 75% of the host's total
    - Hypervisors need 10-20% additional RAM per VM for their purposes
        - (e.g. a VM with 2 GB RAM will take 2.2-2.4 GB RAM on the host)
- On a quad core box with 32 GB RAM, this gets 12 VMs with reasonable performance.
- When rolling out new virtual disks to the lab, Windows will sometimes throw fits about product activation. RFT still works even if Windows is unactivated.

Lessons Learned

- Why did we code so much ourselves?
  - Cost of commercial tools
  - Hassle of open source tools
  - Learning experience for the team
    - Junior developers who want to go on to "real" development roles
    - Provides something to break up the routine of scripting
  - On a team of 10+ it took well under 1 person's time-equivalent to develop this over the years.
- Build everything in small modules.
  - You never know what you might realize you can create if you've created the right microservices.
- Automation doesn't need all the shinier aspects of cloudy computing
  - Create, delete, start, stop, update disks…that's all you really need to do to your VMs.

Lessons Learned

- A lab shared across products has proven to be a better value than creating separate labs for each product, even with the cost to create our in-house tools.
    - Execution is faster. Lab machines are never left unused if there's work to be run. Not every product has to shell out for a lab of its own.
    - Automation practices are kept to a global standard so they work on a standard VM.
    - When you have a shared lab, make sure to share the automation team.
        - We spend more time talking amongst ourselves about automation problems (e.g. unusual UI controls) than we do talking with product-specific people about AUT matters.
        - If we were split by products, automation developers would end up reinventing the wheel a decent amount of the time.

Lessons Learned

- Every tool (e.g. RQM) has its surprises that you can't see until you dive in, so when evaluating one, always quadruple any estimates you initially come up with.
- The forums on jazz.net were helpful when writing the RQM bridge.

Conclusion

- In short, sophisticated test automation capabilities are much easier and cheaper to obtain than you might think!
- Hopefully, my notes will help you on your journey towards test automation excellence.

Questions?

InterConnect
2017

Thank you for attending my talk today.

The slides are on GitHub
http://github.com/mlfreeman2

InterConnect 2017