Large-Scale Integration Testing at Microsoft

SASQAG Talk – February 17th, 2011

Jean Hartmann
Test Architect
Developer Division Engineering Systems
Microsoft Corp.

jeanhar@microsoft.com

Agenda

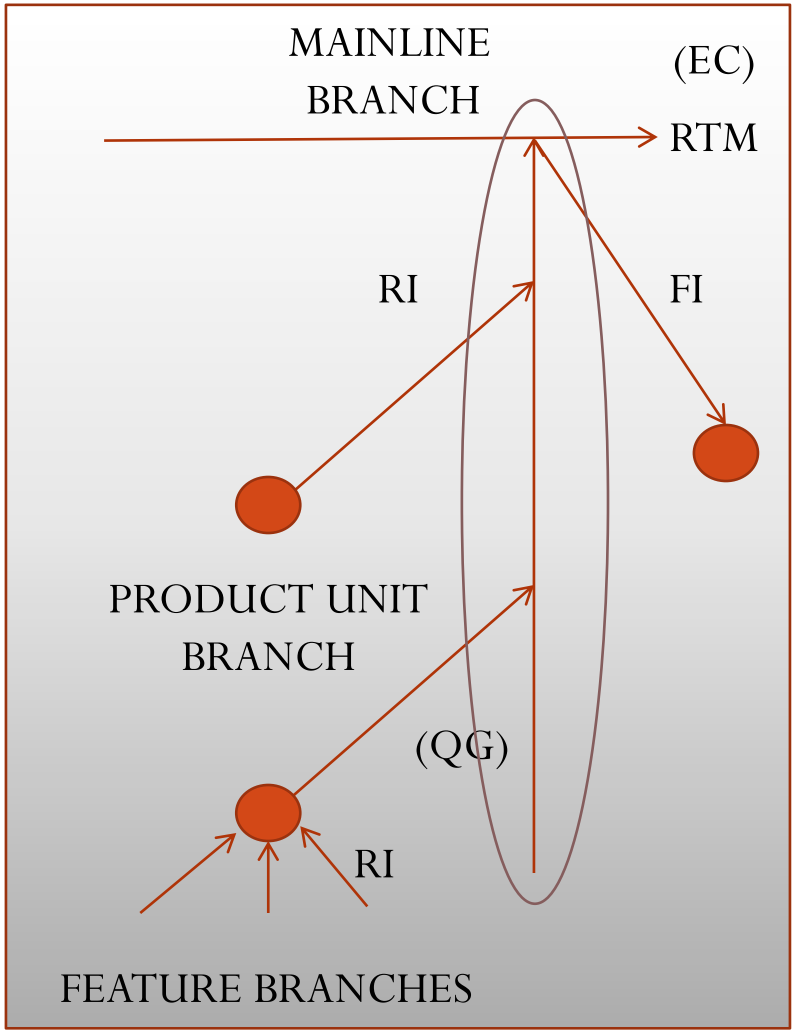
- Background
- Integration Testing @ Microsoft
- Defining Test Suite
- Implementing Test Suite
- Deploying Test Suite
- Enhancing Tools and Infrastructure
- Benefits and Challenges
- Conclusions and Future Work

Background – Developer Division

- Makers of Visual Studio, Expression Suite , Silverlight and the .NET Framework
- 2000+ people work for the division
- Building Visual Studio takes a total of thirteen hours to complete
- Visual Studio contains several millions lines of C++/C# code
- Test collateral amounts to hundreds of thousands of automated tests written in C++/C# and VB

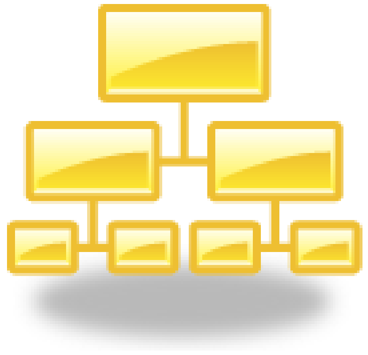
Integration Testing : Terminology

- Build Branches
  - Feature Branches (FB's)
  - Product Unit (PU) Branches
  - Mainline Branch (Visual Studio)
- Integrations
  - Reverse Integrations (RI's)
  - Forward Integrations (FI's)
- Testing
  - Quality Gates (QG)
  - Exit Criteria (EC)



Integration Testing : Code Flow

- Large number of PU branches with complex dependencies
- *Breaking changes* quickly propagate across build tree
- Debugging becomes a nightmare, randomizing teams
- Often cripples PU and mainline builds for days or weeks
- Teams become grid-locked and isolate themselves
- Huge effort to unblock and produce stable mainline build
- Want to get to a "**fearless FI"** scenario!



