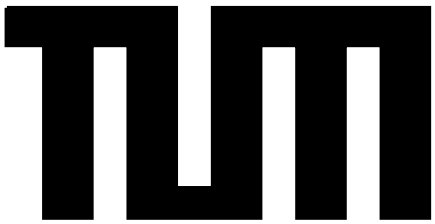


TECHNISCHE
UNIVERSITÄT
MÜNCHEN

Fortgeschrittenenpraktikum

Weiterentwicklung der Managementanwendung SyNeMa für den MNM-Systemmanagementagenten

Karsten Färber

Aufgabensteller: Prof. Dr. Heinz-Gerd Hegering
Betreuer: Stephen Heilbronner
Abgabedatum: 3. August 1998

Inhaltsverzeichnis

1 Einführung

1.1 SyNeMa und System-Management

SyNeMa wurde als Teil eines grösseren Projektes zum Systemmanagement entwickelt. Es stellt ein Management-Programm dar, die über ein SNMP-Interface mit einem auf dem Lehrstuhl weiterentwickelten SNMP-fähigen Hauptagenten kommunizieren kann.

Dieser Hauptagent, der die MIB-II implementiert, wurde unter anderem um eine DPI Server-Schnitstelle erweitert, um eine Kommunikation mit diversen Subagenten zu ermöglichen. Einer dieser Subagenten implementiert die LRZ-UNIX MIB. SyNeMa wurde entwickelt, um die Informationen der LRZ-UNIX MIB mittels einer plattform-unabhängigen Sprache (tcl) und einer entsprechenden GUI (tk) darstellen und manipulieren zu können.

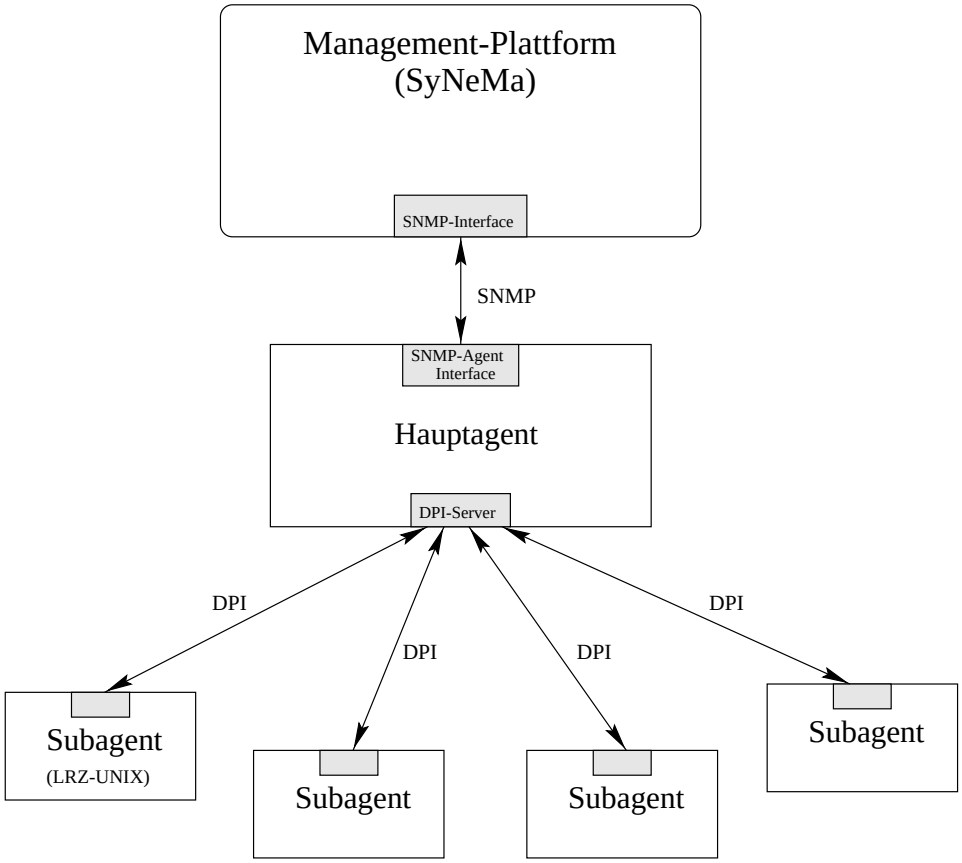


Abbildung 1

Abbildung 1 zeigt wie SyNeMa, Hauptagent und mehrere Subagenten zusammenhängen. SyNeMa verwendet SNMP zur Kommunikation mit dem Hauptagenten, dieser holt sich mittels DPI die nötigen Informationen von den verschiedenen Subagenten.

Der Subagent, der die LRZ-UNIX MIB implementiert, wurde auf HP-UX und SunOS Platformen portiert. Die Ergebnisse im Rahmen dieses Fortgeschrittenen-Praktikums beruhen aber nur auf Erfahrungen mit der HP-UX Version.

1.2 Tcl/Tk

Um SyNeMa möglichst flexibel auf verschiedenen Betriebssytemen als Management-Plattform verwenden zu können, war es notwendig, es mittels einer möglichst Betriebssystem-unaghängigen Sprache zu implementieren. Die Wahl fiel auf die Script-Sprache Tcl von John Ousterhout.

