Storage WG Chartering Update

Prabal Dutta and Deepak Ganesan (co-chairs)
on behalf of
Junzhao Du, Kevin Klues, Ajay Martin,
Gaurav Mather, David Moss, Manju Rajashekhhar
Lot’s of Storage Activity

- Blackbook
- Capsule
- ELF
- FlashDB
- Matchbox
- MicroHash
- TEP103
- TINX
Genesis, History and Status

• Proposed by Deepak Ganesan at the TinyOS BOF during Sensys’06

• Initial membership policy
  – People responsive to the “Request for Participation”

• Weekly Telecons
  – 3/30, 4/06, 4/13, 4/20, …

• Wiki
  – http://tinyos.cs.berkeley.edu/storage

• Mailing List
  – tinyos-storage@millennium.berkeley.edu
Charter

“To define storage interfaces and abstractions, reusable non-volatile data structures, interoperability requirements for components that share flash memory, and reference implementations of storage systems.”
Key Issues

• Performance vs. Portability
  – Enumerate the tradeoffs; Find a balance

• Richer set of storage objects
  – Stream, index, queue, stack, dictionary

• Checkpoints, rollbacks, and compaction
  – A common framework

• Sharing a volume among several objects
  – Simple interleaving? Layered storage stacks?
Early Focus and Annexations

- Lesson 7: Permanent Data Storage
- TEP 128
  - Platform-independent storage abstractions
  - Revisiting volumes
  - Direct storage
- TEP 129
  - Basic Platform-Independent Non-volatile Storage Layers
  - Implementing platform-independent versions of Config, Log, Block, Stream, Index, Queue, Stack, Dictionary
- TEP ***
  - Common framework for checkpointing and rollback
Intersection with Core

• At the storage abstractions and interfaces
  – Volume
  – Config
  – Log
  – Block

• New storage abstractions below, next to, or above?
  – Direct access
  – Volume settings
  – Layered, interleaved log

• Lots of discussion and debate about the basic interfaces
  – Performance vs Portability
  – Generality vs Specificity
  – Will resolve differences through implementation and evaluation
Going Forward

• Low-level system interfaces
  – Bedrock of the storage system
  – Getting them right is important

• High-level, storage-backed, data structures
  – Hide low-level details
  – Enable greater developer productivity (e.g. Java Collections framework)

• …And the messy middle

• Help shape the future of TinyOS storage by joining the Storage Working Group!