A Self-Managing Storage System

Erik Riedel

Hewlett-Packard Labs,

Storage Systems Program

1501 Page Mill Road

Palo Alto, CA 94304

(650) 857-3365

FAX (650) 857-5548

riedel@hpl.hp.com

Talk overview

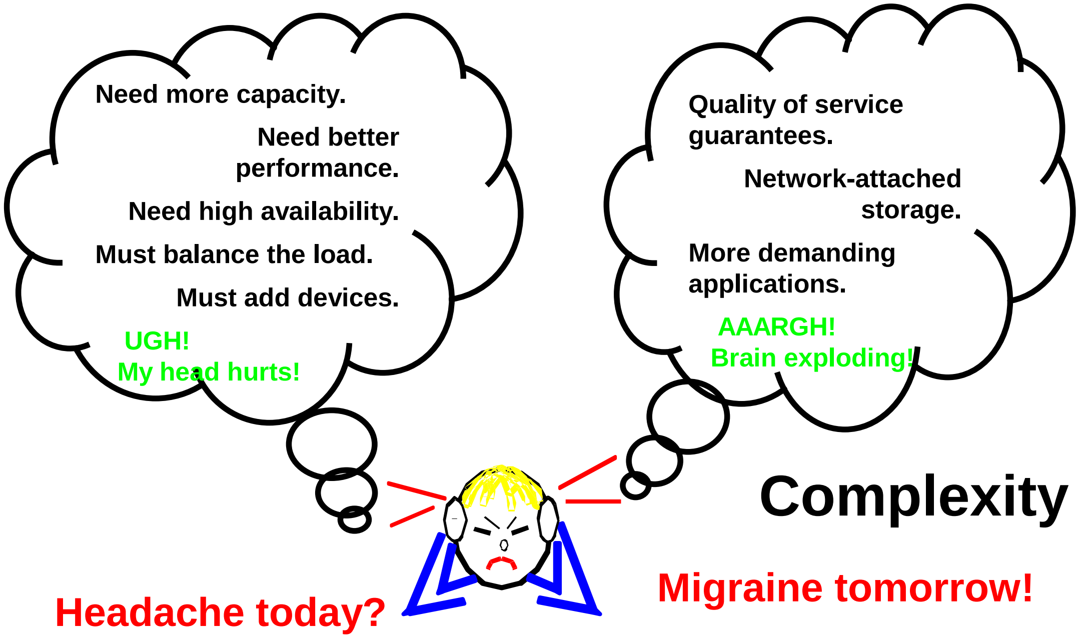
- **Introduction**
  - why storage is important
  - customer problems
  - our goals
- **Our vision - self-managing storage**
- **Research challenges**
- **Prototype**
- **Conclusions**
- **Future**

Introduction – why do we care?

- **Storage systems**
  - **the place where persistent data is kept**
  - the center of the universe!
- **Why?**
  - **information (stored data) is the key to most endeavors**
  - **storage is big business (tens of $billion per year)**
  - **sheer quantities (hundreds of *petabytes* per year)**

  - ***"Storage will dominate our business in a few years"***
    - *Compaq VP, 1998*
  - ***"In 3 to 5 years, we will start seeing servers as peripherals to storage"***
    - *SUN Chief Technology Officer, 1998*
  - ***"We'll plug into whatever servers you have"***
    - *IBM Versatile Storage Server ad, 1999*

Customer problems

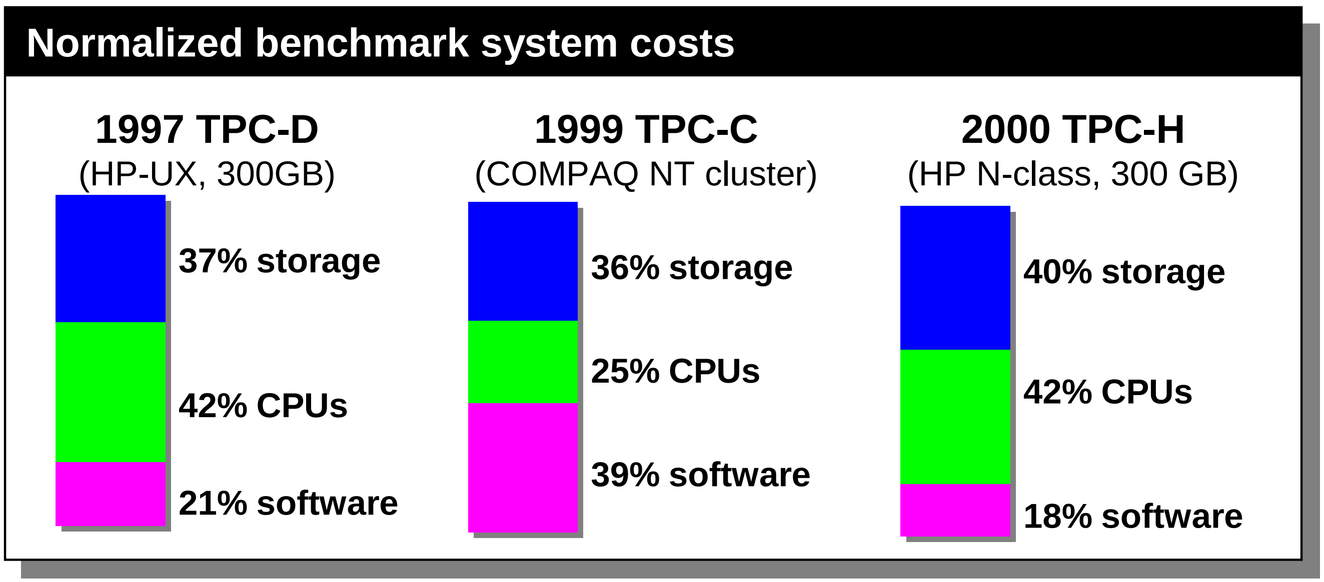


Too many knobs!

RAID 3 layout, across 5 disks on array F, with 64KB stripe size, 30 MB dedicated cache with 128KB sequential read-ahead, delayed write-back with 1 MB NVRAM and max 10 s residence time, dual 100 MB/s links via host interfaces 12/4.3.0 and 16/0.4.3, 1Gb/s trunk links between switches A-3 and B-4, ...

- Business-critical availability
- 150 i/o per sec
- 200ms response time

Knobs



Cost

Scale

- TB systems are common today

- estimates: average size business will have 100 TB of data by 2003

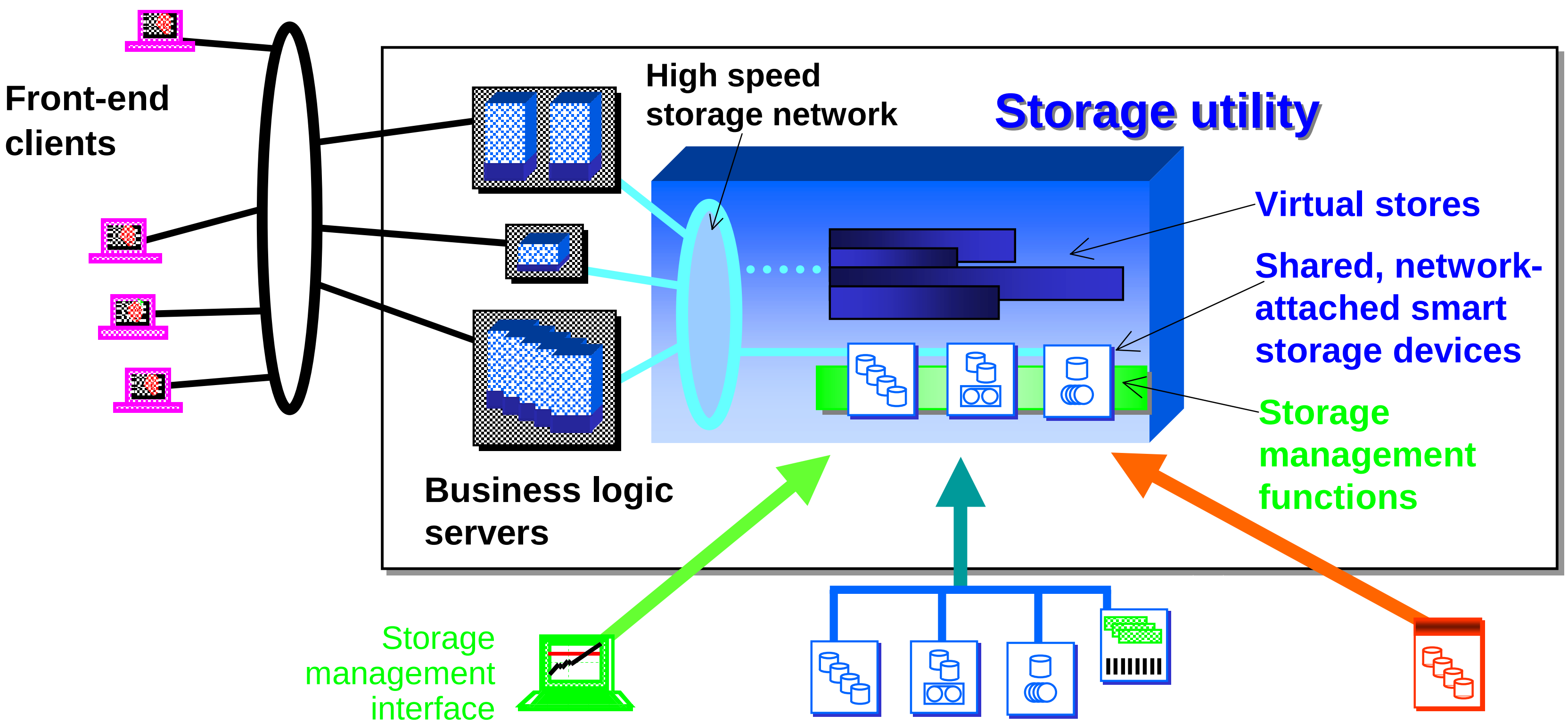
stress-free storage

reduced people-print

Talk overview

- Introduction
- Our vision - self-managing storage
  - the storage utility
  - automatic management lifecycle
- Research challenges
- Prototype
- Conclusions
- Future

The storage utility



Attribute management
• **what** to do, not **how** to do it

Distributed storage manager
• dynamically-mapped, scalable, host-independent storage
• online data migration

Network attached storage devices
• QoS, security, smart devices

The storage utility – how is it done?

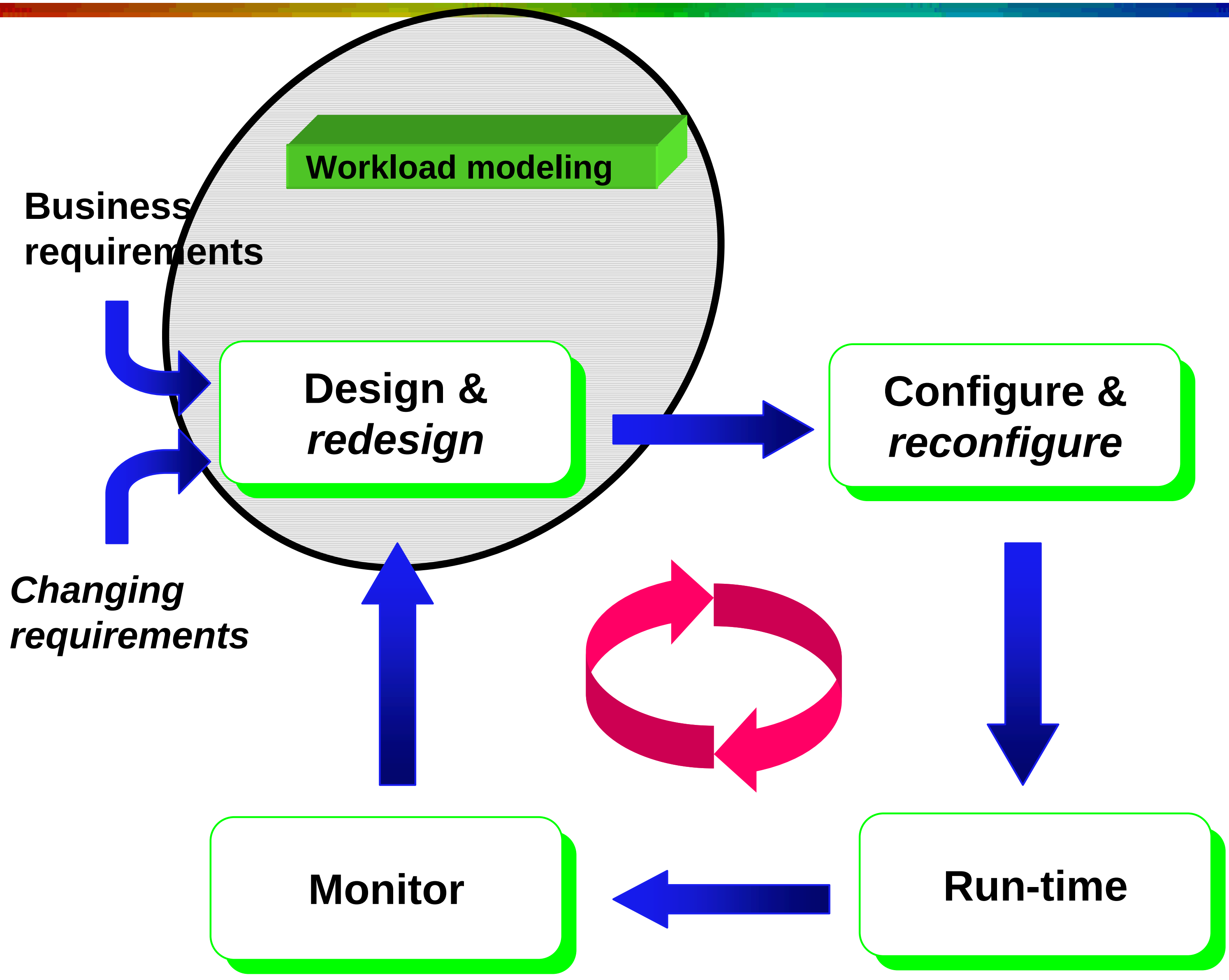
the Big Ideas ...