Um zusätzlich eine ansprechende graphische Darstellung zu ermöglichen, wird das Tcl-Paket Tk (ebenfalls von John Ousterhout) verwendet.

Im Rahmen des Fortgeschrittenen-Praktikums wurde SyNeMa von einer älteren Tcl/Tk Version auf die Version 7.5/4.1 portiert.

1.2.1 Installation von Tcl und Tk

Die Installation der Tcl und der Tk Distribution verläuft ohne Probleme. Zu erhalten sind die Source-Pakete von

`http://www.scriptics.com`.

Die tcl Installation sollte immer vor der tk Installation erfolgen. Nach dem Wechsel in das unix-Verzeichnis `tcl7.5/unix` bzw. `tk4.1/unix` kann man die Pakete compilieren. Mit

```
./configure --prefix=/usr/local/mnm/tcl
```

und einem anschliessenden

```
gmake
```

bzw.

```
gmake install
```

ist bereits alles getan.

Zu beachten ist, dass aus Gründen der Abwärtskompatibilität auf einer HP PA-RISC 1.1 compiliert werden sollte.

Man sollte beachten, dass die entsprechenden Environment-Variablen gesetzt werden:

```
setenv TCL_LIBRARY /usr/local/mnm/tcl/lib/tcl7.5
setenv TK_LIBRARY /usr/local/mnm/tcl/lib/tk4.1
```

Folgende Meldung erscheint standardmässig nach dem compilieren des tk-Pakets und sollte entsprechend beachtet werden:

```
To use the Tnm extension, you have to set the TCLLIBPATH
environment variable or you have to edit the init.tcl file
which is part of your Tcl installation.  See the README file
for further instructions how to do this.

The Tnm extension includes two programs (ntping, straps)
which require root permissions.  Please get root permissions
and type >> make sinstall << to install them setuid root.
```

Weiterhin empfiehlt es sich, im Verzeichnis `/usr/local/mnm/tcl/lib` Links ohne Versionsnummern auf die Libraries zu setzen, also

```
ln -s libtk4.1.a libtk.a
ln -s libtcl7.5.a libtcl.a
```

1.3 BLT-Library

Die BLT-Library ist eine Erweiterung für Tcl/Tk und ist erhältlich über `http://www.cs.uoregon.edu/~jhobbs/work`.

Die Instllation erfolgt im Prinzip genauso wie bei tcl/tk. Im Basisverzeichnis der Distribution ist zuerst

```
./configure --prefix=/usr/local/mnm/tcl
```

einzugeben.

Während der Konfiguration kann es sein, dass das tcl/tk Verzeichnis nicht automatisch gefunden wird. Das Programm fragt dann danach:

```
What directory contains libtcl.a (version 7.4+) ?
What directory contains libtk.a (version 4.0+) ?
```

Bei beiden Fragen mit

```
/usr/local/mnm/tcl/lib
```

antworten. LibBLT läuft bis tcl/tk Version 7.5/4.1.

Es ist darauf zu achten, dass `configure` die X11 Verzeichnisse mittels `xmkmf` findet. Deshalb die Pfadangaben genau überprüfen ! Z.B. wurde hier `/usr/local/lmu/bin/X11` oft als X11 Pfad gefunden, was nicht ideal ist und notfalls manuell in `config.status` geändert werden sollte.

Anschliessend mit

```
gmake
```

compilieren.

Danach im Verzeichnis `src/shared` nochmals mit

```
gmake
```

auch die shared-library Version compilieren.

Mit

```
gmake install
```

im Basisverzeichnis werden die erforderlichen Dateien installiert.

1.4 Scotty

Um SNMP-Kommandos absetzen zu können, wird das Tcl/Tk Erweiterung-Paket Scotty von Jürgen Schönwälder verwendet.

Scotty wurde an der Uni Braunschweig entwickelt und bietet die Möglichkeit, folgende Protokolle innerhalt eines Tcl-Scripts zu verwenden:

1. SNMP (SNMPv1, SNMPv2c, SNMPv2u including access to MIB definitions)
2. ICMP (echo, mask, timestamp and udp/icmp traceroute requests)
3. DNS (a, ptr, hinfo, mx and soa record lookups)
4. HTTP (server and client side)
5. SUN RPC (portmapper, mount, rstat, etherstat, pcnfs services)
6. NTP (version 3 mode 6 request)
7. UDP (send and receive UDP datagrams - no channels yet)

Scotty wurde in der Version 2.1.1 bei der Entwicklung von SyNeMa verwendet. SyNeMa verwendet das SNMPv1 Protokoll, um mit dem Hauptagenten zu kommunizieren. Scotty versteckt die Unterschiede zwischen SNMPv1 und SNMPv2 mittels geeigneter Typ-Umwandlungen soweit wie möglich, so dass eine spätere Umstellung auf SNMPv2 keine umfassende Neuprogrammierung erfordern wird.

1.4.1 SNMP Kommunikation in SyNeMa mittels Scotty

Die SNMP-Kommunikation in Scotty verwendet sogenannte Sessions. Eine Session ist ein (leichgewichtiges) Objekt, das Kontrolle hat über die Adresse des SNMP Kommunikations-Partners, die Authentifizierung und einiger Parameter, die das Verhalten der Kommunikation beeinflussen.

