Eine Netzwerkmanagement-Plattform basierend auf der Tool Command Language (Tcl)

J. Schönwälder H. Langendörfer

Institut für Betriebssysteme und Rechnerverbund
TU Braunschweig
Bültenweg 74/75
D-38106 Braunschweig, Germany
E-mail: schoenw@ibr.cs.tu-bs.de

Zusammenfassung

Im Rahmen eines Forschungsprojekts zum Thema Netzwerkmanagement wird an der TU Braunschweig eine Netzwerkmanagement-Plattform entwickelt, die auf der Tool Command Language (Tcl) basiert. Durch die Verwendung einer Kommandosprache als internes Programmiermodell wird dem System- bzw. Netzwerkadministrator eine Software-Umgebung bereitstellt, die sich leicht zur Lösung der jeweils aktuellen Probleme anpassen läßt. Insbesondere werden die Benutzer des Systems in die Lage versetzt, das System entsprechend der lokalen Gegebenheiten um eigene Applikationen zu erweitern. In diesem Artikel stellen wir die Grundkomponenten Tkined und Scotty vor, erläutern das Zusammenspiel einzelner Komponenten und diskutieren den Aufbau verteilter Management-Applikationen mit Tkined und Scotty.

1 Einleitung

Der Betrieb von Rechnernetzen hat sich in den letzten Jahren durch die rasante Verbreitung von netzwerkfähigen Endgeräten zu einem entscheidenden Faktor für die erfolgreiche Informationsverarbeitung in Unternehmen und Forschungseinrichtungen entwickelt. In Netzwerken, die aus mehreren hundert oder tausend Knoten bestehen, ist die Kontrolle und Optimierung der erreichten Dienstgüte nur durch den Einsatz von speziellen Netzwerkmanagement-Werkzeugen möglich. Angesichts der Vielfalt von verschiedenen Hard- und Softwarekomponenten in einem Rechnernetz müssen Netzwerkmanagement-Systeme ein hohes Maß an Interoperabilität, Flexibilität, Integrationsfähigkeit und nicht zuletzt Performanz aufweisen.

Neben vielen speziellen Hilfsmitteln zur Fehlersuche etc. sind Management-Werkzeuge meist in eine Management-Plattform integriert, die als Trägersystem alle für das Management notwendigen Informationen (über Netzwerkmanagement-Protokolle wie SNMP oder CMIP und eine Konfigurationsdatenbank) zur Verfügung stellt und eine von Netzwerkmanagement-Applikationen benutzbare graphische Oberfläche anbietet (Abbildung 1). Weit verbreitete Systeme dieser Art sind z.B. Cabletron Spectrum, NetView/6000, HP OpenView und SunNet Manager [5].

Die in der Netzwerkmanagement-Plattform ablaufenden Applikationen werden meist von Drittanbietern geliefert. Dies sind entweder die Hersteller von Netzwerkkompo-

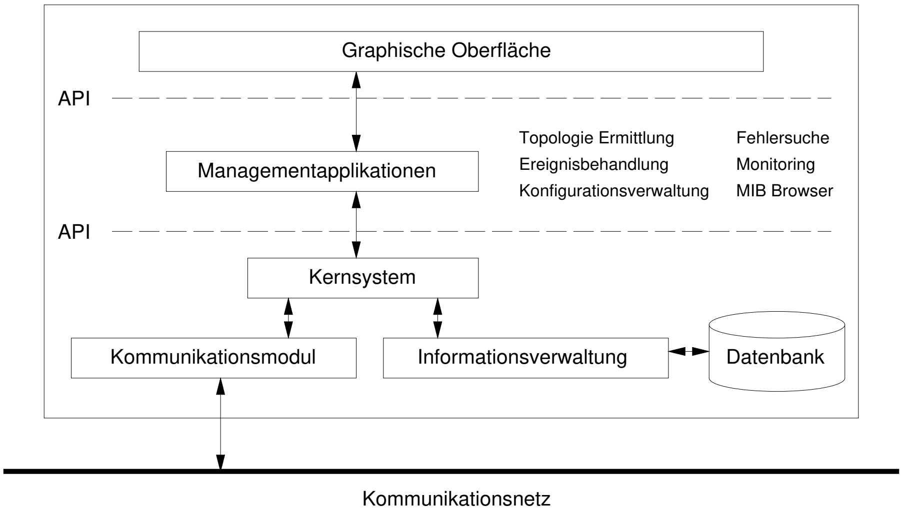

Abbildung 1: Aufbau einer Netzwerkmanagement-Plattform

nenten selbst oder Firmen, die sich auf die Entwicklung von Netzwerkmanagement-Applikationen spezialisiert haben.