- **Goal-directed self-management**
  - specify what to do, not how to do it
- **Automatic (re)design and (re)configuration**
  - to reduce complexity & human effort
- **Predictable behavior through guarantees**
  - QoS = performance + availability + cost
- **Software as the key differentiator**
  - online monitoring, online management

Talk overview

- Introduction
- Our vision - self-managing storage
- Research challenges
  - across all parts of the lifecycle
- Prototype
- Conclusions
- Future

Automatic-management lifecycle



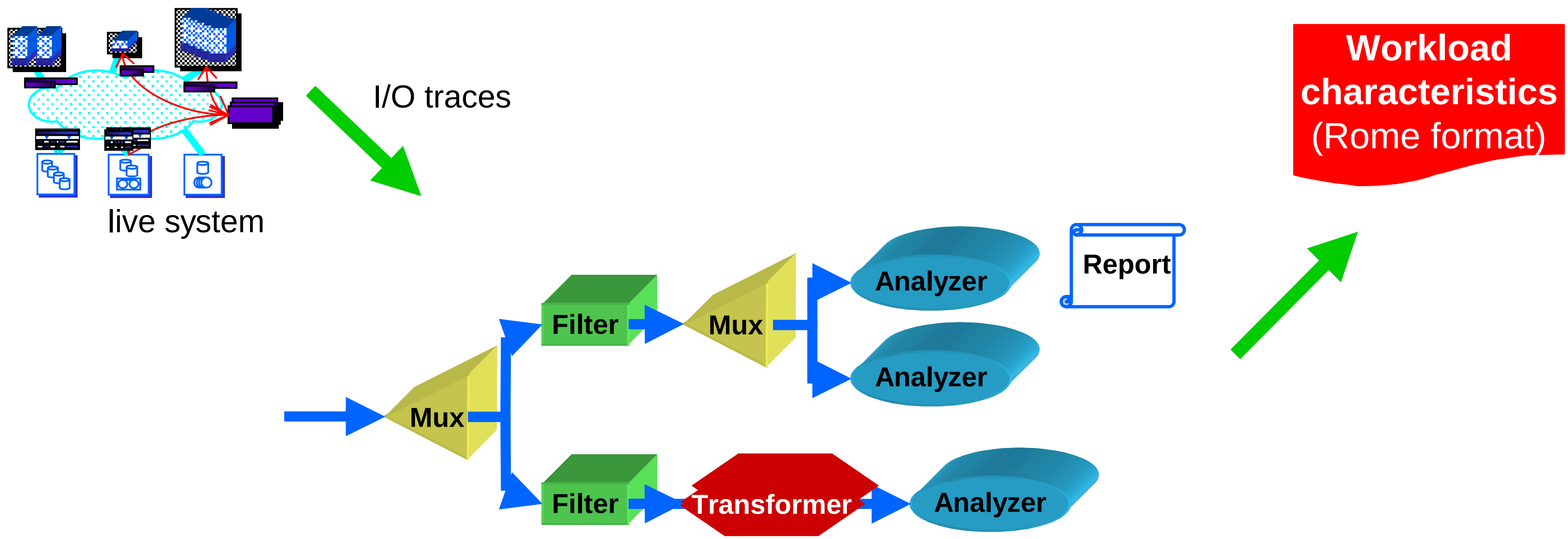
Workload modeling - attributes

- **Workload** = set of **stores** + **streams**
  - **stores** – static requirements (e.g. capacity)
  - **streams** – dynamic workload (e.g. bandwidth)

```
Store store0 {  {capacity 1e9 (bytes)} }
Stream stream0 {
    {boundTo store0}
    {requestRate  {ARW 800 600 200}  (request/sec)}
    {requestSize  {ARW 4096 4096 4096}  (bytes)}
    {sequentialRunCount {mean-variance 20 5} (reqs)}
    # phasing (correlation) behavior
    {onTime 90 (seconds)} {offTime 99 (seconds)}
    {overlapFraction { {stream1 1.0} {stream2 0.0}}}
```

Workload modeling – Rubicon

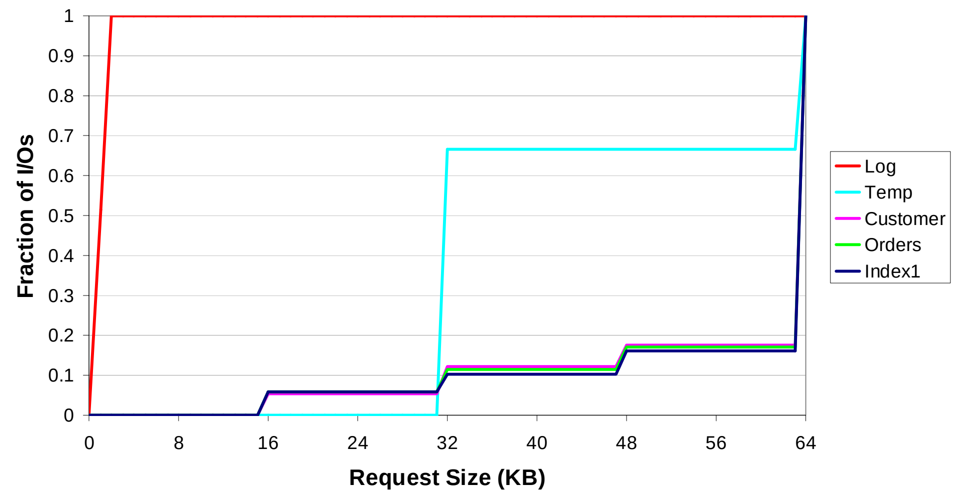
- **Workload characterization**
  - evaluate requirements and behaviors of applications
- **Monitoring and tuning**
  - spot bottlenecks and evaluate design changes



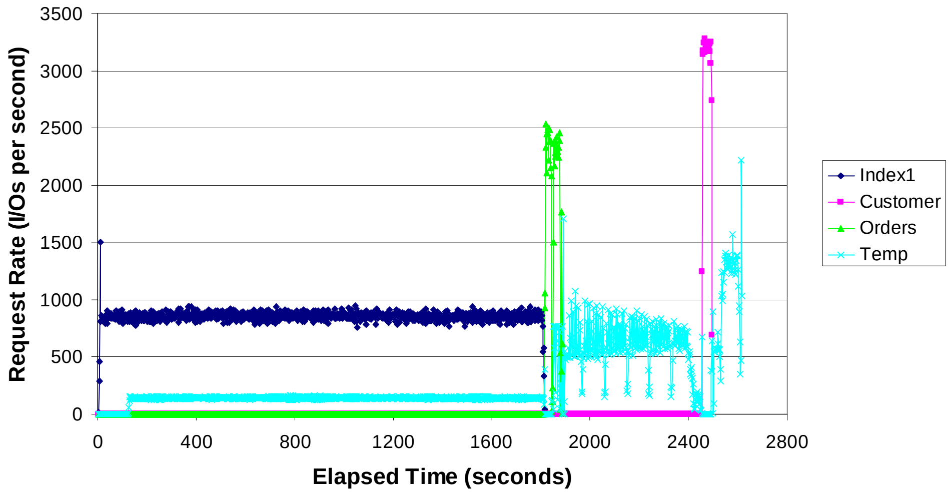
Rubicon

Workload modeling – case study

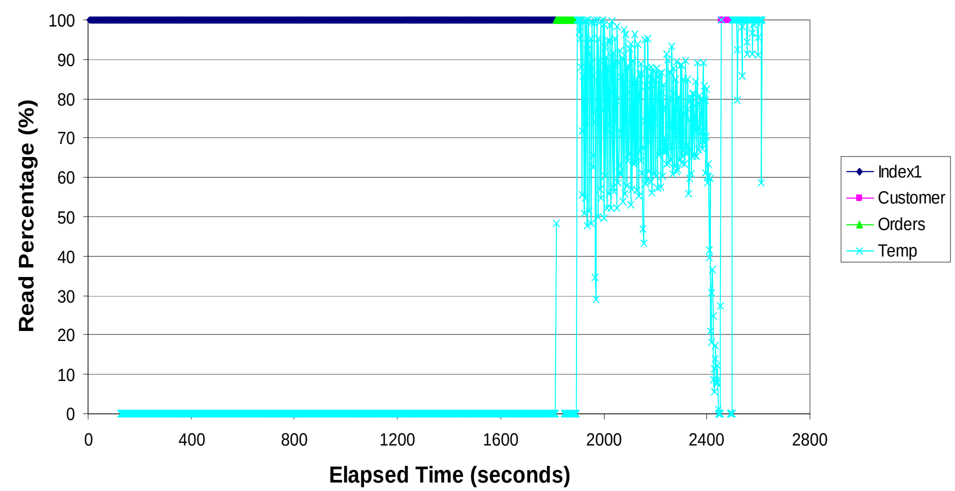
- Decision support (DSS)
  - Oracle
  - 300 GB TPC-D database
  - example data: TPC-D Q5



request size varies across tables, indices, logs, temporary space



queries operate in phases, with widely varying request rates



"read-only" workload does writes too!