Das SNMP Protokoll verwendet festgelegte Datentypen, die von Scotty in entsprechende Tcl String-Typen umgewandelt werden.

1.4.2 Datentypen und ihre Darstellung in Tcl:

OCTET STRING Der SNMP-Typ Octet String ist eine Sequenz von Bytes, die von Scotty in Tcl in einen String umgewandelt wird, in dem die einzelnen Bytes mittels Doppelpunkt getrennt sind (z.B. "84:7c:04:2e").

OBJECT IDENTIFIER werden verwendet, um Informationen innerhalb einer MIB zu adressieren. Ein Object Identifier besteht aus einer Sequenz von positiven Zahlen. Die Folge der Zahlen definiert einen Pfad von der Wurzel des MIB-Baums bis zu einem MIB-Objekt oder MIB-Instanz. Die Darstellung in Tcl geschieht durch einen String, in dem die Zahlen durch Punkte getrennt sind (z.B. "1.3.6.1.3.100.2").

IP-Adressen werden wie gewohnt in 'dotted-notation' dargestellt, also in der Form "aaa.bbb.ccc.ddd".

Integer werden in Tcl auch als Integer dargestellt.

Im Laufe der Testläufe von SyNeMa hat sich herausgestellt, dass die Umwandlung nicht immer zur Zufriedenheit stattfindet. Dies sollte aber in späteren Scotty Versionen kein Problem mehr darstellen.

Folgende Kommandos werden verwendet:

snmp session <node>: Mit diesem Kommando wird eine neue Session zu einer besteimmten node eröffnet. Man erhält einen Identifier für diese Session als Rückgabewert. Optional kann man Community Names angeben, was z.B. notwendig wird, wenn die Standard-Namen (public) nicht verwendet werden, um z.B. unberechtigtes schreiben zu verhindern.

<Id> get <Oi>: Mit Hilfe von get, einem Session Identifier <Id> und einem Object Identifier <Oi> kann man Informationen von einem Agenten zu dem mit <Oi> gekennzeichneten Objekt anfordern.

<Id> set <Oi>: Wie bei get, nur dass hiermit Informationen geschrieben statt gelesen werden können.

mib name <label>: Mit diesem Befehl kann man den (Kurz-)Namen einer Node des MIB-Baums abfragen, wenn man <label> als Object Identifier übergibt.

mib oid <label>: Mit mib oid erhält man den Object Identifier einer MIB Node, wenn man deren Namen als <label> übergibt.

mib syntax <label>: Dieses Kommando verwendet man, wenn man die ASN.1 Syntax einer MIB Node erhalten will. <label> kann ein Node Name oder ein Object Identifier sein. Er wird bei SyNeMa verwendet, um typabhängige Ausgaben oder Funktionen prüfen zu können.

1.4.3 Installation von Scotty und wichtige Anmerkungen

Scotty lässt sich relativ problemlos compilieren, wenn man die Hinweise der Dokumentation beachtet. Nach Wechsel in das `scotty2.1.1/unix` Vezeichnis kann man mit

```
./configure --prefix=/usr/local/mnm/tcl
```

also der default Installation und

```
gmake
```

bzw.

```
gmake install
```

alles compilieren. Es gibt noch die Möglichkeit, verschiedene Zusatzpackages miteinzucompilieren (z.B. CMIP-Support). Das erfordert natürlich, dass diese Pakete auch vorhanden sind. Notwendig für SyNeMa sind sie jedenfalls nicht.

Scotty wird also unter

```
/usr/local/mnm/tcl
```

installiert. Die Daten werden in die entsprechenden Unterverzeichnisse

```
/usr/local/mnm/tcl/bin
/usr/local/mnm/tcl/man
/usr/local/mnm/tcl/lib
/usr/local/mnm/tcl/include
```

geschrieben. Bei der Installation wird leider immer 'mkdir -p' verwendet. Das hat zur Folge, dass die Installation immer bei dem Versuch /usr/local zu erstellen, abbricht. Entweder man erstellt den Verzeichnisbaum vor der Installation von Hand oder man verwendet z.B. gmkdir, um dieses Problem zu umgehen.

Wichtig: Scotty heisst als Tcl-Package tnm, und erstellt auch entsprechende Unterverzeichnisse.

Wenn, wie bei SyNeMa, mit einer nicht-standard MIB gearbeitet wird, muss man unbedingt die MIB-Definitionen Scotty zur Verfügung stellen, damit sowohl Nodes als auch deren Datentypen erkannt werden können. Die MIB-Definitionen liegen unter

```
/usr/local/mnm/tcl/lib/tnm2.1.1/mibs
```

und sollten immer auf dem neuesten Stand gehalten werden.

Alternativ können auch zusätzliche MIB's in ein beliebiges Verzeichnis gestellt werden und mittels der Environment Variable MIBDIRS eingebunden werden.

1.5 BLT Library

SyNeMa verwendet die BLT Library, die verschiedene Windowing Extensions zur Verfuegung stellt. Gegenwärtig wird die Version 1.9 verwendet.

Konfiguration wie gewohnt:

```
./configure --prefix=/usr/local/mnm/tcl
```

und mit

```
gmake
```

bzw.

```
gmake install
```

wird das Programmpaket installiert.

