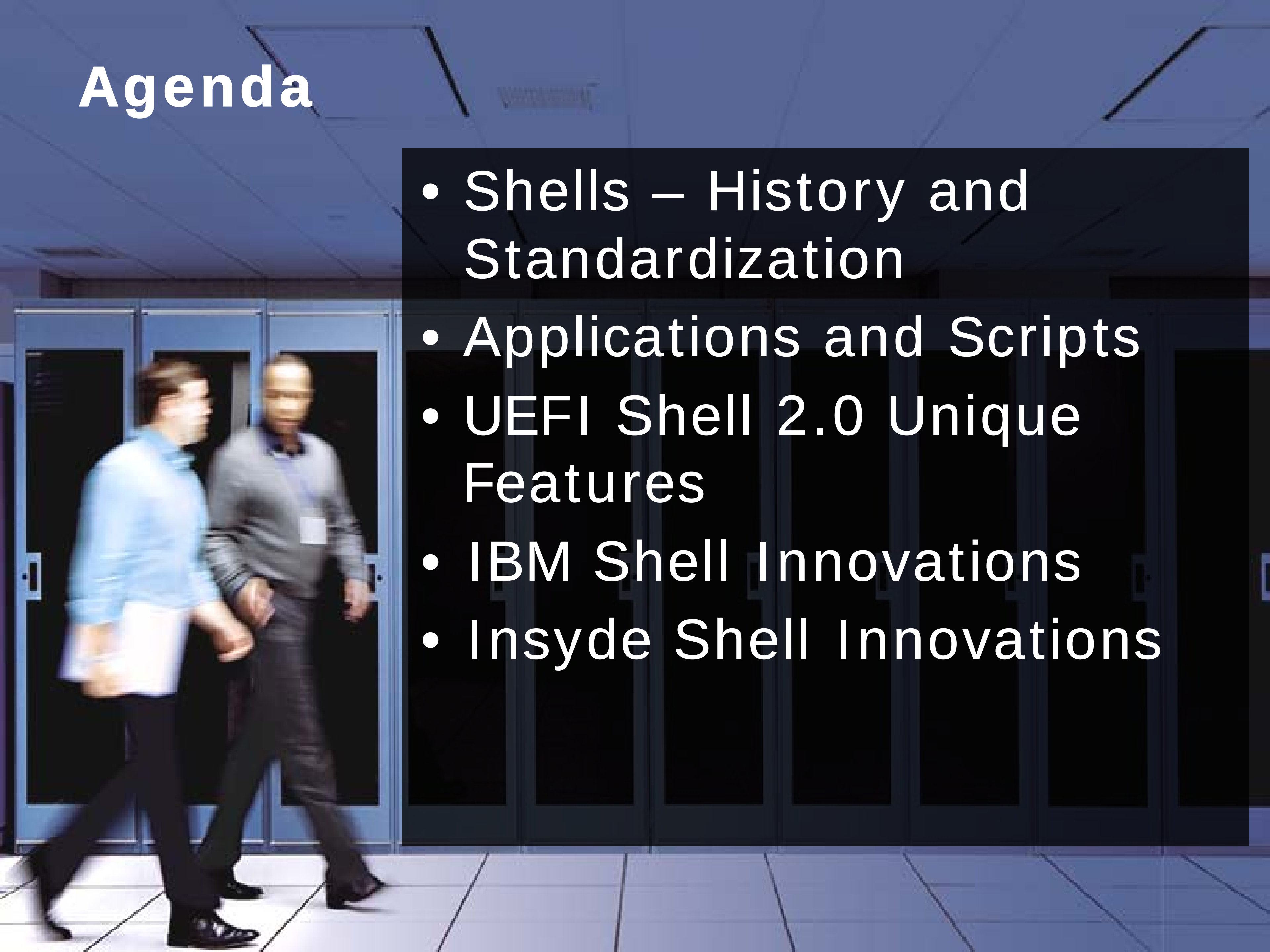
Beyond DOS: The UEFI Shell – a Modern Pre-boot Application Environment

Mike Rothman, Sr. Staff BIOS Software Engineer, Intel
Jaben Carsey , Staff BIOS Engineer, Intel
Nathan C Skalsky, Staff Firmware Engineer, IBM
Jeff Bobzin, UEFI Architect, Insyde

EFIS005

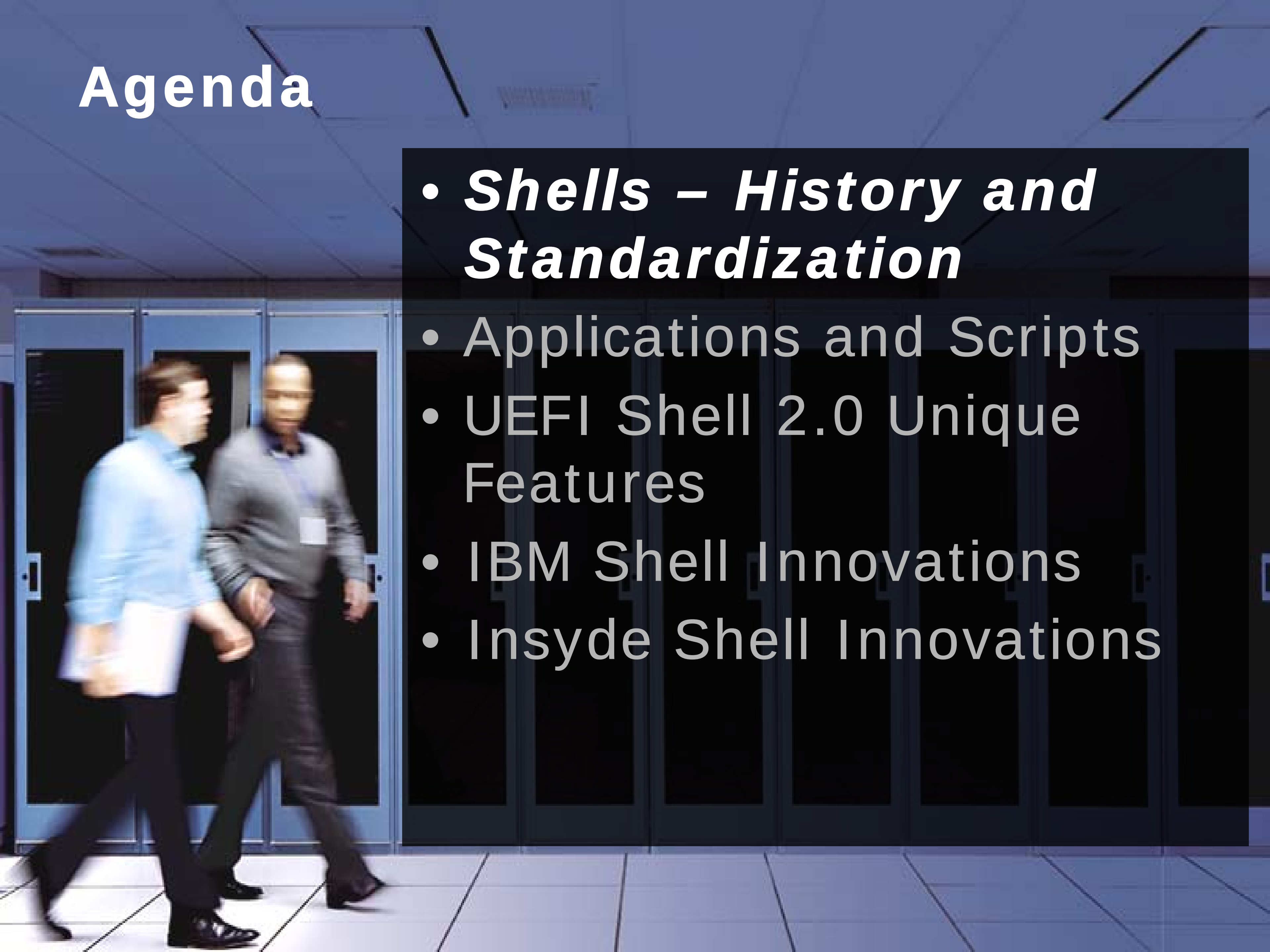
Sponsors of Tomorrow. intel

Agenda



- Shells – History and Standardization
- Applications and Scripts
- UEFI Shell 2.0 Unique Features
- IBM Shell Innovations
- Insyde Shell Innovations

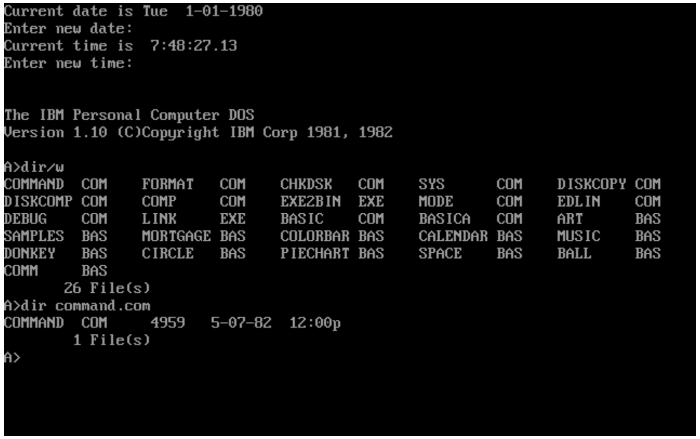
Agenda



- ***Shells – History and Standardization***
- Applications and Scripts
- UEFI Shell 2.0 Unique Features
- IBM Shell Innovations
- Insyde Shell Innovations

Shells – History and Standardization

- History of Command-lines
  - Early 1970’s – Unix arrives
  - Early 1980’s – DOS arrives
  - Today – Most O/Ses expose a command-line
- Command-line uses
  - Scripting
  - Program Launching
  - Abstraction to underlying system
  - Bring-up Target

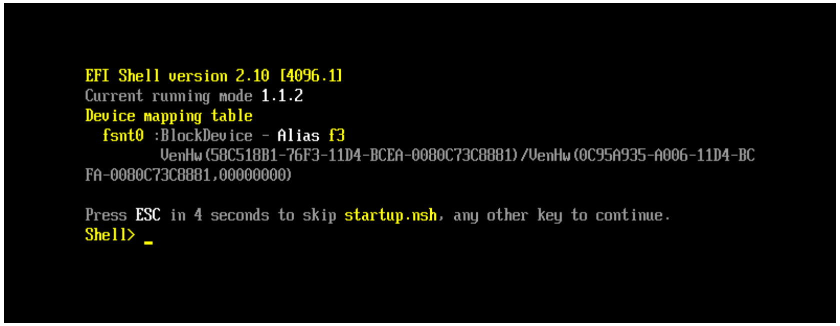


```
Current date is Tue  1-01-1980
Enter new date:
Current time is  7:48:27.13
Enter new time:


The IBM Personal Computer DOS
Version 1.10 (C)Copyright IBM Corp 1981, 1982

A>dir/w
COMMAND  COM    FORMAT   COM    CHKDSK   COM    SYS      COM    DISKCOPY COM
DISKCOMP COM    COMP     COM    EXE2BIN  EXE    MODE     COM    EDLIN    COM
DEBUG    COM    LINK     EXE    BASIC    COM    BASICA   COM    ART      BAS
SAMPLES  BAS    MORTGAGE BAS    COLORBAR BAS    CALENDAR BAS    MUSIC    BAS
DONKEY   BAS    CIRCLE   BAS    PIECHART BAS    SPACE    BAS    BALL     BAS
COMM     BAS
       26 File(s)
A>dir command.com
COMMAND  COM     4959   5-07-82  12:00p
        1 File(s)
A>
```