Integration Testing: Test Perspective

- Central validation suite (DDBasics) existed, but…
- It was not focused, too large and included legacy tests
- Tests written against various automation frameworks
- Tests required difficult-to-reproduce installs/set-ups
- Most tests were UI-based - often unstable
- Nightmare to debug when tests failed
- In short, you had to be a wizard to run & debug these tests!

New Test Suite Selection Criteria

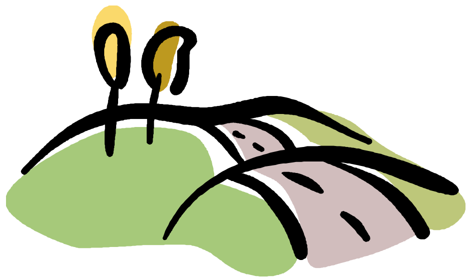
- Had to strike balance between *efficiency* and *effectiveness*
- Apply more stringent test selection criteria from the onset
- Define update process to evolve it
- *Efficiency* criteria
  - Tests execute in a two hour timeframe (excl. setup)
  - Tests are reliable, consistent and easy to debug
- *Effectiveness* criteria
  - Tests exercise key product integration points
  - Tests reflect customer success scenarios (‘happy paths’)
- Result: RI Tests – DDRITs and PURITs

Developer Division RITs (DDRITs)

- Testing major integration points coinciding with top customer success scenarios
- Run in:
  - PU branch prior to an RI
  - Mainline branch after RI
- Test failures block RI (P0 bug)
- Quick turnaround for fixes needed
- Currently, 35 functional and 4 perf tests

Product Unit-specific RITs (PURITs)

- Testing unique integration points between specific PU's
- Tests *donated* by teams affected by breaking changes
- Tests *run* by teams likely to cause breaking changes
- Run prior to RI (together with DDRITs)
- Failures do **not** block RI (P1 bugs)
- Turnaround time for fixes dictated by SLA (service level agreement) between dependent teams
- Currently, ~100 test cases in total



Proposing & Reviewing RITs

- DDRITs
  - *Divisional* review committee established (development and test managers, senior technical staff)
  - Review and approve proposed tests, ensure strict adherence to selection criteria
  - Considered functional and key non-functional (performance) tests (within time constraints)
- PURITs
  - Review committee comprised of staff from the partnering, dependent teams only
  - Test purpose and content were captured in SLAs
- Reviews form integral part of new integration testing process
- Critical in helping teams define new RITs, update existing ones and deprecating old tests

Efficient Failure Resolution

- Goal is to minimize disruption to teams
- Requires good inter- and intra-team communications
- Resolution task needs to have an owner – *PU RI Rep*
- PU RI Rep job description
  - Initial triaging (investigation) and assignment of resulting bugs
  - Collaborating and liaising with:
    - Developers and testers within own team,
    - Dependent partner and central tooling teams to resolve bugs
  - Updating of SLA agreements, if applicable
- Update: centrally managed by build team

