

## 4 Startvorgang und Anpassung von SAP

In diesem Kapitel beschäftigen wir uns mit dem Startvorgang eines SAP-Systems. Wie jedes System unterliegt auch SAP R/3 ständigen Änderungen. Des öfteren im Jahr werden Patches veröffentlicht, die bekannte Fehler beheben und Sicherheitslücken schließen, oftmals kommen neue Funktionen dazu. Diese Fehler werden vom Basisadministratoren behoben, der sich sehr gut in den grundlegenden Einstellungen und Möglichkeiten des SAP Systems auskennen sollte.

Um überhaupt abschätzen zu können, ob ein Patch aus dem OSS-System geholt werden muß, ist es wichtig, daß der Basisadministrator über den Ablauf des Startvorgangs Bescheid weiß und grundlegenden Einstellungen beherrscht. Er muß auch Fehlermeldungen analysieren und entsprechende Maßnahmen ergreifen können. Wir schauen uns deshalb zuerst den Startvorgang des R/3-Systems an.

Die erste Information, die beim Start geschrieben wird, ist eine Logdatei. Sie analysiert der Systemverwalter, wenn es Probleme beim Starten oder Stoppen des SAP-Systems gibt. Je nach SAP-System gibt es die Logdateien, die »startsap\_linux\_17.log« und »stopsap\_linux\_17.log«, oder vielleicht »startsap\_linux\_18.log« und »stopsap\_linux\_18.log« heißen. Hier sind alle Meldungen des Systems gesammelt, die anzeigen, daß bestimmte Dienste des SAP-Systems gestartet werden konnten. Für die Datenbank befindet sich im Verzeichnis »/usr/sap/LNX/adm« eine Datei »startdblog«, die bei jedem Start der Datenbank neu geschrieben wird und Informationen über den Start enthält.

Ist R/3 gestartet und am Laufen, wird das System überwacht und Fehlerhinweise und Ereignisse werden im Systemprotokoll dokumentiert. Wer ein SAP-System starten möchte, muß sich mit dem Benutzer *lnxadm* am KDE-Interface anmelden. Im Homeverzeichnis dieses Benutzers ist das Shellskript *startsap\_lnx\_17* abgelegt. Wer Lust hat, kann sich dieses Shellskript mit einem Texteditor anschauen. Die wichtigsten Dienste, die gestartet werden, sind *saposcol*, die Datenbank und das R/3-System. Das Programm *saposcol* wird in einem Betriebssystem nur einmal gestartet, genauso wie der *vserver*.

Durch den Aufruf der Datei *startsap* werden nacheinander die Datenbank, dann der Messageserver und zum Schluß die Zentralinstanz mit dem Dispatcher hochgefahren. Selbstverständlich werden gestartete Dienste erkannt und mit einer Meldung quittiert. Im Skript sind Parameter hinterlegt, die den Start bestimmter Dienste ermöglichen. Für unser System, das auf nur einem Rechner läuft, ist dies nicht weiter interessant, wer jedoch ein verteiltes System aufgebaut hat, bei dem es mehrere Messageserver gibt, die ein sogenanntes Loadbalancing ermöglichen, hat hier die Möglichkeit, mit *startsap\_lnx\_17* für R3 eine weitere Instanz zu starten.

Mit dem Parameter *check* werden die Datenbank und die SAP-Instanz geprüft und der Parameter *all* fährt die Datenbank und die Instanz hoch.

Das Programm *saposcol* muß zuerst aufgerufen werden, da es die Aufgabe hat, Informationen über das Betriebssystem zu sammeln, um Performan- cewerte zu ermitteln. Diese Informationen werden aus SAP heraus ange- schaut und bewertet. Im SAP wird das Programm *RSCOLL00* intern zum Sammeln der Performancedaten aufgerufen. Ein weiteres Programm, das mit diesem Skript aufgerufen wird, ist das Programm *R3trans* zum Star- ten der SAP-Datenbank. Beim Start des R/3-Systems werden sofort alle Schritte mitprotokolliert, die ablaufen. Im Fehlerfall kann geprüft wer- den, was passiert ist und woran es hapert, wenn der Start mißglückte. Wird der Befehl *startsap\_linux\_17* ausgeführt, werden alle Onlinemeldun- gen in der Datei »startsap\_linux\_17.log« festgehalten, so daß immer noch einmal nachgesehen werden kann, warum ein System nicht starten konn- te. Da die Datenbank zuerst aktiv sein muß, wird die Meldung auf der Konsole und im Log in der Art geschrieben:

```
Starting SAP R/3 LNX Database
Startup-Log is written to /usr/sap/LNX/adm/startdb.log
```

