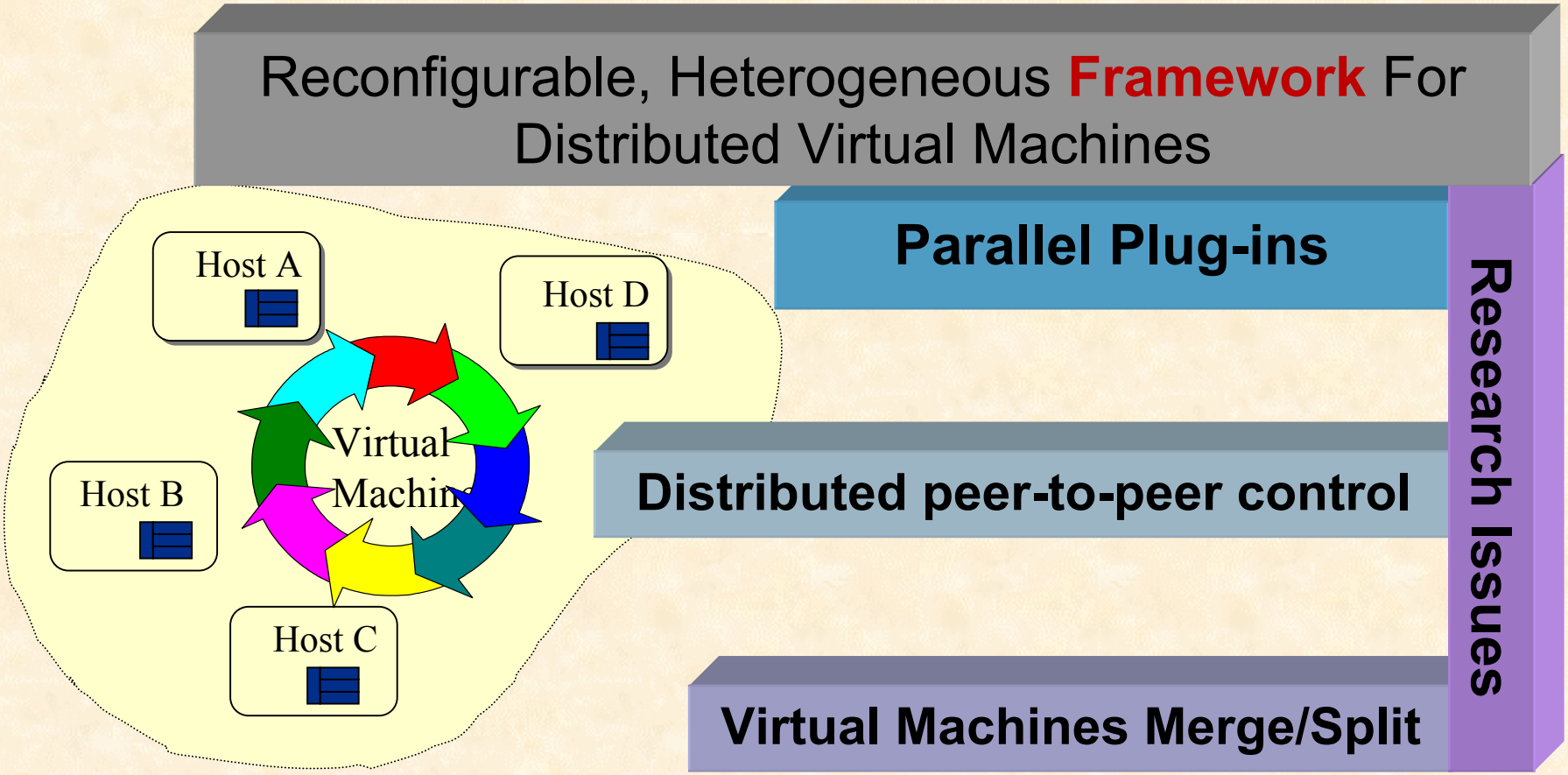
An Architecture for a Multi-threaded Harness Kernel

Wael R. Elwasif
Computer Science & Mathematics Division
Oak Ridge National Laboratory.

elwasifwr@ornl.gov

What is Harness?



