Technische Universität Braunschweig

Informatik-Bericht

94-02

Netzwerkmanagement

Beschreibung des Exponats auf der CeBIT '94

J. Schönwälder

H. Langendörfer

Institut für Betriebssysteme und Rechnerverbund

Technische Universität Braunschweig
Bültenweg 74/75
38106 Braunschweig

März 1994

Vorwort

Das Institut für Betriebssysteme und Rechnerverbund der Technischen Universität Braunschweig beschäftigt sich seit 1991 mit verschiedenen Problemen, die beim Betrieb großer Netzwerke (mehrere hundert Knoten) auftreten. Dabei wird insbesondere auch der Frage nachgegangen, wie sich die ursprünglich getrennten Gebiete Netzwerkmanagement und System-Administration zusammenführen lassen.

Innerhalb der letzten Jahre sind am Institut verschiedene Werkzeuge entstanden, die jeweils Teilaufgaben aus den oben genannten Bereichen unterstützen. Im Unterschied zu anderen Systemen wurde dabei besonderer Wert auf leichte Erweiterbarkeit und Anpaßbarkeit an existierende Umgebungen gelegt. Auf dem Gemeinschaftsstand der Niedersächsischen Hochschulen auf der CeBIT '94 (16. bis 23. März 1994) werden die Werkzeuge der Öffentlichkeit vorgestellt. Dieser Bericht dient als Beschreibung des Exponats. Weiterführende Literatur ist im Literaturverzeichnis angegeben.

Für den außerordentlichen Einsatz bei der Realisierung der Prototypen möchten wir uns bei allen am Projekt beteiligten Studenten bedanken: Sascha Bengsch, Sylvia Dieckmann, Frank Dippel, Guntram Hueske, Michael Kernchen, Stefan Schoek, Ullrich Schoenbeck und Erik Schönfelder. Besonderer Dank gilt auch Stefan Petri, der bei der Integration des von ihm geschriebenen Simulators Hilfe leistete, und allen Personen, die frühere Versionen getestet und uns damit geholfen haben, das System portabel und robust zu gestalten.

J. Schönwälder

H. Langendörfer

Inhaltsverzeichnis

1 Einführung und Motivation

Der Betrieb großer Rechnernetze ist nur durch den Einsatz von speziell für diesen Zweck entwickelter Software möglich. Netzwerkmanagement-Systeme bestehen aus einer Sammlung von Werkzeugen, mit denen das Verhalten des Netzwerks beobachtet und gesteuert werden kann. Derartige Systeme müssen nicht nur effizient sein, sondern auch flexibel einer gegebenen Infrastruktur angepaßt werden können.

Um die geforderte Flexibilität zu erreichen, besitzt das von uns entwickelte Netzwerkmanagement-System eine eingebettete, interpretierte Programmiersprache, die einerseits die grafische Oberfläche von den Management-Applikationen und andererseits die unterstützten Protokolle von den mit ihrer Hilfe durchgeführten Management-Operationen trennt. Alle auf der Messe vorgeführten Werkzeuge sind als Skripte in dieser Programmiersprache realisiert, wodurch das System eine extrem hohe Anpaßbarkeit und Erweiterbarkeit erlangt.

Der Zugriff auf Management-Informationen wird durch spezielle Sprachkonstrukte unterstützt und soweit möglich vereinfacht. Für die Vielzahl von verschiedenen Informationsquellen wurden verschiedene Interpreter realisiert, die alle einen gemeinsamen Sprachkern benutzen. Die Zerlegung in verschiedene Interpreter ermöglicht einfache Erweiterungen für bisher nicht unterstützte Protokollfamilien und sorgt andererseits für eine gute Effizienz, da die einzelnen Programme klein bleiben.

Als Beispiele für die Implementierung von Management-Operationen mit Hilfe einer Skript-Sprache werden Werkzeuge zur automatischen Ermittlung der Netzwerktopologie und zum Monitoring von TCP/IP Netzen vorgestellt. Die Anbindung der Werkzeuge, die als externe Prozesse realisiert sind, an die Oberfläche erfolgt über ein einfaches Kommunikationsprotokoll. Das Protokoll ist durch Sprachkonstrukte gekapselt, so daß die Benutzung von Diensten der Oberfläche innerhalb der Skripte wie ein lokaler Prozeduraufruf erscheint.

Einige der vorgestellten Werkzeuge verwenden am Institut entwickelte Verfahren (z.B. zur Topologie-Ermittlung), während andere Standard-Funktionen von Netzwerkmanagement-Systemen nachbilden. Dabei steht grundsätzlich der Gedanke der Erweiterbarkeit im Vordergrund des Entwurfs. So können die implementierten Werkzeuge als Beispiele für die Realisierung eigener, der gegebenen Infrastruktur angepaßter Werkzeuge dienen.