## 4.1 Die Logdateien

Um interessante Meldungen im Logfile zu provoizieren, habe ich den Da- tenbankaufruf testweise mit einem falschen Parameter belegt. Das Log schrieb dann folgende Meldungen aus, wenn das Skript *startsap\_linux\_17* ausgeführt wird:

```
----- Sam Dez 15
09:34:48 CET 2001 LOGFILE FOR STARTING SAP DB
----- Sam Dez 15
09:34:48 CET 2001 checking required environment variables
DBNAME is >LNX<
DBROOT is >/usr/sap/LNX/db<
----- Sam Dez 15 09:34:48 CET 2001 starting vserver
INFO 10004: Vserver started.
----- Sam Dez 15
09:34:48 CET 2001 Connect to the database to check the database state:
R3trans: connect check finished with return code: 12 Database not
available
----- Sam Dez 15
09:34:48 CET 2001 starting database
getparam: cannot open param file for 'LNX @?'"??:

@' Run-directory (null) is missing
Please create it and start again
```

```
Restart Database ...
*** ERROR -8888: SERVERDB NOT ACCESSIBLE,
database not running SET MONITOR OFF
Database LNX not started : -32768 database
not running SET MONITOR ON
Database LNX not started : -32768 database
not running CALL DBPARAMS
Database LNX not started : -32768 database
not running
----- Sam Dez 15
09:34:51 CET 2001 Restart Database ...
*** ERROR -8888: SERVERDB NOT ACCESSIBLE,
database not running SET MONITOR OFF
Database LNX not started : -32768 database
not running SET MONITOR ON
Database LNX not started : -32768 database
not running CALL DBPARAMS
Database LNX not started : -32768 database
not running
----- Sam Dez 15
09:34:51 CET 2001 Connect to the database
to verify, that the database is now open
R3trans check finished with return code: 12

*** ERROR: Restart of database failed
        Notify Database Administrator.
----- Sam Dez 15
09:34:51 CET 2001
/usr/sap/LNX/SYS/exe/run/startdb: Terminating
with error code 2
```

Durch das Skript wird versucht, auf den Fehlerfall angemessen zu reagieren. Es wird bereits eine Fehlermeldung mit einem Lösungsvorschlag generiert und ein automatischer Neustart der Datenbank versucht, der natürlich ebenfalls fehlschlägt. Schließlich wird dem Administrator noch mitgeteilt, an wen er sich wenden soll, nämlich an den Datenbankadministrator, und es wird noch einmal ein Fehlercode ausgegeben, der das weitere Einkreisen des Fehlers zulässt.

Die Fehlercodes haben folgende Bedeutung:

0	Alles ist in Ordnung.
2	Der Aufruf erfolgte mit falschen oder fehlenden Parametern.
3	Der User ist für diese Aktion überhaupt nicht berechtigt.
4	Eine Environment-Variable ist nicht gesetzt.
5	Der Datenbankstart schlug fehl.
6	Die Datenbank auf dem entfernten Datenbank-Server läuft nicht.
7	Der Start der Instanz schlug fehl.
8	Das Startprofil des SAP-Systems existiert nicht. Die drei SAP-Startprofile werden in einem eigenem Kapitel behandelt.
9	Der User ist überhaupt nicht berechtigt, die Datenbank zu starten.
12	Dies deutet auf ein Datenbankproblem hin oder auf Dateien mit falsch gesetzten Berechtigungen.

Wer ein bißchen spielen möchte, kann ja das Programm *R3trans* manuell aufrufen und sich die Ergebnisse anschauen. Das Programm wird eigentlich aus dem Skript heraus aufgerufen und erzeugt dann die oben dokumentierten Fehlernummern. Wird es manuell aufgerufen, verhält es sich ähnlich. Es ist bei der LNX-Demo im Verzeichnis `»usr/sap/LNX/SYS/exe/run/«` abgelegt und heißt *R3trans*. Als Parameter kann zum Beispiel *R3trans -d -w* angegeben werden. Wir erhalten bei dem oben genannten Fehler die folgende Ausschrift: `»Database is not available via /usr/sap/LNX /SYS/exe/run/R3trans -d -w«`.

