QUASAR™

Accounting Software

Installation and Setup (Version 1.4.7)

P. Tonnellier B. Pepers

22 June 2005

Contents

Chapter 1

Overview

This document covers the installation and setup of Quasar Accounting. It is for system administrators who need to install Quasar on different operating systems and configure Quasar to work with different database drivers.

Note:
- Linux™ is a trademark of Linus Torvalds
- Quasar™ is a trademark of Linux Canada Inc.

Chapter 2

Compiling Quasar

If you have a Quasar RPM package, you can skip to the "Installing Quasar" chapter since Quasar has already been built for you. If you are experienced in building projects from their source then the simple version of the rest of this chapter is that you can use the normal "`./configure`, `make`, `make install`" process so with just a quick review of the required packages you should be able to build Quasar without any problem.

Quasar requires the following software for building:

- the g++ compiler
- other standard devel tools such as make
- Qt version 3.0 and higher
- autoconf version 2.57 and higher
- Tcl version 8.3 and higher
- Tk version 8.3 and higher
- ICU version 3.2 and higher

You will need to have the development packages installed for each of these which on RPM based Linux distributions like Red Hat and SuSE can be found as packages with "`-devel`" in the name.

The ICU package is for I18N support and is from IBM. Linux Canada have created ICU packages for all the distributions we create Quasar packages for but if you are using a different distribution, you will need to get the ICU source and install it before you can compile Quasar. The ICU source is available from http://icu.sourceforge.net. Also if your distribution included ICU, but its a much older version, then you will need to update this to 3.2 before you compile Quasar.

Once you've made sure you have the required software installed to build Quasar, you need to extract the source code using a command similar to this:

```
tar xzf quasar-1.4.3_GPL.tgz
```

You may need to use a different file name depending on which version of Quasar you are building. This will create a directory called "quasar-1.4.3_GPL" which you need to go into using "`cd quasar-1.4.3_GPL`". Once in this directory you need to use the configure script which will inspect your system and make sure all that Quasar needs is installed. You can run the configure script like this:

```
./configure
```

It should print out a long list of tests and then at the end it generates the files needed for building Quasar on your computer. If the configure fails for some reason, review the line that failed and see if it tells you what is needed (for instance if the line that fails is the one thats checking for the "tcl.h" include file then you likely are missing the Tcl development package and need to install it). Fix any failures and then run the "`./configure`" command again to try configuring the build again.

Once the configure script has completed properly you can compile Quasar using this command:

```
make
```

This should build all the parts of Quasar on your system. After running make you should run it again to see if any parts failed. If you get any messages on the second make that its trying to do some work followed by some error messages, then the make failed. Try to see if you can tell the problem from the error message but if not, post the problem to the Linux Canada Inc. Quasar development mailing list and we will help you out.

There is an option at this point to test running Quasar from your home directory as a regular user without installing it. To do this you should rename the directory from quasar-1.4.3_GPL to ~/Quasar and you can run this command in a terminal window:

```
cd ~/Quasar
bin/quasard
```

It should indicate that the server is started and running on port 3599. Now you need to edit or use bin/quasar_setup to setup the database types you want to use (review the specific chapter for setup information of each database type) and when this is working you can use bin/quasar_setup to create a company database. If this works you can try importing the test data using quasar_setup and then run “bin/quasar” to run the client and connect to the company. When you create a new company it has one user created called “admin” and the password for that user is set to “admin” as well so you can use that to login to your test company.

Running from your home directory can be good for a quick test and is how Quasar is run by developers that are adding to the program but some of the permissions on directories (for Firebird in particular) can be tricky to get right and it is not how Quasar should be used in a live environment.

Chapter 3

Installing Quasar

If you are building Quasar yourself, you will need to install from the source code you compiled in the previous chapter. If instead you have downloaded a pre-built RPM package for your distribution, then read the section on installing from RPM.

3.1 Install From Source

Once the make is successful you can install Quasar. You need to be the root user to do this so use the "su" command to become root if you are not already. Once you are the root user you can use this command to install Quasar:

```
    make install
```

This will install Quasar to the /opt/quasar directory and will setup xinetd to run the quasar_clientd program. It also create a group and a user called "quasar" which is the user that quasar_clientd runs as. This installs all the parts of Quasar including the server, client, and database drivers. You can also choose to just install part of the package by using a command like this:

```
    make install_server
```

Alternatively you could just use the "install" script directly which is all the above make command does. Just run "./install <type>" where <type> can be "server", "client", ... For more information on what files are needed for each part of Quasar you need to review the install script.

