Hochverfügbarkeit in der Standard Edition

Dr. Frank Haney
Consultant
Jena

Schlüsselworte:

Oracle Database Standard Edition, Hochverfügbarkeit, Standby Database

Die Herausforderung

Vor allem bei neuen Oracle-Implementierungen steht zwangsläufig die Frage, wie hochverfügbar das System sein muß. Eine Antwort ist in der Regel nicht ganz einfach. Natürlich kann und muß man versuchen, maximale Zeiten für eine Wiederherstellung des Systems sowohl nach einem teilweisen oder gar kompletten Ausfall als auch für mehr oder weniger planbare Wartungsmaßnahmen zu definieren. Diese werden dann in einem *Service Level Agreement* fixiert. Bestimmend dabei sind die Anforderungen des Geschäftsbetriebs, die einen entsprechenden Hard- und Software-Einsatz erzwingen. Allerdings gibt es zwei limitierende Sachverhalte: Zum einen ist nicht alles, was wünschenswert erscheint, auch finanzierbar. Redundante Hardware und entsprechende Software-Lizenzen verursachen erheblich Kosten. Zum anderen müssen auch das Know-how und die personellen Ressourcen vorhanden sein, um die implementierte komplexe Infrastruktur zu betreiben. Hochverfügbare Systeme werden ja geschaffen, um längere Zeiträume störungsfrei zu laufen. Wenn aber dann doch etwas Unvorhergesehenes und Ungeplantes passiert, ist ein erheblicher Erfahrungsschatz erforderlich, um das Problem zu verstehen und adäquat zu reagieren. Der folgende Beitrag hat mit beiden limitierenden Faktoren zu tun. Was kann für die Hochverfügbarkeit eines Datenbanksystems getan werden, das die Standard Edition (im folgenden SE) benutzt und von nicht ausreichend spezialisiertem Personal betreut wird?
Da entsteht die Frage, welche Hochverfügbarkeitslösungen von Oracle in der SE nutzbar und in der gegebenen Situation sinnvoll sind, um sich gegen die verschiedenen Fehler möglichst gut abzusichern. Bei genauerer Betrachtung findet man, daß fast alle entsprechende Funktionalität an die Enterprise Edition (EE) gebunden ist, so z.B. Streams, Data Guard, Flashback Database, oder gar zusätzlich zu dieser lizenziert werden muß. Im Einzelnen soll das hier nicht dargestellt werden, weil es zu weit vom Thema wegführen würde. Konkret geht es darum, mit den Mitteln der SE wenigstens ansatzweise das zu realisieren, was Oracle die *Maximum Availability Architecture* (MAA) nennt. Das bedeutet, sich durch Schaffung entsprechender Redundanz gegen Ausfall eines Servers genauso abzusichern wie gegen den Verlust der ganzen Datenbank.
Wie kann das nun mit Mitteln der SE erreicht werden? (Alle weiteren Ausführungen beziehen sich auf Oracle 11 Release 2, lassen sich eingeschränkt aber auch auf andere Releases übertragen.)

- **Serverausfall**: Durch Redundanz wird sichergestellt, daß die Arbeit mit der Datenbank beim Ausfall eines Servers unterbrechungsfrei weitergehen kann. Bei Oracle ist

die Basis dafür der Einsatz der Grid Infrastructure, die als Automatic Storage Management (ASM) die Arbeit mehrerer Instanzen auf der gleichen Datenbank ermöglicht und als Clusterware diese Instanzen in einem Konstrukt zusammenfaßt. Bekannt ist diese Feature als Real Application Cluster (RAC). Normalerweise ist das eine zusätzlich kostenpflichtige Option der EE. In der SE (nicht in der SE One) kann man RAC ohne (!) zusätzliche Lizenzkosten verwenden, solange man für das Cluster als Ganzes die Lizenzbedingungen der SE einhält. Da die SE im Unterschied zur EE auf Socket-Basis lizenziert wird, ist mit den modernen Multicore-Prozessoren RAC in der SE eine kostengünstige Hochverfügbarkeitslösung. Ein Zweiknoten-Cluster mit zwei Sockets je Server ist eine derzeit durchaus übliche Implementierung, falls keine genuinen Features der EE benötigt werden.