Es gibt noch weitere Parameter, die ich hier nur anzeigen, aber nicht näher erläutern möchte, da sie normalerweise nicht benötigt werden. *R3trans* wird eigentlich nur aufgerufen, wenn sehr große Tabellen oder Reportvarianten von einem Client in einen anderen kopiert werden sollen. Ansonsten wird *R3trans* immer durch andere Tools aufgerufen, wie hier durch das Startskript oder durch das Tool *tp*, um Transporte zwischen SAP-Systemen durchzuführen.

```
usage: /usr/sap/LNX/SYS/exe/run/R3trans
[<options>] <control_file> The control_file describes what R3trans has to
do. The following options are possible:
-c f1 f2 : Copy file f1 to f2 with character set conversion.
-d       : DB connect. Test if SAP database is available.
-i file  : Import from file without using a control file.
-l file  : List the contents of file to the log file.
-m file  : List the contents of file to allow tp to create a cofile.
-t       : Test. All database changes are rolled back.
-u <int> : Unconditional modes.
- v -    : Verbose. Write more details to the log file.
- w file : Log file. The default log file is 'trans.log'.
- x      : DB connect without access on any SAP table.
```

Diese Liste mit möglichen Parametern wird als Hilfsdatei ausgegeben. Wirklich interessant finde ich den Parameter *-u*, der Modusangaben in Zahlenform annimmt. Wer also jemals auf eine Angabe *-u <Zahl>* stößt, braucht zumindest die Information, was diese Zahlenangabe bedeutet:

1	Mit der Angabe 1 wird der Änderungsstatus ignoriert, so daß bereits importierte Transportaufträge wiederholt werden können.
2	Erlaubt es; Originale zu überschreiben.
3	Erlaubt sogar das Überschreiben systemabhängiger Objekte.
6	Ersetzt bereits reparierte Objekte.
8	Bei dem Wert 8 wird der Zugriff auf Tabellen erlaubt, die normalerweise für <i>R3trans</i> gesperrt sind.

Ein Aufruf dieses Tools hat beispielsweise folgendes Aussehen:

```
Linux:lnxadm 54> /usr/sap/LNX/SYS/exe/run/R3trans -d -w
```

#### 4.1.1 Die Kontrolldatei des Programms R3trans

Eine Kontrolldatei würde die Daten steuern, die hier übertragen werden sollen. Als Kontrolldatei könnte also folgendes in einer Textdatei angegeben werden:

```
r3.control file:
source client=000
target client=150
select * from VARI
```

Der Befehlsaufruf schaut folgendermaßen aus:

```
R3trans -w ./r3trans.log ./r3control file
```

Das obige theoretische Beispiel überträgt die Varianten aus der Tabelle *VARI* von Mandant 000 in den Mandanten 150. Mit diesem Beispiel möchte ich den Aufruf verdeutlichen, muß aber explizit darauf hinweisen, daß diese Art der Datenübertragung zu Inkonsistenzen der SAP-Datenbank führen kann! Um den oben erzeugten Fehler einzukreisen, ist zuerst zu prüfen, ob die Datenbank überhaupt startbar ist. Dazu ruft man das Programm */usr/sap/LNX/SYS/exe/run/startdb* auf.

Als User habe ich erst einmal *lnxadm* genommen. Für Datenbanken gibt es aber auch den User *sqlnx*. Also meldet man sich noch einmal mit dieser Userkennung an und startet die Datenbank. Das funktioniert. Das Programm *saposcol* übrigens kann aus dem Verzeichnis *»/usr/sap/LNX/SYS/exe/run«* unter der Userkennung *root* gestartet werden, um die Warnung *»Effective userid not root! Expect problems!«* zu umgehen. In diesem Fall müssen wir natürlich entweder das Skript anpassen, so daß das Programm *saposcol* nicht doppelt aufgerufen wird, oder wir bekommen die Meldung, daß *saposcol* bereits läuft. In diesem Fall ignorieren wir diese Fehlermeldung. Schließlich wird solch eine Aktion ja nur zu Testzwecken durchgeführt!

