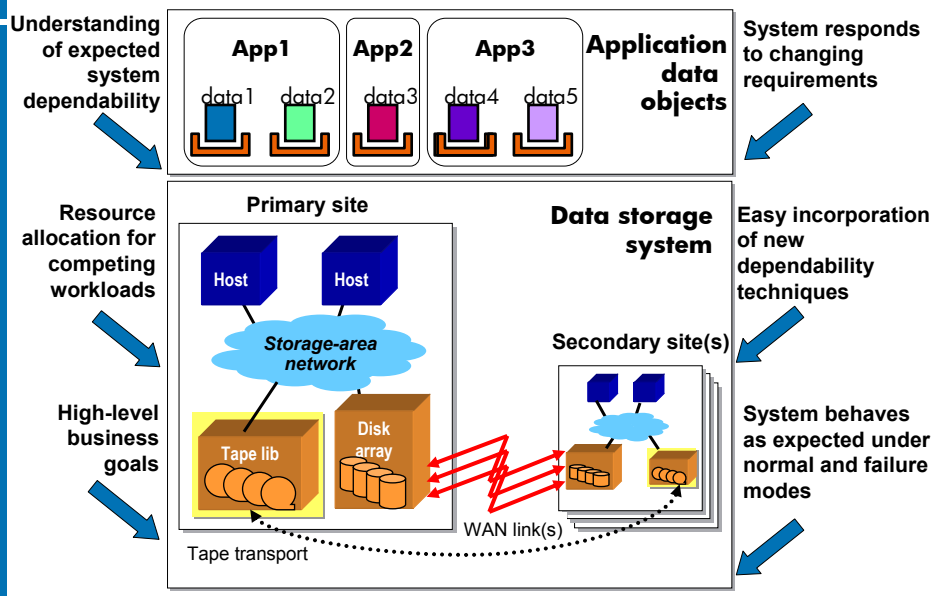
Automating data dependability

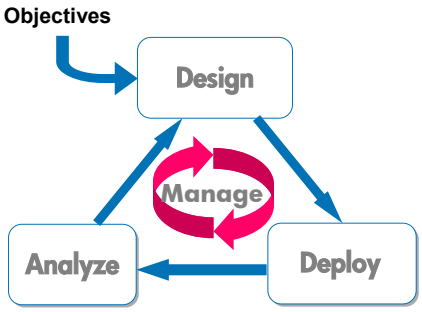
47th Meeting of IFIP WG 10.4 – "Autonomic Web Computing"
January 26-30, 2005
Rincon, Puerto Rico, USA

Kim Keeton, Dirk Beyer, Jeff Chase, Arif Merchant, Pano Santos and John Wilkes
Hewlett-Packard Labs and Duke University
{kimberly.keeton,dirk.beyer,arif.merchant,cipriano.santos,john.wilkes}@hp.com, chase@cs.duke.edu