- **Datenbankverlust**: RAC alleine taugt nicht als Absicherung vor Datenverlust. Natürlich wird man immer auch eine adäquate Backup-Strategie implementieren. Allerdings läßt sich damit keine Hochverfügbarkeit gewährleisten, weil bei einem Verlust von Daten, z.B. durch eine fehlende oder beschädigte Datendatei, diese aus dem Backup wiederhergestellt und dann mit den Log-Dateien recovert werden müssen. Für diese Zeit steht die Datenbank in der Regel nicht für Nutzeraktionen zur Verfügung. Die Lösung des Problems ist ein Replikat der Datenbank, zu dem bei einem Fehler des Originals umgeschaltet wird. Auf diese Weise läßt sich auch Vorkehrung für den Katastrophenfall treffen. Oracle hat dafür eine ganze Reihe von Möglichkeiten, die alle eines gemeinsam haben, sie sind nur in der EE verfügbar. Das gilt auch für die naheliegende Variante Data Guard als Physical Standby. Die Frage ist nun, ob man diese Funktionalität mit Mitteln der SE nachbauen kann. Das impliziert das Problem, eine Standby-Datenbank an einem entfernten Ort als Replikat der originalen Datenbank einzurichten und manuell mit dem Original zu synchronisieren. Wie das geht und in welchem Umfang sich dabei Data Guard-Funktionalität gewinnen läßt, soll der Hauptgegenstand des Beitrags sein. Zunächst aber ein paar Bemerkungen zu RAC.

RAC in der Standard Edition

Hier gibt es wenig Spezifisches. Die Grid Infrastructure (Clusterware + ASM) kennt keine Unterscheidung von SE und EE und wird ganz normal installiert.
Bei der RAC-Datenbank müssen die Grenzen der SE für das ganze Cluster eingehalten werden. Daneben gilt für die zusatzkostenfreie Nutzung der RAC-Option:

- Als Shared Storage darf nur ASM verwendet werden, kein anderes Cluster File System oder RAW. (Das gilt auch für OCFS bzw. OCFS2!)
- Alle (!) Datenbankdateien, die ASM speichern kann, müssen auch in ASM gespeichert werden. (Die Liste ist lang! Es gibt zwei Ausnahmen, auf die ich später noch zurückkomme.)
- Alle nicht-Datenbankdateien von Oracle müssen in ACFS oder einem lokalen File System gespeichert werden.
- Es darf keine anderes Cluster Management außer der Oracle Clusterware verwendet werden.

Weitere Einschränkungen bezüglich für das RAC sinnvoller Funktionalität ergeben sich aus allgemein in der SE fehlenden Features wie paralleles Backup und Recovery, Parallel Query/DML etc.

Defizite einer manuellen Standby Database gegenüber Data Guard

Die Frage ist, worauf man alles verzichten muß, wenn man eine Standby-Lösung mit den Mitteln der SE quasi nachbaut. Das möchte ich im Folgenden kurz zusammenfassen. Die Beantwortung der sich daraus ergebende Frage, ob eine solche Lösung unter diesen Randbedingungen überhaupt sinnvoll ist, möchte für ein zu ziehendes Fazit zurückstellen. Wo also muß man mit mehr oder weniger großen Einschränkungen gegenüber der EE leben? Die beiden gravierendsten sind:

- Keine Archivierung über das Netzwerk an ein Remote-Ziel (Das Attribut SERVICE bei der Konfiguration von Archivierungszielen ist in der SE ungültig.)
- Kein Managed Recovery (Archive Logs werden auf der Standby-Seite nicht automatisch angewendet.)

Damit zusammenhängend und darüber hinaus gibt es weitere Funktionalität, die in der SE nicht zur Verfügung steht. Hier eine unvollständige Liste:

- Keine Erzeugung der Standby-DB von der aktiven Primär-DB
- Keine Standby Redo Logs und damit kein Real Time Apply der Logs
- Kein Graceful Switchover
- Kein Data Guard Broker und deswegen auch kein Fast Start Failover
- Keine selbständige Erkennung und Behebung von Log Gaps
- Keine Propagierung von physischen Änderungen an der Primär-DB zur Standby-Seite (neue Tablespaces oder Datendateien)

