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.

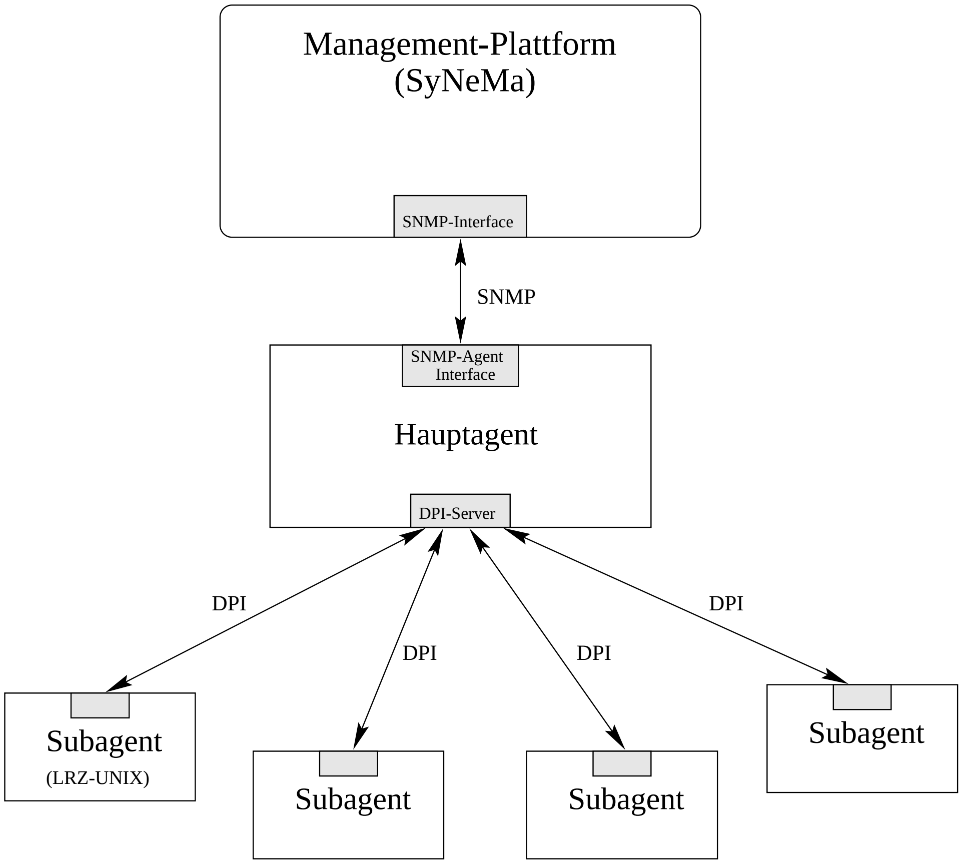


Abbildung 1

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.

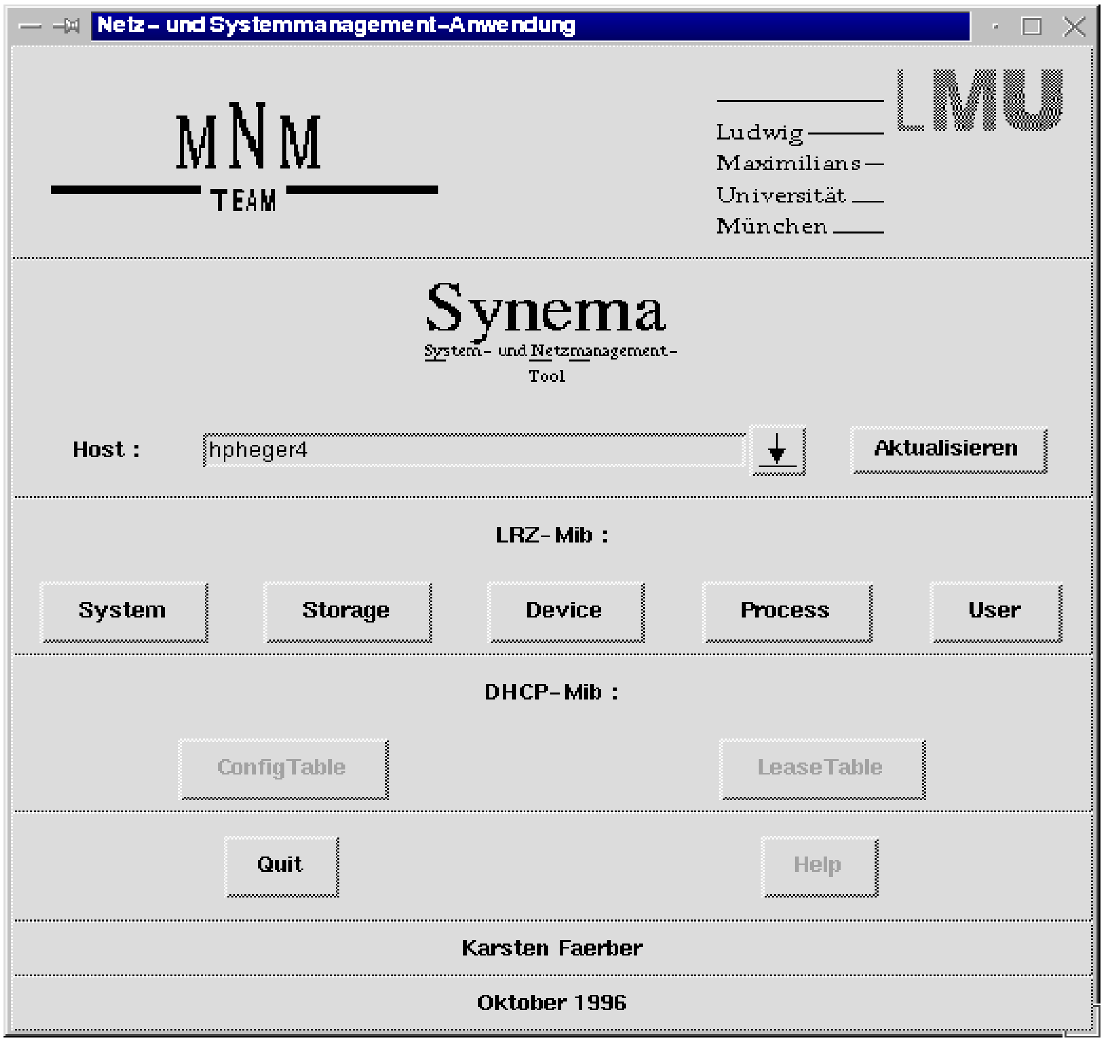


Abbildung 2

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.

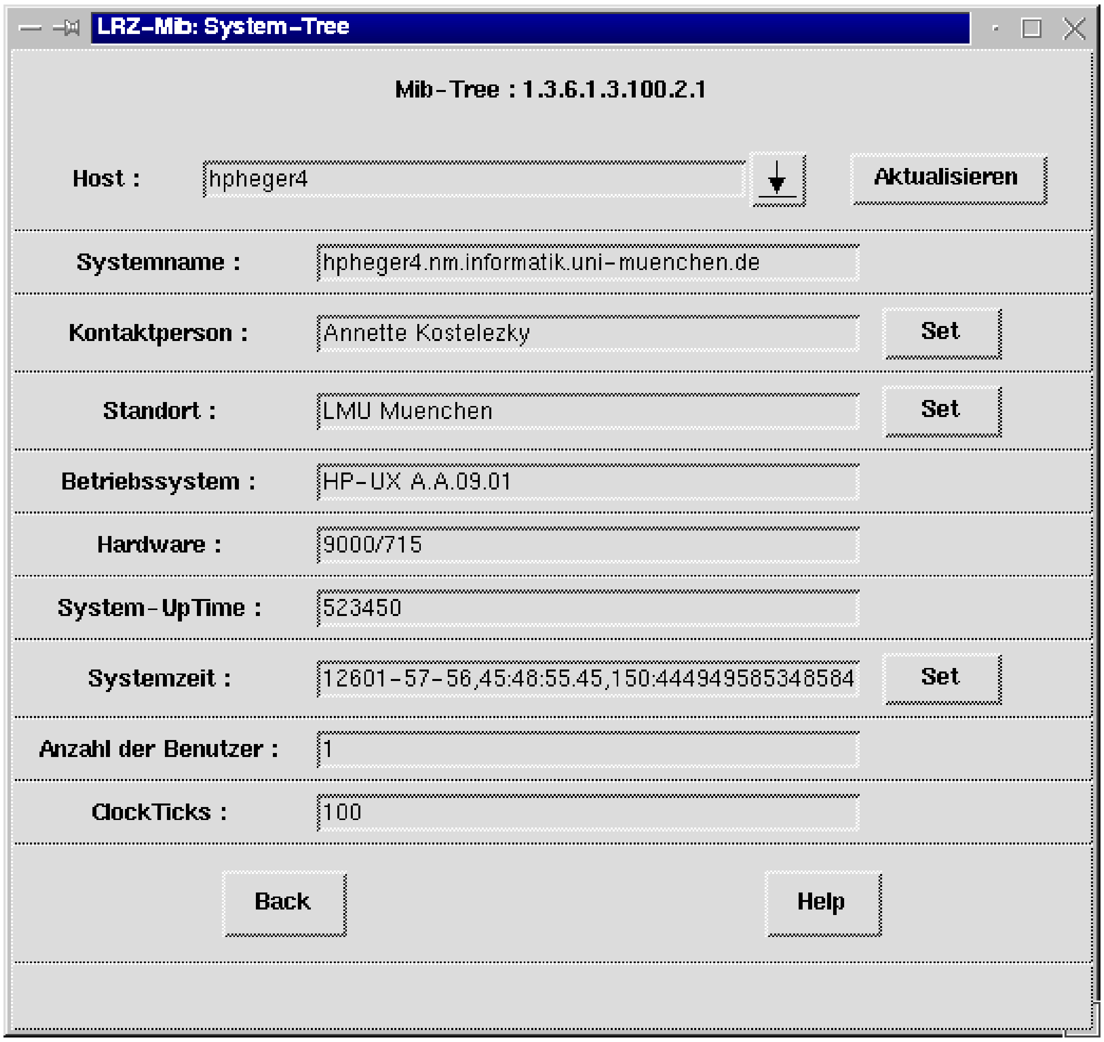


Abbildung 3

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.

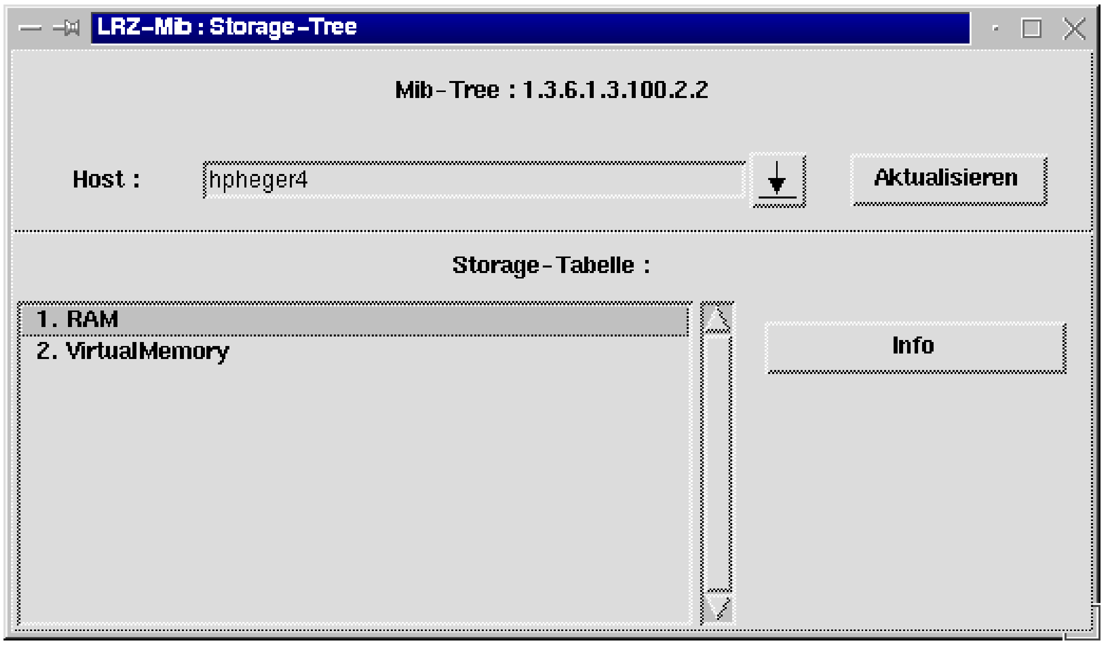


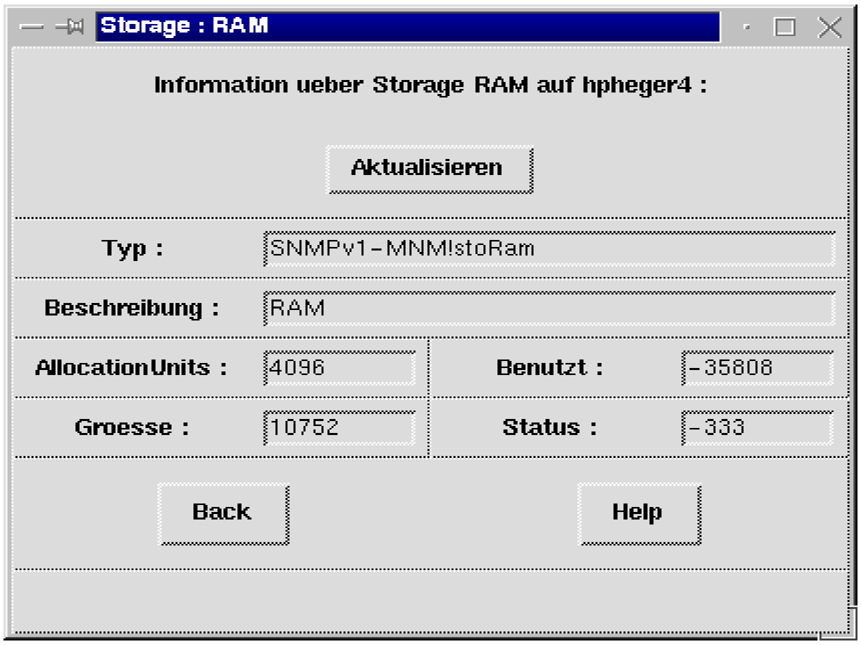
Abbildung 4

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.