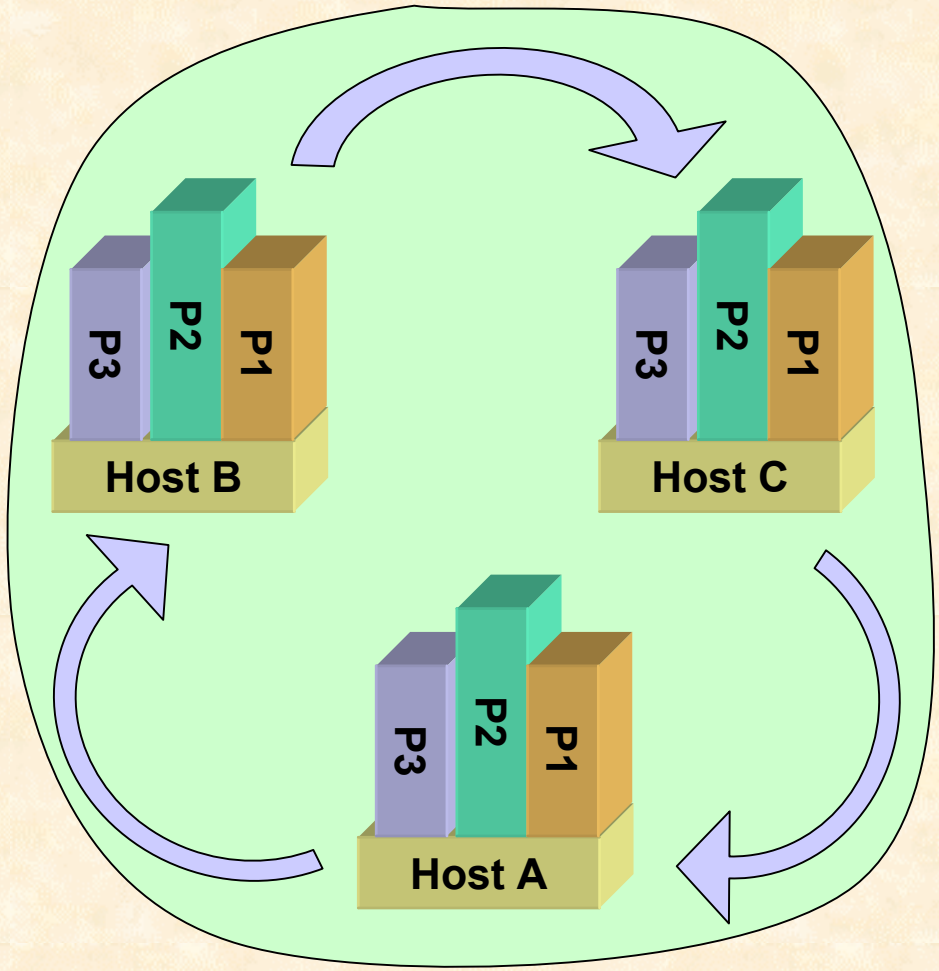
The Harness Virtual Machine

Light weight Kernel

Extensible through plug-ins

Assembled into virtual machine

Using distributed peer-to-peer control



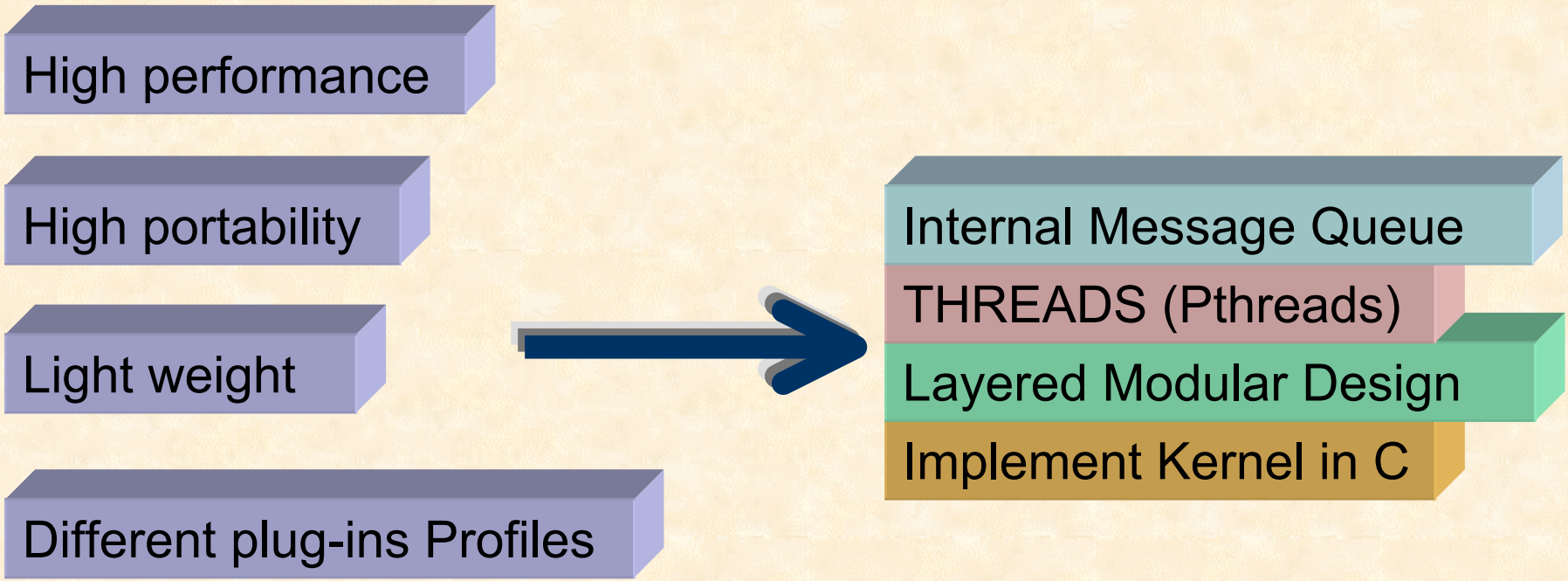
The Harness Project

Java Implementation, PVM Plug-in

