Modernising the 10 year old Day Trader Legacy Application to JBOSS EAP

Includes a Technical Diary of a 2 day MVP migration.

Nov 2018

Benjamin Farr

Senior Solutions Architect

redhat

Agenda

- Why Modernise Applications
- Patterns in Application Modernisation
- Overview of the 10 year old Legacy Day Trader Application
- Building and running the Day Trader Application as is.
- Diary of an Lift and Shift (MVP) Migration
- Best Practises and Recommendations for Migrations

Why Modernise Applications ?

Reduce Risk

- Security concerns
- Supportability (Skills availability)
- Legacy system complex and risky to change
- Reduce Technical Debt

Reduce Cost

- Increase Speed
- Legacy system is undocumented and has lack of tests
- Standardise technology
- Increase Quality
- Enable DevOps
- Increase Agility

Increase Revenue

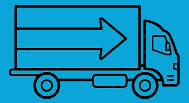
- Increase Reliability
- External interoperability
- Increase Throughput
- Use new features

PATTERNS IN APPLICATION MODERNISATION

Spectrum of Modernisation Options

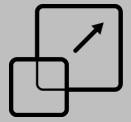
LIFT & SHIFT

- **Containerise** existing workloads
- Deploy them on a **PaaS**
- Keep external integrations and data on legacy
- Legacy applications have to be well written and suited



CONNECT & EXTEND

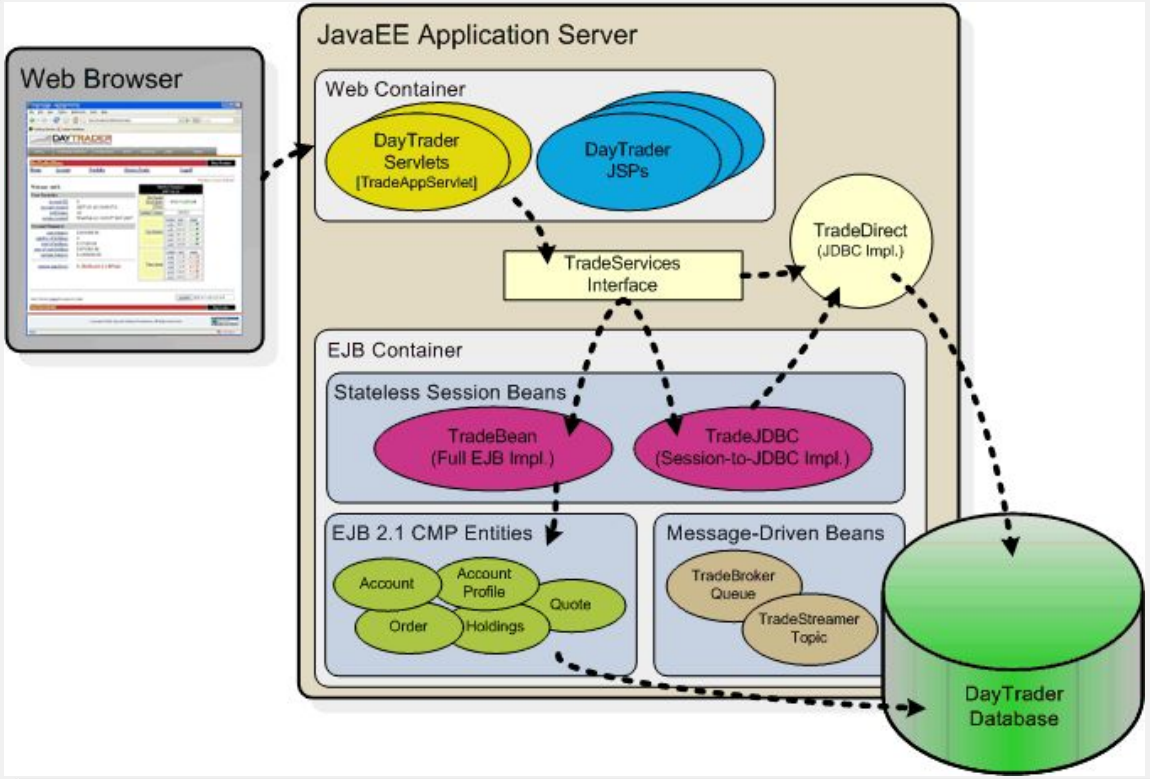
- **Legacy remains intact**
- New layer - new capabilities
- Deploy on **PaaS**
- **New integration points** between legacy and new layers **(Need for Agile Integration)**