Abblidung 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.

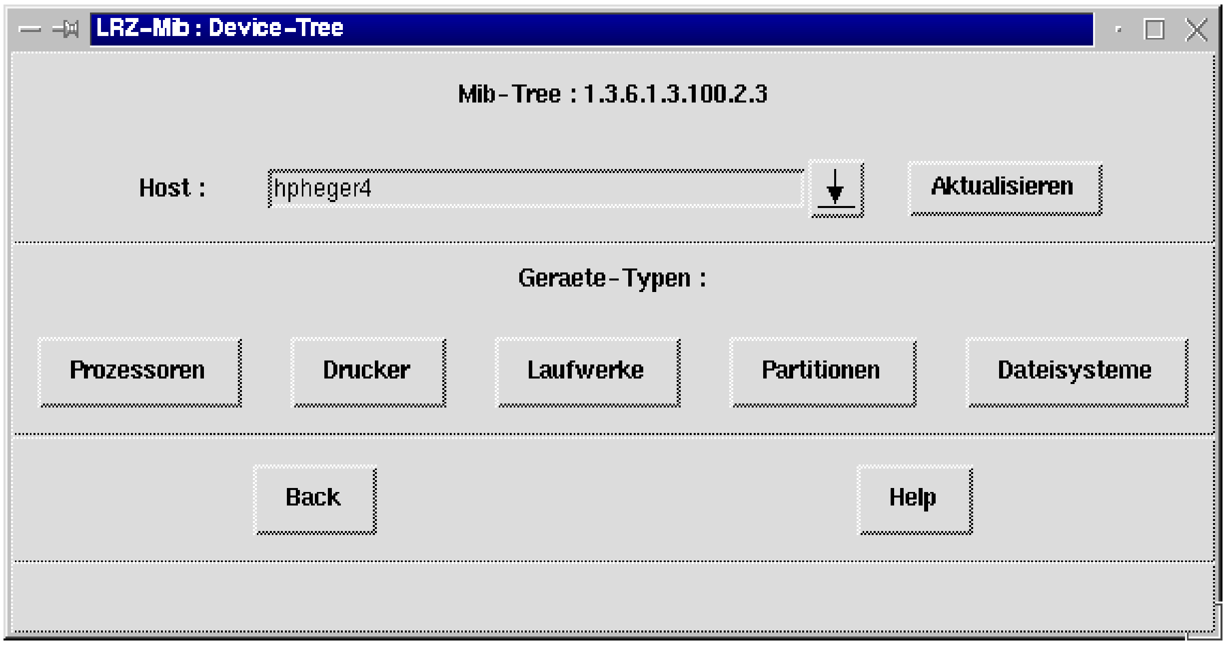


Abbildung 6

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

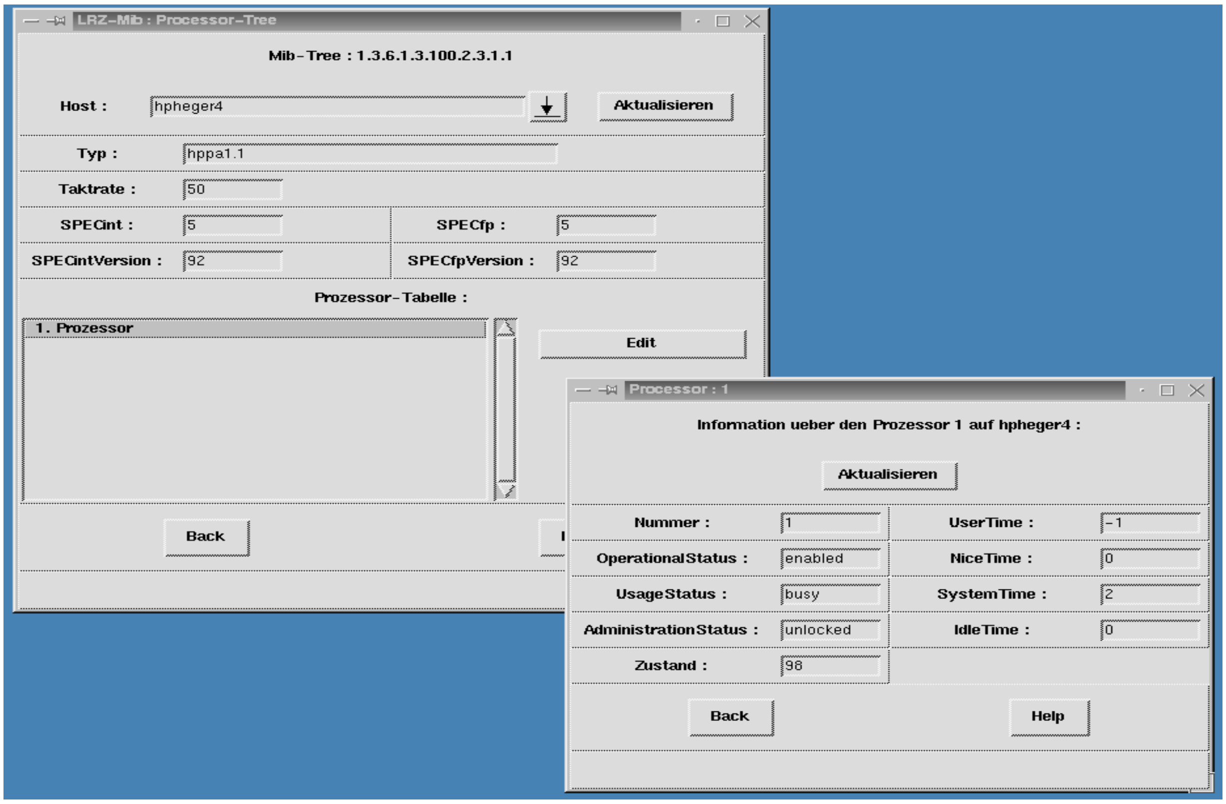
In der Prozessor-Gruppe können folgende Informationen über die im System vorhandenen Prozessoren abgefragt werden:

- 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.



Printer-Gruppe

Hier können Informationen zu lokalen Druckern und Drucker-Qüues abgerufen werden. Für lokale Drucker sind folgende Informationen abrufbar:

- DruckerName: Der Name des Druckers als Textstring.
- DruckerIndex: Eine Identifikationsnummer.
- DruckerOrt: Ein Textstring, der den physischen Standort beschreibt.
- OperationalStatus: Läuft der Drucker ? Möglich sind enabled oder disabled.
- UsageStatus: Nutzungsstatus des Druckers. Möglich sind idle oder busy.
- Administrative Status: Möglich sind unlocked, locked und shuttung down.
- Availability Status: Möglich sind inTest, failed, powerOff, offline, not installed.
- Aktion: Möglich sind hier enable, disable, lock, unlock, shutting down, test, online.

Für Queues sind abrufbar:

- Name: Name der Queue.
- Status: enabled oder disabled.
- DruckerOrt: Textstring, der den Ort beschreibt.

- Aktion: accept oder reject.
- DruckerIndex: Eine Identifikationsnummer.
- DruckerName: Der Name des Druckers. Durch Klicken auf den Info-Button daneben kommt man in die Bildschirmmaske des lokalen Druckers mit diesem Namen.

Anschliessend wird die Drucker-Queue dargestellt. Man kann einen Druck-Job auswählen und durch Doppelklick oder mit dem Info-Button weitere Infos ausgeben lassen. Dies sind:

- Queue: Die Queue Nummer.
- ID: Eine eindeutige ID des Druck-Jobs.
- Besitzer: Der Besitzer des Auftrags.
- Eintritt: Zeitpunkt der Vergabe des Auftrags in Datum und Uhrzeit. Kann 'unknown' beinhalten.
- Priorität: Die Priorität des Jobs. Das System, auf dem der Job gedruckt werden soll, kann möglicherweise keine Prioritäten behandeln. In diesem Fall ist die Priorität ohne Wirkung.
- Grösse: Die Grösse des Jobs in Kilobyte.
- Gedruckt: Die Anzahl der bereits gedruckten Kilobytes.
- Aktion: Möglich sind: cancel, hold und resume.

In Abbildung 8 ist beispielhaft eine Auswahl einer Druckerqueue dargestellt (allerdings ohne Druckjobs).

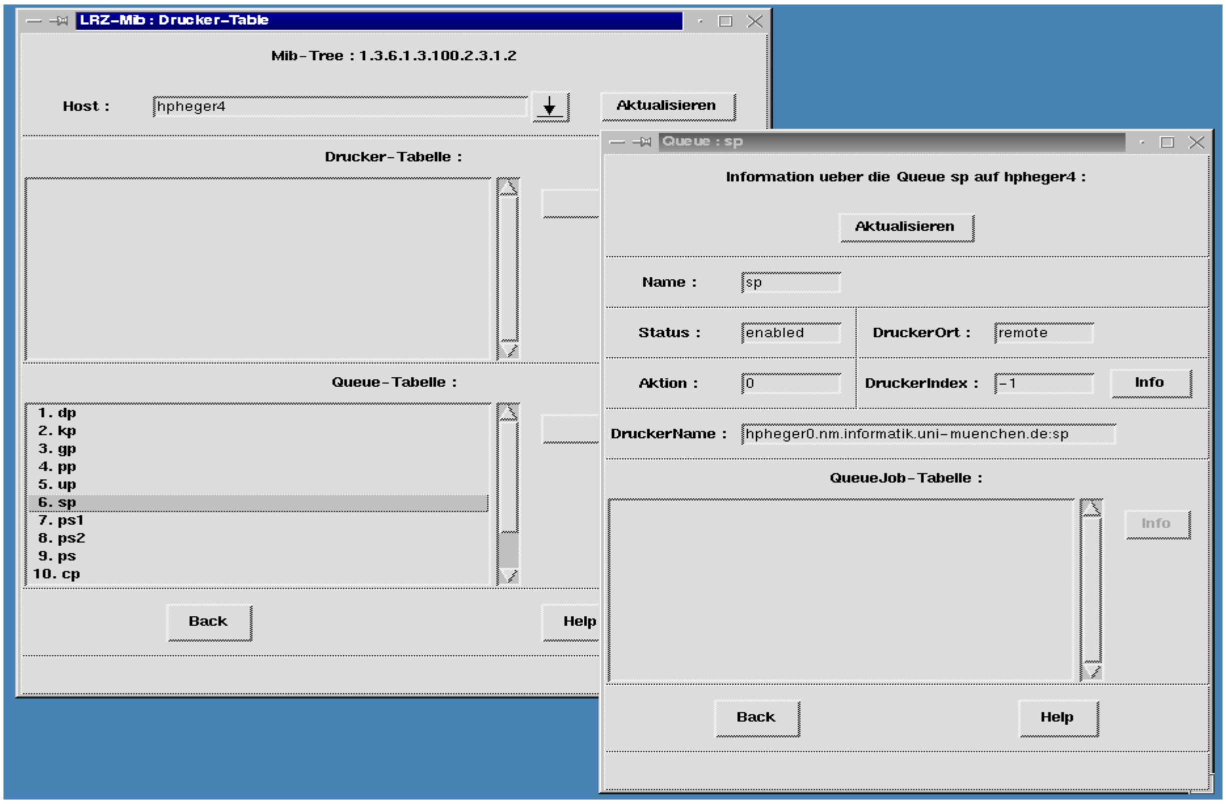


Abbildung 8

Es hat sich bei der Benutzung herausgestellt, dass das Auslesen der Druck-Jobs einer Queue durch den Agenten so lange dauert, dass ein SNMP-Timeout eintritt, bevor die Daten von SyNeMa empfangen werden können. Bislang konnte noch keine Lösung für dieses Problem gefunden werden.

Laufwerks-Gruppe

Die Laufwerks-Gruppe umfasst alle Laufwerke (Harddisks, CD-ROM, etc.) des Systems. Diese werden in einer Tabelle angegeben und folgende Informationen dazu können abgerufen werden:

- Zugriff: Die Zugriffs-Rechts für diese Platte. Möglich sind: read-write und read-only.
- Medium: Folgende Media-Typen sind möglich: unknown, hardDisk, opticalDisk, floppyDisk.
- Wechselbar: Möglich sind hier: removable und not-removable.
- Kapazität: Die Kapazität des Laufwerks in Kilobytes.
- Bezeichner: Der Bezeichner des Laufwerks. Auf UNIX-Systemen in der üblichen '/dev'-Bezeichnung
- Busy: Die Anzahl der Time-Ticks, die die Platte aktiv war.
- Transferrate: Transferrate in Kilobytes pro Sekunde. Dies ist kein akkurater Wert, da der Agent bislang noch keine Möglichkeit hatte, diesen zu bestimmen.
- Reads: Die Anzahl der Read-Operationen des Laufwerks.
- Words: Die Anzahl der übertragenen Wörter.
- Transfers: Die Anzahl der Transfers des Laufwerks.