Ein weiteres Problem tritt auf, wenn die Primärseite ein RAC ist, wo es ja die weiter oben genannten Beschränkungen gibt. Wie erzeugt und pflegt man eine Standby-DB, wenn alle relevanten Dateien der Primärseite im ASM liegen müssen? Natürlich gibt es Möglichkeiten, Dateien aus dem ASM herauszuholen und zur Standby-Seite zu kopieren, z.B. FTP oder DBMS_FILE_TRANSFER. Das ist aber umständlich und verkompliziert die Abläufe. Zum Glück kommt Oracle uns hier bei den Lizenzbedingungen für RAC in der SE entgegen und erlaubt folgende Ausnahmen:

- RMAN-Backups in das lokale File System, mit denen man eine Standby-DB überhaupt erst implementieren kann
- Ein *lokales* Archivierungsziel außerhalb von ASM, das zum manuellen Nachziehen der Standby-DB genutzt werden kann

Implementierung der manuellen Standby Database

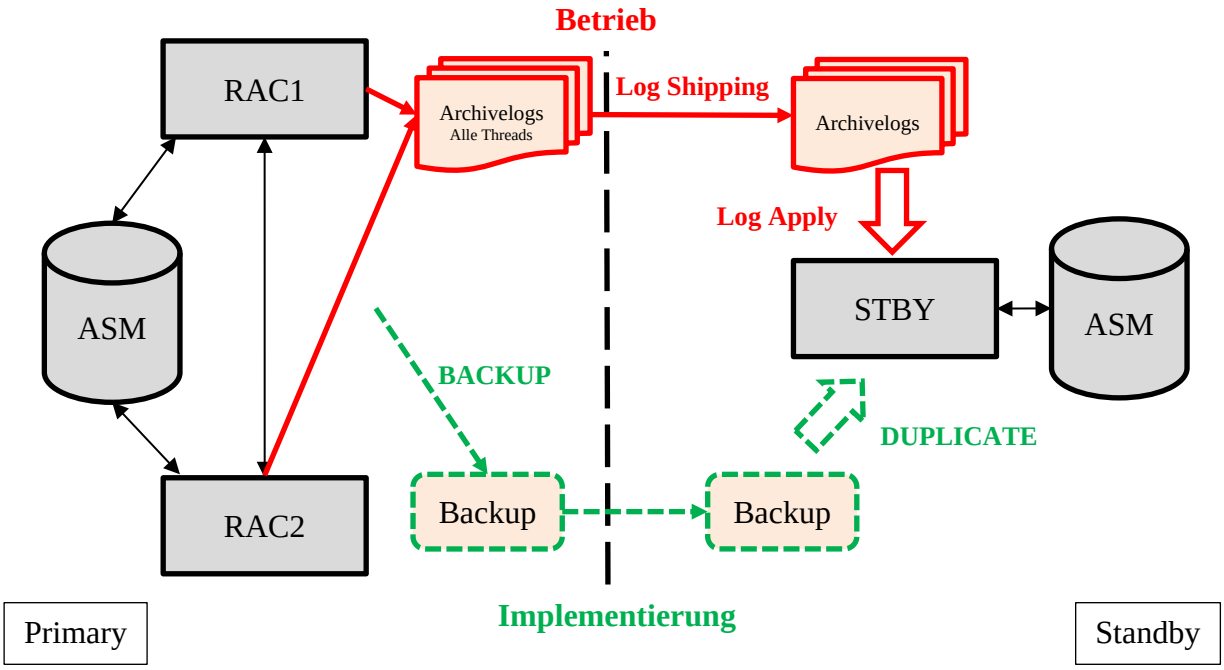

Abbildung 1: Die Zielstruktur

Wenn man die oben genannten Einschränkungen einrechnet, dann unterscheidet sich die Implementierung nicht besonders von Data Guard.
Zunächst einmal muß das Standby-System vorbereitet werden:

- Benutzer und Gruppen müssen angelegt und konfiguriert werden
- Herstellung der SSH-Konnektivität zur Primärseite
- Konfiguration des Betriebssystems entsprechend den Oracle-Vorgaben
- Da ASM verwendet werden soll, muß die Oracle Grid Infrastructure für einen Standalone Server installiert werden. (Siehe meinen Vortrag auf der diesjährigen DOAG-Datenbankkonferenz in Düsseldorf.)
- Einrichtung der ASM-Diskgroups, am besten genauso wie auf dem Original
- Konfiguration und Start eines Listeners aus dem Grid Home
- Installation der Datenbanksoftware
- Installation des Cloud Control Agent