Workload modeling – lessons learned

- **Lessons learned**
  - list of important characteristics is longer than you think
  - distributions, not averages, are important
- **Some characteristics of interest**
  - request size dist, request rate dist, read:write ratio
  - spatial locality (esp. sequentiality), temporal locality
  - phasing & correlation behavior
- **Open questions**
  - what characteristics needed when
    - for workload regeneration, QoS specs, performance prediction
  - modeling the scaling behavior of applications
    - e.g. changes with number of users, size of database
  - semantic mapping between app and storage requirements?
    - how to know when you're doing better

Workload modeling – related work

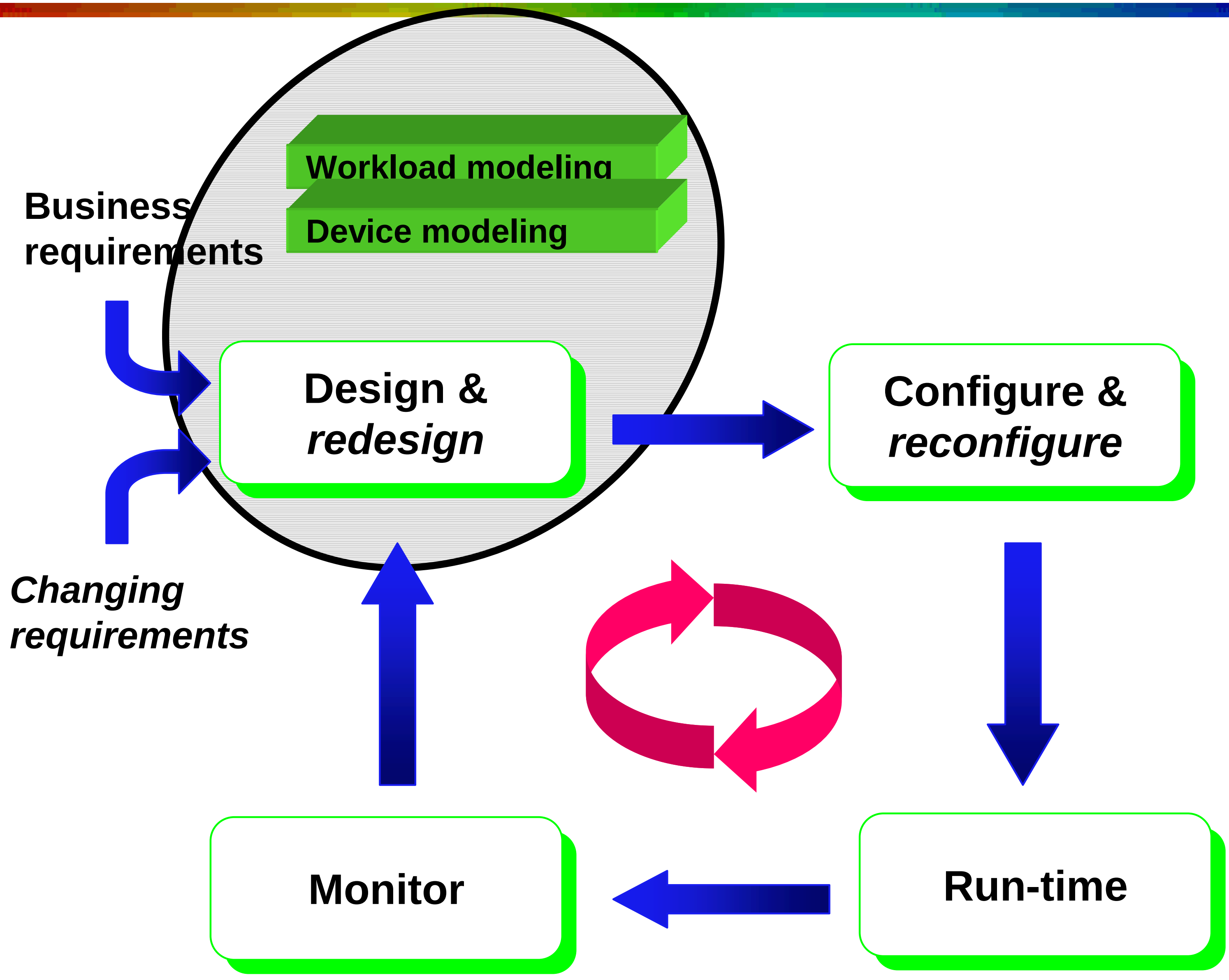
Workload characterization case studies

- File system tracing
  - [Ousterhout85, Miller91, Ramakrishnan92, Baker91, Gribble98]
- Network tracing
  - [Caceres91, Paxson94, Paxson97]
- I/O tracing
  - [Bates91, Ruemmler93, Gomez98, Hsu99]

Tools

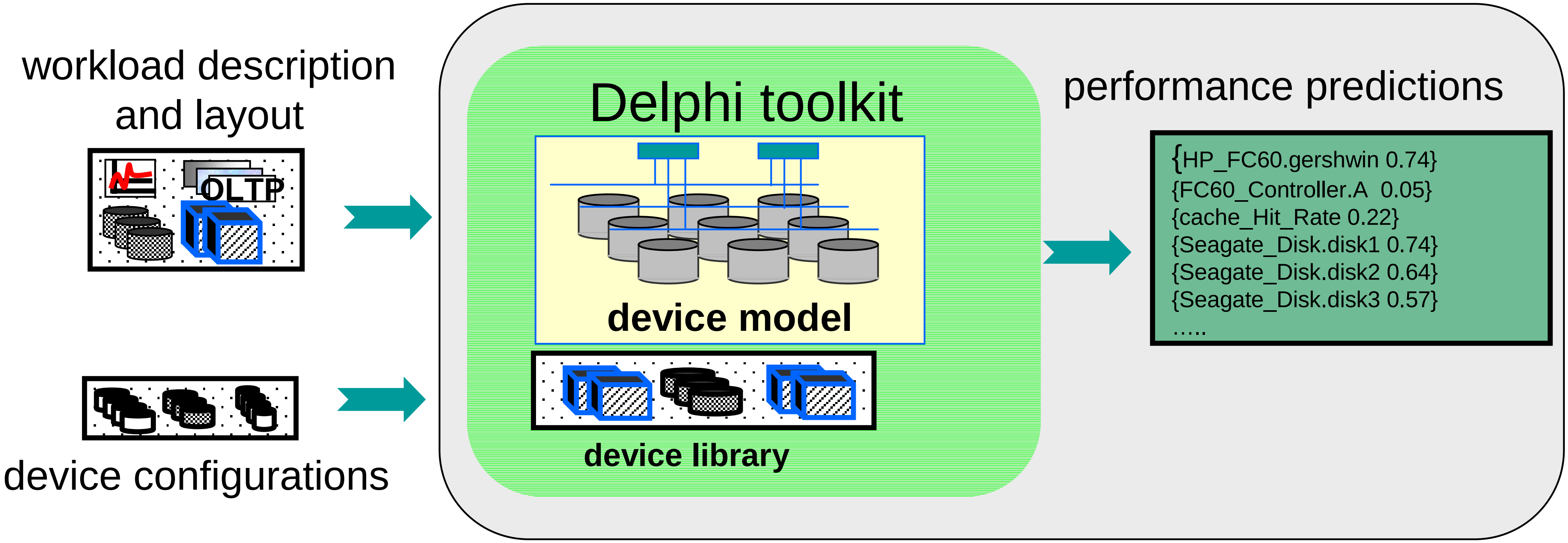
- Offline trace gathering, analysis and visualization
  - [Grimsrud95, IBM99]
- Extensible trace analysis
  - Tramp [Touati91]
- Network packet filters
  - [Mogul87, McCanne93]
- Trace visualization
  - [Heath91, Malony91, Hibbard94, Eick96, Aiken96, Livny97]

Automatic-management lifecycle



Device modeling - Delphi toolkit

- Fast, detailed and robust analytical models
  - disks, raid controllers, caches
  - incremental model evaluation
- Quickly build models for variety of architectures
  - modular, flexible toolkit
  - reuse components and device calibrations



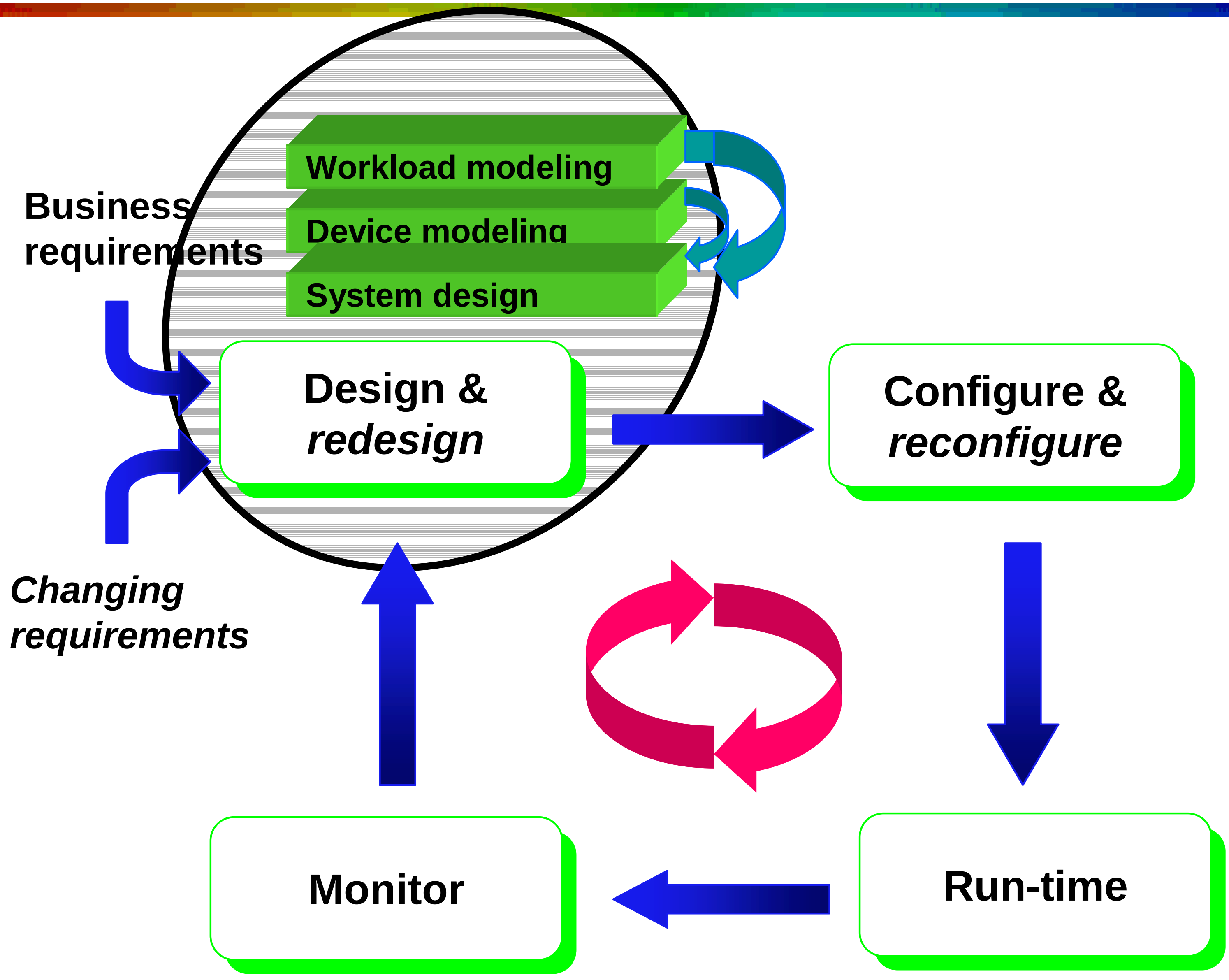
Device modeling - lessons learned

- **Lessons learned**
  - worry about tradeoff between accuracy and performance
    - for simulations (high accuracy)
    - as input to optimization steps (high performance)
    - solution - set of increasing fidelity device models
  - need a tool to automatically extract model parameters
    - on a per-device basis
  - modular design to maximize re-use
- **Open questions**
  - can we continue to ignore host/server behavior in models
    - hardware path, operating system effects
  - how can we model very complex workload characteristics
    - e.g. fractal characteristics
  - how to incorporate performability

Device modeling – related work

- **Ruemmler and Wilkes, 1993**
  - accurate disk drive simulation model – prioritized components
  - detailed characteristics for two disk drives
- **Worthington, et al., 1995**
  - black-box techniques for extracting SCSI disk parameters
- **Shriver, et al., 1997**
  - disk drive model by composing models of individual components
  - performance prediction depends on input workload and predictions of lower-level models
- **Pythia [Pentakalos, et al., 1997]**
  - automatically builds and solves analytic model of storage system
  - inputs: graphical representation of system and workload
  - Pythia/WK: uses clustering algorithms to characterize workloads
- **Disk arrays**
  - [Thomasian94 , Merchant96, Menon97]

