

Robust Hot Swap Infrastructure Software: the Key Ingredients

Mark Overgaard
President
Pigeon Point Systems
May 16, 2001



Overview

- Pigeon Point Systems Background
- Review of PICMG Hot Swap Specs
- Key Ingredients for Robust Hot Swap Infrastructure Software
- Delivering Those Key Ingredients for Windows and Linux
- Conclusion



Pigeon Point Systems

- Founded in 1997 to focus on high availability system s/w for CompactPCI
- Authored software aspects of CompactPCI Hot Swap Specification (PICMG 2.1 R1.0)
 - Led software aspects of Revision 2.0—just adopted by PICMG in January, 2001
- Chair of PICMG Software Interoperability Technical Subcommittee (for PICMG 2.12)
- Led software aspects of first demo and products for Specific Use hot swap



Pigeon Point Systems (Cont.)

- First demo and product shipments for General Use Hot Swap products
- Licensed Hot Swap Kit to Microsoft for Windows XP/2002 Embedded
 - Also to leading CompactPCI platform & board vendors
- Member of Advisory Board: StarGen, Inc.
- Founding member of StarFabric Working Group

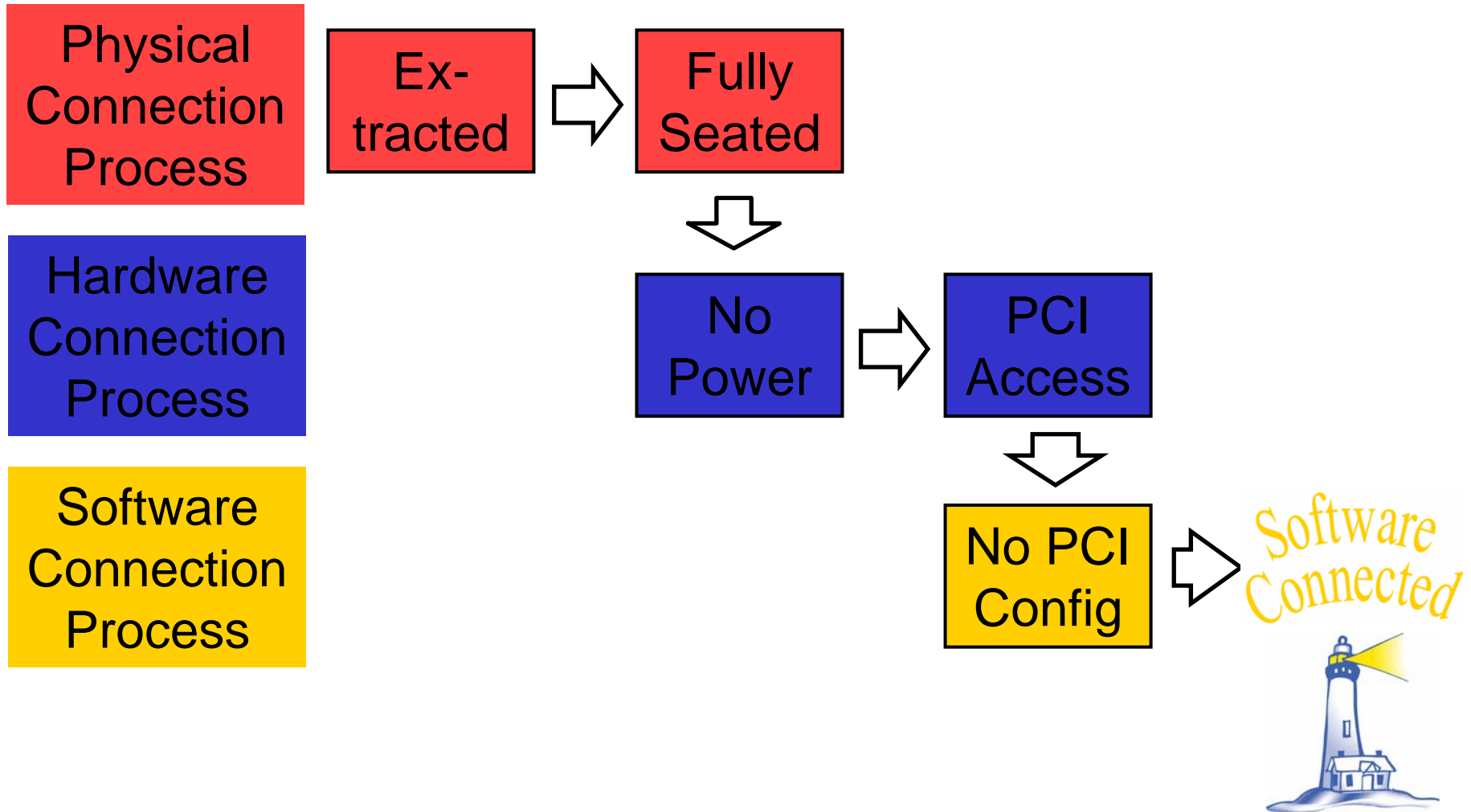


CompactPCI Hot Swap Approach

- Layered processes address different aspects, including hardware and software connection and disconnection
- Standardized h/w interface for s/w connection simplifies management
- Incremental system models:
 - Allow choice of appropriate capability level
 - Preserve high interoperability
 - Distinguished by how the h/w and s/w connection processes are controlled
- Specified by PICMG 2.1, with revision 2.0 just adopted in 01/01