In Abbildung 9 ist dargestellt, wie eines der beiden Laufwerke des hosts 'hpheger4' ausgewählt und dessen Informationen dargestellt werden.

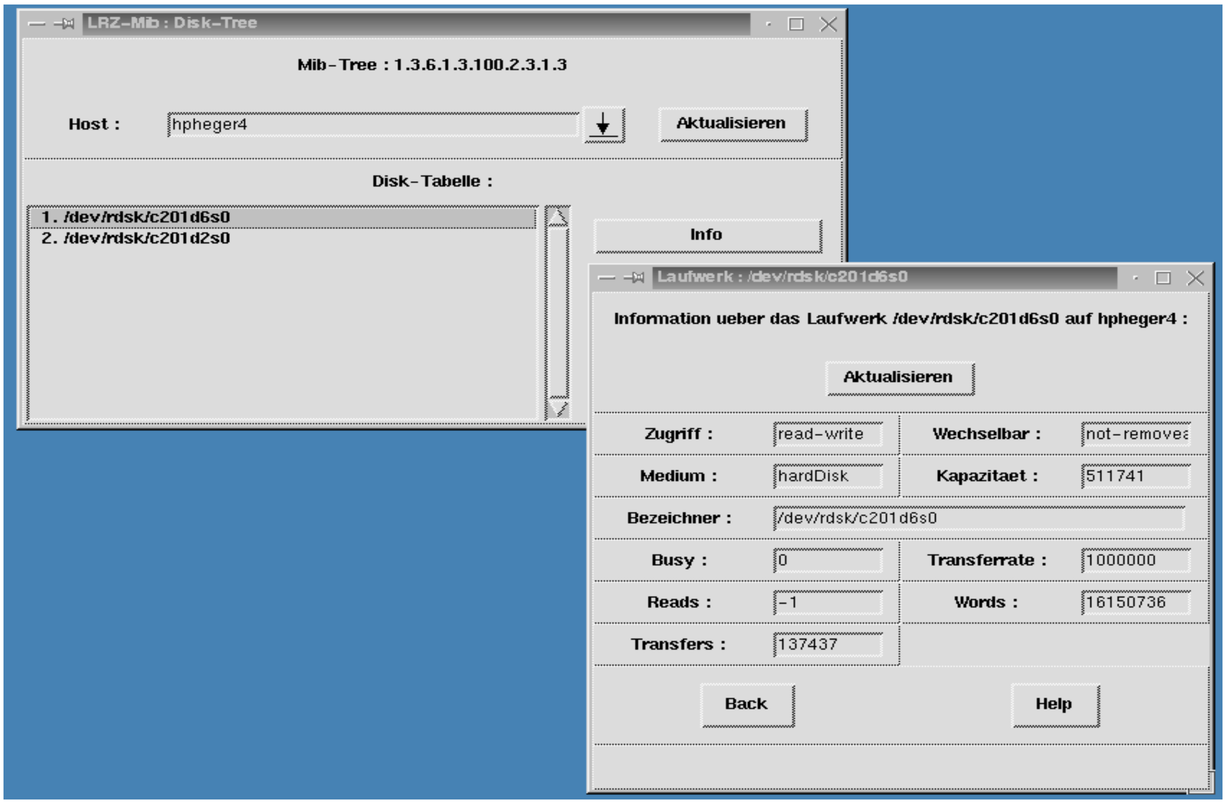


Abbildung 9

Die Partitions-Gruppe

In der Partitions Gruppe werden Informationen zu den gerade gemounteten Partitionen des ausgewählten Hosts dargestellt.

Alle gemounteten Dateisysteme sind in einer Tabelle zusammengefasst. Mit Doppelklick oder Anwahl und klicken des Info-Buttons erhält man mehr Informationen zur Ausgewählten Partition. Im einzelnen sind dies:

- Label: Textstring mit dem Label dar Partition.
- ID: Der Deskriptor der Partition, überlichweise ein Device als '/dev'-Schreibweise in UNIX.
- Size: Die Grösse der Partition.
- DiskIndex: Ein Zeiger auf das Laufwerk der Partition. Mit dem Info-Button kommt man in die entsprechende Laufwerks-Maske.
- FSIndex: Ein Zeiger auf das Filesystem der Partition. Mit dem Info-Button kommt man in die Filesystem-Maske und erhält zusätzliche Informationen über das Filesystem.

Abbildung 10 zeigt eine ausgewählte Partition auf 'hpheger4' und die Infos dazu.

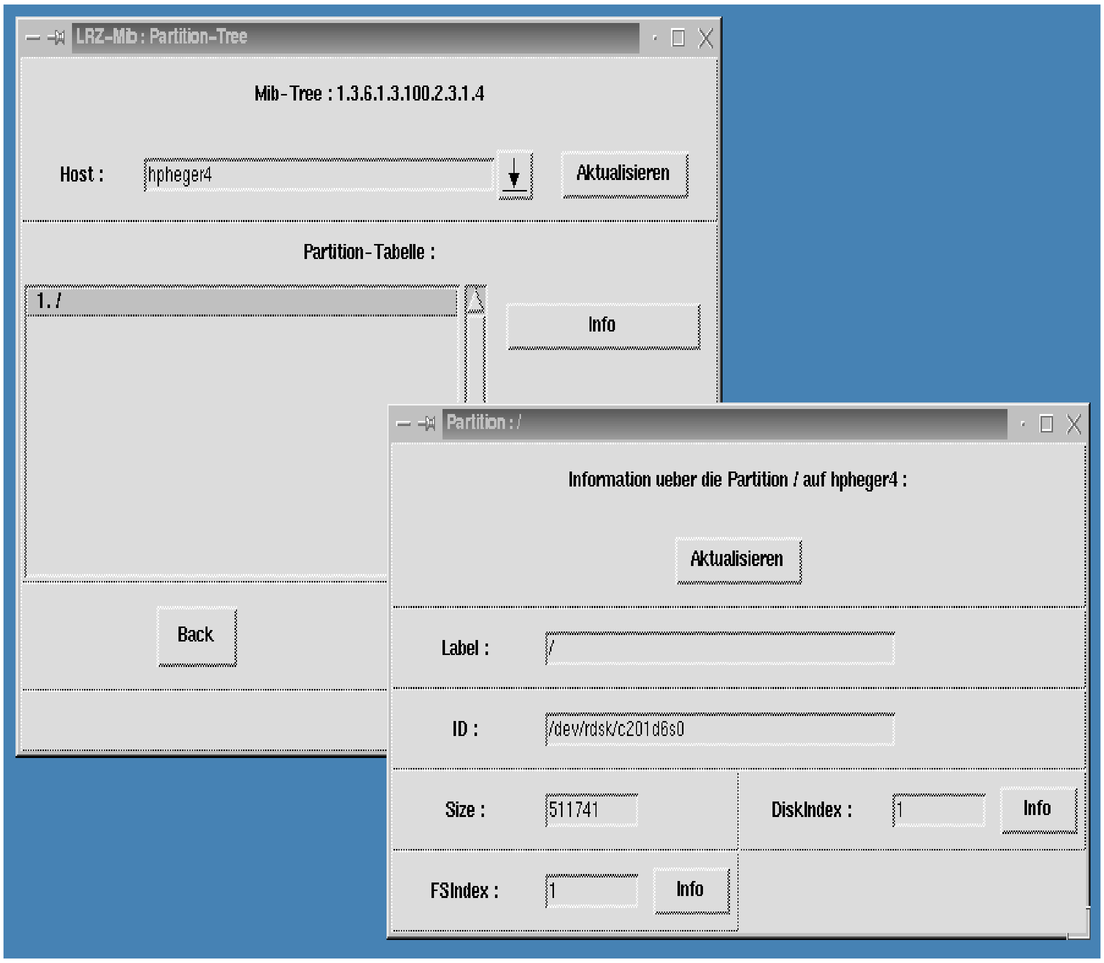


Abbildung 10

Die Dateisystem-Gruppe

In der Dateisystem-Gruppe werden gemountete und vordefinierte Dateisysteme in einer Tabelle angezeigt. Falls ein vordefiniertes Dateisystem gemountet ist, erscheint es natürlich in beiden Tabellen.

Für die vordefinierten Dateisysteme können folgende Informationen angezeigt werden:

- Name: Der Name des Dateisystems als Textstring, üblicherweise als UNIX-Device
- Typ: Eine Typangabe zu dem Dateisystem. Mögliche Angaben sind: Other, Unknown, BerkeleyFFS, Sys5FS, Fat, HPFS, MacHFS, MFS, VNode, NTFS, Journaled, iso9660 (CD-ROM), NFS, Netware, AFS, DFS, RFS, BFS, UFS, HFS.
- MountPoint: Pfad, unter dem das Dateisystem gemountet ist.
- MountOptions: Optionen, mit denen das Dateisystem gemountet wurde.
- ID: Eine ID, die die Partition identifiziert.
- Aktion: Aktionen, die ausgeführt werden können. Im Moment ist nur das mounten des Dateisystems möglich (1). Dies funktioniert allerdings nicht in dem gewünschten Masse.

- Backup-Level-Tabelle: Eine Tabelle, der in Backups von Level 0 bis 10 angegeben werden, sofern sie durch den Agenten berücksichtigt werden.

Für gemountete Dateisysteme können folgende Informationen angezeigt werden:

- Name: Der Name des gemounteten Dateisystems in üblicher UNIX Notation, also z.B. der Device-Name bei vordefinierten Dateisystemen oder <host>:/<pfad> bei remote Dateisystemen.
- Typ: Eine Typangabe (siehe bei vordefinierten Dateisystemen).
- MountPoint: Pfad, unter dem das Dateisystem gemountet ist.
- MountOptions: Optionen, mit denen das Dateisystem gemountet wurde.
- BlockSize: Die Grösse der Blöcke des Dateisystems in Byte.
- Block Count: Die Gesamtanzahl der Blöcke des Dateisystems.
- BlocksFree: Anzahl der freien Blöcke.
- BlocksAvailable: Anzahl der Blöcke, die für nicht-privilegierte User frei sind.
- InodeCount: Anzahl der Inodes (Angabe ist abhängig vom Dateisystem-Typ, bei NFS kann z.B. kein InodeCount angegeben werden). Falls keine Angabe möglich, wird der Wert auf -1 gesetzt.
- InodeAvailable: Anzahl der verfügbaren Inodes. Falls keine Angabe möglich, wird der Wert ebenfalls auf -1 gesetzt.
- Quota: Mögliche Werte: quotaon(1) und quotaoff(2). Kann per SNMP-Set eingestellt werden.
- Action: Für gemountete Dateisysteme sind bislang keine Aktionen vorgesehen.

Abbildung 11 zeigt wie ein gemountetes Dateisystem (hpgeher0:/proj) ausgewählt und dessen Infos angezeigt werden.

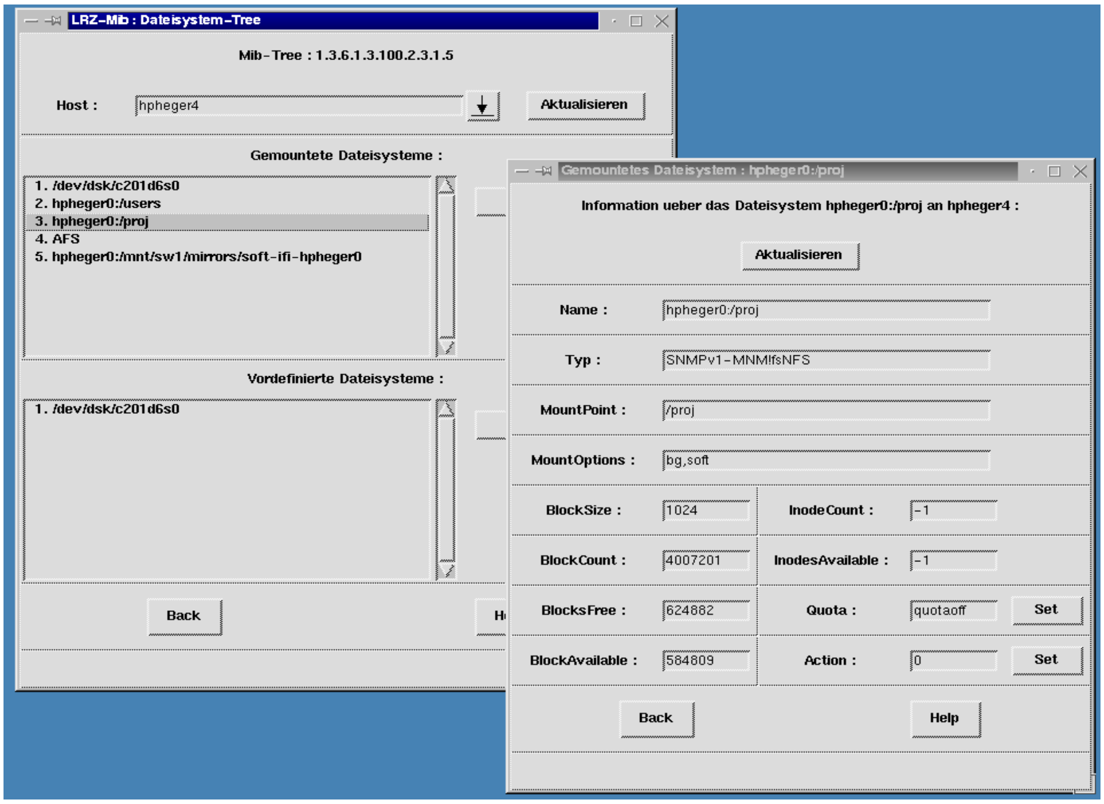


Abbildung 11

2.1.6 Das Process-Menü

Mit Hilfe des Process-Menüs können Angaben über laufende Prozesse des ausgewählten hosts abgerufen werden und Manipulationen (wie z.B. Signale an Prozesse senden) über SNMP-Sets ausgeführt werden.