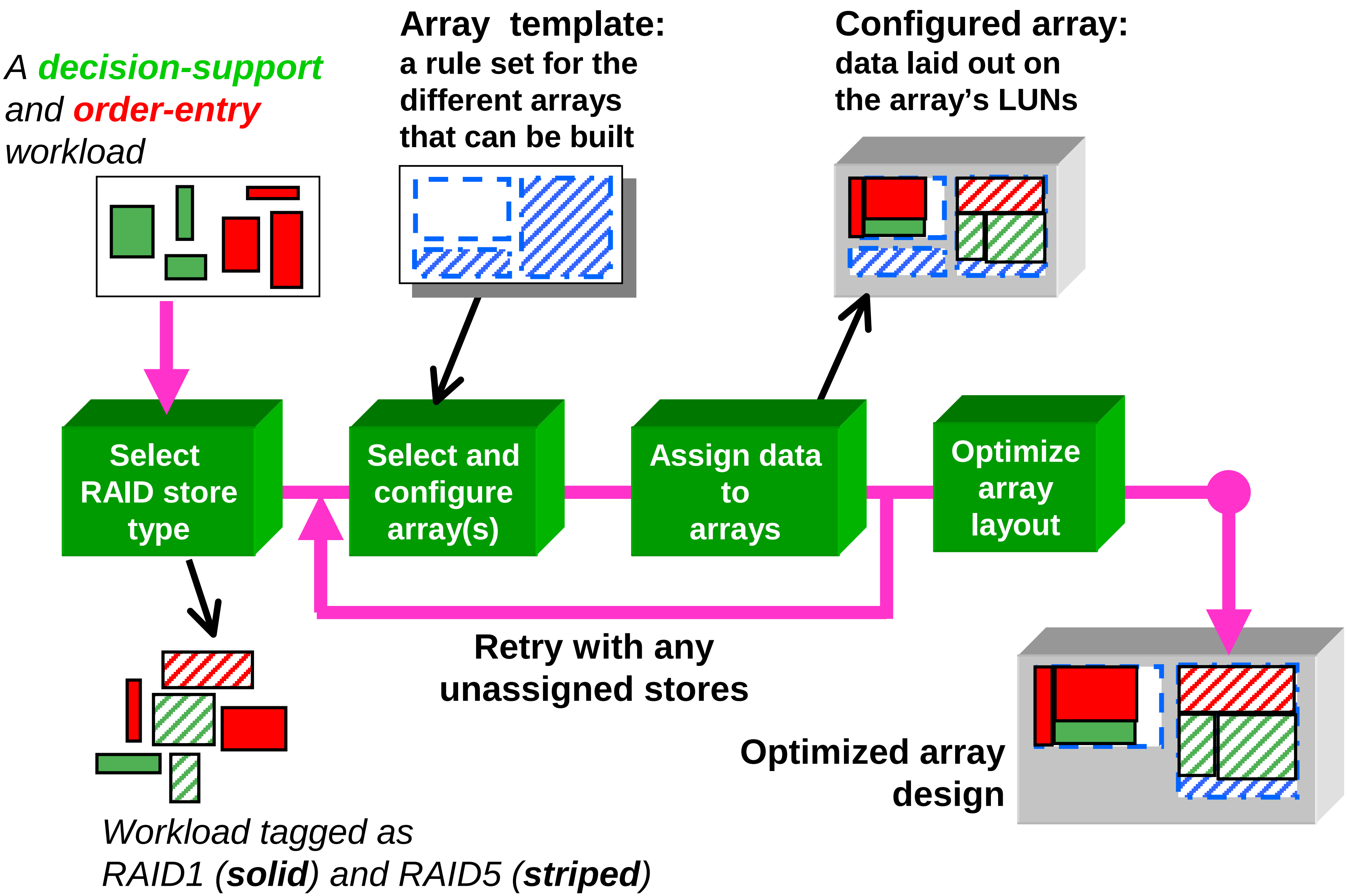
Automatic-management lifecycle



System design - solver basics

- Concise workload characterization
  - library of models for common workload types
  - automatically characterized from running workload
- Fast, acceptable-fidelity device models
  - executed in inner loop of optimizer
  - library of storage device models & characterizations
- Solver
  - constraint-based, multiple-dimensional bin-packing
- Search-space exploration algorithms
  - heuristics for trying “what ifs?”
    - good news: simple ones work well
  - utility-based objectives, modulated by business goals
    - minimum cost, maximum availability, balanced load, …

System design - optimization



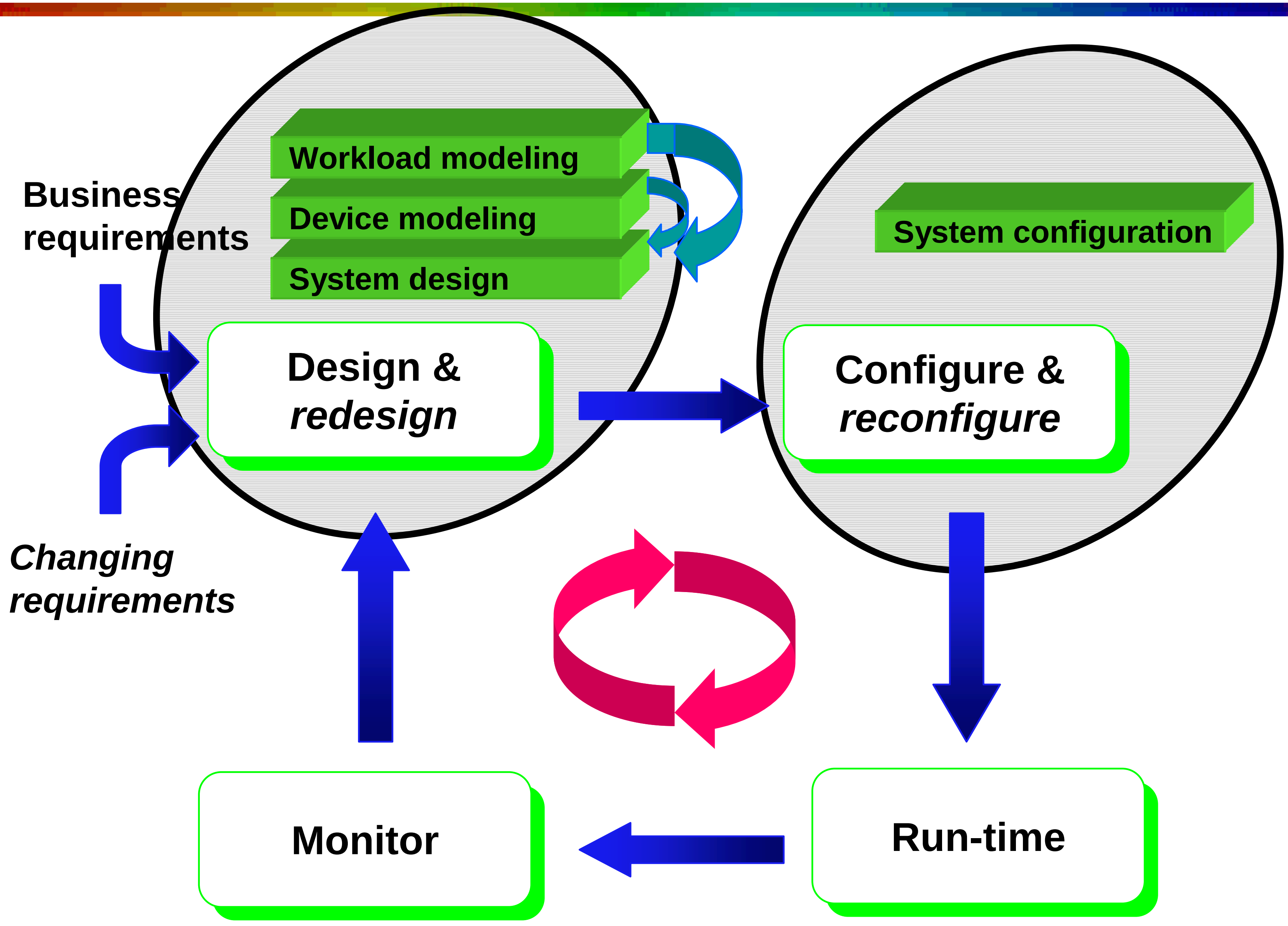
System design - lessons learned

- **Lessons learned**
  - can meet/beat human-created designs, fully automatically
    - example DSS system is 30% cheaper
  - search problem tractable with simple heuristics
    - e.g. greedy search
- **Open questions**
  - optimal vs. adequate - when to quit
  - what objectives and constraints work best
    - e.g. cost of reconfiguring system
  - generalizing system design
    - for network environment - separate SAN design work
    - to include host and applications
      - currently assumed to be unchangeable
      - no feedback loop to application behavior

System design - related work

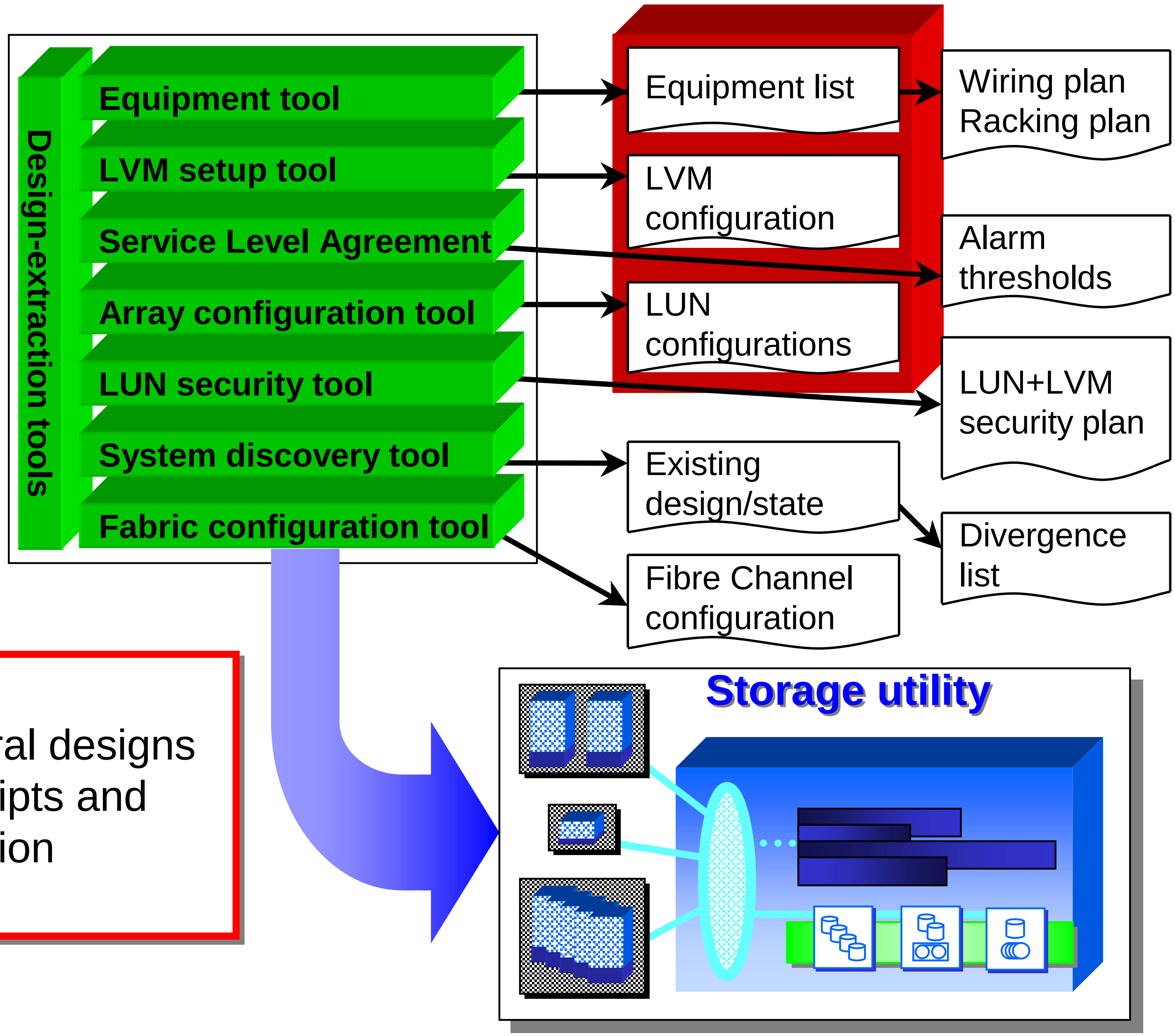
- Storage management [Gelb89]
  - Logical view of data separate from physical device characteristics – simplifies management
- File assignment problem
  - Files placed on devices by optimizing objective(s)
  - [Dowdy82, Wolf89, Pattipati90, Awerbuch93]
- Optimization algorithms
  - Bin-packing heuristics [Coffman84]
  - Toyoda gradient [Toyoda75]
  - Simulated annealing [Drexl88]
  - Relaxation approaches [Pattipati90, Trick92]
  - Genetic algorithms [Chu97]