Daneben sind auch Vorbereitungen auf der Primärseite notwendig:

- Erzwingen des Logging: `alter database force logging;`
- Konfiguration eines zusätzlichen lokalen Archivierungsziels
- Anlegen einer Textkopie des SPFILE und Kopieren zur Standby-Seite
- Kopieren der Paßwortdatei zur Standbyseite
- Aufnahme der Standby-DB in die Namensauflösung (tnsnames.ora) aller beteiligten Parteien: Original, Standby, Clients

Dazu kommen Anpassungen auf der Standby-Seite wie die Bearbeitung der Parameterdatei und Generierung eines SPFILE. Es müssen vor allem Parameter angepaßt werden, die sich auf die Primärdatenbank als RAC beziehen. Die Implementierung der Standby-Datenbank

startet mit einem RMAN-Vollbackup des Originals in ein Verzeichnis außerhalb des in ASM befindlichen Fast Recovery Area (FRA):

```
rman target /
RMAN> sql 'alter system archive log current';
RMAN> backup as backupset database format
'/oracle/standby/backup/initial_for_standby_%U.bkp' include current
controlfile plus archivelog;
```

Dieses wird dann zur Standby-Seite transferiert:

```
scp /oracle/standby/backup/* standby-server:/oracle/standby/backup
```

Jetzt kommt der entscheidende Schritt, das Anlegen der Standby-Datenbank aus dem Backup. Die Standby-Instanz muß sich dazu im Status NOMOUNT befinden.

```
rman target sys/sys-password@original-cs auxiliary sys/ sys-
password@standby-cs
RMAN> duplicate database for standby nofilenamecheck dorecover;
```

Dabei kann es zu Fehlermeldungen kommen, die man mehr oder weniger ignorieren kann, zB.
`RMAN-06053: unable to perform media recovery because of missing log`
Diese resultieren vor allem daraus, daß das Recovery nicht bis zur aktuellen SCN durchgeführt werden kann.
Damit ist die Standby-DB angelegt. Im nächsten Schritt geht es darum, sich um ihre Aktualisierung zu kümmern.

Aktualisierung der Standby Datenbank

Nun muß die Standby-Seite möglichst synchron mit dem Original gehalten werden. Da es in der SE keine Archivierung an ein Remote-Ziel (automatisches Log Shipping) und auch kein Managed Recovery für die Standby-DB gibt, ist das nur in Grenzen möglich. Man muß sich immer der Tatsache bewußt bleiben, daß sich mit einer manuellen Lösung nie der gleiche Schutzstatus wie mit Data Guard erreichen läßt.
Im Internet finden sich mehr oder weniger elaborierte Varianten für diese Aufgabe. Ich bin der Auffassung, daß man es so einfach wie möglich halten sollte. Im gegebenen Projekt wurde folgender Ablauf realisiert:

- Die aktuellen Online Redo Logs aller Threads der Primärseite werden archiviert.
- Die Archive Logs werden mit rsync zur Standby Seite kopiert. Die Verwendung von rsysnc hat gegenüber scp den Vorteil, daß nur neue Dateien kopiert werden.
- Das Recovery der Standby-DB wird durchgeführt.
- Alte Archive Logs werden auf der Standby Seite gelöscht. Das Zeitfenster muß entsprechend großzügig gewählt werden.

Im Shell-Skript (Ausschnitt)sieht das so aus:

```
#!/bin/bash
# Archivierung der aktuellen Log-Dateien der Primaerdatenbanken
$ORACLE_HOME/bin/sqlplus sys/sys_password@primary as sysdba <<EOF
alter system archive log current;
```

```
exit
EOF

# Transport zur Standby-Seite
rsync -e ssh -Pazv node1:/oracle/standby/archivelog/*.arc /oracle/standby/archivelog/
rsync -e ssh -Pazv node2:/oracle/standby/archivelog/*.arc /oracle/standby/archivelog/

# Recovery der Standby DB
$ORACLE_HOME/bin/sqlplus sys/sys_password@standby as sysdba <<EOF
recover automatic standby database;
cancel
exit
EOF

# Alte Archivelogs loeschen aelter als 8 Tage
find /oracle/standby/archivelog -type f -mtime +8 –delete
exit
```