Self-managing dependable data service



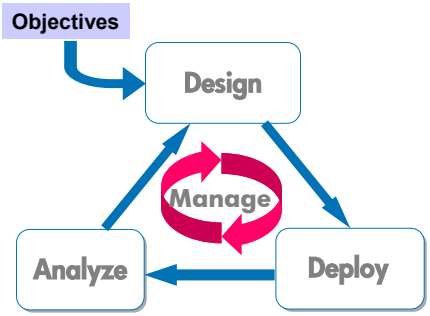
Automated data dependability



- Defining the desired level of service
- Designing
- Deploying
- Analyzing

} the system that powers the service

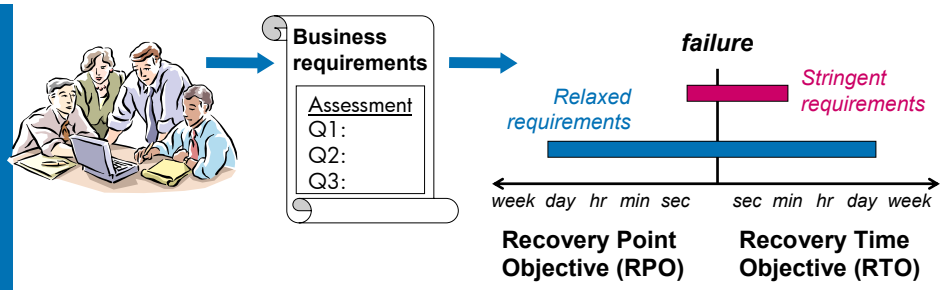
Outline



- Defining the desired level of service
- Designing
- Deploying
- Analyzing

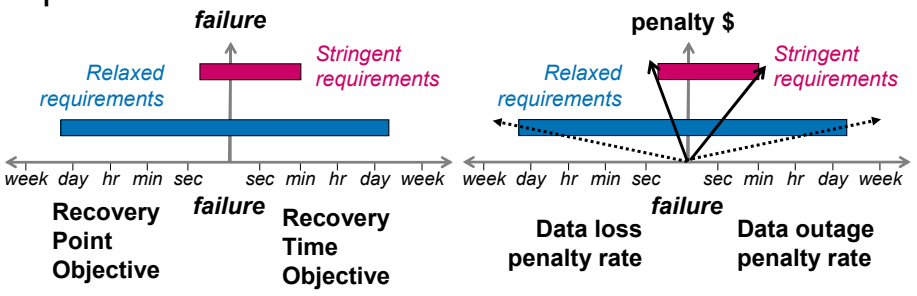
} the system that powers the service

Challenge: expressing dependability goals



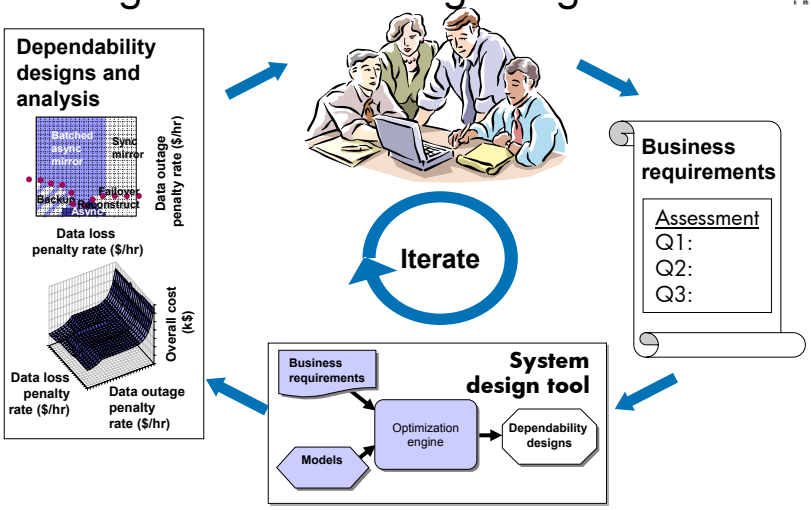
- Better (e.g., more quantitative) goals lead to better system designs
  - Users can't always state goals quantitatively
  - Specifying quantitative utility functions even harder
  - Users often possess intangible goals (e.g., manageability, training)
- Challenges:
  - Capturing utility-based goals in a quantitative fashion
  - Expressing intangible goals

Approach: quantitative utility-based specifications



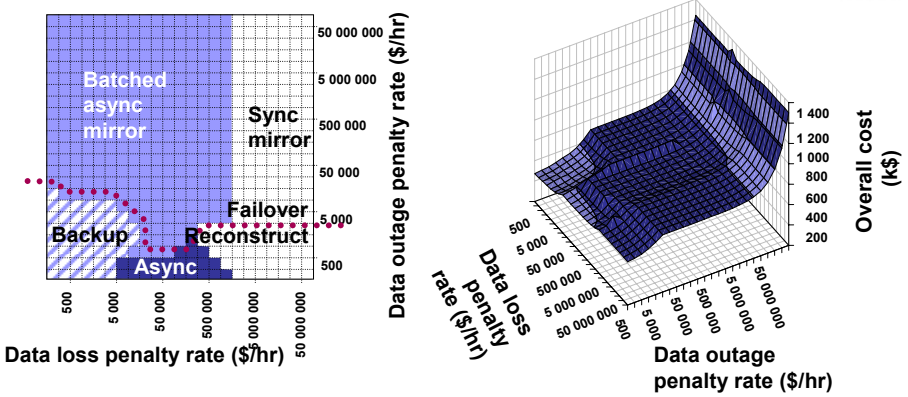
- Data outage penalty rate ($/hour)
  - How long before the system is back up?
- Data loss penalty rate ($/hour)
  - How much recent data can the system discard?
- Time-varying penalty rates
  - Allow differentiation between short and long durations
  - Allow specification of constraints (RTO/RPO + violation penalties)

