SELFCON: An Architecture for Self-Configuration of Networks

Raouf Boutaba, Salima Omari, and Ajay Pal Singh Virk

Abstract: **Traditional configuration management involves complex labor-intensive processes performed by experts. The configuration tasks such as installing or reconfiguring a system, provisioning network services and allocating resources typically involve a large number of activities involving multiple network elements. The network elements may be associated with proprietary configuration management instrumentation and may also be spread across heterogeneous network domains thereby increasing the complexity of configuration management.**

This paper introduces an architecture for the self-configuration of networks (SELFCON). The proposed architecture involves a directory server, which is used to maintain configuration information. The configuration information stored in the directory server is modeled using the standard DEN specification thereby allowing effective exchange of network, system and configuration management data among heterogeneous management domains. SELFCON associates configuration intelligence with the components of the network, rather than limit it to a centralized management station. The network elements are notified about related changes in configuration policies, based upon which, they perform self-configuration. SELFCON is able to provide automation of configuration management and also an effective unifying framework for enterprise management.

Index Terms: **Configuration management, self-configuration, programmable networks, Directory Enabled Networks (DEN), Lightweight Directory Access Protocol (LDAP).**

I. INTRODUCTION

Configuration management is concerned with initializing a network, maintaining relationships among network components, and the status of the network components themselves during network operation. The increase in the complexity of networks has resulted in a significant increase in configuration management activities. The present configuration management approaches involve complex labor-intensive processes. The configuration management tasks are also increased due to the presence of multiple network elements present in different management domains.

The traditional configuration methods such as the Simple Network Management Protocol (SNMP), or the Command Line Interface (CLI) are not very effective in managing the increased configuration management tasks. These traditional methods are based on the centralized manager/agent model. This approach involves centralization of configuration intelligence in a central management station like the SNMP Manager. However, an increase in the number and complexity of network elements can increase the complexity and overload the central management station. Besides, the central management station may have limited access to multiple heterogeneous management domains. The solution to these issues lies in the automation of configuration management.

This paper proposes an architecture (SELFCON) to automate configuration management by allowing self-configuration of the network. The network is composed of self-configurable network elements. A self-configurable network element has the ability to configure itself dynamically in response to associated events. The self-configuration of the network is performed using configuration data derived from network-wide configuration policies, which reflect the configuration management goals of the enterprise managing the network.

This paper emphasizes the use of a standard directory to maintain configuration information. The configuration data is described according to the standard Directory Enabled Networks (DEN) specification [1], [2]. The DEN specification is an industry initiative standardized within the Distributed Management Task Force (DMTF) to provide a uniform model representing users, profiles, applications, and network services.

The network elements register at the directory server for notification of changes related to configuration policies. Upon receiving change notifications, the network elements perform self-configuration thereby enabling the network to respond dynamically to changes in configuration policies or network state. The configuration information is replicated across multiple directory servers to allow information sharing across the network. This approach constitutes a good alternative to the traditional methods that involve distribution of configuration information among heterogeneous databases spread across multiple management domains and accessed by different protocols.

This paper is organized as follows. Section II contains an overview of configuration management. Section III describes the architecture for self-configuration of networks. Section IV describes the implementation of SELFCON using the Netscape directory server and Oplet Runtime Environment, which is used for programming of network elements. Section V contains related work. Finally, Section VI concludes the paper.

II. OVERVIEW OF CONFIGURATION MANAGEMENT

Configuration Management is concerned with the initializati-

Manuscript received July 15, 2001.

R. Boutaba, S. Omari, and A. P. S. Virk are with Department of Computer Science, University of Waterloo Waterloo, ON N2L 3G1, Canada, e-mail: {rboutaba, osa, apsvirk}@bbcr.uwaterloo.ca.

This research work was supported by research grants from Nortel Networks and the Natural Sciences and Engineering Research Council of Canada.

on, maintenance, and shutdown of individual components and logical subsystems within the total configuration of computer and communications resources of an installation. The following sub-sections discuss configuration management issues, current configuration management practices and trends in configuration management.