Nun meldet man sich bei Verwendung der LNX-SAP-Demo 4.5 als User *lnxadm* an und führt *startsap\_linux\_17* aus. SAP merkt so, welche Dienste schon gestartet sind und schreibt entsprechende Meldungen aus:

```
-----
11:26:16 16.12.2001 WARNING: Effective
userid not root! Expect problems !
Collector already running ... don't launch
saposcol already running
Checking SAP R/3 LNX Database
-----
Database is running
Starting SAP R/3 Instance -----
Startup-Log is written to
/usr/sap/LNX/adm/startsap_linux_17.log
Instance on host linux started
```

Die Instanz läuft erst einmal wieder. Zwar wird weiterhin erkannt, daß dieses Programm *saposcol* nicht vom Benutzer *root* aufgerufen wurde, doch wir haben dieses Programm ja bereits als Benutzer *root* manuell gestartet. Natürlich kann man sich von vornherein mit dem User *lnxadm* anmelden und auf der Konsole den Befehl *su -l sqlnx* eingeben. Nun wer-

den für diese eine Konsole die Einstellungen des Benutzers *sqdlnx* genommen. Hier startet man die Datenbank, verläßt das Konsolenfenster mit *exit* und startet das SAP-System. So spart man sich die Anmeldung am KDE für den Datenbank-User.

Zurück zu den Logdateien. Wird zum Beispiel der Server zweimal gestartet, ist dies auch im Log »startdb.log« ersichtlich:

```
ERROR 10107: Socket address already in use,
probably a vserver is running!
*** ERROR -8888: SERVERDB NOT ACCESSIBLE,
database not running
```

Wird das SAP-GUI gestartet, ohne daß eine Verbindung zur Datenbank besteht, wird folgende Fehlermeldung auf der Konsole ausgeschrieben, es wird aber auch ein Fenster auf dem KDE-Desktop mit diesem Fehlerhinweis eingeblendet:

```
15.12. 10:01:22.63 ERROR:
GuiConnection: cannot connect to appserver:
Error: partner not reached (host linux, service 3217)
```

```
Sat Dec 15 10:01:22 2001 Release 46B
Component NI (network interface),
version 34 rc = -10, module niuxi.c,
line 858 Detail NiPConnect
System Call connect Error No 111
'Verbindungsaufbau abgelehnt'
```

Wie bereits erwähnt, werden alle Vorgänge protokolliert, natürlich auch die Beendigung des Datenbankservers. Hier wird die Datei »stopsap\_linux\_17.log« geschrieben, die die Meldungen enthält, die auch auf der Konsole stehen:

```
Trace of system startup/check of R/3 System LNX on Sat Dez 15 10:03:34 CET
2001
Called command: /usr/sap/LNX/adm/stopsap_linux_17
Stopping the SAP R/3 LNX Processes -----
```

```
Stopping SAP R/3 LNX Database
Shutdown-Log is written to /usr/sap/LNX/adm/stopdb.log
Database stopped
```

In der Datei »/usr/sap/LNX/adm/stopdb.log« steht dann folgender Eintrag:

```
linux:lnxadm 67> more stopdb.log
```

```
Sam Dez 15 10:03:34 CET 2001 LOGFILE FOR STOPPING SAP DB
Sam Dez 15 10:03:34 CET 2001 checking required environment variables
DBNAME is >LNX<
DBROOT is >/usr/sap/LNX/db<
Sam Dez 15 10:03:34 CET 2001 Connect to the database to check the database
state:
```

```
R3trans check finished with return code: 12
There is no database connect possible
Database is probably already stopped.
Javacore
```

#### 4.1.2 Log beim Aufruf der grafischen Oberfläche

Es kann vorkommen, daß beim Starten des grafischen Interfaces ein Dump auf die Konsole geschrieben wird. Diese Informationen braucht man zur Analyse des Fehlers. Alle diese wichtigen Informationen werden auch in eine Textdatei des aktuellen Verzeichnisses unter dem Namen »javacore<nummer>.txt« gesichert und können mit jeder Textverarbeitung ausgelesen werden.

In diesem Dump stehen unter anderem das Datum und Uhrzeit des Abbruchs und die Versionsnummer des verwendeten Java Developer Kits.

Weitere wichtige Informationen sind der Name des Hostrechners, die Version der Bibliothek *glibc* und Informationen über den Speicherverbrauch. Im folgenden nur kurze Ausschnitte, ein solcher Dump ist natürlich viel größer:

```
Mon Dec 17 15:59:21 2001
```

```
SIGSEGV received at be9ffa24 in /opt/PlatinGUI/bin/./lib/libJPlatin.so.
Processing terminated
jre full version "JDK 1.1.8 IBM build I118-19991221 (JIT enabled: jitc)"
Operating Environment
```

```
-----
Host           : linux.
OS Level       : 2.2.16.#1 Wed Aug 2 20:22:26 GMT 2000
glibc Version  : 2.1.3
No. of Procs   : 1
Memory Info:
    total:    used:    free:  shared: buffers:  cached:
Mem:  267948032 163811328 104136704      0 12087296 31264768
Swap: 133885952 115126272 18759680
MemTotal:    261668 kB
MemFree:     101696 kB
```

```
MemShared:      0 kB
Buffers:        11804 kB
Cached:         30532 kB
BigTotal:       0 kB
BigFree:        0 kB
SwapTotal:     130748 kB
SwapFree:      18320 kB
```