Challenge: understanding design choices



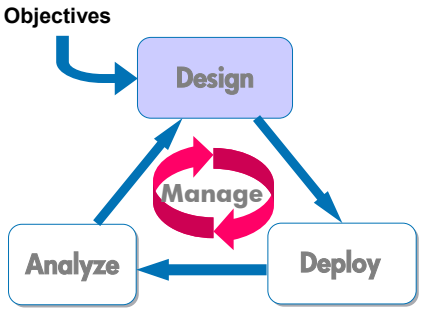
- Challenge: giving users feedback on design choice implications

Challenge: design space exploration



- Representing different choices for different objects
- Illustrating sensitivity to input choices
  - Business requirements, workload characteristics, failure likelihoods
- How to avoid overwhelming user with too much info?

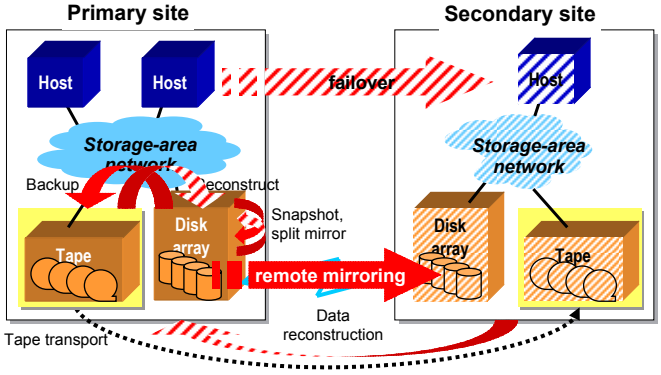
Outline



- Defining the desired level of service
- Designing
- Deploying
- Analyzing

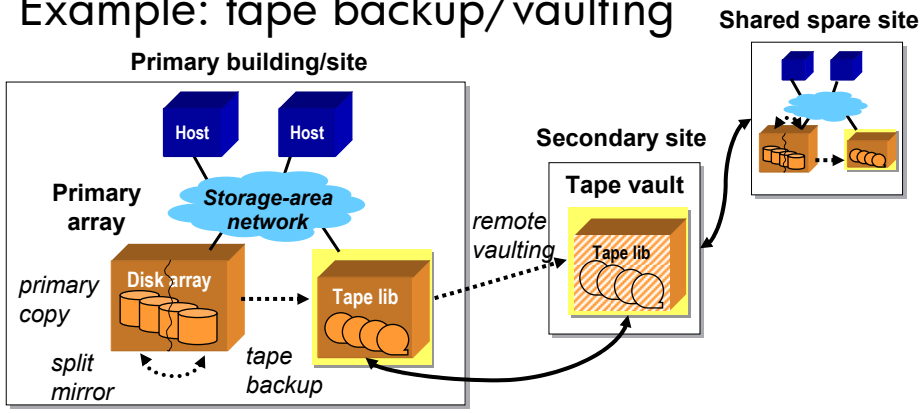
} the system that powers the service

Challenge: automating system design



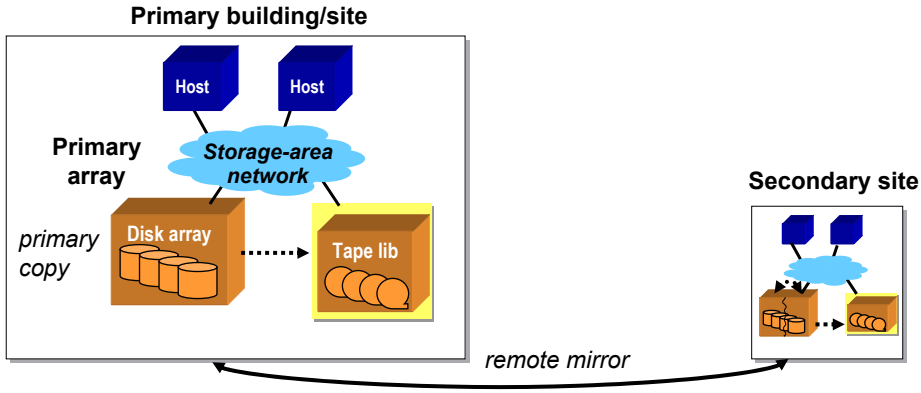