A. Configuration Management Issues

The present configuration management approaches involve complex labor-intensive processes. They are also error prone and costly. There are several reasons for these issues faced by traditional configuration management practices.

1. Configuration management typically involves configuration of vendor-specific network elements spread across several management domains. The network elements may have their respective vendor-specific proprietary configuration methods to effect configuration changes. Configuration management also has to deal with multi-service networks. An example of a multi-service network may be an ATM/SONET backbone integrated with IP services. Configuration management in such a scenario requires labor-intensive processes involving detailed knowledge of the network, and the respective configuration tools associated with the network.
2. The traditional configuration management methods are based on the centralized manager/agent model. This approach involves centralization of configuration intelligence in a central management station like the SNMP Manager. However, an increase in the number and complexity of network elements can increase the complexity and overload the central management station. Besides, the central management station may have limited access to multiple heterogeneous management domains.
3. The traditional configuration management approaches involve distribution of configuration data across heterogeneous databases present in several management domains. The lack of a common, unifying information repository hinders effective information exchange across the network. The traditional configuration management approaches also lack elaborate schema standardization required to effectively model the configuration of network elements and services.

B. Trends in Configuration Management

The current configuration management practices such as the Simple Network Management Protocol (SNMP) approach or the Command Line Interface (CLI) approach perform the configuration of network elements on an individual basis, which is a major drawback in the configuration and management of large scale networks and emerging network services. Configuration management in such cases is also prone to configuration inconsistencies that may result in failures or performance degradation. In general, traditional configuration practices cannot effectively provide configuration management services to large-scale networks spread across several management domains. The automation of configuration management aims to address this issue by providing effective and consistent configuration of large-scale networks. The Directory-enabled network management and Policy-based network management are emerging technologies that are being used to implement automation of configuration management.

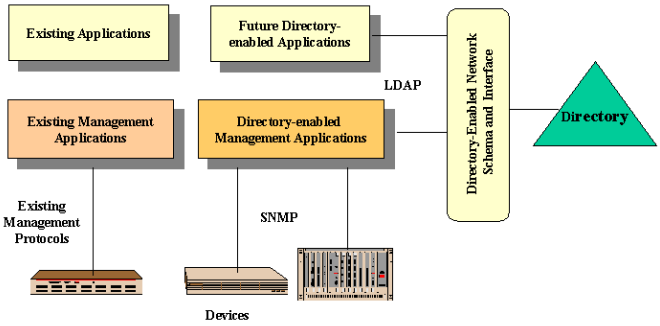

Fig. 1. Directory-enabled network management.

Directory-enabled network management involves maintaining configuration information in a special repository known as directory. The network resources (devices, operating systems, management tools, and applications) use the directory to perform the following functions–provide information about themselves, discover other resources, and obtain configuration information from other resources. The DEN environment allows heterogeneous network elements and services to function as interoperating components of the network.

Fig. 1 depicts Directory-enabled network management. The standard DEN specification is used to model the configuration information maintained in the directory. The standard network management protocols (such as SNMP) are used as the means of communication with the network elements. The Lightweight Directory Access Protocol (LDAP) [3], [4] is used to access the directory.

Policy-based network management [5], [6] uses policies to describe network behavior with a high level of abstraction. A policy is a representation of an objective to be implemented in the management domain and is applied using a set of policy rules. A policy rule is comprised of a set of conditions and a corresponding set of actions. Policy-based network management defines two key elements—Policy decision points (PDP) and Policy enforcement points (PEP). PDP is a process that makes decisions based on the policy rules. The PEP is an agent running on or within a resource that enforces the policy decision.

Policy-based network management employs the Common Open Policy Service (COPS) protocol [7] to standardize the communication between the PDP and the PEP. The policies are stored in a policy repository. The policy repository may be a directory or any other storage device such as a relational database. Fig. 2 depicts Policy-based network management.

The PDP monitors the network state by evaluating configuration data obtained from various sources such as the SNMP management station or directly from the SNMP management agents. In the latter case, SNMP communication is required between the PDP and the management agent. The proponents of the widely deployed SNMP standard have proposed the idea of improving the SNMP protocol to effectively perform dynamic network configuration.