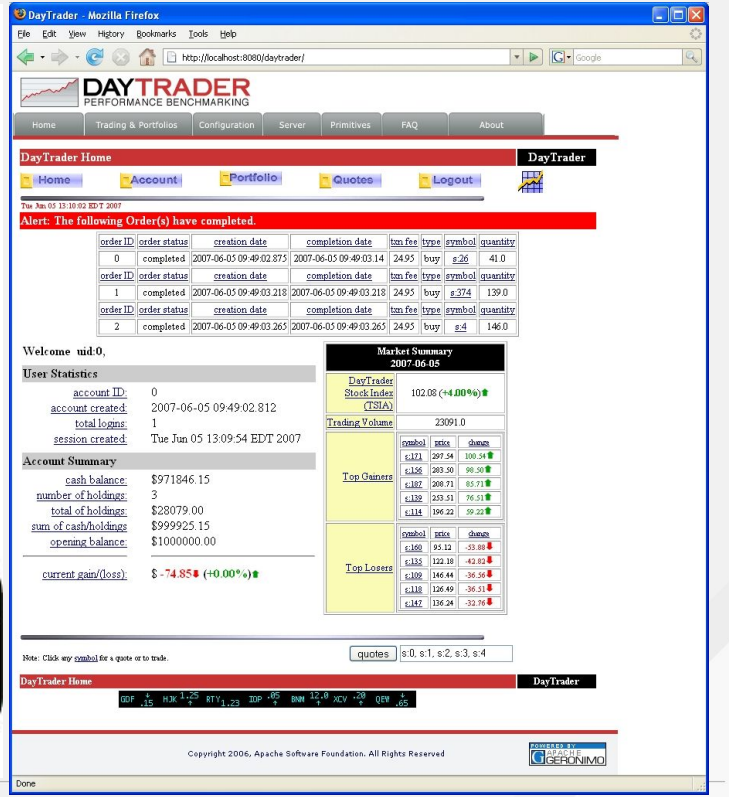


RIP & RE-WRITE

- **Legacy is totally replaced**
- New interfaces and data
- Use **PaaS** to run
- Some data and features can be re-wrapped, but mostly are retired.

Overview of the Day Trader Application





redhat.

Build and run the Day Trader Application as is

Building and Running the Day Trader Application on Legacy Geronimo Server

1. Used the Geronimo v2.2 version which was based on Java EE6.

   http://geronimo.apache.org/GMOxDOC22/daytrader-a-more-complex-application.html

2. Checkout Source Code *

   svn co http://svn.apache.org/repos/asf/geronimo/daytrader/tags/daytrader-2.2.1/

3. Compile with Java 6

   docker run -it --rm -v "$(pwd)":/local/git jamesdbloom/docker-java6-maven mvn clean install

4. Run Geronimo

   docker run -p 8080:8080 jaydm/geronimo:2.2.1

5. Install application manually via the Geronmo Web Server

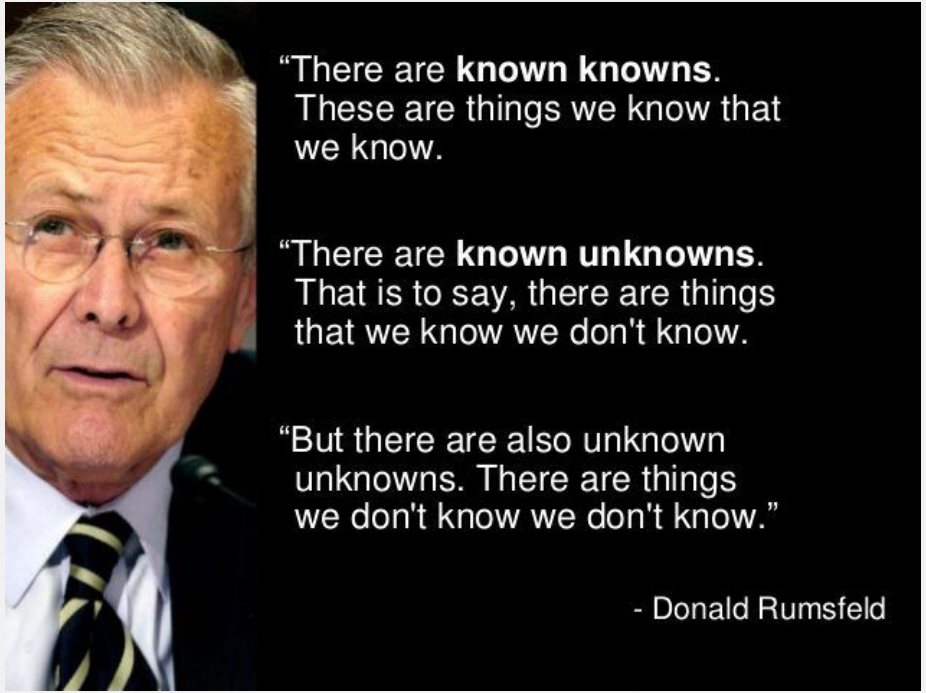
*(and remove unneccessary plugins in pom.xml to allow building)