3.2 Install From RPM

If you have downloaded RPM packages for your distribution then you can simply install the packages you require. You will need to install the "server" package on one system on your network, and the "client" package on any workstations that you want to be able to use Quasar. The client can also be installed on the server computer if you want to be able to use Quasar from it.

Before you install Quasar though, you will need to be sure you've installed the ICU version 3.2 packages. They are available on our ftp server in the pub/icu_3.2 directory and you will need to install the icu and libicu package. You will need the libicu-devel package if you want to recompile Quasar from scratch or make source code changes.

Example commands to install the packages on the server:

```
    rpm -i icu-3.2-1.i386.rpm
    rpm -i libicu-3.2-1.i386.rpm
    rpm -i quasar-server-1.4.3_GPL.i586.rpm
```

And an example of installing on the client workstation is:

```
    rpm -i icu-3.2-1.i386.rpm
    rpm -i libicu-3.2-1.i386.rpm
    rpm -i quasar-client-1.4.3_GPL.i586.rpm
```

And finally if you are installing all of Quasar on one computer then you can do this:

```
    rpm -i icu-3.2-1.i386.rpm
    rpm -i libicu-3.2-1.i386.rpm
    rpm -i quasar-client-1.4.3_GPL.i586.rpm
    rpm -i quasar-server-1.4.3_GPL.i586.rpm
```

Chapter 4

Quasar Setup

In a typical Quasar setup you would have a server computer and one or more workstation computers. You could also have a separate database server. For single users or small stores though, you may want to run the database, server, and client all on one computer. The setup of each of these is described below and even though they are described separately, if you want to run it all on one computer then just follow each sections instructions.

Much of the setup of Quasar uses the tool called quasar_setup. This tool is used for both setup and for administration tasks. You use it to setup the database drivers so Quasar can use your databases, for creating and deleting companies, for updating company databases to a new version, and for doing backups and restores of companies. This program needs to be run as the "root" so that it has the permissions to do everything required. You can do this by logging in as "root" or else by using the kdesu or gnomesu tools. To run quasar_setup as "root" on a kde system, you would use this command:

```
kdesu /opt/quasar/bin/quasar_setup
```

4.1 Server Setup

The server is the computer that runs the quasar server and is the main point that all clients connect to to get a list of companies you've created. It also makes connections to the database for the clients. When you installed the "server" Quasar package (or installed from source), the xinetd daemon was setup so that the quasar_clientd program is run when any users try to connect to the server to talk to Quasar. The quasar_clientd program is used as a centralized data service to the Quasar clients. It gives list of data such as company definitions, report definitions, ... to the clients and allows the client to get the data from the server. This provides one central repository for Quasar reports so if you need to add a new report you can just install it on the server and it will show up on all the client systems.

The Quasar configuration files are stored in /opt/quasar/config. They are XML files and can be editted manually but a program called quasar_setup has been provided to set all configuration parameters. Usually all you will need to do with a new Quasar installation is setup the database driver and then use quasar_setup to create a new company. Once you get to that point you can refer to the Users Guide to find out how to setup a new company for use.

The Quasar server configuration is stored in the server.cfg file and you can set the values from the Server tab in the quasar_setup program. The fields in this screen are:

- Port: The port is the port number that will be used by the quasard server program. The quasard server program is not normally used though since its better to let xinetd or inetd handle the function of listening on a port for connections and starting quasar_clientd to handle each connection. If the port is blank or zero, the default port of 3599 will be used.

- System Id: The system id is used when using replication with Quasar which is an advanced subject not covered here. It should be left at the default value of zero.

- Data Dir: The data directory is where quasar_clientd will look for data files for the clients. It needs to be readable by the "quasar" user on your system since this is what quasar_clientd runs as.

- Driver Dir: The driver directory is where the database drivers for Quasar are stored.

- Backup Dir: This is the default directory that quasar_setup will use when doing backups and restores of companies.

- Import Dir: This is the default directory that quasar_setup will use when doing XML imports.