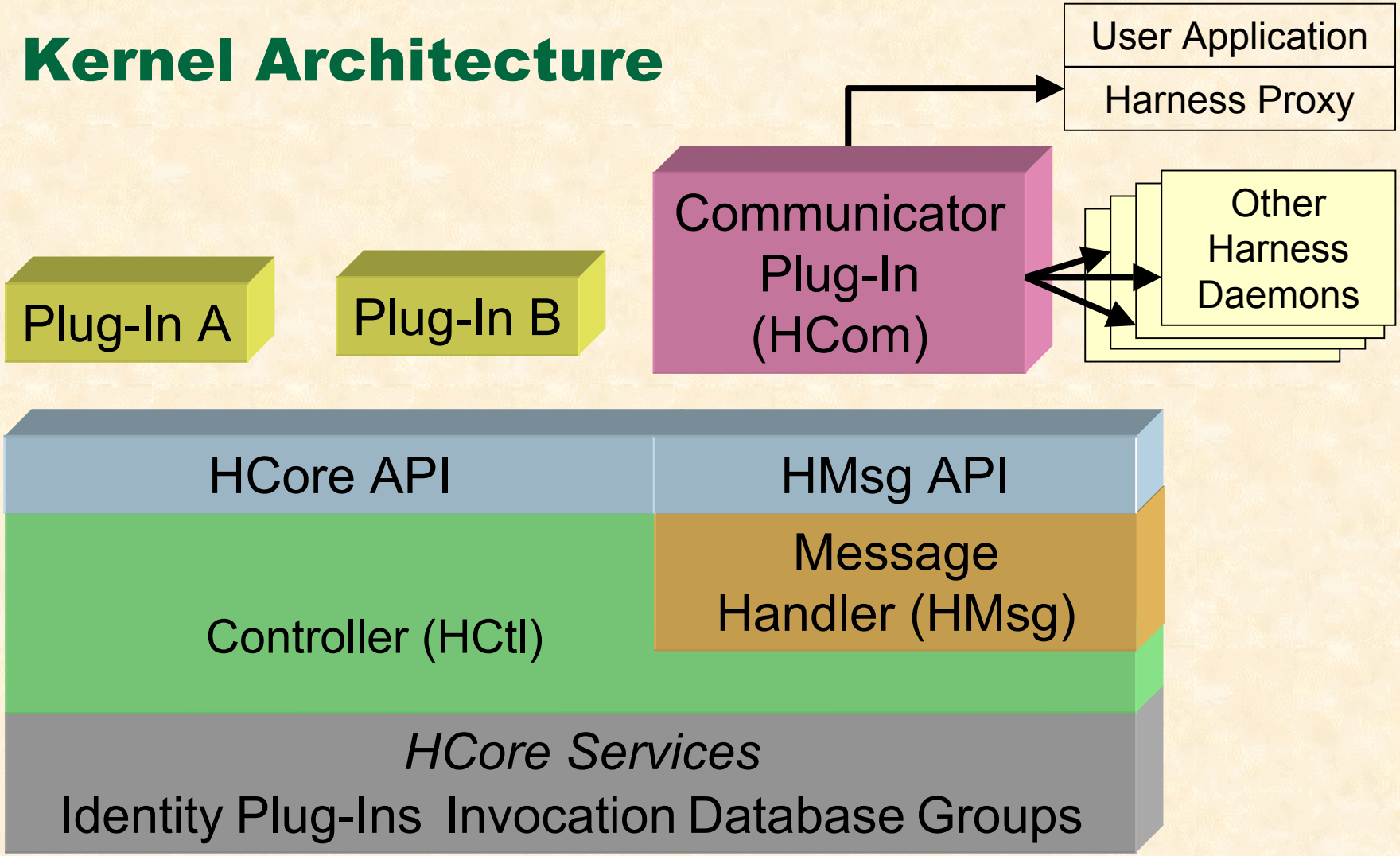
Fault Tolerant MPI Plug-in

Distributed Control, High Performance Kernel

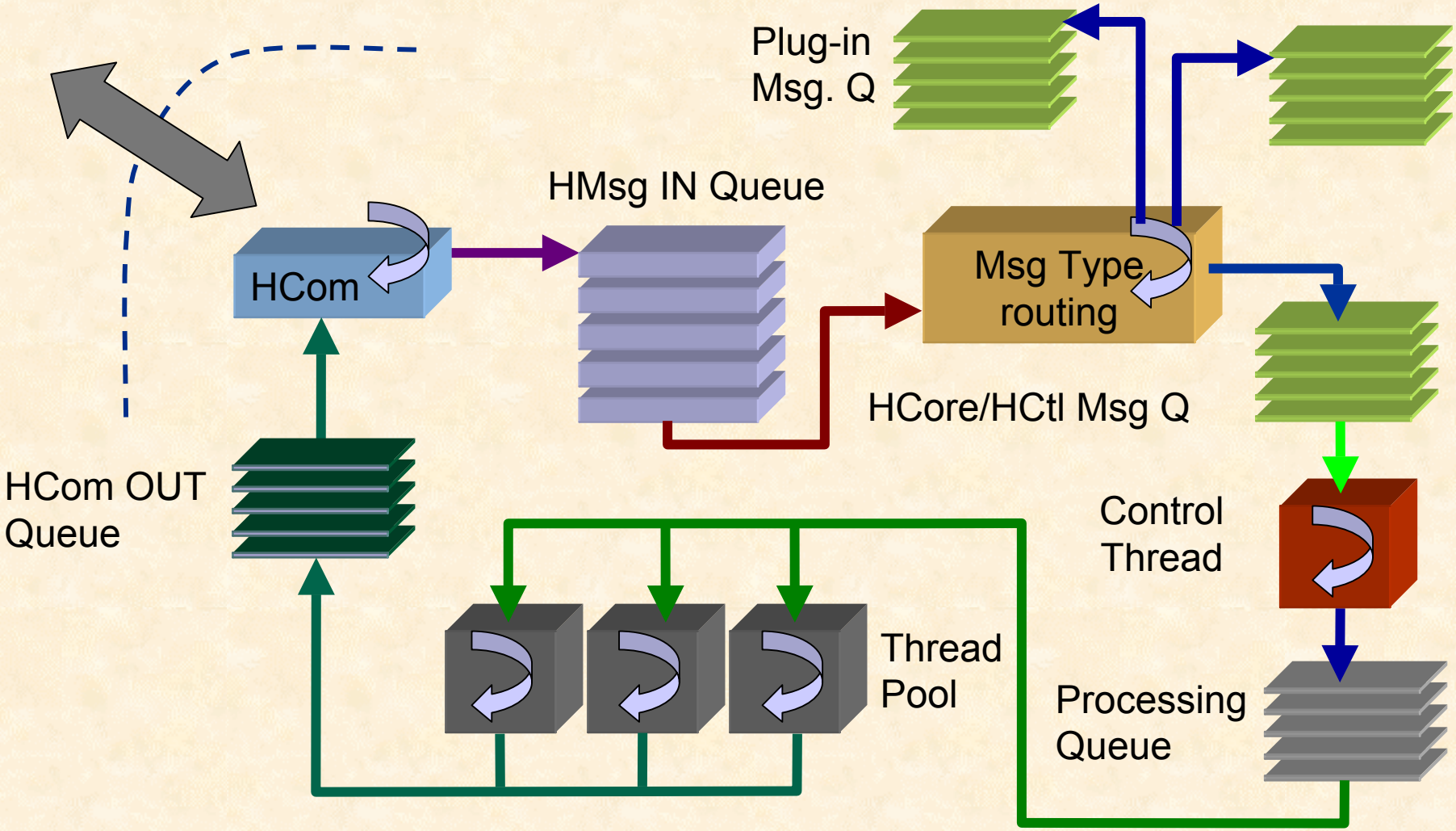
The ORNL Harness Kernel : Requirements



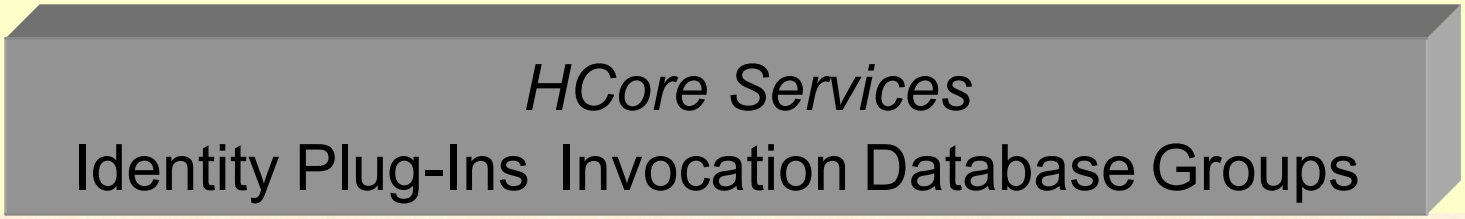
Kernel Architecture



Data Flow in Harness Kernel



Kernel Architecture



HCore Services

- **Identification**: Opaque HID – Locally generated and globally unique.
- **Plug-ins:**
  - **Lifetime management**: load, unload plug-ins.
  - **Interface definition**: register plugin functions.
  - **Functions access**: call function, access function.
- **Termination**: h_exit()

Plug-ins Functions Interface

- **What to do about the API:**
  - Use an IDL – flexible BUT complicated.
  - Use container object – simple and fits the message passing paradigm.
- **The Harness argument, h_arg.**
  - Encapsulates argc, argv, ***argt***, ***args***.
  - Standard plugin function signature:

