iSAN: Storage Area Network Management Modeling Simulation

Ramani Routray[*], Sandeep Gopisetty[*], Pallavi Galgali[±], Amit Modi[±], Shripad Nadgowda[±]
[*]IBM Almaden Research Center, [±] IBM India Systems and Technology Lab
[*]*{routrayr, skg}@us.ibm.com*, [±]*{pgalgali, amitmodi, shripad.nadgowda}@in.ibm.com*

Abstract

Storage Management plays an important role in ensuring the service level agreements (availability, reliability, performance etc..) that are critical to the operation of resilient business IT infrastructure. Also, Storage Resource Management (SRM) is becoming the largest component in the overall cost of ownership of any large enterprise IT environment. Considering these functional requirements and business opportunities, several SRM suites have cropped up in the market place to provide uniform and interoperable management. But, development and test of these suites require access to huge set of heterogeneous multi-vendor Storage Area Network (SAN) devices like Fiber Channel Switches, Storage Subsystems, Tape Libraries, Servers etc... It's almost impractical for a SRM Suite software manufacturer to own and manage these huge varieties of devices.

Management modules of SAN devices have become logically independent components with the emergence of CIM and SMI-S. In this paper, we propose a framework and implementation named iSAN (imitation Storage Area Network) that models the management module of the devices. Our tool can be used to perform a) simulation of management module ranging from individual device to large scale multi-vendor heterogeneous SAN for enterprise b) what-if-analysis of enterprise IT environment before modeling the changes. This tool provides efficiency and cost-effectiveness with respect to development, test of SRM suites and planning of IT environment by removing their dependence on high cost SAN hardware.

We have implemented this tool iSAN, and our experiment results show that one can attain the above mentioned functional objectives and bring significant productivity to enterprise IT environment management.

1. Introduction

Enterprise IT environment management constitutes of server, storage and network management. Storage resources management and SAN management is the major part of any large scale data centric enterprise. Basic structure of a SAN is represented in Fig-1. Any SAN management software is composed of basic components like discovery, control, monitoring, planning and reporting. Advanced analytics like predictive analysis, problem determination are built on top. Considering the heterogeneous and multi-vendor nature of the environment, performing the above tasks in a uniform fashion is a big challenge. To address these challenges and bring in uniformity and interoperability, DMTF[1] and SNIA[5] have come up with several standards like CIM[2], SMI-S[5] and WBEM[3]. Based on these standards, most of the SAN devices like Storage Subsystems, Tape Libraries and Fiber Channel Switches are shipped with a management module called Common Information Model (CIM) agent. For some devices, CIM agent is embedded in the device and for other devices, it is externally installed on a server. Popular systems management products like IBM TotalStorage Productivity Center (TPC), HP AppIQ, EMC Control Center exploit the above technologies to achieve their functionality. These tools also use SNMP and proprietary management methods in conjunction with CIM agent.

CIM agent is generally composed of Common Information Model Object Manager (CIMOM) and a set of provider(s). CIMOM is the basic server infrastructure. Some popular open source CIMOMs are Pegasus, Sun WBEM and SNIA. Provider is the pluggable library that realizes the CIM profiles by serving the requested information out from the device.
In this paper, we present a framework and implementation that simulates the CIM agent of any SAN device. Each individual device CIM agent can be

simulated in this framework based on the specification defined in a XML file. Based on user supplied specification, we can simulate multiple CIM agents that combined together represent a SAN. In this case, information exposed by simulated CIM agents need to be correlated to one another to virtually represent a SAN from management modeling perspective. We use database for storing device configuration details, a standard set of providers and a service for generating and correlating data across CIM agents. Modeling and implementation of profiles like fabric profile, array profile, server profile etc… in correlated and temporal fashion provides a completely virtual SAN.

SRM vendors spend huge amount of money in setting up heterogeneous environment to develop and test their software or access geographically distributed heterogeneous environments over network. SNIA also hosts such a lab for interoperability and certification testing.

Administrators also do not have access to any standard primitives that can help them perform impact analysis before they deploy new hardware or perform certain configuration change operation on their SAN.