2 Architektur

Die meisten verfügbaren Netzwerkmanagement-Plattformen bieten zur Integration neuer Applikationen entsprechende Programmierschnittstellen in der Sprache C an. Diese Schnittstellen manipulieren typischerweise recht komplexe Datenstrukturen (z.B. XOM/XMP [2]) und erfordern oftmals spezielle Entwicklungsumgebungen und -lizenzen. In einigen Systemen kann auch die Stabilität und Performanz des Gesamtsystems durch neue, fehlerhafte oder schlecht geschriebene Applikationen negativ beeinflußt werden.

Wir verfolgen im Gegensatz dazu die Idee, Netzwerkmanagement-Applikationen in einer interpretierten Sprache zu implementieren. Dienste der Netzwerkmanagement-Plattform werden durch spezielle Sprachkonstrukte bereitgestellt. Komplexe Datenstrukturen, wie sie beispielsweise bei der Manipulation von ASN.1 [14] Werten entstehen, werden auf einfache String-Darstellungen und Listen reduziert. Applikationen laufen als eigenständige miteinander kommunizierende Prozesse, so daß Programmierfehler nicht die Stabilität des Gesamtsystems beeinflussen.

Als Interpretersprache wird die ursprünglich an der University of Berkeley entwickelte Tool Command Language (Tcl) benutzt [9]. Dabei handelt es sich um einen Kommandointerpreter, der sich insbesondere durch eine klar definierte Schnittstelle sehr leicht um neue Sprachkonstrukte erweitern läßt. Die wichtigsten Gründe zur Wahl von Tcl als Interpretersprache waren

- die hohe Portabilität (POSIX) und die freie Verfügbarkeit,
- die leichte Erlernbarkeit des Tcl Sprachumfangs, da Kommandosprachen jedem Computerbenutzer mehr oder weniger vertraut sind,

– die hervorragende C Schnittstelle zur Einbettung neuer Kommandos und

– die Verfügbarkeit des auf Tcl basierenden Tk Toolkits.

Die Schwächen von Tcl im Vergleich zu anderen Sprachen wie Scheme [3], Python [15] oder Perl [16] (geringe Laufzeiteffizienz, fehlendes Modulkonzept, keine zusammengesetzten Datentypen, etc.) waren für uns nur von geringer Bedeutung, da wir durch die Definition geeigneter neuer Kommandos derartige Probleme in den Griff bekommen können.