Layered Connection Processes



Incremental System Models

Redundant System
Slot System

Addressed by PICMG
2.13, in development

High
Availability
System

Full Hot Swap
System

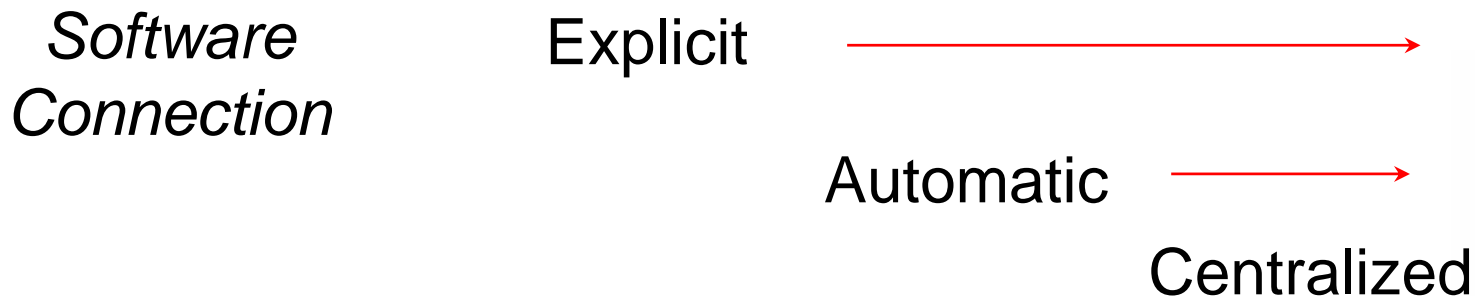
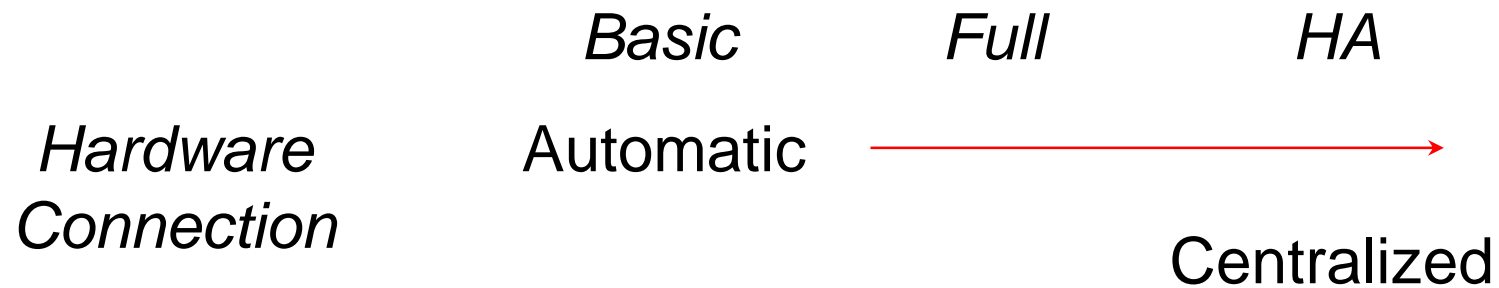
Basic Hot Swap
System

↑
INCREASING
COMPLEXITY



Types of Connection Control

- Explicit interaction w/ app. or operator
- Automatic with board insert/extract
- Centralized via system software



Centralized Hardware Connection Control

- HA platforms have radial controls for each slot via hot swap controller (HSC)
 - Allows HA system software to initiate connection or disconnection of a board
- All hot swap boards honor these controls
- Non-HA platforms default controls to support automatic H/W connection only
- Software interface to these controls is platform-specific



Automatic S/W Connection Control

- Switch in handle activates CompactPCI signal: ENUM#
 - After insertion & as 1st step of extraction
 - Processor detects via interrupt or polling
 - Detailed S/W interface is platform-specific
- On extraction, LED signals "OK to remove"
- Spec-defined Hot Swap Control/Status Register (HS_CSR) allows board-independent management
 - "Alternate" HS_CSR definitions allowed
- Augmented and clarified in PICMG 2.1R2.0



Key Changes in Revision 2.0 of Hot Swap Spec

- Focus here on changes affecting software
- Clarified definition of hot swap state machines to reduce h/w variations
- New programming interface in HS_CSR
 - Self-identifying; new level 1 is optional
 - Adds Device Hiding to address mid-transaction extraction problem
 - Adds Pending INSert/EXtract (PIE) bit
 - Ensures s/w synch w/ handle state
- Addressing startup issues on hot-inserted boards



7	6	5	4	3	2	1	0	Bit Number
INS	EXT	PI		LOO	PIE	EIM	DHA	Field Name

Device Hiding Arm
 1 = Device Hiding Arm
 0 = Device Hiding Disarm

ENUM# Signal Mask
 1 = Mask Signal
 0 = Enable Signal

Pending INSert or EXTRACT
 1 = One of them is pending
 0 = Neither is pending

LED ON/OFF
 1 = LED ON
 0 = LED OFF

Programming Interface
 0 = INS, EXT, LOO, EIM
 1 = Add PIE, Device Hiding
 2, 3 = Reserved

ENUM# Status - Extraction
 1 = ENUM# asserted
 0 = Not asserted

ENUM# Status - Insertion
 1 = ENUM# asserted
 0 = Not asserted

PICMG 2.1R2.0

HS_CSR w/

New Fields

Device Hiding

- PICMG 2.1R1.0 had narrow timing window for system crash
 - Accessing HS_CSR of extraction-approved board, while polling for new ENUM#
 - Potential h/w disconnect during HS_CSR access
- R2.0 board w/ Device Hiding (DH) enabled disappears from config space when:
 - Fully ready for extraction, with LED on
 - Hardware connected, but with handle switch still open



Three Invocation Scenarios for Device Hiding

- Operator-initiated s/w disconnect via handle switch
 - Handle switch opens to launch disconnect; DH invoked on completion, with LED on
- Software-initiated s/w disconnect
 - LED turned on when disconnect completes; DH invoked when handle switch opens
- Postponed s/w connection (e.g., on insertion)
 - Board is h/w connected, but DH remains invoked while handle switch remains open
- DH canceled when handle switch closes (locks) for all scenarios



Device Accessibility After Release of Local Reset

- PCI 2.2 allows device to take up to 2^{25} PCI cycles to respond to config cycles after reset
 - One relevant instance: CompactPCI interface on hot-inserted board
- During this period, R2.0 board can:
 - “Initially Retry” config access
 - Can hang system host non-interruptably on config retries
 - Not recommended in 2.1R2.0
 - Typical behavior for conventional PCI
 - “Initially Not Respond” to config accesses
 - No system host hang on retried config cycles
 - Recommended in R2.0, potentially via strapping option



