

# HTAR and HPSS

Jerry Shoopman

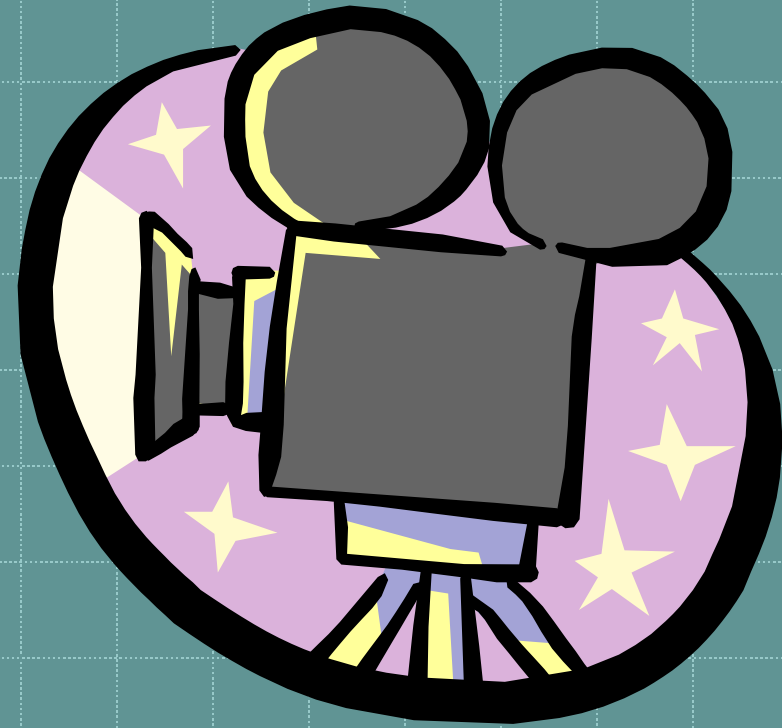
May 4, 2004

UCRL-PRES-203740

**This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48**

# Overview

- ◆ What is htar?
- ◆ A few htar definitions...
- ◆ How does htar work?
- ◆ What are some features of htar?
- ◆ What are some of the limitations?
- ◆ Why would I want to use htar?
- ◆ Why wouldn't I want to use htar?
- ◆ What is in the future for htar?



# What is htar?

- ◆ **HPSS Tape Archiver**
- ◆ Client API-based interface written by Mike Gleicher
  - ◆ Originally commissioned by LLNL in 2000
  - ◆ Was released as an add-on to R4.5 (for at least one site)
  - ◆ Widely available as part of the HPSS Offering in R5.1 and possibly in the R4.5 patch
- ◆ File bundling and manipulation tool.
  - ◆ Basically bundles small files on a platform into large files in storage. This increases performance for users and reduces metadata use in HPSS.

# A few htar definitions...

- ◆ **Archive File** – the large file that htar creates out of the small files
- ◆ **Member File** – one of the small files contained in the htar *Archive File*
- ◆ **Index File** – a catalog of the *Member Files* contained in the *Archive File*
- ◆ **Consistency File** – the file at the end of an *Archive File* used to verify the consistency of the *Archive File* and the *Index File*

# What are some features of htar?

- ◆ Compatible with tar
  - ◆ Uses the POSIX 1003.1 tar file format for the *Archive File* so regular UNIX tar can be used on the htar *Archive File*
  - ◆ Command line options were chosen to be as compatible as possible with AIX tar
- ◆ Ability to work with *Archive Files* that are:
  - ◆ HPSS-resident
  - ◆ Local filesystem resident (-E option)
  - ◆ Remote-machine resident (LLNL version only)

# Htar features cont...

- ◆ Allows users to list contents of the *Archive File* in storage without staging the *Archive File*
  - ◆ *Archive File* and *Index File* may (probably do) reside in different COSs
- ◆ Support for HSI auto-cos selection for *Archive File* and/or *Index File* (not used at LLNL, but used at ORNL and (?) NERSC)
- ◆ Allows users to randomly extract file(s) from *Archive File*
- ◆ Can create an *Index File* from an existing tar file
- ◆ Recursive, ease-of-use makes it attractive to users

