Subversion for OS400

This document covers the initial setup and configuration of a Subversion server on OS/400. This document does not go into a lot of details about Subversion itself or how to use it. The features and usage of Subversion on OS/400 are the same as Subversion on any other operating system. One notable difference is that at the time of this writing only the server aspects of Subversion have been ported to OS/400. To access the server you must use a Subversion client on Windows, Linux, BSD, UNIX, OS X etc.

For more information about Subversion, please refer to the free book "Version Control with Subversion" which you can find at this location:

http://svnbook.red-bean.com/

This and other books about Subversion are also available for purchase at bookstores everywhere.

http://www.amazon.com/s/ref=nb_ss_b/104-5352130-0540750?url=search-alias%3Dstripbooks&field-keywords=subversion

At the time of this writing, the port is based on the Subversion 1.4.x branch.

Goals

The goal of this project was to bring a free version control solution to iSeries customers, particularly those that are using WDSC and are looking for version control for their Java/Web artifacts. WDSC comes with an excellent CVS client, but you cannot run a CVS server on OS/400. This requires the user to setup a Linux/Unix server on their network for version control.

Subversion also offers excellent integration with Eclipse, via the Subclipse plug-in, and many people consider Subversion to be a vastly superior version control solution to CVS. In fact, Subversion won the 2005 Jolt Product Excellence award for the best product for Change and Configuration Management.

Subversion is of course both a client and a server; however for the purposes of this port, the primary goal was to enable the Subversion server to run as a native OS/400 application. It is assumed that the client will most often be a Windows or Linux desktop.

As of the 1.3.0 release, there is now support for the Subversion Client on OS/400 as well. However, only the file:// and svn:// protocols are supported at this time. The main goal in porting the client is to provide more opportunity for writing server-side scripts that work with the repository. We do not anticipate the client being commonly used from a 5250 session to access a Subversion repository on another platform, although that would work if the svn:// protocol was used.

Therefore, at the time of this release, these are the areas of Subversion that have been ported to OS/400:

svnadmin - The administration module. This module is used for creating and managing Subversion repositories.

svnserve – The proprietary, lightweight, Subversion server. This server implements the svn:// protocol.

mod_dav_svn/mod_authz_svn – The Apache WebDAV module. This allows you to run a Subversion server inside the Apache HTTP server. This lets you leverage Apache features such as SSL and the various authentication mechanisms that Apache supports.

svnlook – A Subversion program commonly used in server hook script to read info from the repository.

svn – The Subversion client. Currently only the file:// and svn:// protocols are supported.

It should be noted that Subversion currently supports two repository formats: native filesystem, and Berkeley DB (BDB). We do not intend to support BDB on OS/400 now or in the future. The native filesystem backend has become the repository of choice and has been the default in Subversion since 1.2. On OS/400, native filesystem means that it resides in the IFS as stream files.

Finally, this port is primarily aimed at providing version control support for Java/Web/PC artifacts. It is not meant to be a solution for "OS/400 version control" such as for your RPG/CL/COBOL. If you want to store the source for your OS/400 application in stream files and store them in Subversion you are free to do so, and it will work fine. However, the problem with this scenario is that the rest of the tooling you need to make this work does not really exist. It is possible that will change over time with future enhancements to WDSC, but today it is not really a viable solution for most people.

Requirements

Subversion for OS/400 requires V5R4 or higher of OS/400.

NOTE: As of V5R4, a different version of Subversion for OS/400 is required and will be available as a separate download. IBM changed Apache in V5R4 to be native UTF8 instead of native EBCDIC. This requires a different port of Subversion, however, that port is actually much simpler, and allows us to just use the standard Subversion source code plus a few modifications that we contributed back to Subversion and is now in the trunk repository.

NOTE: Due to changes in the Apache API for V6R1, a different version of Subversion for OS/400 is required and will be available as a separate download.

For maximum portability, Subversion is written in C and uses the Apache Portable Runtime (APR) to gain platform portability. APR is a low-level C library that was written by the Apache Software Foundation to facilitate the port of the Apache HTTP Server and modules to different operating systems. Since Apache is ported to OS/400 and supported by IBM, OS/400 includes APR in the form of *SRVPGM objects that reside in library QHTTPSVR. Subversion links to these service programs and uses the APR functions contained within to do all of its work.

During the porting process we encountered some defects in APR which IBM fixed and created a PTF for. You should not attempt to use Subversion without applying these fixes first. You can obtain the PTF by applying the latest HTTP Group PTF.

V5R4M0: SF99114 - HTTP Group PTF Level 4 or higher

***** IMPORTANT *** - Subversion will not function correctly without the PTF, so please be sure it is applied before your proceed. This is true even if you do not plan on using the HTTP Server!**

In addition to licensed program 5722DG1 (Apache HTTP Server), you also need to have licensed program 5722SS1 Option 30 (QShell) installed. While Subversion itself does not require QShell, the OS/400 command wrappers that are provided for Subversion do make use of QShell API's to perform certain requests.

Installation

While it is possible to build Subversion on OS/400 from source, it is assumed that most people will want to just use the version that has already been compiled by SoftLanding Systems. Actually, even if you want to build from source you still need to install this version first so that you can get the OS/400-specific build system that was created for Subversion.

Subversion is delivered in the form of a save file. As a convention, the name of the save file will be the name of the library saved within it, although you can use the DSPSAVF command to verify this. The save file will likely come to you in a zip file that you need to unzip and FTP into a save file on your iSeries server. Once you have the save file in a library on OS/400, run the following command to restore Subversion.

```
RSTLIB SAVLIB(SVN14PRD2) DEV(*SAVF) SAVF(SVN14PRD2) RSTLIB(SUBVERSION)
```

In this example, we are restoring to a library named SUBVERSION. You are free to choose any library name you desire.