Für Planungszwecke wurde außerdem ein bereits vorhandener Netzwerksimulator in das System integriert. Durch die Nutzung der grafischen Oberfläche kann der Simulator interaktiv konfiguriert werden. Die an den Meßpunkten anfallenden Daten lassen sich während eines Simulationslaufs grafisch darstellen.

Im folgenden Kapitel geben wir zunächst einen kleinen Überblick über die Systemarchitektur. Anschließend werden ausgehend von der grafischen Oberfläche anhand von Beispielen einzelne Bausteine des Systems beschrieben. Der Bericht endet mit einer Zusammenfassung und einem Ausblick auf zukünftige Aktivitäten.

2 Architektur

Die Architektur des Netzwerkmanagement-Systems (Abb. 1) setzt sich aus drei wesentlichen Schichten, der Benutzeroberfläche, der Skript-Ebene und der Interpreter-Ebene zusammen. Innerhalb dieser Schichten existieren verschiedene unabhängige Module, wodurch eine leichte Portierbarkeit und Modifizierbarkeit von ganzen Systemteilen nach dem Baukastenprinzip erreicht wird.

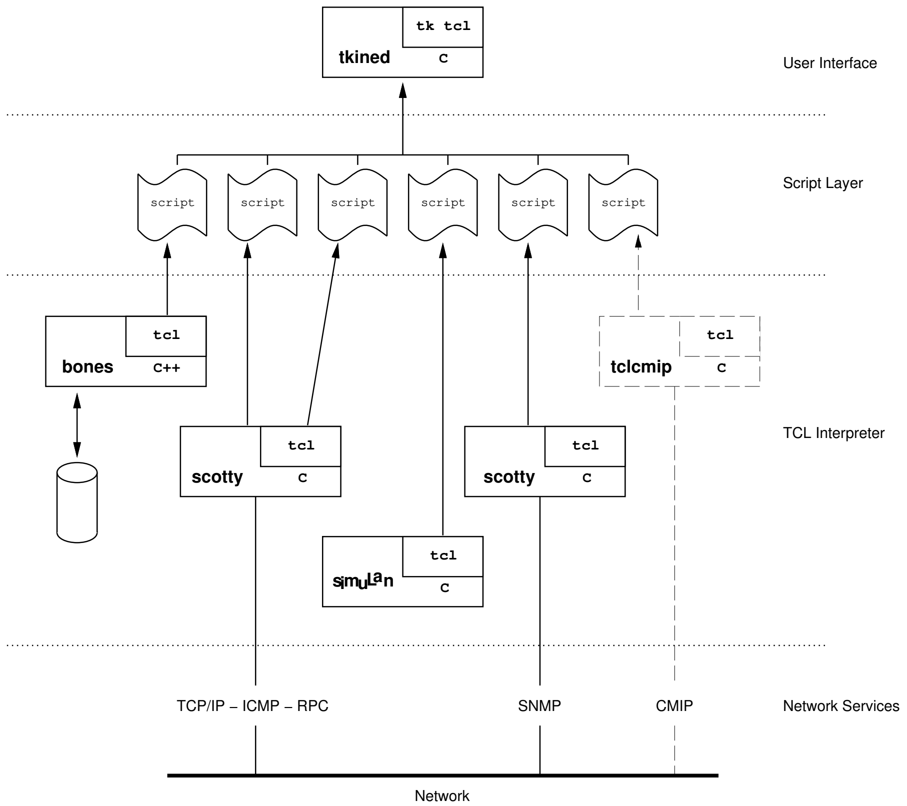

Abbildung 1: Systemarchitektur

Die unterste Ebene besteht aus einer Sammlung von Interpretern basierend auf der an der Universität in Berkeley entwickelten Tool Command Language (TCL) [4]. Die Tool Command Language ist eine erweiterbare Sprache mit Variablen, Prozeduren, Kontrollstrukturen wie `if`, `while` und `for`, arithmetischen Ausdrücken, assoziativen Arrays und Listen. TCL ist als C-Bibliothek implementiert, die sehr einfach in andere Programme eingebettet und durch applikationspezifische Befehle erweitert werden kann.

Die in der untersten Ebene angesiedelten Interpreter erweitern den universellen TCL-Sprachkern um Befehle, die den effizienten Zugriff auf Netzwerkmanagement-Informationen erlauben oder spezielle Dienste (z.B. Datenhaltung oder Netzwerksimulation) bereitstellen.

Auf diese Interpreter setzen in der nächsthöheren Ebene TCL-Skripte auf, die jeweils nach Funktionsbereichen gegliedert Management-Werkzeuge realisieren. Auf dieser Ebene können auf einem relativ abstrakten Niveau Verfahren effizient implementiert werden.