Automatic-management lifecycle



Configuration - research challenges

New design



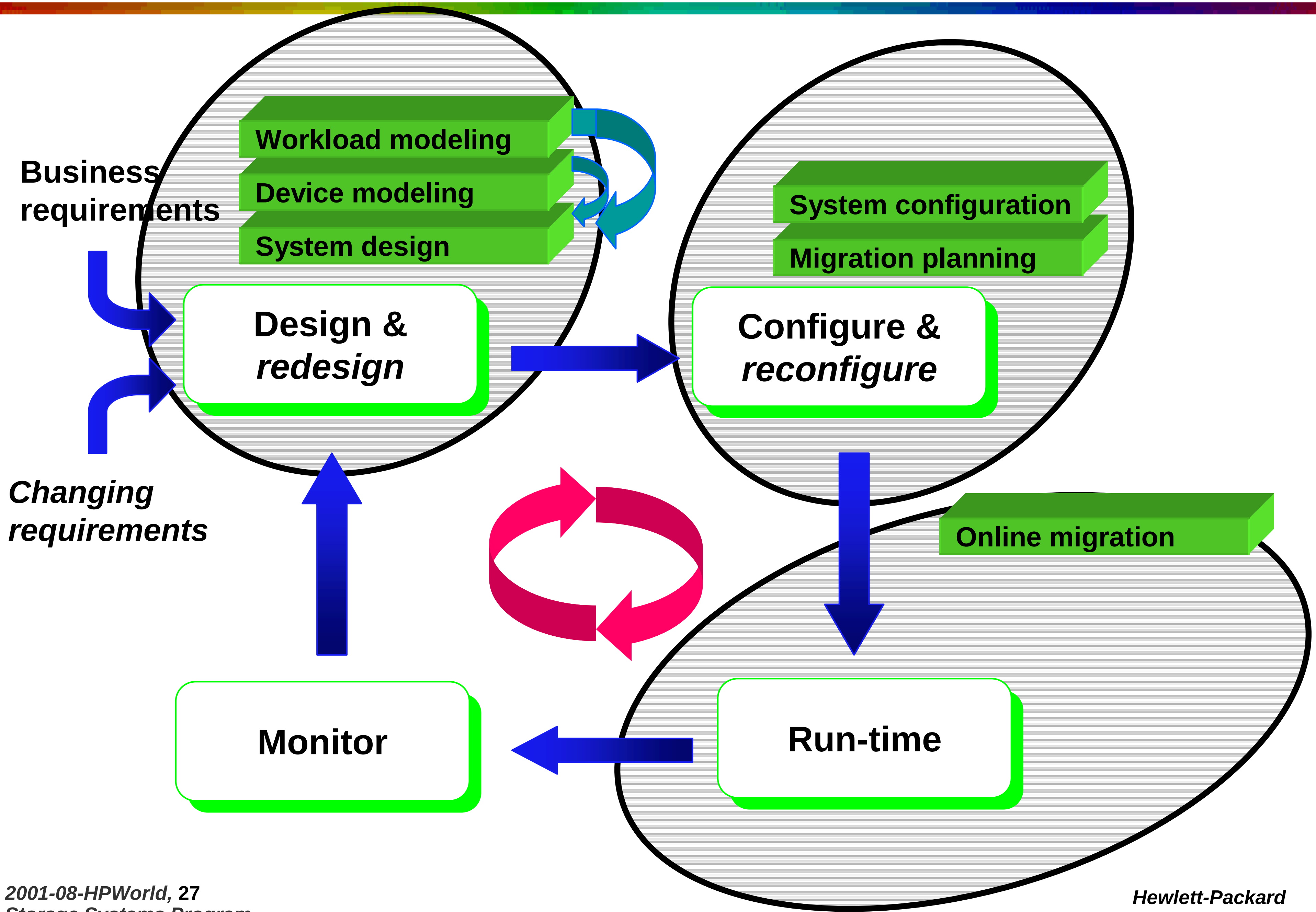
Panopticon

- converts OS-neutral designs into OS-specific scripts and performs configuration commands

Configuration - issues

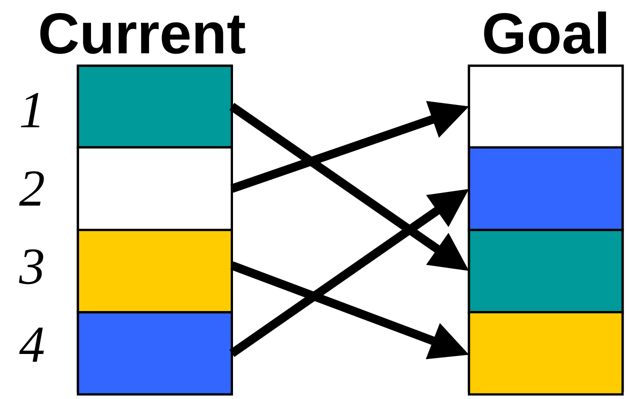
- How to do system discovery
  - e.g. existing state, presence of new devices
  - dealing with inconsistent information
  - in a scalable fashion
- How to abstractly describe storage devices
  - for system discovery output
  - for input to tools that perform changes
  - across vendors, across operating environments
- How to automate the physical design process
  - e.g. physical space allocation, wiring, power, cooling

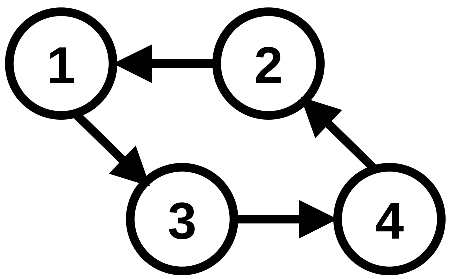
Automatic-management lifecycle



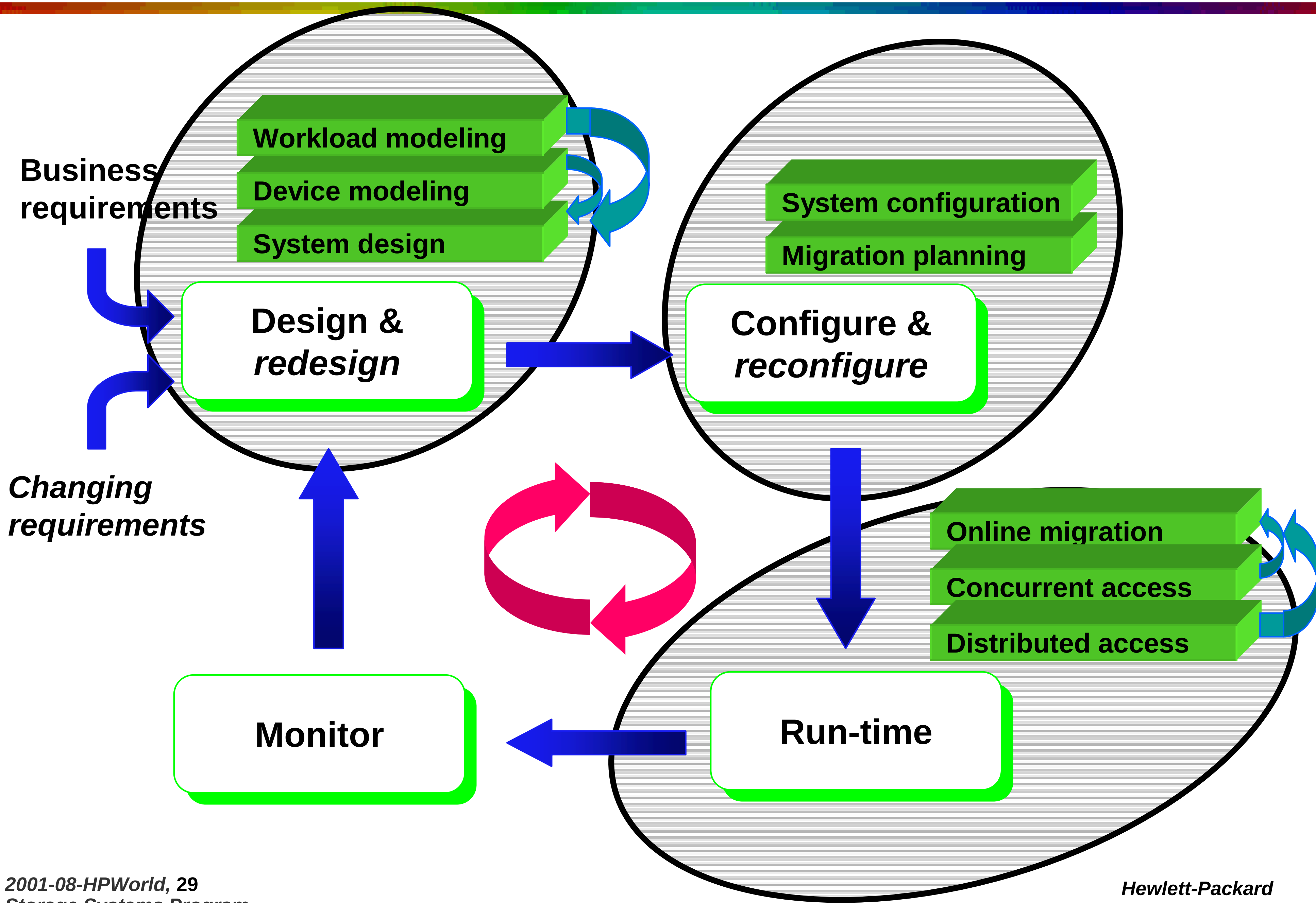
Migration - research challenges

- Move system to new configuration
  - may require moving data
  - may require changing configurations
- Build a migration plan
  - generalize for variable-sized data
  - allow parallel execution
  - determine required free space
  - plan for data movement with constraints
    - e.g. capacity, performance, availability
- Perform migration – *online,* continue normal service
  - help from runtime system
    - virtualization & device hooks
  - design must optimize for performance during migration