1.6 System-Agenten

Damit SyNeMa sinnvolle Ergebnisse liefern kann, müssen die Systemagenten laufen. Diese koennen mit Hilfe des Scripts `starte_alleagenten` gestartet werden.

Ohne diese Agenten erhält SyNeMa keine Daten und gibt die Fehlermeldung "`Error: SNMPGet: No Response`" zurück.

2 SyNeMa

2.1 Aufbau und Verwendung von SyNeMa

Imfolgenden soll der Aufbau des Programms SyNeMa erläutert werden. Zu diesem Zweck werden die verschiedenen Menüpunkte der Reihe nach erklärt.

2.1.1 Aufruf und Initialisierung von SyNeMa

SyNeMa wird aus dem Synema-Basisverzeichnis heraus mittels

```
./synema
```

gestartet.

Nach dem Laden von Libraries und MIB-Definitionen wird noch die Datei

```
.synemarc
```

abgearbeitet.

Diese Datei ist eine Textdatei, die im Home-Verzeichnis des Users liegen sollte und das Initialisieren von Get- und Set-Community und einer Hostliste für die Host-Auswahl erlaubt.

Die Communities werden folgendermassen gesetzt:

```
GET_COMMUNITY <community-name>
SET_COMMUNITY <community-name>
```

Um eine Selection List zu erstellen, wird folgendes Format verwendet:

```
BEGIN SelectionHosts
<Liste von Hostnamen, einer pro Zeile>
END SelectionHosts
```

Hier eine Beispieldatei:

```
# Kommentar; bitte keine Tabs verwenden !

# Schluesselwort GET_COMMUNITY
GET_COMMUNITY public
# Schluesselwort SET_COMMUNITY
SET_COMMUNITY public

# Schluesselwort BEGIN SelectionHosts
BEGIN SelectionHosts
   hpheger4
   sunhegering4
   sunhegering8
   sunhegering7
   ibmhegering1
END SelectionHosts
```

2.1.2 Hauptmenü

Abbildung 2 zeigt das Hauptmenü von SyNeMa.