By simulating CIM agents of multiple SAN devices in a collective fashion, virtual SAN can be exposed to a SRM suite. SRM suites can use iSAN for the following purposes:

- Visual workbench to drag and drop different devices and connect them together to get a virtual SAN from management perspective.
- SRM suite can perform regular management functions like discovery, configuration and monitoring on the virtual SAN as if it is a real SAN.
- Planning of a SAN from scratch or extension of an existing SAN in a virtual environment to perform what-if-analysis.
- Impact analysis against fault injection or growth pattern in a temporal fashion.
- Minimize network latency by presenting a virtual SAN close to the management server.
- Snapshot any real live CIM agent and replay it using iSAN's record-replay mechanism.

It is important to note that the virtual SAN with the above features brings in significant value add to the management suite because these tools never really require the existence of a real heterogeneous SAN for it's development and testing.

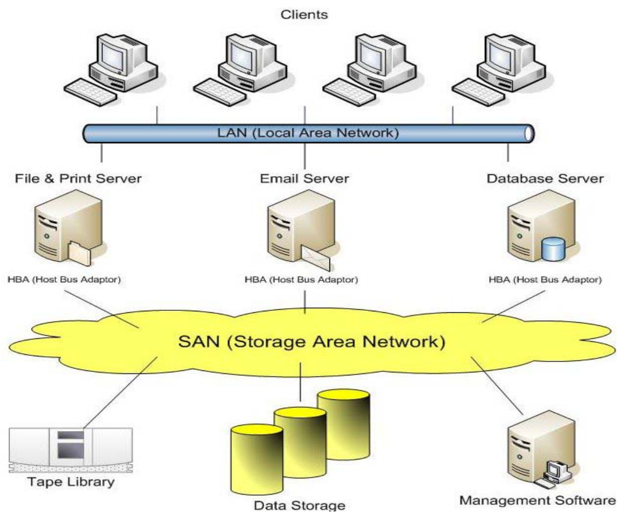

Fig. 1. Representation of a storage area network where servers (such as file and print servers, email servers and database servers) access data storage in subsystems and tape libraries via host bus adapters over a switched network.

The rest of the paper is organized as follows: Section 2 describes the system architecture overview of iSAN. Section 3 describes experimental evaluation, evaluation strategy and results. We present related and future work in Section 4 and conclude in Section 5.

2. System Overview

This section provides an overview of iSAN. It discusses the input, output and key components of this tool in following subsections. Architecture and components of iSAN is shown in Figure-2. Words like "user" and "administrator" have been used interchangeably and mean the user of iSAN.

iSAN obtains input about the required SAN environment through a visual workbench and generates a set of correlated CIM agent(s). iSAN's generation of virtual SAN environment is classified into the following two broad categories:

Snapshot Approach: Using this approach, user can take snapshot of a set of device CIM agent(s) that represent a real live SAN. Snapshot image is then replayed through iSAN.

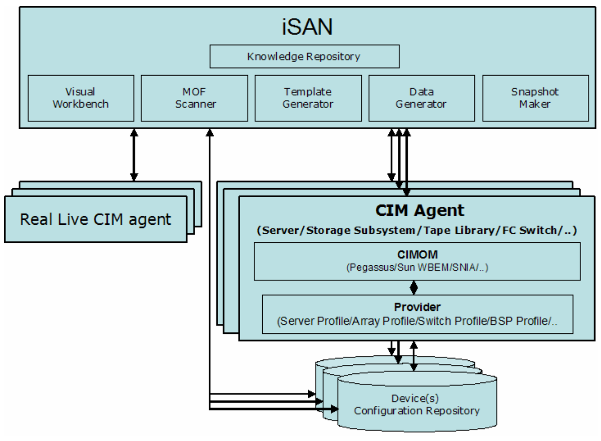
iSAN
Knowledge Repository
Visual Workbench
MOF Scanner
Template Generator
Data Generator
Snapshot Maker
Real Live CIM agent
CIM Agent
(Server/Storage Subsystem/Tape Library/FC Switch/..)
CIMOM
(Pegassus/Sun WBEM/SNIA/..)
Provider
(Server Profile/Array Profile/Switch Profile/BSP Profile/..
Device(s) Configuration Repository

Fig. 2 : Architecture of iSAN

This approach helps minimize the network latency (e.g. users accessing SNIA lab over WAN can run the exact same set of device CIM agents from their local LAN). Also, users can extend the snapshot through the visual workbench and perform *what-if-analysis*. In this approach, data from one or more CIM agent is retrieved by traversing the profiles. All the retrieved information is persisted in iSAN device configuration repository. Managed Object Format (MOF) files describing CIM class definitions are also queried from real live CIM agents and compiled into the iSAN simulated CIM agents.