PC DOS 1.1

Modern version

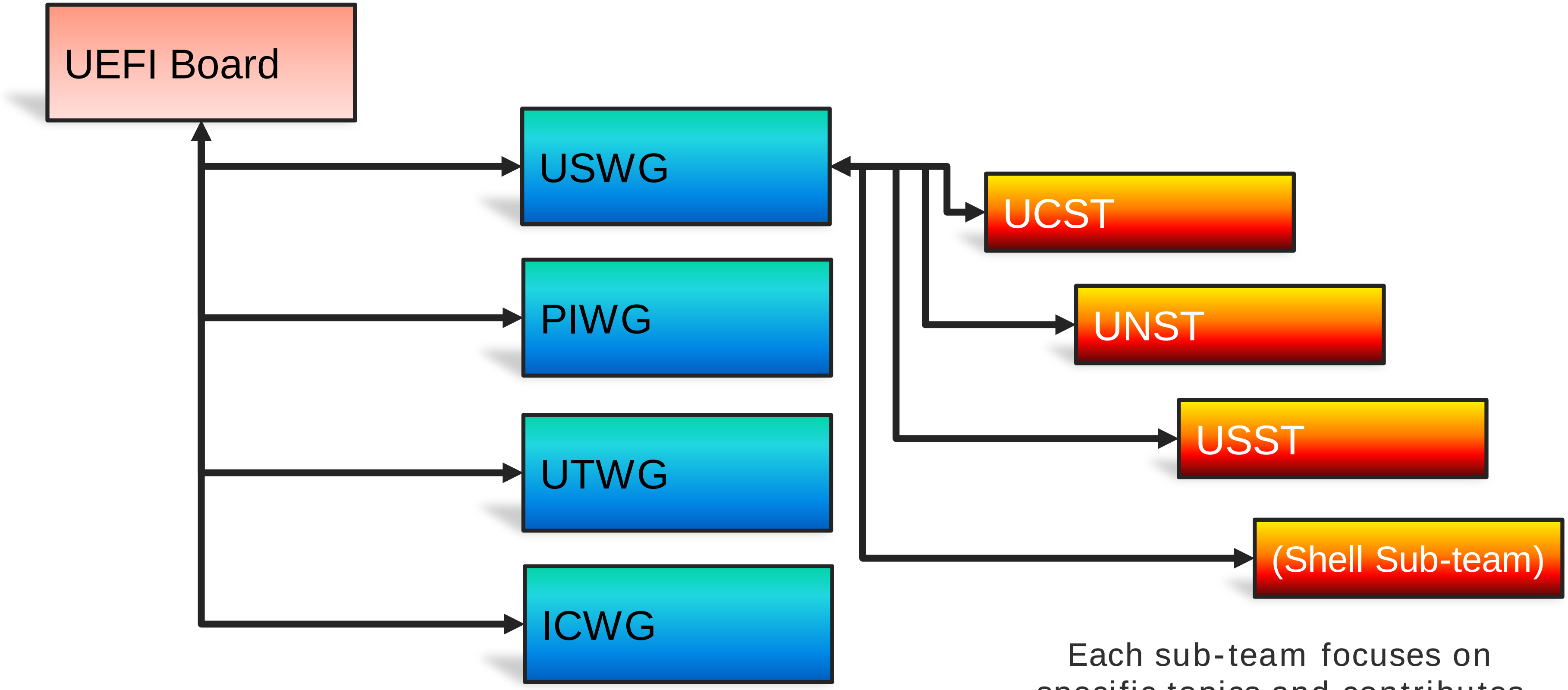
of DOS



```
EFI Shell version 2.10 [4096.1]
Current running mode 1.1.2
Device mapping table
  fsnt0 :BlockDevice - Alias f3
        VenHw(58C518B1-76F3-11D4-BCEA-0080C73C8881)/VenHw(0C95A935-A006-11D4-BC
FA-0080C73C8881,00000000)

Press ESC in 4 seconds to skip startup.nsh, any other key to continue.
Shell> _
```

UEFI Shell

Shells – History and Standardization



Publications/Decisions ratified by the board

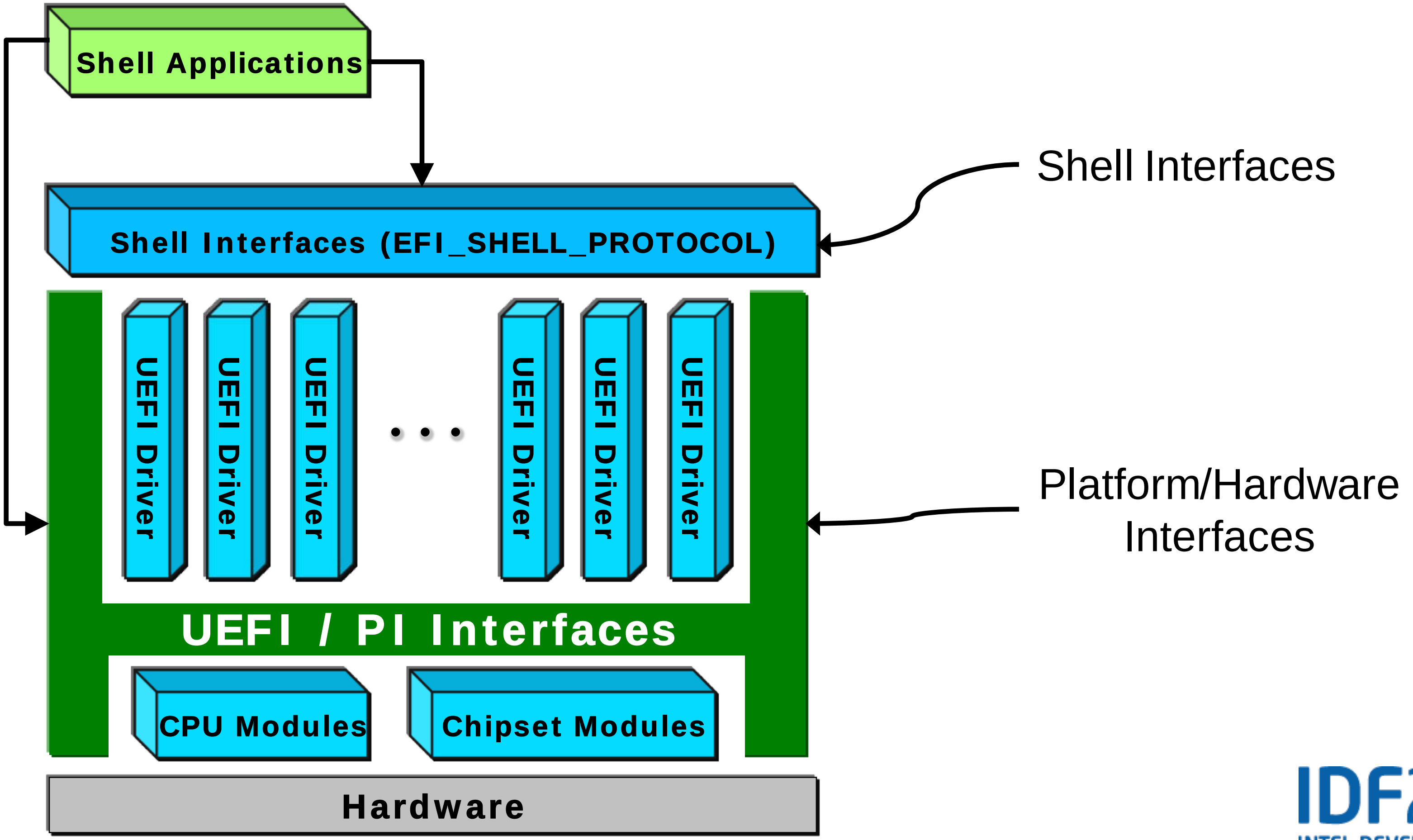
Each work group approves/delivers different content to the public

Each sub-team focuses on specific topics and contributes material to the work group

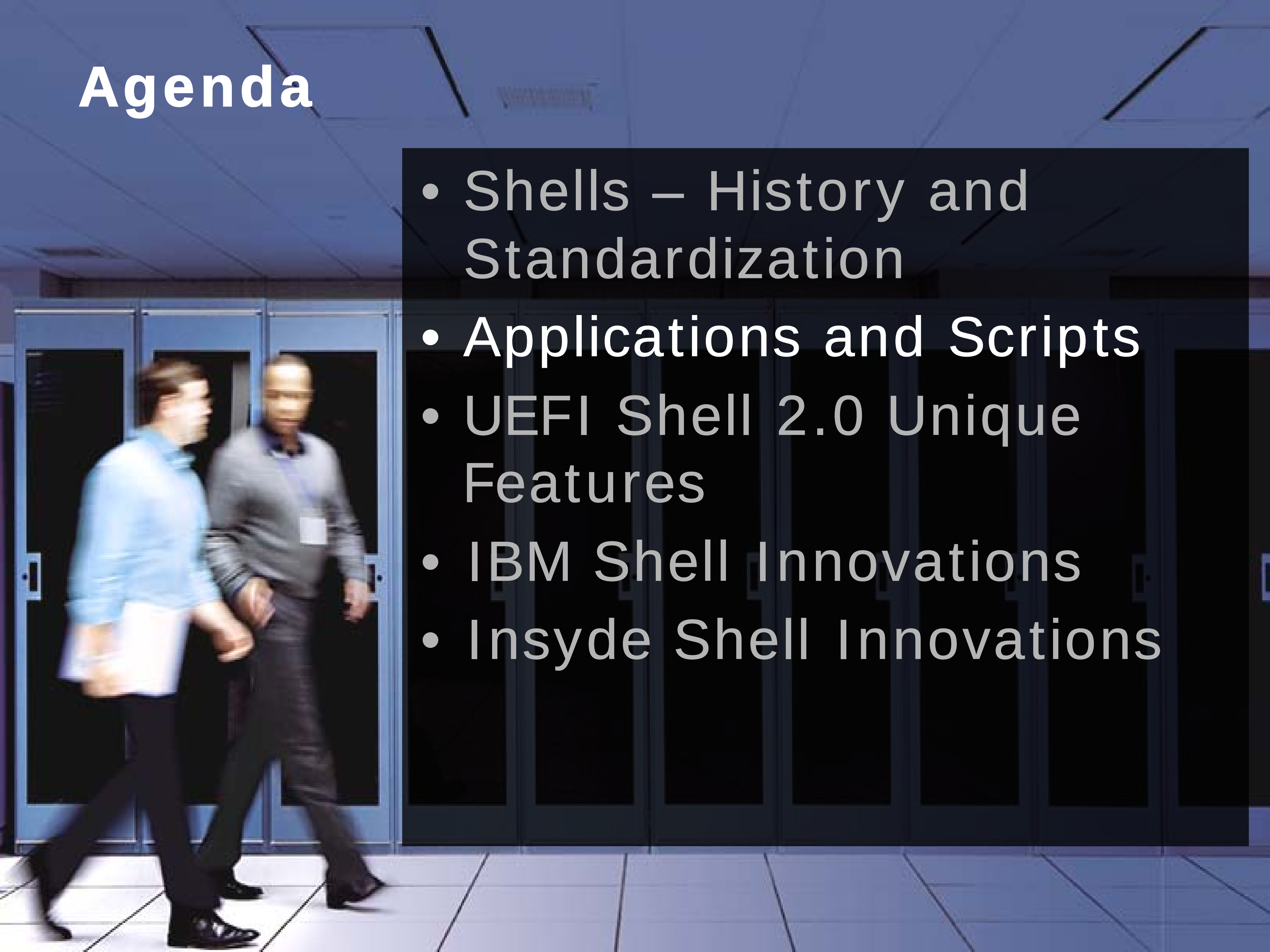
The presence of standards enables interoperability

