Hippodrome: running circles around storage administration

Eric Anderson, Michael Hobbs, Kimberly Keeton, Susan Spence, Mustafa Uysal and Alistair Veitch

Hewlett-Packard Labs

Storage Systems Department

http://www.hpl.hp.com/SSP/

Hewlett-Packard Laboratories

invent

Challenges of data center storage

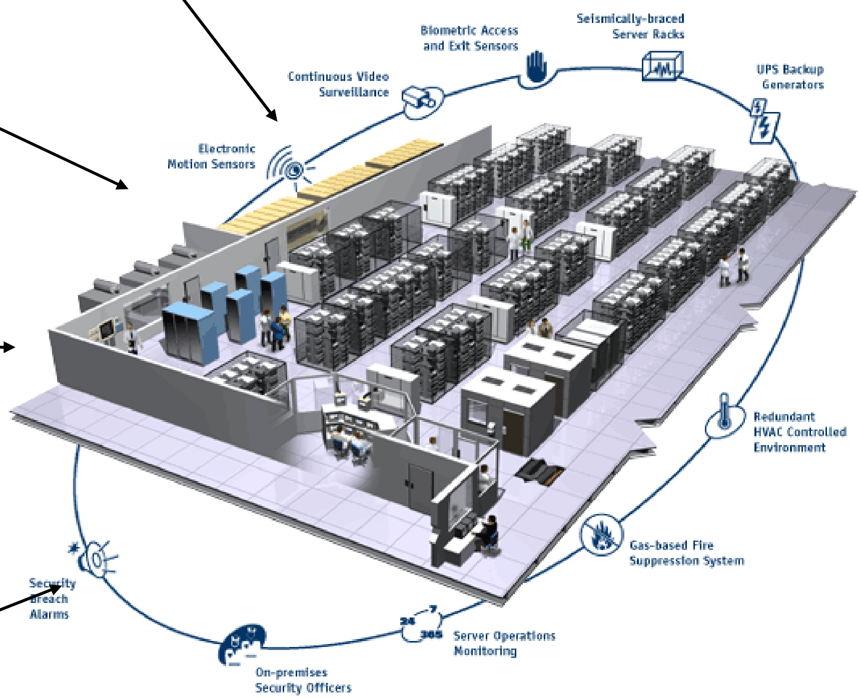
high cost of administration

high availability

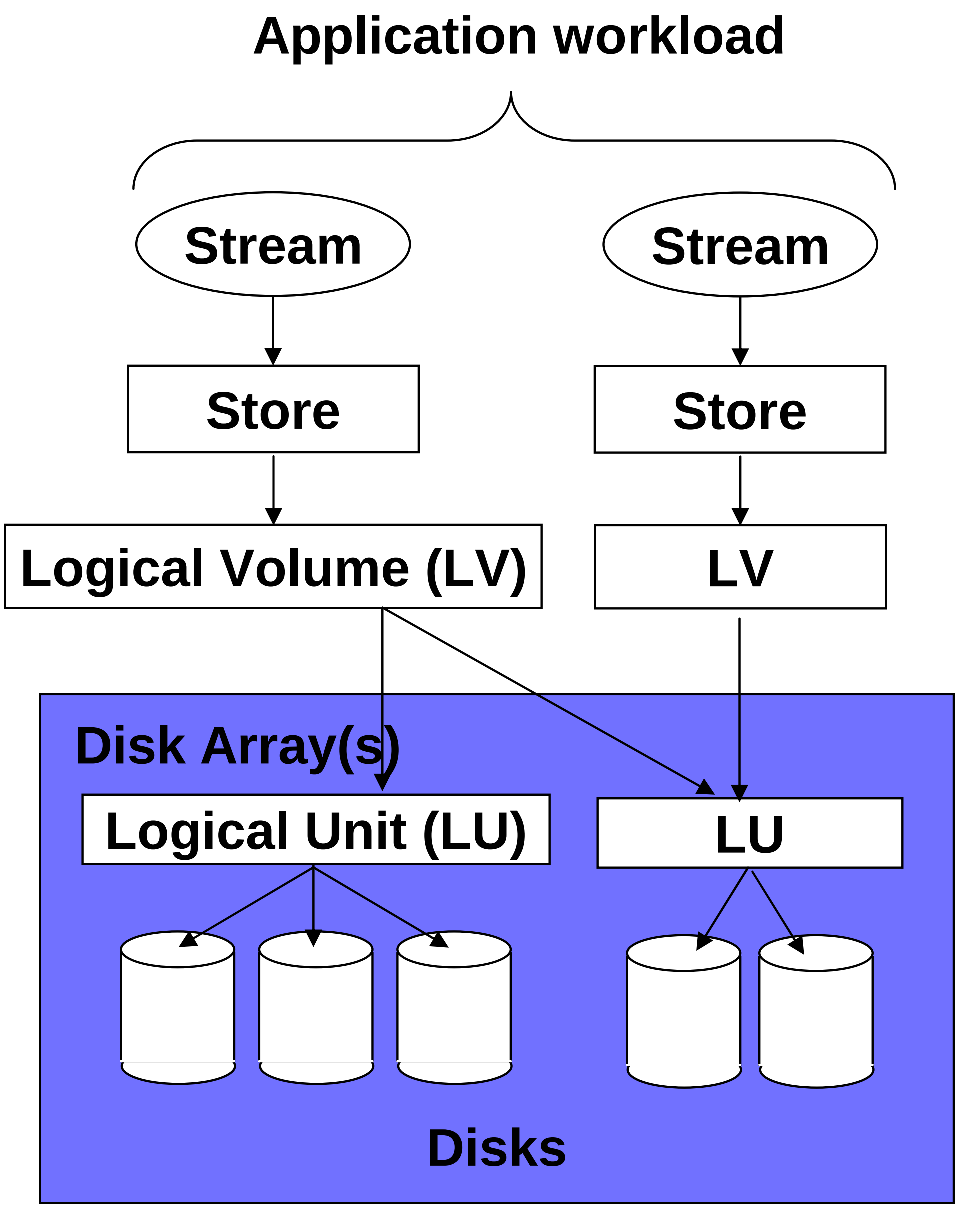
predictable performance

rate of growth

scale and complexity



Storage system configuration



- Application workloads
  - Streams of I/O requests
  - Stores: application data
- How to allocate storage resources?
- How to map workload onto storage devices?

Storage system configuration challenges

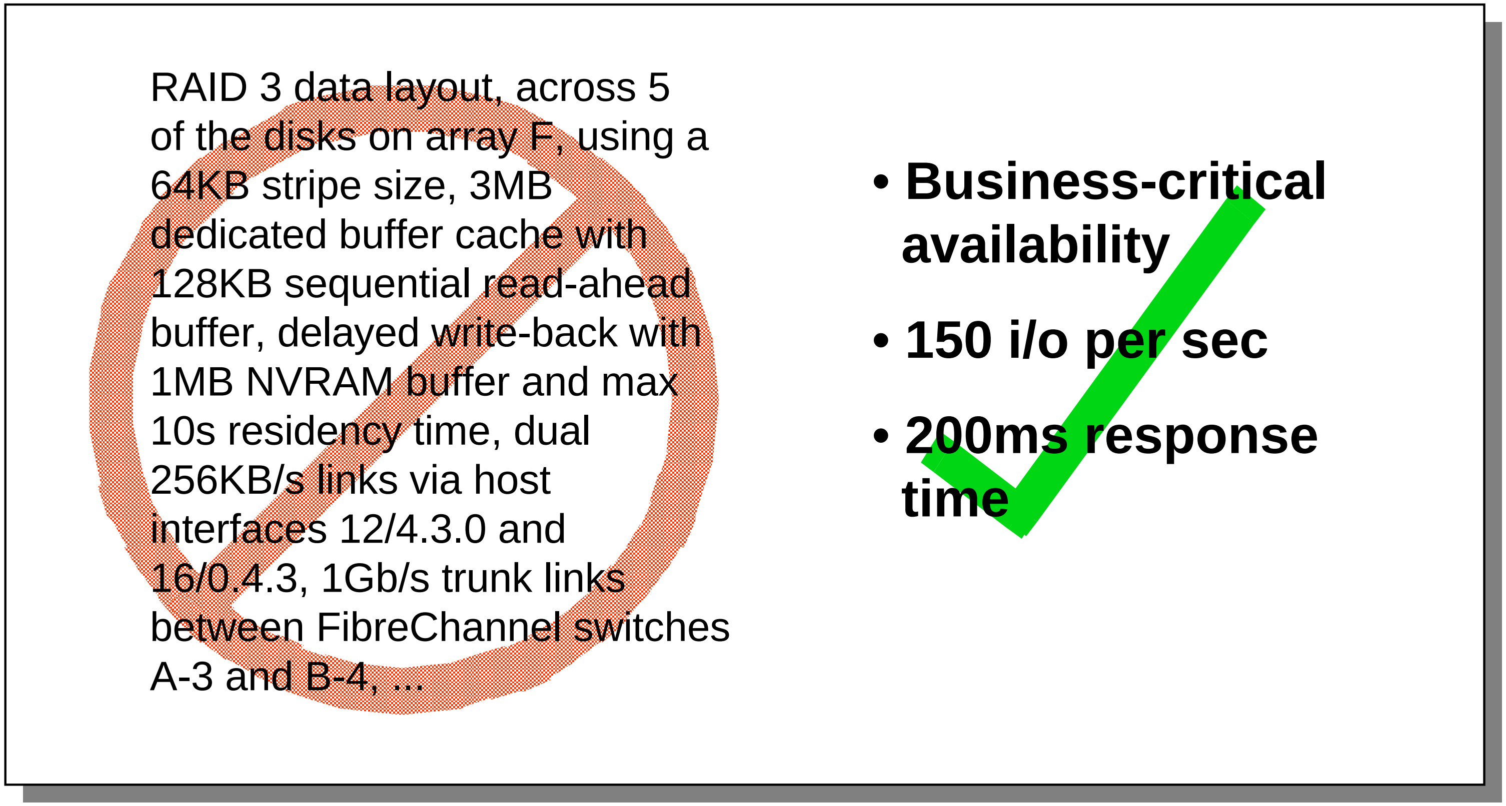
- **Design storage system to match workload requirements**
  - Which storage devices?
  - Configurations?
  - How to map workload data?
  - Too many choices
  - Insufficient workload knowledge
- **Implement design**
  - Configure disk arrays and hosts
  - Migrate existing app data (if any) to new design
  - Tedious and error-prone
  - Mistakes hard to find