Configuration Approach: Using this approach, user can specify the configuration of the required SAN environment. iSAN generates the required data and hosts the CIM agent(s) that represents the virtual SAN environment. Data generation in this approach is governed by device templates. Templates are nothing but canned definitions of CIM implementations and definitions for a given device from a particular manufacturer.

Following subsections describe individual components of iSAN in detail.

2.1 Visual Workbench

Visual workbench is a visual editor where user can specify the input for iSAN and visualize the generated virtual SAN. For snapshot approach, user specifies information about the real live CIM agent(s). For configuration approach, user drags and drops SAN devices into the visual workbench and establishes connectivity. Also, certain level of random connectivity among devices can also be left for iSAN to determine. User specification from graphical description is converted into a XML document. This XML document depicts the devices, device configurations and their connectivity. SAN planners that take declarative requirement specifications and generate the SAN blueprint can also be plugged into the visual workbench. Once, an operational iSAN environment is established, it can be visually tweaked for *what-if-analysis*.

2.2 Snapshot Maker

Snapshot Maker takes the snapshot of any live CIM agent. Live device CIM agent is basically a collection of CIM instances that realizes a collection of profiles. CIM instance is instance of a CIM class. Definition of CIM classes, supported profiles etc.. are also exposed by CIM agent in a standard fashion. Based on these CIM agent definitions, snapshot maker scans the namespace(s). For each of the namespace, CIM instance(s) of all of the CIM classes are retrieved from the CIM agent by performing *enumerateInstances* API invocation. This invocation is performed by the CIM client that is embedded inside iSAN. Namespaces, CIM class names, enumerated CIM instances along with CIM objectpath(s) are cleansed, indexed and persisted in the device configuration repository. This component follows a very standard *cookie-cutter* approach that can snapshot any CIM agent. Enumeration of CIM instances are done with all expanded options:

```
enumerateInstances (
CIMObjectpath OP_with_className_and_namespace,
boolean deep = true,
boolean localOnly = false,
boolean includeQualifiers = true,
boolean includeClassOrigin = true,
String[] propertyList = null );
```

Cleansing of the retrieved involves formatting of the data with respect to the delivering CIMOM. Indexing of the information helps in replaying back the persisted information efficiently through iSAN provider. Definition of CIM classes that are represented in a MOF (Managed Object Format) is also compiled into the iSAN CIMOM from the real live CIMOM. Details of MOF Scanner, Provider are described in subsequent sections.

But, this approach described so far assumes exclusive access of snapshot maker to the real CIM agent. Because, real live CIM agent can be reflecting the changing configurations that might not be consistent across start time and end time of the snapshot process. So, to capture a consistent device configuration from the real CIM agent, device is left untouched during the snapshot process. This process of snapshot making is called offline snapshot.

In certain scenarios, access to real live CIMOM access can not be restricted only to iSAN snapshot maker during the snapshot process. So, to solve this problem, at the beginning of the online snapshot process, iSAN subscribes to CIM indications of all the instance creation, deletion and modifications. In the due course of CIM instances being copied by snapshot maker, if any instances are modified, iSAN copies the old data first to the device configuration repository before the instance gets modified in the real live CIM agent. CIM instances that get created after the beginning of the snapshot process do not get copied by the snapshot maker. CIM instances that are deleted after the beginning and before the end of the snapshot process get copied by the snapshot maker. Snapshot created using this process is called point-in-time snapshot. But, this approach is restricted by limitations of certain device CIM agents that are being snapshotted.

Creation of snapshot of CIM agents reduce network latency of SRM testing frameworks significantly by presenting a virtual SAN in a local LAN. Also, this approach removes the constraint of SRM development and testing on SAN availability.

2.3 MOF Scanner

Snapshot maker persists the CIM instances from the live CIM agent. But, the CIM class definitions from real live CIMOM represented in Managed Object Format (MOF) needs also to be copied for compilation into the iSAN CIMOM. During the snapshot process, CIM class definitions are also copied by MOF scanner and compiled into the iSAN CIMOM. iSAN also exposes a support matrix describing the device types, manufacturer and CIM agent versions that are supported. iSAN scans MOF files of the multi-vendor devices for each CIM agent version from real live CIM agents and stores them in it's knowledge repository. MOF scanning involves retrieval of class definitions, class hierarchy, methods and qualifiers by use of CIM calls like *enumerateClasses*.