- Automatically designing dependable (storage) system
  - From scratch
  - Based on existing legacy system
- Choosing appropriate techniques to protect workload data, and how to set config parameters
- Allocating physical resources to protection workloads

Example: tape backup/vaulting



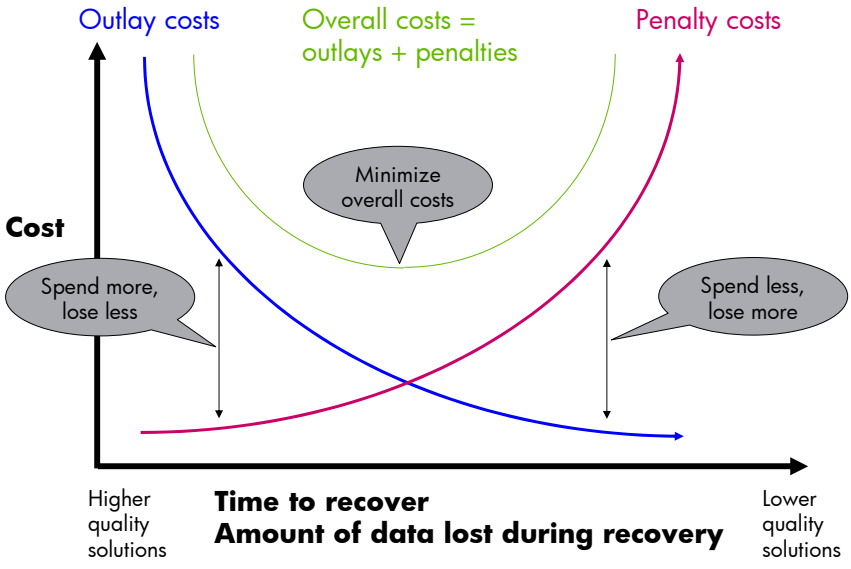
- Backup configuration questions:
  - How long between successive backups?
  - How often to do full vs. incremental backups?
  - How long should backup window be?
  - How long to keep backups?
- Vaulting configuration questions:
  - How often to ship tapes offsite?
  - How long to delay before shipping?
  - What to ship offsite?