Today’s approach

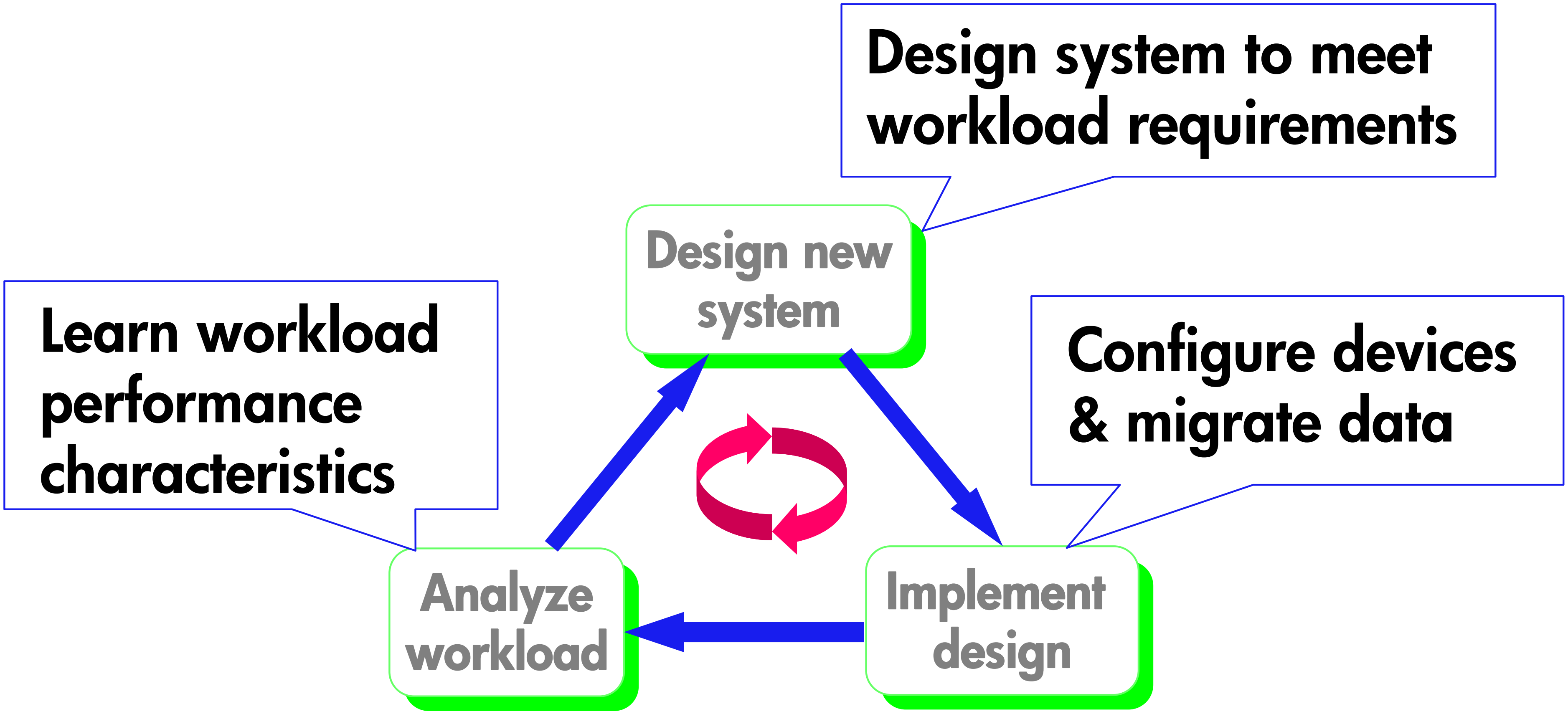
- **Storage system configuration is a naturally iterative process**
- **Human experts use “rules of thumb”**
- **Resulting systems:**
  - **Take too long to develop**
  - **Over-provisioned (cost too much)**
  - **Under-provisioned (perform poorly)**

Our solution

- Automate tasks that are difficult for human admins
- Goal-directed management
  - Users specify what they want (goals), not how to achieve it (implementation)



Hippodrome: automating storage config

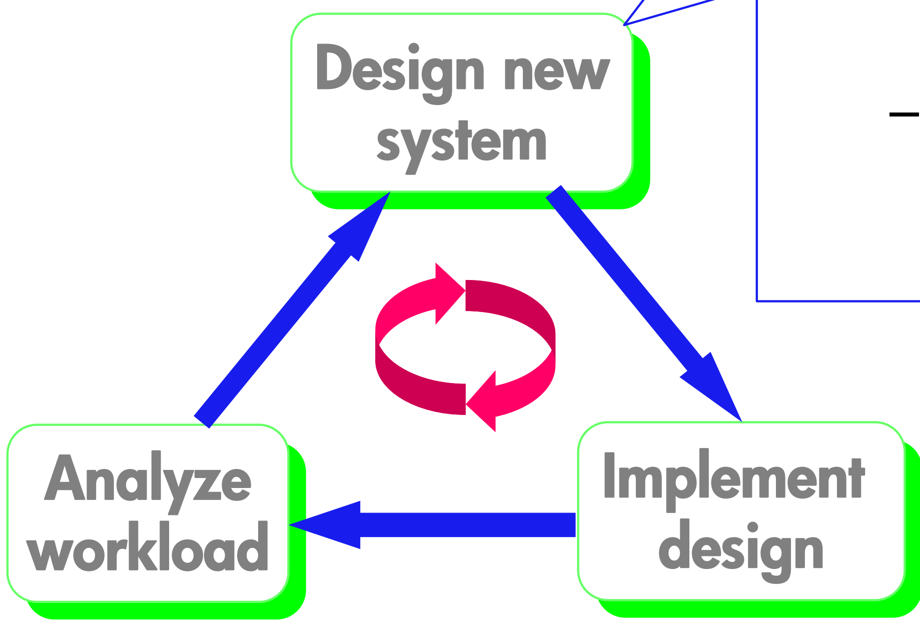


- Automatically designs and configures storage system

Outline

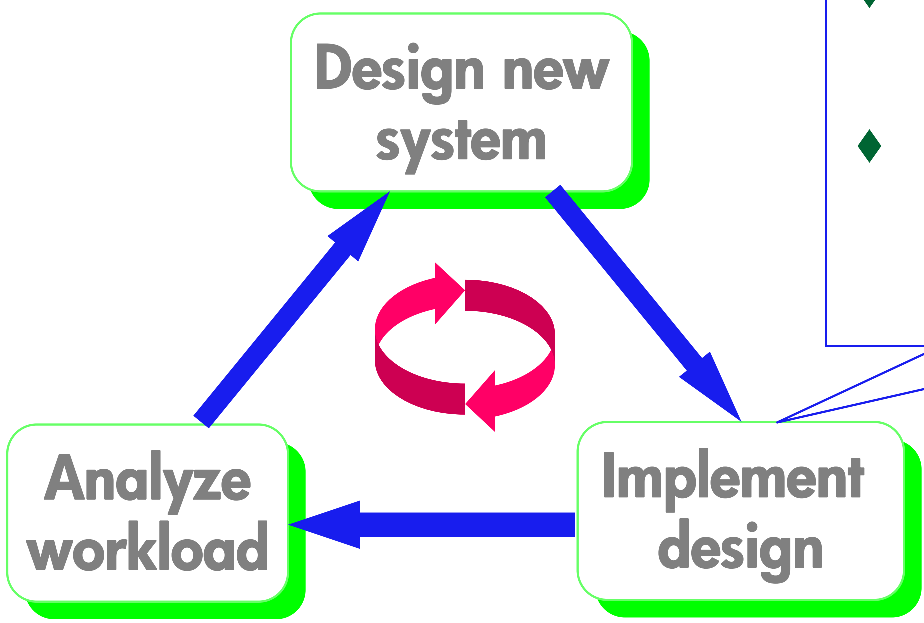
- Introduction
- Hippodrome approach
  - Hippodrome component overview
  - Loop operation
  - Questions to answer
- Hippodrome loop components
- Experimental methodology and results
- Related work
- Conclusions

Hippodrome components



- Efficiently explore large design space
  - Constraint-based optimization
  - Model-based performance predictions