- Admin Password: The admin password is the default password for the admin user. When a new company is created in quasar_setup, it will have one user defined called “admin” and this is the password that user will have by default.

Normally the only part of the server configuration that will be changed is the default admin user’s password. Most of the setup on the server for Quasar is in the driver configuration. Please read the “Database Setup” chapter for more information on this.

4.2 Workstation Setup

Each workstation runs the “quasar” program which is the GUI interface to Quasar Accounting. The first time it is run you can select the locale you are using and the address (host name or IP address) of the server computer. It will then connect to the server (giving an error if it can’t connect) and present you with a list of companies that have been created on the server and a place to type in your username and password. Once you’ve typed in a valid username and password and its been verified you will get the main user interface screen of Quasar (which can be customized so each user gets a different screen).

You can use the quasar_setup program on the client computer as well. When run there it will only setup the two client configuration items which are:

- Locale Dir: The directory holding the localized messages and help screens.
- Station: The station number when in Quasar

The user configuration such as the color, font, and things like the default server, company, and user name for the login screen come from the user.cfg file found in the users home directory in the .quasar directory. This configuration is normally set from within Quasar but it could also be installed by copying the file from another computer or else editing the file manually.

4.3 Windows Setup

The Quasar client program that runs on Linux Workstations can also be used on computers running Windows. You will need the Quasar143.exe program downloaded from our web site or from our CD-ROM and then execute this setup program on your Windows computer and follow the instructions to install the Quasar client.

Chapter 5

Database Setup

Quasar support three database types currently. They are the open source PostgreSQL and Firebird databases and the commercial Sybase SQL Anywhere database. Each type of database has its own configuration screen in the quasar_setup program and although each one has different parameters, they mostly are regarding how to connect to the database and how to create new databases.

5.1 PostgreSQL

The PostgreSQL driver for Quasar Accounting is for PostgreSQL versions 7.3 and higher. To setup this driver you first need to do some configuration of PostgreSQL. A description of how to do this that works with SuSE 9.1 and PostgreSQL 7.4.5 is given here though there may be slight differences if you have a different Linux distribution or a different PostgreSQL version.

To setup PostgreSQL you need to configure the server part of PostgreSQL and then add users for DBA and non-DBA access. The DBA user needs to be able to create databases and is used by quasar_setup when you create a new company. The non-DBA user is used for regular client access and doesn't need to be able to create databases. This example will use the name "quasar_dba" for the DBA user and "quasar" for the non-DBA user but you can pick something else or even just use one user for both if you decide to.

To setup PostgreSQL on SuSE 9.1 the service first has to be started. The command to do this (as root) is:

```
/etc/init.d postgresql start
```

Next you need to become the “postgres” user to configure PostgreSQL. You can switch to the “postgres” user using this command:

```
su - postgres
```

Now you need to add the “quasar_dba” user to PostgreSQL. You can use the “createuser” command to do this. For example:

```
createuser -d -E -P quasar_dba
```

The “-d” option is required so that the “quasar_dba” user can create databases. The “-E” is so that the password is stored encrypted and the “-P” is to make the createuser command ask you for a password. Type in the password you’ve chosen for “quasar_dba” twice and the “quasar_dba” user will be added and ready to use.

The “quasar” user can also be created at this point using a command like this:

```
createuser -E -P quasar
```

Next you need to change to the “data” directory and edit some of the configuration files found there. Start first with the “pg_hba.conf” file which is used to control how PostgreSQL validates users. You can read the configuration file and manual from PostgreSQL on how to setup this file but for a simple case I just added a line like this:

```
host all all 127.0.0.1 255.255.255.255 md5
host all all 192.168.1.0 255.255.255.0 md5
```

This allows connection on my local network (all my computers are using 192.168.1.x for IP addresses) or locally using the loopback address of 127.0.0.1 as long as a valid username and password are given. There are a lot of different ways this can be setup so please read the manual from PostgreSQL on the setup of the “pg_hba.conf” file for more information.

The other configuation file that can to be changed is “postgresql.conf” which is also found in the “data” directory. Edit it with your favorite text editor and find the line with “#tcpip_socket = false” and change it to “tcpip_socket = true” instead. This change is so that PostgreSQL will allow TCP/IP network connections from other computers on your network which is not strictly required anymore since the database connection is now made on the server by quasar_clientd but you may still want to do this so you can use other third-party utilities from the workstation computers.