Die oberste Ebene des Systems bildet eine grafische Oberfläche, die sowohl mit dem Benutzer als auch mit den Skripten kommuniziert. Die Oberfläche stellt dem Benutzer einen einheitlichen Zugang zu den Management-Applikationen bereit, während sie gleichzeitig den Skripten eine einheitliche und einfach zu benutzende Programmierschnittstelle bietet.

3 Netzwerkeditor `tkined`

Die grafische Oberfläche des Netzwerkmanagement-Systems präsentiert sich dem Benutzer gegenüber als Netzwerkeditor. Dieser Editor erlaubt die Erstellung von Netzwerkplänen, die die Grundlage vieler Tätigkeiten während des Netzwerk-Lebenszyklus sind [8]. Die Bedienung des Netzwerkeditors ähnelt der Bedienung eines objektorientierten Zeichenprogramms und ist daher sehr schnell zu erlernen.

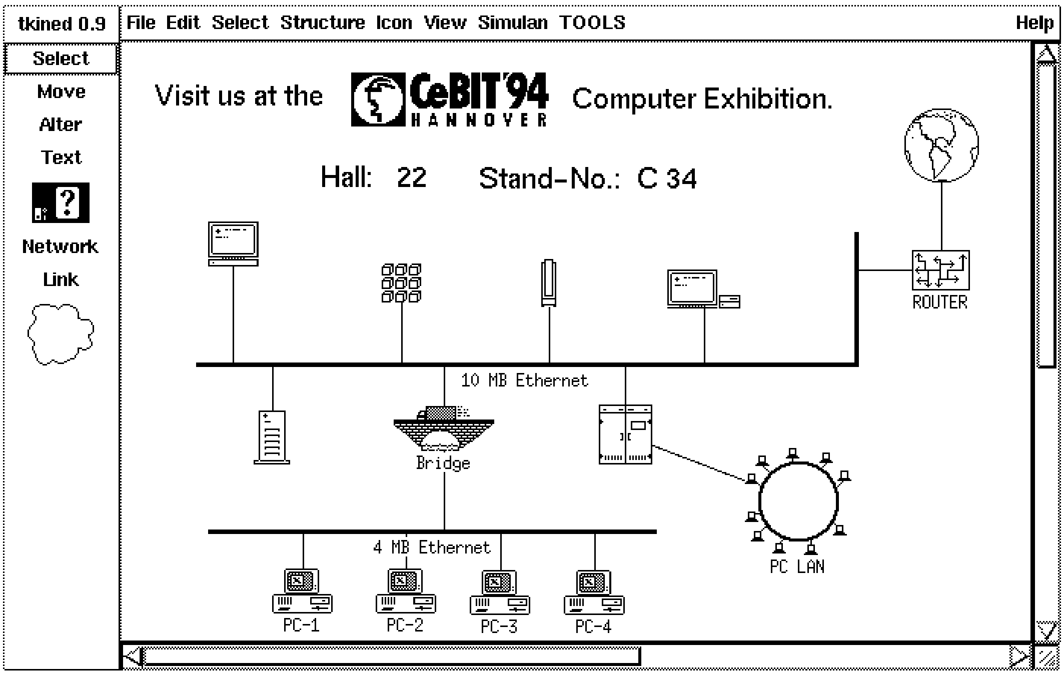

Abbildung 2: `tkined`

In Abb. 2 sind die verschiedenen unterstützten Objekttypen zu erkennen. Netzwerke werden durch dicke Linien dargestellt, die aus mehreren Segmenten bestehen können. Vernetzte Geräte werden durch benutzerdefinierbare Icons dargestellt, wobei durch die Verwendung von Link-Objekten (dünne Linien) Verbindungen hergestellt werden können. Einmal erzeugte Verbindungen werden von `tkined` verwaltet und automatisch aktualisiert, wenn sich die Position eines Icons verändert.

Neben diesen grundlegenden Basisobjekten ermöglicht `tkined` die hierarchische Strukturierung durch Gruppenobjekte, die eine Menge von anderen `tkined`-Objekten zusammenfassen. Gruppen können sowohl expandiert (alle Mitglieder der Gruppe sind sichtbar) als auch zu einem Icon zusammengefaßt dargestellt werden. Durch die hierachische Strukturierung lassen sich auch Pläne großer Netzwerke mit mehreren hundert Knoten bearbeiten.

Zur Beschriftung stehen Textobjekte zur Verfügung, die beliebig angeordnet und in verschiedenen Fonts und unterschiedlichen Farben dargestellt werden können. Zusätzlich lassen sich Grafiken in den Editor einbinden, was beispielsweise zur Erstellung von Kabelplänen bei bereits existierenden Grundrißplänen nützlich ist.

