Multi-Platform Application Development

*The Nimble Programmer*

SHARE Session 8657
Abstract

In today's IT environments and into the foreseeable future, enterprise applications are multi-platform by design and implementation. The days of an application being composed on instructions that are run on only one platform, implemented in only one language, are gone forever.

Application programmers now and into the future must be able to move from platform to platform and language to language with ease and effectiveness. This is required in order to design, develop, debug, and maintain the rich applications which are available today and being enhanced in the future.

IBM's collection of application development tools enable application development teams to collaborate on designs, implementations, testing, and maintenance of these complex, mult-platform applications. Come and learn how to use these product features to support multi-platform application design, development, test, and support.

Application programmers can use common tools to work across multiple platforms, multiple runtime environments, and multiple programming languages. By using these common tools, teams are more productive, more efficient, and more accurate even as the environments within which they develop become more complex.
Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
The Changing Application Landscape

- Applications today are a mix of technologies
  - Back-end servers
  - Database technologies
  - Mid-tier application servers
  - Multiple user interface types
  - Mash-ups of information from multiple sources
- Implemented in a variety of different programming languages
  - COBOL
  - PL/I
  - C/C++
  - Java
  - HTML
  - javascript
  - Perl
  - PHP … and the list goes on and on
Applications seem to be user-interface heavy … but not really

- There is a large emphasis on user interfaces
  - Multitude of device types
    - Old-school laptops/desktops
    - Mobile devices
    - Tablet computers
    - Fit-for-purpose machines (printers with touch-screens, desktop information stations, kiosks, ATMs, and so on)
  - Multiple ways of coalescing information
    - Collaboration sites (LotusLive, Facebook, MySpace, iGoogle)
    - News feeds
    - E-Mail
    - Instant Message
- And yet there is an increased attention to programmatic access to services
  - SOA
  - RESTful services
Agenda

● The Changing Application Landscape
● Multi-platform applications
● Past – Dedicated development teams
● Future – Nimble teams
● Tools Support the team
● Summary
Multi-platform Applications

- Counter to what we were seeing in the early 2000s
  - Consolidation of popular environments
  - Rigid structures for inter-system communication
- We are seeing an explosion of systems, devices, interfaces, and languages
  - The programming community is becoming more diverse
  - More heterogeneous
  - More multi-platform
- Existing applications are not being de-commissioned
  - The data they manage is what is in demand
  - These applications are useful
  - Are depended upon by all those that have built on top and around them
- Development teams face a challenge
  - How to manage, maintain, and even enhance existing applications
  - And also create new and exciting, cutting edge applications
Multi-platform applications often look like this
Multi-platform Application Development

Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
Dedicated Development Teams

- Teams have been organized by platform
  - “Mainframe” developers
  - “distributed” developers
  - “web” developers
  - “mobile app” developers
  - … and the “architecture group”
“Mainframe” developers understand the deep, dark secrets of the largest systems in the data-center. Often working with tools and programs that were written long ago by people no longer with the organization. Their tools may seem primitive, but the teams always seem to have an answer for any problem.
Development teams were often isolated

Java developers live for the application server environments within which their applications run. Their expertise ranges from deep understanding and usage of object-oriented constructs, to experience in building dynamically constructed user interfaces from within the application server. And Java applications are running in z/OS as well, as batch or online transaction processing programs.
Development teams were often isolated

Web designers provide the common style and elegance to an organization's application user interfaces, like interior designers ensure a living space is pleasurable to be in. Over time, web designers have become more and more like programmers because the environments required them to learn HTML, CSS, XSLT, and other constructs.
Development teams were often isolated

Mobile application developers are the new kids on the block. This has emerged as a highly active community of developers writing a huge number of applications for a large number of different devices. Many of these applications require programmatic interaction with a multitude of data sources.
Development teams were often isolated

Architecture groups, when they exist in an organization, define the high level structure of application solutions, defining external interfaces between application sub-systems, determining the best place to host elements of the application, and handing down designs to be implemented by multiple teams.
Development teams were often isolated. Unfortunately, there is often a gap between the groups which hinders communication and collective understanding of what each of the teams faces with their particular environment.
Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
Emerging trends – Nimble teams

- Either by necessity or desire, Application developers are
  - increasingly multi-lingual
  - Increasingly multi-platform knowledgeable
  - Increasingly multi-environment enabled

- At the same time, organizations are challenged
  - Address new business opportunities in different geographies, with different clients
  - Create cutting-edge applications using the latest features of the latest devices
  - Maintain and extend existing applications upon which new applications depend

- Programming teams will need to shift quickly
  - From project to project
  - From platform to platform
  - From runtime environment to runtime environment … including those on z/OS!