Example: remote mirroring

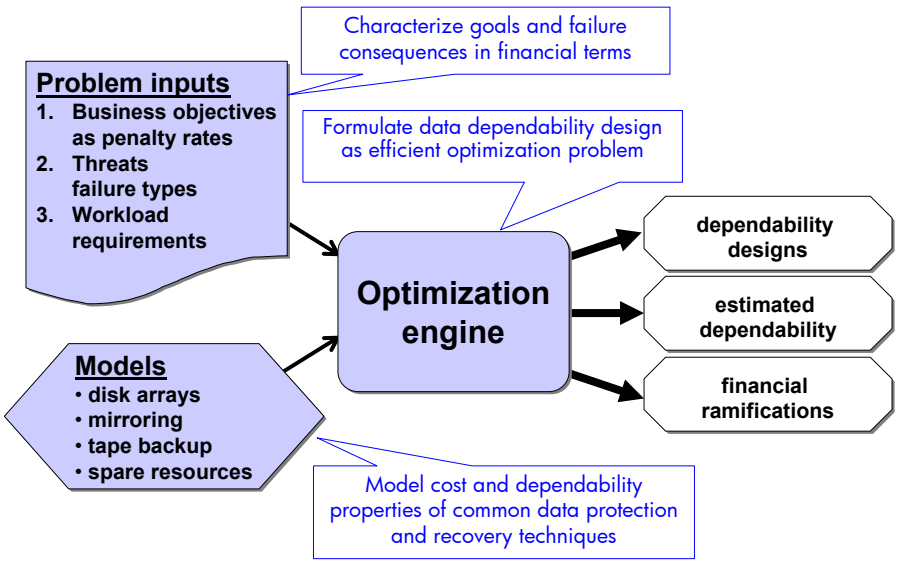


- Remote mirroring configuration questions:
  - What protocol to use – synchronous or asynchronous?
  - If asynchronous batch protocol, how long to coalesce updates?
  - How many network links to use?

Determining the right solution



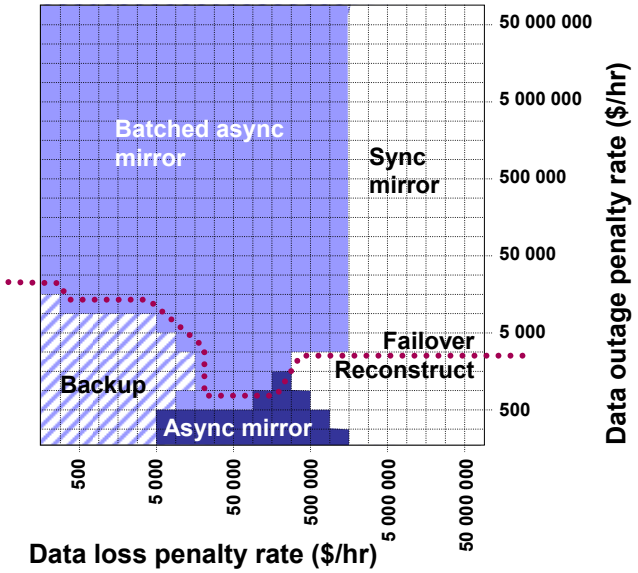
Approach: dependability as an optimization problem [FAST04]



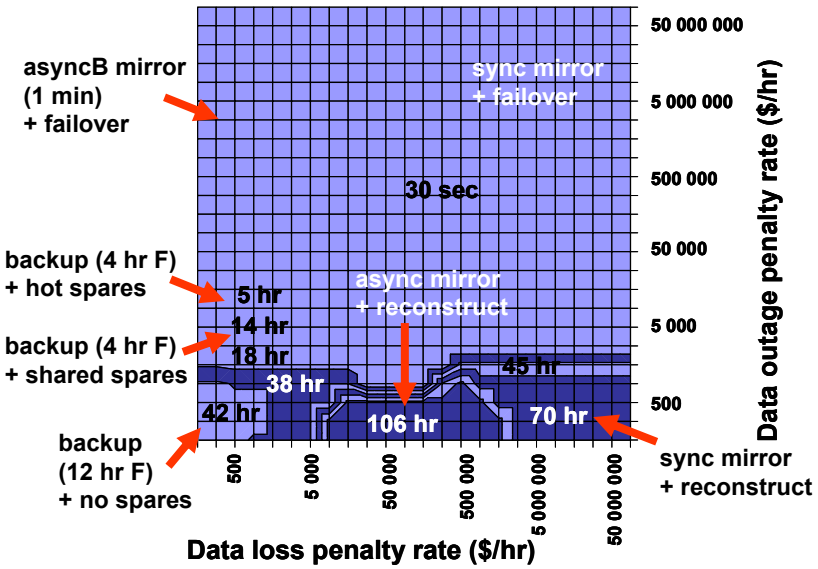
Mixed integer programming formulation

- Objective function
  - Minimize overall business cost = outlays + penalties
- Decision variables
  - Binary variables to select an alternative and its configuration
  - Integer variables for number of bandwidth devices (e.g., mirroring links or tape drives)
- Constraints
  - Allowable design alternatives
  - Bandwidth and capacity provisioning
  - Linearization constraints
- Solver prototype
  - Implementation using off-the-shelf optimization engine (CPLEX)