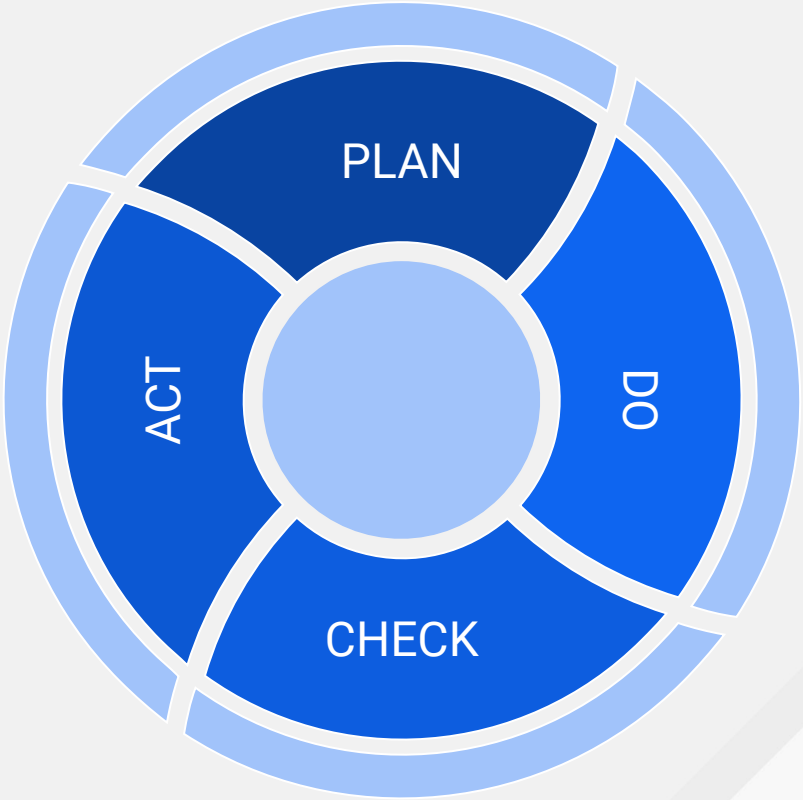
Building and Running on Geronimo

Diary of an Lift and Shift (MVP) Migration

redhat.

Micro Methodology Used





Diary of an Lift and Shift (MVP) Migration

1. Evaluation
2. Build Changes for mvn clean
3. Build Changes for mvn install

1hr
Evaluation and understanding of current project structure, what is needed and what isn't for the MVP migration. Understand XML files (web.xml, persistence.xml etc) which may need to change due to upgrading.

1hr
Incrementally change pom.xml running "mvn clean" after each change which will highlight issues along the way. We do not need to compile yet as the clean will first draw out issues to resolve

2hrs
Incrementally change pom.xml running "mvn install" after each change which will highlight issues along the way. We do not need to compile yet as the clean will first draw out issues to resolve

Diary of an Lift and Shift (MVP) Migration

4. Webservices Rewrite → 5. WebService dependencies → 6. Build Complete

1hr
Webservices required redevelopment to move from legacy to CXF services

2hr
After rewrite of dependencies change the webservice clients

1hr
Further changes needed until mvn install ran successfully

Diary of an Lift and Shift (MVP) Migration

7. Examine and test EAR

2hrs
Now examine the libs within the ear to understand where some libs are implicitly provided by JBOSS 7 or through a module
Another way to find packaged errors is simply to try and deploy it in EAP 7

8. JNDI name changes

1hr
JBOSS requires changes in JNDI names used in datasources, queue names etc which are referenced in the code.
Its better to break and fix it, ie. prefer to remove and reinstate