Die Umgebungsvariablen befinden sich etwas weiter unten:

```
Environment Variables -
    PWD=/usr/sap/LNX/adm
    DBROOT=/usr/sap/LNX/db
    PAGER=/usr/bin/less -sM
    VENDOR=suse
    HOSTNAME=linux
    DIR_LIBRARY=/usr/sap/LNX/SYS/exe/run
    DBENV_CSH=SET
    LD_LIBRARY_PATH=/usr/jre118/lib/linux/native_threads:/opt/Platin
GUI/bin/./lib:/opt/kde/lib:/opt/kde2/lib:/usr/sap/LNX/SYS/exe/run
```

Die Angaben über die Threads:

```
Total Thread
Count: 10
Active Thread
Count: 8 JNI
Thread Count: 0
```

Sollte das Linux-System mit einem falschen Startumgebung aufgerufen werden, kann es zu folgendem Phänomen kommen: Die *startdb* ist in Ordnung und der Applikationsserver ist gestartet. Allerdings bricht die Anmeldung beim *guistart* ab und das Fenster schließt sich. In der Datei »startdb.log« wird ein Hinweis ausgeschrieben:

```
x_start: Unsuccessful startup, see /usr/sap/LNX/db/wrk/LNX/knl diag for
errors
```

Diese Diagnosedatei ist mit dem Befehl *more* einsehbar:

```
Linux:sqlnx> cd /usr/sap/LNX/db/wrk/LNX/knl diag
Linux:sqlnx> more knl diag
01-05 08:15:14 31629    11571 dbstart LINUX6.2.10  Build011-000-038-102
01-05 08:15:14 31629    11572 dbstart key for ipcresources0x440089ea
01-05 08:15:14 31629    11519 init_shm SHMCHUNK size0x7F800000
```

```
01-05 08:15:14 31629 50000
TCLUSTERtw;al;ut;2000*sv;10*ev;ti,100*dw,sn,rc,c s;30000*us;compress
01-05 08:15:14 31629 50001 TCLUSTER
```

Manchmal hilft es ja weiter! Zumindest sollte bekannt sein, daß so eine Datei existiert!

### 4.1.3 Netzwerkprobleme

Sollte der nächste Fehler auftreten, sagt uns die Erfahrung, woran es liegt: wir haben ein Problem mit dem Netzwerk. Wirkliche Hilfestellung bekommt man durch ein weiteres Log. Nach Anmeldung an einer neu gestarteten Konsole kommt die Ausschrift

```
Release 46B
Component NI (network interface), version 34
rc = -10, module niuxi.c, line 927
Detail NiPConnect2
System Call SQ_ERROR
Error No 111
```

'Verbindungsaufbau abgelehnt'

Beim Start werden Protokolldateien mit dem Namen »util.prot« erzeugt. Davon werden maximal neun Stück angelegt, ehe wieder mit der ersten begonnen und die entsprechende Datei überschrieben wird. Diese Dateien haben folgenden Inhalt:

```
----- UTILITY 6.2.9 (2002-01-05 08:14:14) --
USE SERVERDB 'LNX' ON 'linux'
+++ sqlaconnect, err: 5
*** ERROR -8888: SERVERDB NOT ACCESSIBLE,
database not running SESSION END
----- UTILITY 6.2.9 (2002-01-05 09:51:23) --
USE SERVERDB 'LNX' ON 'linux' 09:51:23
CONNECT SUPERDBA IDENTIFIED BY :X 09:51:23
SHUTDOWN
SESSION END
```

So haben wir noch eine Datei, aus der unter Umständen ein Hinweis zu ersehen ist, wenn der Start von SAP fehlschlägt.

#### 4.1.4 Abgelehnter Verbindungsaufbau

Es gibt unter Umständen noch ein Startproblem, das die Meldung 111 anzeigt: Der Verbindungsaufbau wird abgelehnt.