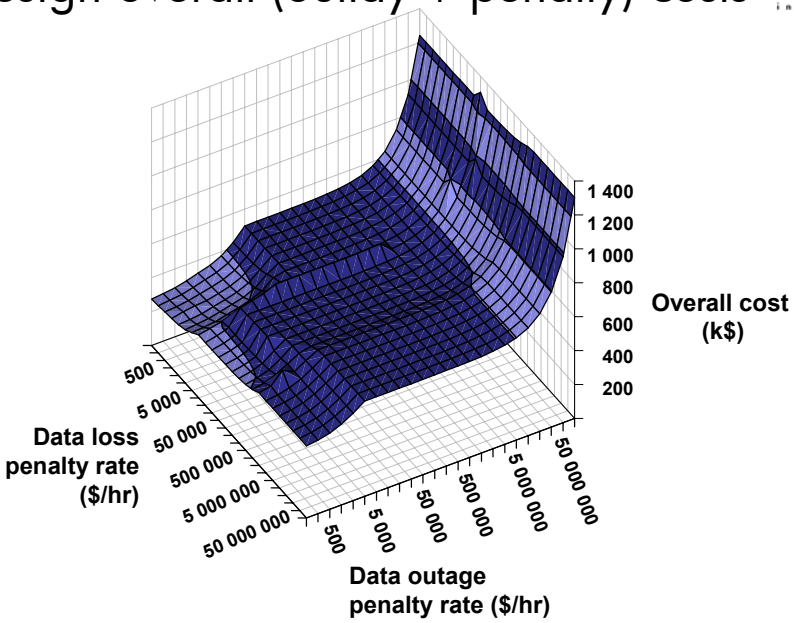
Design space exploration



Design recovery time

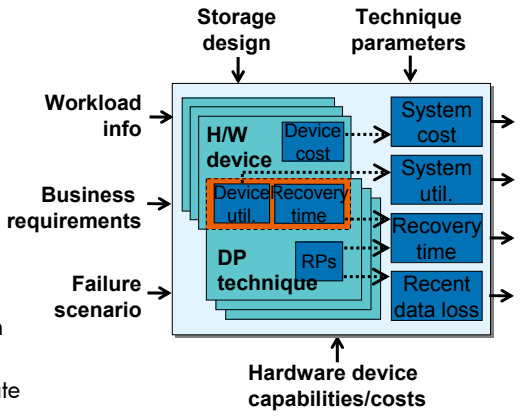


Design overall (outlay + penalty) costs



Challenge: new dependability techniques

- Issues:
  - Easily incorporate new techniques
  - Complex storage solutions: multiple techniques
- Approach: extensible modeling framework [DSN04]
  - Model secondary copy commonalities
    - Full vs. partial representation
    - Copy frequency, retention
    - Time for updates to propagate
  - Composition rules to evaluate overall solution recovery time and data loss