Once these changes have been made you can logout of being the "postgres" user and back to root. Then you need to restart the PostgreSQL service for it to notice the changes. Use this command to do this:

```
/etc/init.d/postgresql restart
```

Also if you want the PostgreSQL service to be available when you boot up (as you certainly would if you are using it for Quasar) then you can run this command to add it to the startup scripts on booting:

```
chkconfig postgresql on
```

Once you've configured PostgreSQL, you will need to setup the postgresql.cfg file for Quasar to work with PostgreSQL. The config file is setup from quasar_setup in the Drivers tab. Select the PostgreSQL driver and click on the Configure button to bring up the PostgreSQL driver configuration screen. The fields in the screen are as follows:

- Hostname: The hostname should be the IP address or hostname of the computer that is running the PostgreSQL server. This allows you to run the database on a different computer than the Quasar server if desired though for most people, running the database server on the same computer as the Quasar server will be fine. If you leave the hostname blank, PostgreSQL will use a unix-socket connection rather than a TCP/IP socket which will be faster for local connections. If you specify "localhost" or use another hostname, the PostgreSQL server will have to be configured to allow TCP/IP connections as mentioned earlier.
- Port: The port is the socket port to use for the server connection. The default and normal port that PostgreSQL's server runs on is 5432 and this will be used if the port in the postgresql.cfg is blank or zero. You shouldn't need to change this.
- Library: The library is the full file path of the libpq.so library supplied by PostgreSQL for clients to access the database. If its blank, Quasar will try to make a best guess checking for likely names in /usr/lib but its best to specify the filename.

- DBA Username: Enter the username you picked for DBA access.
- DBA Password: Enter the password you gave to the DBA user.
- Username: Enter the username you picked for non-DBA access.
- Password: Enter the password you gave to the non-DBA user.
- Character Set: Enter the character set you want used when you create company databases. The default is UNICODE which should work with any type of text in the world but there is a small speed penalty to be paid for this and you may decide to instead use ISO8859_1 or some other character set that is a more exact match for your needs.

Now that the config files are setup, you should be able to use quasar_setup to create companies using PostgreSQL as the database type. If creating the company fails, the first thing to test is whether you can use the "quasar_dba" user you created from the "psql" command like this:

```
psql -h localhost -U quasar_dba template1
```

If you have left the hostname blank in the postgresql.cfg file then leave off the "-h localhost" above and if you are using a different hostname than localhost, specify that in the above command. You should be prompted for a password and once you enter in the password you used for "quasar_dba" you should get a prompt like "template1=# ". If this doesn't work then there is something wrong with your PostgreSQL setup or your password for "quasar_dba". If it does work then you should verify the password you placed in postgresql.cfg is correct and that you specified the "-d" options to createuser when you created the "quasar_dba" user.

If you are having any problems with PostgreSQL, use the "Test" button in the configuration screen. It will try to verify that all your configuration options work and will give you messages suggestion what may be wrong if something fails. Once you've created a company in quasar_setup, access from the client should work unless you've made a mistake with the non-DBA username and password.

5.2 Firebird/Interbase

The Firebird driver for Quasar Accounting is for virtually any version of Firebird (the latest tested with is 1.5.2) and it should work for the Interbase database as well from Borland. As of version 1.5.2 there is still some care needed in choosing which compile of Firebird to install. There is the CS version which is the Classic Server and there is the SS version which is the Super Server. They differ in how they do internal threading and how they will take advantage of multiple processor computers. There is also two versions of the SS release. One is called NPTL which is for systems using the new threading model and the other isn't called NPTL which is for older Linux distributions. Most of the current distributions now use the new threading model so if you want to use the SS version of Firebird, you will need the NPTL compile of it. The CS version will work on old and new systems.

Once you've installed Firebird from the RPM or tar file, you will need to create a user for non-DBA access. To do this you need to become the root user and then get the current sysdba password from the SYSDBA.password file in /opt/firebird. Once you know it, you can do these commands as root:

```
export FIREBIRD=/opt/firebird
export PATH=$FIREBIRD/bin:$PATH
gsec -user sysdba -password xxx

GSEC> add quasar -pw yyy
GSEC> quit
```

Replace xxx with the sysdba password and replace yyy with the password you want the non-DBA user to use. You can also replace "quasar" in the add command with some other name if you want to use a different name for the non-DBA user.