Nach Aufruf des Menüs erhält man eine graphische Darstellung des Loads in der UNIX-üblichen Aufteilung über 1, 5 und 15 Minuten. Zusätzlich bekommt man folgende Angaben:

- max. Prozess-Anzahl: Die Höchstzahl der Prozesse auf dem Host.
- max. Prozess-Grösse: Die maximale Grösse, die ein Prozess im Speicher allokieren kann (in Kilobytes).
- akt. Prozess-Anzahl: Die Anzahl der gerade im System befindlichen Prozesse.
- akt.-max. Prozess-Grösse: Die Grösse des momentan grössten Prozesses im System.
- akt.-max. Prozess-CPU-Zeit: Die CPU-Zeit des am längsten laufenden Prozesses in Sekunden.

Zusätzlich befindet sich im unteren Abschnitt des Fensters eine Tabelle mit allen sich gerade im System befindlichen Prozessen (Prozesstabelle).

Durch Auswahl eines Prozesses und klicken des Edit Buttons erhält man detailierte Informationen zu diesem Prozess:

- Name: Name des Prozesses

- PID: Die Process-ID
- Parent PID: Die PID des Parent-Prozesses. Falls kein direkter Parent existiert, ist dies immer der init-Prozess (1). Durch Anklicken von Info kann man den Process-Edit des Parents aufrufen.
- UID: Die User-ID unter der der Prozess läuft. Durch Anklicken von Info erhält man detailierte Informationen zu dieser ID (Beschreibung unter User-Menü).
- GID: Group-ID unter der der Prozess läuft, falls feststellbar, sonst -1. Mit Anklicken von Info erhält man auch hier detailierte Informationen.
- Status: Möglich sind: sleeping, ready, running, idle, zombie, stopped.
- Grösse: Die Grösse des Prozesses in Kilobyte.
- CPU-Time: Summe von User- und Systemzeit, die der Prozess verbraucht hat.
- Priorität: Die Priorität des Prozesses.
- Nice: Das Setzen eines Wertes mit Set bewirkt ein Renice des Prozesses.
- Signal: Mit Set kann ein Signal an den Prozess geschickt werden.
- SSYS: True, wenn Swapper oder Pager.
- SLOAD: True, wenn Prozess im Hauptspeicher ist.
- SULOCK: True, wenn der User ein Swappen unterbunden hat.
- SSWAP: True, wenn Prozess geswappt ist.
- PHYSIO: True, wenn Prozess physischen I/O bewirkt.

Mit dem Button Graphic: CPU-TIME kann man sich eine Gesamtübersicht der CPU-Zeiten graphisch anzeigen lassen.

Mit dem Button Graphic: SIZE kann man sich eine Gesamtübersicht der Prozess-Grössen graphisch anzeigen lassen.

Abbildung 12 zeigt das Prozess-Menü und einen ausgewählten Prozess mit detailierten Angaben.

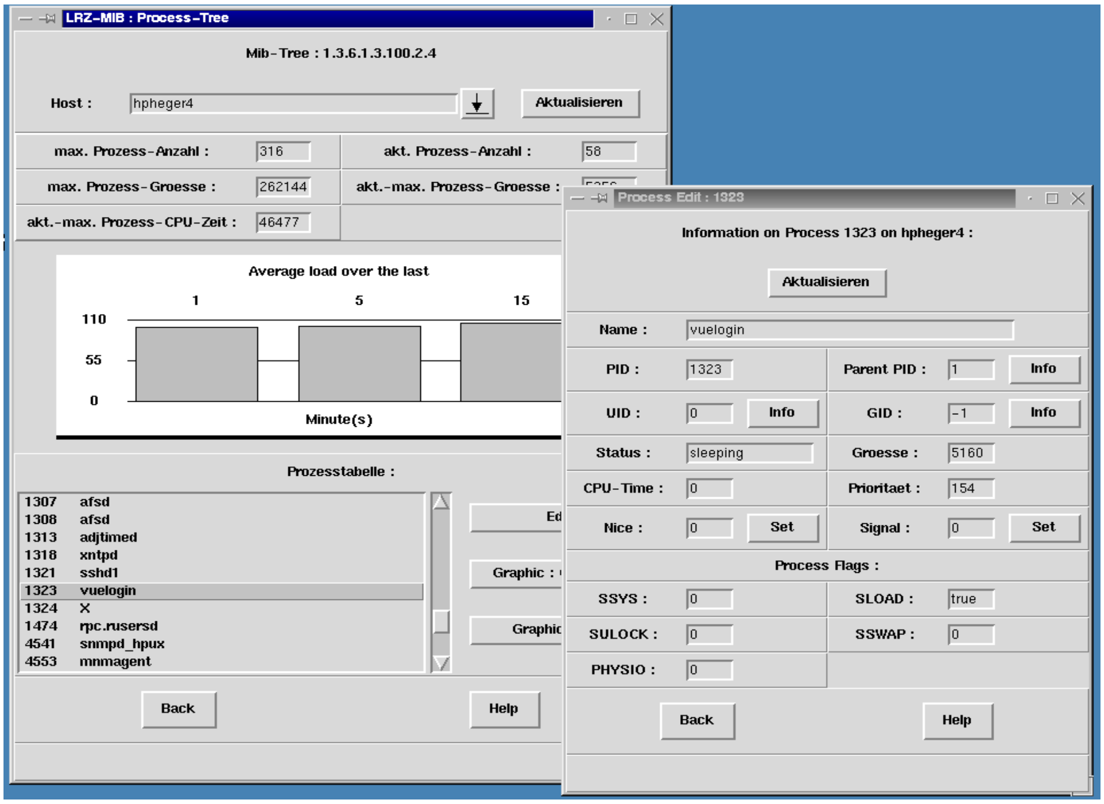


Abbildung 12

2.1.7 Das User-Menü

Das User-Menü dient zum Anzeigen und Manipulieren von Benutzer- und Gruppen-Informationen. Zusätzlich kann man Informationen zu den gegenwärtig eingeloggten Benutzern abrufen.

Die User-Gruppe

Nach Aufrufen der User-Gruppe werden alle User-Einträge des hosts angezeigt. Der Agent kann bis dato allerdings nur die Einträge der /etc/passwd Datei anzeigen und keine NIS oder NIS+ Einträge oder ähnliches.

Mit Edit, New oder Delete können die Einträge manipuliert werden. Folgende Informationen können abgerufen werden:

- Login: Die Kennung des Users.
- Passwort: Der Passwort Eintrag aus der /etc/passwd Datei (verschlüsselt).
- ID: Die UID der Kennung.
- GroupID: Die GID der Kennung.
- Comment: Kommentar aus /etc/passwd.
- Home: Das Home-Verzeichnis der Kennung.
- Shell: Die Login-Shell der Kennung.
- Status: Status der Kennung.

Zusätzlich wird eine Tabelle mit den Quotas der Kennung angegeben.

Die Group-Gruppe

ähnlich wie bei der User-Gruppe können hier Gruppen-Einträge manipuliert werden. Auch hier werden nur Informationen aus /etc/group gelesen.

Folgende Informationen können ausgegeben werden:

- Name: Der Name der Gruppe.
- Passwort: Passwort, normalerweise auf *.
- ID: Die ID der Gruppe.
- Status: Der Status der Gruppe.

Ausserdem wird eine Tabelle mit den Kennungen der Benutzer der Gruppe ausgegeben.

Die Who-Gruppe

Man kann mit der Who-Gruppe abfragen, wer auf dem host eingeloggt ist und zu den einzelnen Usern die Angaben der User-Gruppe ausgeben lassen.

Abbildung 13 zeigt die User-Gruppe mit Auswahl des Users 'admin'.

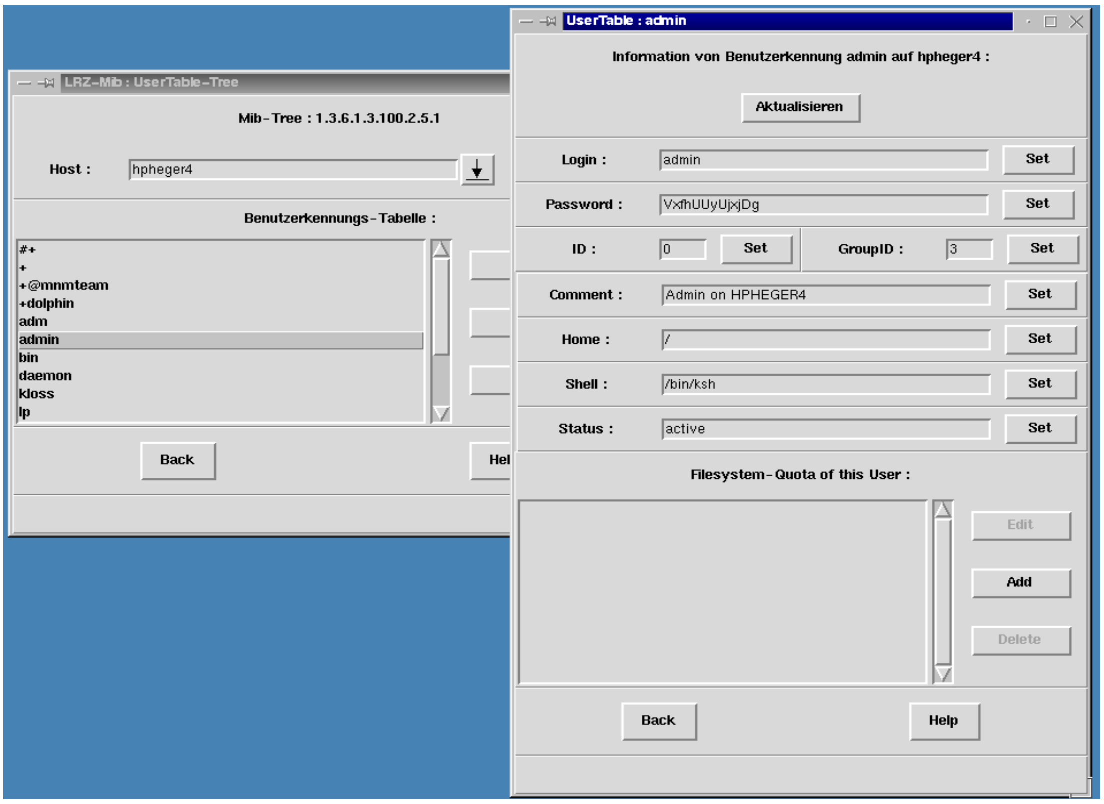


Abbildung 13

2.2 Strukturierung des Programmcodes

Da dieses Fortgeschrittenenpraktikum nur eine Erweiterung und Ergänzung des bereits bestehenden Programms SyNeMa darstellt, wird im Rahmen dieses Kapitels nicht bis in jede Einzelheit auf den Code eingegangen, da dies unter dem betreffenden Fortgeschrittenenpraktikum nachzulesen ist.

Synema ist in Tcl/Tk programmiert und in einzelne Programmodule unterteilt, um die Uebersicht zu erleichtern. Abbildung 14 zeigt die Struktur der einzelnen Module.

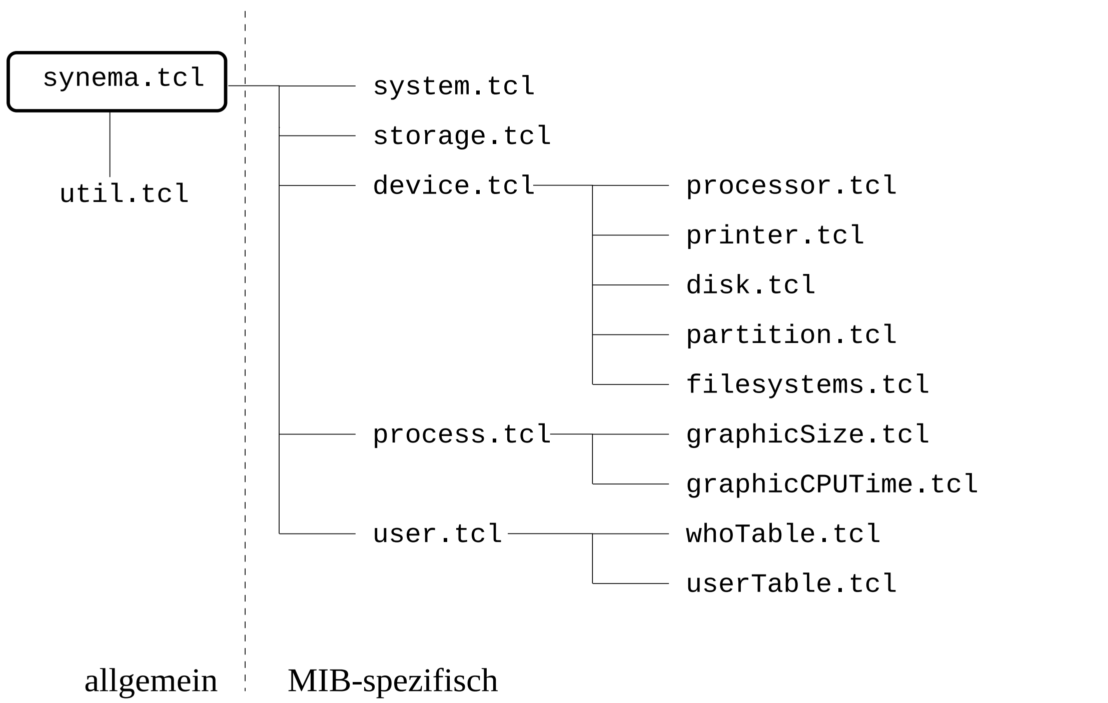


Abbildung 14

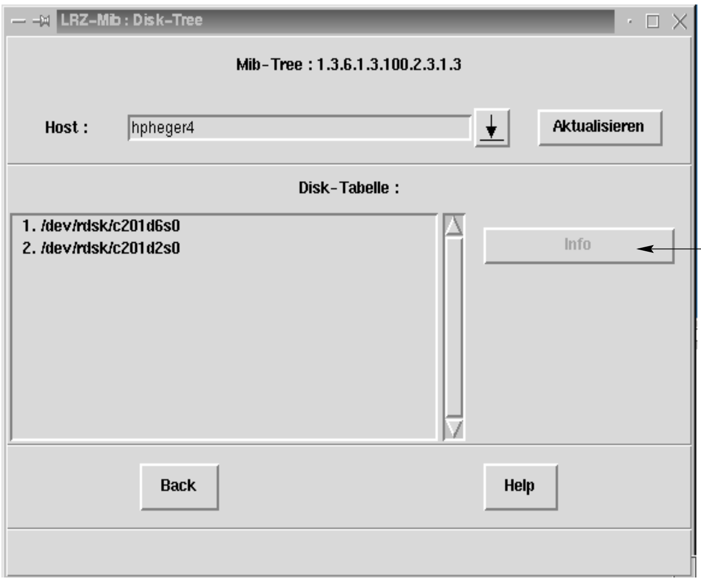
Im allgemeinen Teil ist das Huptmodul 'synema.tcl' und ein Modul 'util.tcl' zu finden. Das Hauptmodul initialisiert globale Variablen und stellt das Hauptmenü zu Verfügung. 'util.tcl' ist ein Modul, dass Hilfs-Funktionen zur Verfügung stellt, die z.B. Octet-Strings in lesbare (ASCII-)Strings umwandelt und ähnliches.