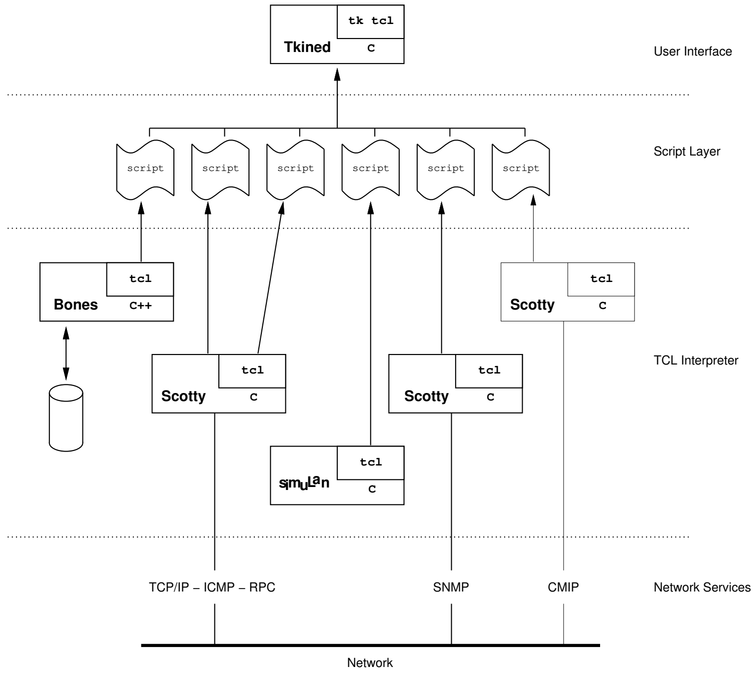

Abbildung 2: Architektur

Die Architektur unseres Systems ist in Abbildung 2 dargestellt. Der Benutzer interagiert mit der graphischen Oberfläche Tkined, die als Netzwerkeditor die Darstellung eines Rechnernetzes erlaubt. Netzwerkmanagement-Applikationen werden als Tcl-Skripte formuliert und werden von speziellen Tcl-Interpretern abgearbeitet, die über Erweiterungen zum Zugriff auf verschiedene Informationsquellen im Netzwerk verfügen. Applikationen registrieren die von ihnen angebotenen Funktionen normalerweise als Menü beim Tkined Editor. Über diese Menüs kann der Benutzer die angebotenen Funktionen auf die im Editor selektierte Menge von Objekten anwenden.

Die Tcl-Interpreter besitzen Erweiterungen zum Zugriff auf Datenbanken, auf Netzwerkdienste oder auf andere Applikationen wie z.B. einen Netzwerksimulator. Im nächsten Kapitel wird die graphische Oberfläche Tkined bschrieben, und in Kapitel 4 werden einige Tcl-Erweiterungen des Scotty Interpreters erläutert. Dem Aufbau verteilter Management-Applikationen mit Hilfe von Scotty widmet sich Kapitel 5. Eine kurze Zusammenfassung und ein Ausblick befindet sich in Kapitel 6.

3 Tkined

Die graphische Oberfläche unserer Plattform besteht aus einem Netzwerkeditor, der die interaktive Erstellung von Netzwerkplänen erlaubt und in seiner Bedienung objektorientierten Zeichenprogrammen ähnelt. Tkined bietet dazu folgende Objekt-Typen an: `NODE` Objekte repräsentieren Netzwerkknoten (z.B. Rechner, Bridges, Sternkoppler) und `NETWORK` Objekte das eigentliche Kommunikationsmedium (Ethernet Kabel, X.25 Leitungen, Funkstrecken). Verbindungen zwischen `NODE` Objekten und `NETWORK` Objekten werden durch `LINK` Objekte dargestellt.

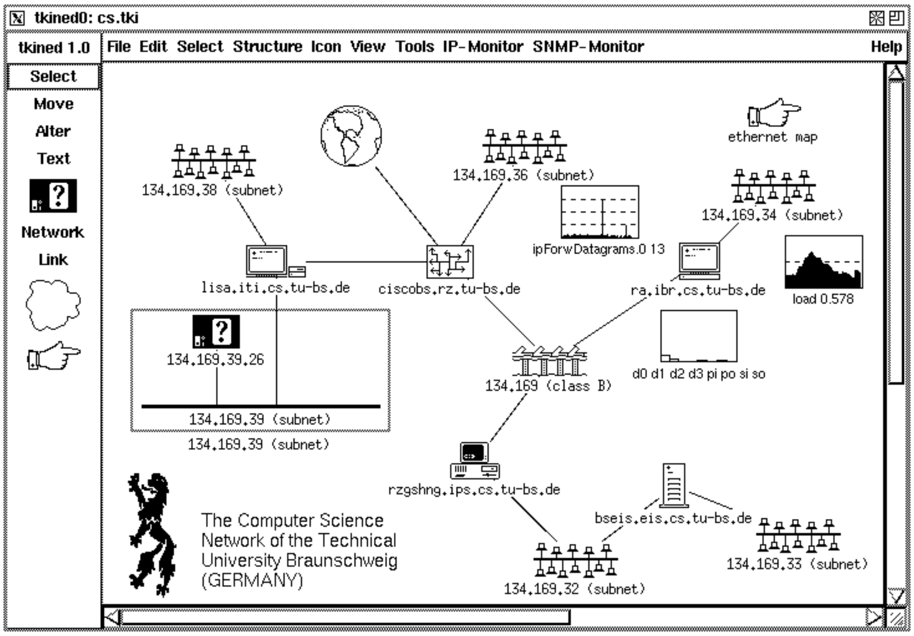

Abbildung 3: Tkined Schnappschuß

In Abbildung 3 sieht man ein `NETWORK` Objekt mit der Beschriftung "134.169.39 (subnet)" und diverse `NODE` Objekte, die der realen Form der Geräte nachempfunden sind. Dünne Linien kennzeichnen Verbindungen (`LINK` Objekte). Die Verbindungen werden automatisch aktualisiert, wenn die an den Endpunkten befindlichen Objekte bewegt werden.