# LLNL-only htar features



- ◆ Features specific to htar implementation at LLNL:
  - ◆ Works with *Archive Files* on remote-machines (non-HPSS)
  - ◆ Verify options for new and existing *Archive Files*
  - ◆ Option to delete local file(s) during create after successful copy to the *Archive File*
  - ◆ Option to extract directly from tape
  - ◆ Notifies users during extract if the *Archive File* resides on tape
  - ◆ Option to override the built-in maximum number of *Member Files* in the *Archive File*
  - ◆ Supports stdin file list and stdout extraction for compatibility with Hopper (GUI interface)

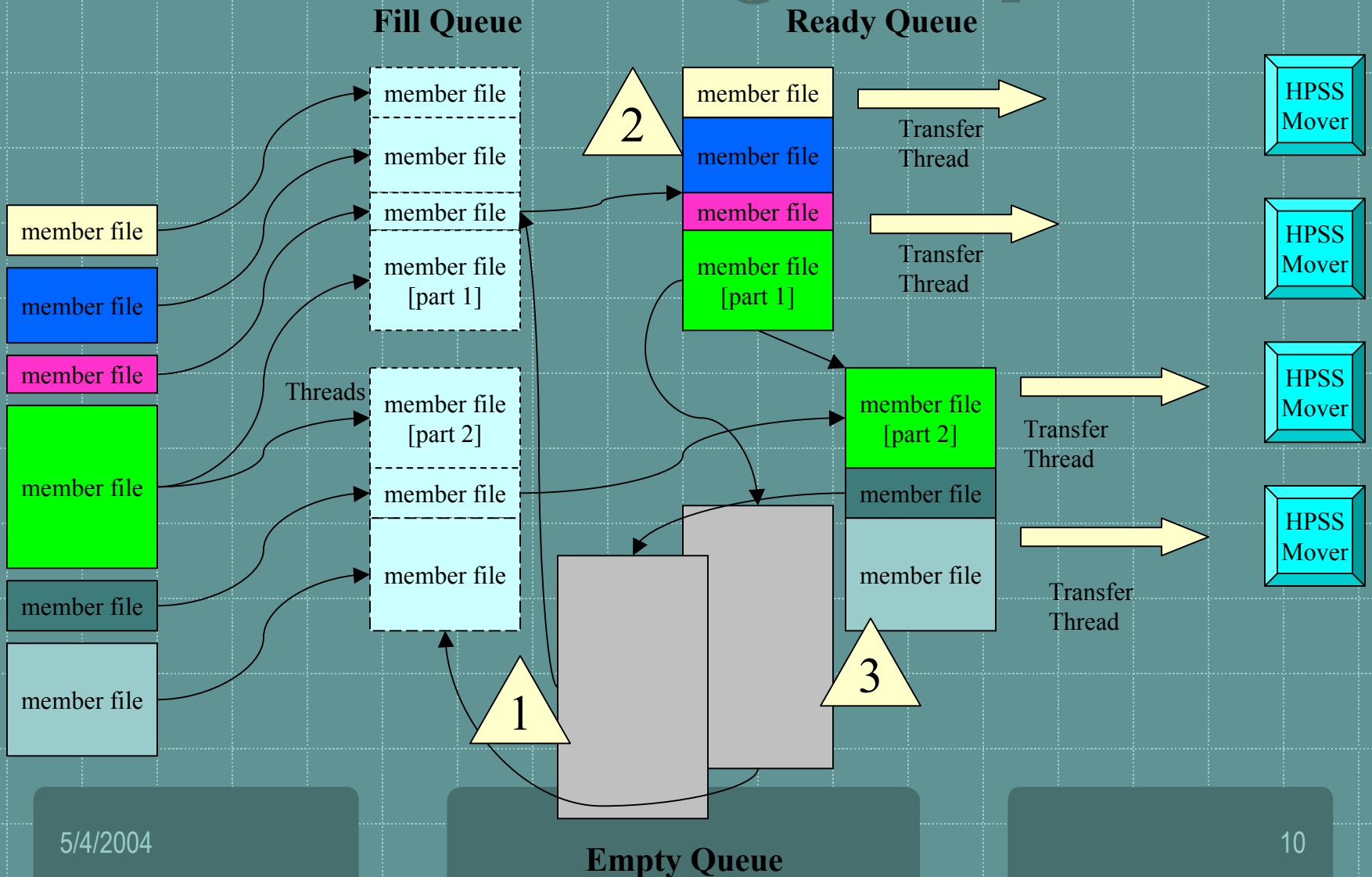
# Examples of using htar

- ◆ Bundles the local files file1, file2 and file3 into an *Archive File* called myfiles.tar in HPSS home directory
  - ◆ `htar -cf myfiles.tar file1 file2 file3`
- ◆ Displays the names of the *Member Files* in the myfiles.tar in the HPSS home directory
  - ◆ `htar -tf myfiles.tar`
- ◆ Extracts file2 from myfiles.tar, retrieves file2 from storage, and writes file2 to the current local directory
  - ◆ `htar -xf myfiles.tar file2`

# How does htar work?

- ◆ Htar uses multiple threads to read *Member Files* from local disk and construct the *Archive File*
- ◆ A fairly elaborate, coordinated buffering scheme is used to optimize reading of small local files.
  - ◆ Buffering scheme also manages various phases of data collation and data transfer

# Htar buffering example



# Implementation details

- ◆ Htar runs on AIX (DCE and non-DCE), Tru64 UNIX, IRIX, Solaris, Linux, Mac and Windows
  - ◆ Htar runs on all platforms that HSI runs on, although it is not extensively tested (yet) on machines that aren't used at LLNL
  - ◆ This includes all flavors of linux (intel/sparc/alpha/ia64), hpux, freebsd, and unicos (all models)
- ◆ On LLNL HPSS platforms, htar server is built using the HPSS DCE-based client API library
- ◆ To run the htar client on non-HPSS platforms, it must be built using the HSI non-DCE client library