Alle anderen Module sind MIB-spezifisch und damit auch direkt von der Struktur der MNM-MIB abhängig. Die fünf Module 'system.tcl', 'storage,tcl', 'device.tcl', 'process.tcl' und 'user.tcl' reflektieren die jeweiligen Gruppen der MIB. Je nach Umfang und Komplexität der MIB-Gruppen sind diese Module in weitere Untermodule aufgeteilt, um die Grösse der Module in einem erträglichen Mass zu halten.

2.2.1 Grundsätzliches

Der Aufbau aller Module (mit Ausnahme von 'util.tcl') ist im Prinzip immer gleich:

Es gibt eine Funktion, die die jeweiligen Fensterstrukturen festlegt (der Name der Funktion beginnt normalerweise mit 'readShow') und eine Funktion, die das Fenster mit Variablenwerten füllt (deren Name normalerweise mit 'aktualisiere' beginnt). Zusätzlich gibt es noch einige Funktionen, deren Name mit 'Bind' aufhört. Diese binden Events, wie z.B. Mausklicks an verschiedene Aktionen. Dies soll durch Abbildung 15 verdeutlicht werden anhand des Laufwerks-Menüs.



diskMibInfoBind:
Info-Button Aktion festlegen

readShowDiskMib: Aufbau des Fensters (wo erscheint was im Fenster)

aktualisiereDiskMib: Werte im Fenster ausgeben und Fenster darstellen

diskMibOneClickBind: Mausklick Aktion festlegen

Abbildung 15

2.2.2 Das Hauptmodul 'synema.tcl'

Im Hauptmodul werden folgende Aufgaben wahrgenommen:

1. BLT Modul und Scotty Modul laden. Diese liegen unter /usr/local/mnm/tcl/lib.
2. Init Files laden. Dabei handelt es sich um init-Files fuer verschiedene Tcl Definitionen.
3. MIB File laden. Die Definitionen fuer die mnm-unix.mib werden an dieser Stelle geladen. Zuerst werden alle Pfade aus der Environment-Variable $MIBDIRS ausgelesen, danach im default-path von scotty (/usr/local/mnm/tcl/lib/tnm2.1.2/mibs) gesucht. Der erste Treffer wird geladen, eventuelle andere MIB Definitionenin späteren Pfaden ignoriert.
4. Globale Variablen festlegen (OID Anfänge, Community Strings etc.).
5. restliche Module laden.
6. Hauptmenü anzeigen und entspechend der Auswahl des Benutzers in andere Module verzweigen.

Folgende Programmzeilen sorgen für diese Aufgaben:

```
# ***************************************************
#
# tnm extension (scotty) laden   (KCF 07.08.96)
#
```

```
# ***************************************************
load /usr/local/mnm/tcl/lib/libBLT.sl
package require -exact Tnm 2.1.1

# ***************************************************
#
# init-Files laden      (KCF 14.08.96)
#
# ***************************************************
source /usr/local/mnm/tcl/lib/tcl7.5/init.tcl
source /usr/local/mnm/tcl/lib/tnm2.1.1/library/init.tcl

# ***************************************************
#
# MIB File laden
#
# ***************************************************
set mibdirs [array get env MIBDIRS]
set ind [string first ":"$mibdirs]
set success 0
if {$ind == -1} {
        set loadpath [string range $mibdirs 8 end]/mnm-unix.mib
} else {
        set loadpath [string range $mibdirs 8 [expr $ind-1]]/mnm-unix.mib
}
# Versuchen, das mib-file zu laden
if {[file exists $loadpath]} {
        puts Loading MIB definitions from $loadpath ..."
        mib load $loadpath
        set success 1
}
while {[string first ":"$mibdirs] > -1 && $success == 0} {
        set mibdirs [string range $mibdirs [expr $ind+1] end]
        set ind [string first ":"$mibdirs]
        if {$ind == -1} {
                set loadpath [string range $mibdirs 0 end]/mnm-unix.mib
        } else {
                set loadpath [string range $mibdirs 0 [expr $ind-1]]/mnm-unix.mib
        }
        # Versuchen, das mib-file zu laden
        if {[file exists $loadpath]} {
                puts Loading MIB definitions from $loadpath ..."
                mib load $loadpath
                set success 1
        }
}

if {$success == 0} {
        puts Loading MIB definitions from default path"
        mib load mnm-unix.mib
```

```
}

# *************************************************
#
# OIDs setzen
#
# *************************************************

global system_mib storage_mib device_mib process_mib user_mib lrz_mib

set lrz_mib "1.3.6.1.3.100.2"
set system_mib "$lrz_mib.1"
set storage_mib "$lrz_mib.2"
set device_mib "$lrz_mib.3"
set process_mib "$lrz_mib.4"
set user_mib "$lrz_mib.5"

# *************************************************
#
# Community-Strings setzen
#
# Set community muss private sein, geandert 21.08.96 KCF
#
# *************************************************
global get_community set_community

set get_community public
set set_community mnm3agens

# *************************************************
#
# Unterprogramme einbinden
#
# *************************************************

set path /users/stud/faerberk/Projects/SysMgmtTool/source/
global path

source $path./dragdrop.tcl
source $path./util.tcl
source $path./error_win.tcl
source $path./user.tcl
source $path./system.tcl
source $path./process.tcl
source $path./community.tcl
source $path./storage.tcl
source $path./device.tcl
source $path./help.tcl
source $path./selectionHost.tcl
source $path./readFile.tcl
```

2.3 Änderungen der Pfade und der Programmversionen

Bei Änderungen der Pfade, wie z.B. die im Lehrstuhl vorgesehene Änderung von /swhp/mnm nach /usr/local/mnm ist folgendes zu beachten:

- Bei Synema selbst sind nur die Pfade im Hauptmodul `synema.tcl` zu ändern (siehe vorheriges Kapitel).
- Die Programmpakete Tcl, Tk, Scotty und BLT sollten neu compiliert (mit –prefix=/usr/local/mnm) und unter den neuen Pfaden installiert werden.
- Wenn Environment-Variablen verwendet werden, sollten diese entsprechend geändert werden (also z.B. auf setenv TCL_LIBRARY /usr/local/mnm/tcl/lib/tcl7.5)

Sollten Programmpakete wie z.B. Tcl/Tk oder Scotty auf eine neue Version gebracht werden, gibt es einige Punkte zu beachten:

- Neue Scotty-Versionen sind oft nicht kompatibel zu älteren Releases. Hinweise dazu sind ueblicherweise in der Source-Distribution des Programmpakets enthalten. Deshalb sollte man immer den Nutzen durch neue Funktionalitäten dem Aufwand möglicher änderungen im SyNeMa-Source gegenueberstellen.
- Tcl/Tk Updates sind normalerweise abwärtskompatibel (Ausnahme war die änderung auf 7.5/4.1). Updates können deshalb ueblicherweise ohne tiefgreifende änderungen am SyNeMa-Source durchgefuehrt werden. Es muss nur im `synema.tcl` Hauptmodul das 'package require' Kommando auf die neue Version geändert werden.

3 Zusammenfassung und Ausblick

Zusammenfassend kann man aus den Erfahrungen im Umgang mit SyNeMa folgendes sagen:

SNMP eignet sich nur bedingt, um damit grössere Verwaltungs- und Administrationsaufgaben wahrzunehmen. Dies liegt zum einen an den mangelhaften Sicherheitsvorkehrungen (zumindest bei dem hier verwendeten SNMPv1) und zum anderen an der noch zu unflexiblen Handhabung, wie z.B. bei den ungewollten Timeouts bei der Druckerabfrage.

Auf der anderen Seite stellt die Verwendung von standardisierten Protokollen (SNMP und DPI) ein Fortschritt gegenüber proprietären Protokollen dar und ermöglicht so die Benutzung von verschiedenen Agenten, die man auch selber modifizieren kann, solange sie das Protokoll einhalten.

Die Aufteilung in Haupt- und Subagenten erlaubt weiterhin eine Teilung in Plattform- bzw. Hardware-Aghängige und -Unabhängige Agenten und erleichtert die Portierung auf verschiedene Systeme.

SyNeMa an sich hat den Vorteil, dass es Plattform- bzw. Hardware-Unabhängig ist und damit gute Vorausstzungen als Management-Programm hat. Andererseits ist es so stark von der MNM-MIB abhängig, dass eine modulare Verwendung mit anderen MIB's bzw. Agenten nicht möglich ist. Auch muss jede änderung der MIB im SyNeMa angepasst werden.

Durch die Verwendung zukünftiger Scotty Versionen ist es durchaus möglich, einige der angesprochenen Probleme zu beseitigen. So sollten z.B. Konvertierungsprobleme der Variablenformate damit besser zu bewältigen sein. Und durch die Verwendung von SNMPv2 können auch Sicherheitsmängel zumindest teilweise abgeschafft werden.

4 Literatur

[1] Ousterhout
[2] Tcl/Tk Dokumentation (Man-Pages)
[3] Scotty Dokumentation aus http://wwwhome.cs.utwente.nl/~schoenw/scotty/
[4] Pusch, B., Erweiterung eines SNMPv2-Agenten für Systemmanagementaufgaben, Technische Universität München, Dezember, 1994

