The University of Reading

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Virtualized Environments for the Harness High Performance Computing Workbench

Björn Könning[1,2], Christian Engelmann[1,2],
Stephen L. Scott[1], and Al Geist[1]

[1] Oak Ridge National Laboratory, Oak Ridge, USA

[2] The University of Reading, Reading, UK

Motivation

- Increasing diversity in HPC platforms between and within centers
- Frequent hard- and software upgrades (more than once a year)

➔ Constant need for porting, recompiling, and retuning existing or newly developed applications to new or changing environments:

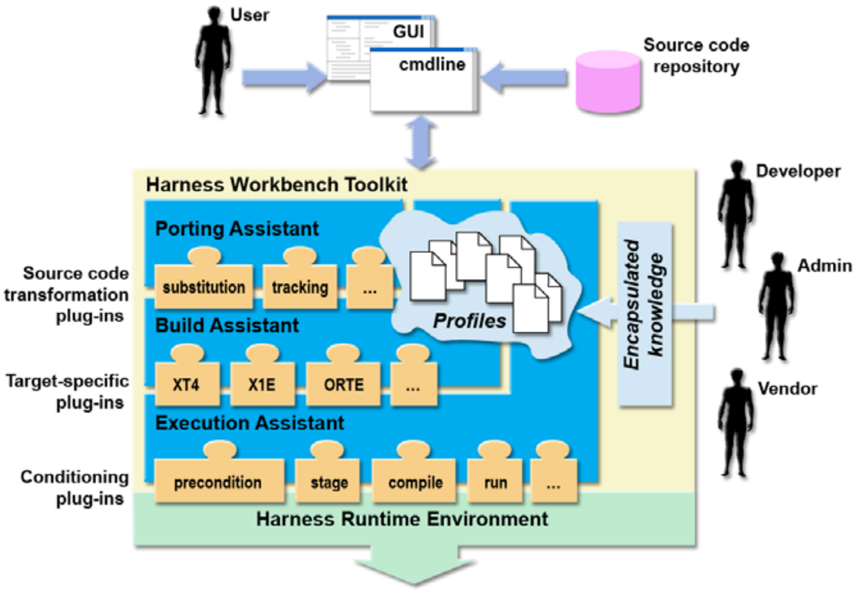
  - Where to deploy scientific applications (sources and binaries)?
  - Which compiler/linker and compiler/linker flags to use?
  - Does the system perform cross-compilation?
  - Which system libraries to link and where to find them?
  - How to find and use dependent software packages?
  - Which system-specific workarounds to use?
  - What needs to be in the batch job script?

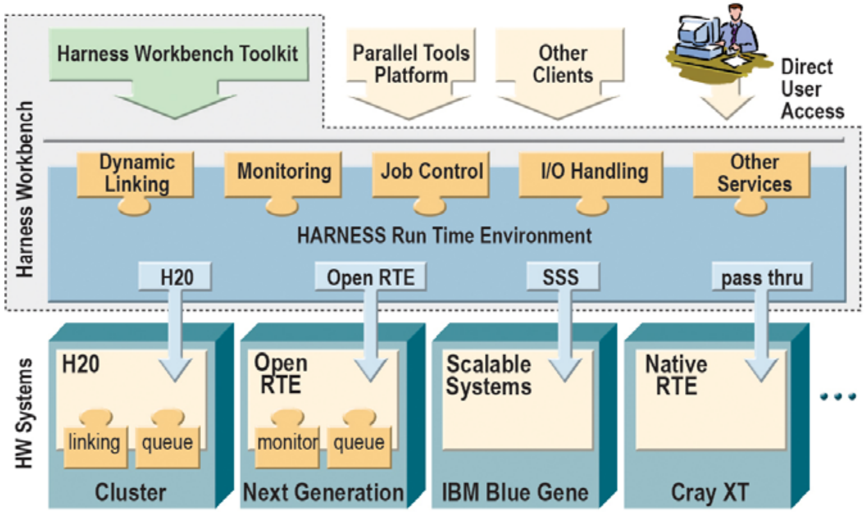
Objectives

- Simplify software development and deployment by making entire software environments portable
- Design a concept for virtualized software environments for scientific HPC applications
- Develop a tool for creating virtualized environments on different HPC platforms
- Develop a tool for starting applications in virtualized environments on different HPC platforms