# What are some limitations?

- ◆ Requires a disk storage class – htar can't write direct-to-tape (*this limitation goes away with R5.1 version*)
- ◆ To be tar-compatible, htar is limited to a maximum **Member File** size of 8GB
- ◆ Can't append a new *Member File* to an existing *Archive File*
- ◆ Can't do an rsync-style append to an existing *Archive File*
- ◆ Can't update/replace *Member Files* in an existing *Archive File*

# Why would I want to use htar?

- ◆ Your system is being overwhelmed by a disproportionate number of small files
- ◆ Users who store many small files are complaining about slow performance
- ◆ Users could benefit from an organizational tool that stores their data in a (long-life) standard format
- ◆ Users are comfortable learning a new interface (and *you're* comfortable supporting one)
- ◆ You already support hsi or are preparing to support it

# Why wouldn't I want to use htar?

- ◆ Your users already write very large files
- ◆ Your users want to see their individual Member file names in the HPSS name space
- ◆ You don't want to support "yet another interface"
- ◆ You don't want to manage a wish list from users who want htar to do everything under the sun...

# What's in the future for htar?

- ◆ At LLNL:
  - ◆ Verify & delete functionality (testing now)
  - ◆ Checksums
  - ◆ I/O error retry options (highly desired by users)
- ◆ On hold, but requested by LLNL users:
  - ◆ Append & rsync-style append/update
  - ◆ Delete a file or subtree from an Archive file
  - ◆ Use wildcards for extract, list, etc...
  - ◆ Exclude option
  - ◆ Create local or HPSS tar file from a list of HPSS files



# Htar's future cont...

- ◆ Mike is planning to revisit the buffering scheme, with the intent of being able to transfer multiple large files in parallel
- ◆ Mike also plans to be able to support multi-node htar at some point after multi-node HSI support is available.
  - ◆ This is a little trickier for htar than for hsi, since all, some or none of the member files may reside on global shared filesystem(s).