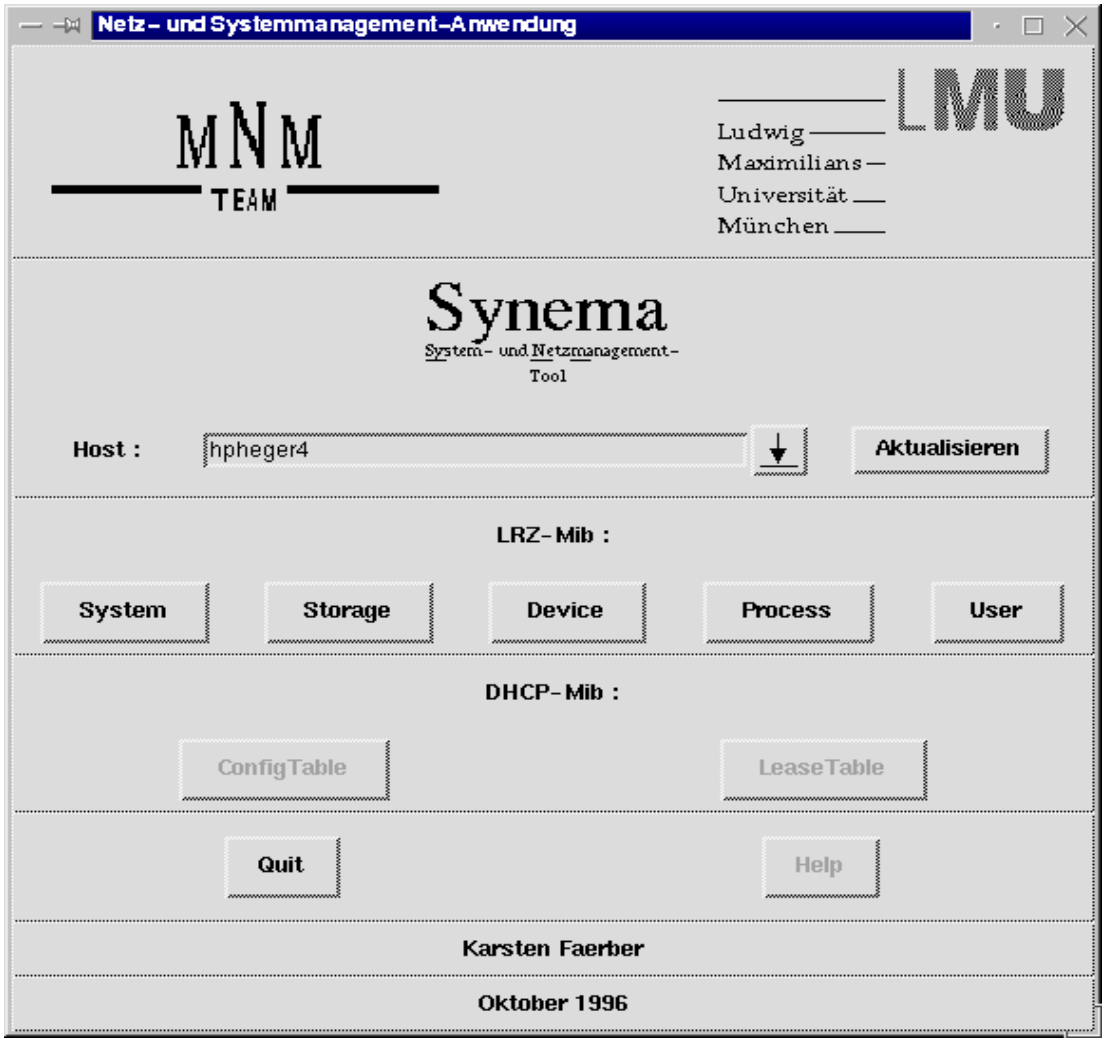


Abbildung 2

Im obersten Teil des Anfangsbildschirms befindet sich die Eingabemaske für den Host, mit dem eine Session aufgebaut werden soll (in diesem Fall 'hpheger4'). Es ist möglich eine default-Liste zur Auswahl bereit zu stellen. Dies geschieht über die Datei .synemarc, deren Aufbau im zweiten Teil dieses Kapitels erläutert wird.

Dieser obere Teil bleibt bei allen späteren Menüs erhalten und bietet die Möglichkeit, in allen Untermenüs den Host zu wechseln. Mit dem Button 'Aktualisieren' kann man die Daten erneut abfragen und darstellen lassen.

In der Mitte der Bildschirmmaske sind fünf Buttons zur Auswahl der zur Verfügung stehenden Untermenüs.

Darunter befindet sich ein bislang noch nicht implementierter Teil zur DHCP-Konfiguration mittels einer DHCP-MIB.

2.1.3 System-Menü

Abbildung 3 zeigt die Bildschirmmaske des System-Menüs.

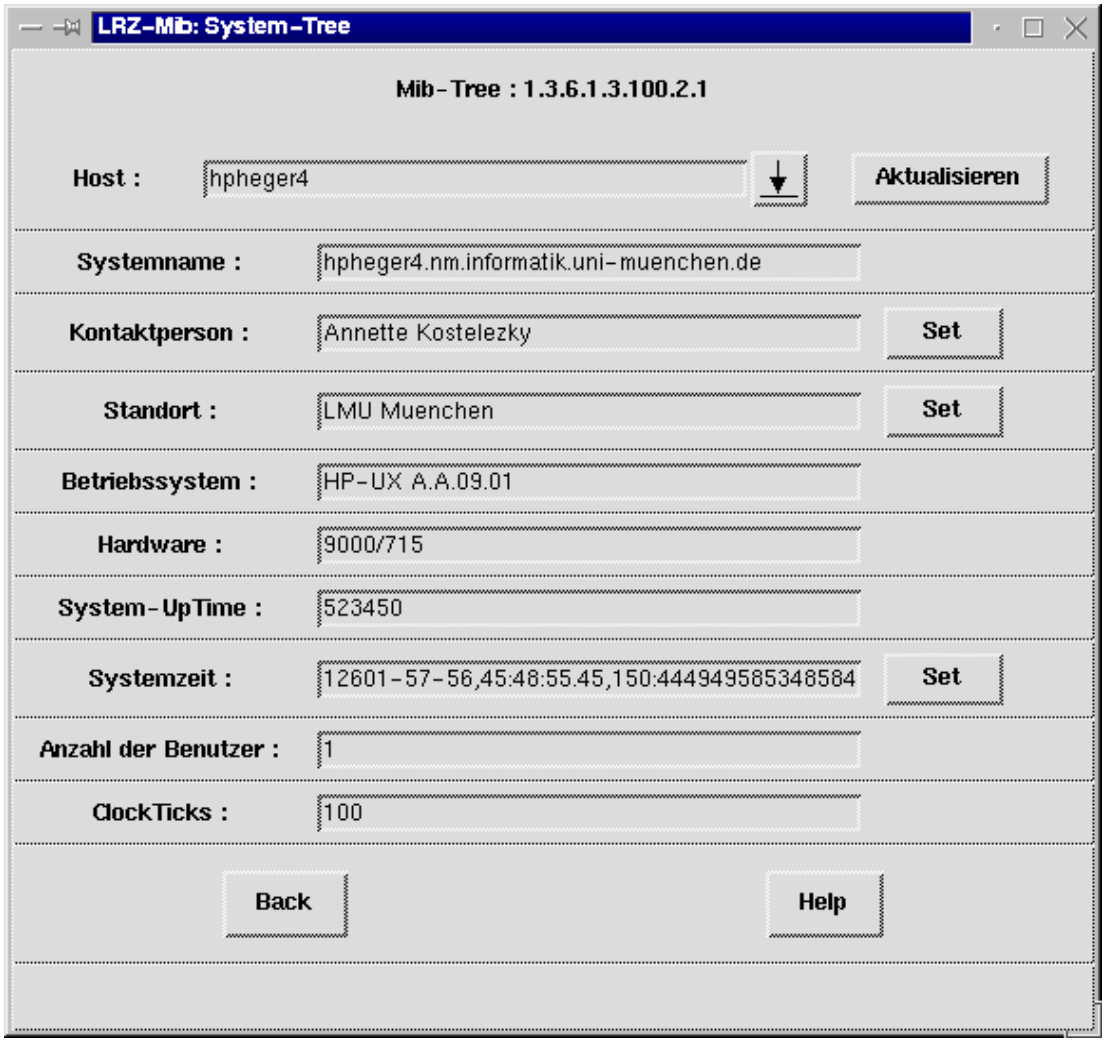


Abbildung 3

In der obersten Zeile erscheint der aktuelle Pfad im MIB-Baum (hier 1.3.6.1.3.100.2.1 also die Systemgruppe der MIB), dessen Daten in der Bildschirmmaske dargestellt werden.