2.4 Template Generator

iSAN supports several types of devices to be dragged and dropped and interconnected in the visual workbench. Each device manufacturer exposes device information in compliance with CIM. But, vendor specific naming convention and internal device component arrangement are still proprietary. To be exact and close to the real devices, iSAN scans through the CIM instances of real live CIM agent and builds a template. These templates for different deice types are stored in the knowledge repository of iSAN. Each template is uniquely identified by a tuple: e.g. *template (device =* Storage Subsystem*, type =* model-2107*, manufacturer =* IBM*, CIM Agent Version =* 5.0*).* Templates capture i) naming conventions ii) profiles supported iii) modeling and extension used by vendors for CIM entities – number of instances, associations iv) device control flow for configuration change action. Generation of template is a continuous process. This process is similar to the process of automatically generating DTD /XML schema from a given XML document. Schema becomes more concrete and correct when it encounters a large variety of XML documents that conforms to the schema. Template generator is validated against CIM data of devices of same type and CIM agent version with a variety and range of configurations. Template can also be generated manually by discussing the structure of devices with device manufacturers. Template generated automatically gets verified by device manufactures before being certified for use. Sample portion of a template for class IBMTSESS_StorageSystem of IBM DS8000 Storage Subsystem CIM agent version 5.1.1 is shown below:

```
<Class Name="IBMTSESS_StorageSystem"
       Namespace="/root/ibm"
       NOI=""
       CfgExp="[NUMBER_OF_STORAGE_SYSTEM]">
  <Property Name="cache" Type="string"
            Random="false">
    <Value></Value>
  </Property>
  <Property Name="Caption" Type="string"
            Random="false">
    <Value>IBM TotalStorage DS8000</Value>
  </Property>
  <Property Name="codelevel" Type="string"
            Random="false">
    <Value>5.2.0.917</Value>
  </Property>
  <Property Name="CreationClassName"
            Type="string" Random="false"
            Key="true">
    <Value>IBMTSESS_StorageSystem</Value>
  </Property>
  <Property Name="Dedicated" Type="uint16[]"
            Random="false">
    <Value>3,15</Value>
  </Property>
  ......
</Class>
```

In the example given above, `CfgExp` is a pointer to the configuration parameters specified by the user. Template for the class IBMTSESS_DiskDrive given below captures the naming conventions:

```
<Class Name="IBMTSESS_DiskDrive"
       Namespace="/root/ibm"
       NOI=""
       CfgExp="[NUMBER_OF_EXTENTS]"
       AutoGenerate="false">
  <Property Name="DeviceID" Type="string"
       Random="true" Key="true"
       Multiple="true">
    <Value>6</Value>
    <Value>6</Value>
  </Property>
  <Property Name="Name" Type="string"
    Random="false" Refers="DeviceID"
    Prefix="Disk Drive ">
  </Property>
  <Property Name="InstanceID" Type="string"
       Random="false" Key="true"
       Refers="true" Multiple="true">
    <Value External="IBMTSESS_StorageSystem"
    Prefix="" Suffix="-">Name</Value>
    <Value Prefix="" Suffix="">DeviceID
    </Value>
  </Property>
  <Property Name="TargetAddress" Type="string"
       Random="true" Prefix="" Suffix=""
       Separator="." Multiple="true">
    <Value>3</Value>
    <Value>3</Value>
    <Value>3</Value>
    <Value>3</Value>
  </Property>
</Class>
```

2.5 Data Generator

As described in sub sections 2.3 and 2.4, MOF definitions of classes and template describing the modeling structure of devices are available in knowledge repository. Based on these available capabilities, user is allowed to drag and drop devices into the visual workbench to construct a SAN. Once, the specification of SAN is obtained as user input through the workbench, data needs to be generated to support the device configurations. This component deals with generation of data based on SAN configuration specification. User specifies low level specifications for each device connected in the SAN along with the device connectivity. Consider an example scenario, where the user connects two servers to a fabric that contains four fiber channel switches and one storage subsystem. This basic SAN with a no-single-point-of-failure connectivity of server to storage is depicted in Figure-3.

Detailed configuration about each individual entity can also be specified through the visual workbench. Connectivity with device configuration created through the visual workbench generates a XML document. Detailed information in this regard is explained in Section 3.