Dieses Skript wird dann per Cron-Job ausgeführt. Die Frequenz richtet sich nach der Transaktionslast der Primärseite und nach dem Datenverlust, den man gegebenenfalls zu tolerieren bereit ist. Da ist ein Kompromiß notwendig.
Das Recovery bringt natürlich regelmäßig eine Fehlermeldung, weil die nächste Log-Sequenz fehlt, die ja auf der Primärseite noch nicht archiviert worden ist.
Noch ein Hinweis: Unter Umständen müssen auch Backupskripte angepaßt werden. Denn es wäre kontraproduktiv, wenn auf der Primärseite Archivelogs verschwinden, die noch nicht zur Standby-Seite übertragen sind. Es empfiehlt sich also nicht, zwischen zwei Skriptläufen die Archive Logs mit der Option `DELETE INPUT` zu sichern.

Test und Überwachung der Funktionalität

Natürlich muß man überprüfen, ob die Archive Logs auch auf der Standby-Seite angewendet werden. Das ist nun nicht so einfach wie bei Data Guard. Dort braucht man nur zu überprüfen, ob für die letzte übertragene Log-Sequenz in der Spalte `APPLIED` von `V$ARCHIVED_LOG` YES steht. Das läßt sich im gegebenen Fall nicht anwenden, weil auf der Standby-Seite diese View gar nicht gefüllt wird. Es besteht nur die Möglichkeit, auf der Primärseite eine Änderung an Daten oder Objekten durchzuführen, dann die Standby-Seite zu aktualisieren und danach die Standby-DB Read Only zu öffnen, um zu überprüfen, ob die Änderungen auch angekommen sind. Dann kann man sie wieder schließen und mounten, damit der nächste reguläre Synchronisationszyklus stattfinden kann.
Das ist nun aber ein Verfahren, mit dem man einmalig die Implementierung testet. Es ist weniger geeignet, um kontinuierlich zu überwachen, daß die Standby-Seite nicht zu weit hinter dem Original zurück ist. Im Falle von Data Guard hat man dafür neben verschiedenen Views die Hochverfügbarkeitskonsole des Enterprise Managers. In der SE funktioniert das nicht. Ich habe eine Weile gesucht, um eine Abfrage zu finden, die zuverlässig anzeigt, ob die Standby-Seite länger als einen Synchronisationszyklus zurück ist. Einige Views werden nicht befüllt, wie z.B. `V$ARCHIVED_LOG`, andere beim Synchronisieren nicht inkrementiert, wie z.B. `V$LOG`. Möglich wäre die `V$LOG_HISTORY`. Ich habe dann die `V$DATAFILE_HEADER` genommen, in der `CHECKPOINT_SCN` und `CHECKPOINT_TIME` bei jeder Synchronisation fortgeschrieben werden. Folgende Abfrage löst also das Problem:

```
SQL> select sysdate-max(checkpoint_time) from v$datafile_header;
```

Diese kann man unter Angabe geeigneter Schwellenwerte in eine Nagios-Überwachung integrieren. Natürlich ließe sich auch eine komplexere Verfahrensweise implementieren.

Vorgehensweise bei physischen Änderungen am Original

Hier ist in erster Linie gemeint, was beim Erstellen eines neuen Tablespace oder dem Hinzufügen einer neuen Datendatei zu einem vorhandenen passiert, bzw. zu tun ist. Probleme mit den Pfaden gibt es nicht, da wir uns ja entschlossen haben, auf beiden Seiten ASM mit dem gleichen Layout zu verwenden. Nun gibt es den Parameter `STANDBY_FILE_MANAGEMENT`, der auf `AUTO` gestellt werden kann und dann dafür sorgt, daß eine neue Datendatei automatisch auch auf der Standby-Seite angelegt wird. Die positive Botschaft ist, daß das auch in der SE funktioniert, die negative dagegen, daß man es nicht darf, weil es genuine Funktionalität von Data Guard und damit der EE ist.
Was passiert also, wenn der Parameter, wie in der SE erforderlich, auf `MANUAL` gesetzt wird, bei derartigen Änderungen am Original? Das notwendige Recovery auf der Standby-Seite schlägt fehl mit:

```
ORA-00283: Recovery Session wegen Fehlern abgebrochen
ORA-01274: Datendatei '…' kann nicht hinzugefugt werden - Datei konnte
nicht erstellt werden
```