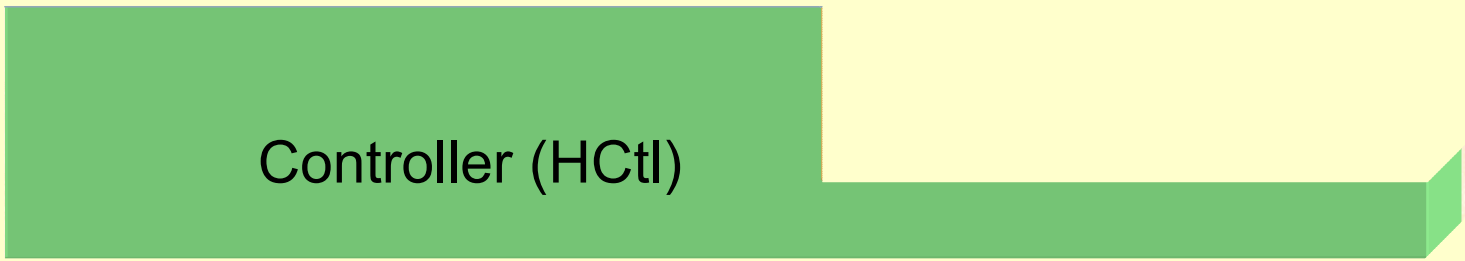
    ```
    h_arg foo(h_arg in).
    ```

  - Utility functions:
    `h_setArg(), h_getArg(), ..etc.`

The Database Service: HBase

- State storage managed by the kernel.
- Table/Record/Key based storage model.
- Tables are:
  - Local/Global/Group – ***Where***?
  - Private/Public – ***Who***?
- Unique/Multiple keys.
- Records are ***h_arg***'s.
- Predefined + user defined query functions.
- Atomic update.

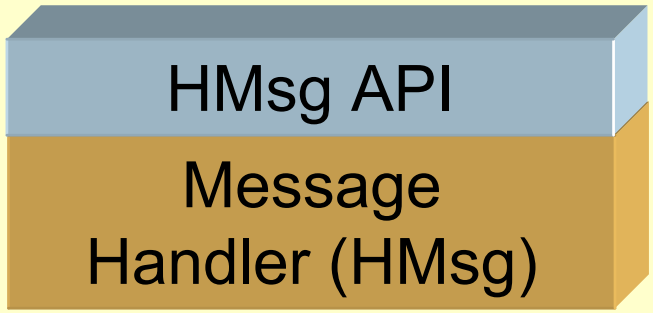
Kernel Architecture



The Controller: HCtl

- Arbitrate access to HCore services.
- Can implement different paradigms:
  - Local-only control.
  - Master daemon.
  - ***Peer-to-peer distributed control.***
- Groups:
  - GLOBAL/CONTOL/ user-defined.
  - Determine context of Harness commands.
  - Different from PVM groups.