The IETF Configuration Management with SNMP (SNMP-

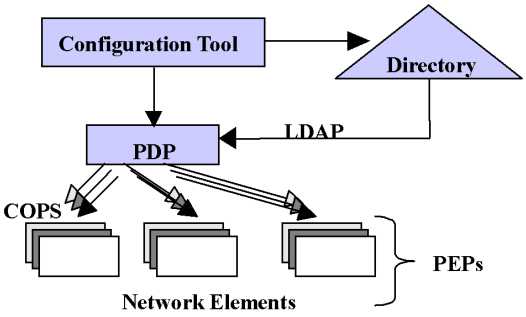

Fig. 2. Policy-based Networking.

CONF) working group has been involved in promoting SNMP as an effective protocol for dynamic network configuration. The SNMPCONF working group has been involved in developing MIB modules necessary to facilitate configuration management, specifically MIB modules which describe network characteristics that can be used by management entities making policy decisions at a network level or a device level. However, the SNMP protocol needs to address a number of issues in order to be effective for dynamic network configuration. SNMP needs to support more functionally rich operations, rather than the simple GET and SET operations. SNMP also needs to support a flexible management information model rather than the relatively rigid Structure of Management Information (SMI) model. Therefore, the IETF Next Generation Structure of Management Information (SMing) working group is involved in developing a standards-track specification for the next generation data definition language for specifying network management data. The working group has used the SMIng language developed in the IETF Network Management Research Group as a starting point. SMIng represents a superset of the SMIv2 (Structure of Management Information v2) and the SPPI (Structure of Policy Provisioning Information).

The SNMP and Policy-based network management methods involve centralization of configuration management. The configuration decisions are made by the management station in case of SNMP and by the PDP in case of Policy-based network management. The centralization of network management leads to scalability problems, particularly in case of large-scale heterogeneous networks. Although it is possible to implement Policy-based Networking (PBN) in a distributed manner, PBN aims at implementing an effective decision making architecture. The DEN specification on the other hand aims at effective modeling of the network and providing a centralized repository that can be directly accessed by the network resources in order to perform configuration management tasks.

III. SELF-CONFIGURATION OF NETWORKS

A. Directory and Directory Services

The directory is a special database designed for fast lookup of information. The directory is not a general-purpose data store and is designed to effectively store and retrieve information. The directory repository model is based on entries, each of which is a collection of attributes. Each entry is uniquely identified by its distinguished name (DN). The directory is comprised of hierarchically related objects involving parent-child relationships. This approach of modeling information is very effective in representing the physical and logical aspects of networks.

Directory service is a service that provides access to the directory. SELFCON uses the Lightweight Directory Access Protocol (LDAP) [3], [4] to access the directory. LDAP also defines an object-based data model and selection-based query language to query the information maintained in the directory. In order to implement a true exchange of network, system and service management data among heterogeneous network elements and management tools, a standard schema for description of data is required. The directory-enabled network (DEN) specification provides a solution to the problem of schema standardization [1], [2]. The DEN specification that has been standardized within the DMTF defines a set of abstract and concrete directory classes for modeling network elements, profiles and services in an appropriate manner in order to promote interoperability.

The DEN specification is an extension of the Common Information Model (CIM) of DMTF. CIM is an object-oriented approach to the management of information and systems. The DEN specification builds on CIM by defining network elements and services, along with general concepts of profiles and policies that can be used with CIM objects to model the functions and behavior of a system.

The fundamental purpose of DEN is to provide a common, unifying repository that is used to store data and information about the data (metadata) for multiple applications to share and use. The directory information is replicated among multiple directory servers using directory replication [8]. The use of a logically centralized, physically distributed repository enables DEN to define an approach to manage the network as opposed to an element in the network. This approach makes configuration management using DEN fundamentally different from and advantageous compared to other traditional approaches.

The other major advantage of using DEN to perform configuration management is the rich information model provided by the DEN specification to describe a multi-service network. The DEN specification allows many disparate services and technologies to be conceptually, logically, and physically modeled as a single cooperating platform. This feature of the DEN specification allows configuration management to be performed easily and effectively across several heterogeneous management domains.