Congratulations! Subversion is now installed. The next step is to configure it to run on your server.

Upgrade

If you have an older version of Subversion already installed, to upgrade you just want to stop any servers that may be running, and then follow the same instructions as a first-time installation. In other words, just restore right over the top of the existing installation.

Instead of following the steps on the next page for Initial Configuration, just run these two commands:

```
ADDLIBLE SUBVERSION
```

```
BNDSVNPGM
```

The second command will run fairly quickly on most systems. When it completes, just restart your servers.

NOTE: If you are using the Apache server, please refer to that section of this document for some important changes you need to make to your Apache configuration file before you restart your server.

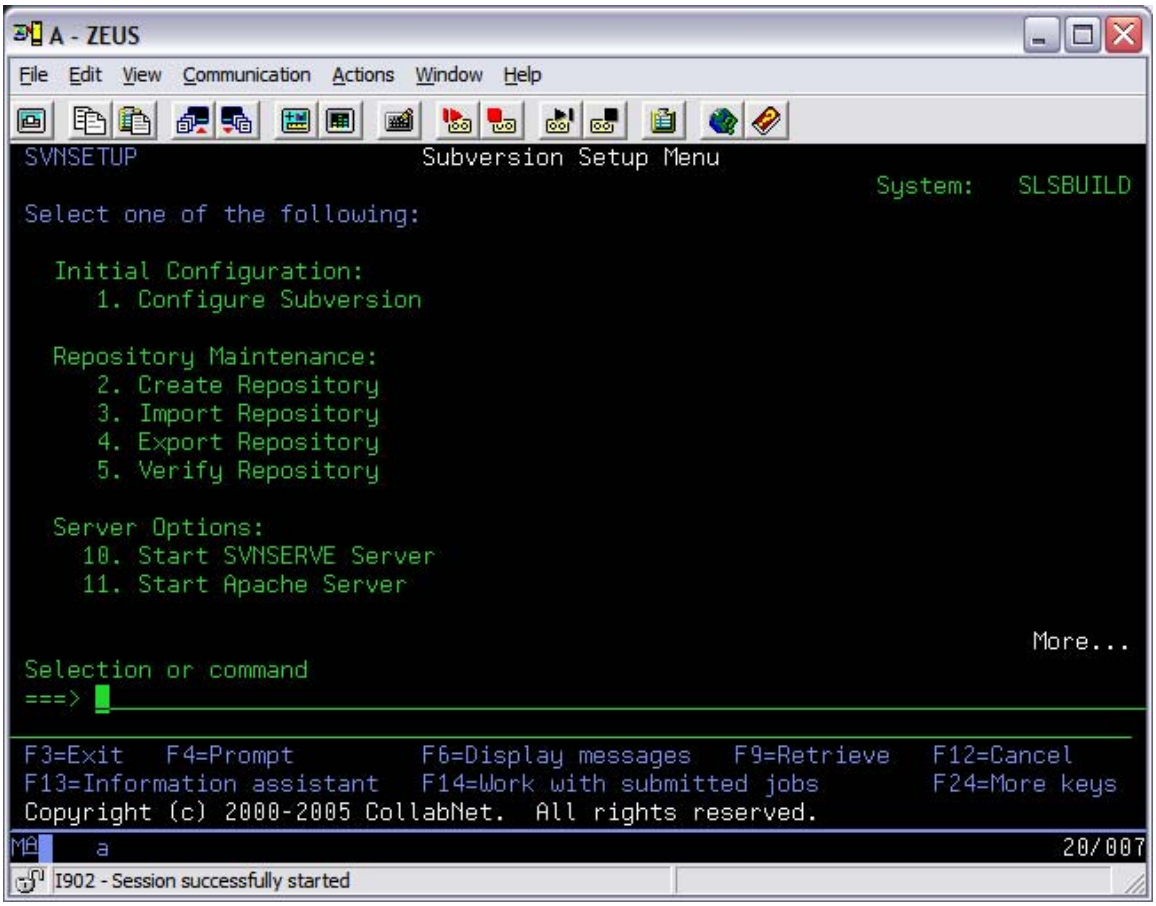
Initial Configuration

From an iSeries 5250 session, add the library you restored Subversion to, to your library list.

```
ADDLIBLE SUBVERSION
```

Then type the following command to bring up the Subversion Setup menu.

```
GO SVNSETUP
```