2.1R2.0 Requirements

- All Hot Swap boards should Initially Not Respond
- Initially Retrying boards must:
 - Comply with 2^{25} cycle limit
 - Document factors that determine max retry period and ways to minimize it
- Hot swap infrastructure s/w identifying hot-inserted board with transparent P2P bridge:
 - Waits 2^{25} cycles for secondary config cycles
 - Thereby avoids impact from Initially Retrying devices behind that bridge



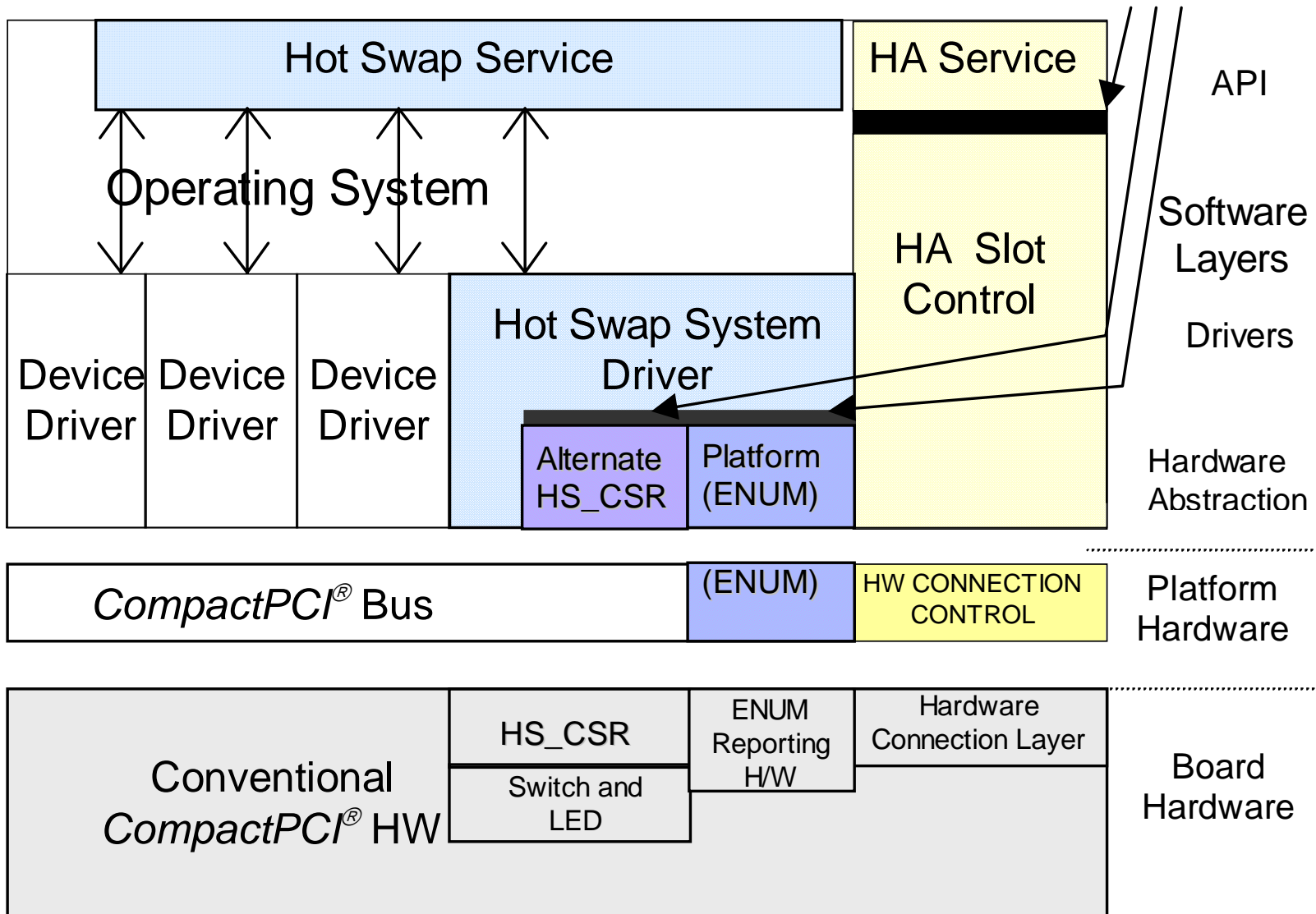
Hot Swap Infrastructure (HSI) Interface Spec (PICMG 2.12)

- Defines platform-, board- and OS-independent interfaces to
 - Platform Driver that fields ENUM# signal
 - Alternate HS_CSR Drivers for non-standard HS_CSR
 - HA Slot Control Driver
- Handled by Software Interoperability subcomm., based on APIs designed by PPS
- Additional 2.12-compliant HSIs coming from:
 - Microsoft in Windows XP Embedded (w/ PPS)
 - Wind River Systems in Foundation HA
- Open source repository for 2.12 low-level modules started by MontaVista Software



HSI Architecture

Interfaces Defined in 2.12



PICMG 2.12R2.0 Project Just Starting; Preliminary Goals

- Alignment with 2.1R2.0, including HS_CSR additions
- Defining common mechanisms for mapping between geographic & PCI logical addresses
- Updating to current releases for Windows and Linux
- Adding further OS coverage
- New set(s) of APIs addressing Redundant Systems Slot architectures