Harness HPC Workbench

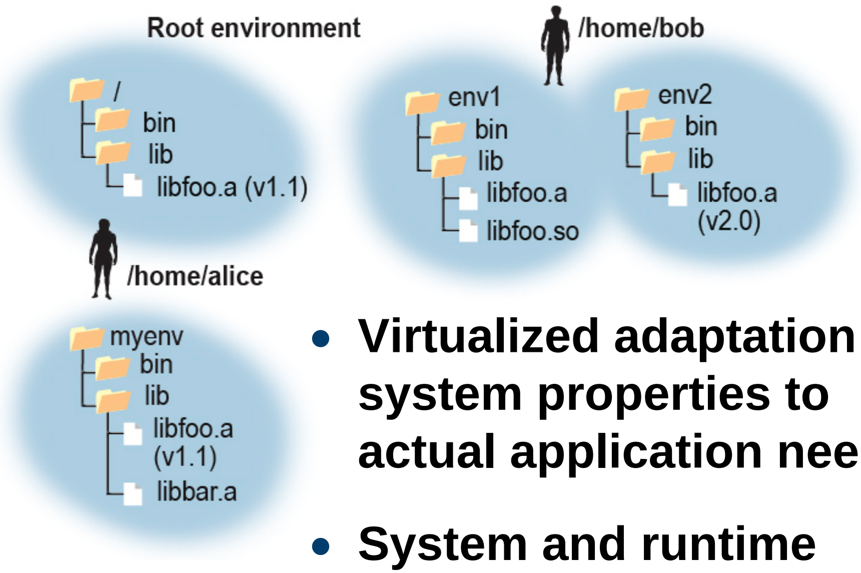
- Harness workbench toolkit
  - Unified development, deployment, and execution
  - Common view across diverse HPC platforms
  - User-space installation and virtual environments
- Next-generation runtime environment
  - Flexible, adaptive, lightweight framework
  - Management of runtime tasks
  - Support for diverse HPC platforms





Virtualized Environments

- Application dependencies may cause conflicts with system-wide installed libraries.
- Use co-existing, alternative user-space installations.
- Provide isolated installation environments (“sandboxes”).
- These can inherit from one another to build nested hierarchies.