Once you've added the non-DBA user, you will need to setup the firebird.cfg file for Quasar to work with Firebird. The config file is setup from quasar_setup in the Drivers tab. Select the Firebird driver and click on the Configure button to bring up the Firebird driver configuration screen. The fields in the screen are as follows:

- Hostname: The hostname should be the IP address or hostname of the computer that is running the Firebird server. This allows you to run the database on a different computer than the Quasar server if desired though for most people, running the database server on the same computer as the Quasar server will be fine. This defaults to localhost which should normally be what you want to use.

- Port: This is the port that the Firebird server is running on which defaults to 3050 since thats the normal one Firebird uses.
- Library: The library is the full file path of the libfbclient.so library supplied by Firebird for clients to access the database.
- DBA Password: This is the password for the normal Firebird DBA users called "sysdba". When you first install Firebird, it will create the sysdba user and give it a random password. This password is stored in the /opt/firebird directory in a file called SYSDBA.password. Copy the value from there for ISC_PASSWD into this field.
- Username: Enter the username you picked for non-DBA access.
- Password: Enter the password you gave to the non-DBA user.
- Database Directory: This is the directory where Quasar will create Firebird databases. It defaults to /opt/quasar/databases which should be all you need. If you use a different directory, you will need to get the permissions right so that the Firebird database server can write to it.
- Block Size: This is the block size to use for new databases. The default of 4096 should be fine.
- Character Set: This is the character set to use for new companies. It needs to be a character set than is supported by both Firebird and Qt.

With these setup you should be able to use quasar_setup to create a Firebird company. If you can't, you should try the following:

1. Check that the firebird server is running using this command:

   ```
   telnet <address> 3050
   ```

   Replace <address> with localhost if running on the server or with the address of the server computer if running telnet on the client. You should get "Connected to <address>" as part of the messages and you can then use Control-] to get a "telnet>" prompt and then "quit" to exit. If you get "Connection refused" or some other error then Firebird is not setup to run properly. You should try this telnet command on both the server and the client computer so you can verify the network connection is not a problem.

2. If the above worked then try to connect to security.fdb like this:

   ```
   export FIREBIRD=/opt/firebird
   PATH=$FIREBIRD/bin:$PATH
   isql -u sysdba -p xxx localhost:/opt/firebird/security.fdb
   ```

   Replace the “xxx” with the sysdba password from the SYSDBA.password file and then this should connect you to the “security.fdb” database and you should get a “SQL> ” prompt. If you get any errors then there is again something wrong with the Firebird setup.

3. Finally to test creating a database use this command at the “SQL>” prompt:

   ```
   create database 'localhost:/opt/quasar/databases/foo.fdb'
   user 'sysdba' password 'xxx';
   ```

   The above should all be typed on one line and again replace the ’xxx’ with the actual sysdba password from SYSDBA.password. You should just get back to the “SQL>” prompt which means the database create worked and since this is basically what quasar_setup does, it should also be able to create a company database.

Common problems with Firebird can be that the “sysdba” user’s password is not entered properly in firebird.cfg or that the wrong version of Firebird has been installed (for instance not using the NPTL version on newer Linux distributions). There can also be problems if you are trying to connect from a client over a network that has a firewall that is not allowing port 3050 traffic through or if the permissions on the directory you are trying to create a company database are not allowing the firebird server to create new files. This happens when you are running Quasar from your home directory and you will need to make sure that there is write access to the ~/Quasar/databases directory.

If you are having any problems with Firebird, use the “Test” button in the configuration screen. It will try to verify that all your configuration options work and will give you messages suggestion what may be wrong if something fails. Once you’ve created a company in quasar_setup, access from the client should work unless you’ve made a mistake with the non-DBA username and password.

5.3 Sybase Setup

The Sybase database driver for Quasar Accounting is for the Sybase Adaptive Server Anywhere (ASA) product which comes as part of the Sybase SQL Studio suite.