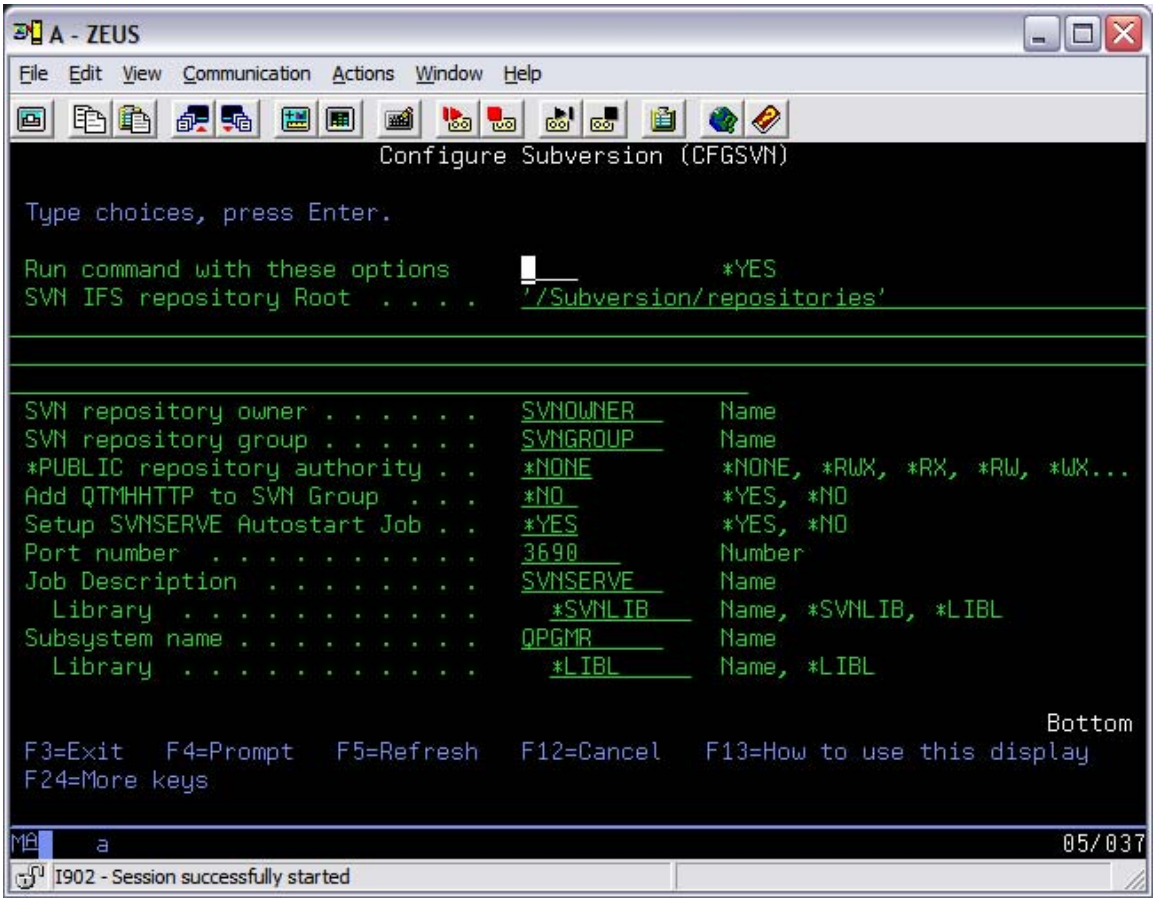
There is nothing special about this menu, it is just provided so that you have a catalog of the various commands you can run that are related to Subversion. This menu and each of the commands that are provided include extensive online help that describes what they do and what requirements they have to run them.

Before you do anything with Subversion, you must run option 1, the CFGSVN command. The purpose of this command is to create a user and group profile for Subversion, an IFS location for your repositories and also setup the default permissions on the IFS location. The number one problem when using Subversion on any platform is permissions on the repository. We have tried to make this as easy as possible for you to get right.

Rather than make any of these commands adopt QSECOFR authority, we require that you are signed on as a user with *ALLOBJ and *SECADM special authority to run this command. This is so that you have the authority you need to create the profiles and set the permissions in the IFS.

The profiles that Subversion creates are just standard users with no password and with no special authorities.

This is what you see when you take option 1.



Press HELP on the command if you have questions about the various parameters. Again, the basic idea of the command is to create an IFS location and a user and group profile to own that location. When you run a Subversion server process, you just need to make sure you run it as either the user that is the owner of the repository, or is a member of the group profile.

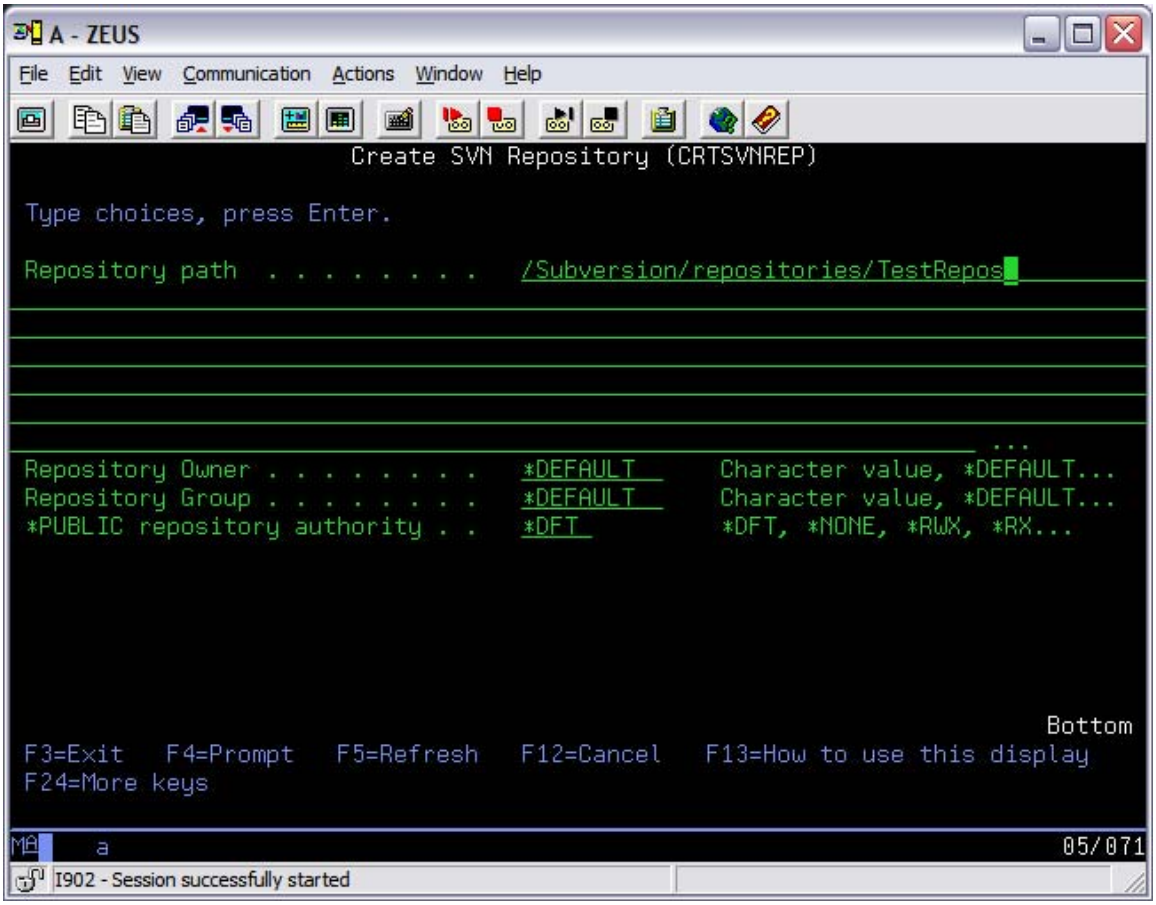
There are two server options for Subversion, a lightweight proprietary server (SVNSERVE) and a WebDAV provider that runs inside the Apache HTTP Server. When using the former, you would typically run the server as the owner of the repository, when using the latter; we recommend that you change the Apache user profile to be a member of the group you create for your repository.