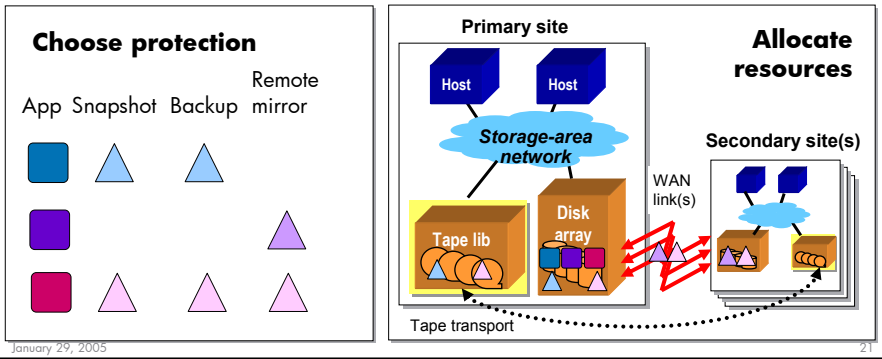


Open questions: new dependability techniques

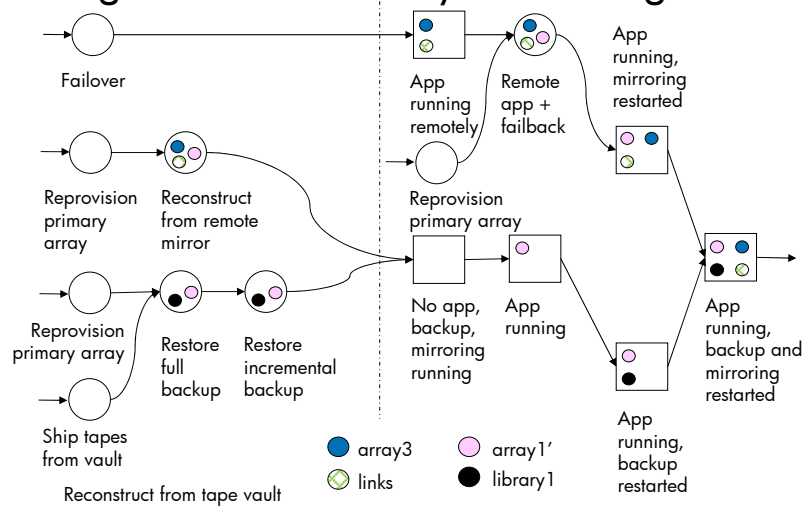
- "Grammar" to describe reasonable combinations of dependability techniques
- Extending framework to higher-level techniques (e.g., logging, checkpointing)
- Modeling tradeoffs between:
  - Techniques at different layers of stack
    - Block-level replication vs. log shipping
  - Techniques using different resources
    - Recompute vs. store intermediate results

Challenge: competing data objects

- Must choose protection and recovery alternatives and allocate physical resources per data object
- Potential approaches:
  - Two-phase optimization heuristic
  - Evaluation + randomized search



Challenge: failure recovery scheduling

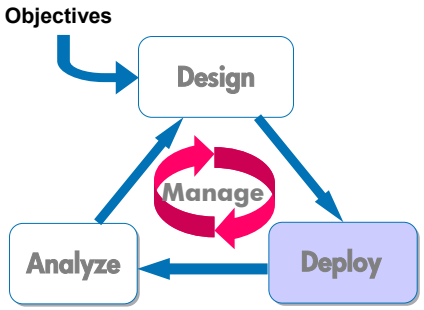


- Choosing the best set of recovery operations
- Determining how to schedule recovery operations and unaffected workloads

End-to-end dependability design

- Goal: end-to-end dependability
  - Business processes and applications are unit of dependability
  - Continuous service operation ("business continuity")
- Challenges:
  - Provisioning system resources (servers, storage, networks)
  - Effectively using techniques at all levels of application stack
    - Snapshots, checkpointing, logging and replication
    - Failover and recomputation of results
  - Managing interactions and tradeoffs between techniques
  - Translating end-to-end dependability goals into system component goals

Outline



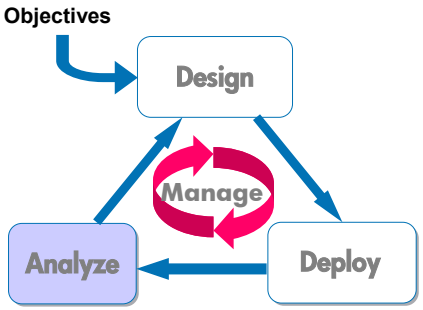
- Defining the desired level of service
- Designing
- Deploying
- Analyzing

} the system that powers the service

Deployment challenges

- Implementing dependable storage designs
  - Ex: interacting with backup software to adjust backup frequency
- Implementing recovery operations in response to failures
- Providing online data layout
  - Ex: RAID level selection [Anderson, et al., FAST2002]
- Migrating data in response to system changes