4 Topologie-Ermittlung

In großen Netzwerken ist die Erstellung und Pflege einer Netzwerk-Karte mit großem Aufwand verbunden. Deshalb bietet es sich an, die Erstellung soweit wie möglich automatisch vorzunehmen. Leider wurde bei der Entwicklung der heute weit verbreiteten Protokolle diesem Problem keine besondere Aufmerksamkeit geschenkt, so daß für die Ermittlung von Netzwerktopologien je nach Protokollfamilie spezielle Verfahren entwickelt werden müssen.

Das von uns zur Topologie-Ermittlung in TCP/IP Netzwerken entwickelte Verfahren läßt sich durch folgende Eigenschaften charakterisieren [7]:

- Aktives, zielgerichtetes Durchsuchen eines Adreßraums.
- Nutzung von Informationsquellen, die in jedem IP Netzwerk vorhanden sind.
- Behandlung widersprüchlicher Informationen durch Mehrheitsentscheide.
- Trennung des Ermittlungsverfahrens von Hilfsmitteln zur Darstellung.

Das Verfahren erreicht eine recht hohe Erkennungsrate und läßt sich in schnellen lokalen Netzen effizient einsetzen. Da keinerlei Konfigurationsinformationen benötigt werden, ist es mit entsprechend verringerter Geschwindigkeit auch in globalen Netzen einsetzbar.

Das Ergebnis der Topologie-Ermittlung ist eine Menge von Rechnern und Netzwerken, die anschließend durch ein Layout-Werkzeug für TCP/IP Netze angeordnet werden können. Dabei werden Informationen des Domain Name Service (DNS) benutzt, um z.B. die Icons der Rechner zu bestimmen. Ein eingebauter Graphen-Layout-Algorithmus übernimmt die Positionierung der Icons, nachdem zuvor alle "unwichtigen" Geräte zusammen mit ihren Netzwerken zu Gruppen zusammengefaßt wurden. Das Ergebnis ist eine Sicht auf ein TCP/IP Netzwerk, die sich an Teilnetzen und den Gateways orientiert.

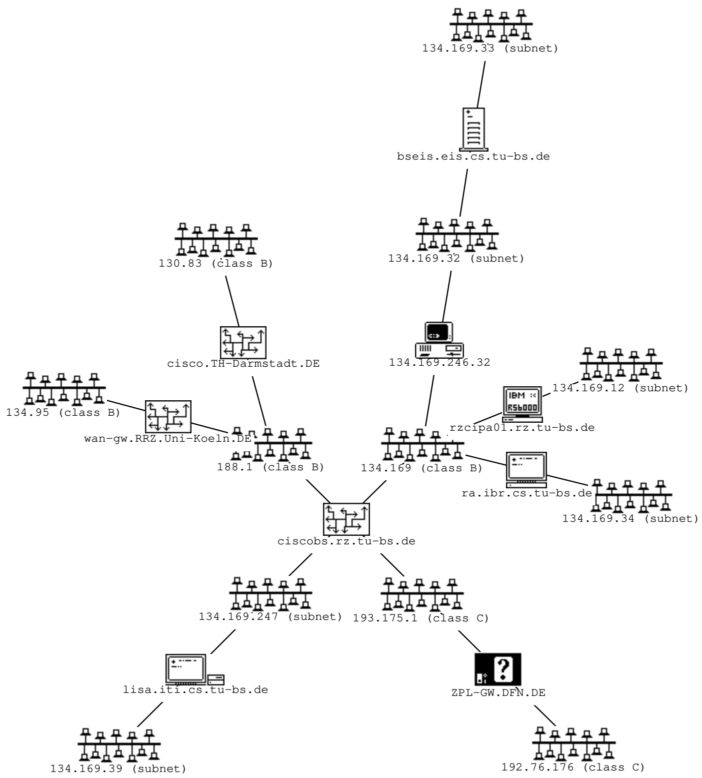

Abbildung 3: Netzwerk-Topologie

Die Abb. 3 zeigt eine Netzwerktopologie, wie sie nach Durchführung der Topologie-Ermittlung und des Layout-Verfahrens im `tkined` Editor dargestellt wird.

5 Fehlersuche

Ein wichtiges Einsatzgebiet für ein Netzwerkmanagement-System ist die Unterstützung der Netzwerkbetreiber bei der Fehlersuche. Hierzu wurden für den Prototyp eine Reihe von Standardverfahren mit Hilfe der TCL-Interpreter implementiert und somit in der `tkined` Oberfläche verfügbar gemacht.