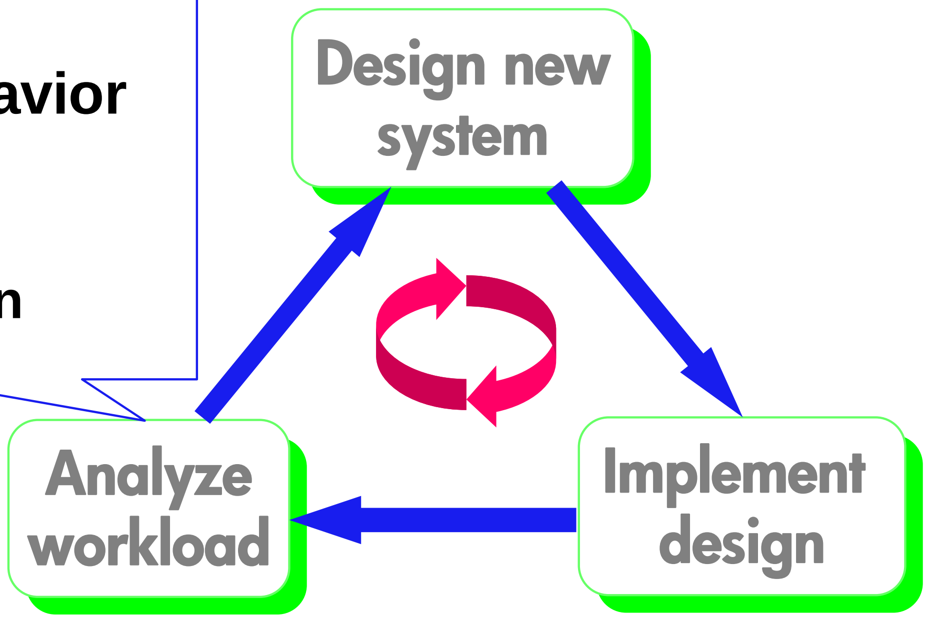
Hippodrome components



- Configure devices & hosts
- Migrate data from existing system to target system

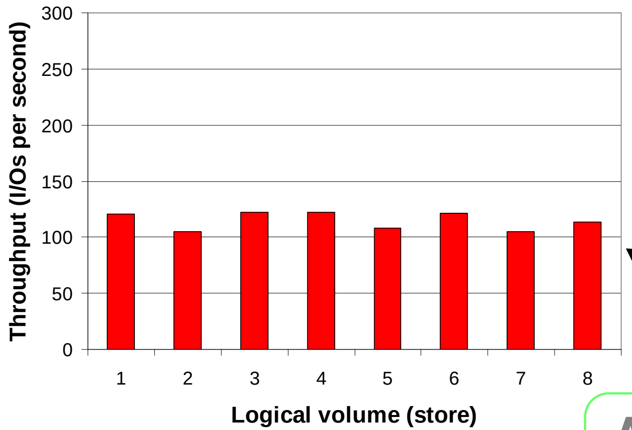
Hippodrome components

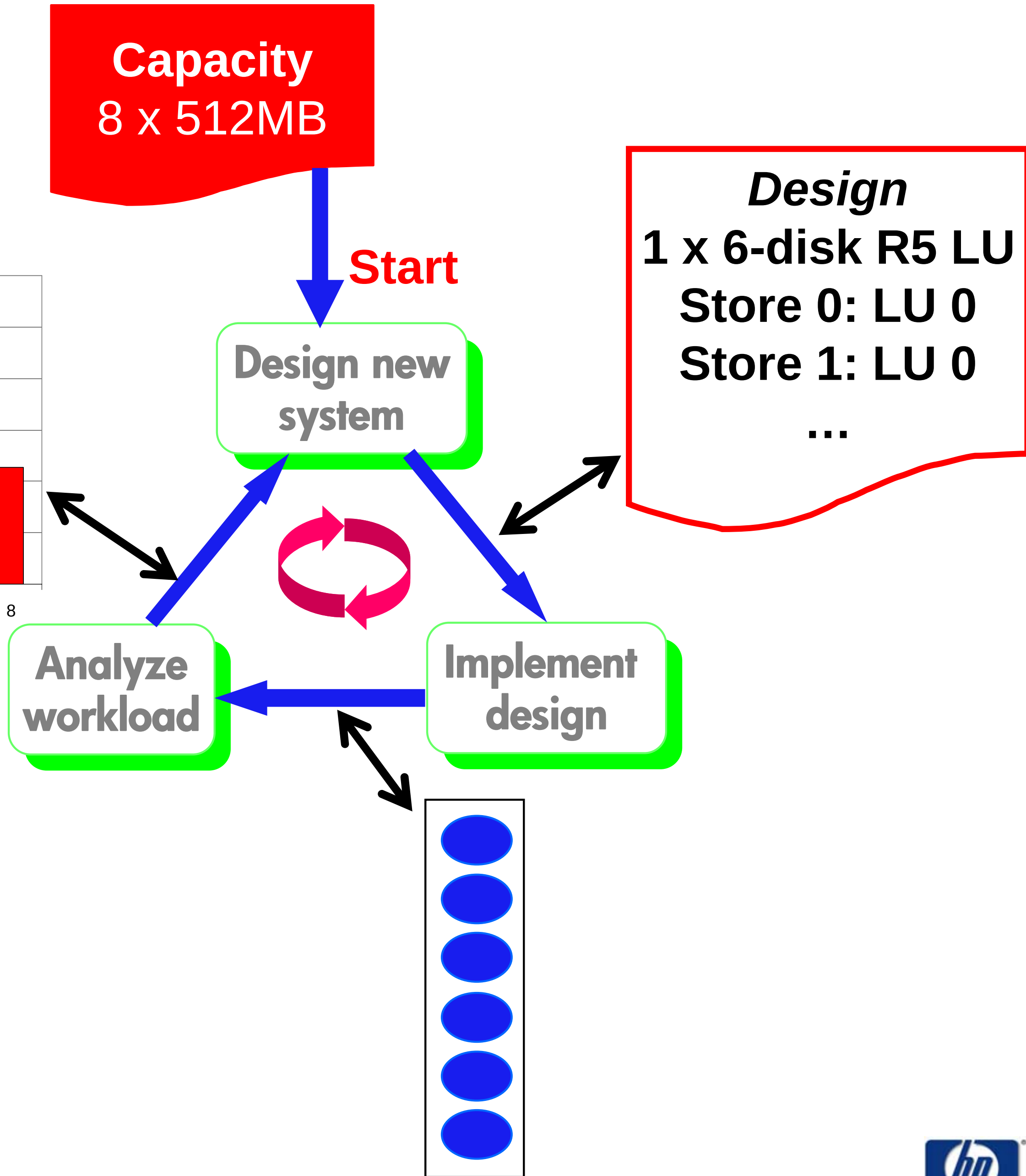
- Understand workload behavior
  - I/O trace -> declarative specification



Hippodrome loop operation: example

Iteration 1





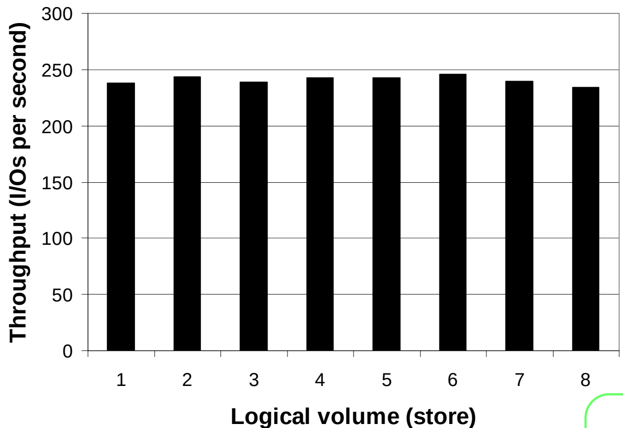
Hippodrome loop operation: example

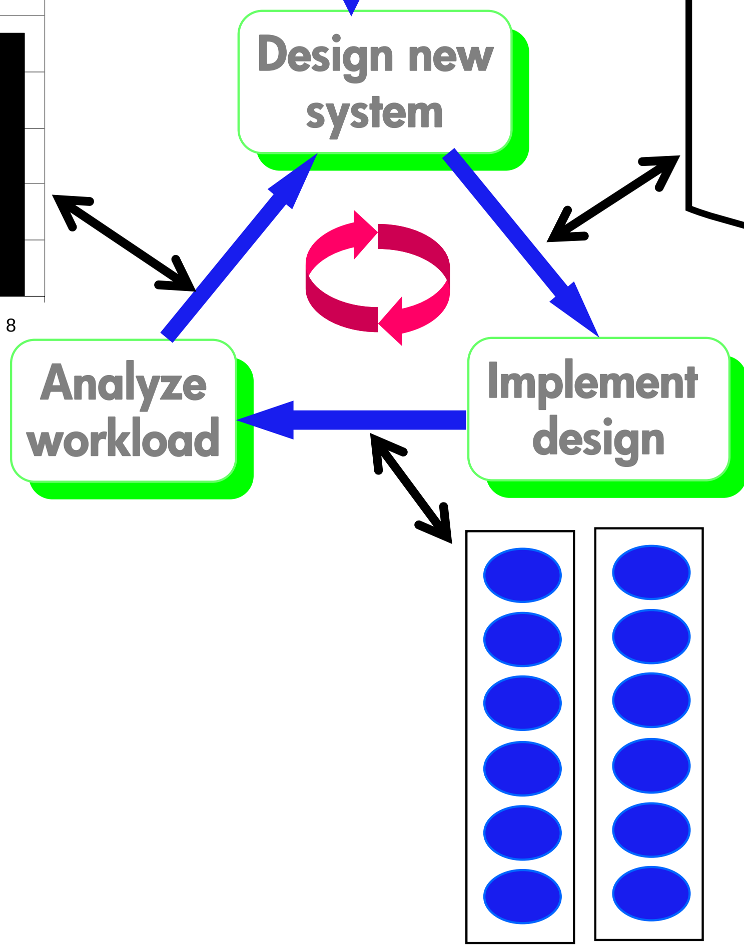
Iteration 2

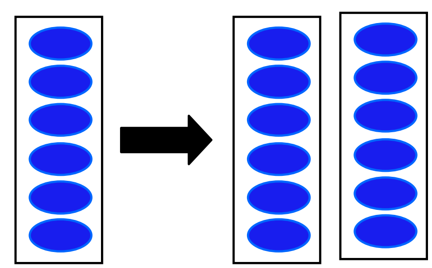
Capacity & performance
8 x 512MB
120 req/sec

Add new resources

Design
2 x 6-disk R5 LU
Store 0: LU 0
Store 1: LU 1
…







Migrate to new resources

Hewlett-Packard Laboratories

Questions for automatic loop

- Does it converge to a valid storage design?
  - Yes

Questions for automatic loop

- Does it converge to a valid storage design?
  - Yes
- How long does it take to converge?
  - A few loop iterations

Questions for automatic loop

- Does it converge to a valid storage design?
  - Yes
- How long does it take to converge?
  - A few loop iterations
- Is the solution stable (i.e., doesn’t oscillate)?
  - Yes

Questions for automatic loop

- Does it converge to a valid storage design?
  - Yes
- How long does it take to converge?
  - A few loop iterations
- Is the solution stable (i.e., doesn’t oscillate)?
  - Yes
- How little information must be provided?
  - Capacity requirements only