Shells – History and Standardization

- Reusable code regardless of UEFI implementation.
  - Due to scripting and programmable methods being standardized.

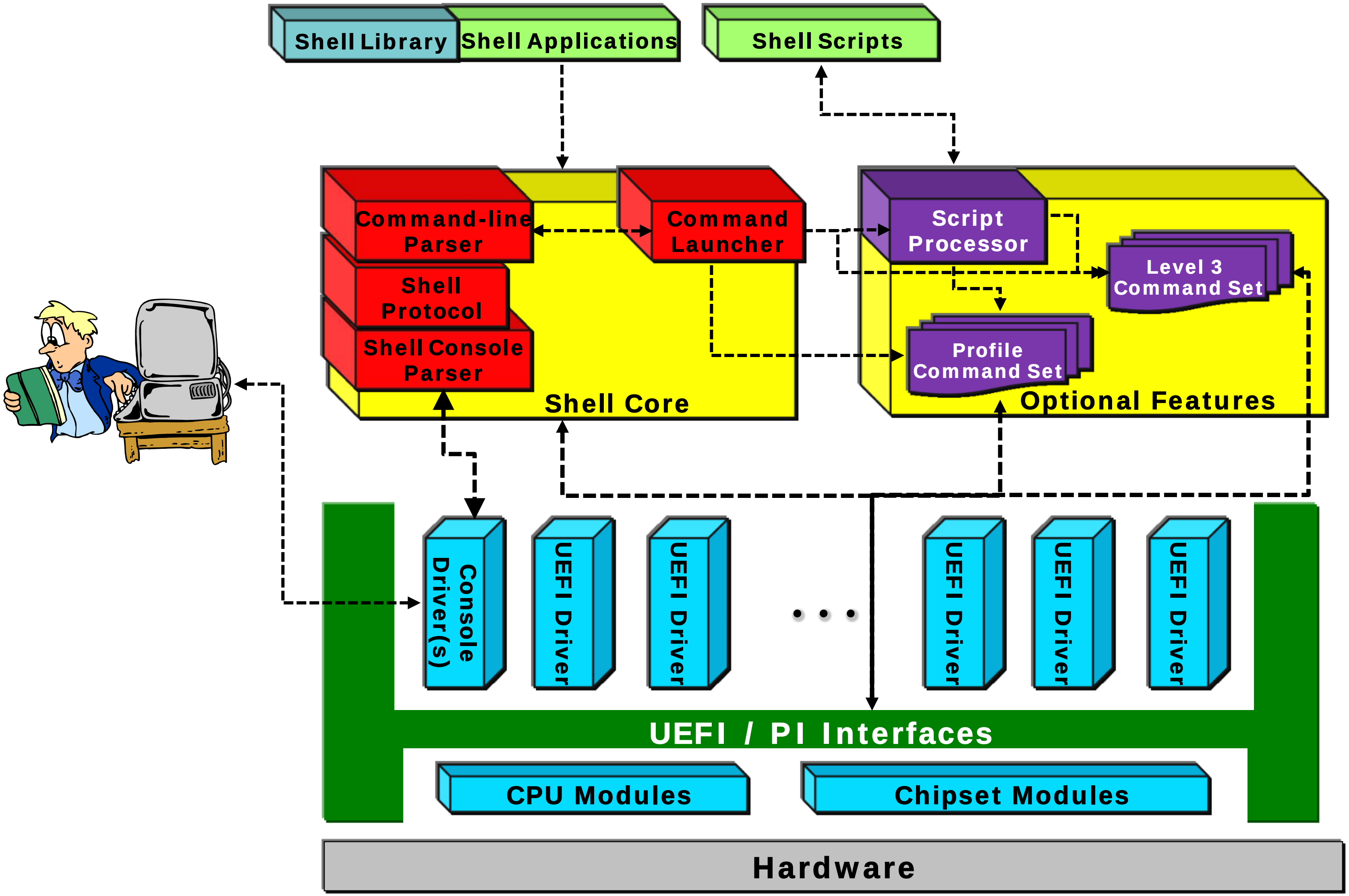


Agenda



- Shells – History and Standardization
- **Applications and Scripts**
- UEFI Shell 2.0 Unique Features
- IBM Shell Innovations
- Insyde Shell Innovations

A UEFI Shell 2.0 Architecture



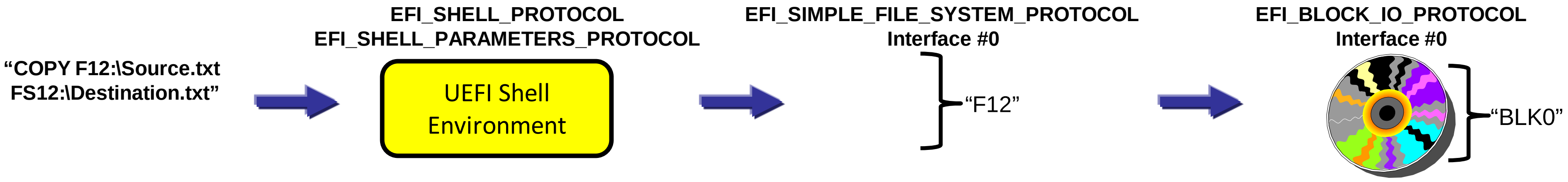
Shell Applications

- UEFI Shell 2.0 applications are compiled C code binaries that:
  - Use a Shell protocol
    - **EfiShellProtocol** – provides APIs for file IO and Shell Environment IO
    - **EfiShellParametersProtocol** – provides Std I/O and Argc/Argv
  - Optionally use UEFI protocols
  - Are launched from command line, script, or in startup parameters to the shell itself

Shell Applications replace EFI Shell Extensions

Shell Scripts

- Shell Scripts (.nsh files) provide automated execution of sequences of shell commands, shell or UEFI applications, and other shell scripts
- Support complex logic via For, If, and Goto
- Route human readable commands to correct hardware



"COPY F12:\Source.txt FS12:\Destination.txt"

EFI_SHELL_PROTOCOL
EFI_SHELL_PARAMETERS_PROTOCOL

UEFI Shell Environment

EFI_SIMPLE_FILE_SYSTEM_PROTOCOL
Interface #0

"F12"

EFI_BLOCK_IO_PROTOCOL
Interface #0

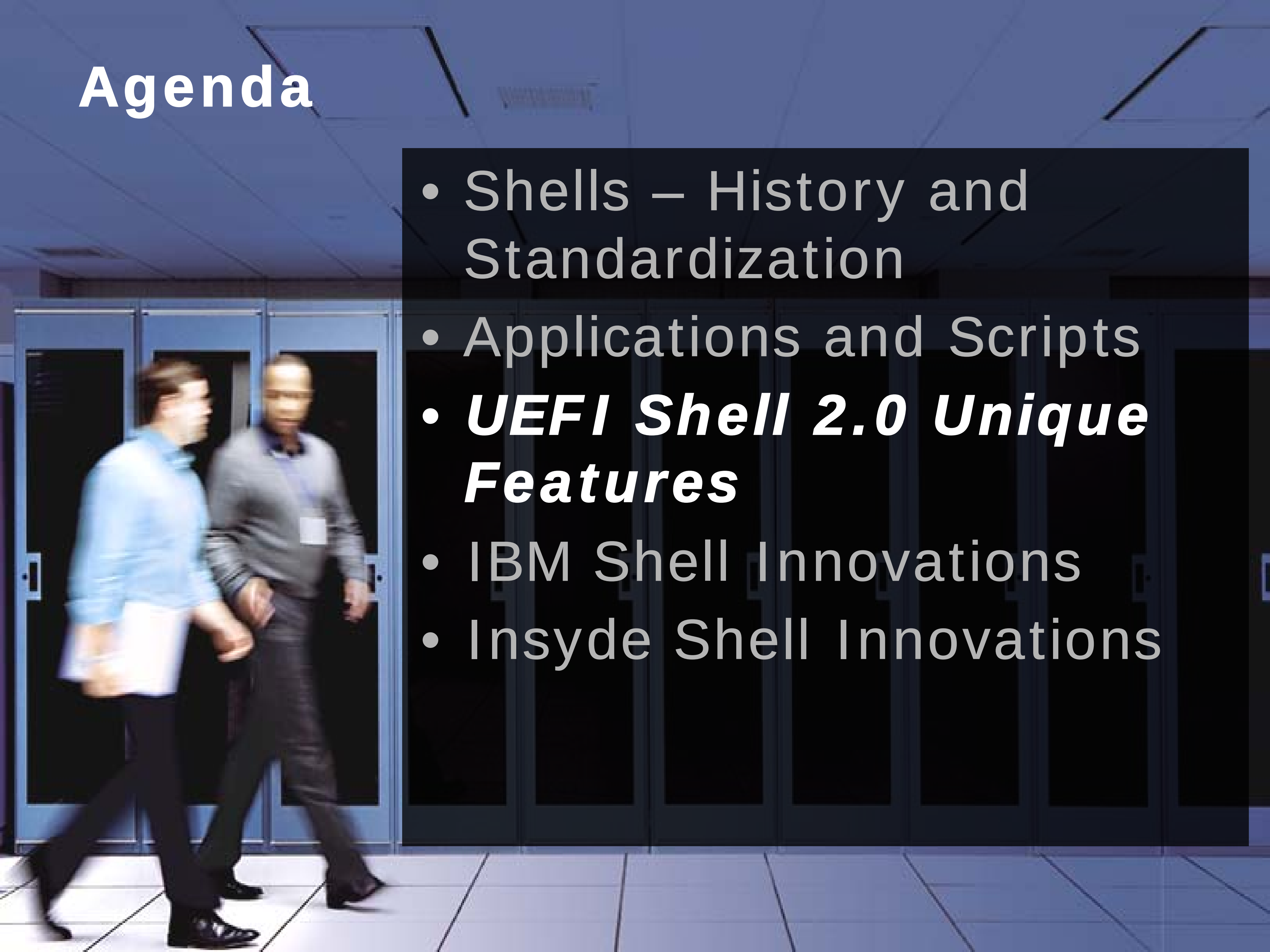
"BLK0"

What's Changed?