Ein neuer Tablespace wird zwar auf der Standby-Seite angelegt, als Datendatei aber nur ein Dummy in `$ORACLE_HOME/dbs` mit einem kryptischen Namen wie `UNNAMED00013` angegeben. Die Lösung ist aber nicht schwer. Man muß die Datei einfach mit richtigem Namen an der richtigen Stelle neu anlegen und das Recovery wiederholen. Deb Namen des Dummy besorgt man sich aus der `V$DATAFILE`.

```
SQL> alter database create datafile '$ORACLE_HOME/dbs /UNNAMED00013' as
'+ORADATA';
SQL> recover automatic standby database;
```

Wegen Oracle Managed Files (OMF) darf man nur die ASM-Diskgroup und nicht den vollständigen Pfad und Namen angeben.
Einige Änderungen werden dagegen problemlos automatisch an die Standby-Seite weitergereicht:

- Einstellung der Maximalgröße einer Datei
- Änderung der aktuellen Dateigröße mit `RESIZE`
- Allokierung neuer Extents bei Nutzung von `AUTOEXTEND ON` für die Dateien
- Löschen von Tablespaces

Failover zur Standby-Seite

Wie schon in der Auflistung der Einschränkungen von Standby in der SE festgestellt, gibt es keinen einfachen Rollentausch (Switchover) beider Seiten. Damit ist Standby in der SE ungeeignet als Hochverfügbarkeitslösung für geplante Auszeiten (Wartungsmaßnahmen). Es bleiben ungeplante Auszeiten, wie ein Crash des Originalsystems oder ein Ausfall des primärseitigen Storage. Das läßt sich auch in der SE durch ein Failover zur Standby-Seite absichern. Dieses nun erfordert nun keine besonders komplexen Operationen:

- Es sollte ein Versuch unternommen werden, die aktuellen Logdateien der Primärseite zu archivieren.
- Alle noch nicht übertragenen Archive Logs sollten nach Möglichkeit zur Standby-Seite kopiert und dort angewendet werden.
- Die Standby-DB wird aktiviert.
- Danach kann sie READ WRITE geöffnet werden.
- Die Clients müssen auf die Standby-Seite zugreifen.

Was ist nun aber zu tun, wenn man nach Behebung des Problems auf der Primärseite zurückschwenken möchte? Auch da sind die Möglichkeiten gegenüber Data Guard begrenzt. Letztlich läuft es darauf hinaus, daß zweimal das hier beschrieben Verfahren angewendet wird:

- Die ehemalige Primärseite wird als Standby-DB implementiert.
- Dorthin wird ein Failover durchgeführt.
- Um den Ausgangszustand wiederherzustellen, wir die ursprüngliche Standby-DB neu implementiert.

Daraus ergibt sich auch, daß man sich genau überlegen wird, ob man den Fehler auf der Primärseite sucht und behebt oder zur Standby-Seite schwenkt. Die Herstellung des Originalzustandes ist doch recht aufwendig.
Natürlich kann man das Failover auch in Skripte packen, die auf ein primärseitiges Event reagieren, z.B. mittels Fast Application Notification (FAN - Bestandteil der Oracle Grid Infrastructure), dem Fast Start Failover von Data Guard nachempfunden. Ob man aber die Kontrolle über das Failover aus der Hand geben sollte, ist nach meiner Ansicht hier noch zweifelhafter als bei Data Guard.

Fazit

Mit einer manuellen Standby-Datenbank kann man das ganz elementare Bedürfnis nach einem möglichen Desaster-Recovery befriedigen. Nicht mehr, aber auch nicht weniger. Man muß sich auch immer der Tatsache bewußt bleiben, daß sich auf diese Weise kein völliger Schutz vor Datenverlust erreichen läßt. Einen solchen versprechen auch die Anbieter kommerzieller Tools wie DBVisit nicht. Denn sie können ja auch nichts anders tun, als die Funktionalität von Data Guard so weit wie möglich mittels Skripten nachzubauen und per graphischer Oberflächen nutzerfreundlicher zu gestalten. Die Einschränkungen der SE gelten natürlich fort.

Kontaktadresse:
Dr. Frank Haney
Anna-Siemsen-Str. 5
D-07745 Jena

Telefon: +49(0)3641-210224
E-Mail: info@haney.it
Internet: http://www.haney.it