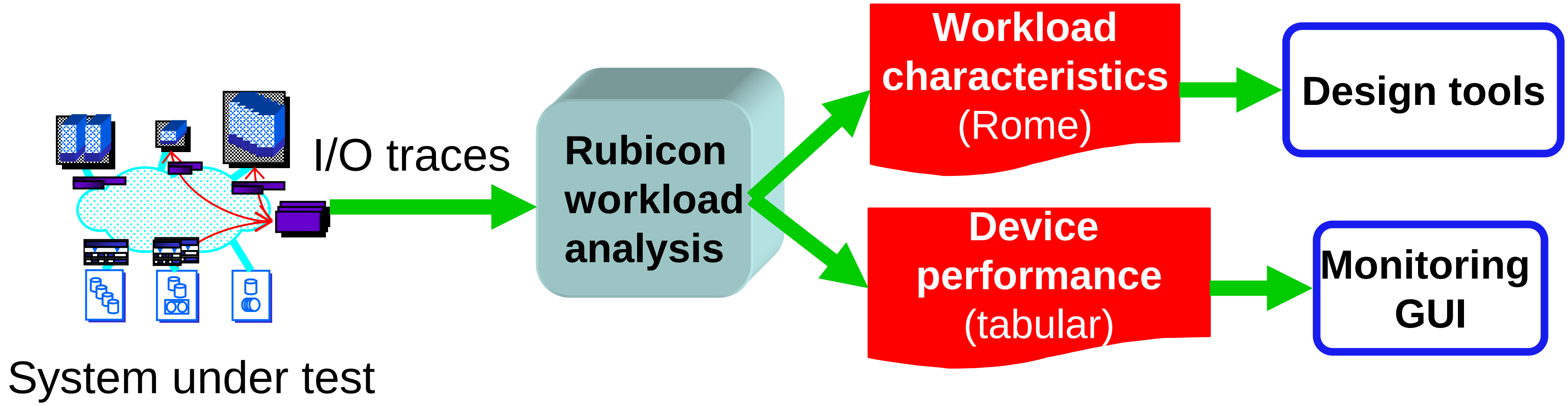
Questions for automatic loop

- Does it converge to a valid storage design?
  - Yes
- How long does it take to converge?
  - A few loop iterations
- Is the solution stable (i.e., doesn't oscillate)?
  - Yes
- How little information must be provided?
  - Capacity requirements only
- How do its solutions compare with solutions from a human administrator?
  - Performance within 15% of human solutions

Outline

- Introduction
- Hippodrome approach
- Hippodrome loop components
  - Workload analysis
  - System design – solver
  - System design – storage device models
  - Configuration
  - Migration planning and execution
- Experimental methodology and results
- Related work
- Conclusions

Rubicon workload analysis



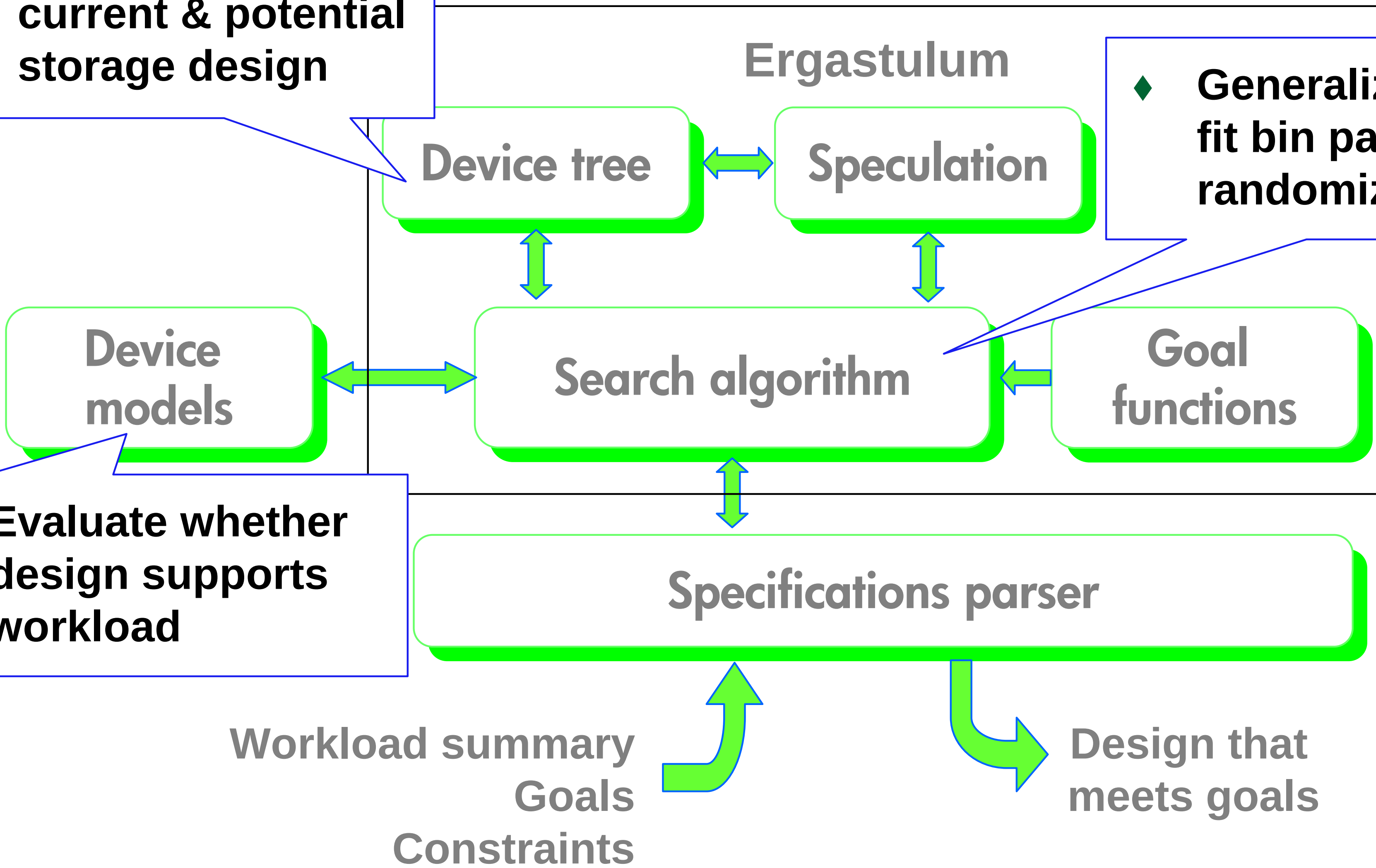
- **Goal:**
  - Succinctly capture important workload requirements and behaviors by analyzing I/O trace
- Workload = set of stores + streams
  - Stores capture static requirements (e.g., capacity)
  - Streams capture dynamic workload requirements (e.g., bandwidth, request size) to a store
- Workload behaviors of interest:
  - request size and rate, read:write ratio
  - spatial and temporal locality, burstiness
  - phased behavior, correlations between parts of storage system

Storage system design complexity

- **Numerous configuration parameters:**
  - Number and type of arrays, LU sizes, RAID types, array configuration, SAN layout, store allocation, …
  - Millions of candidate designs for a large problem!
- **Multiple (non-additive) constraints**
  - Capacity, performance utilization
  - Simpler version of problem (capacity-only constraints and fixed-size disks) is NP-complete
- **Human sys admins don't find best solutions**
  - Hard to handle complexity
  - Solutions may under- or over-provision
- **Goals:**
  - Automate difficult search space exploration
  - Near-optimal solutions in a short time

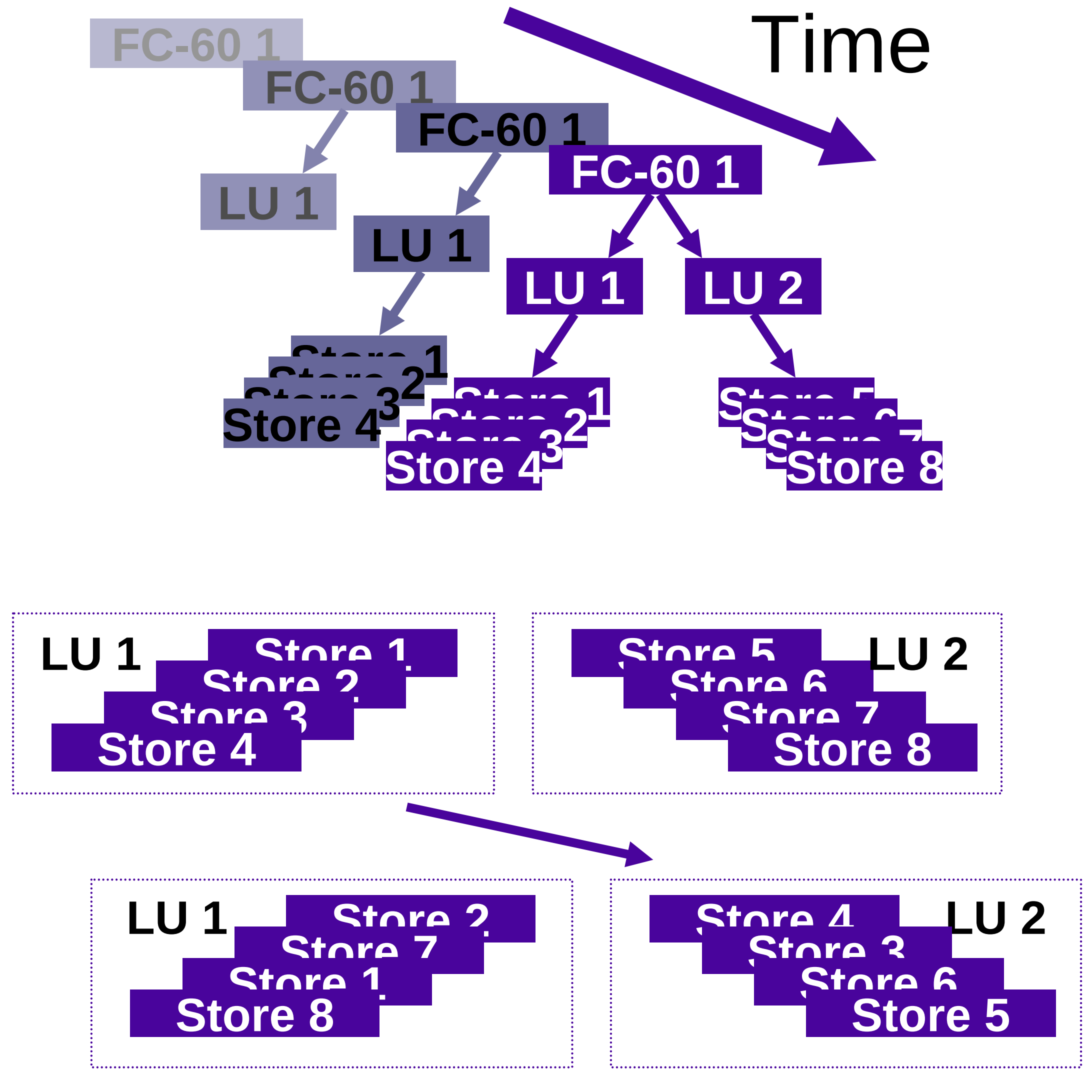
Ergastulum solver: automated design

- **Data structure: current & potential storage design**
- **Generalized best-fit bin packing w/ randomization**
- **Evaluate whether design supports workload**