Zur Strukturierung von großen Plänen dienen `GROUP` Objekte, die andere Tkined Objekte enthalten können und entweder als Icon oder als umschließendes Rechteck dargestellt werden. Abbildung 3 zeigt die Teilnetze 134.169.38, 134.169.36 und 134.169.34 als iconisierte `GROUP` Objekte. Die Gruppe mit dem Teilnetz 134.169.39 ist expandiert dargestellt. Zur Verwaltung sehr großer Pläne bieten sich außerdem `REFERENCE` Objekte an, die einen Verweis auf eine weitere Tkined Datei enthalten (ähnlich den Links in Hypertext-Systemen). Verweise können auch auf Dateien zeigen, die via anonymous FTP erreichbar sind.

Zur Beschriftung eines Netzwerkplans stehen `TEXT` Objekte zur Verfügung. Mit `IMAGE` Objekten können beliebige Grafiken in den Editor geladen werden. Sämtliche Objekte können in der Farbe, in der Schriftgröße oder dem zu verwendenden Icon beliebig konfiguriert werden.

Zur Darstellung von einfachen Graphen oder Balkendiagrammen dienen `STRIPCHART` und `BARCHART` Objekte. Sie erlauben eine platzsparende Darstellung von Daten, die den Zustand des Netzes beschreiben. In Abbildung 3 zeigen zwei `STRIPCHART` Objekte die Auslastung eines Knotens und die Anzahl weitergeleiteter IP Packete an. Ein `BARCHART` Objekt zeigt die Platten-Aktivitäten an diesem Knoten. Weitere Tkined Objekte dienen Applikationen zur Erzeugung neuer Menüs (`MENU`), zum Start von Applikationen (`INTERPRETER`) oder zur Kontrolle eines Augabefensters (`LOG`).

Zu den Objekten lassen sich beliebige Attribute definieren. Ausnahme bilden `LINK` Objekte, die lediglich eine Verbindung anzeigen und keine weitere Information tragen. Die Attribute können von Applikationen benutzt werden, um Parameter oder Statusinformationen an einem Objekt zu hinterlegen. Attribute werden automatisch vom Tkined Editor mit dem Netzplan abgespeichert und beim Laden restauriert. Applikationen können beim Tkined Editor zusätzlich Informationen hinterlegen, die ihnen nach dem Laden eines Netzplans wieder zugestellt werden sollen. Dies ermöglicht den automatischen Neustart von Applikationen beim Laden von Tkined Dateien.

4 Scotty

Hinter dem Namen Scotty verbirgt sich ein Tcl-Interpreter, der als Grundlage von einer Reihe von Applikationen dient. Die Applikationen umfassen Verfahren zur Topologieermittlung in TCP/IP Netzen [11], Monitore zur Überwachung von Rechner-Aktivitäten, Browser zum Zugriff auf Netzwerkinformationen und Skripte zur Ereignisfilterung.

Der Scotty Interpreter besteht im wesentlichen aus Tcl-Erweiterungen, die Schnittstellen zu den TCP/IP Protokollen ICMP, TCP, UDP, NTP und SNMP [4, 13] bereitstellen. Außerdem ist der Zugriff auf das Domain Name System (DNS) und einige weit verbreitete Sun RPCs möglich. Eine Schnittstelle zum CMIP Protokoll [7, 6] der ISO/OSI ist ebenfalls vorhanden. Sie basiert auf dem OSIMIS Packet [8], das ihrerseits auf das ISO Development Environment (ISODE) [10] aufsetzt und im Vergleich zur SNMP Implementation recht groß ist.