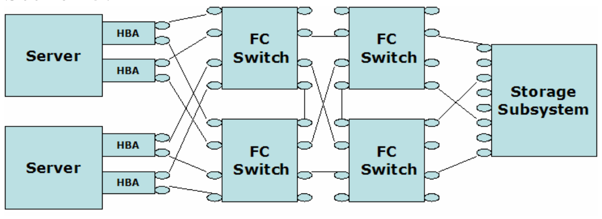

Fig. 3: Sample SAN with no single point of failure connectivity

Fiber channel port connectivity between SAN devices can be done using one of the following approaches: i) user explicitly specifies the connectivity between ports by using the visual workbench primitives ii) user can opt for a random connectivity decided by iSAN iii) output of planners can be integrated with iSAN e.g. SRM planners provides connectivity template for no-single-point-of-failure.

Based on the user input, iSAN picks the appropriate template from the knowledge base and generates the CIM data required to represent the device configurations. It is important to note that generated data for different CIM agents need to be correlated to represent the SAN in a cohesive fashion. For example, ProtocolEnpoint along with ActiveConnection information from FC switch CIM agent needs to be correlated with server and storage subsystem. LUN information from server must be correlated with Volume information from storage subsystem. Zone configuration information of fabric must be correlated with server and storage ports.

CIM instances generated by data generator component is persisted in a persistent data store called device configuration repository. Generated data could be persisted in one or multiple repositories. At this point, CIM agent becomes operational based on the supplied specification. But, data generator also continuously modifies the data in the repositories due to the control operations (e.g. creation of volume in a storage subsystem would require an addition of StorageVolume, modification of attributes of StoragePool, indication about the creation of new volume, creation of associations like AllocatedFromStoragePool etc…). Also, for support of Block Server Performance (BSP) profile and switch profile, data generator continually populates the repositories with required information using a background daemon thread.

Generated data is served back to consumers like SRM software through providers.

2.6 Device Configuration Repository

CIM information collected using snapshot approach or generated CIM information using configuration approach is persisted in one or multiple device configuration repositories. This repository can be a simply a set of files or can be a relational database. CIM instances stored in repository can be i) serialized objects on files ii) instance shredded into properties and values in file/database iii) BLOBs in database. We have used database as repository and stored CIM instances as BLOBs due to scalability and performance reasons. At a very high level, repository contains following four categories of information linked together:

- Information about the connectivity and credentials of the CIM agent.
- Information about the namespace(s) associated with the CIM agent.
- Information about CIM classes associated with the namespace.
- Information about the CIM data (CIM objectpath, CIM instance) associated with the CIM classes.

2.7 Provider

CIM providers are the pluggable modules that get plugged into the CIMOM for device operations. iSAN contains a set of providers to serve data out of the repository based on the types of open source CIMOMs. Due to the relational nature of the repository, CIM providers of iSAN also match to SQL queries. iSAN implements instance provider, association provider, method provider and indication provider.

2.8 Knowledge Repository

Knowledge Repository is a set of MOF files and templates stored persistently for use of iSAN. Based on the contained data in the repository, visual workbench exposes a support matrix of supported devices that can be used for construction of virtual SAN. A device of any given type is manufactured by multiple vendors. Vendors release device CIM agents very frequently to be in line with the fast changing CIM specifications and device capabilities. So, MOF files are stored for each device manufactured by certain vendor for a given release.

3. Experimental Evaluation

This section describes our experimental evaluations criteria. We have focused much on the correlation of different types of CIM agents that combined together represent a heterogeneous SAN. Our current types of device simulation include Server, Host Bus Adapter, Fiber Channel Switch, Storage Subsystem and Tape Library. For each of these devices, we compiled templates for multiple versions from multiple vendors. Here is a list of all the profiles that are supported :

We would like to state that performance comparison and CIMOM scalability are not part of our evaluation. This is because, we used available open source CIMOMs[17, 18]. For large scale configuration of device or for large number devices, we have used database as the backend data store. Providers also match out to SQL queries, since data store is a database. So, scalability of data store or provider is not a concern in iSAN model. Scalability and performance studies[14] of open source CIMOMs have been done in the past.