5 Anhang

5.1 MNM-UNIX MIB

```
-- MNM-UNIX-MIB

-- Original-Autor: Uwe Krieger, krieger@informatik.tu-muenchen.de
-- Neubearbeitung: Stephan Haslbeck, haslbecs@nm.informatik.uni-muenchen.de

-- Change log:

-- haslbecs 961121: Unstimmigkeiten entfernt
-- haslbecs 961118: Luecken in Numerierung entfernt
-- haslbecs 961016: Bedeutung von 'diskState' geaendert
-- haslbecs 961014: einige Ungereimtheiten korrigiert
-- haslbecs 961007: ueberarbeitet, unnoetiges entfernt


-- *********************************************************


MNM-UNIX DEFINITIONS ::= BEGIN

   IMPORTS
        experimental
           FROM RFC1155-SMI
        OBJECT-TYPE
           FROM RFC-1212
        TRAP-TYPE
           FROM RFC-1215;


   -- some definitions for broken parsers (real parsers will complain)

   --    nullOID       OBJECT IDENTIFIER ::= { ccitt 0 }
   -- internet      OBJECT IDENTIFIER ::= { iso(1) org(3) dod(6) 1 }
   -- directory     OBJECT IDENTIFIER ::= { internet 1 }
   -- mgmt          OBJECT IDENTIFIER ::= { internet 2 }
   -- experimental  OBJECT IDENTIFIER ::= { internet 3 }
   -- private       OBJECT IDENTIFIER ::= { internet 4 }
   -- enterprises   OBJECT IDENTIFIER ::= { private 1 }


   -- some definitions taken over from SNMPv2

   TruthValue ::= INTEGER { true(1), false(2) }

   DateAndTime ::= OCTET STRING

   RowStatus ::=  INTEGER {
                    active(1),
                    notInService(2),
                    notReady(3),
                    createAndGo(4),
                    createAndWait(5),
                    destroy(6)
                 }

   -- mandatoryly the MNM-MIB is fixed under experimental.100 !
-- with registration the tree probably has to be moved.

lrz OBJECT IDENTIFIER ::= { experimental 100 }

-- Fora Bastian Pusch (puschb@informatik.tu-muenchen.de)
-- LRZ-LOG OBJECT IDENTIFIER ::= { lrz 1 }

mnm-unix OBJECT IDENTIFIER ::= { lrz 2 }


-- Groups in MNM-UNIX-MIB

system      OBJECT IDENTIFIER ::= { mnm-unix 1 }
storage     OBJECT IDENTIFIER ::= { mnm-unix 2 }
device      OBJECT IDENTIFIER ::= { mnm-unix 3 }
process     OBJECT IDENTIFIER ::= { mnm-unix 4 }
user        OBJECT IDENTIFIER ::= { mnm-unix 5 }


-- The UNIX-MNM System Group
-- Implementation of the System Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.

sysName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The system's fully qualified system and domain name,
           e.g. sunhegering5@informatik.tu-muenchen.de."
   ::={ system 1 }

sysContact OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The textual identification of the contact
           person for this managed system, together with
           information on how to contact this person."
   ::={ system 2 }

sysLocation OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The physical location of this node."
   ::={ system 3 }

sysOs OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The textual description of the operating
           system that the managed node is running."
```

```
   ::={ system 4 }

sysHardware OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The textual description of the managed
           node, e.g. SUN ELC."
   ::={ system 5 }

sysUptime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The time since last boot in seconds."
   ::={ system 6 }

sysDate OBJECT-TYPE
   SYNTAX DateAndTime
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The host's notation of local date and time."
   ::={ system 7 }

sysUsers OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The number of users logged on to the system."
   ::={ system 8 }

sysClockTicks OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The clock ticks per second."
   ::={ system 9 }


-- The UNIX-MNM Storage Group
-- Implementation of the Storage Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned

stoTypes OBJECT IDENTIFIER ::= { storage 1 }
stoRam            OBJECT IDENTIFIER ::= { stoTypes 1 }
stoVirtualMemory  OBJECT IDENTIFIER ::= { stoTypes 2 }


stoTable OBJECT-TYPE
    SYNTAX SEQUENCE OF StoEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
           "The table with logical storage areas on this host."
::= { storage 2 }

stoEntry OBJECT-TYPE
    SYNTAX StoEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
           "One entry for each logical storage area
           on the host."
    INDEX { stoIndex }
    ::= { stoTable 1 }

StoEntry ::= SEQUENCE {
    stoIndex            INTEGER,
    stoType             OBJECT IDENTIFIER,
    stoDescr            OCTET STRING,
    stoAllocationUnits  INTEGER,
    stoSize             INTEGER,
    stoUsed             INTEGER,
    stoState            INTEGER
}

stoIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The unique value of this instance."
    ::= { stoEntry 1 }

stoType OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The type of this instance."
    ::= { stoEntry 2 }

stoDescr OBJECT-TYPE
    SYNTAX OCTET STRING
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "A string describing the instance, e.g. virtual memory
            if stoType is StoTypes 3."
    ::= { stoEntry 3 }

stoAllocationUnits OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The size in bytes of one chunk of this instance."
    ::= { stoEntry 4 }

stoSize OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The size in kilobytes, as might been seen by the
            operating system."
   ::= { stoEntry 5 }
```

```
stoUsed OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The kilobytes of used storage of this instance.
            Returns -1 if not available."
    ::= { stoEntry 6 }

stoState OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
            "The percentage of used storage of this instance.
            Returns -1 if not available."
    ::= { stoEntry 7 }


-- The UNIX-MNM Device Group
-- Implementation of the Device Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.
-- some nice type definitions following

devTypes OBJECT IDENTIFIER ::= { device 1 }

processor OBJECT IDENTIFIER  ::= { devTypes 1 }
printer OBJECT IDENTIFIER    ::= { devTypes 2 }
disk OBJECT IDENTIFIER       ::= { devTypes 3 }
partition OBJECT IDENTIFIER  ::= { devTypes 4 }
filesystem OBJECT IDENTIFIER ::= { devTypes 5 }
network OBJECT IDENTIFIER    ::= { devTypes 6 }
video OBJECT IDENTIFIER      ::= { devTypes 7 }
audio OBJECT IDENTIFIER      ::= { devTypes 8 }
modem OBJECT IDENTIFIER      ::= { devTypes 9 }
pointing OBJECT IDENTIFIER   ::= { devTypes 10 }
tape OBJECT IDENTIFIER       ::= { devTypes 11 }
openprom OBJECT IDENTIFIER   ::= { devTypes 12 }
bus OBJECT IDENTIFIER        ::= { devTypes 13 }
controller OBJECT IDENTIFIER ::= { devTypes 14 }


deviceTable OBJECT-TYPE
   SYNTAX SEQUENCE OF DeviceEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
        "The table with all host's devices, if
        defined above."
   ::={ device 2 }

deviceEntry OBJECT-TYPE
   SYNTAX DeviceEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "One entry for each detected and defined device."
   INDEX { devIndex }
   ::={ deviceTable 1 }

DeviceEntry ::= SEQUENCE {
   devIndex  INTEGER,
   devType   OBJECT IDENTIFIER,
   devDescr  OCTET STRING,
   devStatus INTEGER
}

devIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "An unique value for each device."
   ::={ deviceEntry 1 }

devType OBJECT-TYPE
   SYNTAX OBJECT IDENTIFIER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The object identifier defined above for this instance.
           If the type is one of devProcessor, devPrinter or devDisk an
           correspondending entry exists to this device."
   ::={ deviceEntry 2 }

devDescr OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "A textual description of the device."
   ::={ deviceEntry 3 }

devStatus OBJECT-TYPE
   SYNTAX INTEGER {
              unknown(1),
              running(2),
              down(3),
              warning(4)
          }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "An unique value for each device."
   ::={ deviceEntry 4 }


-- The UNIX-MNM Processor Group
-- Implementation of the Processor Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.

cpuType OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "Type of CPU, eg. SPARC."
   ::={ processor 1 }

cpuClockRate OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
```

```
   STATUS mandatory
   DESCRIPTION
         "The cpu's clock rate."
   ::={ processor 2 }

cpuTable OBJECT-TYPE
   SYNTAX SEQUENCE OF CpuEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
         "A table with all processors in the host."
   ::={ processor 3 }

cpuEntry OBJECT-TYPE
   SYNTAX CpuEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
         "A (conceptual) entry for each processor
         of the host. Only one entry for a single
         processor machine."
   INDEX   { cpuNr }
   ::={ cpuTable 1 }

CpuEntry ::= SEQUENCE {
   cpuNr         INTEGER,
   cpuOpStat     INTEGER,
   cpuUsStat     INTEGER,
   cpuAdStat     INTEGER,
   cpuUserTime   INTEGER,
   cpuNiceTime   INTEGER,
   cpuSystemTime INTEGER,
   cpuIdleTime   INTEGER,
   cpuAction     INTEGER
}

cpuNr OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The internal number of the processor."
   ::={ cpuEntry 1 }

cpuOpStat OBJECT-TYPE
   SYNTAX INTEGER {
            enabled(1),
            disabled(2)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The operational status derived from ISO-10164-2.
          Processor enabled means that the processor is
          available for the user or applications. Disabled means that the
          processor is not available to the user or applications.
          Not to be implemented for single-processor-machines."
   ::={ cpuEntry 2 }

cpuUsStat OBJECT-TYPE
   SYNTAX INTEGER{
            idle(1),
```

```
            busy(2)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The processor's Usage State (ISO-10164-2). Idle means
          that no job is running on the processor. Busy means that
          a job is running on the processor. Not to be implemented
          for single-processor-machines."
   ::={ cpuEntry 3 }

cpuAdStat OBJECT-TYPE
   SYNTAX INTEGER {
            unlocked(1),
            locked(2),
            shuttingdown(3)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The processor's Administration State (ISO-10164-2).
          Unlocked means that usage is allowed, shuttingdown
          means that the usage for new users is not allowed
          and after the last user has is job finished the
          administrative state will be locked, what means that
          the processor is not available for any user."
   ::={ cpuEntry 4 }

cpuUserTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The percentage of time the processor was used
         in user mode (calculated over the last 0.5 seconds)."
   ::={ cpuEntry 5 }

cpuNiceTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The percentage of time the processor was used
         in user mode (calculated over the last 0.5 seconds)."
   ::={ cpuEntry 6 }

cpuSystemTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The percentage of time the processor was used
         in system mode (calculated over the last 0.5 seconds)."
   ::={ cpuEntry 7 }

cpuIdleTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The percentage of time the processeor was idle
         (calculated over the last 0.5 seconds)."
```

```
   ::={ cpuEntry 8 }

cpuAction OBJECT-TYPE
   SYNTAX INTEGER {
             enable(1),
             disable(2),
             lock(3),
             unlock(4),
             shuttingdown(5)
         }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The actions defined for systems management."
   ::={ cpuEntry 9 }


cpuSpecInt OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The Result of the SPECint benchmark."
   ::={ processor 4 }

cpuSpecIntYear OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The version of the benchmark,
          the year (SPECint92)."
   ::={ processor 5 }

cpuSpecFp OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The result of the SPECfp benchmark."
   ::={ processor 6 }

cpuSpecFpYear OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The version of the benchmark,
          the year (SPECfp92)."
   ::={ processor 7 }


-- The UNIX-MNM Printer Group
-- Implementation of the Printer Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.
-- Information about File Systems

printerTable OBJECT-TYPE
    SYNTAX SEQUENCE OF PrinterEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
           "The (conceptual) table of printers local to the
           host. A correspondent entry exists in the
           deviceTable with type devPrinter."
    ::= { printer 1 }

printerEntry OBJECT-TYPE
    SYNTAX PrinterEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
           "A (conceptual) entry for one printer local to the
           host.  The devIndex in the index represents
           the entry in the DeviceTable that corresponds to
           the PrinterEntry."
    INDEX { devIndex }
    ::= { printerTable 1 }

PrinterEntry ::= SEQUENCE {
    printerIndex     INTEGER,
    printerName      OCTET STRING,
    printerLocation  OCTET STRING,
    printerOpStatus  INTEGER,
    printerUsStatus  INTEGER,
    printerAdStatus  INTEGER,
    printerAvStatus  INTEGER,
    printerAction    INTEGER
}

printerIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "An unique value for each known printer."
   ::={ printerEntry 1 }

printerName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "An unique name for each known printer. It's a good
          idea, to choose a name containing manufacturer or/and
          the type of the printer."
   ::={ printerEntry 2 }

printerLocation OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "A String describing the pysical location of the
          printer. Perhaps you might wish to add also a
          person that is responsible to this printer."
   ::={ printerEntry 3 }

printerOpStatus OBJECT-TYPE
   SYNTAX INTEGER {
             enabled(1),
             disabled(2)
          }
```

```
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "Enabled means that the printer is available for the
          users. Disabled means that the printer is not avail-
          able for users."
   ::={ printerEntry 4 }

printerUsStatus OBJECT-TYPE
   SYNTAX INTEGER {
            idle(1),
            busy(2)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "Idle means that the printer is mandatorily not printing.
          Busy means that the printer is mandatorily printing. The
          number of queued requests does NOT influence the Usage
          State."
   ::={ printerEntry 5 }

printerAdStatus OBJECT-TYPE
   SYNTAX INTEGER {
            unlocked(1),
            locked(2),
            shuttingdown(6)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The printer's administrative state. Unlocked means that
          printing is allowed, shutting down means that new
          printing requests will be discarded and locked means that
          the printer is not available to any user."
   ::={ printerEntry 6 }

printerAvStatus OBJECT-TYPE
   SYNTAX INTEGER {
            inTest(1),
            failed(2),
            powerOff(3),
            offLine(4),
            notInstalled(5)
         }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The printer's availability status. inTest means that the
          ressource is undergoing a test procedure. The administrative
          state is locked. failed means that an internal failure has
          occurred, powerOff means that the printer is not powered on,
          offLine means that the the printer is not online,
          notIinstalled means that either the printer has no connection
          to the server and/or is physically gone."
   ::={ printerEntry 7 }

printerAction OBJECT-TYPE
   SYNTAX INTEGER {
            enable(1),
            disable(2),
            lock(3),
            unlock(4),
            shuttingdown(5),
            test(6),
            online(7)
         }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The actions defined for systems management. 1 to 5 are
          self explaining, test performs a testing routine and
          online sets the printer online."
   ::={ printerEntry 8 }


queueTable OBJECT-TYPE
   SYNTAX SEQUENCE OF QueueEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "A (conceptual) table of printer queues for this host."
   ::={ printer 2 }

queueEntry OBJECT-TYPE
   SYNTAX QueueEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "A (conceptual) entry for each printer queue installed
           at this host."
   INDEX { queueIndex }
   ::={ queueTable 1 }

QueueEntry ::= SEQUENCE {
   queueIndex           INTEGER,
   queueName            OCTET STRING,
   queueStatus          INTEGER,
   queuePrinterLocation INTEGER,
   queuePrinterIndex    INTEGER,