Automatic-management lifecycle



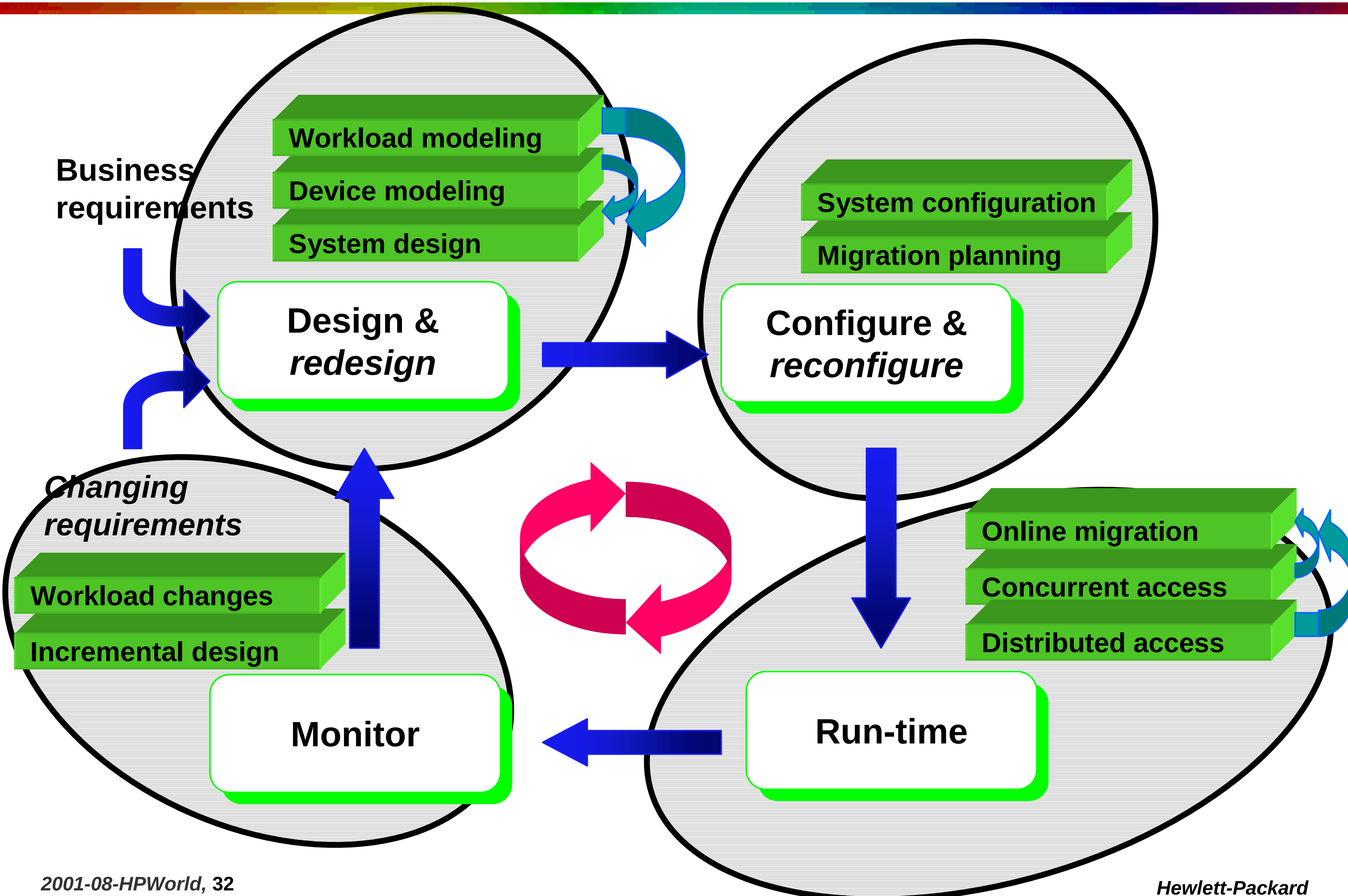
Runtime system - research challenges

- Ensuring metadata is always available
  - even in the face of network partitioning [Golding99]
- Managing concurrency at the large scale
  - optimistic concurrency control protocols [Amiri00]
- Enforcing security in a multi-host environment
  - must to be done directly at storage device in a shared-resource environment
  - Carnegie Mellon NASD [Gobioff99, Gibson98]
- QoS enforcement
  - how should these be specified?
  - what portions should be enforced by which component?
  - how can violations be detected? handled?
  - [Golubchik99, Bruno99, Wijayaratne00]

Runtime system - related work

- CMU network-attached disks
  - disks present file-like objects
  - many disks aggregated to make system
  - [Gibson97, Gibson98]
- Distributed storage service
  - MIT Logical disks [deJonge93]
  - Compaq/DEC SRC Petal [Lee96]
  - U of Arizona Swarm [Hartman99]
- Distributed file systems
  - CMU Andrew FS [Howard88]
  - Berkeley Zebra [Hartman93]
  - Berkeley xFS [Anderson95]
  - Compaq SRC Frangipani (FS for Petal) [Thekkath97]

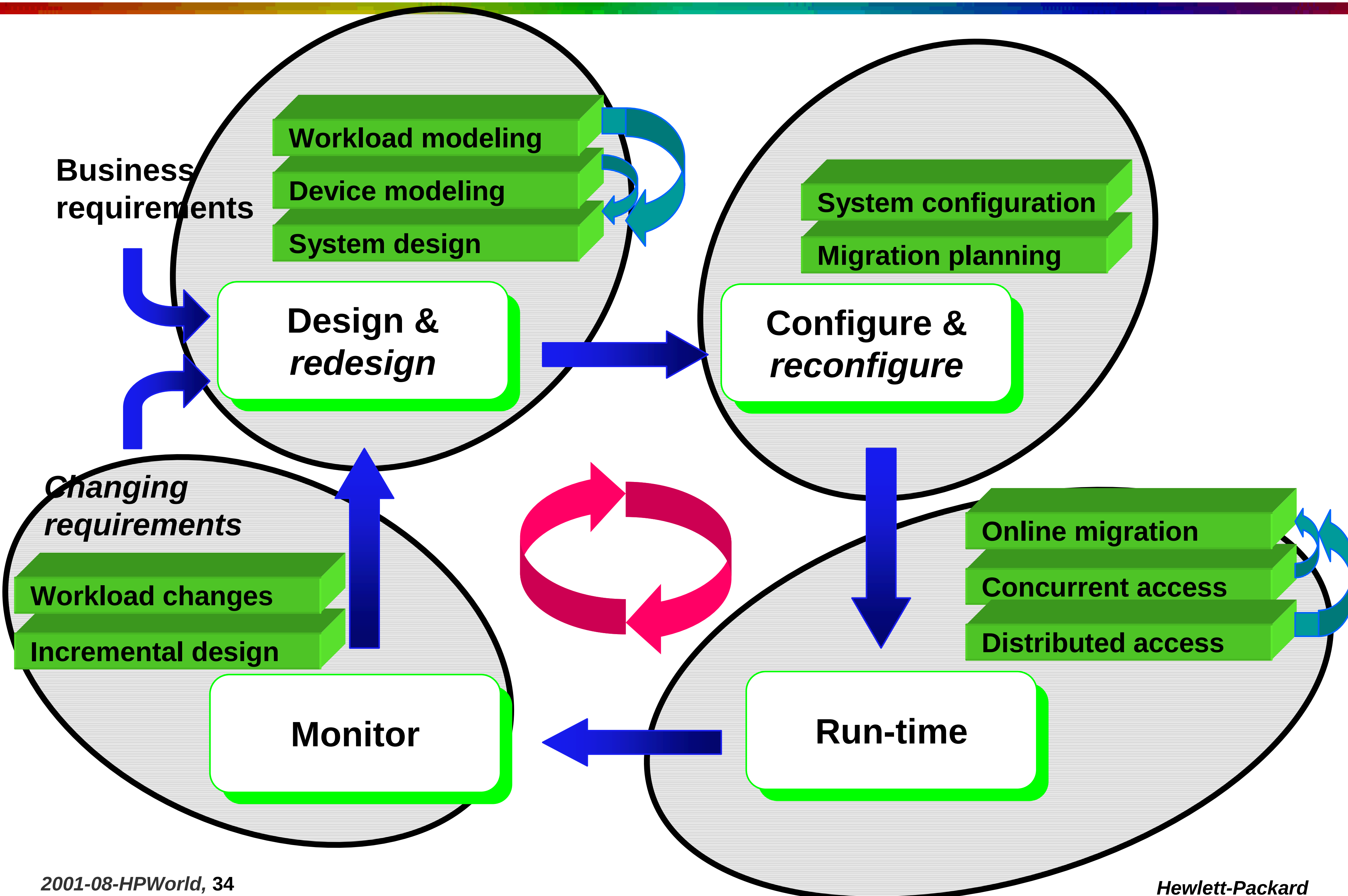
Automatic-management lifecycle



Monitoring - research issues

- What quantities must be monitored
  - to detect component failures
  - to detect performance bottlenecks
  - to enforce QoS requirements/detect QoS violations
  - to detect performance trends
- How to monitor in a scalable fashion
- How to monitor in a flexible fashion
  - e.g. attributes that are specific to one type of device
- How to translate between levels of abstraction
  - e.g. LUNs vs. logical volumes vs. database tables

Automatic-management lifecycle

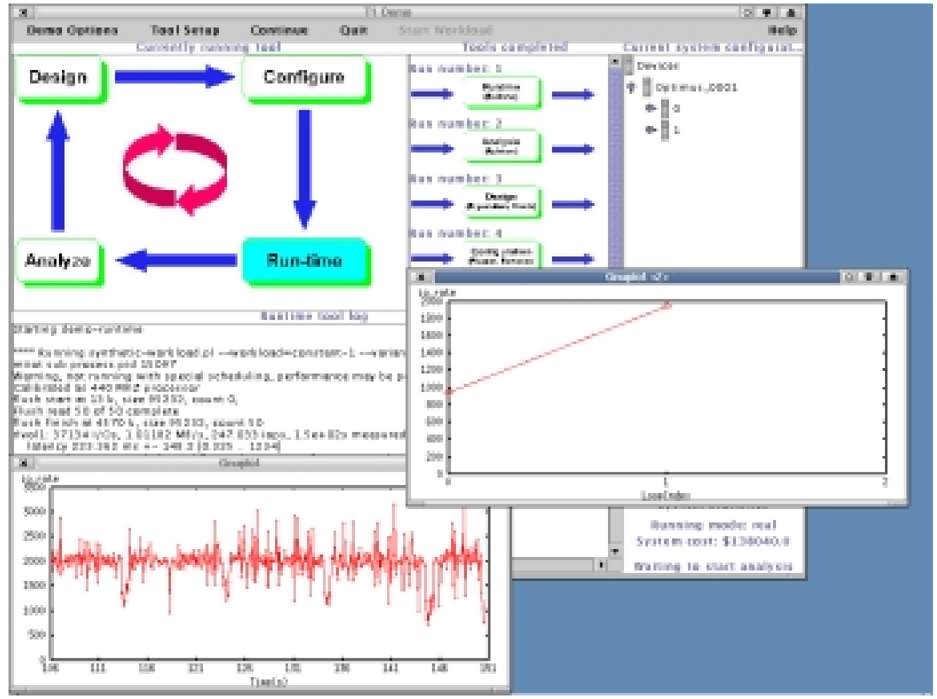


Talk overview

- Introduction
- Our vision - self-managing storage
- Research challenges
- Prototype
  - closing the loop for a complete system
- Conclusions
- Future

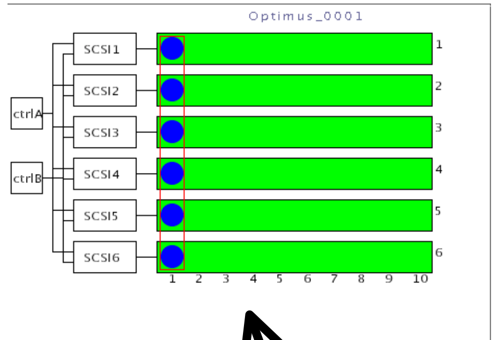
Prototype - tying it all together

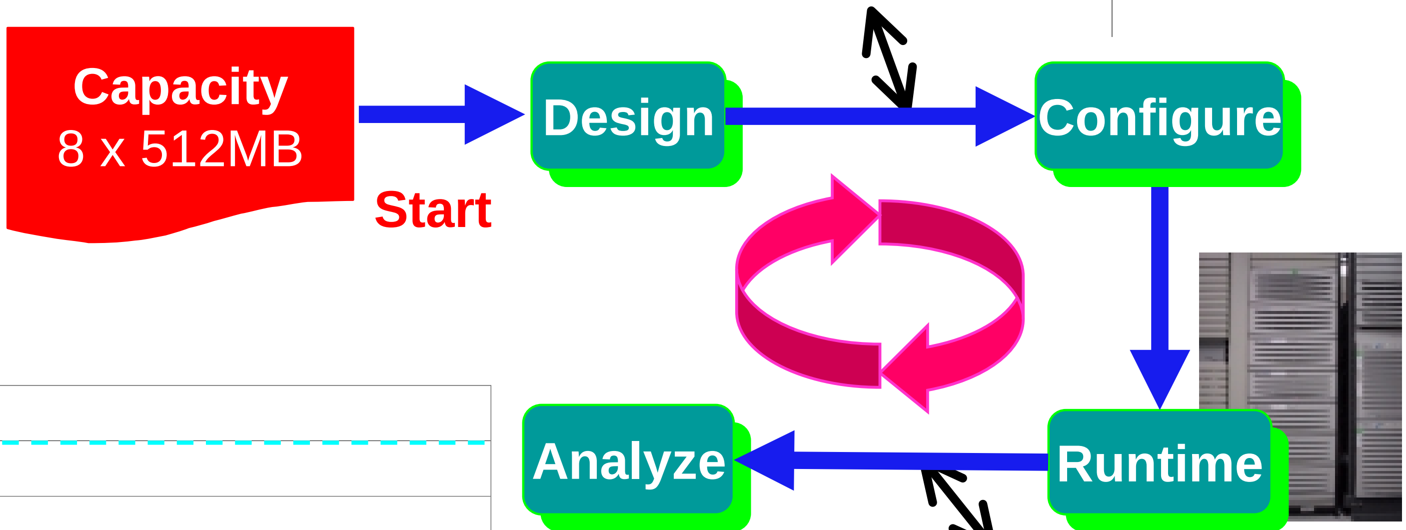
- User interface
  - *observe* loop steps
  - fully automatic
  - no user input required
- Demo
  - midrange system
  - multiple configurations with varying performance