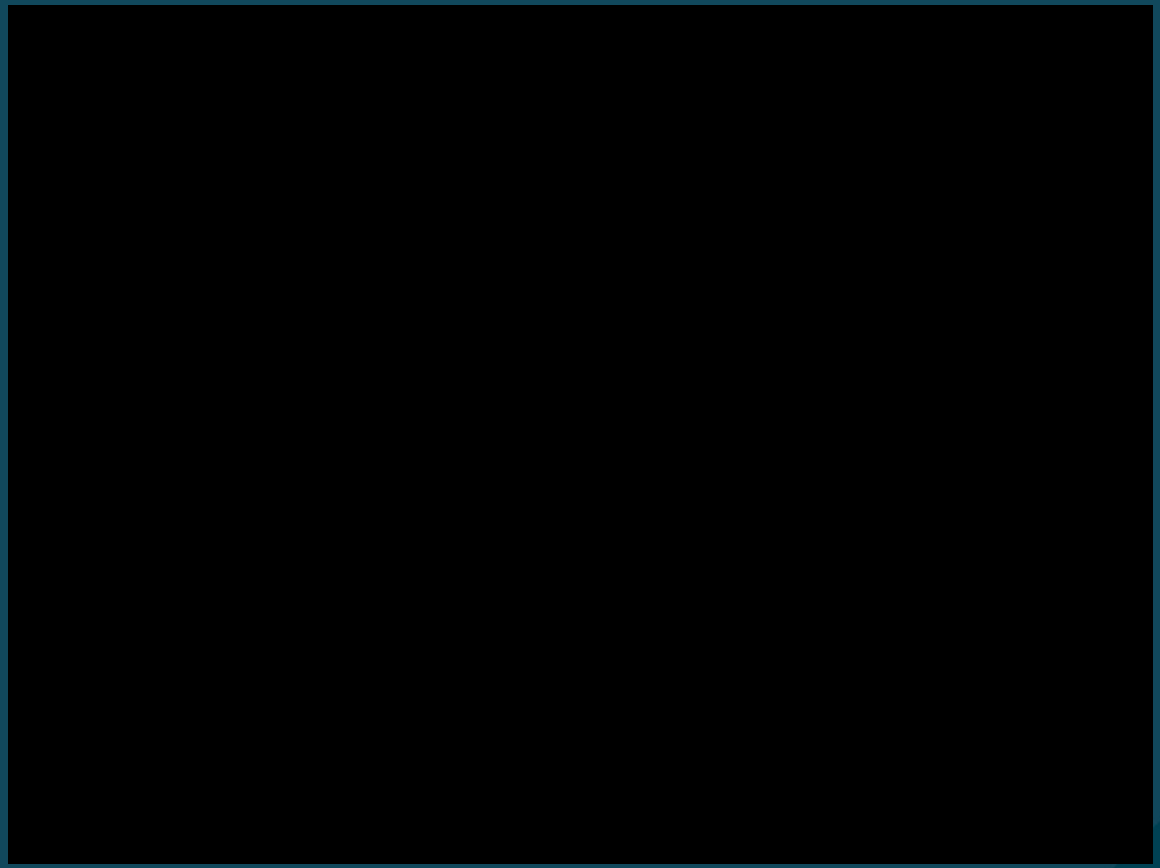
9. Resolve Dependency Problem

3hrs
TradeAction is a problem as its not in the right area.
When introducing EJB3 it caused a maven cyclic dependency, so this needs to be refactored