Eine andere Fehlerursache könnte das Editieren einer Profildatei auf Betriebssystemebene sein. Es ist gefährlich, die Einstellungen einer Profildatei mit einem normalen Editor zu verändern. Änderungen der Profilparameter sollten immer aus dem SAP-System heraus ausgeführt werden. Wurden bereits Werte verstellt und SAP startet nicht mehr, bleibt nichts anderes übrig, als die Einstellungen wieder zurückzunehmen. Hat man Glück, gibt es noch eine alte Datei mit der Endung ».bak«, die automatisch vom SAP-System angelegt wurde. In diesem Fall kann man die Datei sehr schnell durch eine Kopie in den Originalzustand zurückbringen.

#### 4.1.5 Problem mit dem Service 3218

Es gibt einen Fehler »Partner not reached, Service 3218«. Der Service-dienst unter Linux muß aktiv sein. Dies verifizieren wir mit dem Kommando `netstat -a | grep 3218`.

```
wasadm> netstat -a | grep 3218
udp        0      0 local host: 32779      local host: 3218      VERBUNDEN
udp        0      0 local host: 32780      local host: 3218      VERBUNDEN
udp        0      0 *:3218                 *: *
unix 2      [ ACC ]     STREAM  HÖRT          46498 /tmp/.sapstream3218
```

Dieser Service wird von der Startdatei `startsap` gestartet. Bei dem Begriff `udp` handelt es sich um ein paketorientiertes Protokoll, das aktiv sein muß, damit die Daten zwischen den SAP-Diensten ausgetauscht werden können.

#### 4.1.6 Weitere Hilfe

Im SAP-Menü wird auch eine Hilfe angeboten. Die Hilfedateien müssen erst eingebunden werden. In der Profildatei muß ein Parameter gesetzt werden, je nachdem, ob die Hilfedateien als HTML oder in einem anderen Format vorliegen. Da diese Einrichtung generell gilt, sollten die Einstellungen in der Datei »default.pfl« getroffen werden. Für die Einbindung der Dokumente als Dateien wird der folgende Eintrag vorgenommen:

```
eu/iwb/help_type = 5
eu/iwb/installed_languages = DE
eu/iwb/path_wi_n32 = \\server\docu\610\html\help\helpdata
```

Liegt die Dokumentation auf einem Webserver wie zum Beispiel dem Apache vor, ist die Einbindung folgendermaßen vorzunehmen:

```
eu/iwb/help_type = 2
eu/iwb/installed_languages = DE
eu/iwb/server_winn32 = <server>[:<port>]
eu/iwb/path_winn32 = docu\45a\PIainHtml\Helpdata
```

Es fällt auf, daß hier ein Eintrag »installed\_languages« vorhanden ist. Mit jedem ausgelieferten SAP-System werden verschiedene Sprachunterstützungen angeboten. Nur wenn die Texte und Beschriftungen in der Datenbank installiert sind, ist es möglich, die betreffende Sprache auszuwählen oder hier fest einzustellen. Wir brauchen uns an dieser Stelle nicht den Kopf zu zerbrechen, denn Deutsch und Englisch werden standardmäßig ausgeliefert. Schließlich handelt es sich bei SAP ja um eine deutsche Firma, und daß Englisch unterstützt werden muß, versteht sich von selbst. Ansonsten kann das SAP-System mit mehr als zwanzig Sprachen ausgeliefert werden, so daß es keine Schwierigkeiten geben sollte. Ein Problem ist eher, daß manche Masken mit englischen Bezeichnungen statt mit deutscher Beschriftung dargestellt werden, obwohl als Sprache Deutsch gewählt wurde. Doch wird dies sicherlich im Laufe der Zeit auch abgestellt.

## 4.2 Verbindungen definieren mit sm59

Die Transaktion *sm59* ruft das Bild mit dem schönen Namen *RSRFCRFC* auf. Hier werden verschiedene RFC-Destinationen aufgeführt. Der Begriff »RFC« steht für »Remote Function Calls«. Diese Funktionsaufrufe müssen vom Administrator an das eigene System angepaßt werden.

- ◆ R/3-Verbindungen
- ◆ Interne Verbindungen
- ◆ Logische Destinationen
- ◆ SNA/CPI-C-Verbindungen
- ◆ TCP/IP-Verbindungen
- ◆ Verbindungen über ABAP/4-Treiber

Manche Belegungen sind schon voreingestellt, wie beispielsweise die Angabe zum SAP-OSS.