For snapshot approach, we have taken snapshot of a medium scale SAN that is used by our facility. This SAN contains three IBM DS4300, two IBM DS6000, one IBM DS8000, four Brocade Silkworm fiber channel switches, two QLogic fiber channel switches, two McData fiber channel switches and 27 servers. Thirty five CIM agents manage the SAN described above containing 41 devices. iSAN performed a parallel snapshot operation of these 34 CIM agents and stored the information in eight database instances. Three servers were used by iSAN to simulate and replay the 34 CIM agents. Time taken by iSAN snapshot routine to snapshot a live CIM agent varies based on the size on the configuration of the devices that dictates the number of CIM instances.

Without any reference to network latency and latency of the device for data generation, Fig-4 shows scalability of time taken to snapshot devices for number of instances. Using the configuration approach, we tried to simulate the above SAN with an addition of two tape libraries. Configuration of each device that was specified using the visual workbench captured information of following nature:

```
Server [
       Type = eSeries Model = 1849
```

```
        Processor = Intel Xeon
        Software = MS Office, WAS
        Manufacturer = IBM
        OS = Windows 2003
        HBA [
          Type = QLA2460 Model = CK
          Manufacturer = QLogic
          Port = 2
       ]
]
Fabric [
        ZoneConfig = [
                ZoneSet = composition
                Zone = composition
                ZoneMember = composition
                …
        ]
        FC Switch [
            Type = 4100 Model = Silkworm
            Manufacturer = Brocade
            Port = 8 NUM = 4
          ]
]
StorageSubsystem [
        Type = 800 Model = 2107
        Manufacturer = IBM
        StorageExtents = composition
        StoragePools = extent composition
        StorageVolumes = composition
        MaskingMapping = composition]

Connectivity [
  visually wired / random / planner
]
```

Data generators generated data for the 43 devices and hosted the simulated iSAN CIM agents on four servers.

For Block Server Performance profile and switch port statistics, iSAN updates the device repositories on a periodic basis.

Control functions were also implemented in iSAN providers. Examples of some of the control functions are: creation/deletion of Volumes and Pools on a storage subsystem, creation of Zone, ZoneSet, addition/removal of Zone to ZoneSet or ZoneMember to Zone etc.. Indication generation was also integrated.

Information like ProtocolEndpoint, LUN etc… exposed by the server, switch and storage subsystem CIM agents were in sync with one another.

In addition to the CIM defined indexes, iSAN indexing contributed to a faster association and reference provider with an additional one time cost of generating the indexes during snapshot. A comparative analysis is described in Fig- 5.

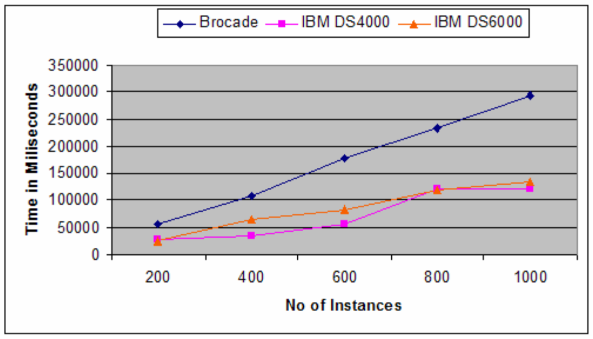

Fig. 4: Time taken to snapshot Device CIM Agents

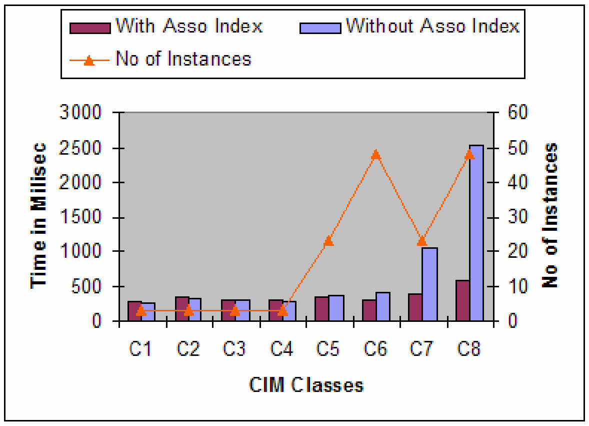

Fig. 5: Effect of iSAN indexing on Association providers

Based on the generated data, we ran the analytic tools offered by TPC to check for any best practice violation. System administrators can take snapshot of their existing environment and then extend it using iSAN to the planned environment. High level analytics like configuration analysis can then be performed on the projected environment. Visual workbench allows this mode of what-if-analysis. Otherwise, users can provide the configuration of their projected environment to iSAN to generate what-if-analysis.