B. Dynamic Self-Configuration of Networks

This paper proposes an architecture (SELFCON) that uses the directory service to implement dynamic self-configuration of networks. The DEN specification is used to model the heterogeneous network elements and services. Directory-based self-configuration of networks enables the network to respond dynamically to changes in the configuration policies.

The network uses two categories of configuration data to implement self-configuration. The first category of configuration data is associated with relatively static configuration parameters such as IP addresses, or routing algorithms. The second category of configuration data is associated with service provisioning information, which may include control policies such as

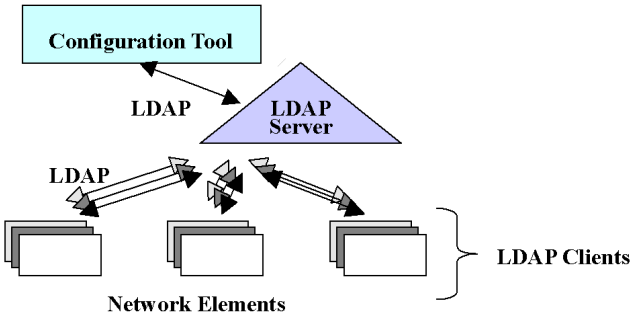

Fig. 3. Directory-based self-configuration of networks.

QoS policies for provisioning of DiffServ and IntServ services or security policies for implementing VPN services. The DEN specification is used to model both the categories of configuration data. Fig. 3 depicts the general architecture of Directory-based self-configuration of networks.

The configuration tool allows the human manager to specify configuration policies and download the configuration policies to the directory server. The standard LDAP protocol is used as the means of communication between the configuration tool and the directory and also between the network elements and the directory. The network elements are notified about any related change in the configuration policies in order to enable the network elements to perform self-configuration.

The issue of self-configuration of networks poses an interesting issue regarding the notification of changes in the configuration policies to the network elements. The network elements can obtain information about related changes by implementing any of the following two techniques—periodic polling of the directory or invoking a persistent search operation on the directory. In case of periodic polling of the directory, the network element periodically connects to the directory and refreshes it's configuration data. However, this approach suffers from the drawback of invoking significant overhead on the network elements, directory server and network resources.

The other approach that can be implemented by the network elements is to invoke the persistent search operation on the directory. The persistent search operation [9] alters the standard LDAP search operation so that it does not end after the initial set of entries matching the search criteria are returned. Instead, the LDAP server keeps the search operation active. This approach provides LDAP clients and servers participating in persistent search an active channel through which entries that change (and additional information about the changes that occur) can be communicated. The persistent search operation suffers from the drawback that it requires an active TCP connection, which otherwise may not have been kept active. The maintenance of an active TCP connection between the LDAP clients and the server constitutes undesirable overhead at the server level and also results in increased utilization of network resources, particularly in cases where persistent search is invoked by a large number of LDAP clients.

SELFCON provides dynamic notification of changes in the directory to the network elements without requiring an active LDAP session. LDAP has the important feature of providing mechanisms for enhancing the base set of services offered by

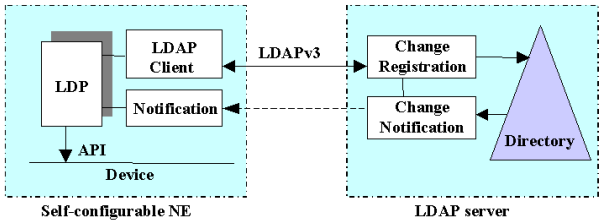

Fig. 4. Functional architecture of SELFCON.

LDAP. An LDAP control is a mechanism that allows additional parameters to be added to previously defined LDAP operations. An LDAP extended operation is a mechanism that allows for new LDAP operations to be defined to enhance the base set of LDAP operations. For example, LDAP extensions have been developed to provide dynamic directory services [10]. The dynamic directory services store information that only persists in its accuracy and value when it is being periodically refreshed.