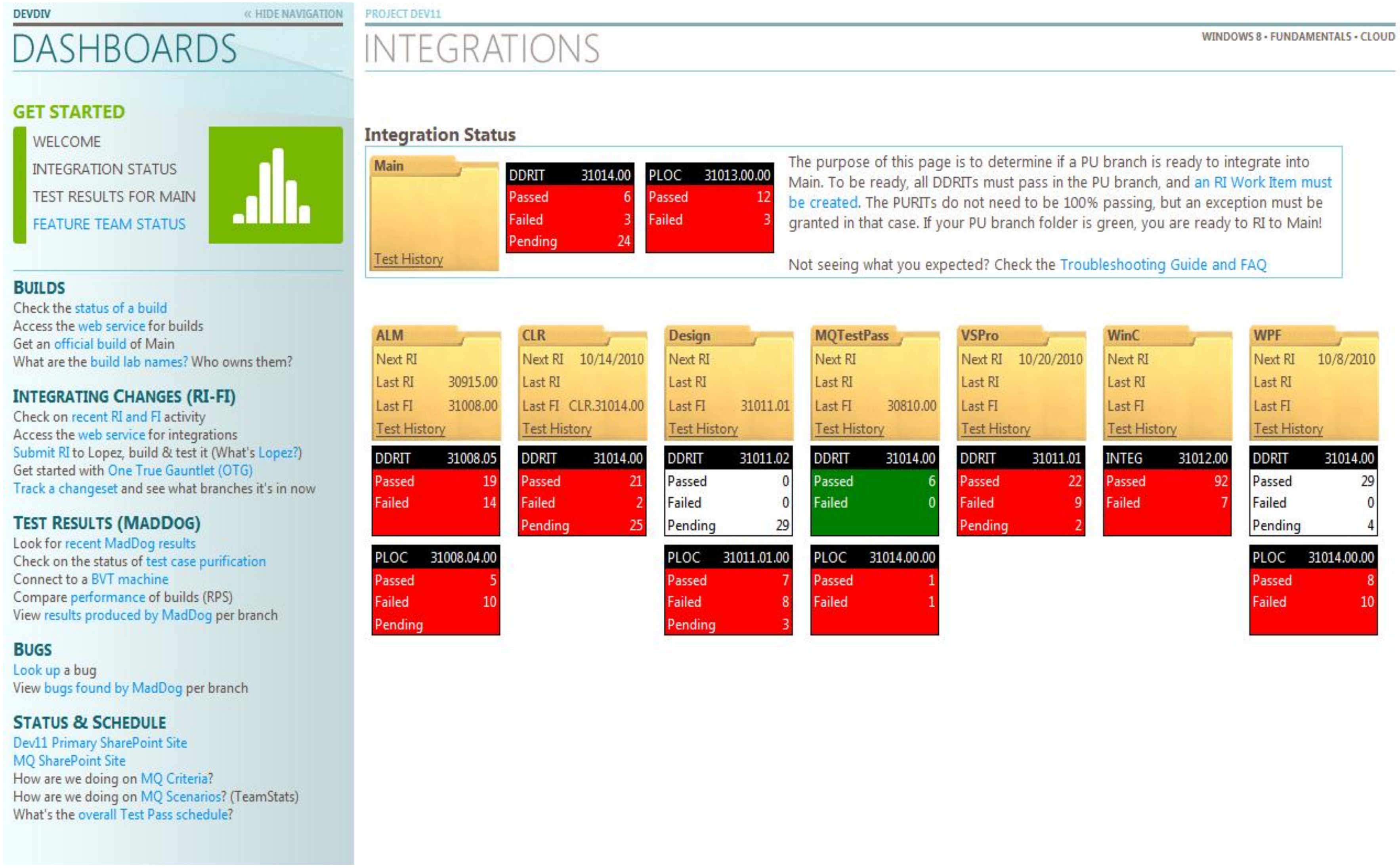
Implementing Integration Tests

- Efficiency and reliability were key drivers
- Implemented by teams proposing RITs
- Tests coded directly against Visual Studio APIs
- Tests ‘purified’ for improved reliability
- Update:
  - Insufficient code review time allocated
  - Test owners had varied experience/knowledge

Executing New Test Suite

- Consolidating our existing, distributed testing lab resources
- Moved to a new private cloud at Redmond Ridge
- For testing purposes, teams can now run the RITs using Virtual Machines (VMs), enabling quick re-imaging of machines using canned virtual machine images (VHDs)
- Test case management system optimizes for quicker set-up and installation to reduce overall test execution time (incl. caching)
- As failures occur, images are reserved for remote debugging
- RITs executed by our central engineering team (DES)

Results Reporting for New Test Suite



DEVDIV « HIDE NAVIGATION

DASHBOARDS

GET STARTED

- WELCOME
- INTEGRATION STATUS
- TEST RESULTS FOR MAIN
- FEATURE TEAM STATUS

BUILDS

Check the status of a build
Access the web service for builds
Get an official build of Main
What are the build lab names? Who owns them?

INTEGRATING CHANGES (RI-FI)

Check on recent RI and FI activity
Access the web service for integrations
Submit RI to Lopez, build & test it (What's Lopez?)
Get started with One True Gauntlet (OTG)
Track a changeset and see what branches it's in now

TEST RESULTS (MADDOG)

Look for recent MadDog results
Check on the status of test case purification
Connect to a BVT machine
Compare performance of builds (RPS)
View results produced by MadDog per branch

BUGS

Look up a bug
View bugs found by MadDog per branch

STATUS & SCHEDULE

Dev11 Primary SharePoint Site
MQ SharePoint Site
How are we doing on MQ Criteria?
How are we doing on MQ Scenarios? (TeamStats)
What's the overall Test Pass schedule?