4. Related Work

Simulation of networked storage and its components has been tried out at various levels.

Most of these simulations fall in the category of storage subsystem simulation. Disk system simulator DiskSim [6] helps understand modern storage subsystem performance and how storage performance relates to overall system performance and to evaluate new storage subsystem architectures. Pantheon [7] is another storage system simulator that supports modeling of wide range of I/O systems for both uniprocessor and parallel computers. IBM Object

Storage Device (OSD) simulator [8] enables the creation of self-managed, shared, and secure storage for storage networks. All these above simulators simulate the core storage subsystem.

IBM SMI-S based storage device simulator [9] is an implementation using CIM and SMI-S which mainly simulates IBM storage subsystems DS6000 and DS8000. This approach uses create/setInstances primitives to populate device repository based on user specification. But, this simulation does not provide correlated multi-device simulation for a whole networked storage environment setup.

SANBlaze [10] has attempted emulation of storage devices (disk drive, tape drive or arrays) by manufacturing target and initiator emulators for Fiber Channel, SAS and other storage protocols. Network modeling has been tried using SNMP based simulations. One of the examples of SNMP based network simulations is Adventnet's simulation toolkit [11]. Netsim [12] is a customizable simulator for high performance point-to-point workstation networks. It can be used for application level performance analysis. SANSim [13] provides simulation of FC SAN. It approaches simulation using event driven methodology by implementing various modules for workload, server, network, storage system etc.

Simulation of core entities like disk subsystems, network switches [6, 7, 12, 13] can be combined with our management module simulation for a complete device simulation. Planning or modeling tools[19, 20, 22] can be integrated with the visual workbench to transform declarative user input to required SAN blueprint. SNMP [11] and any other vendor specific management modules can be plugged into iSAN. Analytical models [15, 16, 21] can then be built into these complete device simulators for performance bottleneck analysis, application impact analysis, fault injection and analysis.

5. Conclusion

In this paper, we propose a simulation framework for storage area network management. This implementation can be used in one of the two ways (snapshot , configuration) to replicate any device management module called CIM agent. This framework removes the dependency of costly SAN hardware from Storage Resource Management suites. Development, test tool of SRM suites, what-if-analysis task of administrators are the key contributions of iSAN. Finally, this tool can be integrated with core device simulators for complete device simulation.

6. References

[1] Distributed Management Task Force (DMTF)
[2] Common Information Model (CIM)
[3] Web Based Enterprise Management (WBEM)
[4] Managed Object Format (MOF)
[5] Storage Networking Industry Association (SNIA)
[6] DiskSim Simulation Environment http://www.pdl.cmu.edu/DiskSim/
[7] HP Pantheon storage system simulator
[8] IBM Object Storage Device simulation http://www.alphaworks.ibm.com/tech/osdsim
[9] IBM SMI-S-Based Storage Device Simulator http://www.alphaworks.ibm.com/tech/smis_simulator
[10] Sanblaze - www.sanblaze.com
[11] Adventnet SNMP based network simulation
[12] M. Uysal et al., "A Customizable Simulator for Workstation Networks", Proceedings of the International Parallel Processing Symposium, April 1997.
[13] Chao-Yang Wang et al., Simulation of Fibre Channel Storage Area Network Using SANSim", Proceedings of 11th IEEE International ICON2003
[14] Sun-Mi Yoo et al., " Performance Evaluations of WBEM Implementations "
[15] Wang, Y. et al., "Execution-driven simulation of network storage systems", proceedings of Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, 2004. (MASCOTS 2004)
[16] LI Chao et al., "Using MMQ Model for Performance Simulation of Storage Area Network", proceedings of 21st International Conference on Data Engineering Workshops (ICDEW'05)
[17] Open Pegassus Project
[18] Sun WBEM Implementation
[19] Molero, X. et al., "A tool for the design and evaluation of fibre channel storage areanetworks", proceedings of 34th Annual Simulation Symposium, 2001
[20] Anderson E. et al, "Quickly finding near-optimal storage designs"
[21] Anderson, E. et al., Ergastulum: An approach to solving the workload and device configuration problem.
[22] Julie W. et al., "Appia: automatic storage area network fabric design", proceedings of USENIX FAST 2002