Der Scotty Interpreter läuft normalerweise in einer Schleife und wartet auf das Eintreffen von Ereignissen. Ereignisse können durch das Ablaufen von Uhren oder durch das Eintreffen von Nachrichten ausgelöst werden und führen in der Regel zur Abarbeitung eines Tcl Kommandos. Diese ereignisgesteuerte Art der Programmierung ist allen Entwicklern von graphischen Oberflächen wohl vertraut und erweist sich hier als extrem angenehm, da auf diese Art und Weise mehrere Abläufe parallel verfolgt werden können, ohne ein komplettes Thread-System zu benötigen. Auf die grundlegende Ereignisschleife wurde ein einfacher Job-Scheduler aufgesetzt, da periodisch zu wiederholende Kommandos sehr häufig auftreten.

Beim Entwurf der neuen Tcl Kommandos wurde darauf geachtet, möglichst einfache und in sich konsistente Schnittstellen zu entwerfen, die sich in Tcl gut einfügen. So liefern Kommandos zur Erzeugung von TCP Verbindungen ganz normale Tcl Filehandle, die mit den üblichen Tcl Kommandos manipuliert werden können. Außerdem war es ein Entwurfsziel, Redundanzen und unnötige Optionen zu vermeiden.

Die Erreichung dieser Ziele hat insbesondere bei dem Entwurf der Schnittstelle zum Simple Network Management Protocol (SNMP) längere Diskussionen ausgelöst. Das SNMP Protokoll benutzt zur Adressierung ASN.1 Object Identifier, die in ihrer Dezimaldarstellung nur schwer lesbar sind. Nur durch das Zusammenspiel mit einem Modul, das Object Identifier in lesbare Namen und zurück verwandelt, lassen sich Quelltexte lesbar gestalten. Die Frage, welche Umwandlung in welchem Kontext sinnvoll ist, konnte erst nach mehreren Revisionen und durch Erfahrungen mit anderen SNMP Schnittstellen befriedigend gelöst werden.

Die so entstandene Tcl Schnittstelle zu SNMP besteht aus einem Kommando zum Erzeugen von sogenannten Sessions und einem Kommando zum Zugriff auf die statische Beschreibung einer Management Information Base (MIB). Das `mib` Kommando arbeitet auf einer internen Darstellung des MIB Baums, die aus den ASN.1 Definitionen aufgebaut wird. Ein eingebetteter Parser konvertiert die ASN.1 Beschreibung in das interne Format. Um die Zeiten zum Parsen der MIB Beschreibungen zu reduzieren, wird ein komprimiertes Format der MIB Definition wieder abgespeichert und beim nächsten Laden benutzt, falls die Original Definition nicht ein neueres Datum besitzt. Dadurch ließ sich die Ladezeit selbst für mehrere tausend Variablen kurz halten, ohne an Flexibilität zu verlieren.

Bei der Implementation der SNMP Protokolloperationen wurde von vornherein die Unterstützung von SNMP Version 1 und Version 2 berücksichtigt. Durch die direkte Umwandlung von SNMP Variable-Bindings in Tcl Listen konnte eine gute Performanz erreicht werden, da das Erzeugen von entsprechenden C Strukturen entfällt. Sämtliche SNMP Operationen werden auf einer Session ausgeführt. Die Parameter der Session bestimmen dabei die zu benutzende Version und legen die Quelle und das Ziel des Nachrichtenaustauschs fest. Die meisten der SNMP Operationen können asynchron ausgeführt werden, indem beim Kommandoaufruf ein Tcl Kommando angegeben wird, das beim Eintreffen der Antwort oder beim Auftreten eines Fehlers ausgeführt wird.

Die Scotty SNMP Erweiterung kann auch als erweiterbarer SNMP Agent verwendet werden. In diesem Fall werden SNMP Variablen an Tcl Variablen gebunden, so daß Manipulationen von Tcl Variablen automatisch als Änderungen von SNMP Variablen sichtbar werden. Durch die Definition von Tcl Variablen Traces können Änderungen von SNMP Variablen durch Set-Operationen zur Ausführung von Tcl Prozeduren führen. Damit ist die Programmierung von schreibbaren MIB Variablen möglich.

5 Verteiltes Management

Die bisher beschriebenen Komponenten Tkined und Scotty sowie die darauf aufbauenden Skripte realisieren das klassische Organisationsmodell, in dem eine zentrale Station die Rolle des Managers übernimmt, die die übrigen Komponenten eines Zuständigkeitsbereichs (Domäne) kontrolliert. Dieser zentrale Ansatz hat natürlich einige Nachteile:

- Die Kontrolle und Steuerung des Netzwerks hängt von der Verfügbarkeit einer zentralen Maschine ab.

