Transparent Continuous Access to Infinite Data: A New Vision for Enterprise Data Management

Don Banks and Christina Mercier
Auspex Systems
5200 Great America Parkway
Santa Clara, CA 95054
dbanks@auspex.com and cmercier@auspex.com
+1 408-986-2000
FAX: +1 408-986-2020

Abstract: This paper presents the "Transparent Continuous Access to Infinite Data" (TCAID) architecture, developed at Auspex Systems, which addresses the enterprise data management challenge. The model presented here separates policy from mechanism and treats the solution space as a bounded area described by simple data management parameters which encompasses a wide spectrum of possible data management solutions. Using this model existing data management point products become a series of customer-specified policies implemented by pre-packaged mechanisms that operate on customer-specified target data.

1. Introduction

Organizations are centralizing control of their distributed data in order to share and exploit the content. The Transparent Continuous Access to Infinite Data (TCAID) architecture addresses the centralized control of globally distributed data by supporting a set of data management solutions capable of implementing many policies over the data irrespective of its location. The architecture makes the set of services offered by the "cloud" of data management platforms function as a single cohesive solution.

The TCAID model provides access to client data so that even in the event of failures and/or loss of the primary copy of the data, client programs are unaware of the failure and continue to access it as if the primary is still the source of the data and no failure has occurred. No logical limits exist on how large the total collection of data can grow for either individual files or entire file systems including copies of those files and file systems.

2. The Problem with Current Data Management Solutions

Data volumes supporting the enterprise, and even single applications, have expanded to tens of terabytes of data. Data Management products such as HSM, disaster recovery, archive, and high availability, are showing signs of strain managing such enormous data sets and are reaching their breaking point. These products are often limited by file size, internal database performance, recovery mechanisms, etc. The backup window has disappeared as file systems have grown to tens of terabytes and terabyte file transfers for backup and migration are saturating network bandwidth. Data management products, dependent upon the availability of tape drives or other media devices to deliver their service, time-out or "hang" critical client requests. Products that have worked well for managing data in the enterprise have suddenly slowed down, or worse, stopped working all together as thresholds, queues, and/or resources have been exceeded.

3. The Vision: A New Paradigm for Data Management

Transparent Continuous Access to Infinite Data (TCAID) is an architecture which addresses the limitations encountered with current Enterprise Data Management products. TCAID separates policy from mechanism and presents a model that treats the solution space as a bounded area described by simple data management parameters, encompassing a wide spectrum of possible data management solutions. Individual solutions, such as backup, migration, as well as new ways of managing data not available in current products, can be described by applying customer-specified policies to the data to be managed. TCAID technology provides high-speed replication and a simple, manageable way to describe policy over a set of data. The TCAID architecture allows growth to hundreds of terabytes. It provides an integrated solution which is only constrained by the amount and type of data storage hardware available.

4. The TCAID Architecture

The TCAID architecture is an "embedded" model in the sense that its internal architecture is completely transparent to clients during operation. For the network filesystem client, the data is available over a logical connection to a server. Where and how the data is physically stored and accessed is not visible or of concern to the client or the client network filesystem protocol. TCAID is an architecture directed at the fileserver environment, not a local filesystem implementation. This is an underlying assumption that permeates the design. The fact that all requests (whether NFS, SMB or some other supported network file system protocol) come from the network rather than through a local API or local application request, supports and constrains the TCAID design.

There are two ways the model must be viewed. The first is the data-centered view (Figure 1), which illustrates the concepts by which data is managed to achieve policy-driven data management services. The second is the client-centered view (Figure 2), which represents the infrastructure and logical presentation of the TCAID-enabled file servers to clients.

4.1 The Data-Centered View

The TCAID model encompasses data management and high availability products by describing data replication as a three dimensional model. The first dimension (shown on the x-axis in Figure 1) is the logical distance from the source copy that the replicated data resides (local disk or tape versus disk or tape on another server). The second dimension (shown on the y-axis) is the time allowed to elapse between the time the source copy is modified and the time that the subsequent target copy is created (mirroring delay). The third dimension (shown on the z-axis) represents some number of coherent replicated copies in time that are to be managed and made available as historical views of the data (e.g., today's version, yesterday's version, etc.). While the x and y axes represent in a logical fashion the physical topology over which the data is spread, the z axis allows time to progress backwards (e.g., last Tuesday's data, last month's data or last year's data) to represent the illusion of infinite versions of data kept in the system managed "archives".