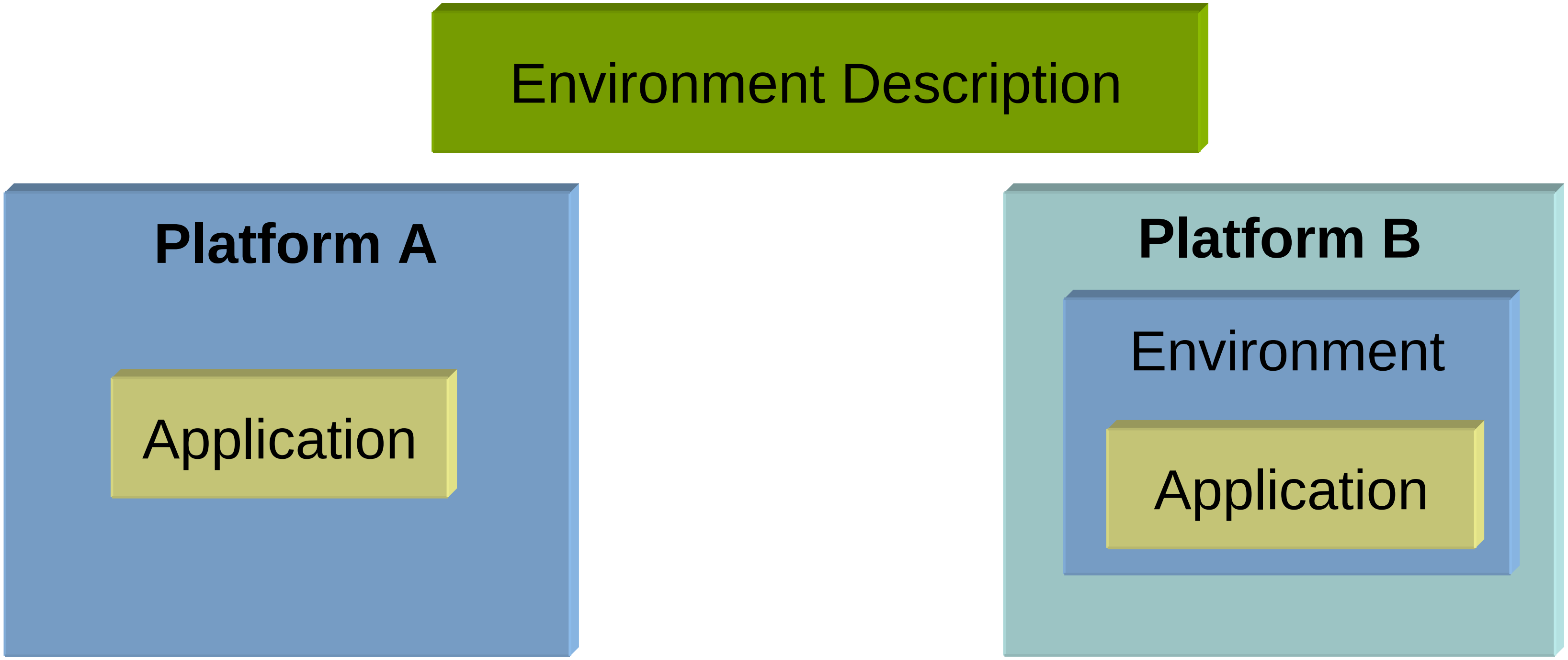


- Virtualized adaptation of system properties to actual application needs
- System and runtime environment virtualization