Walkthrough of the Technical Challenges

MVP Migration Technical Details

Building and Running for JBOSS EAP 7



Best Practises and Recommendations for Migrations

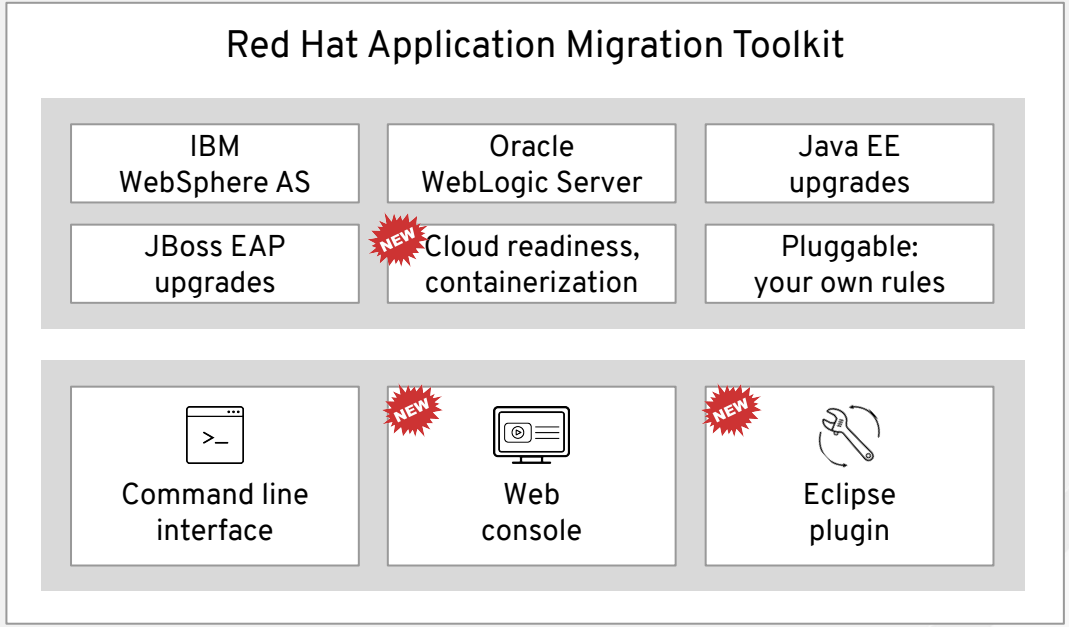
Based on the MVP migration experience - Day Trader is an example that would recommend to Connect and Extend

redhat.

RED HAT® APPLICATION MIGRATION TOOLKIT

Catalyze large scale application modernizations and migrations

- Automate analysis
- Support effort estimation
- Accelerate code migration
- Free & Open Source



RED HAT® APPLICATION MIGRATION TOOLKIT

RED HAT APPLICATION MIGRATION TOOLKIT

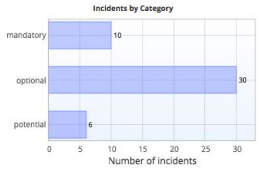
All Applications | Dashboard | Issues | Application Details | Unparsable | Dependencies | Remote Services | EJBs | JPA | About | Send Feedback

Dashboard

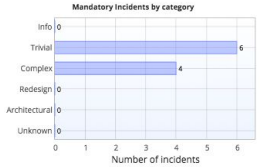
monolith

Dashboard report aggregating findings from the analysis.

Incidents by Category	Incidents	Total Story Points
mandatory	10	18
optional	30	0
potential	6	0
cloud-mandatory	0	0
cloud-optional	0	0

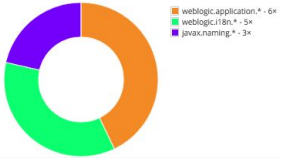


Mandatory Incidents by Type	Incidents	Total Story Points
Info	0	0
Trivial	6	6
Complex	4	12
Redesign	0	0
Architectural	0	0
Unknown	0	0



Java Incidents by Package	Incidents
weblogic.application.*	6
weblogic.i18n.*	5
javax.naming.*	3

Note: this does not include XML files and "possible" issues.



RED HAT® APPLICATION MIGRATION TOOLKIT

RED HAT APPLICATION MIGRATION TOOLKIT

All Applications | Dashboard | Issues | Application Details | Unparsable | Dependencies | Remote Services | EJBs | JPA | About | Send Feedback