- Netzwerkpartitionierungen koppeln unter Umständen ganze Teilnetze vom Management ab.
- Langfristige Aufgaben wie die Erfassung von Trend-Statistiken erfordern einen durchgängigen Betrieb der Managementstation (Probleme mit Backups etc.).
- Durch die Verdichtung von Rohdaten (z.B. die Anzahl an einer Schnittstelle empfangener und gesendeter Bytes) zu abgeleiteten Größen (z.B. der Last auf einer Schnittstelle) und die Überwachung von Grenzwerten durch das zentrale System wird das Kommunikationssystem selbst relativ stark belastet.

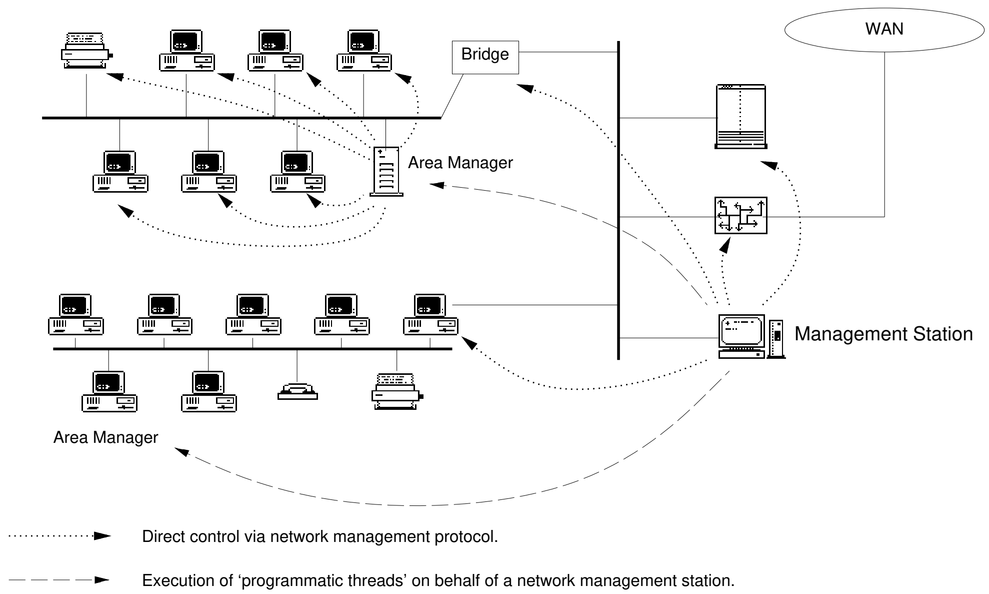

Abbildung 4: Verteiltes Netzwerkmanagement

Alle diese Probleme lassen sich vermeiden, wenn ein verteiltes Management-System eingesetzt wird, in dem die Überwachung eines Teilnetzes dezentral und möglichst nah am zu kontrollierenden Teilnetz erfolgt. Die zentrale Station übernimmt dann im wesentlichen die Überwachung der sogenannten Mid-Level Manager oder Area Manager (Abbildung 4).

Mit Hilfe der Tcl-Erweiterungen von Scotty ist es möglich, die Kommunikation zwischen der zentralen Netzwerkmanagement-Station und den Mid-Level Managern über das SNMP-Protokoll abzuwickeln, da ein Scotty Prozeß gleichzeitig Agenten- und Managerrolle einnehmen kann. Die Steuerung und Kontrolle des Mid-Level Managers erfolgt durch eine spezielle Management Information Base (MIB), die die zu bearbeitenden Aufgaben und die bisher erhaltenen Ergebnisse enthält. Derzeit wird eine überarbeitete Form der MLM-MIB [1] benutzt, die im wesentlichen aus vier Tabellen besteht: Die `mlmScriptTable` enthält Informationen über alle am Agenten bekannten Tcl Skripte. Der zu diesen Skripten gehörenden Quelltext kann in der `mlmSourceTable` gefunden werden. Diese Tabelle erlaubt auch das Laden neuer Skripte über SNMP. Die `mlmExecutionTable` beschreibt in Ausführung befindliche Skripte mit ihren Parametern und die `mlmResultTable` nimmt die während der Ausführung erzeugten Ergebnisse auf.