Virtualized Environment Workflow

1. Develop application for local platform

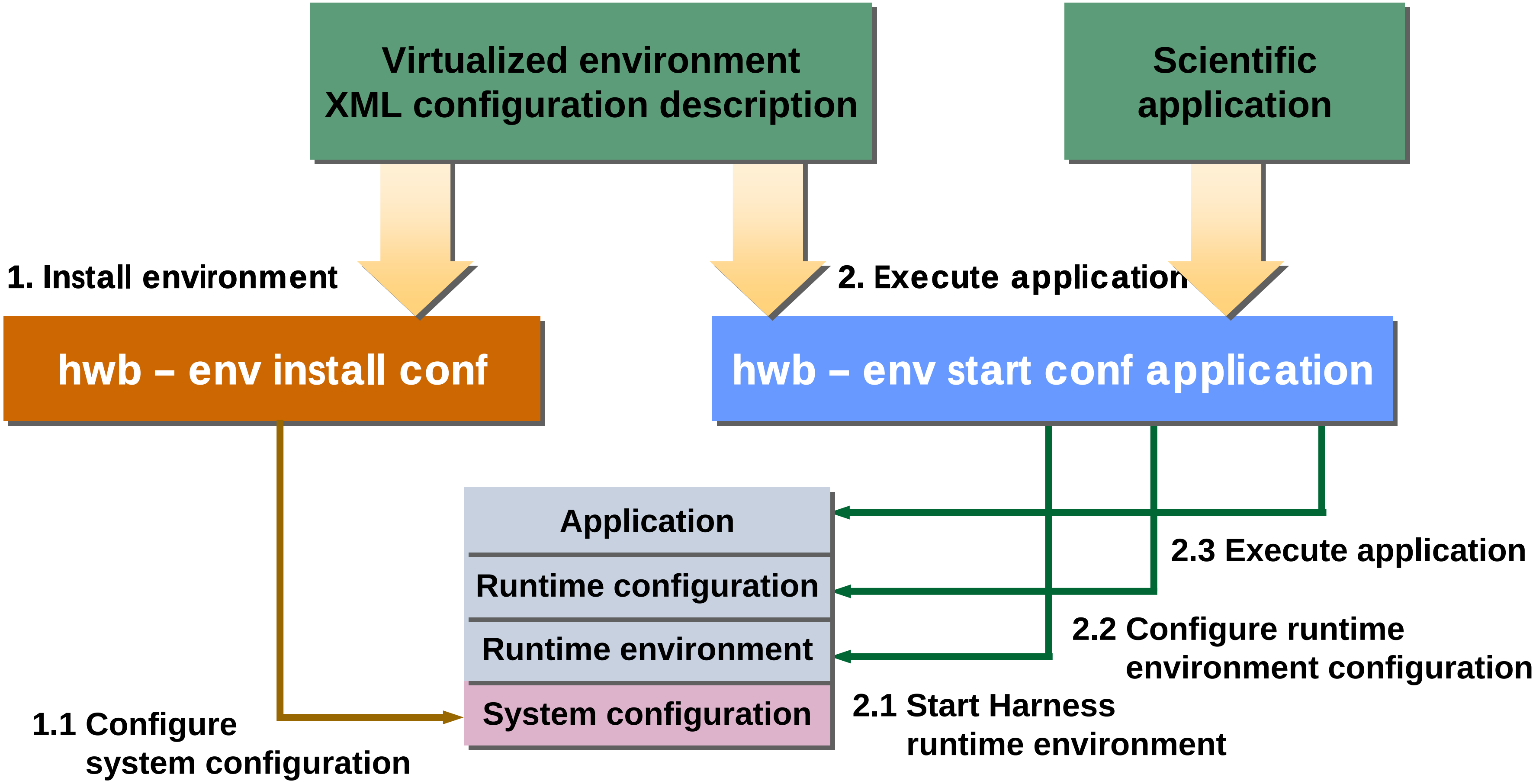
[illegible] application”



Approach

- Initial focus on:
  - Well-known and widely-available **chroot** mechanism
  - File system and shell environment variables only
  - Fine-grain configuration mechanisms, e.g., files, directories
  - Working prototype at the runtime environment level
- Future focus on:
  - Configuration of system services and access to external resources (quality of service, security, and isolation)
  - Coarse-grain configuration mechanisms, e.g., software packages or OS distributions
  - Advanced virtualization technologies, like Xen

Design and Detailed Workflow



Unix Shell Virtualization Configuration

```
<var>
<name>PATH</name>
<value>/home/user/apps</value>
</var>

<var>
<name>PATH</name>
<value>/home/user/apps</value>
<action>modify</action>
<insertPosition>append</insertPosition>
</var>
```

- Fine-grain configuration for shell variables
- Creation of new shell variables
- Modification of existing shell variables
- Detailed XML schema available