Darunter, wie bereits eingangs erwähnt, der Host mit dem die Session aufgebaut wurde.

Folgende Angaben aus der Systemgruppe der MIB werden dargestellt:

- Systemname (sysName): Der Name des Systems als FQDN
- Kontaktperson (sysContact): Ein Textstring, der die verantwortliche Person für dieses System identifiziert
- Standort (sysLocation): Ein Textstring, der den Standort identifiziert
- Betriebssystem (sysOs): Ein Textstring, der die Art des Betriebssystems identifiziert
- Hardware (sysHardware): Beschreibung der Node (z.B. HP7000)
- System-Uptime (sysUptime): Die Zeit seit dem letzten Systemstart in Sekunden
- Systemzeit (sysDate): Die Systemzeit der Node

- Anzahl der Benutzer (sysUsers): Wie viele User eingeloggt sind
- ClockTicks (sysClockTicks): Die Zahl der ClockTicks der Node pro Sekunde

Kontaktperson, Standort und Systemzeit haben neben dem Ausgabefeld einen 'Set' Button. Das bedeutet, dass diese Angaben editierbar und mittels eines SNMP Set schreibbar sind.

Um einen SNMP Set durchführen zu können, ist es notwendig, den korrekten SNMP Community String anzugeben. Dieser wird mittels der Datei .synemarc angegeben. Das Programm nimmt als default-Werte für read und write jeweils 'public' an.

2.1.4 Storage-Menü

Abbildung 4 zeigt die Bildschirmmaske nach dem Aufruf des Storage-Menüs.

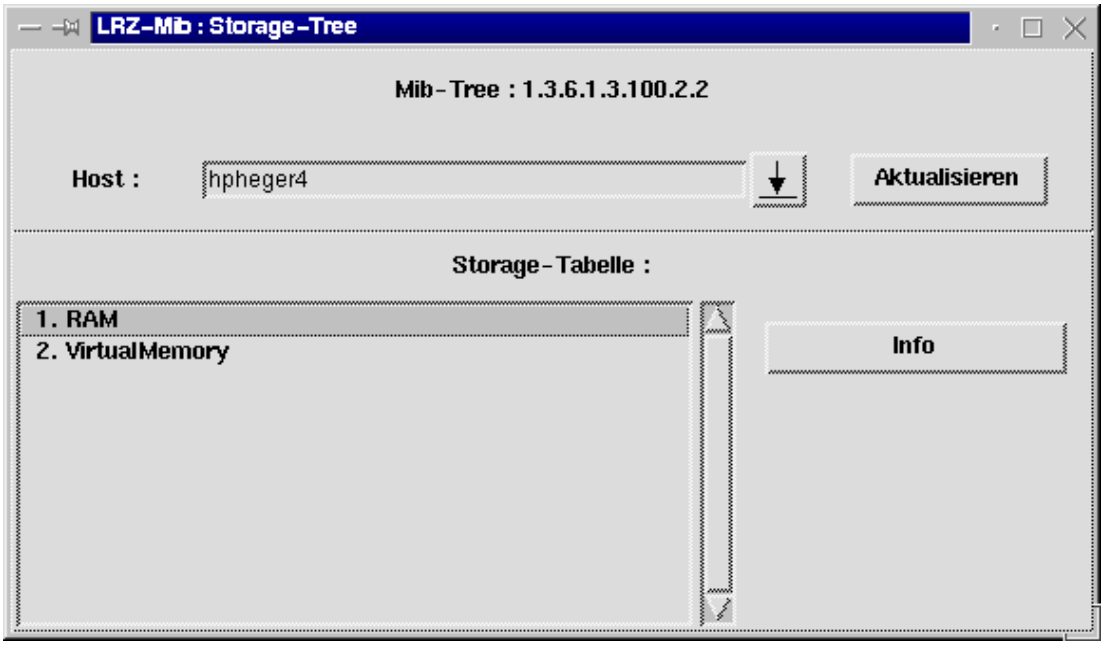


Abbildung 4

In der Storage Bildschirmmaske werden keine Plattenspeicher angegeben. Diese finden sich unter dem im nächsten Kapitel beschriebenen Device-Menü.

In Abbildung 4 wird beispielhaft die Storage-Tabelle des Rechners 'hpheger4' angegeben. Durch Doppelklick oder durch Einfachklick und anschliessendem Klick auf dem 'Info'-Button werden weitere Informationen zu dem jeweils ausgewählten Item geliefert.

Im Moment sind die Typen auf RAM und Virtual-Memory beschränkt, aber SyNeMa kann flexibel auf Erweiterungen der MIB und des Agenten reagieren, weil es einfach alle Instanzen auflistet, die von Agenten zureuckgegeben werden.