Key Ingredients for Robust Hot Swap Infrastructure S/W

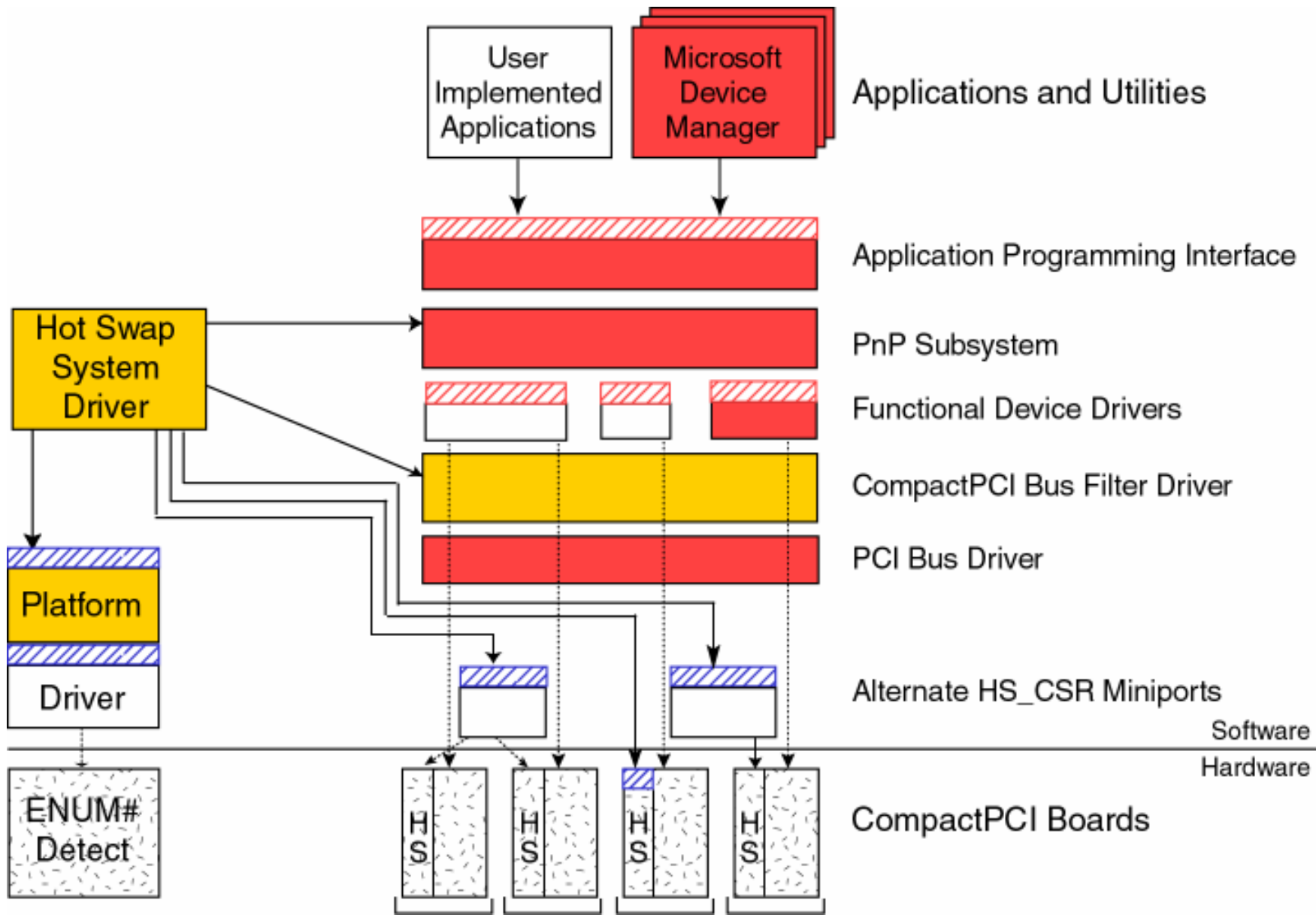
- PICMG spec compliance
- Integration with native O/S conventions and management frameworks
 - Critical to leverage existing software, especially for “mass market” O/Ss, such as Windows and Linux
- Aggressive automated test infrastructure for
 - HS infrastructure software
 - Hot-swap aware boards and applications
- Remainder of presentation shows how these ingredients can be achieved for
 - Windows 2000 and Windows XP/2002
 - Potentially for Linux with the 2.4 kernel



Hot Swap Support in Win2K & WinXP/2002

- Plug and play (PnP) can do S/W connection
 - But hot-plug/swap not general features
 - PCI and CompactPCI vendors will have to do qualification and support
- PnP device drivers should be hot swap capable and ubiquitous
 - Remove support may not be universal
- Still needed:
 - Hot swap s/w connection control
 - Integration w/ native facilities, such as slot and device designators
 - Configuring P2P bridges for hot insertions





PnP-based HSI

Slot and Device Designators

- Native internal PCI device designator in Win2K is “slot path”
 - Sequence of slot #s up bus tree to bus 0
 - Independent of PCI bus # changes
 - Includes bus # (re-) assignments on system reboots or hot insertion of bus segments
- Win2K also supports “User Interface Numbers” (UI#s)
 - Allows UI references to devices and slots to correspond to physical labels
 - Applied in any user interaction



Slot/Device Designators (Cont.)

- For cPCI, UI#s are physical or geographic slot numbers
- Hot Swap Infrastructure should
 - Add UI# to device nodes associated with physical slots
 - Apply native slot paths for internal device designation
- System-specific mapping between Win2K slot paths and CompactPCI geographic slot numbers is needed