   queuePrinterName     OCTET STRING,
   queueAction          INTEGER
}

queueIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "An unique value for each queue."
   ::={ queueEntry 1 }

queueName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "An unique name for each queue."
   ::={ queueEntry 2 }

queueStatus OBJECT-TYPE
   SYNTAX INTEGER {
            enabled(1),
            disabled(2)
```

```
        }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The queue's status. Enabled means that users can
            place printjobs in the queue. Disables means that
            the queue is disabled."
   ::={ queueEntry 3 }

queuePrinterLocation OBJECT-TYPE
   SYNTAX INTEGER {
             local(1),
             remote(2)
        }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The location of the printer that corresponds to the
            queue."
   ::={ queueEntry 4 }

queuePrinterIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The pointer to the printer in the printer table the
            queue corresponds to. It is -1 if the queue corresponds
            to a remote printer."
   ::={ queueEntry 5 }

queuePrinterName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The whole name of the printer (hostname:printername) the
            queue corresponds to. The name is an empty string if the
            printer is local to the host."
   ::={ queueEntry 6 }

queueAction OBJECT-TYPE
   SYNTAX INTEGER {
            accept(1),
            reject(2)
        }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The actions defined for system management."
   ::={ queueEntry 7 }


queueJobTable OBJECT-TYPE
   SYNTAX SEQUENCE OF QueueJobEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           ""
   ::={ printer 3 }

queueJobEntry OBJECT-TYPE
   SYNTAX QueueJobEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "The list of jobs for this queue."
   INDEX { queueJobID }
   ::={ queueJobTable 1 }

QueueJobEntry ::= SEQUENCE {
   queueJobQueue   INTEGER,
   queueJobID      OCTET STRING,
   queueJobOwner   OCTET STRING,
   queueJobEntered DateAndTime,
   queueJobPrio    INTEGER,
   queueJobSize    INTEGER,
   queueJobPrinted INTEGER,
   queueJobAction  INTEGER
}

queueJobQueue OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The queue in which the job resides (index from table above)."
   ::={ queueJobEntry 1 }

queueJobID OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The ID of the job given by the system."
   ::={ queueJobEntry 2 }

queueJobOwner OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The job's owner."
   ::={ queueJobEntry 3 }

queueJobEntered OBJECT-TYPE
   SYNTAX DateAndTime
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The Date and Time the job entered the queue.
            May contain 'unknown'."
   ::={ queueJobEntry 4 }

queueJobPrio OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The priority of the job. The system manager
           is allowed to change the priority, writing a
           value to this field. If a system does not support
           priorities on print jobs nothing will happen."
   ::={ queueJobEntry 5 }
```

```
queueJobSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The size of the job in kilobytes."
   ::={ queueJobEntry 6 }

queueJobPrinted OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "If the job is mandatorily printed, how many
           kilobytes have been printed yet."
   ::={ queueJobEntry 7 }

queueJobAction OBJECT-TYPE
   SYNTAX INTEGER {
              cancel(1),
              hold(2),
              resume(3)
          }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "The actions defined for system management."
   ::={ queueJobEntry 8 }


-- The UNIX-MNM Disks Group
-- Implementation of the Disks Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.

diskTable OBJECT-TYPE
   SYNTAX SEQUENCE OF DiskEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "A (conceptual) table of hard disks contained
            by the host."
   ::={ disk 1 }

diskEntry OBJECT-TYPE
   SYNTAX DiskEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "An entry for one hard disk."
   INDEX { diskIndex }
   ::={ diskTable 1 }

DiskEntry ::= SEQUENCE {
   diskIndex       INTEGER,
   diskAccess      INTEGER,
   diskMedia       INTEGER,
   diskRemoveable  INTEGER,
   diskCapacity    INTEGER,
   diskLabel       OCTET STRING,
   diskBusy        INTEGER,
   diskReads       INTEGER,
   diskXfer        INTEGER,
   diskXferRate    INTEGER,
   diskWords       INTEGER
}

diskIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "An unique value for each drive."
   ::={ diskEntry 1 }

diskAccess OBJECT-TYPE
   SYNTAX INTEGER {
              read-write(1),
              read-only(2)
          }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The information about rights to write
           information to the disk."
   ::={ diskEntry 2 }

diskMedia OBJECT-TYPE
   SYNTAX INTEGER {
              unknown(1),
              hardDisk(2),
              opticalDisk(3),
              floppyDisk(4)
          }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The type of the disk."
   ::={ diskEntry 3 }

diskRemoveable OBJECT-TYPE
   SYNTAX INTEGER {
              removeable(1),
              not-removeable(2)
          }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "Indicates whether the drive is removable."
   ::={ diskEntry 4 }

diskCapacity OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The total size of this drive."
   ::={ diskEntry 5 }

diskLabel OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
```

```
   DESCRIPTION
           "Disk label is not only a nice name but it stores
           controller, geometry and partitions (but not always)."
   ::={ diskEntry 6 }

diskBusy OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The number of time ticks the disk was busy."
   ::={ diskEntry 7 }

diskReads OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The number of read operations on the disk."
   ::={ diskEntry 8 }

diskXfer OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The number of transfers on the disk."
   ::={ diskEntry 9 }

diskXferRate OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "Approximate transfer rate of the disk in kilobytes
           per second."
   ::={ diskEntry 10 }

diskWords OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The number of words transferred."
   ::={ diskEntry 11 }


-- The UNIX-MNM Partition Group
-- Implementation of the Partition Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.
-- Information about File Systems

partitionTable OBJECT-TYPE
   SYNTAX SEQUENCE OF PartitionEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "A (conceptual) table of all filesystems contained
            by the host."
   ::={ partition 1 }

partitionEntry OBJECT-TYPE
   SYNTAX PartitionEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "An entry for one partition."
   INDEX { partitionIndex }
   ::={ partitionTable 1 }

PartitionEntry ::= SEQUENCE {
   partitionIndex     INTEGER,
   partitionLabel     OCTET STRING,
   partitionID        OCTET STRING,
   partitionSize      INTEGER,
   partitionFSIndex   INTEGER,
   partitionDiskIndex INTEGER
}

partitionIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "An unique value for each partition in this table."
   ::={ partitionEntry 1 }

partitionLabel OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "A textual description of this partition.
            The mount point is used here, if one exists."
   ::={ partitionEntry 2 }

partitionID OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The descriptor representing this partition
           to the operating system. Usually, this is a
           filename in the /dev directory."
   ::={ partitionEntry 3 }

partitionSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The size of the partition. A value of -1 is
            returned, if no value is available."
   ::={ partitionEntry 4 }

partitionFSIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The pointer to the (predefined) filesystem table
            the partition belongs to. If there's no
            entry in the filesystem table for this
```

```
          partition, a value of -1 is returned."
   ::={ partitionEntry 5 }

partitionDiskIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The pointer to the disk in the diskTable the
          partition resides on."
   ::={ partitionEntry 6 }


-- The UNIX-MNM Filesystem Group
-- Implementation of the Filesystem Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.
-- Information about File Systems

fsTypes          OBJECT IDENTIFIER ::= { device 100 }
fsOther          OBJECT IDENTIFIER ::= { fsTypes 1 }
fsUnknown        OBJECT IDENTIFIER ::= { fsTypes 2 }
fsBerkeleyFFS    OBJECT IDENTIFIER ::= { fsTypes 3 }
fsSys5FS         OBJECT IDENTIFIER ::= { fsTypes 4 }
fsFat            OBJECT IDENTIFIER ::= { fsTypes 5 }  -- DOS
fsHPFS           OBJECT IDENTIFIER ::= { fsTypes 6 }  -- OS/2
fsMacHFS         OBJECT IDENTIFIER ::= { fsTypes 7 }  -- MAC
fsMFS            OBJECT IDENTIFIER ::= { fsTypes 8 }  -- MAC
fsVNode          OBJECT IDENTIFIER ::= { fsTypes 9 }
fsNTFS           OBJECT IDENTIFIER ::= { fsTypes 10 }
fsJournaled      OBJECT IDENTIFIER ::= { fsTypes 11 }
fsiso9660        OBJECT IDENTIFIER ::= { fsTypes 12 } -- CD
fsNFS            OBJECT IDENTIFIER ::= { fsTypes 13 }
fsNetWare        OBJECT IDENTIFIER ::= { fsTypes 14 }
fsAFS            OBJECT IDENTIFIER ::= { fsTypes 15 }
fsDFS            OBJECT IDENTIFIER ::= { fsTypes 16 } -- DCE (OSF)
fsRFS            OBJECT IDENTIFIER ::= { fsTypes 17 } --
fsBFS            OBJECT IDENTIFIER ::= { fsTypes 18 } -- SVR4 Boot FS
fsUFS            OBJECT IDENTIFIER ::= { fsTypes 19 }
fsHFS            OBJECT IDENTIFIER ::= { fsTypes 20 } -- HP HFS


predefinedFsTable OBJECT-TYPE
   SYNTAX SEQUENCE OF PredefinedFsEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "The predefined file system table (e.g /etc/fstab in
           SunOS 4.1.x)."
   ::= { filesystem 1 }

predefinedFsEntry OBJECT-TYPE
   SYNTAX PredefinedFsEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "A row in the predefined file system table."
   INDEX { predefinedFsIdentifier }
   ::= { predefinedFsTable 1 }

PredefinedFsEntry ::= SEQUENCE {
   predefinedFsIdentifier     INTEGER,
   predefinedFsName           OCTET STRING,
   predefinedFsType           OBJECT IDENTIFIER,
   predefinedFsMountPoint     OCTET STRING,
   predefinedFsMountOptions   OCTET STRING,
   predefinedFsAction         INTEGER
}

predefinedFsIdentifier OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The descriptor representing this partition to the
           operating system."
   ::= { predefinedFsEntry 1 }

predefinedFsName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The name of the predefined file system."
   ::= { predefinedFsEntry 2 }

predefinedFsType OBJECT-TYPE
   SYNTAX OBJECT IDENTIFIER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "One of the file system types defined above."
   ::= { predefinedFsEntry 3 }

predefinedFsMountPoint OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The mount point of the predefined file system."
   ::= { predefinedFsEntry 4 }

predefinedFsMountOptions  OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The file system specific mount options for this
           predefined file system."
   ::= { predefinedFsEntry 5 }

predefinedFsAction OBJECT-TYPE
   SYNTAX INTEGER {
             mount(1)
          }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
         "The actions defined by systems management. In this
          table you can only mount a filesystem."
   ::={ predefinedFsEntry 6 }

predefinedFsBackupTable OBJECT-TYPE
```

```
   SYNTAX SEQUENCE OF PredefinedFsBackupEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "The backup table for the predefined file systems."
   ::={ filesystem 2 }

predefinedFsBackupEntry OBJECT-TYPE
   SYNTAX PredefinedFsBackupEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "What is important to tell: in this table
           exist only 11 entries (backuplevel 0 down
           to backuplevel 10)."
   INDEX { predefinedFsBackupLevel }
   ::={ predefinedFsBackupTable 1 }

PredefinedFsBackupEntry ::= SEQUENCE {
   predefinedFsBackupFilesystem INTEGER,
   predefinedFsBackupLevel      INTEGER,
   predefinedFsBackupDate       DateAndTime,
   predefinedFsBackupAction     INTEGER
}

predefinedFsBackupFilesystem OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The index of the file system in the predefined file system
            table."
   ::={ predefinedFsBackupEntry 1 }

predefinedFsBackupLevel OBJECT-TYPE
   SYNTAX INTEGER (0..10)
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The Level of the Backup; the first entry in
          the table is numbered 0 and defines a full backup
   ::={ predefinedFsBackupEntry 2 }

predefinedFsBackupDate OBJECT-TYPE
   SYNTAX DateAndTime
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The Date of the Backup."
   ::={ predefinedFsBackupEntry 3 }

predefinedFsBackupAction OBJECT-TYPE
   SYNTAX INTEGER (0..10)
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The Level of backup wished to start.
          Level 0 is full backup."
   ::={ predefinedFsBackupEntry 4 }


mountedFsTable OBJECT-TYPE
   SYNTAX SEQUENCE OF MountedFsEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
     "The file system table."
   ::= { filesystem 3 }

mountedFsEntry OBJECT-TYPE
   SYNTAX MountedFsEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "A row in the file system table."
   INDEX { mountedFsName }
   ::= { mountedFsTable 1 }

MountedFsEntry ::= SEQUENCE {
   mountedFsName            OCTET STRING,
   mountedFsType            OBJECT IDENTIFIER,
   mountedFsMountPoint      OCTET STRING,
   mountedFsMountOptions    OCTET STRING,
   mountedFsBlockSize       INTEGER,
   mountedFsBlockCount      INTEGER,
   mountedFsBlocksFree      INTEGER,
   mountedFsBlocksAvailable INTEGER,
   mountedFsInodeCount      INTEGER,
   mountedFsInodesAvailable INTEGER,
   mountedFsQuota           INTEGER,
   mountedFsAction          INTEGER
}

mountedFsName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The name of the mounted file system."
   ::= { mountedFsEntry 1 }

mountedFsType OBJECT-TYPE
   SYNTAX OBJECT IDENTIFIER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "One of the file system types defined above."
   ::= { mountedFsEntry 2 }

mountedFsMountPoint OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The mount point of the mounted file system."
   ::= { mountedFsEntry 3 }

mountedFsMountOptions  OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The file system specific mount options."
   ::= { mountedFsEntry 4 }
```

```
mountedFsBlockSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The block size of this mounted file system in bytes."
   ::= { mountedFsEntry 5 }

mountedFsBlockCount OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "Total number of blocks of this mounted file-system."
   ::= { mountedFsEntry 6 }

mountedFsBlocksFree OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The number of blocks free in the mounted filesystem."
   ::= { mountedFsEntry 7 }

mountedFsBlocksAvailable OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The number of blocks free in the mounted file system
          to non-privileged users."
   ::= { mountedFsEntry 8 }

mountedFsInodeCount OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The total number of inodes on this mounted filesystem."
   ::= { mountedFsEntry 9 }

mountedFsInodesAvailable OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The number of inodes available."
   ::= { mountedFsEntry 10 }

mountedFsQuota OBJECT-TYPE
   SYNTAX INTEGER {
            quotaon(1),
            quotaoff(2)
         }
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
           "Writing the value 1 to this variable turns
            on quotas for the mounted file sytem; writing
            value 2 turns off quotas."
    ::={ mountedFsEntry 11 }

mountedFsAction OBJECT-TYPE
    SYNTAX INTEGER {
              unmount(1)
           }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
           "The actions defined by systems management. In this table
            you can only unmount a file system. If you want to mount
            a file system, you have to do it in the predefinedFsTable."
    ::={ mountedFsEntry 12 }


-- The UNIX-MNM Process Group
-- Implementation of the Process Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.

maxProcessNumber OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The maximum number of process contexts this
          system can support."
   ::={ process 1 }

maxProcessSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The maximum size of memory a process can allocate
          in kilobytes."