Als Beispiel, wie sich Standardverfahren in unser System integrieren lassen, zeigt Abb. 4 die Implementierung des `finger` Kommandos, das für alle selektierte Rechner im `tkined` Editor anzeigt, wer gerade auf dem Rechner arbeitet. Ohne die Details des Skripts hier zu erläutern, ist unmittelbar einsichtig, daß die Einbettung in die grafische Oberfläche keinerlei programmtechnische Verrenkungen erfordert. Außerdem ist der Aufbau der benötigten TCP/IP Verbindung durch entsprechende

```
#!/usr/local/bin/scotty -inf
proc finger {list} {
    foreach comp $list {
       if {[lindex $comp 0]=="NODE"} {
            set host [lindex $comp 2]
            set ip [lindex $comp 3]
            set result ""
            if {[catch {connect $ip finger} f]} {
                ined acknowledge "Can not connect to $host."
                continue
            }
            puts $f ""
            while {! [eof $f]} {
                lappend result [string trimright [gets $f] "\r\n"]
            }
            shutdown $f all
            close $f
            eval "ined browse {Finger for host $host:} {} $result"
        }
    }
}
ined create TOOL "DEMO" finger
```

Abbildung 4: Implementierung des `finger` Kommandos

TCL-Befehle einfach möglich.

Neben den Kommandos, die sich auf allgemeine TCP/IP Dienste abstützen, gibt es ein weiteres Werkzeug, das zur Fehlersuche das Simple Network Management Protokoll (SNMP) benutzt. Dieses Werkzeug ist insbesondere zur Überwachung von Routern oder Bridges gedacht und bietet entsprechende vorgefertigte Kommandos an.

6 Monitoring

Eine weitere Grundfunktion eines Netzwerkmanagement-Systems ist die Überwachung des laufenden Betriebs. Hierzu bietet der Netzwerkeditor die Möglichkeit, Meßwerte in Form kleiner Stripcharts oder Balkendiagramme darzustellen (siehe Abb. 5). Diese Darstellungsweise erlaubt es, auf dem begrenzten Platz eines Bildschirms sehr viele Informationen gleichzeitig unterzubringen.

Spezielle Monitor-Skripte sammeln periodisch Statusinformationen und reichen sie zur Darstellung in Stripcharts und Balkendiagrammen an den `tkined` Editor. Ein Job-Scheduler übernimmt die Verwaltung der periodisch zu sammelnden Statusinformationen, so daß viele verschiedene Informationsquellen von einem einzigen Prozeß gesammelt werden können. Mit Hilfe des Job-Schedulers lassen sich die

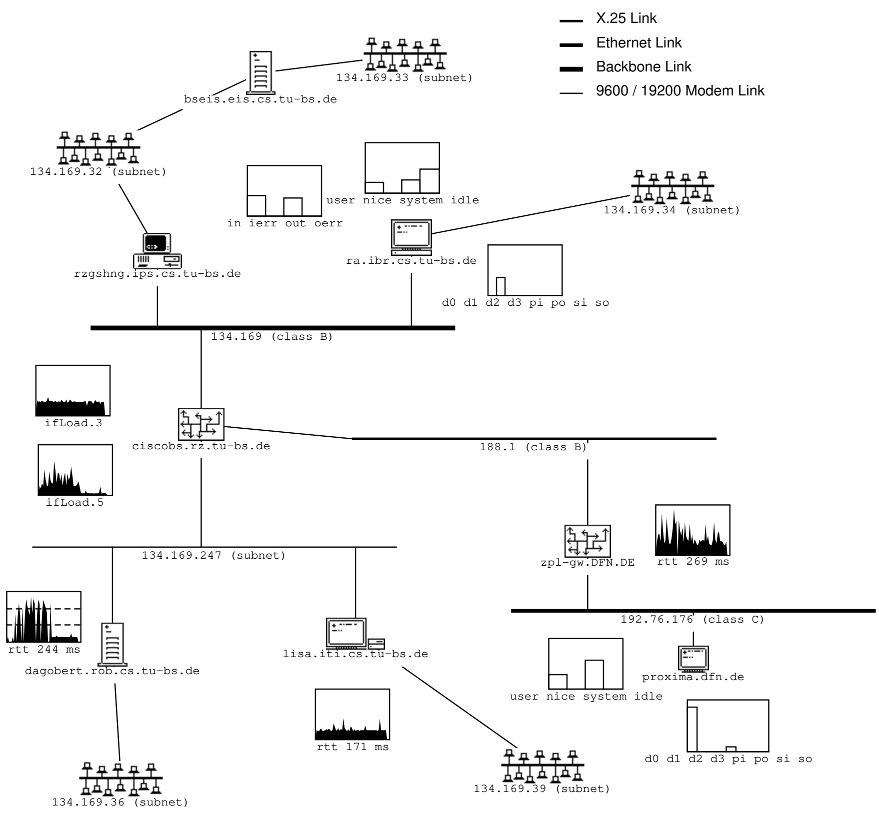

Abbildung 5: Status Überwachung

Abtastraten jederzeit einstellen und ganze Aufträge vorübergehend suspendieren.