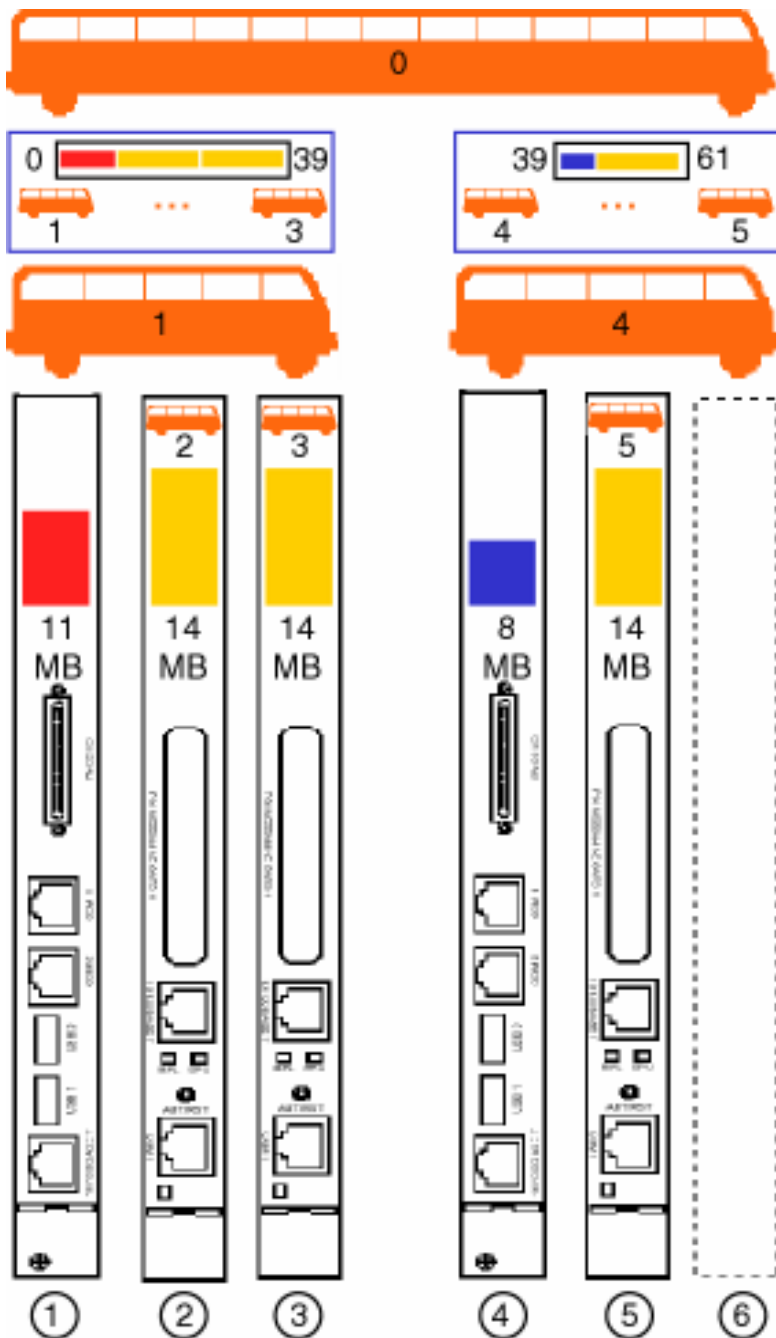


Designating and Configuring Bridges and Slots

- Need to configure PCI-to-PCI bridges to CompactPCI segments for hot insertion
 - Normal BIOS allocates “just enough” space for devices present at boot time
 - Sparse configuration of bridge “windows” leaves room for later insertions
- Slot paths provide convenient way to:
 - Identify bridges for configuration
 - Identify slots for association with geographic slot numbers

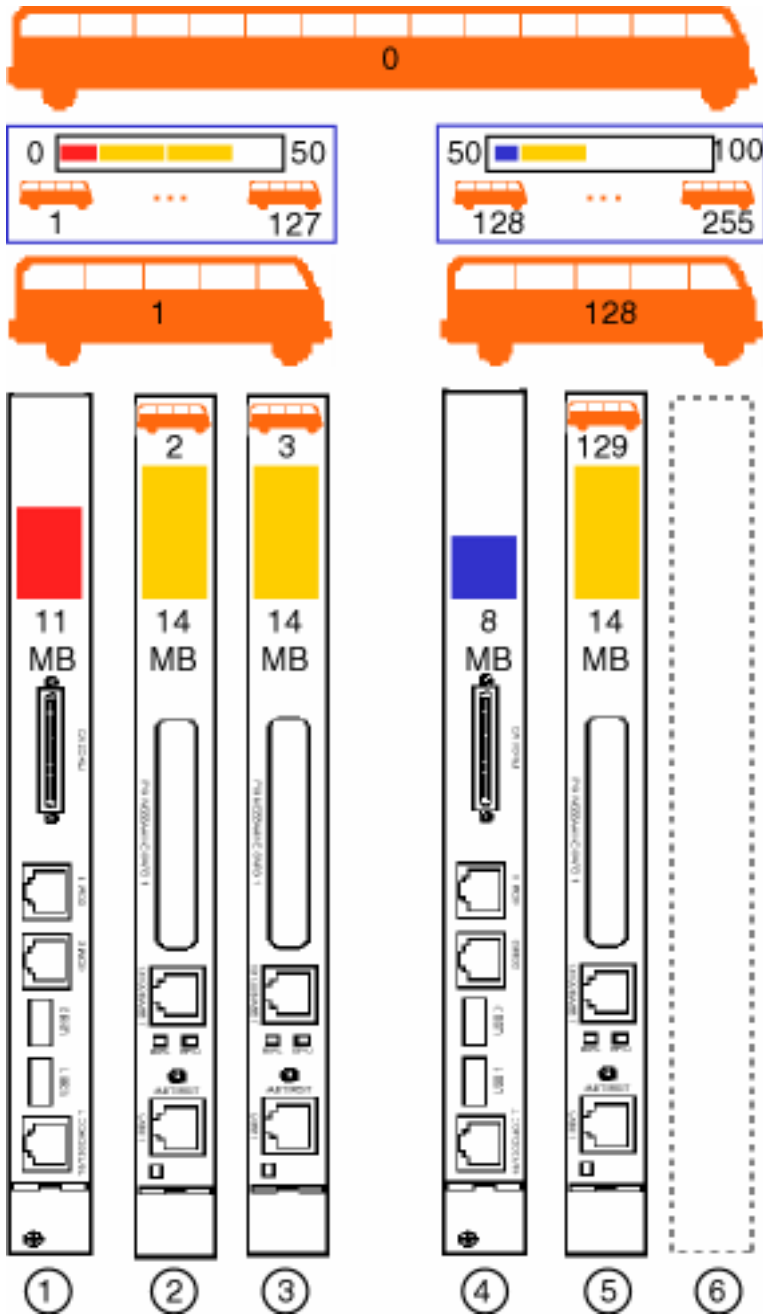


Problems w/ BIOS Initialized Config.