- EFI Shell scripts remain compatible
- UEFI 2.0 Scripts have additional capabilities
  - Query for command availability
  - Consistent Command feature sets

- Old Shell Protocols deprecated
- UEFI Shell Protocols added
  - EFI Shell extensions require porting
  - UEFI applications will work
- New UDK Shell Lib supports both Protocols

Agenda



- Shells – History and Standardization
- Applications and Scripts
- ***UEFI Shell 2.0 Unique Features***
- IBM Shell Innovations
- Insyde Shell Innovations

Key UEFI Shell 2.0 Features

New features provided by UEFI Shell are:

- Configure command sets available to end users configured at built time
- Provide backwards compatibility with existing shell scripts
- Manage firmware image size

```
[PcdsFixedAtBuild]

gEfiShellPkgTokenSpaceGuid.PcdShellSupportLevel  | 3
## bit 0 = Drivers1, bit 1 = Debug1, bit 2 = Install1, bit 3 = Network1
gEfiShellPkgTokenSpaceGuid.PcdShellProfileMask    | 0xF
```

UEFI Shell Command Sets

- Shell Levels manage main features
  - Level 0 – Launching a single application
  - Level 1 – Adds scripting
  - Level 2 – Adds file manipulation
  - Level 3 – Adds UI and information retrieval
- Shell Profiles manage additional commands
  - Install – Adds OS loader configuration
  - Debug – Adds debug
  - Driver – Adds driver manipulation
  - Network – Adds network configuration & test

Unique ability to balance required features and commands against desired binary size

Shell.EFI Image Size Management

- Maximum Image Size
  - – Level 3 shell with all 4 defined profiles
  - – Supports all standard commands and a UI for interaction with an user
- Minimum Image Size
  - – Level 0 shell with no profiles
  - – Supports launching a single application
- Additional extra profiles possible

64 size combinations available!

Agenda

- Shells – History and Standardization
- Applications and Scripts
- UEFI Shell 2.0 Unique Features
- ***IBM Shell Innovations***
- Insyde Shell Innovations

IBM Experience

“Smarter Firmware for a Smarter Planet”

Nathan C. Skalsky
Advisory Firmware Engineer, IBM
September 13th, 2010

Agenda

- IBM’s UEFI Shell 2.0 roadmap
- Key Features of UEFI Shell
- Running the UEFI Shell on IBM System x
- Example Uses of UEFI Shell
  - Bring Up
  - Development
  - System Manufacturing
  - Deployment/Provisioning
  - Maintenance
  - Debug
- Conclusion

UEFI Shell 2.0 Roadmap

- **Compatibility:** All UEFI-compliant System x Servers and Blades.
- **Integrated Shell:** a built-in level 3 UEFI Shell 2.0 is planned to be available via x86 IBM eX5 firmware updates within the next year.
  - Available as a Boot Item
  - Launch-able via UEFI Shell
- **Tools/CLI Strategy:** Current direction is to continue to use OS-based pre-boot deployments environments for flashing/in-band configuration updates. Shell is considered a supplementary command-line environment.

Key Features UEFI Shell 2.0

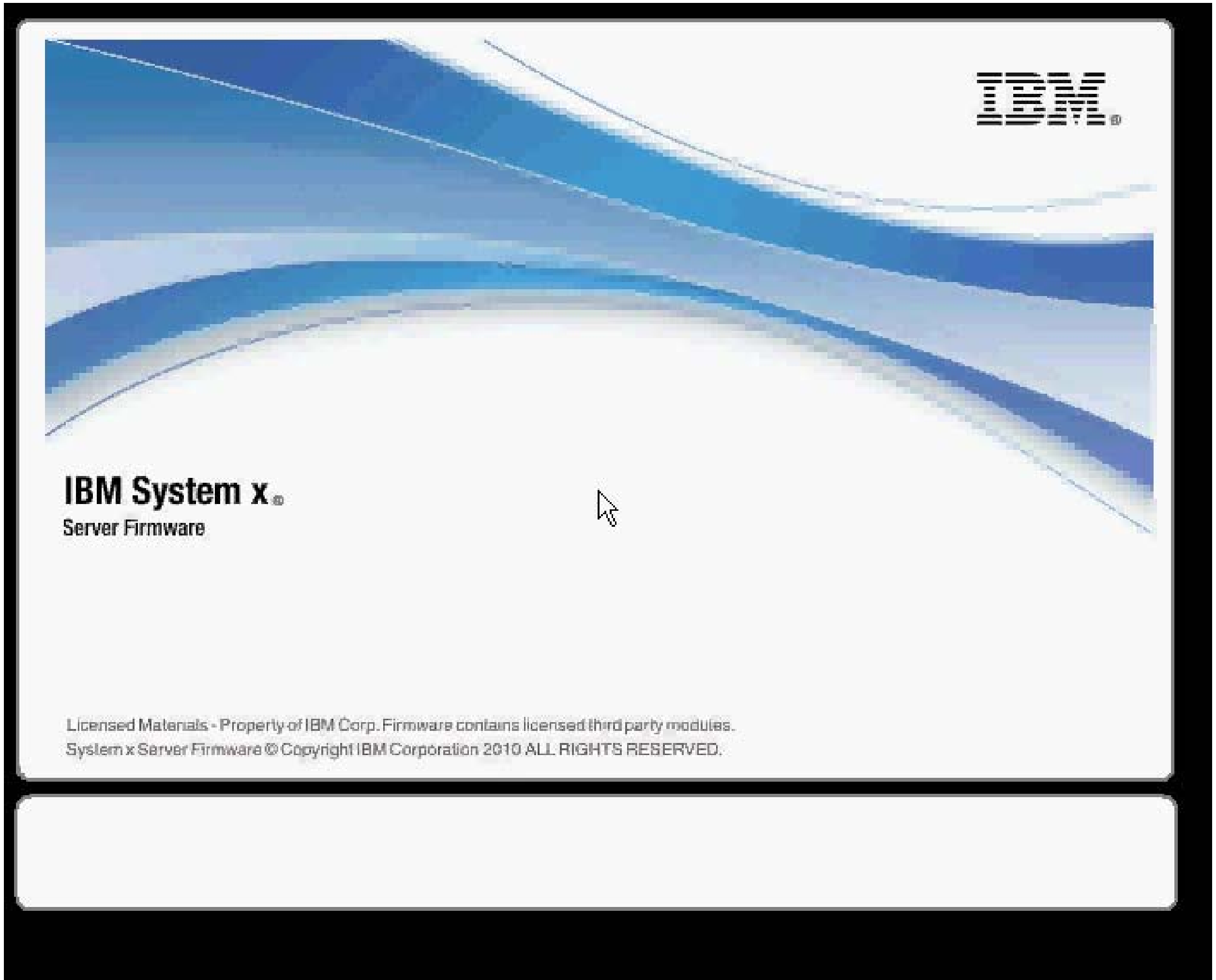
- "Common-Denominator" Preboot CLI
  - No dependencies on OS load / deployment
  - Direct-hardware environment
- Embeddable and flexible foot-print
- Scripting
  - Automating Configuration/deployment tasks
  - Automating Testing (reset tests, verify OS interfaces, run UEFI standards compliance tool (SCT)
- Shell Libraries enable ease of development and portability of applications
- Execute UEFI binaries
- Load/Unload Pre-Boot UEFI/DXE Drivers

Key Application Areas of UEFI Shell 2.0

- Early Hardware “Bring-up” Milestone
- Development and Testing
- Manufacturing
- Deployment/Provisioning
- Maintenance
- Debug / Product Support Investigations

Launching and Using UEFI Shell 2.0

- **One-Time**
  - F1 Setup → Boot Manager → Boot From File
- **As a Boot Option**
  - F1 Setup → Boot Manager → Add Boot Option



Licensed Materials - Property of IBM Corp. Firmware contains licensed third party modules.
System x Server Firmware © Copyright IBM Corporation 2010 ALL RIGHTS RESERVED.