We have developed LDAP extended operations to implement the dynamic notification mechanism. The LDAP extended operations provide change registration and change notification services to the network elements. The network elements use the change registration extended operation to register at the directory server for change notification. The registration is followed by the termination of the LDAP session. The network elements specify the directory entries that are to be monitored as part of the change registration extended operation.

The directory server maintains a list of all valid registration requests and associates the requests with the related network elements. The directory server monitors the directory information tree (DIT) for any changes. The changes to the DIT are compared to the change registration requests registered at the server. If the changes made to the DIT match the criteria specified by a change registration request, then the directory server performs the change notification extended operation. The change notification extended operation notifies the related network element(s) regarding the associated changes made to the DIT, which may represent changes in the configuration policies.

The change notification extended operation may be implemented using two different models—push model or pull model. The pull model involves notifying the network element about the changes to the DIT. The network element is required to reconnect to the directory server in order to access the changed configuration information. The push model of change notification involves automatic transfer of the changed information from the directory server to the network element. The push model has the advantage of reducing the response time to changes in configuration information, as it does not involve the LDAP connection setup and data request associated with the pull model of change notification. The change registration extended operation allows the network element to specify the change notification method—pull model or push model. This allows the network element to implement the change notification approach most suited with regards to the frequency of change of configuration information and importance of the changed configuration information.

Fig. 4 depicts the functional architecture of SELFCON. The

directory server includes the functionality to implement the change registration and change notification mechanisms. The self-configurable network element consists of three important functional units — the LDAP client, notification unit, and the Local Decision Point (LDP).

The LDAP client allows the network element to receive initial configuration information from the directory at the time of startup. The LDAP client also enables the network element to register change registration requests at the directory server. If the changes made to the directory match the criteria specified by a change registration request, then the directory server performs change notification. The directory server notifies the notification unit of the concerned network element(s) about the changes in the configuration information. In case of the pull model of change notification, the notification unit informs the LDAP client, which in turn establishes an LDAP session with the directory server and retrieves the changed configuration information. The LDAP client transfers the changed configuration information to the LDP. In case of the push model of change notification, the directory server provides the notification unit with the changed configuration information, which in turn passes the information to the LDP.

The DEN specification involves a set of classes that are applicable to all management domains, such as physical or logical aspects of managed entities or other aspects that are part of a managed environment. The DEN specification also involves classes related to specific management domains such as System, Device, Network, Database, User, and Service Level Agreement, etc. The network element may subscribe to changes in properties that represent the element or to properties related to the domain or network in accordance with its configuration goals.

The LDP configures the network element in accordance with the changed configuration policies. The LDP uses the appropriate application programming interface (API) to implement the self-configuration of the network element. The LDP may use SNMP API to configure the management information base (MIB) of the device or may use COPS API to implement changes in the policy information base (PIB) of the network element.

The LDAPv3 protocol provides means to implement secure transfer of information between the LDAP server and the LDAP client [11]. The notification messages may be transmitted using any suitable data communications protocol. The LDP uses network programming [12] to implement the self-configuration of the network elements by downloading and installing appropriate software. The standard network programming interfaces such as IEEE P1520 APIs can be used for this purpose [13], [14].

IV. IMPLEMENTATION

The prototype of the SELFCON system was built using the Oplet Runtime Environment [15]. The Oplet Runtime Environment (ORE) supports dynamically injecting customized software services into network devices. The implemented architecture is composed of an embedded Java Virtual machine (JVM) and the ORE. The ORE component provides the substrate on the network device to support the secure downloading, installation and safe execution of services. Since the ORE and its services are constrained to running in the JVM, system stability of the core network device operations is not affected.

Customized ORE services, which run locally on network devices, include monitoring, routing, diagnostic, or other user specified functions. ORE services can monitor and change specific Management Information Base variables locally on the device through the Java MIB API. The direct access to MIB variables on network nodes greatly improves scalability and reduces network traffic compared with using SNMP manager-agent communication [16].

The ORE architecture consists of the ORE environment, oplets and services. Oplets are self-contained downloadable units that encapsulate one or more services, service attributes, authentication information, and resource requirements. Oplets can provide services to other oplets and can depend on other services. The ORE provides means to download oplets, manage the oplet lifecycle, maintain a repository of active services, and track dependencies between oplets and services.