   ::={ process 2 }

curProcessNumber OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The number of loaded processes on this system."
  ::={ process 3 }

curMaxProcessSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The size of the text, data and stack segments of
          the largest process in kilobytes."
   ::={ process 4 }

curMaxProcessTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The cpu-time of the longest-running process in seconds
          (user and system time calculated)."
   ::={ process 5 }
```

```
processTable OBJECT-TYPE
   SYNTAX SEQUENCE OF ProcessEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "The table of processes loaded on this system."
   ::={ process 6 }

processEntry OBJECT-TYPE
   SYNTAX ProcessEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "An entry for each process loaded on this system."
   INDEX { processPID }
   ::={ processTable 1 }

ProcessEntry ::= SEQUENCE {
   processPID        INTEGER,
   processName       OCTET STRING,
   processPPID       INTEGER,
   processUID        INTEGER,
   processGID        INTEGER,
   processState      INTEGER,
   processFlagSSYS   TruthValue,
   processFlagSLOAD  TruthValue,
   processFlagSULOCK TruthValue,
   processFlagSSWAP  TruthValue,
   processFlagPHYSIO TruthValue,
   processCPUTime    INTEGER,
   processSize       INTEGER,
   processPri        INTEGER,
   processNice       INTEGER,
   processSignal     INTEGER
}

processPID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The unique process ID."
   ::={ processEntry 1 }

processName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "Command name used to invoke this process."
   ::={ processEntry 2 }

processPPID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The process ID of the process's parent."
   ::={ processEntry 3 }

processUID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The process owner's User ID."
   ::={ processEntry 4 }

processGID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The process owner's group ID."
   ::={ processEntry 5 }

processState OBJECT-TYPE
   SYNTAX INTEGER {
              sleeping(1),
              ready(2),
              running(3),
              idle(4),
              zombie(5),
              stopped(6)
          }
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The process`s state. sleeping means that the process
           is awaiting an event, ready says that the process
           is runnable, running says that the process is
           currently on the processor, idle is an intermediate state
           in process creation, zombie is an intermediate state in
           process termination and stopped says that the process
           is stopped."
   ::={ processEntry 6 }

processFlagSSYS OBJECT-TYPE
   SYNTAX TruthValue
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
             "True, if swapper or pager"
   ::= { processEntry 7 }

processFlagSLOAD OBJECT-TYPE
   SYNTAX TruthValue
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
             "True, if process is loaded in main memory."
   ::= { processEntry 8 }

processFlagSULOCK OBJECT-TYPE
   SYNTAX TruthValue
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
             "True, if the user requested that the process is
              not to be swapped."
   ::= { processEntry 9 }

processFlagSSWAP OBJECT-TYPE
```

```
   SYNTAX TruthValue
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "True means that the process is being swapped."
   ::= { processEntry 10 }

processFlagPHYSIO OBJECT-TYPE
   SYNTAX TruthValue
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "True means that the process is doing physical IO."
   ::= { processEntry 11 }


processCPUTime OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The sum of user and system time in seconds
          the process has consumed."
   ::={ processEntry 12 }

processSize OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The amount of memory allocated to the process,
          including text, data and stack segment, in kilobytes."
   ::={ processEntry 13 }

processPri OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The priority of this process."
   ::={ processEntry 14 }

processNice OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "Setting this value will cause a renice of
          the process to this value."
   ::={ processEntry 15 }

processSignal OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "Post the value of this column to the process as signal,
          after posting the value is 0 again."
   DEFVAL { 0 }
   ::={ processEntry 16 }
```

```
loadAvg1 OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The average load during the last minute on this
           system. The load is defined as number of processes,
           that are ready for running, multiplicated with 100."
   ::={ process 7 }

loadAvg5 OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
           "The average load during the last 5 minutes on this
           system. The load is defined as number of processes,
           that are ready for running, multiplicated with 100."
   ::={ process 8 }

loadAvg15 OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The average load during the last 15 minutes on this
          system. The load is defined as number of processes,
          that are ready for running, multiplicated with 100."
   ::={ process 9 }

-- The UNIX-MNM User Group
-- (containing information about user accounts, about
-- group accounts and about logged users.)
-- Implementation of the User Group is mandatory for all
-- systems. If an agent is not configured to have a value for
-- any of these variables, a string of length 0 is returned.

userTable OBJECT-TYPE
   SYNTAX SEQUENCE OF UserEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "The user table."
   ::={ user 1 }

userEntry OBJECT-TYPE
   SYNTAX UserEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "An entry in the userTable."
   INDEX  {  userLogin }
   ::={ userTable 1 }

UserEntry ::= SEQUENCE {
   userLogin      OCTET STRING,
   userPasswd     OCTET STRING,
   userID         INTEGER,
   userGroupID    INTEGER,
   userComment    OCTET STRING,
   userHome       OCTET STRING,
```

```
   userShell      OCTET STRING,
   userStatus     RowStatus
}

userLogin OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "This field contains the user's login name."
   ::={ userEntry 1 }

userPasswd OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field contains the user‘s encrypted password."
   ::={ userEntry 2 }

userID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field contains a user identification number (UID)
          that identifies the account by number to the operating
          system."
   ::={ userEntry 3 }

userGroupID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
         "This field contains a group identification number (GID)
         that identifies the user‘s primary group."
   ::={ userEntry 4 }

userComment OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS deprecated
   DESCRIPTION
          "This field was originally used to hold the login information
          needed to submit batch jobs to a mainframe running GECOS from
          Bell Labs. This field is still part of the file passwd in most
          UNIX-derivats. Therefore, if no GECOS mainframe is available,
          just record personal info for each user."
   ::={ userEntry 5 }

userHome OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field identifies the user‘s home directory.
          Editing this field does NOT create the home directory!"
   ::={ userEntry 6 }

userShell OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field identifies the login shell to initially
          provide for the user when he logs in."
   DEFVAL { "/bin/sh" }
   ::={ userEntry 7 }

userStatus OBJECT-TYPE
   SYNTAX RowStatus
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The status column used for creating, modifying,
          and deleting user-accounts on the system. This object
          may not be set to 'active' unil the user's home directory
          is created and valid values for userID, userLogin and userGID
          and userHome are given."
   ::={ userEntry 8 }

userQuotaTable OBJECT-TYPE
   SYNTAX SEQUENCE OF UserQuotaEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
   ""
   ::={ user 2 }

userQuotaEntry OBJECT-TYPE
   SYNTAX UserQuotaEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
           "A table of all the filesystems and the quotas
           the users have in the filesystems."
   INDEX { userQuotaUserLogin, userQuotaFsName }
   ::={ userQuotaTable 1 }

UserQuotaEntry ::= SEQUENCE {
   userQuotaUserLogin OCTET STRING,
   userQuotaFsName    OCTET STRING,
   userQuotaSoft      INTEGER,
   userQuotaHard      INTEGER,
   userQuotaUsed      INTEGER,
   userInodeSoft      INTEGER,
   userInodeHard      INTEGER,
   userInodeUsed      INTEGER
}

userQuotaUserLogin OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The login name of the user affected by the quotas."
   ::={ userQuotaEntry 1 }

userQuotaFsName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
```

```
   DESCRIPTION
          "The name of the filesystem specified by userQuotaFsIndex."
   ::={ userQuotaEntry 2 }

userQuotaSoft OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field provides the soft quota in kilobytes specified
           for the user. A value of 0 means that no quota is
           activated for this user."
   DEFVAL { 0 }
   ::={ userQuotaEntry 3 }

userQuotaHard OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "This field provides the hard quota in kilobytes specified
           for the user. A value of 0 means that no quota is
           activated for this user."
   DEFVAL { 0 }
   ::={ userQuotaEntry 4 }

userQuotaUsed OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "This value shows the user's used disk space in kilobytes.
           Logging this value might be useful to detect reasons for
           ''disk full'' warnings or to detect strange user behavior."
   ::={ userQuotaEntry 5 }

userInodeSoft OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field provides the inode Soft-Quota specified for the
          user. A value of 0 means that no quota is activated for this
          user."
   DEFVAL { 0 }
   ::={ userQuotaEntry 6 }

userInodeHard OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "This field provides the inode Hard-Quota specified for the
          user. A value of 0 means that no quota is activated for this
          user."
   DEFVAL { 0 }
   ::={ userQuotaEntry 7 }

userInodeUsed OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
```

```
   DESCRIPTION
          "This value shows the user's used inodes. Logging
          this value might be useful to detect reasons for ''disk full''
          warnings or to detect strange user behavior."
   ::={ userQuotaEntry 8 }

groupTable OBJECT-TYPE
   SYNTAX SEQUENCE OF GroupEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "The group table (usually from /etc/group)."
   ::= { user 3 }

groupEntry OBJECT-TYPE
   SYNTAX GroupEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "An entry in the group table."
   INDEX   { groupName }
   ::= { groupTable 1 }

GroupEntry ::= SEQUENCE {
   groupName    OCTET STRING,
   groupPasswd  OCTET STRING,
   groupID      INTEGER,
   groupStatus  RowStatus
}

groupName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
     DESCRIPTION
            "The name of this group."
     ::= { groupEntry 1 }

groupPasswd OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-write
   STATUS deprecated
   DESCRIPTION
          "The encrypted password for this group. In
          most UNIX-Versions it is not used anymore."
   ::= { groupEntry 2 }

groupID OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The group's ID in the system. It has to
          be unique."
   ::= { groupEntry 3 }

groupStatus OBJECT-TYPE
   SYNTAX RowStatus
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
```

```
          "The status of this group. This object
          may not be set to 'active' until a valid groupID
          and a valid name are given."
   ::= { groupEntry 5 }


groupUsersTable OBJECT-TYPE
   SYNTAX SEQUENCE OF GroupUsersEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "The list of users belonging to a group."
   ::={ user 4 }

groupUsersEntry OBJECT-TYPE
   SYNTAX GroupUsersEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          ""
   INDEX { groupUsersGroup }
   ::={ groupUsersTable 1 }

GroupUsersEntry ::= SEQUENCE {
   groupUsersGroup  OCTET STRING,
   groupUsersName   OCTET STRING,
   groupUsersStatus RowStatus
}

groupUsersGroup OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The name of the group."
   ::= { groupUsersEntry 1 }

groupUsersName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The user's login."
   ::= { groupUsersEntry 2 }

groupUsersStatus OBJECT-TYPE
   SYNTAX RowStatus
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
          "The status of this group-member. This object
           cannot be set to NotInService. GroupUsersEntries
           are created by the CreateAndGo-Command."
   ::= { groupUsersEntry 3 }


whoTable OBJECT-TYPE
   SYNTAX SEQUENCE OF WhoEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "This table represents who is currently logged on
          to the node on which the agent resides."
   ::= { user 5 }

whoEntry OBJECT-TYPE
   SYNTAX WhoEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION
          "An entry in the who table."
   INDEX { whoIndex }
   ::= { whoTable 1 }

WhoEntry ::= SEQUENCE {
   whoIndex  INTEGER,
   whoName   OCTET STRING,
   whoDevice OCTET STRING,
   whoWhere  OCTET STRING,
   whoTime   DateAndTime
}

whoIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
        "The index of the whoEntry object."
   ::= { whoEntry 1 }

whoName OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The name of a user logged into this system."
   ::= { whoEntry 2 }

whoDevice OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The device the user is using to log into
          this system."
   ::= { whoEntry 3 }

whoWhere OBJECT-TYPE
   SYNTAX OCTET STRING
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
          "The host from which the user is logged into
          this system. '':0.0'' instead of a hostname
          means that the user is using the terminal to log in
          on the system."
   ::= { whoEntry 4 }

whoTime OBJECT-TYPE
   SYNTAX DateAndTime
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
         "The time when the user first logged in."
```

```
   ::= { whoEntry 5 }

-------------------------------
-------------------------------
-- Trap Definitions (commented out for broken parsers)

-- mountedFsLowSpace TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { mountedFsName }
--    DESCRIPTION
--          "The mountedFsLowSpace trap signifies that the number of free
--           blocks in the file system fell below a configurable amount
--           (Upper bound of the hysteresis)."
--    ::= 1

-- mountedFsMoreSpace TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { mountedFsName }
--    DESCRIPTION
--           "The mountedFsMoreSpace trap signifies that the number of free
--            blocks in the file system regained a configurable amount after
--            falling below the number causing the mountedFsLowSpace trap
--            (Lower bound of the hysteresis)."
--    ::= 2

-- curProcessNumberHigh TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { curProcessNumber }
--    DESCRIPTION
--           "The curProcessNumberHigh trap signifies that there are
--            abnormal many loaded processes - (number configurable in
--            percentage of maximal process number)."
--    ::= 3

-- curProcessNumberLower TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { curProcessNumber }
--    DESCRIPTION
--           "The curProcessNumberLower trap signifies that the number of
--            loaded process fell below a configurable bound after reaching
--            the number causing a curProcessNumberHigh trap."
--    ::= 4

-- loadAvgHigh TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { loadAvg1, loadAvg5, loadAvg15 }
--    DESCRIPTION
--           "The loadAvgHigh trap signifies that the load average is very
--            high (values configurable)."
--    ::= 5

-- loadAvgLower TRAP-TYPE
--    ENTERPRISE lrz
--    VARIABLES  { loadAvg1, loadAvg5, loadAvg15 }
--    DESCRIPTION
--           "The loadAvgLower trap signifies that the load average fell
--            below a configurable value."
--    ::= 6

END

-- -----------------------------------------------------
-- *****************************************************
-- -----------------------------------------------------
```