Der Zustand eines Netzwerkelements kann natürlich auch durch die Farbgebung angezeigt werden. Dieses Mittel wird aber derzeit nur selten benutzt, da das System auch beim Betrieb auf monochromen Bildschirmen ohne Beeinträchtigung funktionieren soll.

7 Erfassung von Statistiken

Eng Verbunden mit dem Netzwerk-Monitoring ist die Erfassung von Leistungs- und Fehlerdaten für Planungszwecke oder spätere Analysen. Statistische Daten können Aufschluß über die Entwicklung des Netzzustands und mögliche Problemfelder geben.

Für diesen Zweck wurde ein spezielles Monitoring-Konzept für die zentrale Erfassung und Ablage des Datenmaterials entwickelt. Der Zugriff auf das Datenmaterial erfolgt über einen zentralen Server, so daß das gesammelte Datenmaterial auch zur kooperativen Fehlersuche über das Netz bereitgestellt werden kann. Neben dem

Zugriff auf die Rohdaten kann vom Server bereits verdichtetes Datenmaterial (z.B. Mittelwerte über Stundenintervalle) angefordert werden, so daß der Datentransfer effizient auf die geplante Auswertung abgestimmt werden kann.

Das vom zentralen Server benutzte Datenmodell erlaubt die Vereinheitlichung der Datenrepräsentation, egal mit welchem Protokoll die Daten tatsächlich erfaßt worden sind. Derzeit existieren Konfigurationen zum Sammeln von Lastdaten mit Hilfe des `rstat` RPC und zur Erfassung von MIB-Variablen via SNMP.

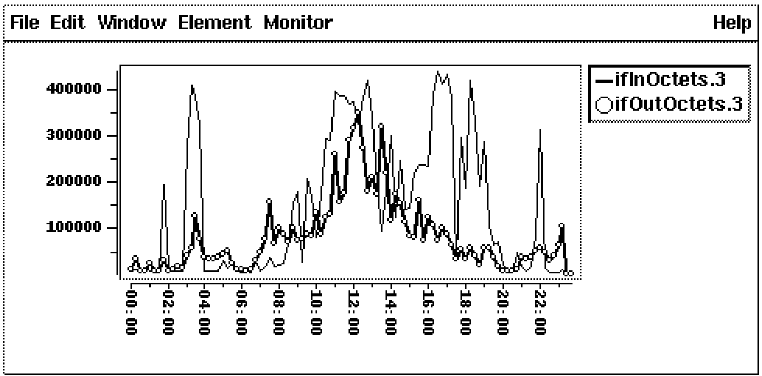

Abbildung 6: Auswertung statistischer Daten mit `tkgraphs`

Zur Auswertung des Materials gibt es neben der Möglichkeit, Daten in `tkined`-Stripcharts zu laden, ein spezielles Auswerteprogramm `tkgraphs` (Abb. 6). Mit Hilfe von `tkgraphs` können Statistiken vom Server abgefragt und einer weiteren Analyse unterzogen werden.

8 Konfigurationsverwaltung mit bones

Eine Hauptaufgabe der System-Administration ist die Verwaltung von Konfigurationsinformationen. Dabei spielen neben technischen Informationen (z.B. vergebene Netzwerkadressen, Rechner-Accounts) auch organisatorische Informationen (welches Gerät gehört zu welchem Projekt etc.) eine wichtige Rolle.

Die mit dem UNIX-Betriebssystem vom Hersteller mitgelieferte Software ist meist nur in der Lage, die für den Betrieb erforderlichen Daten zu verwalten. Das `bones` System erlaubt die zentrale und rechnerunabhängige Verwaltung von Konfigurationsinformationen basierend auf einem objektorientierten Datenmodell. Neben technischen Informationen können Daten über Projekte, Personen, vorhandene Hard- und Software abgelegt werden. Der Zugriff auf die Daten ist netzwerktransparent.

Aus dem zentralen Datenbestand können Konfigurationsinformationen für die einzelnen Rechner zusammengestellt und anschließend verteilt werden. Abb. 7 zeigt die Architektur des Systems. Neben dem TCL-Interpreter `bones` erlaubt der