The ORE services use the Java Forwarding API (JFWD API) to instruct the forwarding engine regarding the handling of packets. The JFWD API is a uniform, platform-independent portal through which application programmers control the forwarding engines of heterogeneous network nodes (e.g., switches and routers). The JFWD implementation across multiple hardware platforms allows abstraction of a platform-neutral forwarding engine and a framework within which new protocols and control data can be described [15].

During the network element initialization, the network element downloads a "self-configuration oplet." The self-configuration oplet installs the following services on the network element—LDAP client, notification unit, and the Local Decision Point (LDP). The LDAP client connects to the LDAP server and downloads the initial configuration information. The LDAP client also registers its interest for change notification related to the configuration information maintained at the directory server. The notification unit expects a change notification from the directory server. The notification unit accepts socket communication from the remote directory server. The LDP is responsible for implementing the changes in the configuration information. The LDP uses SNMP API or JFWD API to implement the new configuration data.

We have used the Netscape directory server [17] to implement the directory. The configuration information is represented in the directory using the standard DEN specification. Fig. 5 depicts the prototype implementation of the Directory-based self-configuration of networks along with the information exchange among the various units.

We have extended the functionality of the Netscape directory server with two server plug-ins, providing the pre-operation and post-operation functions. The directory server, at start-up, loads the server plug-ins and appropriately accesses the plug-in functions during the processing of LDAP operations. The pre-operation function allows the network element to register for a change notification. The post-operation function allows the directory server to notify or send the configuration data to the notification unit of the network element(s).

If the notification unit receives configuration data from the directory server, then it passes the data to the LDP. If the no-

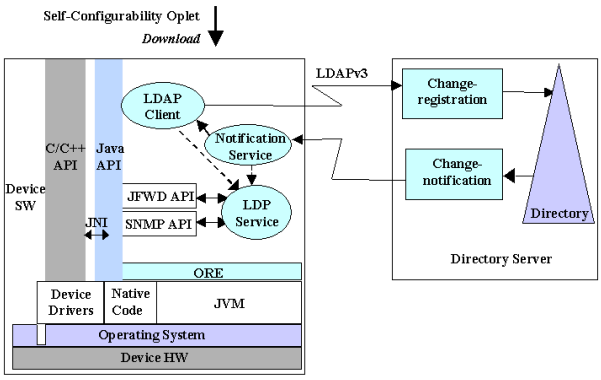

Fig. 5. Prototype for Directory-based self-configuration of networks.

tification unit receives a notification from the directory server, it informs the LDAP client about the notification. The LDAP client initiates a LDAP session with the directory server in order to retrieve the changed configuration information. The LDAPv3 protocol used for transfer of information between the LDAP client and the directory server provides sufficient means for secure transmission of data [11]. The LDAP client upon receiving the configuration information terminates the LDAP session, thereby promoting effective utilization of network and computing resources. The LDP uses the JFWD API or the SNMP API to implement the new configuration data. This process of self-configuration is repeated by the network element whenever there is a related change in the configuration policies.

The functional units of SELFCON—LDP, LDAP client, and notification unit—are being evolved in order to implement advanced configuration management tasks. The LDP is being evolved to support complex inference rules, methods to handle new configuration data structures and effective means to interpret and enforce emerging configuration policies.

SELFCON is also limited by the fact that the LDAP directory lacks support for standard database functionality such as triggers, which may be very crucial for configuration management. SELFCON is being evolved to incorporate the standard Event-Condition-Action (ECA) model [18]. The network administrator using the configuration tool or the network elements can specify ECA requests in place of change registration requests. The ECA requests specify the triggers to be fired when a directory entry is affected, that is, read or written by an operation. In general, the ECA requests specify an Event to monitor, a Condition to check, and an Action to perform if the event occurs and the condition holds. The implementation of the Event-Condition-Action model shall facilitate effective implementation of advanced configuration management tasks such as provisioning network resources, allocating network resources, reporting, managing end-to-end security and offering customized features to network elements and services.