PROJECT DEV11

INTEGRATIONS

WINDOWS 8 • FUNDAMENTALS • CLOUD

Integration Status

Main
Test History

DDRIT	31014.00
Passed	6
Failed	3
Pending	24

PLOC	31013.00.00
Passed	12
Failed	3

The purpose of this page is to determine if a PU branch is ready to integrate into Main. To be ready, all DDRITs must pass in the PU branch, and an RI Work Item must be created. The PURITs do not need to be 100% passing, but an exception must be granted in that case. If your PU branch folder is green, you are ready to RI to Main!

Not seeing what you expected? Check the Troubleshooting Guide and FAQ

ALM	
Next RI	
Last RI	30915.00
Last FI	31008.00
Test History	
DDRIT	31008.05
Passed	19
Failed	14
PLOC	31008.04.00
Passed	5
Failed	10
Pending	

CLR	
Next RI	10/14/2010
Last RI	
Last FI	CLR.31014.00
Test History	
DDRIT	31014.00
Passed	21
Failed	2
Pending	25

Design	
Next RI	
Last RI	
Last FI	31011.01
Test History	
DDRIT	31011.02
Passed	0
Failed	0
Pending	29
PLOC	31011.01.00
Passed	7
Failed	8
Pending	3

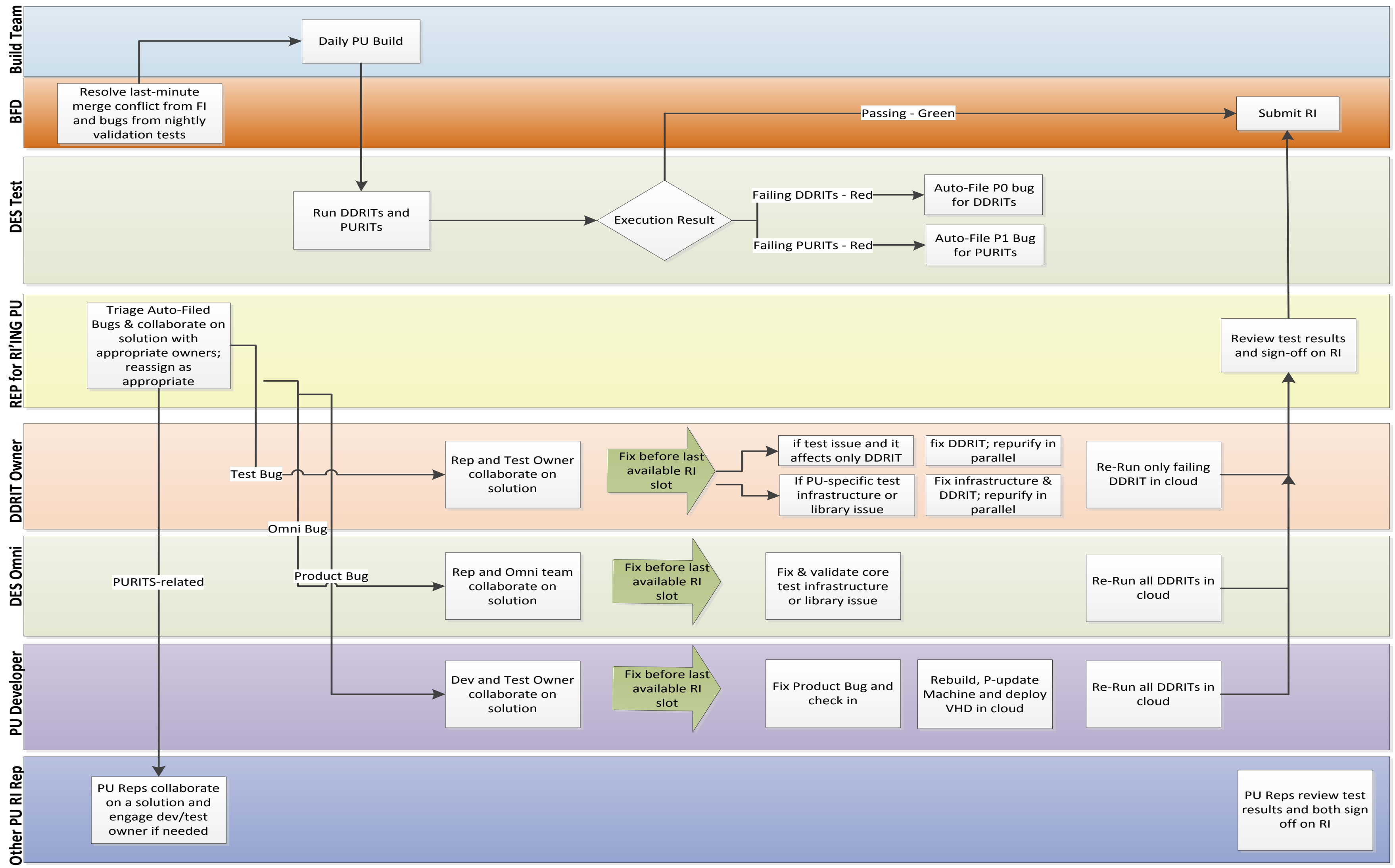
MQTestPass	
Next RI	
Last RI	
Last FI	30810.00
Test History	
DDRIT	31014.00
Passed	6
Failed	0
PLOC	31014.00.00
Passed	1
Failed	1