All Sybase databases first start out with a user called "dba" and the "dba" user has a default password of "sql". There is also a special utility database which is connected to for operations like creating or deleting a database. The password for the utility database is stored in the util_db.ini file found in /opt/sybase/SYBSsa9/bin (at least with Sybase version 9).

The Sybase database server program is called dbsrv9 or dbeng9. The dbsrv9 version is the full server and the dbeng9 is the personal embedded database server. They can be started either at system startup time in one of the startup scripts or it can be automatically started the first time there is a database request. By default Quasar assumes the automatic startup method which is done by passing a command line to use to start the database server if its not already running.

Once you've installed Sybase, you will need to setup the sybase.cfg file for Quasar to work with Sybase. The config file is setup from quasar_setup in the Drivers tab. Select the Sybase driver and click on the Configure button to bring up the Sybase driver configuration screen. The fields in the screen are as follows:

- Connection Params: These are the connection parameters to connect to the database (and also to start the database server if required). If you already had a server running on your network, this could be as simple as "eng=Quasar" assuming you named the Sybase server you started up "Quasar". A more complicated example is when you want to automatically start a database server the first time a connection is made. For this you need to specify the "start" option and give it the command to run the database server. There are many other options you can use to do things like encrypt the communications channel or control the types of network connections it will try to use or to specify an exact hostname of the computer running the Sybase server but you will need to read the Sybase documentation to figure out all the options.
- Library: The library is the full file path of the libdbodbc9_r.so library supplied by Sybase for clients to access the database.

- DBA Password: This is the password for the normal Sybase DBA users called "dba". Since Sybase stores the password in the database file rather than in a global location, this password can be anything you want since when the company is first created it will have a dba user with a password of "sql" and then the Quasar driver will change it to match this configuration parameter.
- Utility Password: This is the password for the "dba" user in the utility database that Sybase uses for things like creating and deleting databases. This password is stored in the util_db.ini file in /opt/sybase/SYBSsa9/bin and this configuration parameter must match what is specified in the util_db.ini file.
- Username: Enter the username you picked for non-DBA access.
- Password: Enter the password you gave to the non-DBA user.
- Database Directory: This is the directory where Quasar will create Sybase databases. It defaults to /opt/quasar/databases which should be all you need.
- Block Size: This is the block size to use for new databases. The default of 4096 should be fine.
- Collation: This is the collation to use for new companies. If left blank, Sybase will pick the best one for you using your current system locale to decide.
- Include Java Support? Check this is you want to use the Java facilities in your databases. They are not used or needed by Quasar and they make your databases quite a bit larger so only check this if you need it.

Another installation point with Sybase is that once its installed, you need to add the library directory from Sybase to the /etc/ld.so.conf file so that the library files are found by Quasar. This needs to be done as root and once the change is done you need to run "ldconfig" which will regenerate the "ld.so.cache" file with the new Sybase libraries.

If you are having any problems with Sybase the best way to test is to see if you can use Sybase outside of Quasar. First would be to try and manually start "dbsrv9" or "dbeng9" using the command line specified in the connection params in sybase.cfg. If this fails it may be because the Sybase libraries are not found or because there is another Sybase server on the network with the same name. The output when it fails should tell you what the problem is. If that works then leave it running and then try to connect to the utility database using these commands:

```
PATH=$PATH:/opt/sybase/SYBSsa9/bin
dbisqlc -c "uid=dba;pwd=xxx;dbn=utility_db;eng=Quasar"
```

Replace the "xxx" with the utilityPassword you specified in sybase_server.cfg and replace the "eng=Quasar" with a different name if you have changed the name of the database server. You should get connected and you can exit using "quit" and pressing F9. If this fails you likely have a problem with the utilityPassword between the Quasar config file and the Sybase util_db.ini file or else you've started up the "dbsrv9" program with command line parameters that aren't allowing utility_db database access.

Next you can try to create a database if you are testing on the server and creating the database from quasar_setup has been a problem. To do this just connect using dbisqlc to the utility_db database as in the last test and then type this command and press F9:

```
create database '/tmp/foo.db';
```

If this works then you can delete the database using:

```
drop database '/tmp/foo.db';
```

Finally if you were able to create a company database but you're having problems connecting to it you can use dbisqlc to connect to it. Use a command like this:

```
dbisqlc -c "uid=dba;pwd=xxx;dbf=/opt/quasar/companies/foo.db;
eng=Quasar"
```

This needs to be entered as one command line and the "xxx" needs to be replaced with the dbaPassword in the sybase_server.cfg file. Also you need to specify the real path to your database file instead of "foo.db" and if you are using a different engine name than "Quasar" then you need to specify that instead of "eng=Quasar". You should get connected and you can try a command like this:

```
    select * from company;
```

If it returns some data then you are connected and if you've got this far then Quasar should also be working since its connecting in basically the same way.