File System Virtualization Configuration

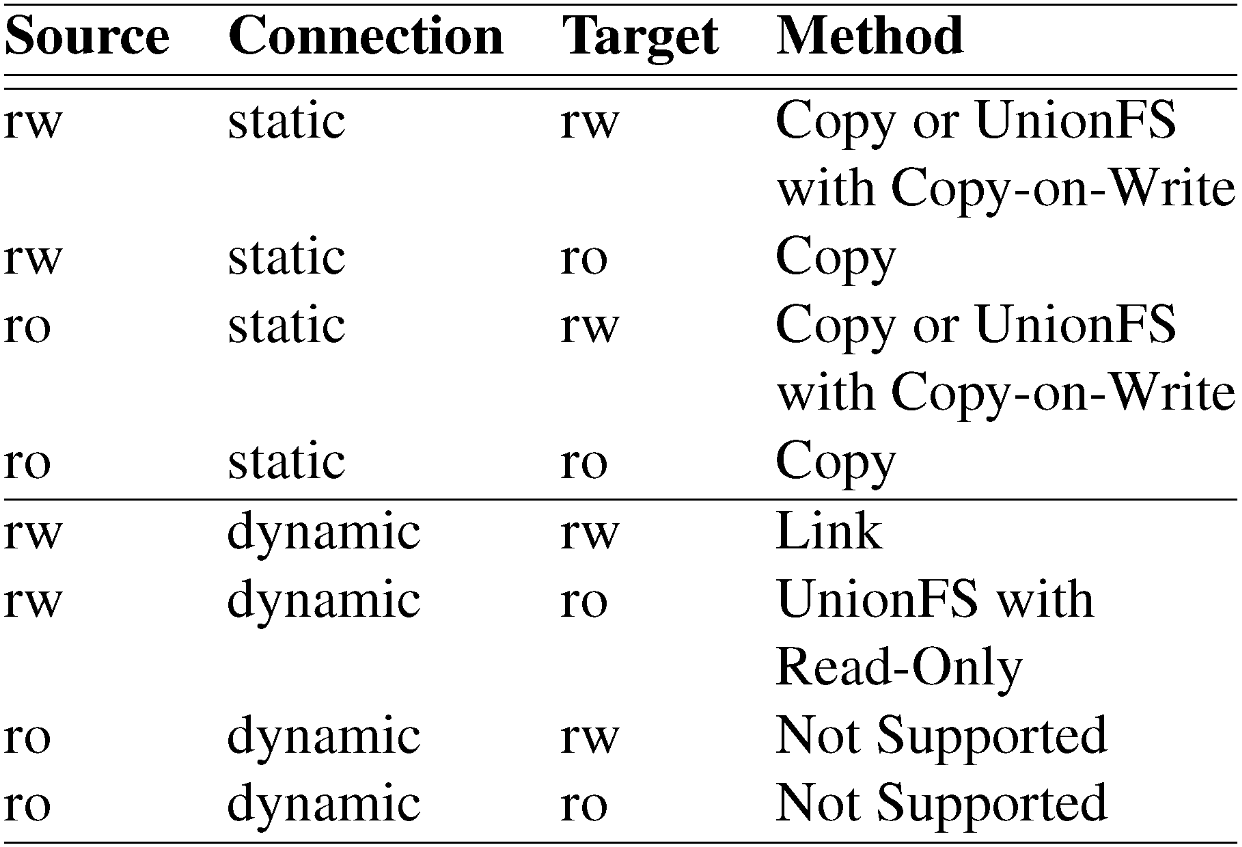
```
<directory>
  <name>lib</name>
  <permission>755</permission>
  <umask>755</umask>
  <integration>copy</integration>
  <file>
    <source>lib/test.conf</source>
  </file>
  <subdir>
    <name>app1/source</name>
    <file>
      <source>lib/test2.conf</source>
      <name>newName.conf</name>
      <integration>copy</integration>
    </file>
    <subdir>
      <name>version</name>
    </subdir>
  </subdir>
</directory>
```

- Fine-grain configuration for files and directories
- Source-destination relationships
- 3 different integration methods (next slide)
- Allows for changing:
  - ❑ Names
  - ❑ Permissions
- Detailed XML schema available

FS Virtualization Configuration Methods

- Copy method
  - Slow virtual environment creation, but fast at run time
  - No connection to original: permissions and content can be changed and are lost after virtual environment destruction
- Link method
  - Fast virtual environment creation, and fast at run time
  - Connection to original: permissions cannot and content can be changed, and is not lost after virtual environment destruction
- UnionFS method
  - Fast virtual environment creation, and fast at run time
  - Configurable connection to the original: copy-on-write, hide-on-delete, and limitation of access rights