- Application programming teams must become Nimble!
Development teams will need to shift from task to task

Today's task:
- Java Application
- Web Application
- "Mainframe" application
- Architecture planning
- Mobile Application

Development Team

Today's task:
- COBOL
- Stored Proc
- DB2
- CICS

Today's task:
- COBOL
- Application
- CTG
- MQ

Today's task:
- Java/JEE
- App
- WAS
- C/C++
- Application

Today's task:
- COBOL
- Application
- MQ

Today's task:
- COBOL
- Application

Today's task:
- Javascript
- User Interface

Today's task:
- Windows

Today's task:
- Android / iOS

Today's task:
- Mobile Systems

Today's task:
- Desktop Systems
Challenges to being Nimble

● There are real challenges to being a nimble application developer
  – Must learn new environments quickly and not forget other environments they've used
  – Must be able to pick up an existing application's source code and work effectively
  – Must be able to switch between environments quickly

● There are benefits as well
  – Obtain skills that are relevant to a wide range of applications and environments
  – Be able to contribute to a variety of projects
  – Increase self-marketability
Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
Historically, application development tools have been platform-centric.
Tools – A Silver Bullet (in more ways than one)

- Integrated Development Environments (IDEs have advanced incredibly in the past several years
  - Cross-platform differences are reduced
  - Access to multiple systems simultaneously is expected
  - Multiple language support is now common-place
  - Integration of multiple development tools into a single development environment is now reality
- At the same time, effective use of an IDE requires education and experience
  - On first sight, there is an overload of information
  - On second sight, there are “hidden” features – Where do I click?
  - On third sight, there are sometimes endless UI elements to interpret and understand
- But past the learning curve …
  - Using an IDE allows programmers to concentrate on the application
  - Regardless of programming language
  - Regardless of runtime environment
Tool-assisted Multi-platform Application Development

Integrated Development Environment (IDE)

- COBOL Application
- CICS
- DB2
- MQ
- Java/JEE Application
- WAS
- C/C++ Application
- MQ
- Rational Developer for zEnterprise
- Rational Software Architect
- Rational Business Developer
- Integrated Development Environment (IDE)
- Rational Team Concert
- Rational Asset Analyzer

Multi-platform Application Development:
- COBOL
- Stored Proc
- AIX
- zEnterprise
- Power Systems
- Desktop Systems
- Windows
- Android / iOS
- Mobile Systems

- Javascript
- User Interface

- COBOL Application
- DB2
- CICS
- MQ
- Java/JEE Application
- WAS
- C/C++ Application
- MQ
- Power Systems
- Desktop Systems
- Windows
- Mobile Systems

Simultaneous development, debug, and test of multiple elements of the composite application.
Interactive debug of multiple parts of the application, spanning multiple processing units in the zEnterprise.

Distributed, coordinated, cross-platform build and deploy to test systems.
Some features of a multi-platform IDE

<table>
<thead>
<tr>
<th>Id</th>
<th>Owned By</th>
<th>P</th>
<th>Status</th>
<th>Resolution</th>
<th>Resolution Date</th>
<th>$</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>64795</td>
<td>Unassigned</td>
<td></td>
<td>Normal</td>
<td>Invalid</td>
<td>Nov 25, 2010 10:13 A</td>
<td>$</td>
<td>Issue with certificate</td>
</tr>
<tr>
<td>3966</td>
<td>Rosalind T Radcliffe</td>
<td></td>
<td>New</td>
<td></td>
<td></td>
<td>$</td>
<td>Drive RAA scans from RTC build engine</td>
</tr>
<tr>
<td>3966</td>
<td>Rosalind T Radcliffe</td>
<td></td>
<td>New</td>
<td></td>
<td></td>
<td>$</td>
<td>RTC Build Integration with RRD</td>
</tr>
<tr>
<td>3912</td>
<td>Rosalind T Radcliffe</td>
<td></td>
<td>New</td>
<td></td>
<td></td>
<td>$</td>
<td>Improve RDi Integration</td>
</tr>
<tr>
<td>30472</td>
<td>Unassigned</td>
<td></td>
<td>Triaged</td>
<td></td>
<td></td>
<td>$</td>
<td>No &quot;man&quot; page exists for the scm command-line tool</td>
</tr>
<tr>
<td>30472</td>
<td>Unassigned</td>
<td></td>
<td>Triaged</td>
<td></td>
<td></td>
<td>$</td>
<td>scm command-line tool is not added to user's command search path on install</td>
</tr>
<tr>
<td>35396</td>
<td>Unassigned</td>
<td></td>
<td>Triaged</td>
<td></td>
<td></td>
<td>$</td>
<td>Need prerequisite functionality for workspace integrity</td>
</tr>
</tbody>
</table>

It is very common for command-line tools to have an associated "man" page on Linux and AIX systems. The "scm" command-line tool, however, does not have such a page.