6 Zusammenfassung und Ausblick

In diesem Aufsatz haben wir die Kernkomponenten einer Netzwerkmanagement-Plattform beschrieben, die sich auf die interpretierte Tool Command Language (Tcl) stützt und Management-Applikationen dadurch ein hohes Maß von Anpaßbarkeit und Erweiterbarkeit verleiht. Nach einer Beschreibung von Tkined und Scotty wurde erläutert, wie sich mit Hilfe des eingebetteten SNMP-Agenten ein verteiltes Management-System realisieren läßt.

Eine besondere Eigenschaft eines auf Scotty basierenden verteilten Management-Systems ist, daß auf der Manager als auch auf der Mid-Level Manager Seite dieselbe Umgebung bereitsteht. Dies macht es sehr einfach, Funktionalitäten zu verlagern. Außerdem ist es auch prinzipiell möglich, dynamisch über die Verlagerung von Skripten zu entscheiden und sogar eine Migration von Skripten zwischen Managern zu erlauben. Allerdings müssen weitere Versuche in dieser Richtung zeigen, ob derartige Strategien von praktischem Nutzen sind.

Die Quellen von Tkined und Scotty sind seit July 1993 über FTP verfügbar. Weitere Informationsquellen und ein Hinweis auf die aktuelle Version findet man im World Wide Web unter der URL http://www.cs.tu-bs.de/ibr/. Das System befindet sich an mehreren Universitäten und Forschungseinrichtungen im Einsatz. Zur Diskussion von Themen mit Bezug zu Tkined und Scotty existiert eine Mailing-Liste. Teilnehmer wird man durch eine E-Mail an tkined-request@ibr.cs.tu-bs.de.

Literatur

[1] J.D. Case and D.B. Levi. SNMP Mid-Level-Manager MIB. Internet Draft 00, SNMP Research, Inc., October 1993.

[2] V. Chandra. XOM and XMP for Programmers. *CMIP Run*, 2Q93:6–8, 1993.

[3] W. Clinger and J. Rees. Revised^4 report on the algorithmic language scheme. Technical report, November 1991.

[4] D.E. Comer. *Internetworking with TCP/IP Volume 1: Principles, Protocols, and Architectures*. Prentice Hall, 2 edition, 1991.

[5] U. Eisenblätter, F. Hommes, and E. Pless. Ein Vergleich von kommerziellen Netzwerk-Management-Systemen. Arbeitspapiere der GMD 800, Gesellschaft für Mathematik und Datenverarbeitung, October 1993.

[6] ISO. Information technology – Open System Interconnection – Common management information protocol specification. International Standard ISO 9596, ISO, 1990.

[7] ISO. Information technology – Open System Interconnection – Common management information service definition. International Standard ISO 9595, ISO, 1991.

[8] G. Knight, G. Pavlou, and S. Walton. Experience of Implementing OSI Management Facilities. In *Proc. International Symposium on Integrated Network Management*, pages 259–270, April 1991.

[9] J.K. Ousterhout. *Tcl and the Tk Toolkit.* Addison-Wesley, April 1994.

[10] M.T. Rose, J.P. Onions, and C.J. Robbins. *The ISO Development Environment: User's Manual.* Performance Systems International, X-Tel Services Ltd, July 1991.

[11] J. Schönwälder and H. Langendörfer. How To Keep Track of Your Network Configuration. In *Proc. 7th Conference on Large Installation System Administration (LISA VII)*, pages 189–193, Monterey (California), November 1993.

[12] J. Schönwälder and H. Langendörfer. Netzwerkmanagement - Beschreibung des Exponats auf der CeBIT '94. Informatikbericht 94-02, TU Braunschweig, 1994.

[13] W. Stallings. *SNMP, SNMPv2 and CMIP: The Practical Guide to Network Management Standards.* Addison Wesley, 1993.

[14] D. Steedman. *Abstract Syntax Notation One (ASN.1): The Tutorial and Reference.* Technology Appraisals, 1990.

[15] G. van Rossum. An Introduction to Python for UNIX/C Programmers. In *Proc. NLUUG '93*, 1993.

[16] L. Wall and R. Schwartz. *Programming Perl.* O'Reilly, 1991.