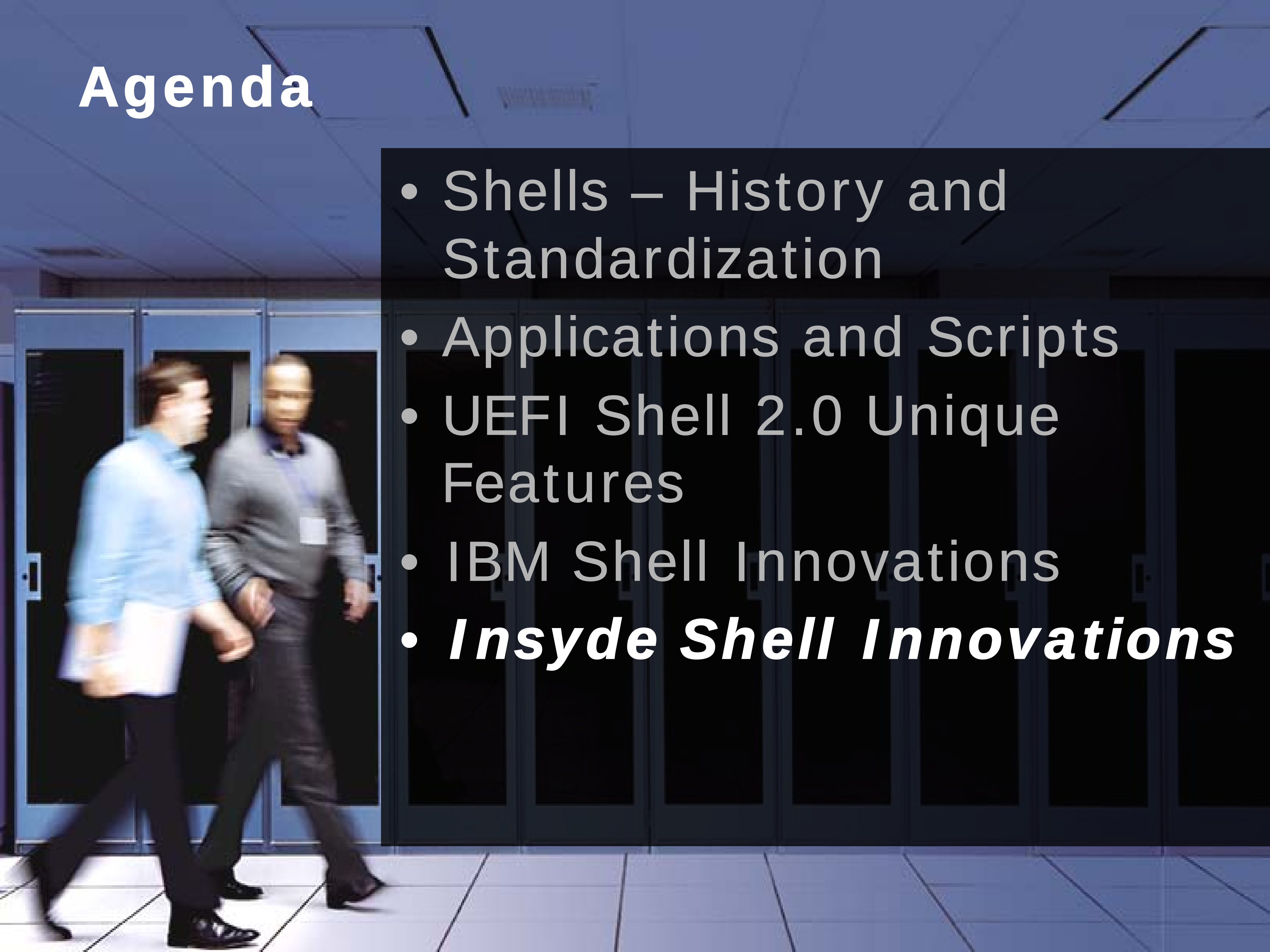
Using UEFI Shell during Development

- Scripting: Startup.nsh (think “Autoexec.bat”)
- Commands: DumpVariable/PCI/IbmHiiParse

Conclusion

- UEFI Shell 2.0 is a powerful “common denominator” environment and set of IO/console libraries
- IBM and IBM vendors heavily leverage the shell for bring-up, development, and debug activities
- Future is bright for automated and interactive configuration, deployment and update use-cases

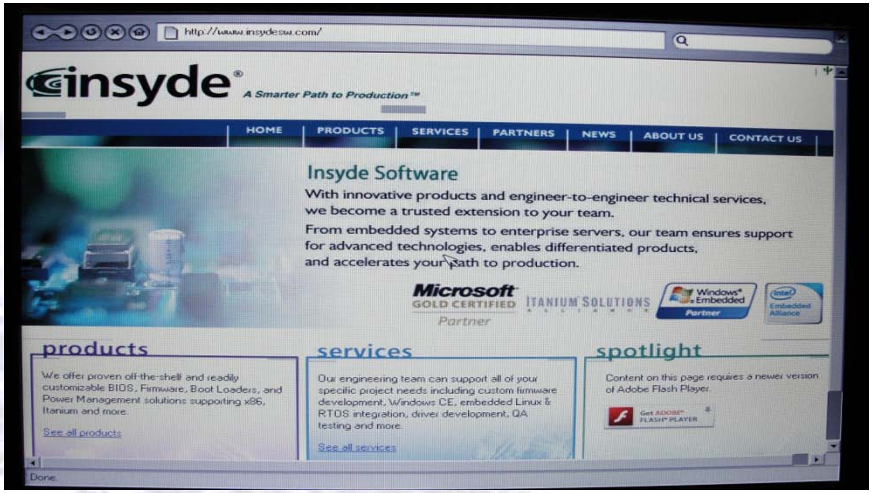
Agenda



- Shells – History and Standardization
- Applications and Scripts
- UEFI Shell 2.0 Unique Features
- IBM Shell Innovations
- ***Insyde Shell Innovations***

Using the Pre-Boot Application Environment

- Network Browsing



Insyde Software

With innovative products and engineer-to-engineer technical services, we become a trusted extension to your team.

From embedded systems to enterprise servers, our team ensures support for advanced technologies, enables differentiated products, and accelerates your path to production.

products

We offer proven off-the-shelf and readily customizable BIOS, Firmware, Boot Loaders, and Power Management solutions supporting x86, Itanium and more.

See all products

services

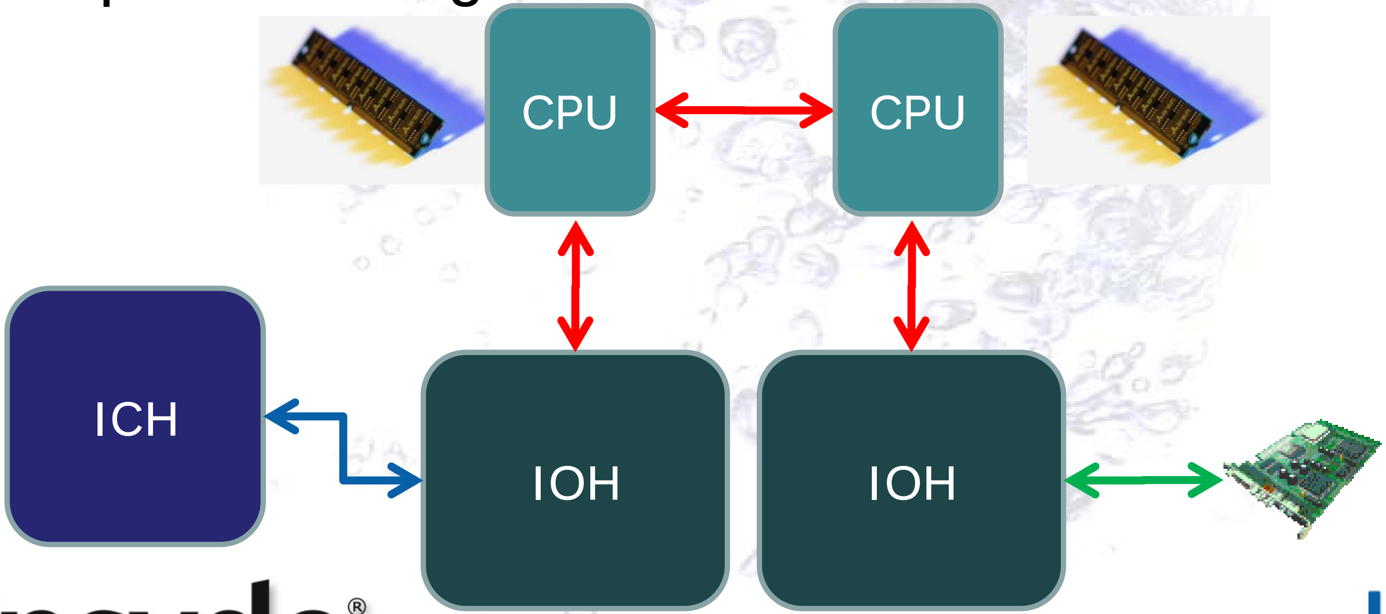
Our engineering team can support all of your specific project needs including custom firmware development, Windows CE, embedded Linux & RTOS integration, driver development, QA testing and more.

See all services

spotlight

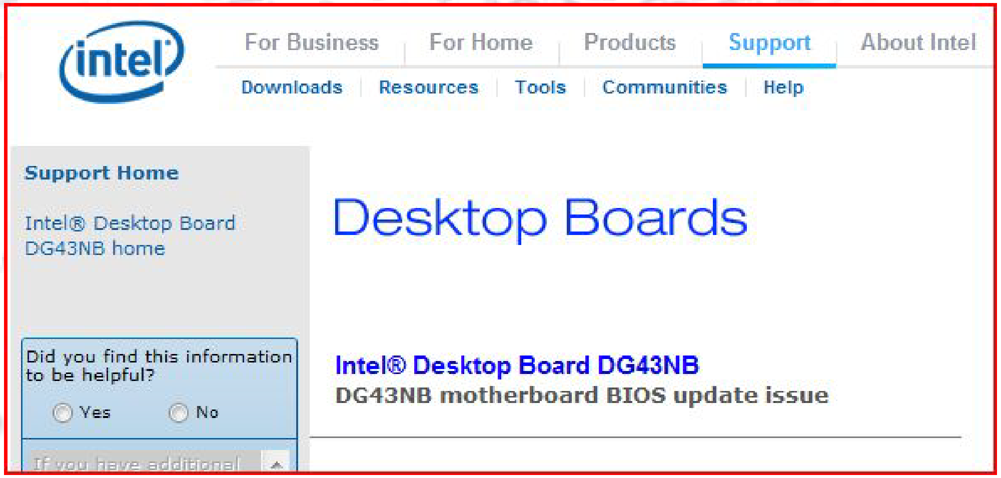
Content on this page requires a newer version of Adobe Flash Player.

- Complex Testing



Riding on top of a Network Capable Shell…

- Extends pre-boot space onto Internet
- Network Browsing Examples:
  - IT department support page
    - Help pages
    - Http download client
    - Access to OS recovery images
  - Remote assist system
    - System drivers download from OEM service site
    - Remote system diagnostic
    - Hardware support page