- Some hot swap actions impossible
 - Any insert in slot #6
 - Replace #4 by Gold
- Basic problem: “just enough” allocations
- Bus# + Device# designations for boards would depend on board population

Slot Paths (SPs) Enable Solutions

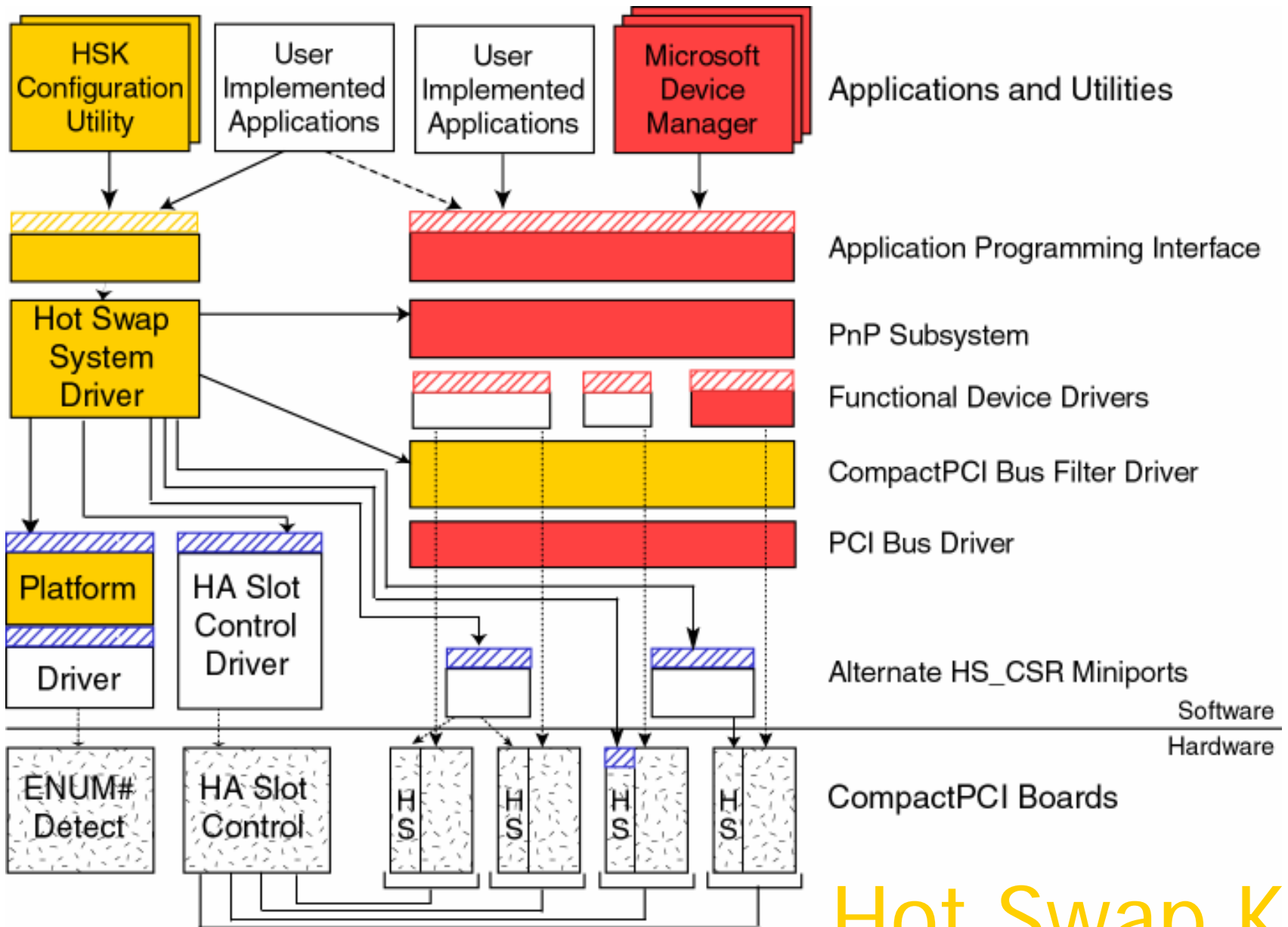


- Config tables can set windows for bridges
 - Designate bridges via SPs
 - System integrator still must choose window sizes
 - Example benefit: any board type hot inserts in slot #6
 - Still no configs w/ >50 MB per bus segment
- Config tables can also associate SPs w/ slot #s
- All tables unaffected by bus # changes

Pigeon Point Hot Swap Kit (HSK) for Win2K

- Adds hot swap support to Win2K, compliant with PICMG specifications:
 - 2.1R1.0: Hot Swap
 - 2.12R1.0: Hot Swap Infrastructure Interface
- Provides system integrator control of
 - Bridge window initializations
 - Slot path <--> UI# (slot #) mappings
- Provides supplementary APIs, for more detailed system control





Hot Swap Kit

Near-term Additions for HSK

- Still in planning; not yet definite
- Updates for PICMG 2.1R2.0, including:
 - Proper handling of Initially Retrying boards
 - Delay before enumerating behind hot inserted P2P bridges
 - Use Device Hiding to avoid “mid-transaction extraction” crashes
- PICMG 2.12-compliant HA slot control APIs
- Adding Windows Management Instrumentation (WMI) interface



Windows Management Instrumentation

- Microsoft's implementation of Distributed Management Task Force (DMTF) initiative
 - Web-Based Enterprise Management (WBEM) is generic; WMI is Microsoft's mapping
- Logically organized model of Windows operation, configuration & status
- COM API for providing and accessing management info
 - XML mapping adopted by DMTF
 - Access via XML over HTTP expected in WinXP



WMI (Cont.)