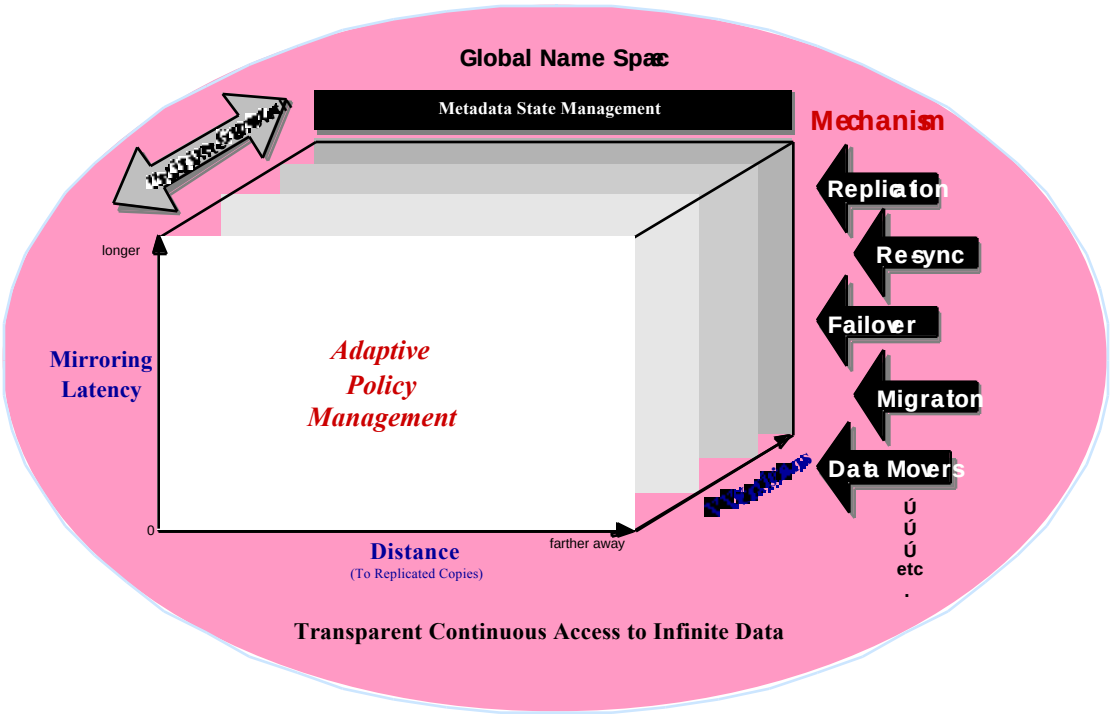

Figure 1: The Data-Centered View

The intent here is to provide sufficiently rich and powerful data management mechanisms such that a custom solution can be assembled and executed on demand based on the policy or policies specified by the administrator for particular data sets. By applying a rich common set of mechanisms, any policy can be implemented. This is the classic separation of policy from mechanism to allow a dynamic, adaptive implementation.

4.2 The Client-Centered View

The second view of the TCAID model is the client view of the TCAID-enabled server (Figure 2). A key element of this architecture is that it distributes data in a controlled and managed fashion that is transparent to the network client. In essence, the data-centered model is overlaid on this infrastructure in order to provide the illusion of consolidated data while at the same time providing the reliability and convenience of automatic distributed replications.