If you are still having any problems with Sybase, use the "Test" button in the configuration screen. It will try to verify that all your configuration options work and will give you messages suggestion what may be wrong if something fails. Once you've created a company in quasar_setup, access from the client should work unless you've made a mistake with the non-DBA username and password.

Something to watch out for though is that if you change the "DBA Password" in quasar_setup and you already have databases created, you will have problems. Since Sybase stores the passwords in the database and not in a global location, connections to any old companies will then fail since Quasar will be trying to use the wrong password. You should pick a good random password to start with and not change it unless absolutely required in which case you will need to connect to each existing database with the dbisqlc command and change their "dba" user's password to match the new value.

A final note is that the native database interface for Sybase is ODBC but that doesn't imply that Quasar uses the unixODBC package on Linux. Instead of that Quasar has its own database driver interface which is independant from unixODBC or any other driver library.

Chapter 6

Appendix

6.1 Trouble-shooting

1. **Trouble connecting to the server from the client:**

   You can manually test the connection to verify its not a Quasar problem. To do this you need to have a telnet client installed. Once you have this you need to enter in this command from a terminal window:

   ```
   telnet server 3599
   ```

   Replace server with the hostname or IP address of your server (which should be the same as you are using in the "Server:" field of the Quasar client login window). The 3599 is the port the Quasar server program runs on. The communications between the client and server are just basic text which is why you can use the telnet program to act like a client.

   If you get a connection you can try typing "version" and the press Enter. You should get back the version of Quasar running on the server. If this works then the Quasar client should also be able to connect to the server and get the list of companies.

   If you get a connection failed error then there is something wrong with your network setup. Perhaps a cable has become unplugged or has failed (test this by trying "ping server" to see if its responding at all). Another possible problem is that you have a firewall between the client and the server which is not allowing port 3599 traffic to go through in which case you need to disable the firewall or configure it to allow port 3599 traffic (as well as other ports though depending on which database type your are using). Another possible problem could be that there is a routing problem which will also show up as a failure to ping the server and can be diagnosed by looking at the "route" command output (assuming you know what it means!). Finally if you can ping the server but the telnet fails then perhaps Quasar is not installed on the server properly or else has been disabled.

If the version command worked, you can try "login company.xml name passwd" to login to a company. Replace company.xml with the filename of your companies definition file in /opt/quasar/data/companies and replace name and passwd with the username and password you want to login as. You should see "login: good" returned or an error that will tell you what is wrong.

2. **Using a terminal window:**

   If you are having problems connecting to a company or something in Quasar is failing, see if you can reproduce the problem after staring the Quasar client from a terminal window. Use what ever terminal window is provided by your Linux GUI (konsole for KDE, xterm in many cases, ...) and once you get a command prompt run Quasar like this:

   ```
   /opt/quasar/bin/quasar
   ```

   Sometimes when Quasar fails it will output some debugging information that will help is pin-point the problem and running Quasar this way will let you see the output which can then be sent to us with a bug report.

3. **Database driver problems**

   The first thing to try if you have any problems with your database setup is the "Test" button found in the configuration screen of the driver in quasar_setup. Using this will run many tests to try and pin-point exactly where your error is occuring and it will make suggestions on how to best fix it.

6.2 Server Programs

The normal setup for the Quasar server is to have xinetd listening to requests on port 3599 and have it start quasar_clientd to handle these connections. You can use xinetd to add limits on who can connect to Quasar specifying such things as IP address matching, time of day limits, and number of client limits. The quasar_clientd program will run as long as the client is connected so to shut things down for some types of maintenance you could edit the /etc/xinetd.d/quasar file and set it to disabled and then restart xinetd so it can see the change using /etc/init.d/xinetd restart. This will then stop new connections and you can check for existing connections using "ps ax—grep quasar_clientd". If any still exist you can either go to the client computers and stop the Quasar client running there or else from the server you could just use the command "killall quasar_clientd" which would kill all the clients (and cause the Quasar client on the workstations to fail). Finally at the end you may want to shutdown the database server if you are using Sybase using "dbstop".