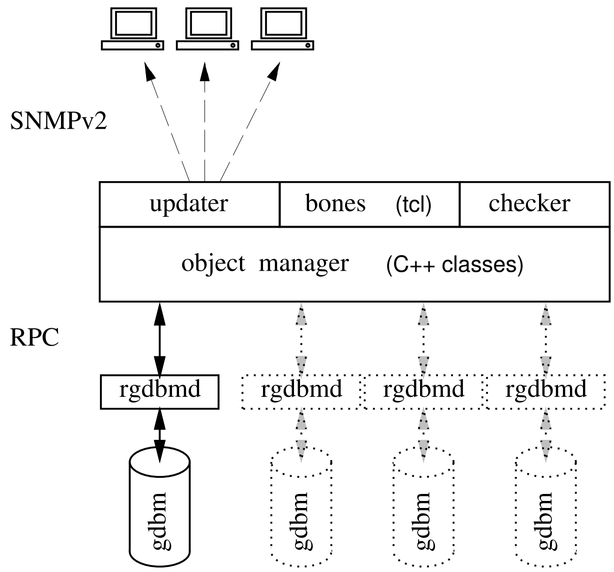

Abbildung 7: Architektur des `bones`-Systems

`updater` die Zusammenstellung der zu verteilenden Konfigurationsinformationen. Zur Konsistenzprüfung dient das Programm `checker`. Alle drei Programme setzen auf den Objektmanager auf, einer C++ Klassenbibliothek, die mit dem Datenserver `rgdbmd` mit Hilfe von Remote Procedure Calls (RPC) kommuniziert.

9 Trouble Tickets

Trouble-Ticket-Systeme dienen der Verwaltung und Speicherung von Fehlerinformationen. Mit Trouble-Tickets werden Fehlerbeschreibungen erfaßt, eingeleitete Maßnahmen festgehalten und außerdem wird die termingerechte Bearbeitung überwacht. Durch die Sammlung bereits gelöster Fälle ergibt sich die Möglichkeit, wiederholt auftretende Probleme durch die Entwicklung automatischer Verfahren zu bearbeiten. Außerdem läßt sich die Behandlung von Fehlerfällen bei Abwesenheit des normalerweise zuständigen Personals anhand ähnlicher Tickets schneller durchführen.

Durch Erweiterung des `bones` Datenmodells um Klassen zur Darstellung von Tickets, Fehlerbeschreibungen, eingeleiteten Maßnahmen, Lösungen und Kontrollinformationen wurde ein integriertes Trouble-Ticket-System realisiert, daß auf sämtliche Informationen der `bones` Klassen direkt zugreifen kann.

Neue Tickets können über einen E-Mail Zugang in das Trouble-Ticket-System eingetragen werden. Die Bearbeitung erfolgt über ein textorientiertes Programm, das noch nicht bearbeitete Tickets anmahnt und die termingerechte Bearbeitung überwacht.

10 Netzwerksimulator simuLan

simuLan ist ein ereignisorientierter Simulator für Computer-Netzwerke [5]. Der Aufbau des Simulators orientiert sich an der Schichten-Architektur von Netzwerkprotokollen. Für die zu simulierenden Protokolle werden entsprechende Module im Simulator konfiguriert. Eine Bibliothek stellt alle Grundoperationen bereit, die zur Implementierung der Protokollmodule benötigt werden.

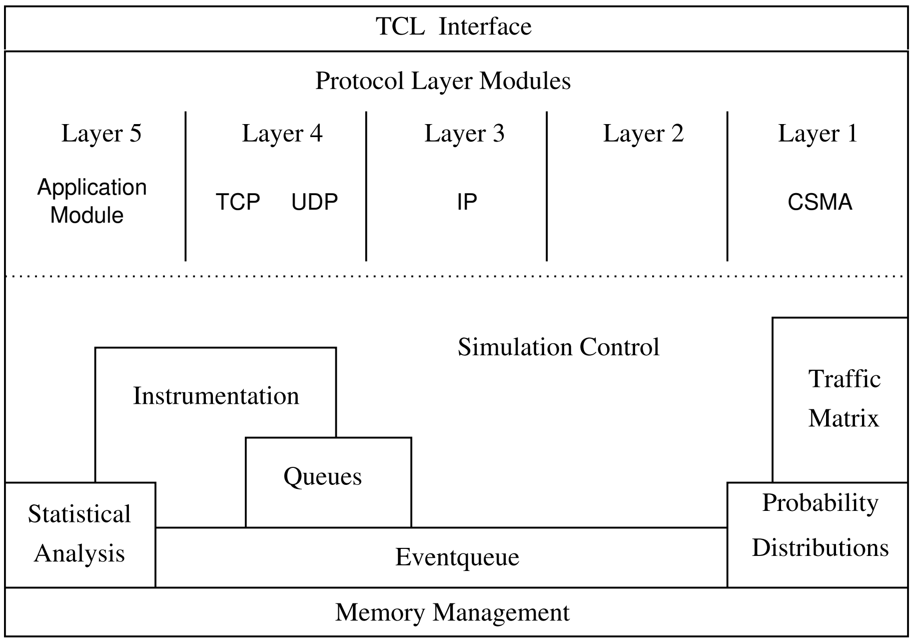

Abbildung 8: Architektur des Netzwerksimulators simuLan

Bei der Beschreibung eines zu simulierenden Systems werden für die jeweiligen Protokollschichten der einzelnen Knoten die zu benutzenden Protokollmodule und ihre Parameter definiert. Simulationsergebnisse können je nach Konfiguration an hierfür vorgesehenen Meßpunkten abgenommen werden. Durch die Integration von simuLan in unser Netzwerkmanagement-System kann das Fortschreiten des Simulators während der Simulation grafisch dargestellt werden (Abb. 9).