The other important step this command does is that it runs the BNDSVNPGM command which re-binds all of the Subversion programs and service programs to the library you installed it in and

for your version of the Apache APR *SRVPGM objects. When you install new versions of Subversion in the future, you will just need to run the BNDSVNPGM command to re-do this important step.

Create a Repository

Take the option to create a repository, it prompts the CRTSVNREP command.

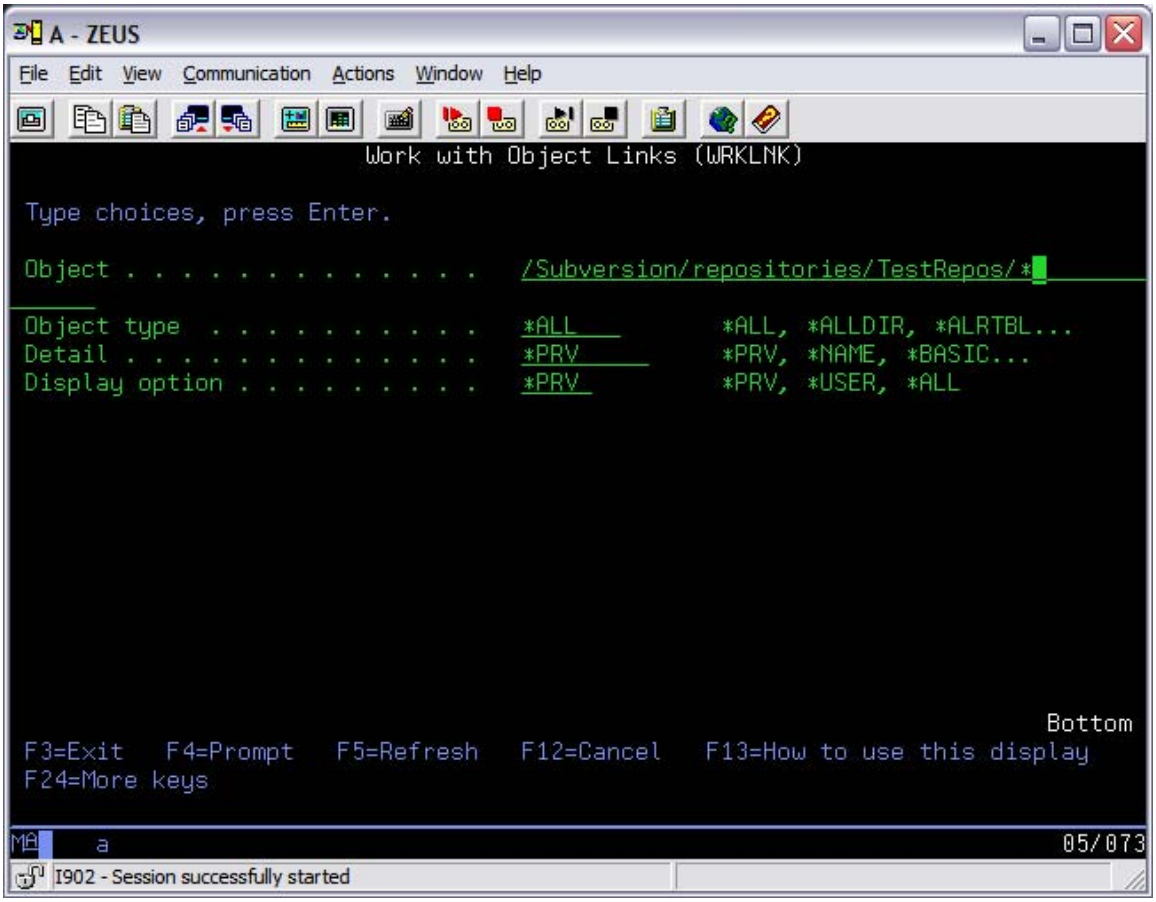


Specify a repository path that lives within the path you specified in the CFGSVN command. That way the repository will get all of the right permissions. Also, if you put all of your repositories under one folder, you can serve them all with a single server instance.

