Empirical Results from the Transformation of a large Commercial Technical Computing Environment

Dr. Rahul Razdan
Outline

• Context

• Requirements

• Analysis and Project Launch

• Solution Architecture

• Results

• Future Work
Context

• Industry: Electronics Design Automation ($4B):
  – Design Tools for Semi-Conductor Design ($250B)
  – Industry Norms: complexity, release management, M&A

• Group: Hardware Virtual Modeling
  – Features: Run-time Performance, Infrastructure for Virtual Experiments (Modeling, Testing, Coverage, Project Management)
  – Four Sites, 250 developers, most advanced degrees, 3 recent mergers
Compelling Reasons for Change

• Acquisitions and Growth Impact
• Globally-distributed software development teams
• Expanded Product Line with new capability
• Introduction of newly-supported platforms
• Not organized for growth:
  – Internally – product focused versus infrastructure focused
  – Externally – new verification languages, OS changes
Environment: Technical Computing

- Run-Time Performance
- Customer Responsiveness
- Innovation
- Multiple Platforms/Release Cycles
- Limited Skilled Developers
Outline

• Context

• Requirements:

• **Analysis and Project Launch**

• Solution Architecture

• Results

• Future Work
SW Development Heartbeat

- Core SW Development:
  - Build-Link-Validate Cycle
  - Scaled for developers across geographies
  - Scaled for multiple steams
- Accelerating Time-to-market or Increasing Quality involves fundamental restructure of this core process across the enterprise.
- Measure with metrics
Enterprise-Wide Process Metrics

Optimizing All Aspects of Productivity

• Productivity:
  – time to first test
  – incremental time to create new tests
  – coverage/day
  – gates/functions verified/engr mm
  – time to derivative environments

• Predictability:
  – total coverage
  – coverage convergence rate
  – bug convergence rate
  – project resource & convergence stats
    • to plan next project better

• Quality:
  – # respins
  – # functional bugs ID'd in post silicon
  – # field recalls
  – breakout of hardware vs. software bugs

• Human Resource Utilization:
  – % reuse of verification plans
  – % reuse of verif’n environments
  – % reuse of verification components

• Compute Resource Utilization:
  – % of sims running 24x7
  – cycles used for last 10% coverage

• Best Practices Deployment:
  – Automation deployment level
    • block, chip, system, project levels
  – Verification maturity scale
    • Directed testing
    • Automated testing
    • Coverage driven
    • Scalable coverage driven
  – Reuse
Observations

• Issues:
  – Release Planning Ad-hoc
  – Test infrastructure large, complex, and difficult to handle with IT environment (ex: performance)
  – Coordination between sites very difficult
  – Release management error-prone and high stress
  – PV, CM, and PM not tier-one career paths

• Solutions: No commercial solutions available
Project Launch

• Organizational Decisions:
  – Integration of PV, CM, and PM under GM (not popular)
  – Develop a separately resourced project for infrastructure

• Project Launch:
  – Pull two key architect level individuals for this work (very unpopular)
  – Resource appropriately to do the job (capital, services)…many skeptics in finance organization
  – Explicitly manage the process of change
Outline

• Context

• Requirements:

• Analysis and Project Launch

• Solution Architecture

• Results

• Future Work
Environment Maturity Model

- Motivation (the why) drives downward from the upper layers to trigger change
- Implementation and process schema (the how) provide the foundation for the model
Foundations

• Central storage structure
• Common language (Perl)
• Core modules
  – Command-line processing
  – Messaging and logging
  – Common parsing framework
  – Site customization
  – Object Data Definitions
  – Platform classification
Infrastructure and Policies

- Infrastructure
  - Fault-tolerant central storage
  - Robust local network
  - Controlled image configurations
  - Dedicated servers
  - Monitoring & Management

- Policies
  - Managed growth
  - Defined API for tools
  - Resource management
Configuration Management

- Tools
  - ClearCase
  - MultiSite
- Resources
  - Central VOB/View servers
  - Central registry and licensing servers
- Methodology
  - Branching and Merging
  - Trigger conventions
Project/Variant

- Context for developer activity
  - Policy-based control
  - Standard build/install
- Managed data
  - Dependency kits
  - User environment
  - Build components
  - Project policy
  - Configuration Management
  - Testing environments
Server Farm

- Foundation
  - Hardware
  - Tools
  - DRM (eg LSF)
- Services
  - Meeting the user need
- Bridging the gap
  - Management services
  - BuildJob
  - TestJob
  - AutoControl sequencing
KitExchange

- Inter-project collaboration
  - Software integration
  - Distributed build support
  - Development merge support
- Managed Data
  - KitExchange meta-data
  - Content depots
- Flexible Architecture
  - Communication plugins
  - Fall-back data sources
Development Processes

- Coordination
  - Merge schedules
  - External dependency validation
- Quality
  - Perpetual release readiness
- Release Engineering
  - Decision criteria
  - Unified Release
- Applied Governance
  - Control and measurement
  - Policies to address internal and external compliance
  - Drive consistency and best practices
  - Benefits-driven model
Business Processes

- Early Adopter engagements
- Requirements gathering
- Product release model
- Solutions integration
Results after 6 years

**Before**
- 200,000 avg daily tests
- Unknown # of projects
- 1 site
- 80 R&D/PV engineers
- 3 release streams
- ? cpu [? hosts] server farm

**After**
- 6.6 million avg daily tests [10.2 million peak]
- 349 projects
- 7 sites [plus other satellite locations]
- 270 R&D/PV engineers
- 7 release streams [current and future]
- 2187 cpu [1041 hosts] server farm

Scaling capability while growing customer satisfaction!
And Less Chaos...
Outline

• Context

• Requirements:

• Analysis and Project Launch

• Solution Architecture

• Results

• Future Work
SW Integration Efforts Outweigh Pure HW

Factors into productivity, quality, predictability risks

The Process Today

The Desired Process

Whole Product View

Team Ratios

Solution
Application
Middleware
OS
Firmware
HW

SW Mgr

HW Mgr

10

4

1

Software
Design
Code

Hardware
Design
Build

Chip
Design
Fab
Chip Debug

Chip Respins

Concurrent Flow

System Integration & Debug

HW Integration & Debug

System Integration & SW Debug

Design Phase

Design
Code

Design
Build

Design
Chip Debug

Fab
Cadence & IBM Joint Customer Solution View

- Design & Verification Plan to Closure
- System Wide Management

- System Level Design & Verification

- HW Design, Verification, Implementation
- SW Design, Debug, and Environment

- System Validation, Logic Signoff
- Environment

- Design to Silicon

Roles:
- HW Design Engineers
- System Design Engineers
- Verification Methodology
- System validation Engineers
- System Engineers
- Lifecycle Automation
- Exec & Project Manager
- Embedded SW Developers
- Embedded SW Developers

- Verification Engineers
Lessons Learned

• Treat as a whole system – cannot look at piece parts
• Swallow hard and make the decision to go for it – it must be central to the business
• Need to make the investment with the right focus
• Processes developed internally can open the door for an infrastructure element in the products delivered to your customers – leads to opportunity for IBM and Cadence to partner further to deliver to the industry
• It’s not pie-in-the-sky. It works !!
Questions
NC-Verilog
P0 Activity on Previous Releases
Customer Support Summary

- Overall Volume is decreasing and resolution time is improving.
  - Total number of days that support cases are outstanding in August is about a third of what it was in June (6400 vs 17,000).
  - Calls not resulting in a PCR is down to 7.96 days. Calls resulting in PCRs still high at 57.2 days.
- Satisfaction survey results improved slightly in August.