Nach einem Klick auf Info erhält man die in Abbildung 5 dargestellte Maske.

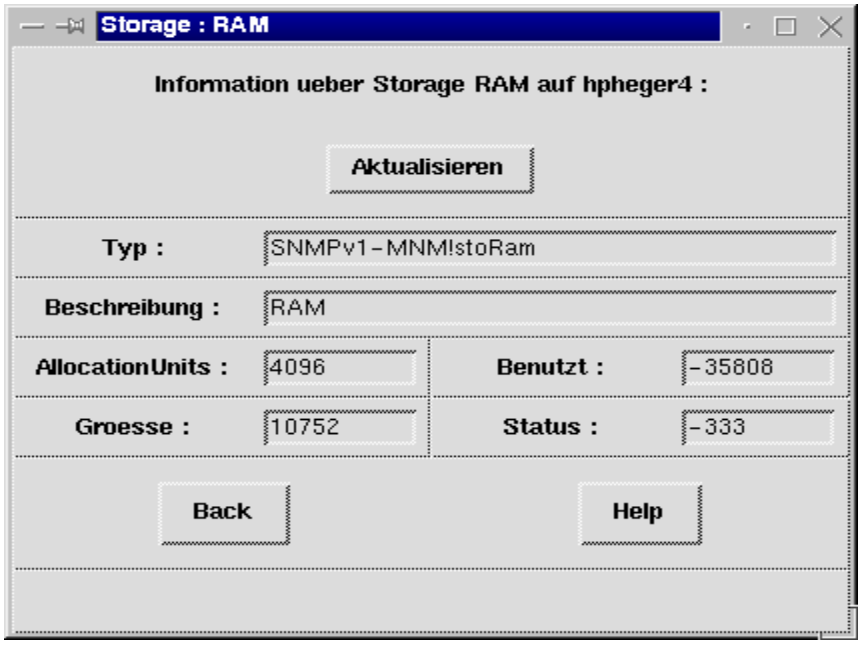


Abbildung 5

Dabei haben die Einträge folgende Bedeutungen:

- Typ: Der Typ der Storage (im Beispiel SNMPv1-MNM!stoRAM), der eindeutig sein muss.
- Beschreibung: Eine Text-String, der eine Beschreibung des Typs beinhaltet.
- Allocation Units: Die Grösse eines 'chunks' der Instanz im Bytes (im Beispiel 4096 Bytes beim RAM).
- Grösse: Die Gesamtgrösse in Kilobytes. Die Angaben können von Programmen, die die Speichergrösse auf andere Weise feststellen, abweichen.
- Benutzt: Anzahl der Kilobytes, die von der Instanz verwendet werden.
- Status: Die Prozent der Instanz, die gerade verwendet werden.

Bei der Betrachtung der Abblidung 5 fällt auf, dass die Daten für Benutzt und Status keine sinnvollen Werte enthalten. Das liegt an Problemen, die der Subagent bei der Bestimmung dieser Daten hatte. Möglicherweise kann dies in späteren Versionen verbessert werden.

2.1.5 Device-Menü

Abbildung 6 zeigt die Bildschirmmaske des Device-Menüs.

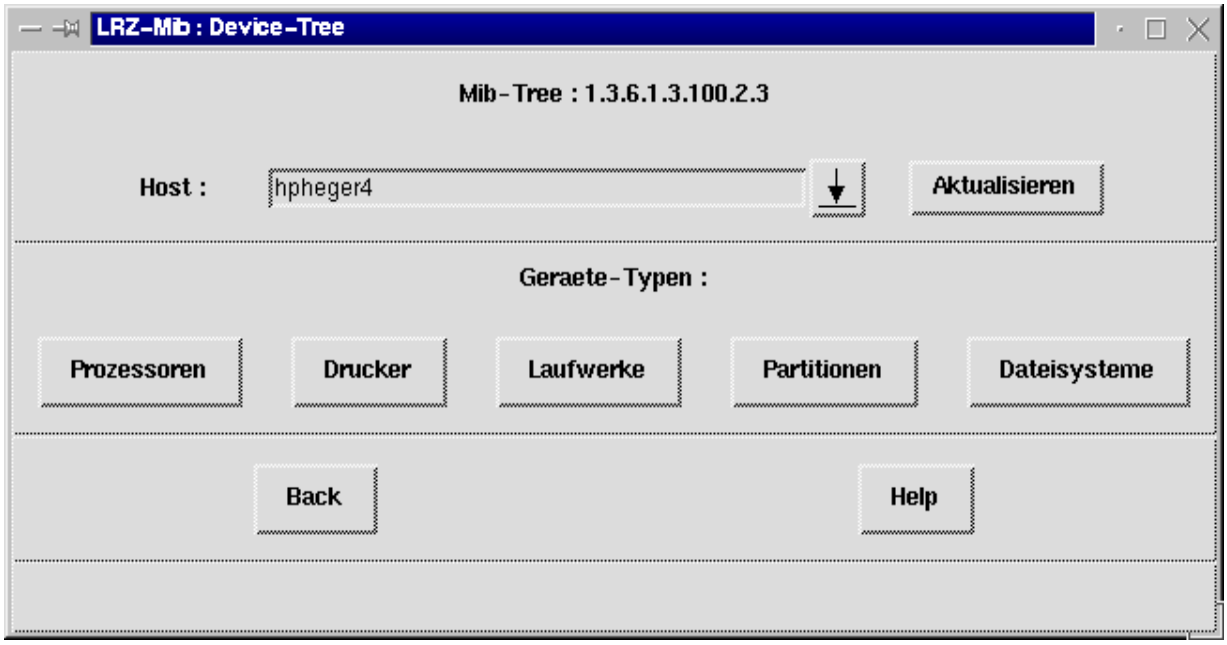


Abbildung 6

Folgende Device-Typen sind im der MIB vorgesehen:

