Transfer and Cataloguing

Gavin McCance
Simone Campana
Roberto Santinelli
Andrea Sciaba

Evolution of WLCG Data and
Storage Management Workshop
Amsterdam
17 June 2010

Outline

- Few slides as input for discussion
- While I'll probably talk in terms of the FTS and LFC software (and I can't resist) really the discussion is about the architecture
- Transfer first then cataloguing

FTS – the software

- Ex. the odd spectacular bug it's rather stable and boring
- Core concept of the channel (point to point link) as the "thing you manage" was simple
  - Come from MONARC – manage network
  - Deploy at T0 and T1s only
- But…

FTS issues / lessons

- Channel multiplicity horror
  - Affects operations – configuration
  - Conceptually cumbersome to manage many point-point links in an any-to-any world
  - STAR channel constipation, group channels hard to configure
- Non-communicating FTS server instances
  - Overload: A T1's FTS can control write but not read
  - You need to know which one to submit to
  - No magic: you need to know where your files are
- FTS | SRM interface too loosely coupled
  - Really hard for ops to debug and trace the whole stack
  - No real back-pressure – afterthought of "SRM_RATHER_BUSY" was an afterthought
  - Easy to overload storage

Transfer architecture discussion

- No solutions: food for discussion
- Abandon fixed channel concept and include storage bandwidth
  - Still the need to control the networking bandwidth used for T2/T2
  - Easy to say (it's an n x n matrix)
  - Some possible solutions need to be prototyped (the global omniscient beast vs. back-pressure). Simulate?
- Submit anywhere – use standard MQ to glue it
  - Move away from a model where you have to know – if replica 1 is not available, get the file over there
- Move scheduled WAN transfer functions closer to storage?
  - Help with operations
  - Is the "file" as the data management primitive correct? Sets or chunks (with 'getNext' iterator) might help the storage balance better

LFC – the software

- Secure namespace to tell you where the replicas are
- Absolutely stable and boring
- Used in both global, cloud and local (tier-2) mode for many VOs both large and small
  - Can be replicated for availability

- Lessons learned from LFC
  - Bulk operations were clearly needed
  - Strong desire now for **standard** http-based access
  - Some unforeseen 'admin' operations
    - e.g. rename storage element
  - Would have been nicer to allow more experiment specific metadata – external joins are still a pain

Cataloguing

- Apart from specifics on any product…
- …consistency with storage and between catalogues is the main challenge

Cataloguing consistency

- Current consistency model is not resilient to failures
  - Storage failures lead to dangling entries to be cleaned up manually. Catalogue failures lead to orphaned files.
  - Namespace scanning for diffs is expensive (srmLs 'abuse')
  - Multiplicity of catalogues – experiment book-keeping, {global, cloud, local} replica catalogues, storage catalogues
- [advert] proposed demonstrator to use reliable message (i.e. industry standard MQ) as backbone of the reliability
  - All interested catalogues can 'subscribe' for new files / deleted files
  - Eventual consistency model
- Add GUIDs to storage catalogs to remove the need for local file catalogue
  - Lost files can be broadcast on the "lost" topic to interested catalogues
  - Also for corrupted "bad" files (not readable, no GUID)

Summary

- FTS and LFC function as intended at ~WLCG scale
- The conceptual model of WLCG's transfer system is too simple
  - Need to consider storage bandwidth as well as network
    - But global n x n optimisation is hard
  - A global system would allow more magic (strawman)
  - FTS | SRM interface is too loosely coupled
  - Is the "file" the correct primitive for unordered bulk operations?
- Consistency is the key challenge for cataloguing
  - Add GUIDs to storage catalogue
  - Use industry standard messaging a backbone of reliability and storage / catalogue integration