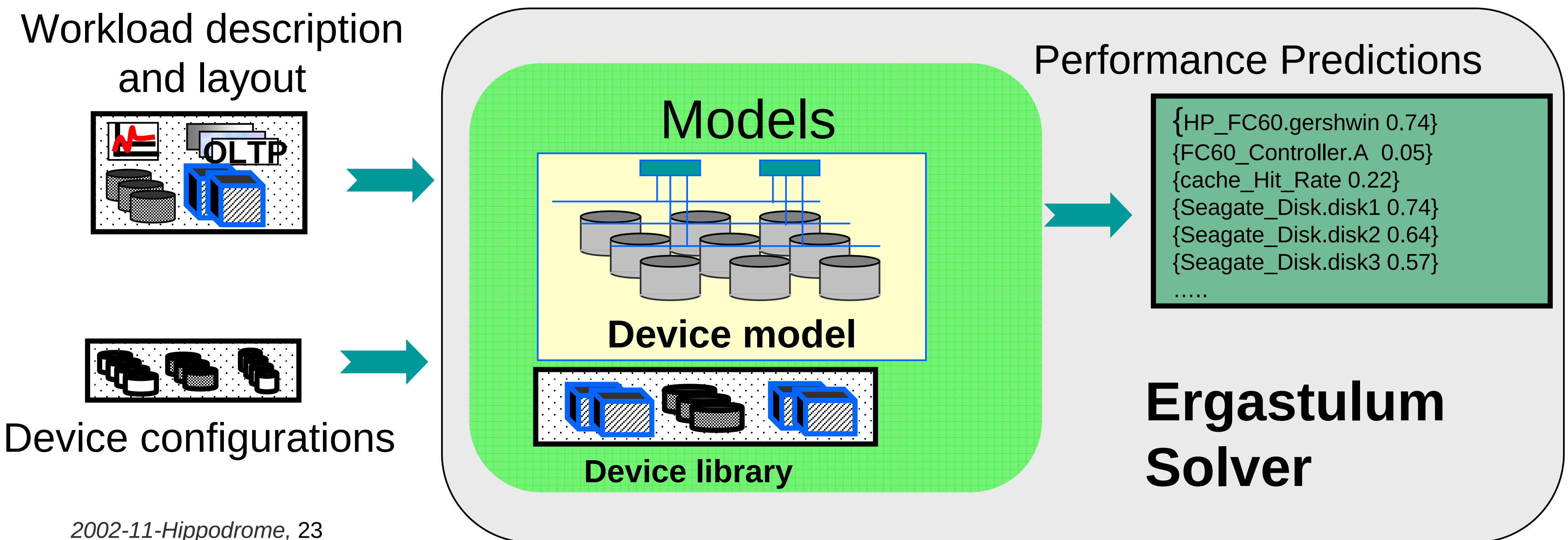
Ergastulum solver – search algorithm

- **Inputs:**
  - Workload specification
  - Goals and constraints
- **Step 1: initial system design and data mapping**
  - For each store, add store:
    - Add to existing LU,
    - Extend LU,
    - Change LU configuration, or
    - Add new device/LU
- **Step 2: improve system design and mapping**
  - Reassign LUs to reduce no. LUs
  - Reassign stores to balance load
- **Output:**
  - System design, including device specification and data mappings



Storage device models

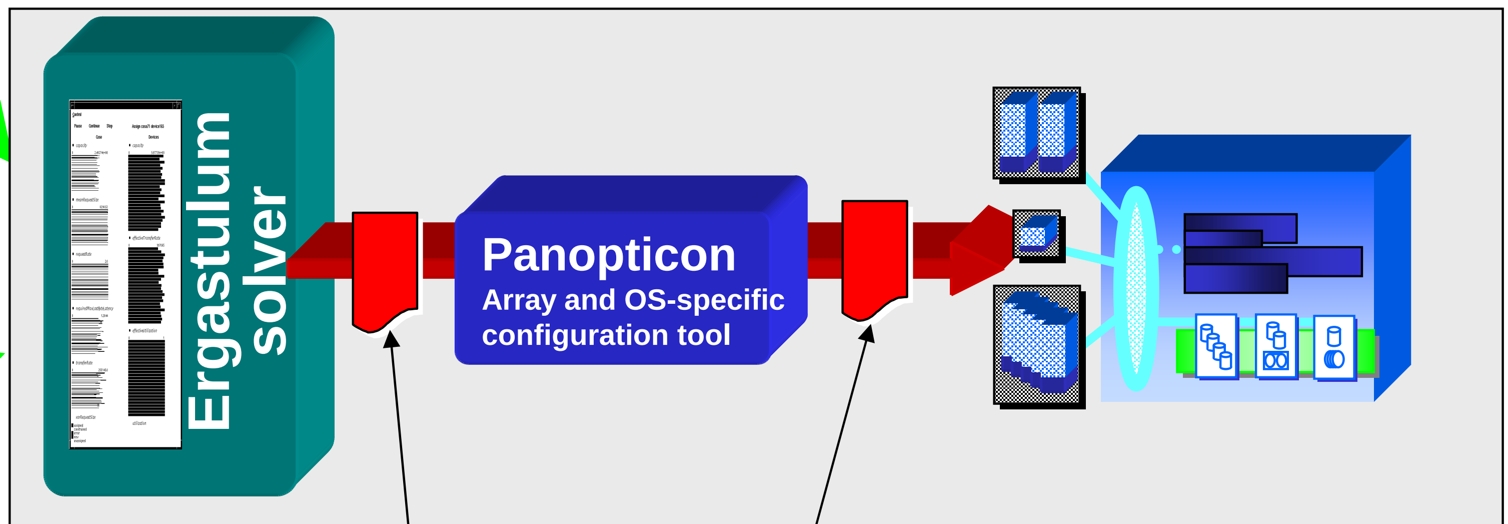
- **Fast, detailed and robust analytical models**
- **Approach 1: Delphi modular toolkit**
  - **Potential for reusable components: disks raid controllers, caches**
  - **Components impose workload transformations**
  - **Requires considerable human effort**
- **Approach 2: table-based models**
  - **Table of measured values to enable automatic creation**
  - **Look up workload utilization in table; use linear interpolation**



Automatic configuration

Workload specification

Business needs

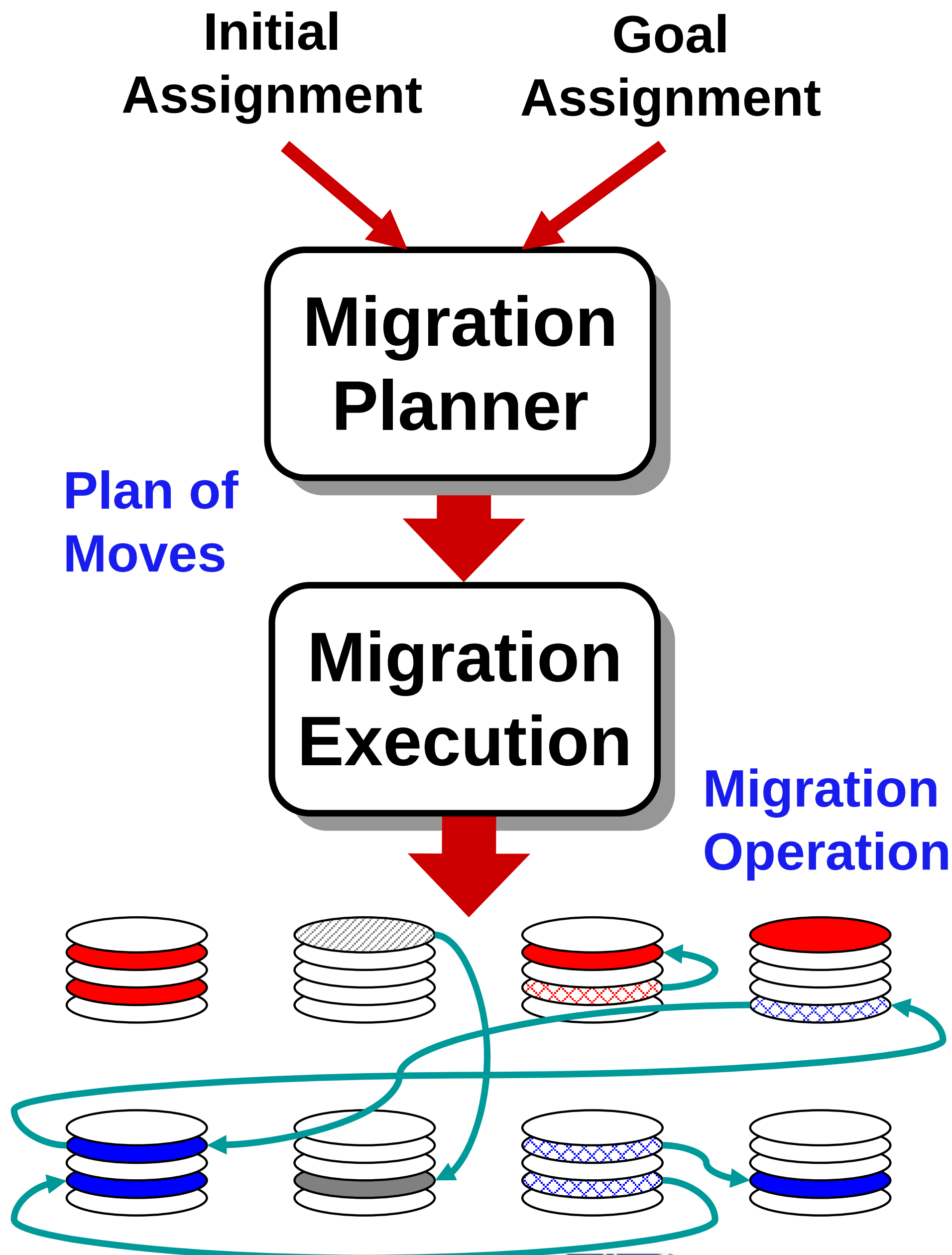


Design
(OS-neutral description of what to put on each storage device, and how to configure them)

Configuration
(LVM scripts, disk array LUs, ...)