Desktop Boards

Intel® Desktop Board DG43NB

DG43NB motherboard BIOS update issue

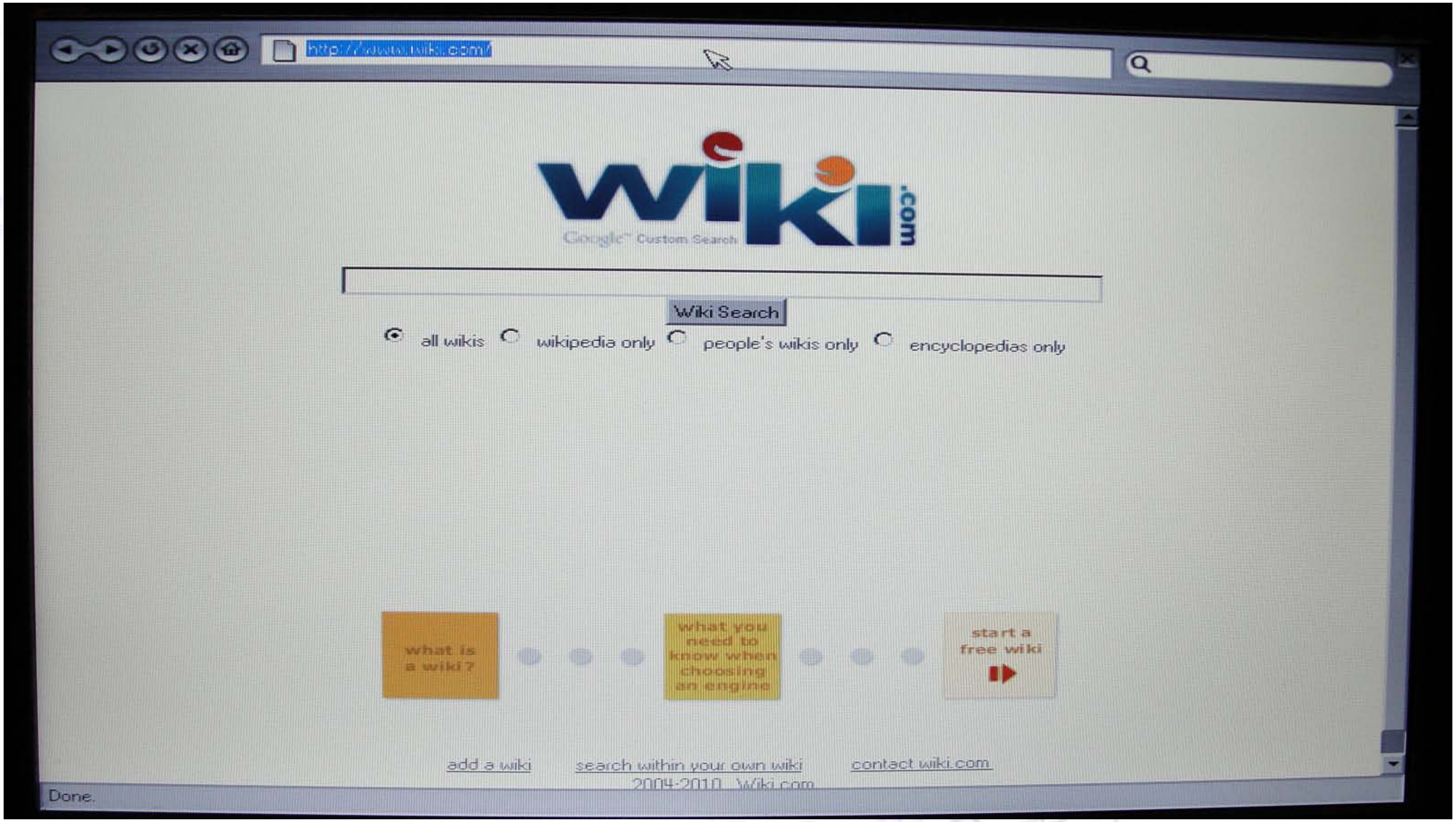
Did you find this information to be helpful?

○ Yes ○ No

If you have additional

Demo of Network Browsing

Network Browsing in UEFI…coming soon!



http://www.wiki.com/

● all wikis ○ wikipedia only ○ people's wikis only ○ encyclopedias only

Networking sets applications free in the pre-boot space

Complex Testing in a shell application

- Test hardware features not supported in OS
- Accelerate hardware feature development
  - Simpler debug environment than OS
  - More control for probing error conditions
- Enable efficient testing of features
  - Rapid test cycles booting just to UEFI Shell
  - Specific error cases can be validated

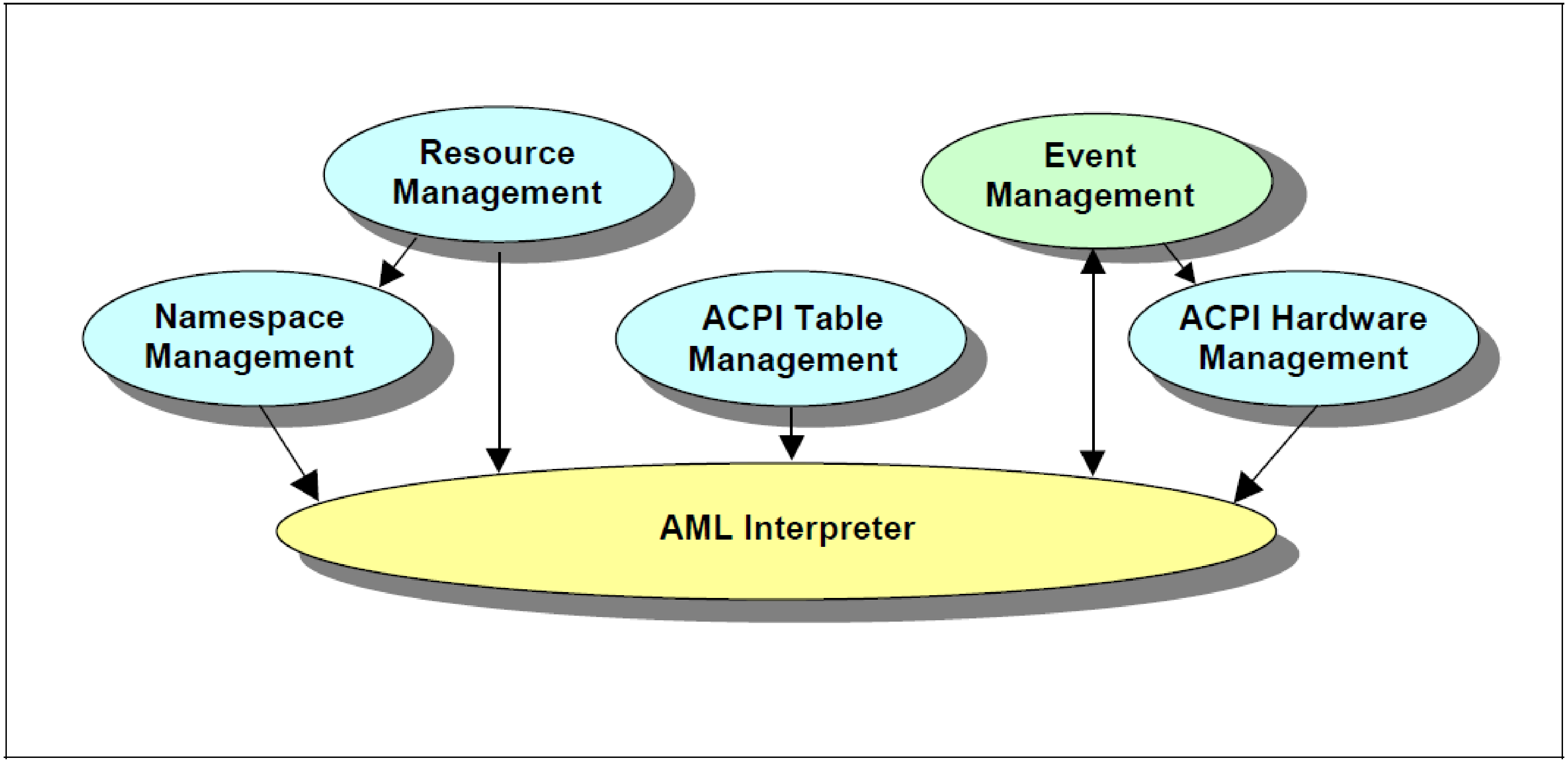
RAS Feature development and testing…

- OSes have limited support
  - Processor offline
  - Memory offline
- Need “live” ACPI environment
  - Methods and events supported
  - ACPI Component Architecture (ACPICA) is candidate
    - Designed for OS integration
    - Open source code base
    - Used in Linux and other Oses
    - Excellent APIs for OS abstraction

- Why port ACPICA subsystem to UEFI shell?
  - UEFI is good fit for rapid testing and prototyping
  - UEFI protocols suitable to provide needed APIs

ACPICA Internals

- Internal Modules of the ACPICA Core Subsystem

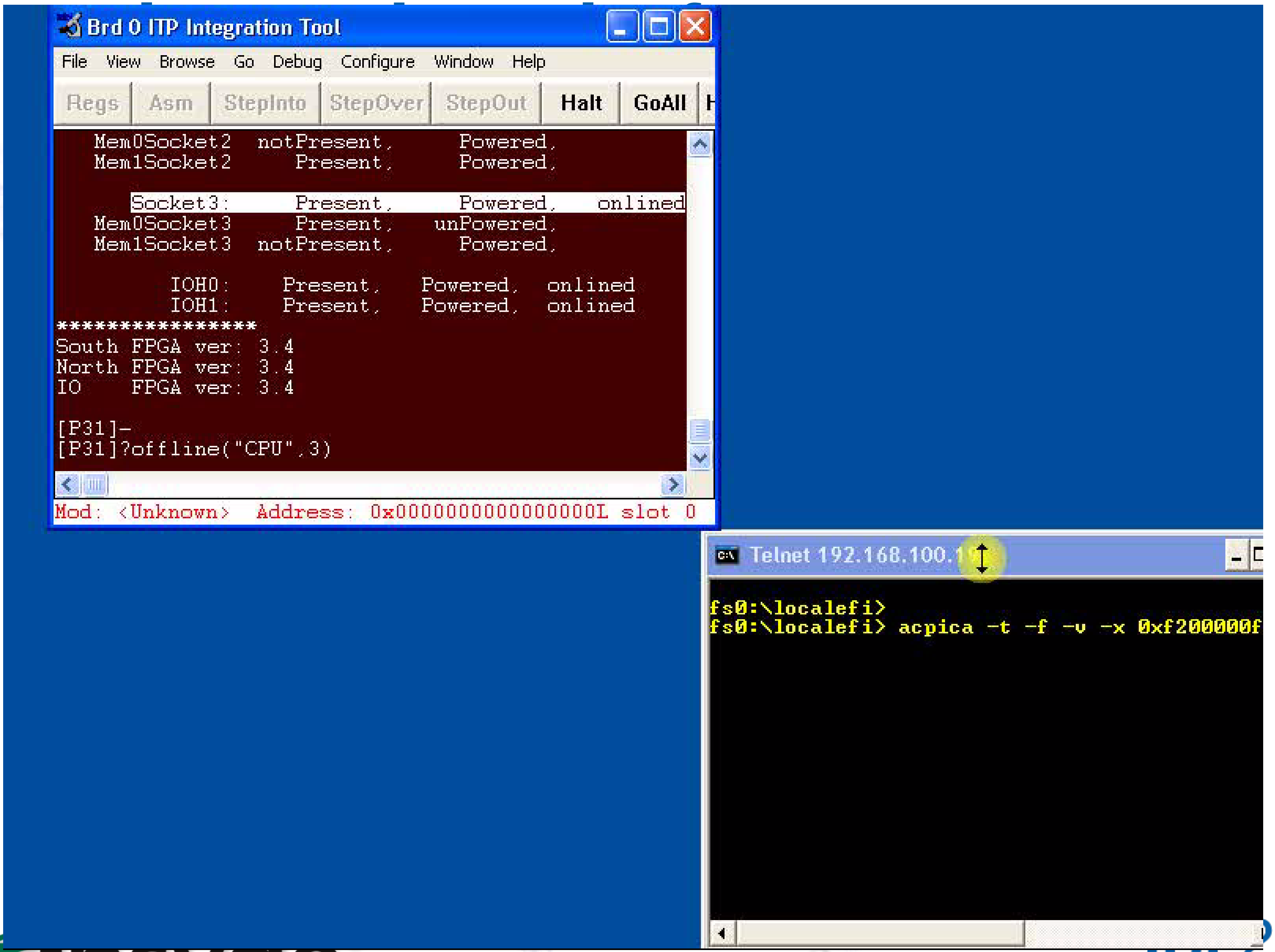


Porting ACPICA

- Use standard C libs
- Use UEFI API to provide hardware access
- Use UEFI periodic timer events to monitor

```
ACPI_STATUS AcpiOsReadPciConfiguration (
ACPI_PCI_ID  *PciId,      UINT32  Register,
void         *Value,      UINT32  Width){

UINT64                Pciex_Address=0;

Pciex_Address = CALC_EFI_PCIEX_ADDRESS (PciId->Bus,
                 PciId->Device,PciId->Function, Register);
switch (Width)    {
case 8:
    Status = gRootBridgeIo->Pci.Read (gRootBridgeIo,
        EfiPciWidthUint8, Pciex_Address, 1, Value);
…
```