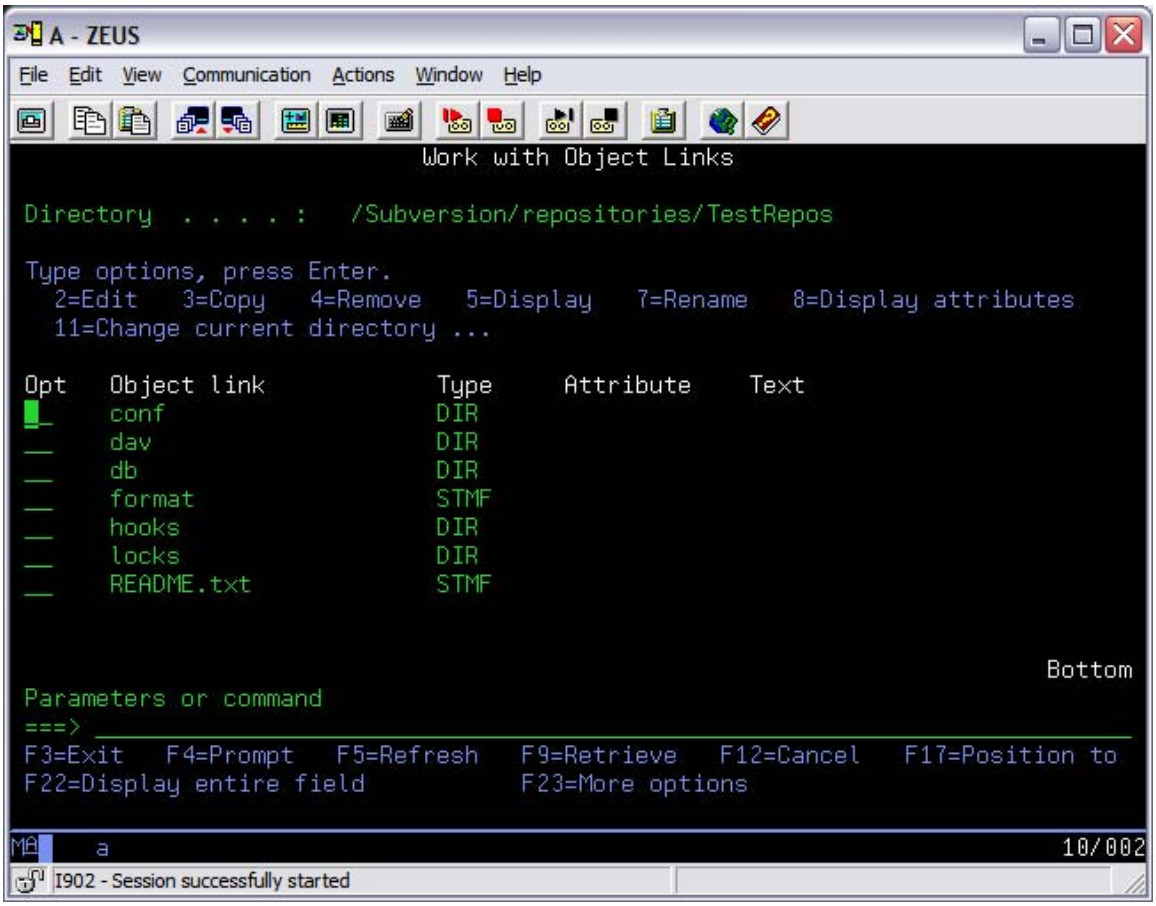
Configure the Repository for SVNSERVE

If you are going to use Apache to serve your repository, you can skip to the section on configuring an Apache instance. If you plan on using the SVNSERVE server option (or both) then there is some configuration to perform within the repository to configure the authorities and users for SVNSERVE.

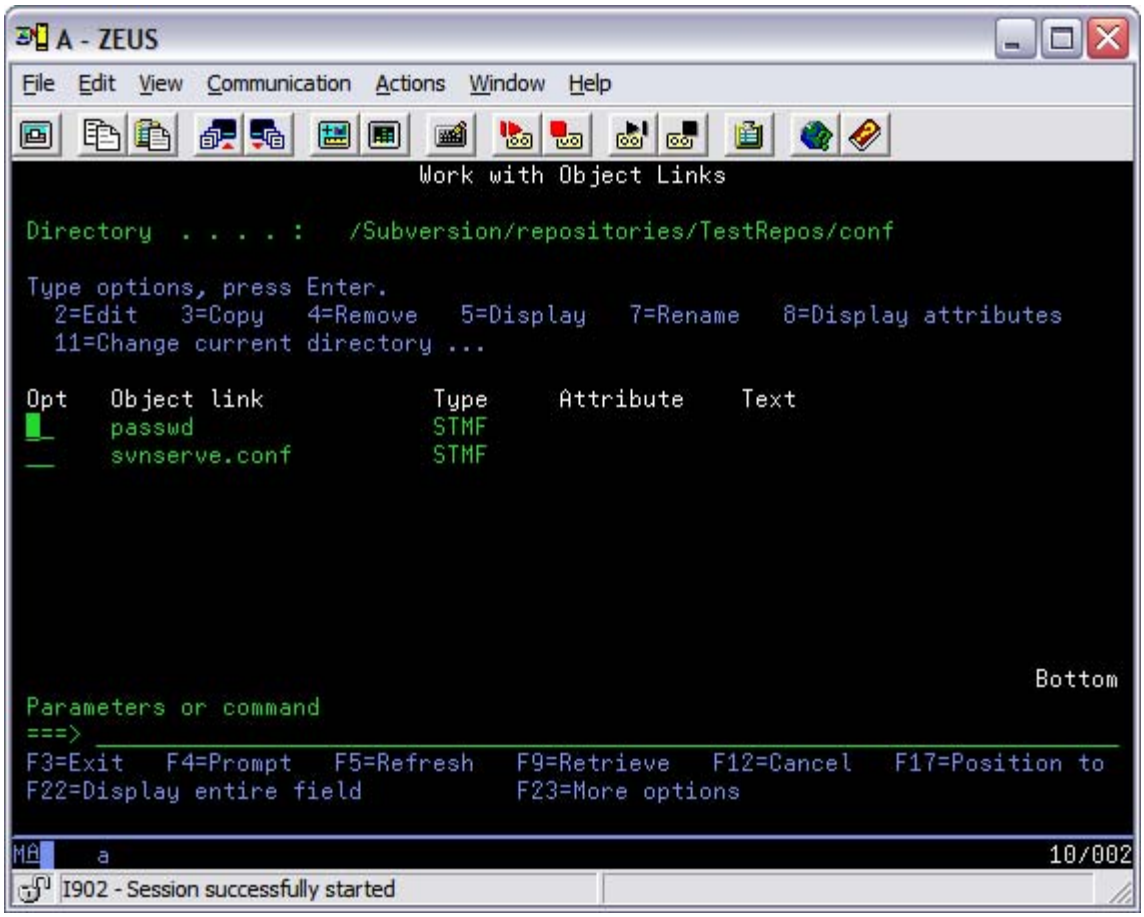
Use the WRKLNK command to navigate to the folder where you created your repository.