Peregrinatio migration planning

- Migration planning shown to be NP hard
  - What order to move stores?
  - How to minimize temp space or number of moves?
  - How to improve performance through parallelism?
- Current implementation:
  - Simple greedy heuristic
  - Can find plans for 100s-1000s stores on 10s-100s of devices
- Developed advanced heuristics that provide:
  - Parallel migration plans
  - Proven upper bounds on temp space
  - Proven upper bounds on # moves

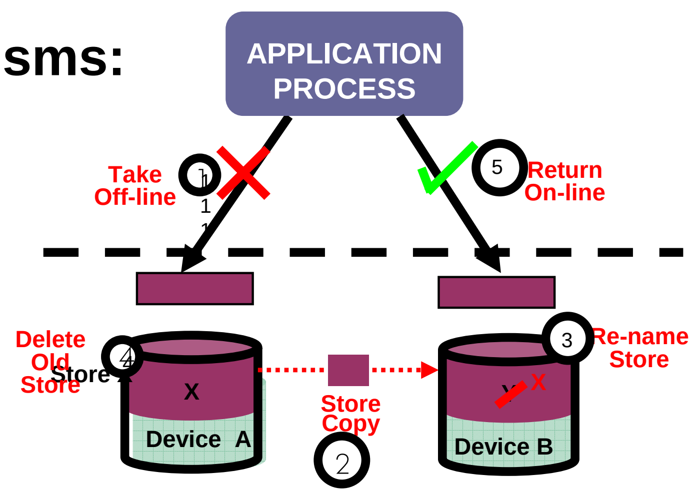


Hewlett-Packard Laboratories

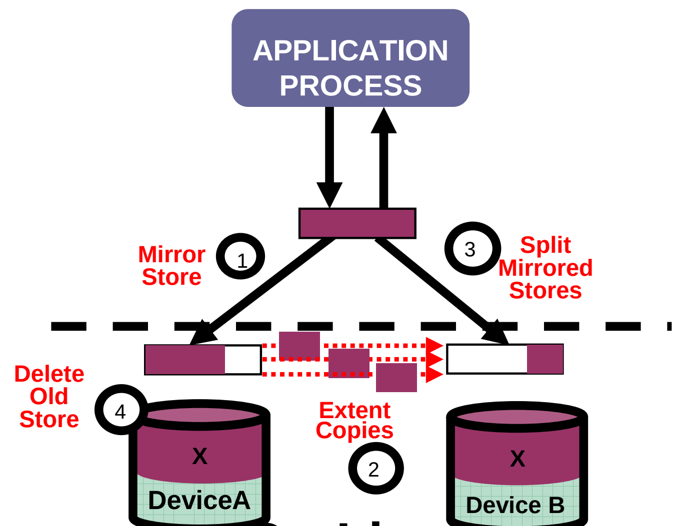
Migration execution

- **What is migration execution?**
  - Moving a store from its current location to a goal location
  - Must be done quickly and reliably
- **Two methods:**
  - Off-line: applications can't access stores during migration
  - On-line: applications can access stores during migration
- **Aqueduct provides online migration with minimal impact on foreground workload**

Mechanisms:



Off-Line



On-Line

The Rome specification system

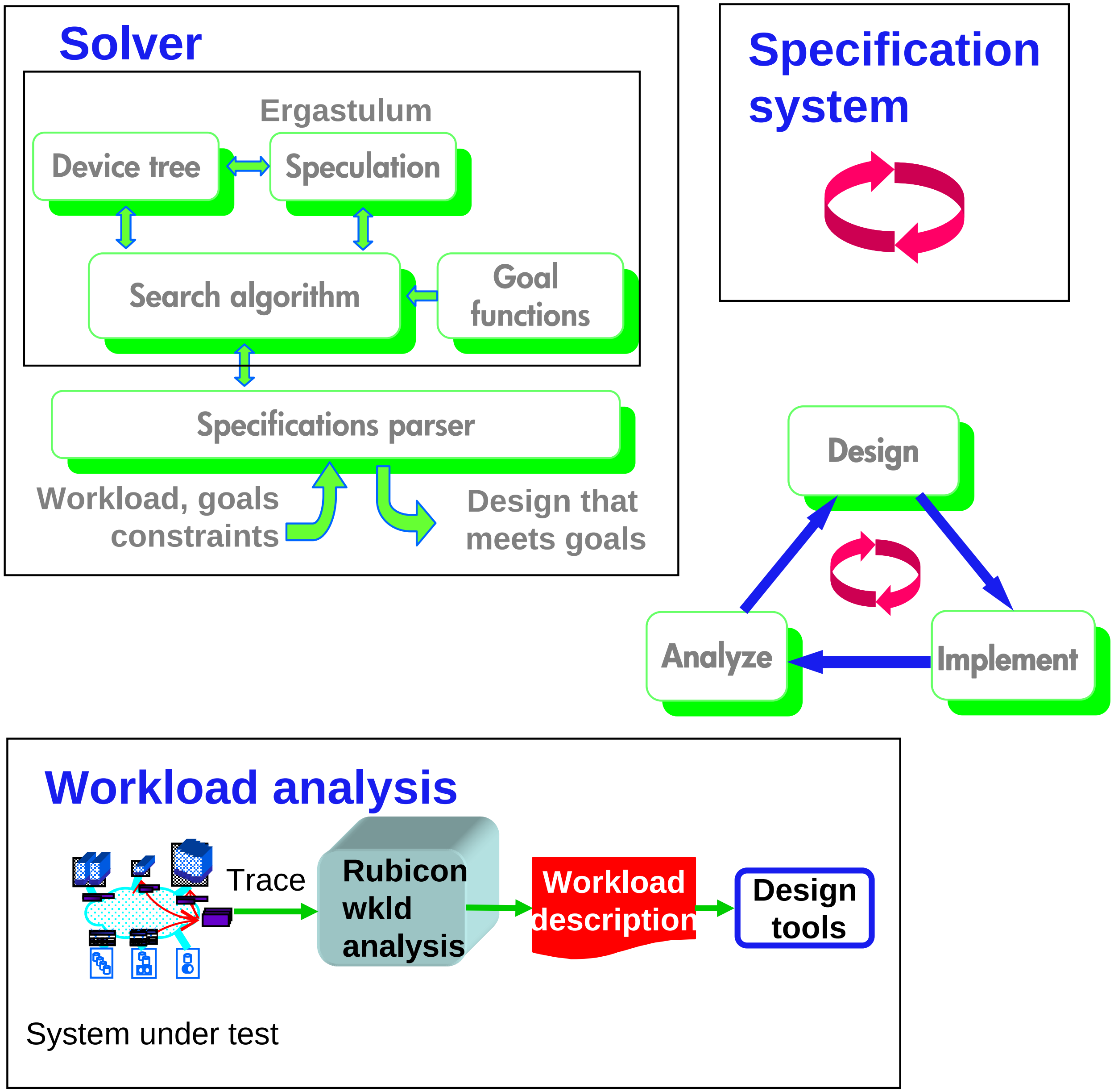
What it does:

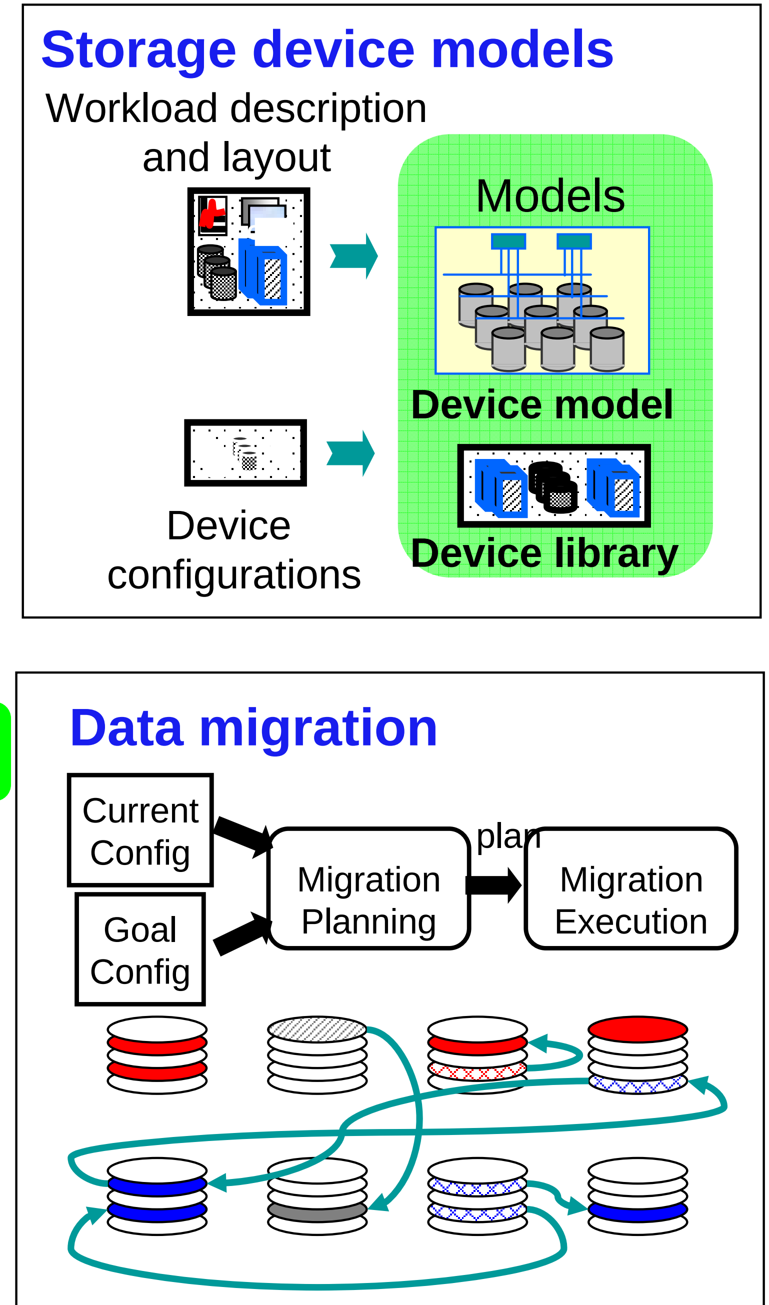
- Records user requirements
- Records workload requirements
- Represents hardware/software configurations
- Describes storage devices
  - capabilities and configuration options
- Describes assignments
  - workload to device mappings