Demo ACPICA running on



Brd 0 ITP Integration Tool
File View Browse Go Debug Configure Window Help
Regs Asm StepInto StepOver StepOut Halt GoAll

```
    Mem0Socket2  notPresent,      Powered,
    Mem1Socket2     Present,      Powered,

       Socket3:     Present,      Powered,    onlined
    Mem0Socket3     Present,    unPowered,
    Mem1Socket3  notPresent,      Powered,

          IOH0:     Present,    Powered,    onlined
          IOH1:     Present,    Powered,    onlined
****************
South FPGA ver: 3.4
North FPGA ver: 3.4
IO    FPGA ver: 3.4

[P31]-
[P31]?offline("CPU",3)
```

```
Mod: <Unknown>  Address: 0x0000000000000000L slot 0
```

```
fs0:\localefi>
fs0:\localefi> acpica -t -f -v -x 0xf200000f
```

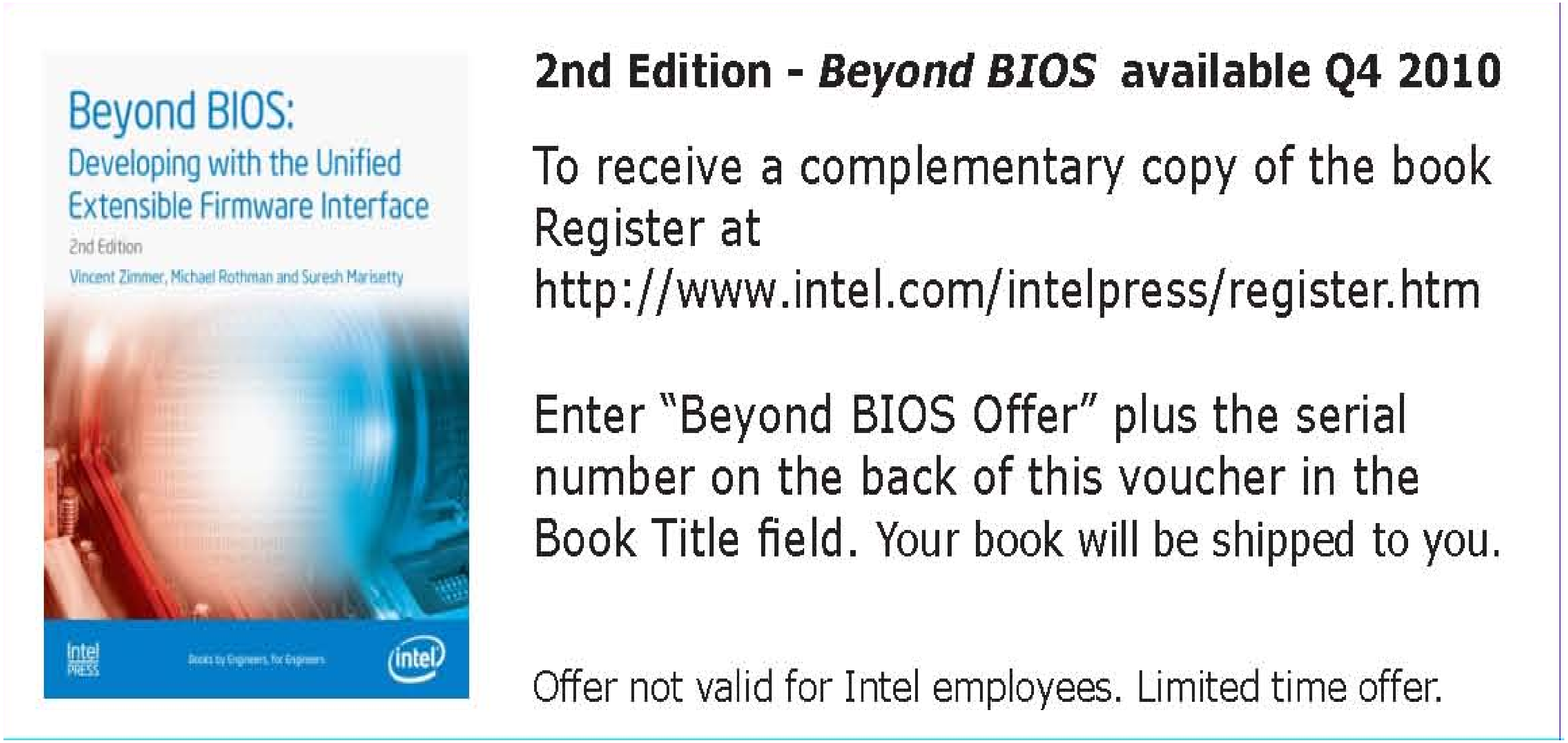
Summary

- Shell 2.0 implementation fully compliant to UEFI Shell Specification now available on tianocore.org
- Configure your shell to meet feature set and image size sweet spot
- Network profile sets applications free in the pre-boot space
- UEFI Application environment is great test harness

Additional sources of information on this topic:

- Other Sessions – Next Slide
- Demos in the showcase – EFI Booth, #160
- More web based info:
  - ***UDK 2010 - http://www.tianocore.Sourceforge.net***
  - ***UEFI Specifications - http://www.uefi.org***
- Book on topic:
  - Beyond BIOS 2nd edition - Intel Press
- Get the UEFI Shell 2.0 specification – www.uefi.org.
- Get the UDK ShellPkg with all the source code from www.tianocore.org

Beyond BIOS 2nd Edition Promotion


Beyond BIOS:
Developing with the Unified Extensible Firmware Interface
2nd Edition
Vincent Zimmer, Michael Rothman and Suresh Marisetty
Intel PRESS
Books by Engineers, for Engineers
intel

2nd Edition - *Beyond BIOS* available Q4 2010

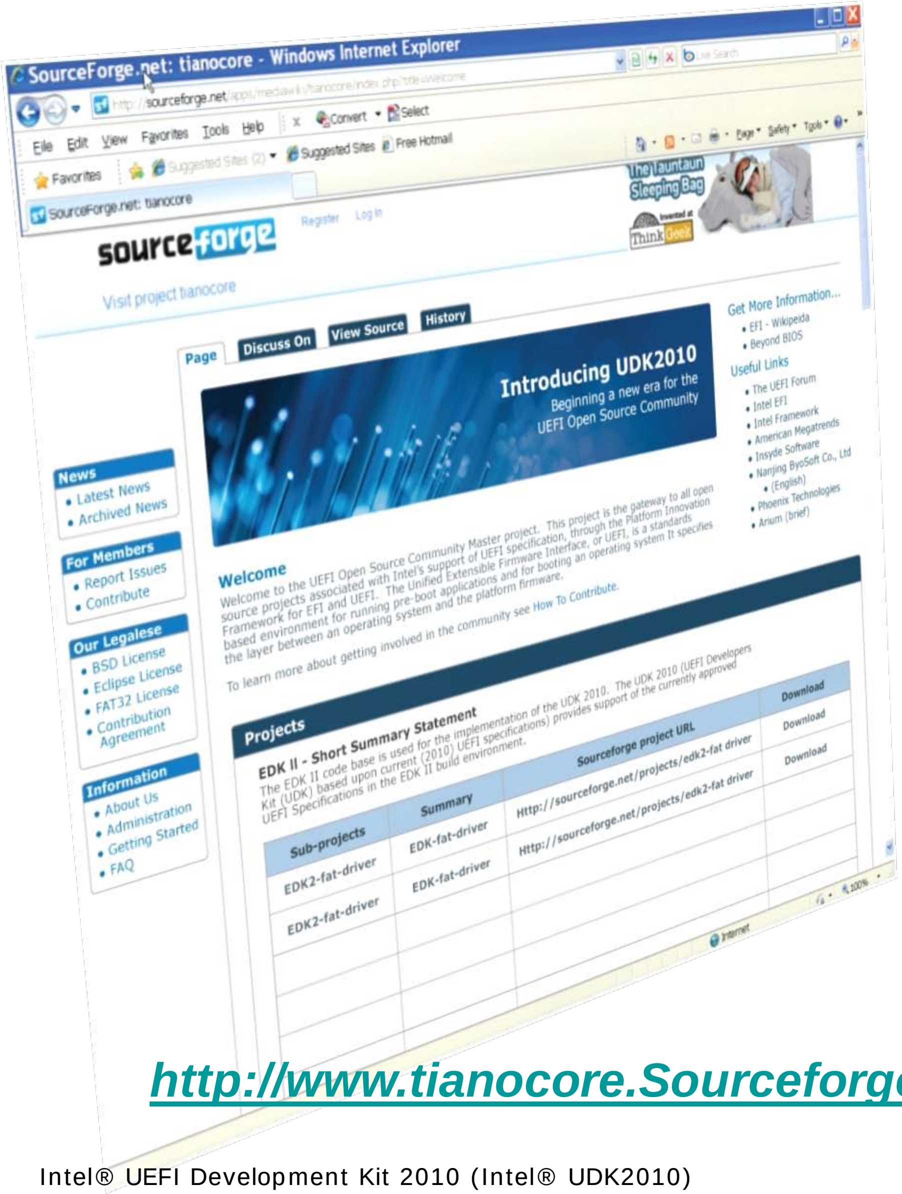
To receive a complementary copy of the book Register at http://www.intel.com/intelpress/register.htm

Enter "Beyond BIOS Offer" plus the serial number on the back of this voucher in the Book Title field. Your book will be shipped to you.

Offer not valid for Intel employees. Limited time offer.