V. RELATED WORK

SELFCON uses the standard DEN specification to model network elements and services. The DEN specification is an extension of the Common Information Model (CIM) of DMTF. CIM is an object-oriented approach to the management of information and systems. There have also been other object-oriented approaches to model networks. The OSI CMIP proposal was based on object-oriented models to organize instrumentation of managed resources at agent's [19]. NETMATE [20], Dolphin [21], and NESTOR [22] have also used object-based approaches to model networks. There have also been commercial products such as HP OpenView and Tivoli TME that have used object-oriented resource models to develop management applications.

The NESTOR system [22] provides an architecture for network self-management and organization. NESTOR automates configuration management by using policy scripts that access and manipulate respective network elements via a resource directory server (RDS). RDS pushes configuration changes to the network elements using a layer of adapters that translate operations on its object-relationship model to actions on the respective elements. The configuration models in the NESTOR system are expressed using the Resource Definition Language (RDL). SELFCON is based on the DEN specification, which is being widely used as the standard to model network elements and services. Besides, SELFCON associates configuration intelligence with the components of the network. The network element is able to perform self-configuration in response to changed configuration policies.

The MANDATE (MAnaging Networks using DAtabase TEchnology) system [23] is a database system designed for effective management of large enterprise networks. The MANDATE design makes an attempt to take advantage of the special characteristics of network data and transactions, and of database technology, to effectively derive the required management functionality. SELFCON improves upon the MANDATE design by using the directory as the data store in place of the database. A directory is better suited to model the heterogeneous nature of networks and to be a common information repository for network elements and services. Besides, the directory is the basis of the standard DEN specification.

In [24], the directory has been used to provide a framework for Directory-enabled SNMPv3 management. The network elements, that are typically SNMP agents access the configuration data maintained in the directory and translate the data into SNMP MIB attributes. This architecture suffers from the drawback that, the network elements perform only initial configuration tasks and do not support dynamic self-configuration.

In [25], the directory is used to maintain configuration information related to the network elements and services. This architecture is used to provide self-healing system capability to the network—the configuration information maintained in the directory is used to configure network elements, only in cases involving damage to the related network management stations. The network elements do not possess the ability to perform self-configuration.

VI. CONCLUSION

The manual processes presently involved with network management are quickly reaching their limits as networks become more complex and implement emerging services. This paper proposes an architecture that combines several techniques such as object modeling using the standard DEN specification, di-

rectory services, and network programming to provide self-configuration of networks. This feature of self-configuration addresses several configuration issues encountered in the most commonly used configuration methods and protocols, such as CLI, SNMP and COPS.

The configuration information is maintained in a standard directory. The configuration information maintained in the directory is modeled using the standard DEN specification. The network elements register for change notification at the directory server using a directory service. The directory server notifies the network elements about any related changes in the configuration information. The network elements upon receiving the changed configuration data are able to perform self-configuration. SELFCON allows increased scalability, as the network elements are able to automatically adapt themselves to the new configuration policies. SELFCON is able to eliminate configuration inconsistencies, reduce configuration management tasks, and is able to provide a unifying framework for effective enterprise management.

The functional units of SELFCON—LDP, LDAP client, and notification unit—are being evolved in order to implement advanced configuration management tasks. SELFCON is also being evolved to incorporate the standard Event-Condition-Action (ECA) model in order to support triggers, which may be very crucial for implementation of advanced configuration management tasks such as provisioning network resources, allocating network resources, reporting, managing end-to-end security and offering customized features to network elements and services.

REFERENCES

[1] John Strassner, *Directory Enabled Networks*, Macmillan Technical Publishing, 1999.

[2] DMTF directory enabled networks (DEN) initiative. Available at http://www.dmtf.org/standards/standard_den.php.

[3] M. Wahl, T. Howes, and S. Kille, "Lightweight directory access protocol (v3)," IETF, RFC 2251, Dec. 1997.

[4] Mark Wilcox, *Implementing LDAP*, Wrox Press, 1999.

[5] Wang Changkun, "Policy-based network management," in *Proc. Int. Conf. Commun. Technol.*, vol. 1, 2000.