11 Zusammenfassung und Ausblick

Das vorgestellte Netzwerkmanagement-System bietet eine portable und erweiterbare Umgebung zur Verwaltung von großen Netzwerken. Durch die Verwendung einer einheitlichen, interpretierten Skript-Sprache können Erweiterungen problemlos integriert werden.

Aus den Bereichen Netzwerkmanagement und System-Administration wurden verschiedene Problemfelder herausgegriffen und Lösungen präsentiert. So liefern die

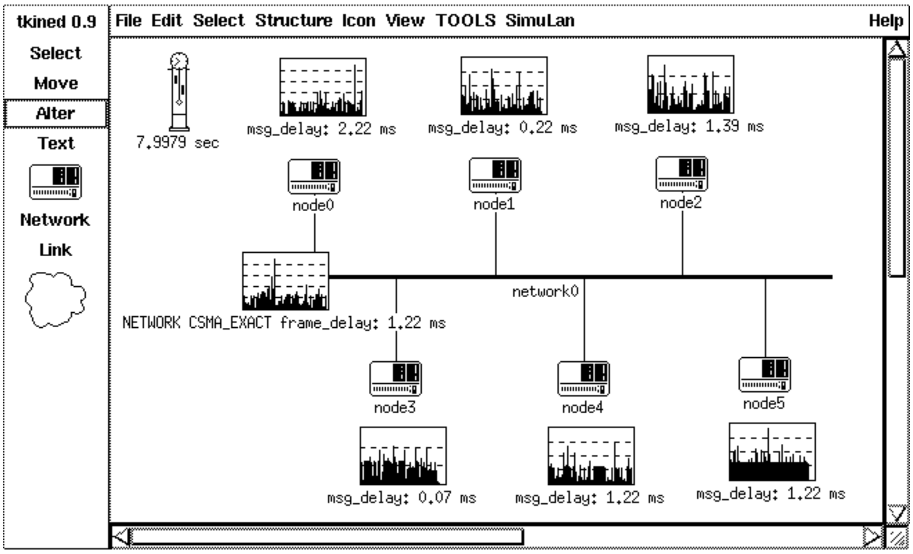

Abbildung 9: Grafische Darstellung eines Simulationslaufs

Werkzeuge zur Topologie-Ermittlung und zum Monitoring einen wichtigen Beitrag zum Netzwerkmanagement. Das `bones` System ermöglicht dem System-Administrator eine konsistente Datenhaltung. Das daran angeschlossene Trouble-Ticket-System unterstützt die Bearbeitung von Fehlermeldungen.

Das auf der CeBIT '94 präsentierte System ist auf dem UNIX Betriebssystem implementiert worden und unter anderem auf Maschinen der Hersteller Sun, DEC, HP, IBM sowie auf PC UNIX Versionen (Linux / BSD386) lauffähig.

Zukünftige Aktivitäten haben die Erschließung weiterer Management-Informationsquellen zum Ziel. Derzeit sind TCL-Erweiterungen für die Protokolle SNMPv2 und CMIP in der Implementierungsphase. Die Verlagerung von Kontrollfunktionen von einer zentralen Station hin zu entsprechend ausgerüsteten Netzwerkknoten ist insbesondere bei der Überwachung von sehr großen Netzwerken oder Hochgeschwindigkeitsnetzen erforderlich. Allerdings sind die mit der Dezentralisierung verbundenen Probleme der Koordination bisher nur unzureichend geklärt und bedürfen weiterer Untersuchungen.

Literatur

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

[2] H. Langendörfer. *Leistungsanalyse von Rechensystemen: Messen, Modellieren, Simulation.* Hanser, München, 1992.

[3] H. Langendörfer und B. Schnor. *Verteilte Systeme.* Hanser, München, 1994.

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

[5] S. Petri. simuLan – Ein Werkzeug zur Simulation lokaler Netze. Diplomarbeit, Institut für Betriebssysteme und Rechnerverbund, TU Braunschweig, 1991.

[6] J. Schönwälder und H. Langendörfer. Administration of large distributed UNIX LANs with BONES. In *Proc. World Conference On Tools and Techniques for System Administration, Networking, and Security*, Arlington (Virginia), April 1993.

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

[8] J. Schönwälder und H. Langendörfer. INED – An Application Independent Network Editor. In *Proc. World Conference On Tools and Techniques for System Administration, Networking, and Security*, Arlington, (Virginia), April 1993.