VSPro	
Next RI	10/20/2010
Last RI	
Last FI	
Test History	
DDRIT	31011.01
Passed	22
Failed	9
Pending	2

WinC	
Next RI	
Last RI	
Last FI	
Test History	
INTEG	31012.00
Passed	92
Failed	7

WPF	
Next RI	10/8/2010
Last RI	
Last FI	
Test History	
DDRIT	31014.00
Passed	29
Failed	0
Pending	4
PLOC	31014.00.00
Passed	8
Failed	10

Deploying the Test Suite: Workflow



Build Team
BFD
DES Test
REP for RI'ING PU
DDRIT Owner
DES Omni
PU Developer
Other PU RI Rep
Daily PU Build
Resolve last-minute merge conflict from FI and bugs from nightly validation tests
Passing - Green
Submit RI
Run DDRITs and PURITs
Execution Result
Failing DDRITs - Red
Auto-File P0 bug for DDRITs
Failing PURITs - Red
Auto-File P1 Bug for PURITs
Triage Auto-Filed Bugs & collaborate on solution with appropriate owners; reassign as appropriate
Review test results and sign-off on RI
Test Bug
Rep and Test Owner collaborate on solution
Fix before last available RI slot
if test issue and it affects only DDRIT
fix DDRIT; repurify in parallel
If PU-specific test infrastructure or library issue
Fix infrastructure & DDRIT; repurify in parallel
Re-Run only failing DDRIT in cloud
Omni Bug
PURITS-related
Product Bug
Rep and Omni team collaborate on solution
Fix before last available RI slot
Fix & validate core test infrastructure or library issue
Re-Run all DDRITs in cloud
Dev and Test Owner collaborate on solution
Fix before last available RI slot
Fix Product Bug and check in
Rebuild, P-update Machine and deploy VHD in cloud
Re-Run all DDRITs in cloud
PU Reps collaborate on a solution and engage dev/test owner if needed
PU Reps review test results and both sign off on RI

Tools and Infrastructure

- New API-based test automation framework
  - Evaluated and consolidated existing frameworks
  - Implemented common logging framework
  - Adapted product interfaces to improve testability
  - Part of Omni (UI+API-based) automation infrastructure
- Test purification and portability (TRA)
  - Providing tool support to ensure test reliability and stability
  - Enhancing test cases (and management system) to capture test case metadata

Benefits and Challenges

- Increased code velocity throughout the build tree (RI/FI)
- Earlier defect detection to prevent major breaking changes
- Faster and more reliable test execution, easier failure analysis
- Highlighted the need for better product testability

- Defining, socializing and executing on this concept
- Introducing the API-based testing concept/product changes
- Establishing new cloud-based infrastructure to execute tests
- Restructuring the build tree to ease test case management, execution and maintenance

Conclusions and Future Work

- Gave insight into existing integration test issues @ Microsoft
- Discussed our integration test strategy for rapid code flow
- Highlighted our process and tool improvements, deliverables and deployment issues

- V2 – towards the automated selection of integration tests