Vouchers available in session room and UEFI Tech showcase booth #160

Intel® UDK2010 Available on tianocore.org


SourceForge.net: tianocore - Windows Internet Explorer
http://sourceforge.net/apps/mediawiki/tianocore/index.php?title=Welcome
File Edit View Favorites Tools Help
Convert Select
Favorites Suggested Sites Free Hotmail
SourceForge.net: tianocore
sourceforge
Register Log In
The Tauntaun Sleeping Bag
ThinkGeek
Visit project tianocore
Page
Discuss On
View Source
History
Internet
100%

News
- Latest News
- Archived News

For Members
- Report Issues
- Contribute

Our Legalese
- BSD License
- Eclipse License
- FAT32 License
- Contribution Agreement

Information
- About Us
- Administration
- Getting Started
- FAQ

Introducing UDK2010
Beginning a new era for the UEFI Open Source Community

Welcome

Welcome to the UEFI Open Source Community Master project. This project is the gateway to all open source projects associated with Intel's support of UEFI specification, through the Platform Innovation Framework for EFI and UEFI. The Unified Extensible Firmware Interface, or UEFI, is a standards based environment for running pre-boot applications and for booting an operating system It specifies the layer between an operating system and the platform firmware.

To learn more about getting involved in the community see How To Contribute.

Projects

EDK II - Short Summary Statement

The EDK II code base is used for the implementation of the UDK 2010. The UDK 2010 (UEFI Developers Kit (UDK) based upon current (2010) UEFI specifications) provides support of the currently approved UEFI Specifications in the EDK II build environment.

Sub-projects	Summary	Sourceforge project URL	Download
EDK2-fat-driver	EDK-fat-driver	Http://sourceforge.net/projects/edk2-fat driver	Download
EDK2-fat-driver	EDK-fat-driver	Http://sourceforge.net/projects/edk2-fat driver	Download

Get More Information...
- EFI - Wikipeida
- Beyond BIOS

Useful Links
- The UEFI Forum
- Intel EFI
- Intel Framework
- American Megatrends
- Insyde Software
- Nanjing ByoSoft Co., Ltd
  - (English)
- Phoenix Technologies
- Arium (brief)

tianocore.org

Intel® UDK2010
Open Source
UEFI Development Kit

Develop. Contribute. Advance.

http://www.tianocore.Sourceforge.net

UEFI PLUGFEST in Taiwan Oct 12-15, 2010

Visit www.uefi.org/events for Event Info and Registration

IDF 2010 UEFI Fall Sessions
Sept. 13, 2010 Moscone Room 2006

	EFI#	Company	Description	Time
✓	S001	Intel, IBM, HP	Introducing the New Intel® UEFI Development Kit: Industry Foundation for Platform Innovation	11:00 AM
✓	S002	Intel, LSI, Dell, Phoenix	UEFI Advancements for Independent Hardware Vendors	1:05 PM
✓	S003	Intel, WindRiver	Boot Loader Solutions for Intel® Atom™ Processor Based Embedded Devices	2:10 PM
✓	S004	Intel, Dell, AMI	Zero-Touch Platform Manageability with UEFI	3:15 PM
✓	S005	Intel, IBM, Insyde	Beyond DOS: The UEFI Shell – a Modern Pre-boot Application Environment	4:20 PM
	Q001	All	UEFI Q & A session with all Speakers	5:25 PM

✓ DONE

Session Presentations - PDFs

The PDF for this Session presentation is available from our IDF Content Catalog at the end of the day at:

intel.com/go/idfsessions

URL is on top of Session Agenda Pages in Pocket Guide

Please Fill out the Session Evaluation Form

Give the completed form to the room monitors as you exit!

Thank You for your input, we use it to improve future Intel Developer Forum events

Q&A

Tweet your questions and comments to @intel_uefi

Legal Disclaimer

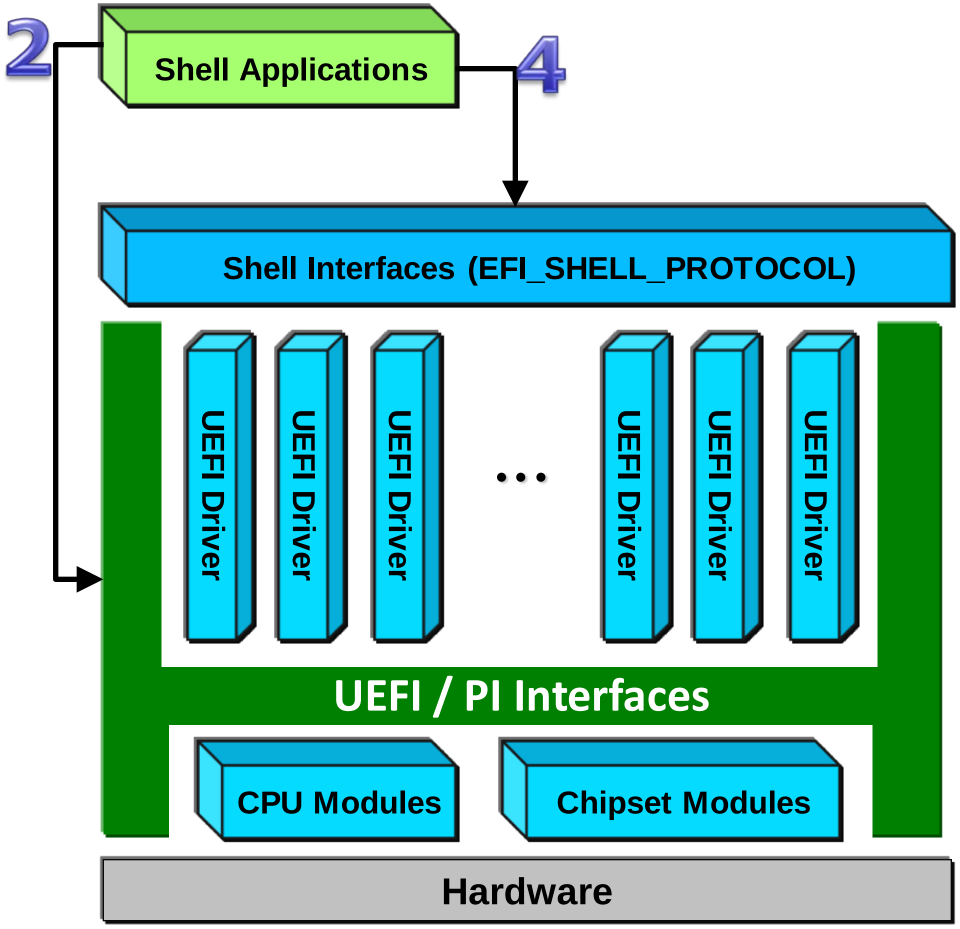
- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPETY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.
- Intel may make changes to specifications and product descriptions at any time, without notice.
- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
- Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.
- Intel, Intel Sponsors of Tomorrow. and Intel Sponsors of Tomorrow. Logo and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
-

Risk Factors

The above statements and any others in this document that refer to plans and expectations for the second quarter, the year and the future are forward-looking statements that involve a number of risks and uncertainties. Many factors could affect Intel’s actual results, and variances from Intel’s current expectations regarding such factors could cause actual results to differ materially from those expressed in these forward-looking statements. Intel presently considers the following to be the important factors that could cause actual results to differ materially from the corporation’s expectations. Demand could be different from Intel's expectations due to factors including changes in business and economic conditions; customer acceptance of Intel’s and competitors’ products; changes in customer order patterns including order cancellations; and changes in the level of inventory at customers. Intel operates in intensely competitive industries that are characterized by a high percentage of costs that are fixed or difficult to reduce in the short term and product demand that is highly variable and difficult to forecast. Additionally, Intel is in the process of transitioning to its next generation of products on 32nm process technology, and there could be execution issues associated with these changes, including product defects and errata along with lower than anticipated manufacturing yields. Revenue and the gross margin percentage are affected by the timing of new Intel product introductions and the demand for and market acceptance of Intel's products; actions taken by Intel's competitors, including product offerings and introductions, marketing programs and pricing pressures and Intel’s response to such actions; defects or disruptions in the supply of materials or resources; and Intel’s ability to respond quickly to technological developments and to incorporate new features into its products. The gross margin percentage could vary significantly from expectations based on changes in revenue levels; product mix and pricing; start-up costs, including costs associated with the new 32nm process technology; variations in inventory valuation, including variations related to the timing of qualifying products for sale; excess or obsolete inventory; manufacturing yields; changes in unit costs; impairments of long-lived assets, including manufacturing, assembly/test and intangible assets; the timing and execution of the manufacturing ramp and associated costs; and capacity utilization. Expenses, particularly certain marketing and compensation expenses, as well as restructuring and asset impairment charges, vary depending on the level of demand for Intel's products and the level of revenue and profits. The majority of our non-marketable equity investment portfolio balance is concentrated in the flash memory market segment, and declines in this market segment or changes in management’s plans with respect to our investment in this market segment could result in significant impairment charges, impacting restructuring charges as well as gains/losses on equity investments and interest and other. Intel's results could be impacted by adverse economic, social, political and physical/infrastructure conditions in countries where Intel, its customers or its suppliers operate, including military conflict and other security risks, natural disasters, infrastructure disruptions, health concerns and fluctuations in currency exchange rates. Intel’s results could be affected by the timing of closing of acquisitions and divestitures. Intel's results could be affected by adverse effects associated with product defects and errata (deviations from published specifications), and by litigation or regulatory matters involving intellectual property, stockholder, consumer, antitrust and other issues, such as the litigation and regulatory matters described in Intel's SEC reports. An unfavorable ruling could include monetary damages or an injunction prohibiting us from manufacturing or selling one or more products, precluding particular business practices, impacting our ability to design our products, or requiring other remedies such as compulsory licensing of intellectual property. A detailed discussion of these and other factors that could affect Intel’s results is included in Intel’s SEC filings, including the report on Form 10-Q for the quarter ended March 27, 2010.

Rev. 5/7/10

Backup Slides



```
  EFI_STATUS
  EFIAPI
  InitializeApplication (
1  IN  EFI_HANDLE                ImageHandle,
   IN  EFI_SYSTEM_HANDLE         *SystemTable
   )
  {
3  EFI_SHELL_APP_INIT (ImageHandle, SystemTable);

   //Program Logic follows
   …
  }
```

1) This item illustrates what the standard entry point for any UEFI compatible binary application or driver looks like. This is the fundamental starting point for all UEFI compatible programs which exposes the underlying UEFI firmware services.
2) During the initialization of a UEFI program, the standard entry point would be used to access the standard runtime and boot services that the UEFI compatible firmware provides.
3) In most shell-aware applications, there would be either a library or macro which would be used to provide access to the underlying shell protocol interfaces. This library/macro isn’t required by the UEFI shell specification, but would commonly be found in many of the available shell-aware programs.
4) In shell-aware applications, the availability of the functions defined in the EFI_SHELL_PROTOCOL can be leveraged.