[6] Y. Nomura *et al.*, "A policy based networking architecture for enterprise networks," in *Proc. IEEE Int. Conf. Commun.*, 1999.

[7] D. Durham *et al.*, "The COPS (common open policy service) protocol," IETF, RFC 2748, Jan. 2000.

[8] IETF LDAP duplication/replication/update protocols (ldup) working group. Available at http://www.ietf.org/.

[9] M. Smith *et al.*, "Persistent search: A simple LDAP change notification mechanism," IETF, Internet Draft, Nov. 2000.

[10] Y. Yaacovi, M. Wahl, and T. Genovese, "Lightweight directory access protocol (v3): Extensions for dynamic directory services," IETF, RFC 2589, May 1999.

[11] M. Wahl *et al.*, "Authentication methods for LDAP," IETF RFC, 2829, May 2000.

[12] D. Tennenhouse *et al.*, "A survey of active network research," *IEEE Commun. Mag.*, vol.35, no. 1, 1997.

[13] "Programming interfaces for IP networks," *White Paper IEEE P1520/TS/IP001*. Available at http://www.ieee-pin.org/doc/draft_docs/IP/p1520tsip001-rev0.4.pdf.

[14] "Programming interfaces for IP routers and switches, an architectural framework document," *IEEE P1520/TS/IP003*. Available at http://www.ieee-pin.org/doc/draft_docs/IP/p1520tsip003-04dec98.pdf.

[15] R. Jaeger *et al.*, "Dynamic classification in silicon-based forwarding engine environments," *IEEE LAN/MAN Workshop*, Australia, 1999.

[16] R. Jaeger, T. Lavian, and J. Hollingsworth, "Open programmable architectures for java-enabled network devices," *Stanford Hot Interconnects*, Aug. 1999.

[17] Netscape directory and security products by netscape marketing. Available at http://home.netscape.com/directorysecurity/index.html.

[18] U. Dayal *et al.*, "The HIPAC project: Combining active databases and timing constraints," *SIGMOD Record, 17(1)*,Mar. 1988.

[19] ITU-T, "Information technology, open systems interconnection, structure of management information: Guidelines for the definitions of managed objects," *Recommendation X.772, ISO/IEC 10165-4*, 1992.

[20] A. Dupuy *et al.*, "NetMate: A network management environment," *IEEE Network Mag.*, 1991.

[21] A. Pell *et al.*, "Managing in a distributed world," in *Proc. 4th IFIP/IEEE Int. Symp. Integrated Network Management*, 1995.

[22] Y. Yemini, A. Konstantinou, and D. Florissi, "NESTOR: An architecture for network self-management and organization," *IEEE J. Select. Areas Commun.*, vol. 18, no.5, May 2000.

[23] J. Haritsa *et al.*, "MANDATE: Managing networks using database technology," *IEEE J. Select. Areas Commun.*, vol. 11, 1993.

[24] S. Omari and R. Boutaba, "Directory supported management with SNMPv3," in *Proc. 10th IFIP/IEEE International Workshop on Distributed Systems Operation and Management (DSOM'99)*, Zurich, Switzerland, Oct. 1999.

[25] B. Cheng *et al.*, "Directory-enabled network management framework for battlefield networks," in *Proc. MILCOM'99*, Atlantic City, 1999.

Raouf Boutaba teaches networks and distributed systems as a professor in the Department of Computer Science of the University of Waterloo since 1999 and conducts research in integrated network and systems management, wired and wireless multimedia networks, and quality of service control in the Internet. He is the program chair of the technical committee on information infrastructure of the IEEE Communications Society since 2000 and the chairman of the IFIP working group on network and distributed systems management since 2001. Dr. Boutaba is the recipient of the Premier's Research Excellence Award in 2000.

Salima Omari is a Ph.D. candidate at the University of Versailles in France. Her research interests include policy-based management and self-managed networks.

Ajay Pal Singh Virk received his Bachelor of Engineering degree in Computer Engineering from Thapar Institute of Engineering and Technology, Patiala, India, in 1997. He is currently pursuing his Masters in Computer Science at the University of Waterloo, Canada. His research interests include Directory-based configuration of networks and Directory-enabled network services.