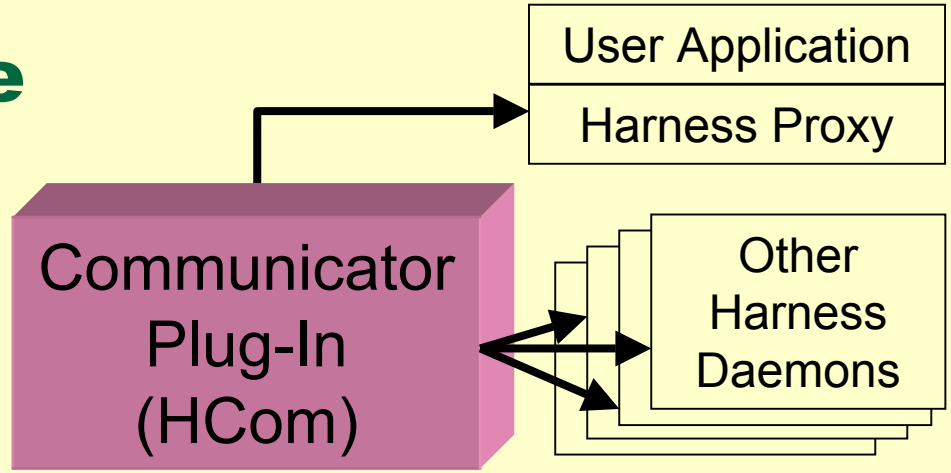
Kernel Architecture



The Message Service: HMsg

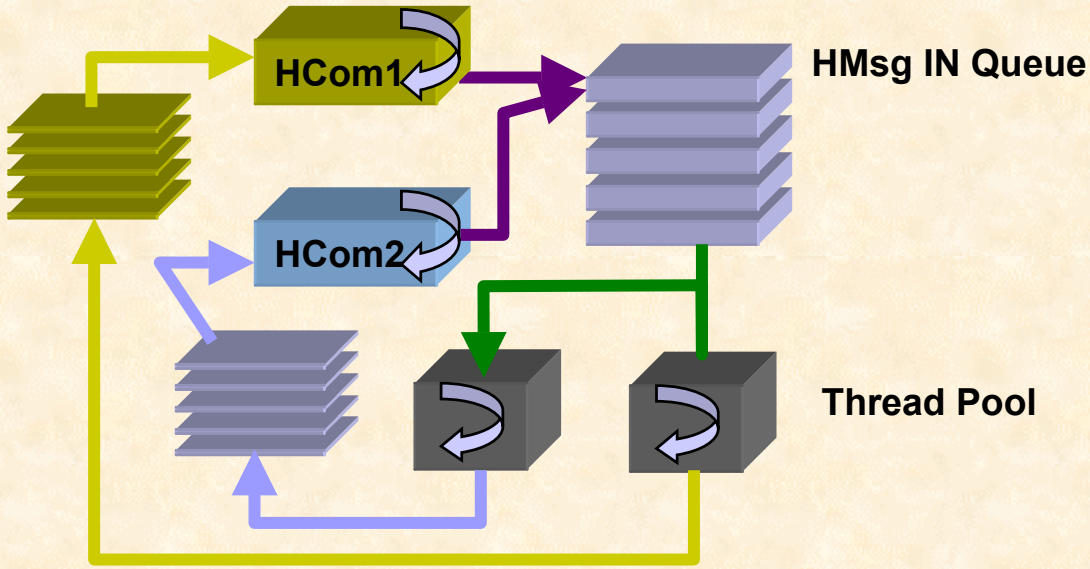
- Not an alternative to high performance messaging.
- Not a plug-in.
- Message encoding/decoding.
- Incoming message routing:
  - Plug-in defined message types and queues.
  - HCtl/HCore message queue.
- Message construction and dispatching.
- ***Independent of transfer protocol.***

Kernel Architecture



The Communicator: HCom

- A ***Special*** plug-in.
- Transfer protocol implementation.
- Design supports multiple HCom's.
- One out-queue per HCom, one in-queue per kernel.



Kernel Architecture

Harness Plug-ins

- Dynamically loaded/unloaded.
- Separate name space
  `plugin1::foo()` and `plugin2::foo()` can coexist.
- Thread safe functions.
- Interface exported in `init_plugin()`.
- Kernel mediated and ***direct*** inter-plug-in calls.
- Can have out-of-band communication.

Implementation Status

- Internal release with local control: December 2001.
- Integration with distributed peer-to-peer control currently in progress.
- Expected release with FT-MPI plug-in: October 2002.

And Thanks TO

Co-authors
David E. Bernholdt, James A Kohl, and Al Geist

Oak Ridge National Laboratory.

For more information:

http://www.csm.ornl.gov/harness
http://icl.cs.utk.edu
http://www.mathcs.emory.edu/harness/

Research supported by
the Mathematics, Information and Computational Sciences Office, US Department of Energy.

Questions?