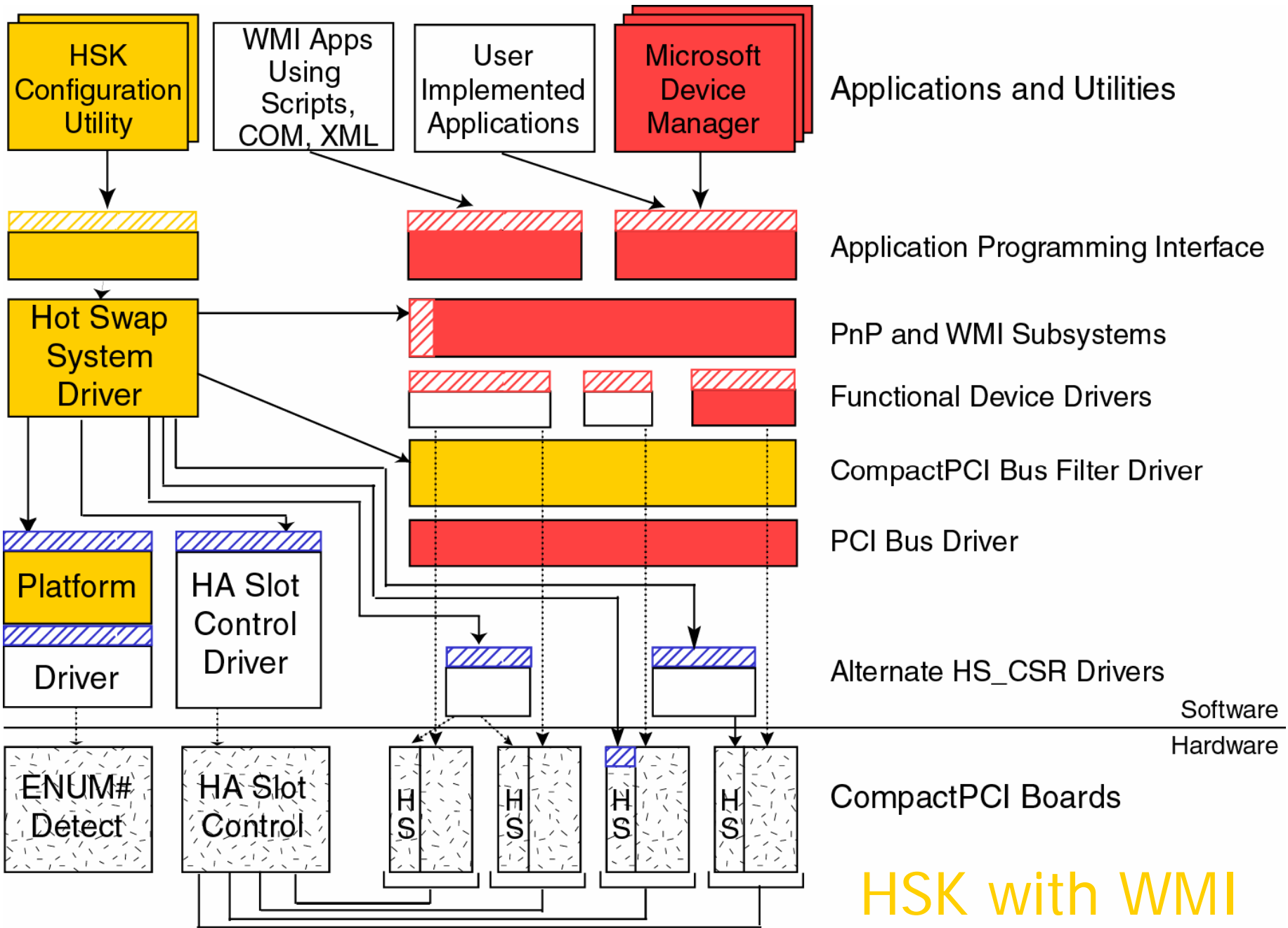
- Extensible by *providers* that link WMI to additional data (including SNMP, DMI...)
- Extensions to Windows Driver Model (WDM) capture data & events from drivers
- Enables desirable attributes
 - Language independence
 - Scriptability
 - Remote-ability



WMI Support for HSK

- Prototype designed, implemented & reviewed with Microsoft in 2000
 - Augmented Hot Swap System Driver uses in-kernel WMI to interact w/ WMI clients
 - New CIM schema provide essentially all functionality of HSK API
 - Microsoft review positive, with several key suggestions
- Productizing now underway, addressing MS suggestions
 - Development focused on a specific platform
 - General release plan for HSK not yet finalized





Potential: Equivalent Hot Swap Support in Linux

- Linux 2.4 kernel includes built-in support for dynamic PCI devices
 - Augmented PCI driver model enables single driver for PCI Hot-Plug, Cardbus & CompactPCI hot swap
 - Similar to Windows Driver Model (WDM), but simpler
 - PCI drivers compatible with that new model should be ubiquitous
- Red Hat 7.1 and other major distributions picking up 2.4 kernel
- Crucial limitations remain, but small additions to 2.4 kernel can address them



Necessary Additions for CompactPCI

- CompactPCI s/w connect. control, preferably:
 - A separate subsystem
 - Compliant with PICMG 2.12
- Hot insertion/removal of P2P bridges
 - Crucial for CompactPCI, including PCI sub-trees
- Dynamic configuration of resource windows in P2P bridges
- Controlled allocation of resource windows at startup
- Identification for CompactPCI slots that is persistent across boots



Preliminary Approach to the Necessary Additions

- Working prototype of this approach done in 2000 on 2.3.99 kernel
- Persistent slot identification mechanism
 - Adding PCI slot paths, as in Win2K
 - Two new functions convert between slot paths & pointers to PCI device descriptors
- Controlled allocation of bridge resource windows at startup
 - Allows user to specify window sizes via either
 - A sequence of macros in a separate file for recompiled kernel, or
 - In the kernel boot param string w/o recompilation
 - One new function retrieves window specs



Preliminary Additions (Cont.)