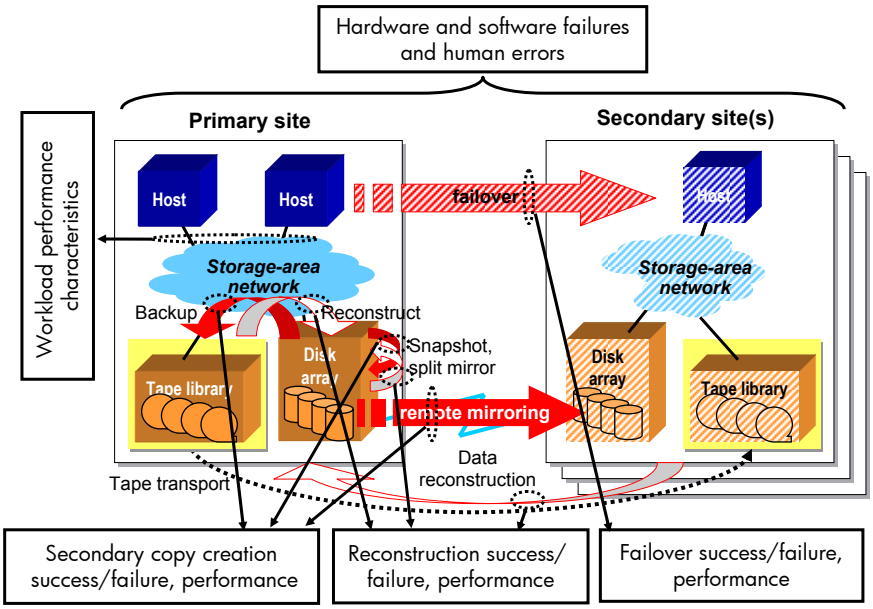
Outline



- Defining the desired level of service
- Designing
- Deploying
- Analyzing

} the system that powers the service

Analysis challenges



Open questions for analysis

- Verifying correct deployment of techniques
- Measuring resource requirements of successful secondary copy creation
- Diagnosing problems when they occur
- Collecting data on recovery behavior
  - Measure disaster drills and naturally occurring problems
  - Proactive small-scale fault injection using virtualization
- Using measurements to iteratively refine models
- Understanding workload characteristics
  - Steady-state behavior, trends and cyclic behavior

Conclusions

- Designing and managing dependable systems is challenging
  - Competing workload demands
  - Dynamic environments
  - Desire that system meets expectations
  - End-to-end dependability
- Automated data dependability provides starting point
  - Define desired level of service
  - Design, deploy, analyze system behind the service
- Wealth of research opportunities – join us!
- Further details available:
  - http://www.hpl.hp.com/SSP/
  - kimberly.keeton@hp.com

Backup slides

Data dependability bibliography

- [FAST04]: "Designing for disasters," K. Keeton, C. Santos, D. Beyer, J. Chase and J. Wilkes, *Proc. 3rd Conference on File and Storage Technologies (FAST)*, March 2004.
- [DSN04]: "A framework for evaluating storage system dependability," K. Keeton and A. Merchant, *Proc. Intl. Symposium on Dependable Systems and Networks (DSN)*, June 2004.
- [SIGOPS04]: "Lessons and challenges in automating data dependability," K. Keeton, D. Beyer, J. Chase, A. Merchant, C. Santos and J. Wilkes, *Proc. 11th SIGOPS European Workshop*, September 2004.
- [SIGOPS02]: "Automating data dependability," K. Keeton and J. Wilkes, *Proc. 10th SIGOPS European Workshop*, September 2002.

- Further details available:
  - http://www.hpl.hp.com/SSP/
  - kimberly.keeton@hp.com

Related work

- Dependability modeling and simulation techniques [Deavours2002, Haverkort2001, Kaaniche1998]
- System administration literature: operational issues [Chervenak1998, daSilva1993]
- Backup and return-on-investment calculators [Sun, EMC]
- Development, application of new data protection techniques [Rhea2003, Wylie2001]
- Specifying and evaluating dependability requirements [Keeton2002, Wilkes2001, Brown2000]
- Automatic storage design for performance goals [Anderson2002, Alvarez2001]
- Automatic tuning of application computation resources in multi-tier environments: [Janakiraman2004]