Issues

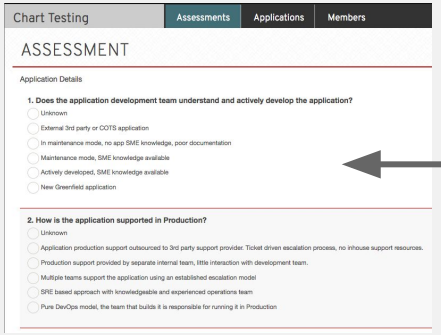
monolith

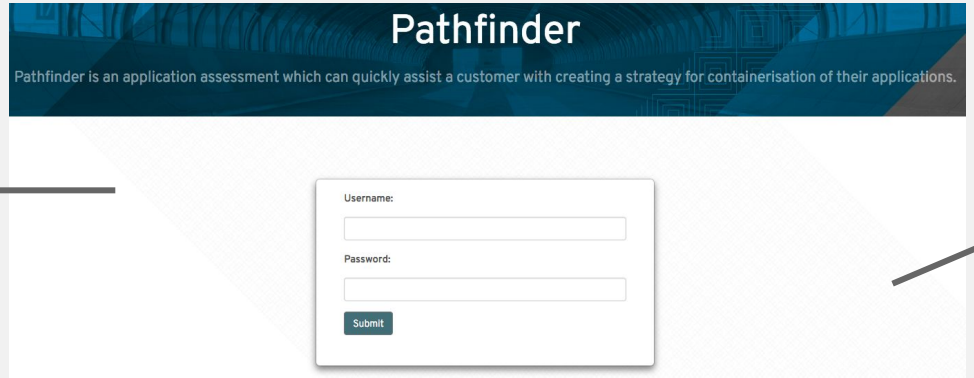
This report provides a concise summary of all issues sorted by category.

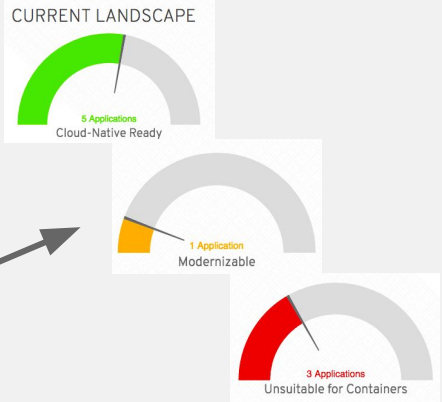
Migration Mandatory

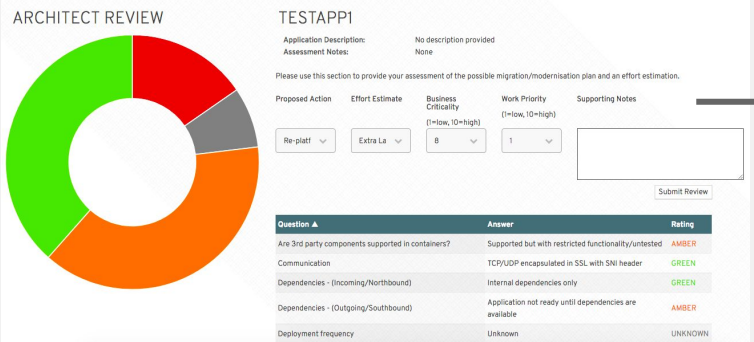
Issue by Category	Incidents Found	Story Points per Incident	Level of Effort	Total Story Points
WebLogic proprietary logger (NonCatalogLogger)	3	1	Trivial change or 1-1 library swap	3
Call of JNDI lookup	2	1	Trivial change or 1-1 library swap	2
Proprietary InitialContext initialization	1	1	Trivial change or 1-1 library swap	1
WebLogic InitialContextFactory	1	3	Complex change with documented solution	3
WebLogic T3 JNDI binding	1	3	Complex change with documented solution	3
WebLogic ApplicationLifecycleEvent	1	3	Complex change with documented solution	3
WebLogic ApplicationLifecycleListener	1	3	Complex change with documented solution	3
	10			18

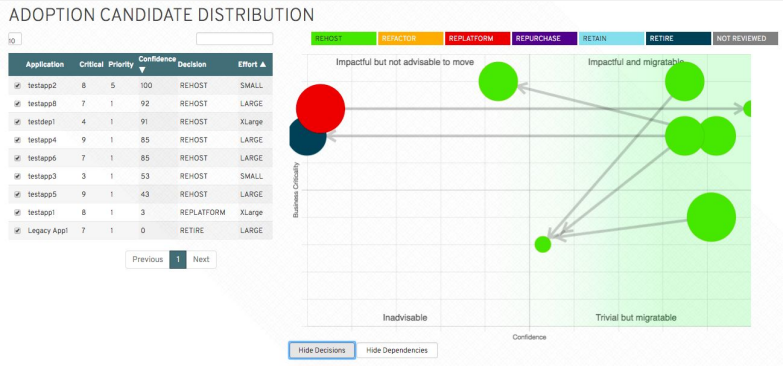
Containerising Applications - Pathfinder Assessment Tool











redhat.