What's in it:

- Rome data model
  - UML(-like) models for all objects in a storage system
- Languages (textual representations)
  - Latin: current Tcl-based variant
  - Greek: future XML variant

Summary: Hippodrome components



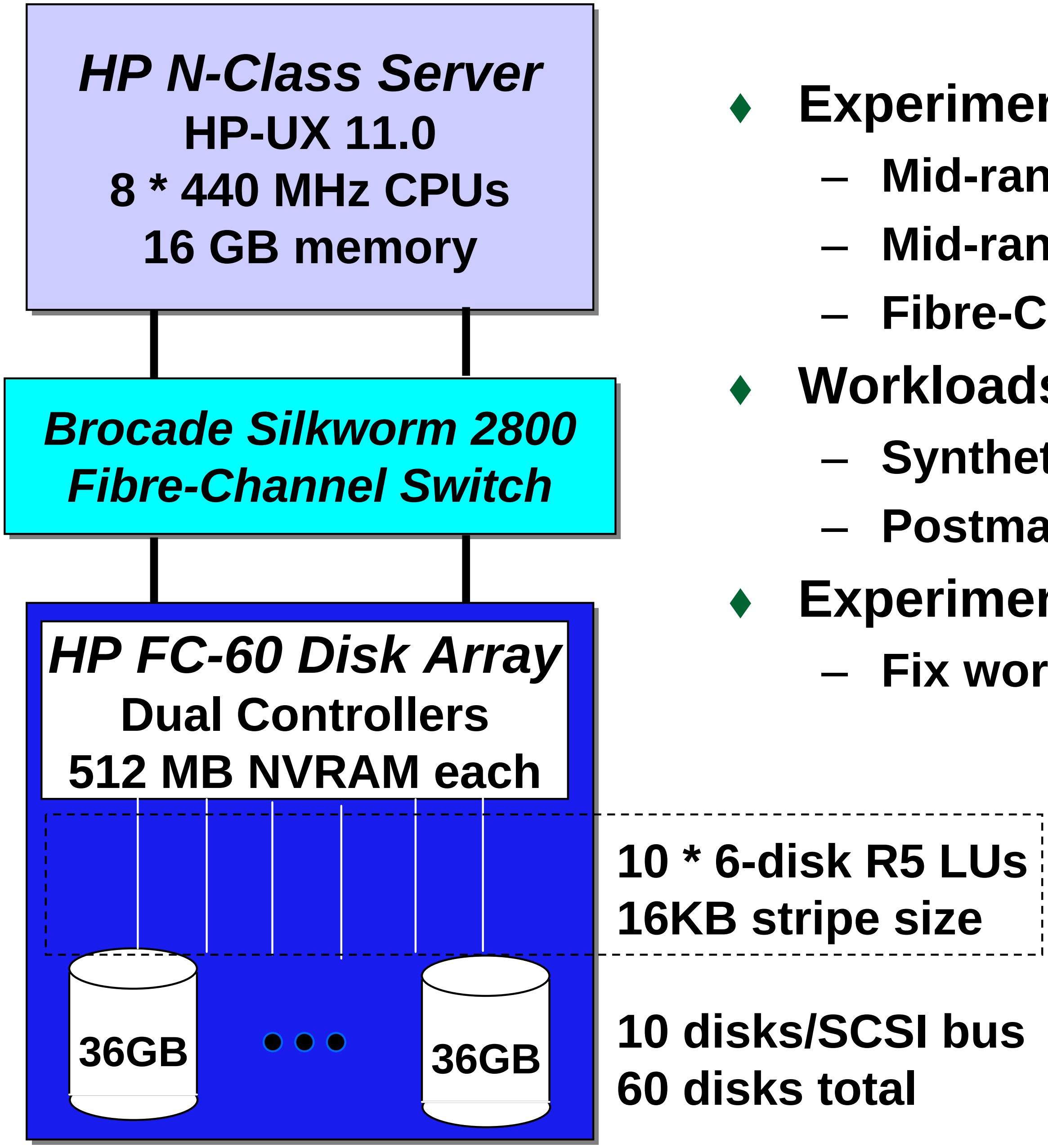


Laboratories
invent

Outline

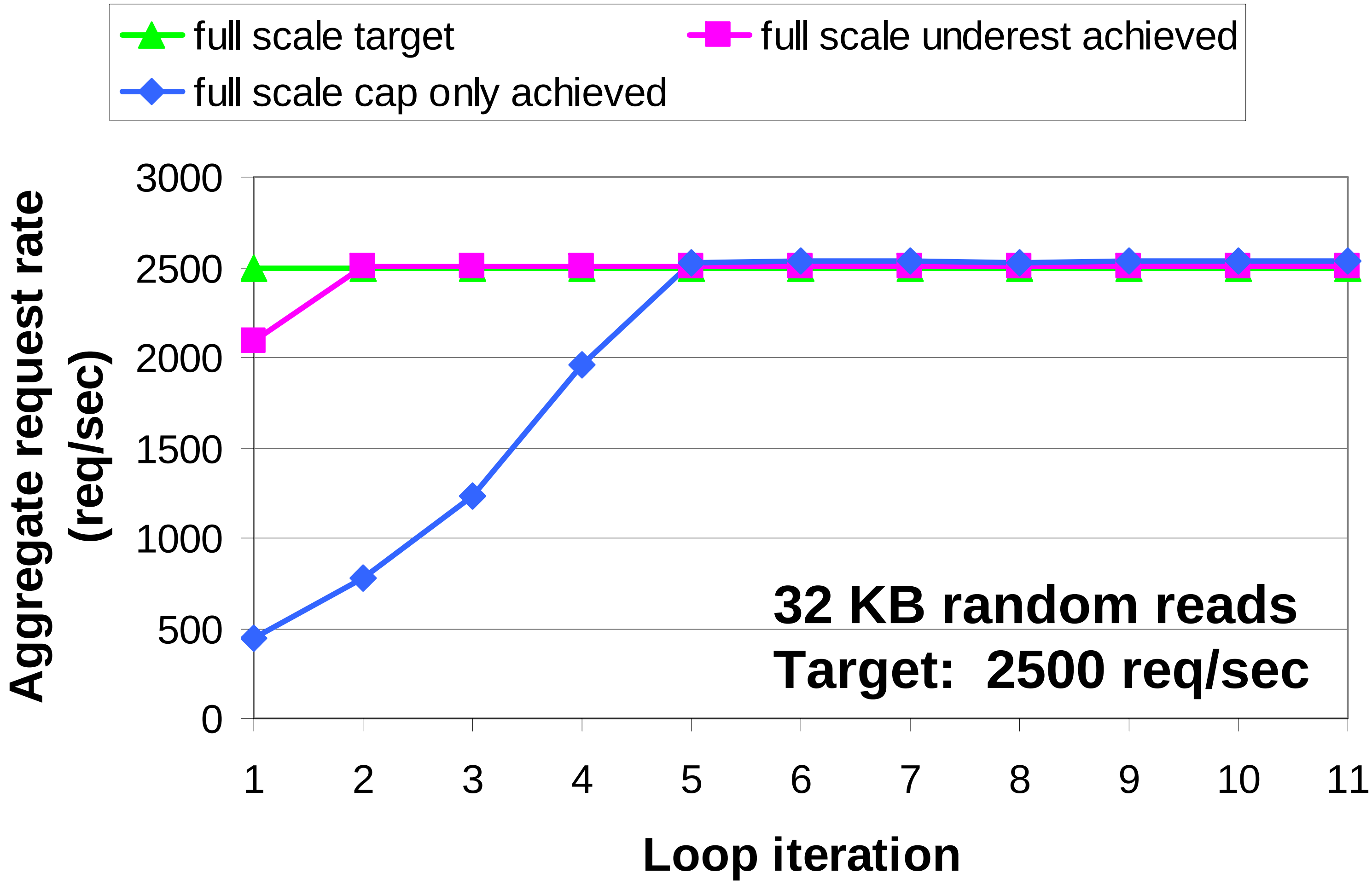
- Introduction
- Hippodrome approach
- Hippodrome loop components
- Experimental methodology and results
- Related work
- Conclusions

Experimental methodology



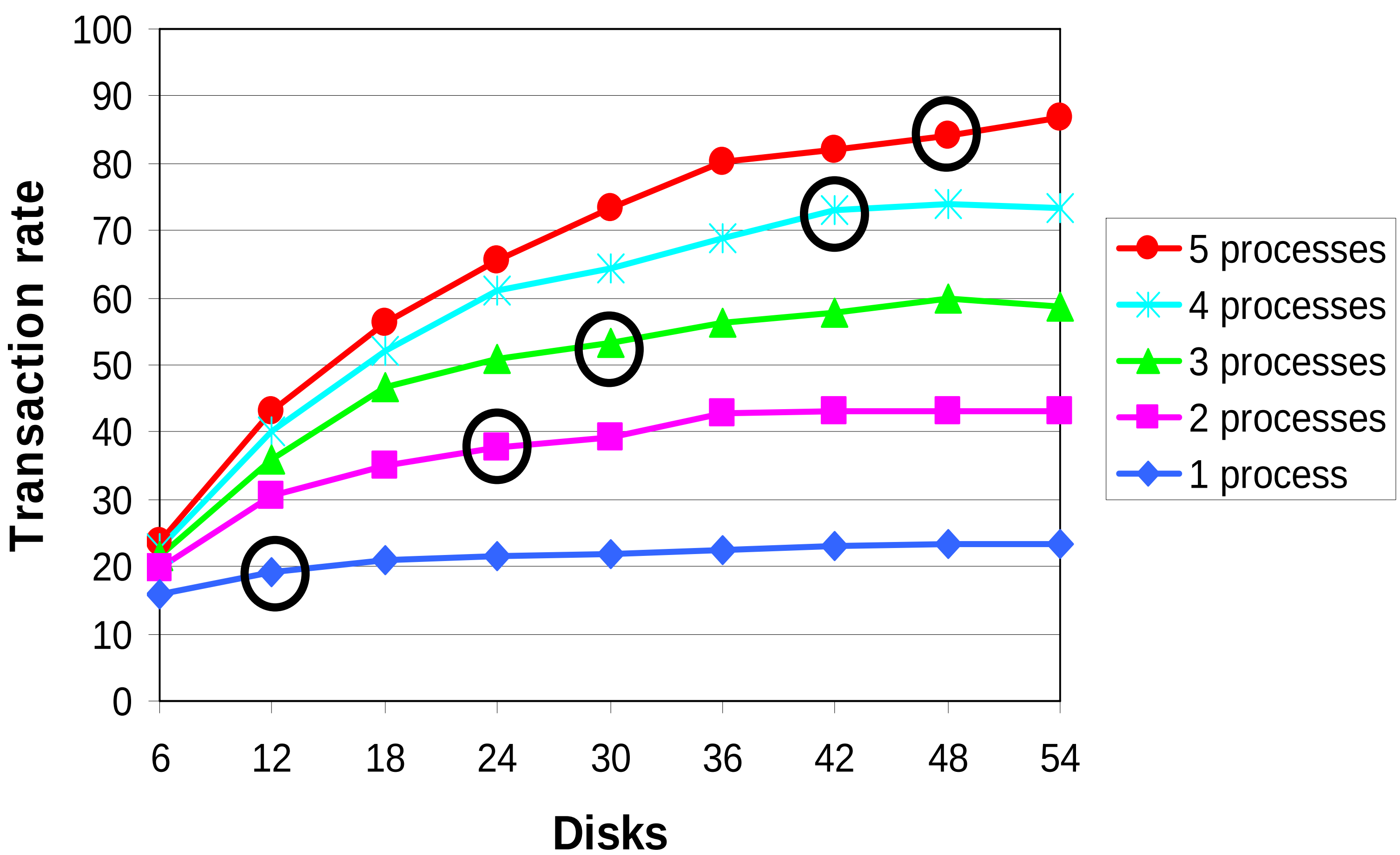
- Experimental system
  - Mid-range SMP host
  - Mid-range disk array
  - Fibre-Channel SAN
- Workloads:
  - Synthetic
  - Postmark
- Experiments:
  - Fix workload, iterate through loop