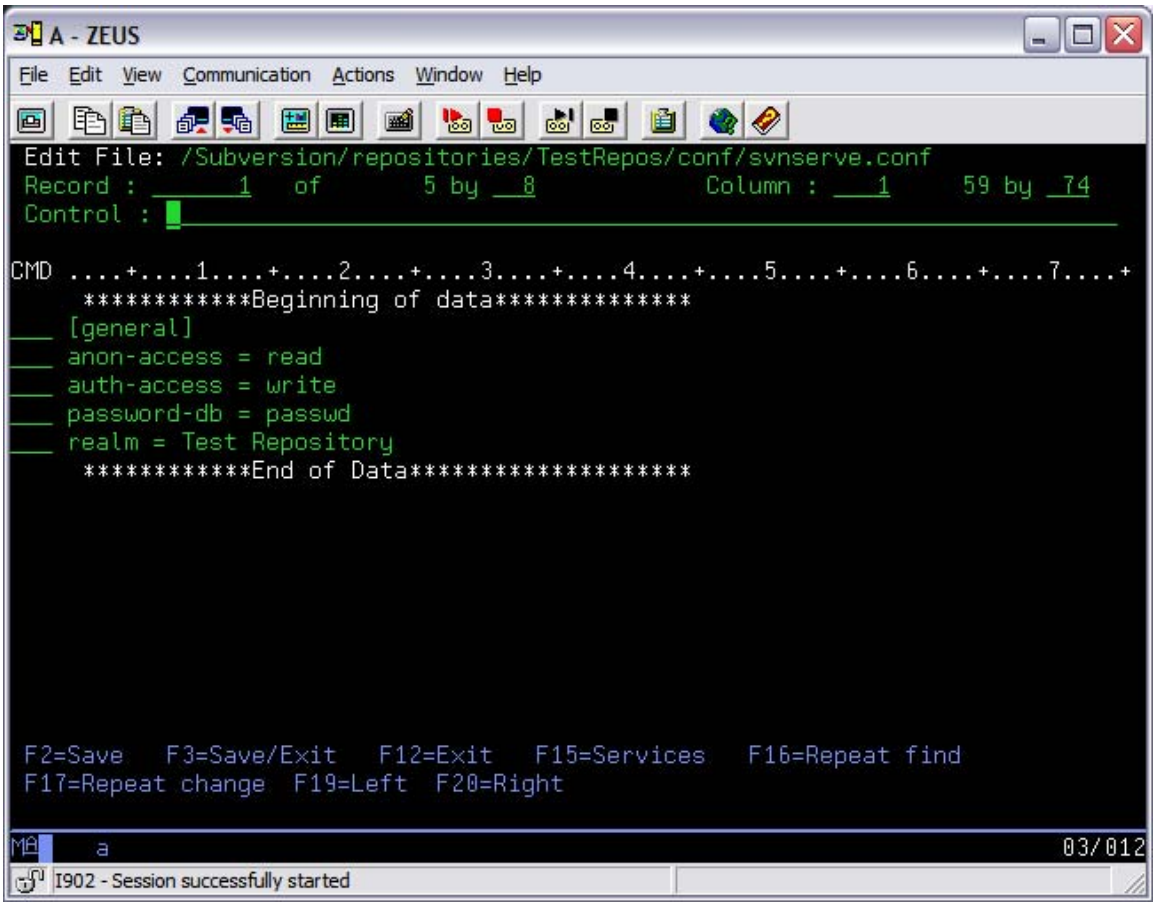
You should see something like this:



Use option 5 to drill-down to the “conf” folder. This is where the SVNSERVE configuration is stored. You should see this:

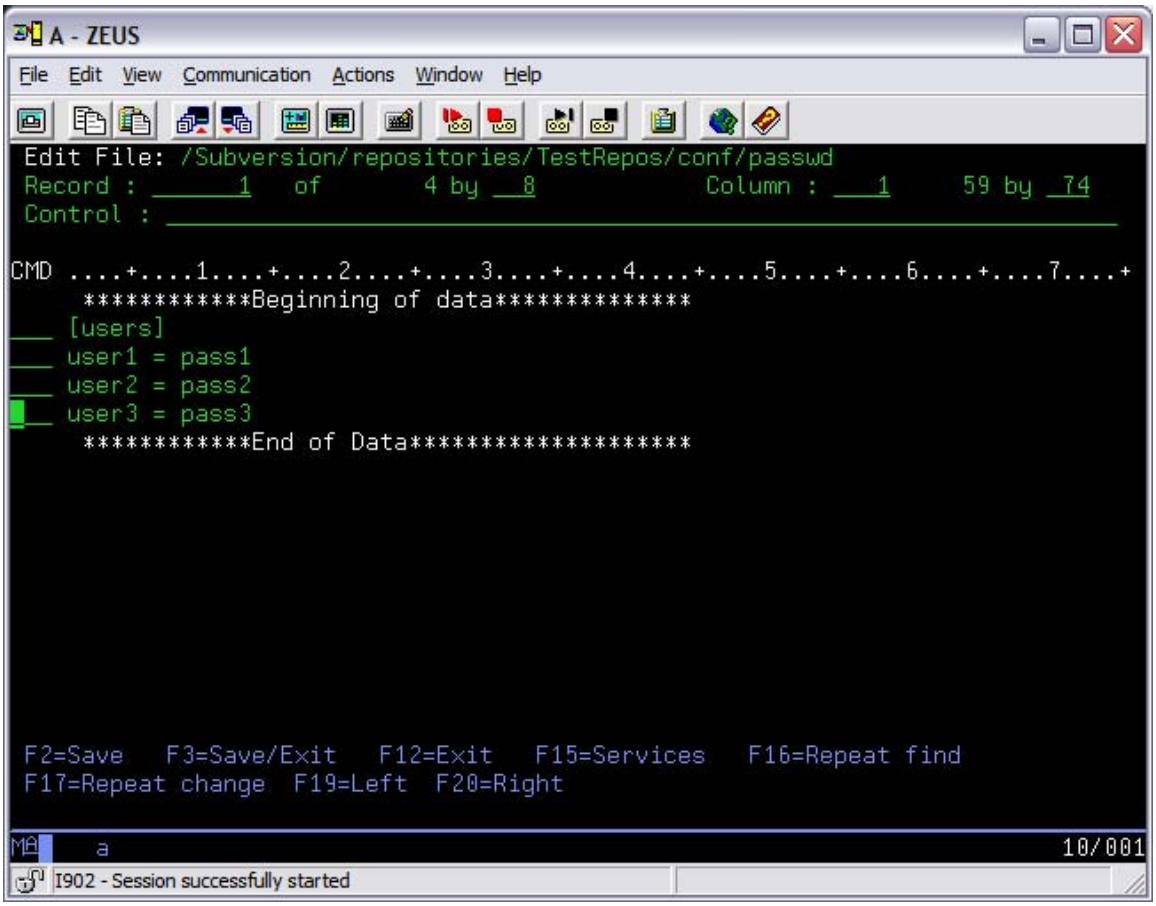


The first thing we want to do is edit our svnserve.conf file. Use option 2 to open the file in the EDTF editor.



There are a lot of comments in the file that explain how it is organized. Edit the file so that it looks something like this. As a default, Subversion creates a password-db file named passwd in the same folder as the configuration file. If you do not want to name your password-db file passwd, then specify the correct name here. If you do want to store it in the same folder as svnserve.conf, then specify the complete path to the file. By specifying the complete path to the file, you can have one master password-db that you point all of your configurations to.

The next step is to edit the password-db file to have some users and passwords.



Yes, it is all clear text, but *PUBLIC should be restricted from seeing the file. The point of SVNSERVE is not to provide high security, it is to be lightweight and easy to configure. The user profile feature primarily exists so that there is a mechanism to uniquely identify each user for the purpose of tracking who did what. If you want to use your iSeries profiles, or a Windows Active Directory, then use the Apache server option. Then you can use anything that Apache supports for authentication, including digital certificates, Kerberos etc.