- Dynamic allocation of resource windows for bridges
 - Modifications to existing code in kernel that validates BIOS' bridge resource windows
- Insertion/removal of PCI sub-trees
 - Generalize the process of inserting/removing devices
 - Recursively include sub-tree behind a P2P bridge
 - Two new functions



Automated Testing and Remote Access Facilities

- Based on 3+ years of hands-on work with hot swappable & RSS cPCI systems
 - Challenges in both development and test
- Challenge #1: cPCI systems can be relatively large, expensive & scarce
- Challenge #2: thorough s/w testing of hot swap, RSS systems must be automated
- Solution: Pigeon Point Virtual Lab allows:
 - Productive use of shared h/w that is geographically distributed
 - Automated control of h/w



Pigeon Point Virtual Lab

- Attachment points to hardware allow
 - Control via relay, on a per-board basis, of...
 - Hot swap handle switch
 - Board reset switch
 - Maybe other jumpered controls, e.g., VxWorks boot ROM vs. PPC-Bug on MCG boards
 - Connection to on-board serial line of target boards
 - Power bar control of main power to system
- Virtual Lab (VL) Server: provides software abstraction for using/managing these controls
- Clients (interactive users or programs):
 - Connect to VL Server via rsh/ssh login, or
 - Use VL client software (Win2K/Linux)



Virtual Lab (Cont.)

- VL Server software
 - Provides management, resource locking, drivers, interactive tools and C/shell API
 - Maintains flat-file database of managed hardware nodes
 - Supported on Win2K and Linux
- VL Client software
 - Provides script-friendly API
 - Connects to server via RSH or SSH
 - Allows any networked Windows/Linux machine to be a test controller
 - For example: engineer's desktop machine can control and initiate test sequences via VL server

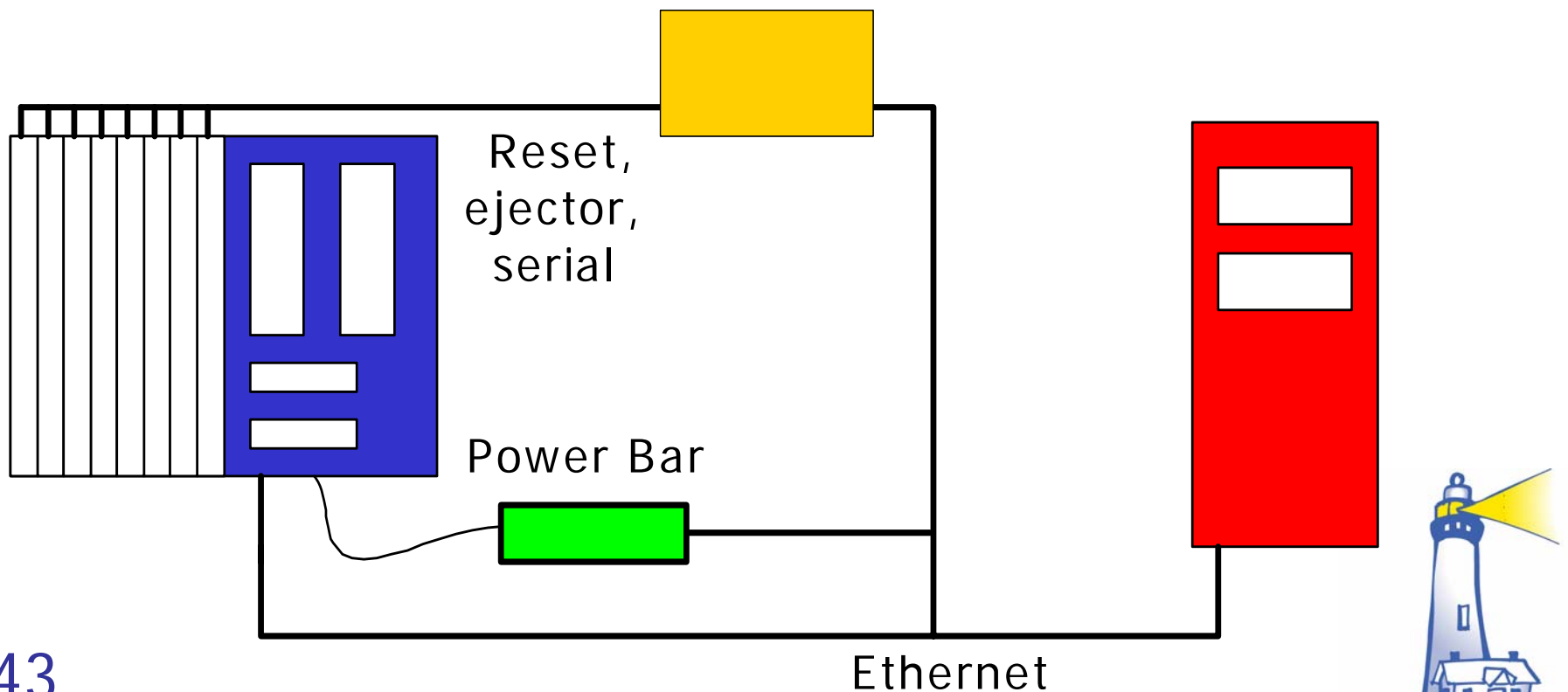


Virtual Lab Diagram

**Target
chassis**

VL server

**VL client/test
master**



Conclusion

- Robust hot swap infrastructure software requires thorough
 - PICMG spec compliance
 - Integration with native O/S conventions and management frameworks
 - Application of automated testing
- Pigeon Point Systems' Hot Swap Kit
 - Fully complies with PICMG 2.1 and 2.12
 - Integrates thoroughly with Win2K conventions for maximum leverage of existing software
 - Is being integrated with WMI framework
 - Is hardened by aggressive automated testing



Conclusion (Cont.)

- Most of HS robustness requirements feasible with Linux, given additions in 2.4 kernel
- Overall: hot swap software and hardware is ready for production use
 - Go for it!





Questions?

Mark Overgaard

Pigeon Point Systems

831-438-1565

mark@pigeonpoint.com