Experimental results: synthetic wkld



- **Both workloads converge to stable design**
- **Performance estimate helps**

Experimental results: PostMark



- 85% - 99% of best human-achieved performance
- Fast convergence to stable design

Experimental results - summary

- **Hippodrome system:**
  - **Rapidly converges to final design**
  - **Designs stable systems**
  - **Requires only capacity information**

- **Resulting solutions:**
  - **Use near-minimal resources**
  - **Perform within 15% of human expert**

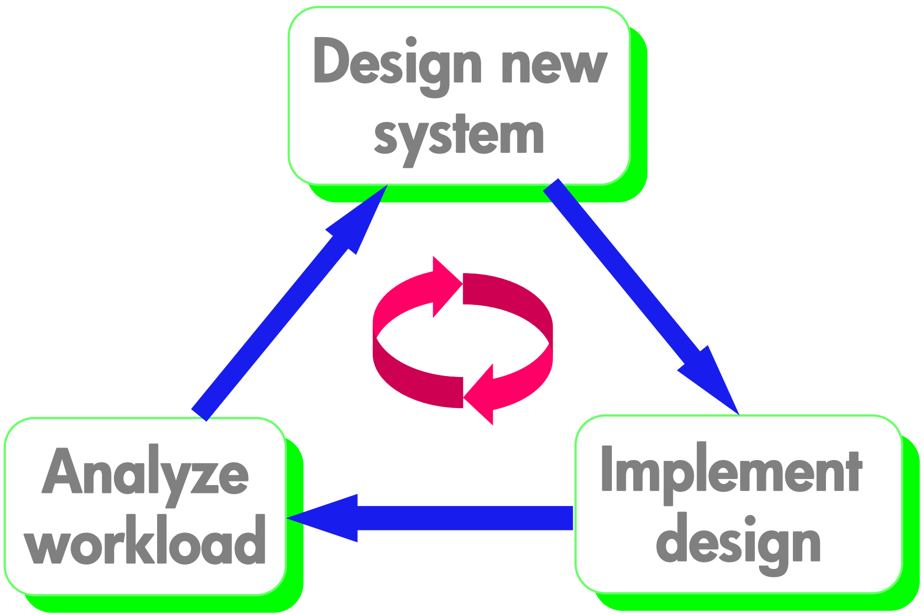
Hippodrome ongoing and future work

- **Continuous adaptation to workloads that change over time**
- **Greater range of workloads**
- **Sensitivity of loop effectiveness to quality of components**
- **Increased sophistication of loop components**

Related work

- Configuration adaptation inside a disk array
  - EMC Symmetrix, HP XP512, Hitachi 9900
  - HP AutoRAID
- Policy-based management
  - IBM StorageTank
- Database solutions
  - Teradata hash-based partitioning
  - UCB River cluster I/O architecture
  - Microsoft AutoAdmin automatic index selection
  - IBM DB2 LEO feedback-driven query optimizer
- Compute resource allocation
  - IBM Oceano automatic server allocation
  - Duke Muse energy-conscious server allocation

Conclusions



- **Key idea: self-managing storage**
- **Benefits to approach:**
  - **Better solutions**
  - **Reduced human error**
- **Reduces expensive management costs through automation**

Acknowledgments

- Hippodrome and its components have been developed by the following Storage System Department members:

- Eric Anderson, Guillermo Alvarez, Michael Hobbs, Mahesh Kallahalla, Kim Keeton, Arif Merchant, Susan Spence, Ram Swaminathan, Mustafa Uysal, Alistair Veitch, Qian Wang and John Wilkes

invent

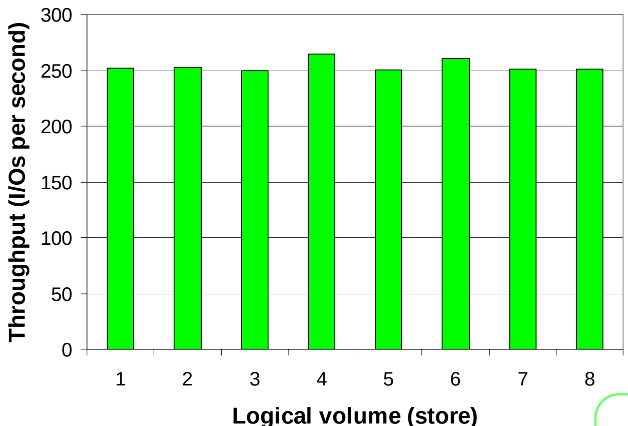
http://www.hpl.hp.com/SSP/

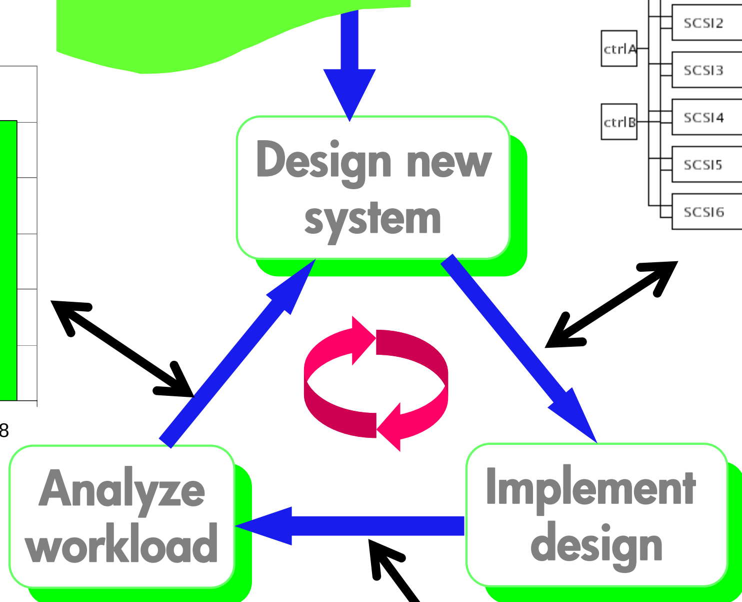
Backup slides

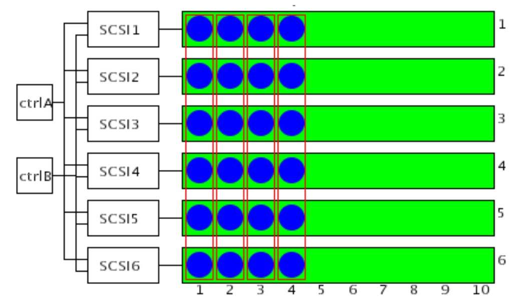
Hippodrome loop operation: example

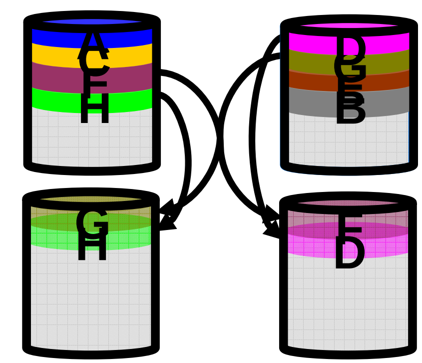
Iteration 3

Capacity &
performance
8 x 512MB
240 req/sec







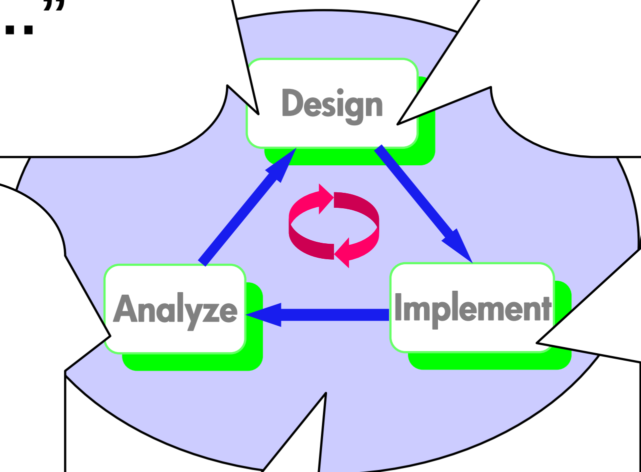


Migrating - 12 to 24 disks

Related FAST '02 papers

Anderson, et al., "Selecting RAID levels…"

Ward, et al., "Appia: Automatic SAN Fabric Design"



Kurmas, et al., "Distillation of Performance-related Characteristics"

Lu, et al., "Aqueduct: Online Data Migration…"

Anderson, et al., "Hippodrome: running circles…"