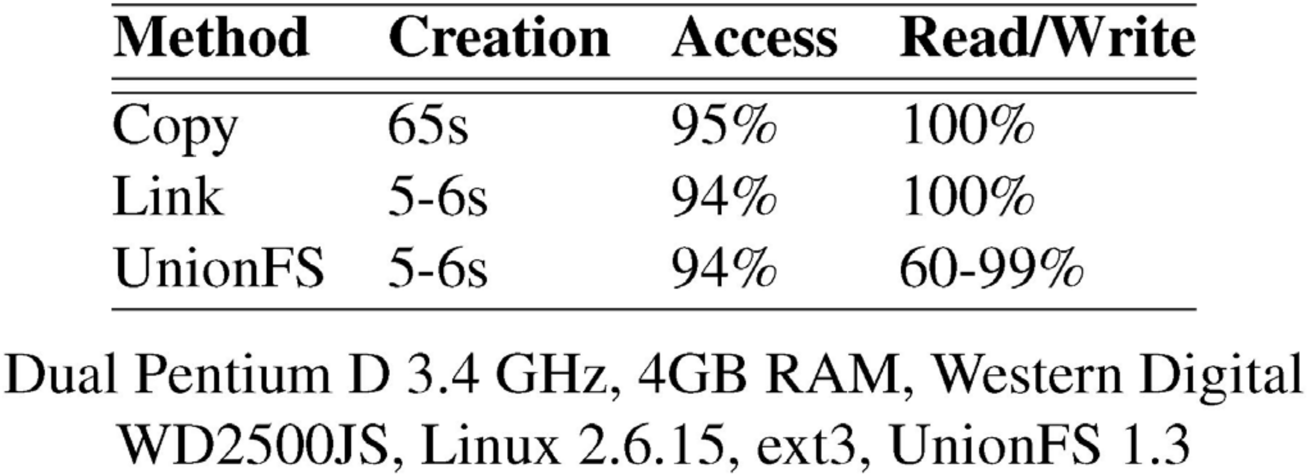
Configuration Method Comparison



Source	Connection	Target	Method
rw	static	rw	Copy or UnionFS with Copy-on-Write
rw	static	ro	Copy
ro	static	rw	Copy or UnionFS with Copy-on-Write
ro	static	ro	Copy
rw	dynamic	rw	Link
rw	dynamic	ro	UnionFS with Read-Only
ro	dynamic	rw	Not Supported
ro	dynamic	ro	Not Supported

Configuration Method Experiments

- Virtualized environment creation test:
  - 32935 files of /bin, /lib, /sbin and /etc from Fedora Core 6
- Virtualized environment access and read/write tests:
  - fopen, Iozone, Postmark, and kernel compilation



Method	Creation	Access	Read/Write
Copy	65s	95%	100%
Link	5-6s	94%	100%
UnionFS	5-6s	94%	60-99%

Dual Pentium D 3.4 GHz, 4GB RAM, Western Digital
WD2500JS, Linux 2.6.15, ext3, UnionFS 1.3

Other Features

- Multiple inheritance
  - Virtualized environment configurations may inherit others
  - Configuration based on inheritance processing order
  - Allows for configurations offered by system administrators to be inherited and modified by users
- Virtual users
  - Sandbox characteristic via virtual users that are added to the system after **chroot**
- XML schema independent from virtualization approach – possible reuse for Xen-like virtualization

Accomplishments and Limitations

- ⇧ Extensible hierarchical virtualized environment description scheme in XML
- ⇧ Utilization of various methods for file system modifications: link, copy, and UnionFS
- ⇩ Runtime environment solution that covers file system and shell environment variables (if any) only
- ⇩ Developed tools limited to the **`chroot`** mechanism with certain system security implications

Future Work

- Abstract XML descriptions of requirements:
  - Application needs
  - System properties
- Focus on other virtualization technologies
  - Xen-like system-level virtualization
  - Pure runtime virtualization, e.g., overriding system calls
- Integration with scalable runtime environments
  - Next-generation Open MPI runtime environment
- Increase collaboration and coordination with other HPC virtualization R&D efforts

Virtualized Environments for the Harness High Performance Computing Workbench

Björn Könning[1,2], Christian Engelmann[1,2],
Stephen L. Scott[1], and Al Geist[1]

[1] Oak Ridge National Laboratory, Oak Ridge, USA
[2] The University of Reading, Reading, UK