Pathfinder Assessment Questions

A number of questions per application covering the following aspects:

- Architectural Suitability
- Dependencies - Hardware
- Dependencies - Operating system
- Dependencies - 3rd party vendor
- Dependencies - (Incoming/Northbound)
- Dependencies - (Outgoing/Southbound)
- Application resiliency
- Communication
- Compliance
- State Management
- Runtime profile
- Observability - Application Logs
- Observability - Application Metrics
- Observability - Application Health
- Level of ownership
- Discovery
- Deployment Complexity
- Application Testing
- Application Security
- Application Configuration
- Clustering
- Existing containerisation

Recommendations for Lift and Shift

Thoughts from years of JBOSS Migration work, where have seen 10,000 lines of codes in one class, and overly engineered systems.

- First concentrate on the build and packaging the application, then the code, this will ensure the right libraries will be used when testing functionally.
- Do not forget often business logic is in the database.
- Unzip resulting binaries and look at their contents not just source code to help analyse.
- Integration tests and side by side testing is best. Spend time to develop integration/system tests over unit tests if nothing exists.
- Rewrite complex code so it can be understood (Tests if not covered)
- As a default choice upgrade the standards used, use latest versions of libraries.
- Take care of software infrastructure and resources first, functional last
- Use all tools available to provide information (code scanners, RHMAT)
- Design thinking is a good approach
- Side by side integration testing may be needed
- Take care when needing to do migrations of JPA, it may not produce same queries
- Rethink each area - Are there any new libs or open source that does this job ?
- Expect the unexpected!- for example Look out for concurrent conditions in Java 8 which will not work as in Java 6, ie looping through a Map and altering keys/values which can throw runtime errors in Java 8.

Upgrading with lack of tests

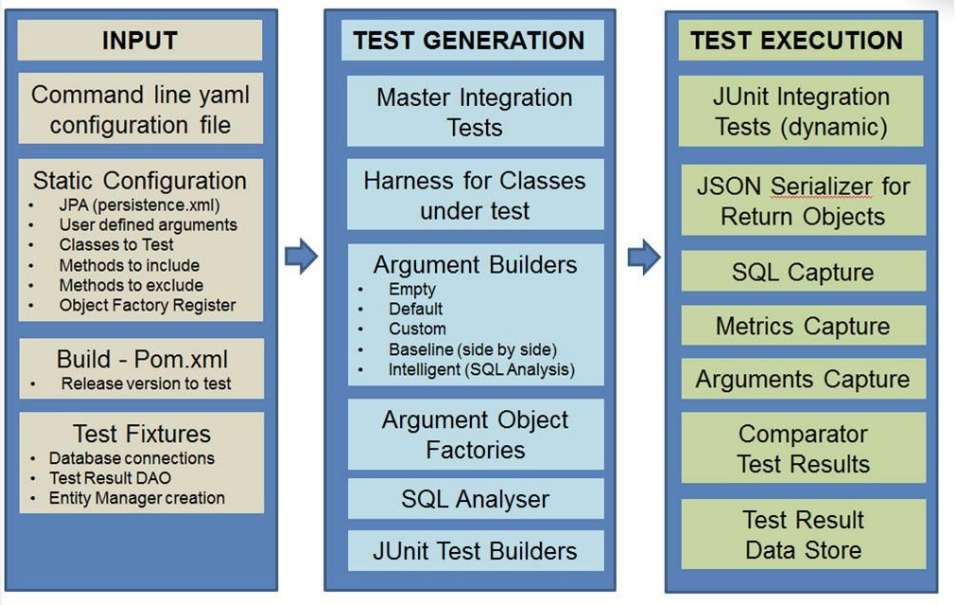
Idea for Automating Test Generation and Execution

The diagram to the right represents a summary of components which were developed to perform detailed side by side analysis when migrating a large system with lack of tests from OpenJPA and Hibernate. But it is applicable framework for any migration.

It was able to automatically generate and run 50,000 test cases invoking methods with various arguments, examine the JPA generated SQL and compare results between the old and new systems.

It highlighted where

- Database queries took longer
- Database queries were different
- Number of Database queries were different
- Java methods return different values for same arguments.



Get in contact for more details.

Summary

- Demonstrated how to run and build legacy applications that would enable side by side testing
- Have seen tight how tightly coupling brings complexity into migrations - Re-examine ROI of lift and shift vs rewrite
- Refactor can be a good option where needing to maintain legacy and using a strangler pattern to incrementally introduce dependencies on microservices hosted on a new platform.
- Diary of an Lift and Shift (MVP) Migration helps to highlight the little issues which can impact on time and cost.
- Follow recommendations and best practises to minimise risk.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos

redhat.