- **processor**
- **printer**
- **disk**
- **partition**
- **filesystem**
- network
- video
- audio
- modem
- poiting
- tape
- openprom
- bus
- controller

Allerdings sind nur die fett gedruckten Typen gegenwärtig durch den Agenten implementiert. Erweiterungen des Agenten können mit SyNeMa nur dargestellt werden, wenn das Programm dahingehend erweitert wird.

Für jeden dieser Typen gibt es in der MIB eine eigene Gruppe. Diese werden nachfolgend kurz beschrieben.

Prozessor-Gruppe

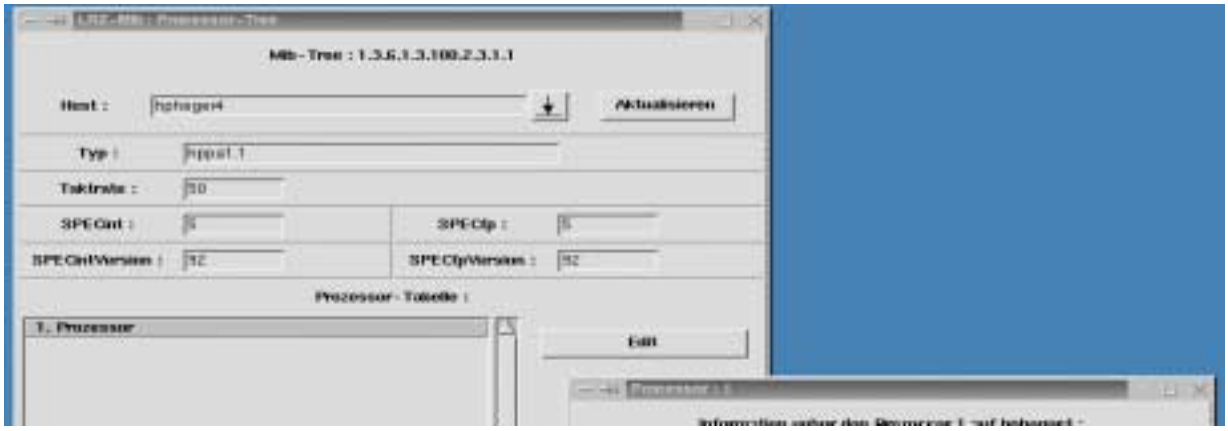
In der Prozessor-Gruppe können folgende Informationen über die im System vorhandenen Prozessoren abgefragt werden:

- Typ: Typ des Prozessors (z.B. hppa1.1).
- Taktrate: Die Taktrate des Prozessors
- SPECint: Das Ergebnis des SPECint Tests. Die dargestellten Ergebnisse sind im Normalfall nicht zutreffend, da vorher ein SPECint Test gemacht werden müsste. Die dargestellten Informationen werden zwar vom Agenten übergeben, dieser kann sie aber nicht überprüfen oder selbst einen Test durchführen.
- SPECintVersion: Die Version der Benchmark.
- SPECfp: Das Ergebnis des SPECfp Tests. Für die Ergebnisse gilt dasselbe wie für den SPECint Test.
- SPECfpVersion: Die Version der SPECfp Benchmark.

Für jeden Prozessor können weitere Informationen abgerufen werden. Dies sind:

- Nummer: Interne Nummer des Prozessors
- Operational Status: Der Status des Prozessors nach ISO-10164-2. Möglich sind enabled und disabled. Bei Single-Prozessor Systemen immer auf enabled.
- UsageStatus: Hier sind idle oder busy möglich.
- AdministrativeStatus: Der Status nach ISO-10164-2. Möglich sind unlocked, locked und shutting down.
- UserTime: Prozent der Nutzung des Prozessors im User-Modus der letzten 0.5 Sekunden.
- NiceTime: Wie UserTime, nur für 'niced' Prozesse
- System Time: Prozent der Nutzung im System-Modus.
- IdleTime: Prozent der Zeit, die der Prozessor idle war in den letzten 05. Sekunden.
- Zustand: Eine Zahl, die den Zustand des Systems angibt. Bislang noch ohne Bedeutung.

Abbildung 7 zeigt eine Auswahl der darstellbaren Informationen für einen ausgewählten Prozessor.


Mib-Tree : 1.3.6.1.3.100.2.3.1.1
Host :
Aktualisieren
Typ :
Taktrate :
SPECint :
SPECfp :
SPECintVersion :
SPECfpVersion :
Prozessor - Tabelle :
1. Prozessor
Edit