A man page is usually MORE informative than the interactive help (scm -h or scm -h or scm --help) but often not as detailed as a command reference (book or InfoCenter format).

Discussion (5 comments)

1. Tim Hahn, Feb 2, 2010, 11:21 AM
   I believe that the right information exists in the InfoCenter. However, if users are using the command-line tool interactively, then they are likely to be very savvy command-line tool users in general. And these types of users are very used to using "man" pages to get quick access to textual help information.

2. John Cameron, Feb 3, 2010, 3:31 PM
   +1 for having some documentation in man format
   -1 for having different content than what is in InfoCenter. ... it is not worth the effort to maintain 2 sets of doc.

3. Evan Hughes, Apr 4, 2010, 4:09 PM
   To do a decent job at man pages we'd also have to have an installer that puts things in the expected place.

4. Tim Hahn, Feb 5, 2010, 10:50 AM
   @Chapin I agree with the point about getting the installer to put the man page information into the appropriate place. I had opened another defect to get a symbolic link to the "scm" tool placed in /usr/bin on Linux systems which is related to this notion of a better installer. I've added a link to that work item.

On the comment about having information different from the InfoCenter - I think that it should be possible to source the InfoCenter help and the man page help from the same source text. This might take some refactoring somehow at build time to send the text to a man page format (ASCII text, moff format) and InfoCenter format (HTML, I believe).
Some features of a multi-platform IDE
Some features of a multi-platform IDE

```java
package com.ibm.us.hahnt.helloworld;

public class HelloWorld {

    /**
     * This is just a VERY simple program to get myself re-acquainted with the RAD/RSA development environment
     *
     * @param args
     *
     * @return
     */
    public static void main(String[] args) {
        for (int i=0; i<args.length; i++) {
            if (i==0) {
                System.out.printf("number of arguments = %s", args.length);
            } else {
                System.out.printf("arg[%d] = %s", args[i], args[i]);
            }
        }
        System.out.println("Hello World");
    }
}
```

number of arguments = 4
arg[0] = first
arg[1] = second
arg[2] = third
arg[3] = fourth
Hello World
Some features of a multi-platform IDE

```
IDENTIFICATION DIVISION.
PROGRAM-ID HELLO.
* IS RECURSIVE INITIAL PROGRAM.
* AUTHOR: Tim Main.
* INSTALLATION.
* DATE WRITTEN: 20100602
* DATE COMPILED.
* SECURITY.

Environment division.

Data division.

file section.

working-storage section.

local-storage section.

01 counter PIC 999 USAGE IS COMP.
77 name PIC X(15).

linkage section.

Procedure division.

* ** program goes here **

* MOVE 1 to counter

ADD 1 to counter

MOVE "Tim" to name

DISPLAY "Hello", name, ";", counter is ",", counter

* EXIT PROGRAM.

* STOP RUN.
```

Subsystem JES

Resource | Parent filter pool | Parent filter | Number of filter strings | Connection-private
---|---|---|---|---
Retrieved jobs | CN-mvso40.rtp.raleigh.ibm.com:com.ibm.z Not applicable | | 1 | Yes
My jobs | anuta.com.ibm.zos.jes Not applicable | | 1 | No
Some features of a multi-platform IDE
Some features of a multi-platform IDE

Hello Everyone!!

Specify the start time: 14:46

Hello world: The value of the time setting is 14:46:18
Agenda

- The Changing Application Landscape
- Multi-platform applications
- Past – Dedicated development teams
- Future – Nimble teams
- Tools Support the team
- Summary
Summary

● Computing environments, and the applications that run in them continue to rise in complexity
  – Multiple platforms
  – Multiple languages
  – Multiple runtime environments

  – And existing environments are not going away!!

● The days of being a siloed, single-purpose, application developer are numbered
  – Younger professionals are already flexible to multiple environments

  – Flexibility to apply to multiple projects is a valued skill for the organization

● The future is wide open for nimble development teams
  – Maintain and enhance existing applications with speed and precision
  – Design and Implement new applications on cutting edge devices

  – Using software development tools that enable multi-platform application development
Useful Links

● Rational Developer for zEnterprise Information:

● Rational Team Concert Information:

● Rational Asset Analyzer Information

● Rational Application Development “Cafes”:

● Jazz Team Blog:
  – http://jazz.net/blog/

● My information
  – email: hahnt@us.ibm.com