Optionally if you don't want to use xinetd you could install Quasar to use its own server program called "quasard". This works just like xinetd and all it does is listen for connections to port 3599 and for each one fire up quasar_clientd passing the connection to it. You can use this way of running Quasar if you are testing running it out of the compile directory in your home directory or you could develop a script to start Quasar up this way and install it in /etc/init.d but the xinetd method is more powerful since it has some abilities to limit the connections and it does all that quasard does and more so why not use it?

A final server program is quasar_posd. This is used when you are using the Point-of-Sale program from Linux Canada Inc. with Quasar Accounting and it is used to post point-of-sale transactions. There should be only one of these processes running at once and the quasar_clientd process starts them as needed and quasar_posd will shutdown automatically when there are no clients left using it so you shouldn't normally have to worry at all about this program.

6.3 Data Files

Quasar stores many of the data files the clients need on the server and the quasar_clientd server sends these files to the clients as needed. Each client has a local cache of data so the normal process of things is that the client asks the server for a list of a certain type of data (reports for instance) to which is returned a list of all the report files available on the server. Then the client for each file would check if a file with the same name exists in the cache. If so then it would send a request to the server for the MD5 hash of the file and it would compare it against the file in the cache and if the hashes match then the cache file is considered to be identical to the one on the server and it can be used. If the file didn't exist in the cache or the hash was different indicating the file has been changed, then the a request is sent to the server for the actual file contents. If the client Qt version is new enough, it will request the data compressed and if the Qt version on the server is also new enough, the data will be sent compressed. After the optional compression the file is sent over the network in base64 encoding so that there is no problem with nulls in binary data files or CR/NL encoding being changed between Windows and Linux clients.

The cache file is found in the users home directory under .quasar/cache and will be created if it doesn't exist. The server only allows data to be sent from the "data" directory on the server which is /opt/quasar/data by default so there is no security risk in this ability being used to get at sensitive system files. It is simply a mechanism to centralize the data storage that Quasar uses so that when a new report, company, or other type of data is added, it doesn't have to be installed on every client computer.

6.4 User Management

Quasar supports a number of database types (three at the moment) and each one handles database users differently. With Sybase users are stored in the database so if you have a database user called "quasar" in two databases with a password of "quasar" and you then change the password of "quasar" in one of the databases, the other one will not change and you will now have different passwords for the "quasar" user in each database. With Firebird though the databases are global and stored outside of any given database so when you create a "quasar" user it not can potentially be used in any database and if you change the password, it will be changed to the new password regardless of which database you connect to. With PostgreSQL you can configure the user authentication in many different ways in the pg_hba.conf file.

With all these differences, Quasar is designed to handle its users outside of the users that the databases provide. This can cause some confusion if you don't understand the process since if you can login to a Sybase company through Quasar using "admin" as a user name then you might suspect you could also connect to the Sybase database using third party tools using "admin" but this is incorrect.

Each Quasar company database has a table called "users". This table has the user's login name, password (stored as a SHA1 hash so its secure), and some other information like the screen they see and the security type they've been set to use. When a user types in their username and password at the Quasar login screen these are sent to the Quasar server along with the company the user wants to connect to. The Quasar server has the knowledge it needs to connect to the database for the company since the various driver .cfg files for each database type specify the username and password of a user powerful enough to create new databases and connect to any database. The Quasar server will validate that the user name used to login exists in the company database and that the SHA1 hash of the password the user gave matches the one in the company database. If this is all valid then the Quasar server will remain connected to the database so it can process SQL commands sent from the client. The username and password that Quasar uses to connect to the database normally will not have the ability to update, insert, or delete from the users table in the database so they will not be able to change other users passwords.

Quasar does allow user administration from the Quasar client which the above explaination should make impossible. How this is done is by sending the requests to the Quasar server. So when a new user is added in the Quasar client, the request is sent to the Quasar server where it will verify that the user trying to add another user has the permissions in their security type to do this and if so it will do the work for them using the more powerful database username and password which are only known by the server.