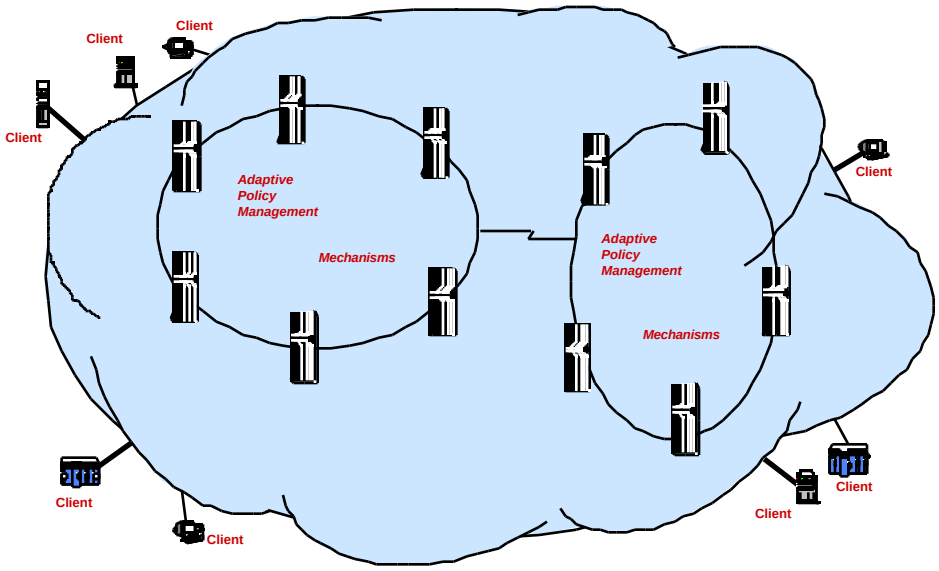

Figure 2: The Client-Centered View

Network clients accessing files through standard network protocols have a logical (rather than a physical) view of a "cloud" of TCAID-enabled servers; the client views a single data server providing a single connection point. Physically this cloud may be composed of any number of file servers capable of taking over for a failed server or filesystem and making the replicated data available in a manner transparent to the client. By specifying a policy that creates one or more replications of the data within the cloud, administrators guarantee that the physical failure of a server providing the primary copy of the data causes little or no effect to the client.

One of the important attributes of this architecture is that it allows the construction of a distributed filesystem within the cloud while presenting a simple single server view to the client. The client does not have to run any special software in order to gain the advantages of the distributed implementation. Location-transparent file naming along with a global name space provides the flexibility required to manage the distributed data.

5. TCAID Technology

The separation of policy and mechanisms in the TCAID architecture allows the system to provide an implementation that can dynamically build a solution to carry out one or more policies specified by the administrator. The mechanisms are transparent and the interface by which the policy is specified is simple to understand and administer. The mechanisms represent the technology while the policies represent the presentation of that technology in an implementation-neutral fashion to the system administrator who must describe how the data is to be managed.

Policy is the means used to define the ways in which data items (e.g., a file, a directory of

files or a file system) are managed with respect to creating and maintaining replications. Policy management is adaptive in both a static and dynamic way. Static adaptation occurs as the system administrator defines the policies through the provided GUI. This is termed static because it occurs once at definition time. The system dynamically chooses "intelligent" defaults for not yet specified parameters and defines/chooses the mechanism(s) required to implement the policy (or policies) specified by the administrator in order to build a custom data management application. Conflicts are recognized at the time they occur and conflicting selections are prevented. Dynamic adaptation occurs as the implementation of policy (i.e., the mechanism) adjusts to changing conditions and/or configuration so that the system can continue to carry out the desired policy in the most efficient and effective way. For example, if asynchronous replication is specified with the copy being no more than 15 minutes out of date but the network bandwidth changes (either up or down), the replication mechanism adjusts by changing the replication window. When policies cannot be carried out as specified, the system provides a notification alert and logs the untoward event(s).

Mechanisms implement policy. They are comprised of independent, modular "building blocks" that provide capabilities required to fully address the policies. Mechanisms work together as a cohesive unit by sharing and operating on common metadata, either locally or distributed across the network. Mechanisms represent both the platform-independent and platform-dependent parts of the system. Depending on the base system infrastructure, strong interfaces (i.e., APIs and/or protocols) allow these mechanisms to be highly portable by enabling alternative shallow-stack or deep-stack implementations of function.

6. Status

Work is under way at Auspex Systems to implement solutions using the TCAID architecture. Auspex is currently early in the prototyping and development phase and expects to have concrete findings and conclusions with respect to the implementation within the next six months. At this point, the TCAID architecture has provided Auspex with a solid foundation from which to build solutions to the data management problem for the consolidated network file server environment.