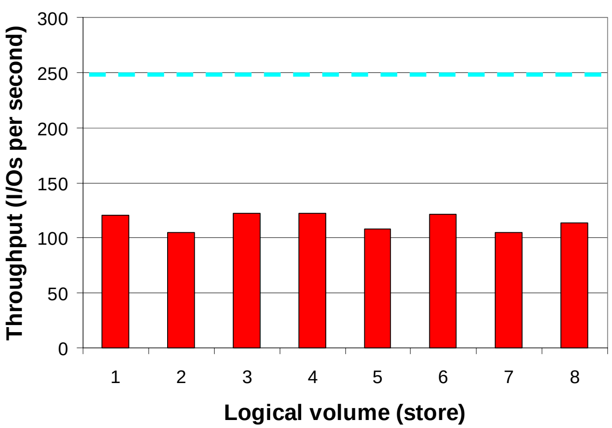


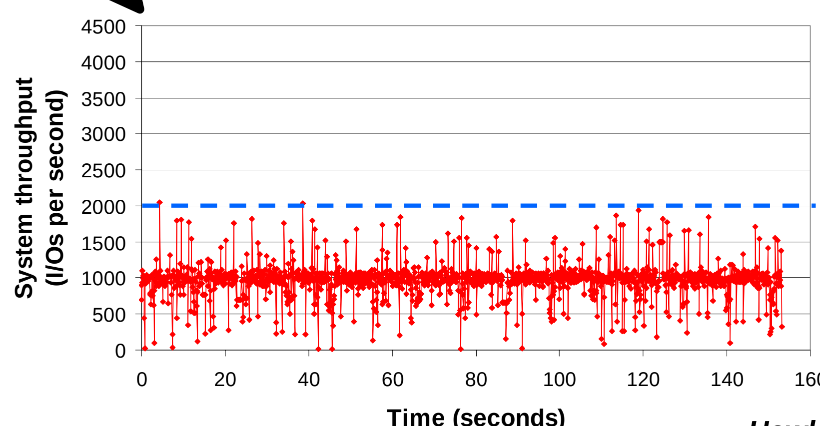
- Hardware
  - N-class host
  - up to 8 volumes, 250 IO/s
  - fibre channel SAN
  - single FC-60 disk array
  - up to 24 disk drives

Prototype - first loop iteration

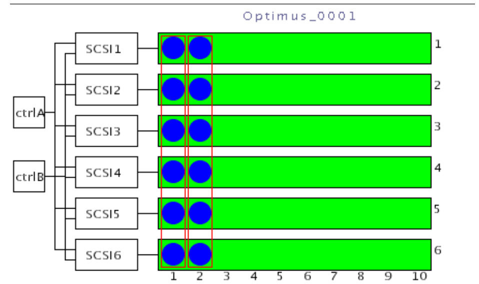


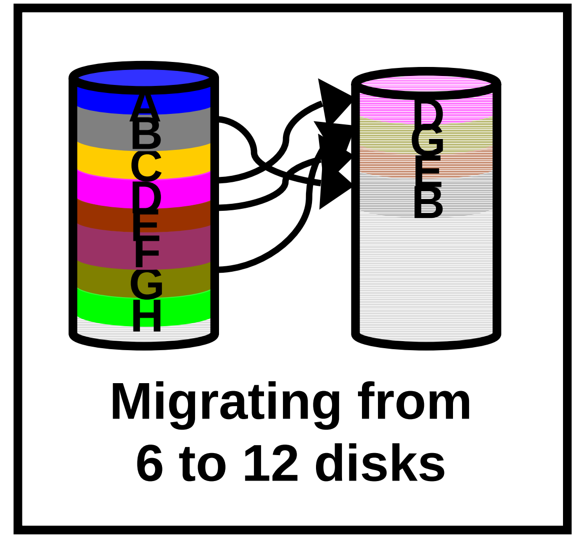


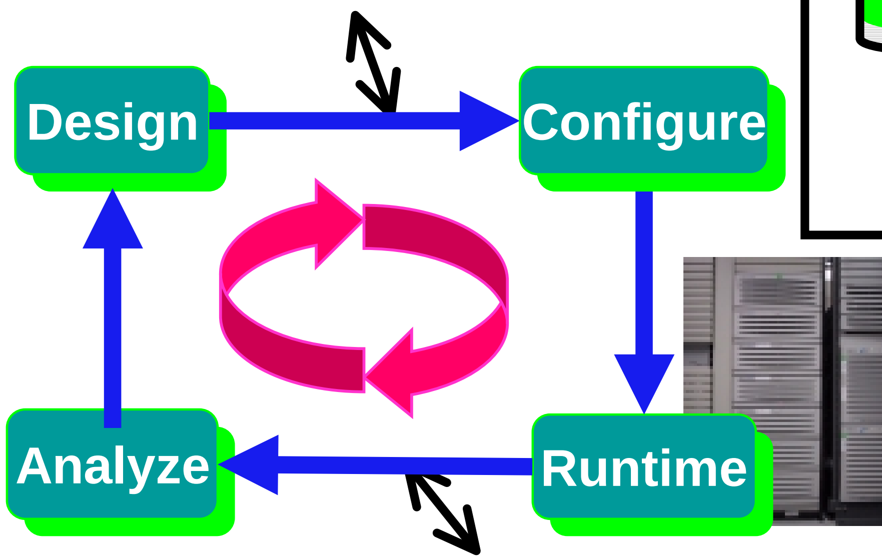


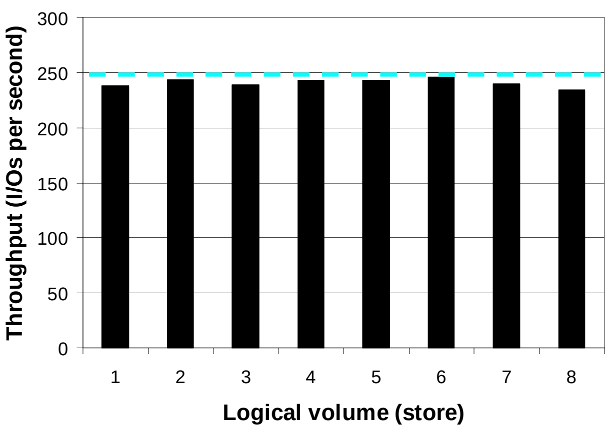


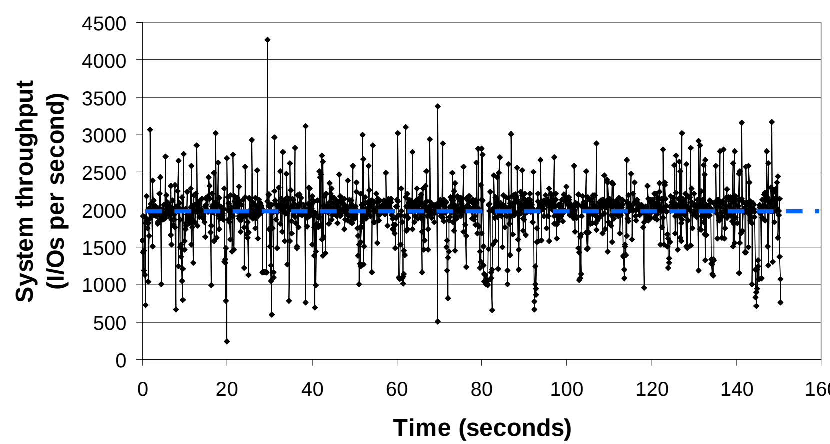
Prototype - second loop iteration





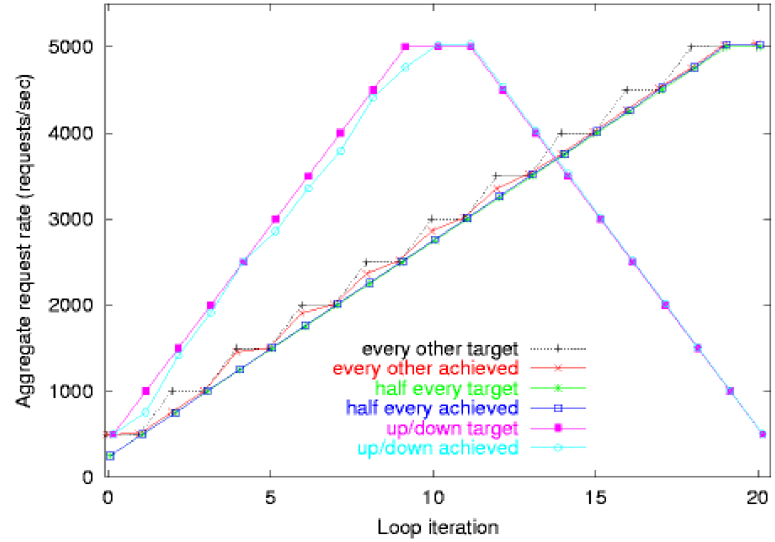




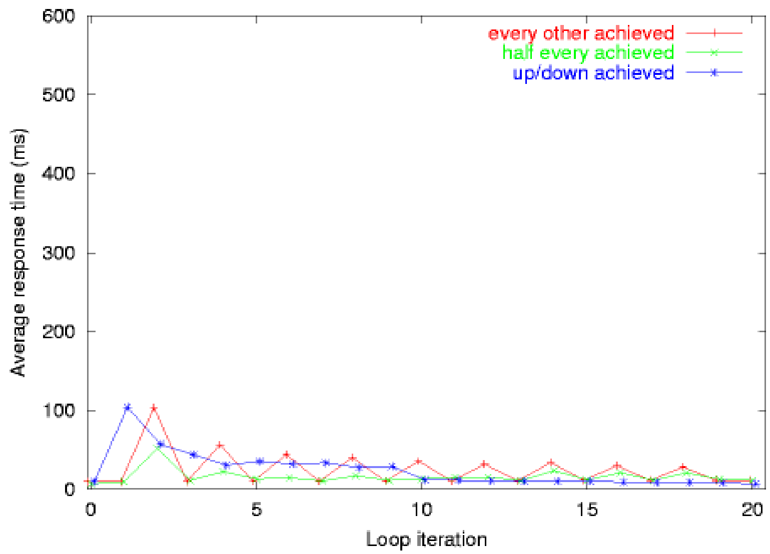


System evolution - synthetic workload

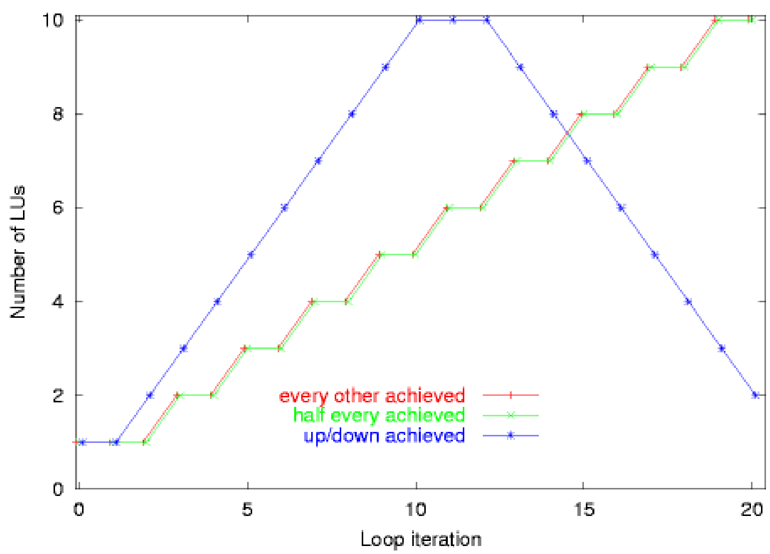
- start with small workload
- workload changes over time
- takeaways
  - fast tracking
  - minimal changes
  - scale appropriately



request rate tracks workload



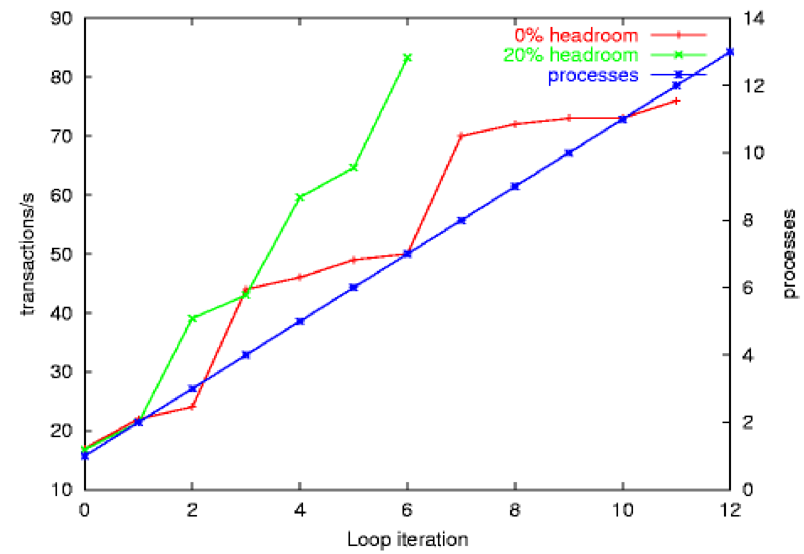
latency remains low across changes



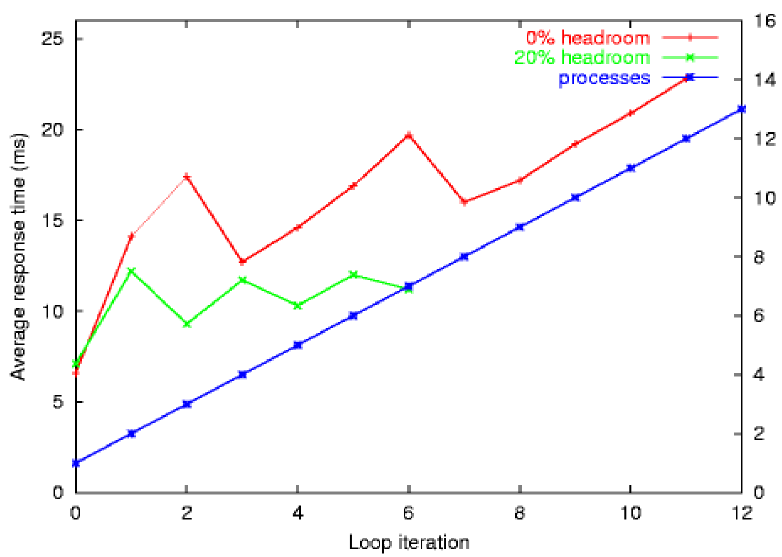
resources increase to meet demands

System evolution - filesystem benchmark

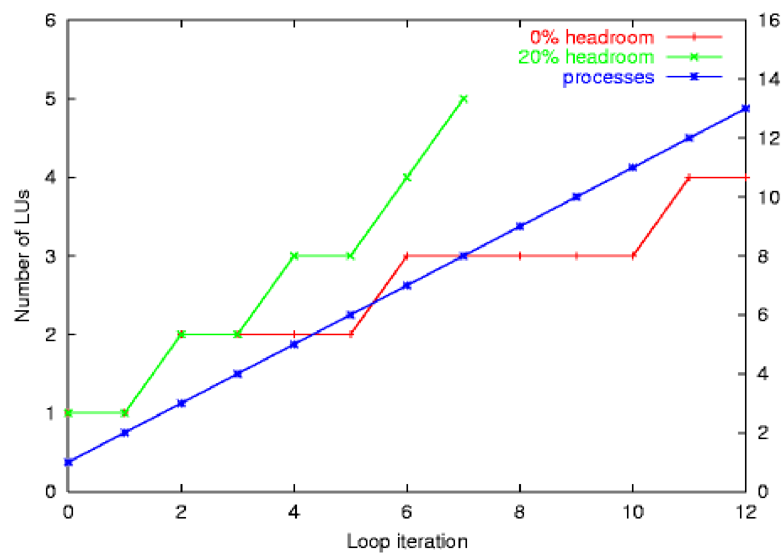
- ***Postmark* benchmark**
- **scale number of processes**
- **takeaways**
  - **realistic workload**
  - **reasonably fast tracking**
  - **some lag w/o headroom**



request rate increases



latency remains reasonable



resources increase to meet demands

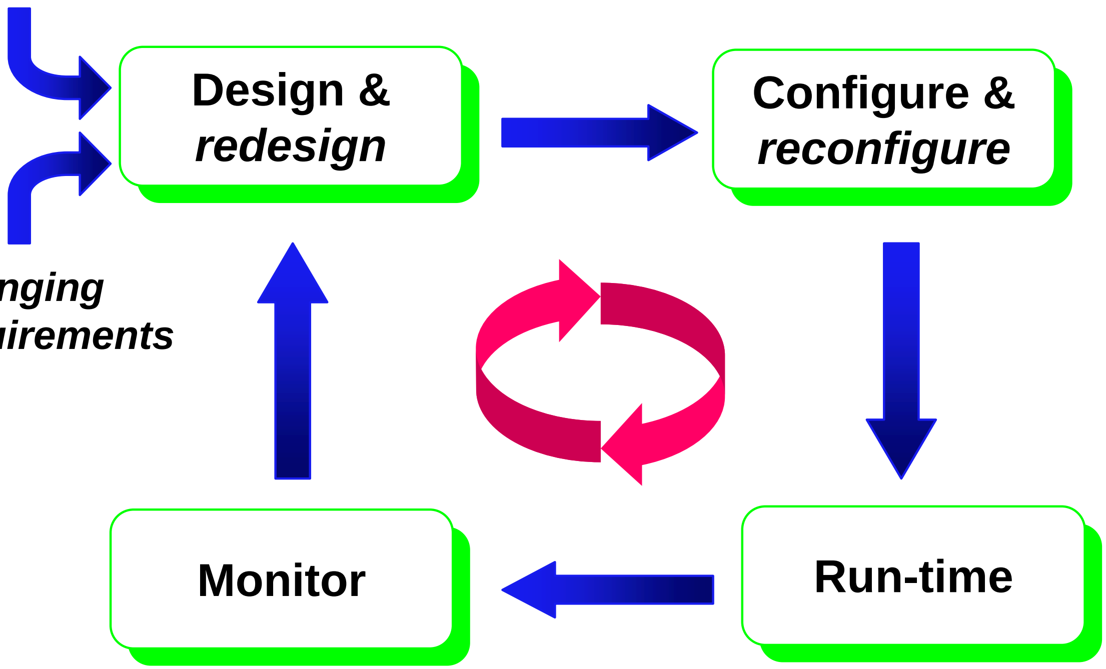
Talk overview

- Introduction
- Our vision - self-managing storage
- Research challenges
- Prototype
- Conclusions
  - self-management works!
- Future

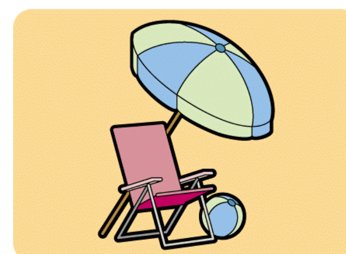
Conclusions

- Review of the big ideas
  - goal-directed self-management really works
  - automatic design and configuration
  - automatic redesign and reconfiguration
  - software & monitoring as the key differentiators
  - predictable behavior through guarantees

Business requirements



Changing requirements

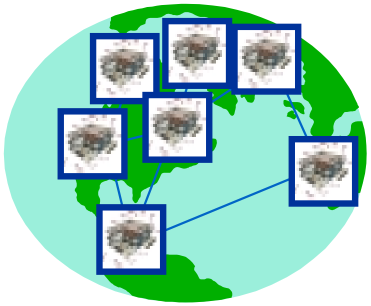


stress-free storage

reduced people-print

The Future

- Extend work to the global scale
  - global data placement
    - workload-optimized placement decisions
    - adaptive consistency
    - highly-distributed security



- Make storage ubiquitous
  - horizontal scaling with storage bricks
    - highly integrated devices - compute & storage
    - small form factor, ubiquitous storage
    - data shadow follows users and uses

Acknowledgements

- SSP: Guillermo Alvarez, Eric Anderson, Sandra Barreto, Michael Hobbs, Mahesh Kallahalla, Kim Keeton, Arif Merchant, Cristina Solorzano, Susan Spence, Ram Swaminathan, Simon Towers, Mustafa Uysal, Alistair Veitch, Qian Wang, John Wilkes
- ex-SSP: Ralph Becker-Szendy, Liz Borowsky, Susie Go, Richard Golding, David Jacobson, Ted Romer, Chris Ruemmler, Mirjana Spasojevic

- To learn more
  - **www.hpl.hp.com/SSP**

References – workload characterization

- Workload characterization
  - [Ousterhout85], [Mogul87], [Baker91] – SOSP
  - [Miller91] – IEEE Mass Storage
  - [Ramakrishnan92], [Gribble98] – ACM SIGMETRICS
  - [Caceres91], [Paxson94] – ACM SIGCOMM
  - [Paxson97] – ACM Transactions on Networking
  - [Bates91] – *VAX I/O Subsystems*
  - [Ruemmler93], [McCanne93], [Roselli00] – USENIX
  - [Gomez98] – Workshop on Workload Characterization
  - [Hsu99] – UC Berkeley Tech Report
  - [Grimsrud95] – IEEE Transactions on Computers
  - [Touati91], [Eick96] – IEEE Software Practice & Experience
  - [Heath91], [Malony91] – IEEE Software
  - [Hibbard94] – IEEE Computer
  - [Aiken96] – Int'l Conference on Data Engineering

References – device modeling

- Device modeling
  - [Ruemmler93] - USENIX
  - [Worthington95], [Shriver97] – ACM SIGMETRICS
  - [Shriver97] – PhD thesis, New York University
  - [Ganger95] – PhD thesis, University of Michigan
  - [Pentakalos97] – IEEE Software Practice & Experience
  - [Thomasian94] – ICDE
  - [Merchant96] – IEEE Transactions on Computers
  - [Menon97] – ICDCS

References – system design & allocation

- **System (re)design and allocation**
  - **[Borowky98] – Workshop on Software and Performance**
  - **[Gelb89] – IBM Systems Journal**
  - **[Dowdy82] – ACM Computing Surveys**
  - **[Wolf89] – ACM SIGMETRICS**
  - **[Pattipati90] – ICDCS**
  - **[Awerbuch93] – ACM STOC**
  - **[Coffman84] – in *Algorithm Design for Computer System Design***
  - **[Toyoda75] – Management Science**
  - **[Drexl88] – Computing**
  - **[Trick92] – Naval Research Logistics**
  - **[Chu97] – Computers and Operations Research**

References – monitoring & runtime

- **Online monitoring**
  - [Miller95] – IEEE Computer
  - [Reed93] – IEEE Scalable Parallel Libraries Conf.
- **Runtime & distributed file system**
  - [Lee96], [Gibson98] – ASPLOS
  - [Gobioff99] – PhD thesis, Carnegie Mellon University
  - [Golding99] – Symposium On Reliable Distributed Systems
  - [Borowsky97] – Int'l Workshop on Quality of Service
  - [Bruno99], [Golubchik99] - IEEE Int'l Conf. on Multimedia Computing
  - [Wijayaratne00] – Multimedia Systems
  - [Gibson97] – SIGMETRICS
  - [deJonge93], [Anderson95], [Thekkath97] – SOSP
  - [Hartman99], [Amiri00] – ICDCS
  - [Howard88] – ACM Transactions on Computer Systems

References – smart devices & availability

- Device intelligence
  - [Wilkes92] – USENIX Workshop on File Systems
  - [Cao94] – ACM Transactions on Computer Systems
  - [Wang99] – Usenix OSDI
  - [Keeton98] – ACM SIGMOD Record
  - [Riedel98] – VLDB
  - [Acharya98] – ASPLOS
  - [Uysal00] – HPCA
  - [Riedel00] – ACM SIGMOD
  - [Lumb00] – Usenix OSDI
  - [Riedel01] – IEEE Computer
- Describing manageability and availability
  - [Brown00] – USENIX Technical Conference

Sources for additional information

- Our web page – www.hpl.hp.com/SSP
- HP SureStore – www.hp.com/storage
- Storage Network Industry Assoc. – www.snia.com
- Disk/Trend – www.disktrend.com
- IDC – www.idc.com

- Tioga, *The Holy Grail of Data Storage Management*
- Farley, *Building Storage Networks*
- Gray & Reuter, *Transaction Processing*
- Bates, *VAX I/O Subsystems: Optimizing Performance*

i n v e n t