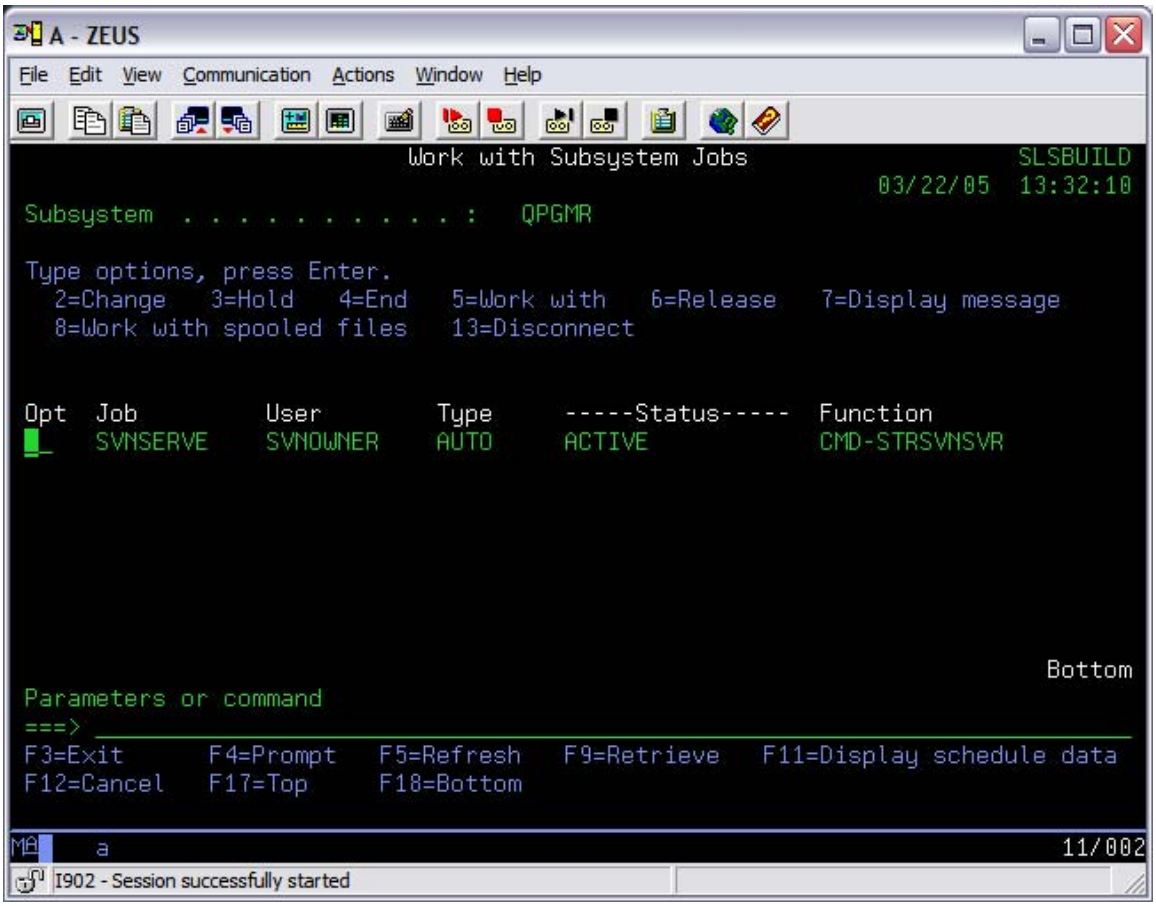
Start SVNSERVE Server

OK, we have a repository and we have configured it for SVNSERVE. All we need to do now is start the SVNSERVE server. When you ran the CFGSVN command you specified a subsystem to put the autostart job in. Just end and start the subsystem to start the server. In our example:

```
STRSBS QPGMR
```

Then run the following to verify that the server is started.

```
WRKSBSJOB QPGMR
```

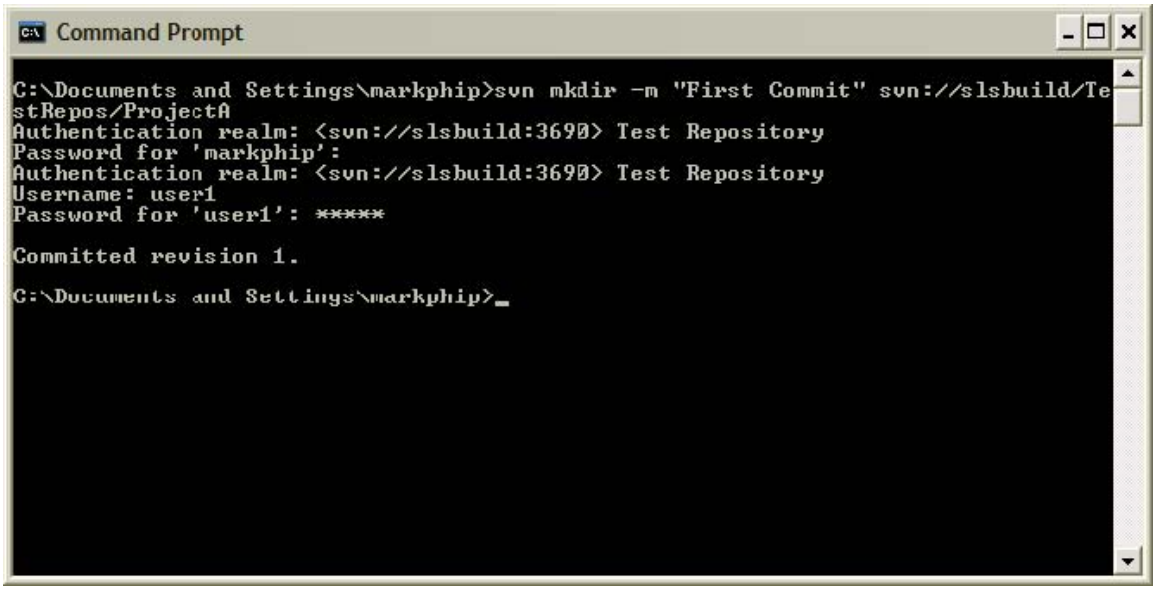

A - ZEUS
File Edit View Communication Actions Window Help
Work with Subsystem Jobs
SLSBUILD
03/22/05 13:32:10
Subsystem . . . . . : QPGMR
Type options, press Enter.
2=Change 3=Hold 4=End 5=Work with 6=Release 7=Display message
8=Work with spooled files 13=Disconnect
Opt Job User Type -----Status----- Function
SVNSERVE SVNOWNER AUTO ACTIVE CMD-STRSVNSVR
Bottom
Parameters or command
===>
F3=Exit F4=Prompt F5=Refresh F9=Retrieve F11=Display schedule data
F12=Cancel F17=Top F18=Bottom
MA a 11/002
I902 - Session successfully started

Test the Server

You have to use a Subversion client to test the server. In the following example, I use the command line client on Windows to run a command to make a directory directly in the server repository. This is the command:

```
svn mkdir –m “First Commit” svn://hostname:port/ReposName/FolderName
```

Substitute your hostname, port number and repository name as shown (if you use port 3690 you can leave off the port number from the URL:



Notice that it prompted me to login before committing.

Congratulations, you now have a working server and repository.

Configuring an Apache Server Instance

If you are going to use the Apache server, be sure that the Apache user profile, QTMHHTTP was modified to be a member of the Subversion Group profile.

Create an Apache Server Instance

Use the web-based iSeries Administrator to create a new Apache instance. You can skip this step if you want to add it to an existing instance.

Edit the Apache Configuration File

You can do this in the web-based Admin console, or directly in the IFS. Most, but not all, of this configuration can be defined using the GUI, but it is easier to just show the configuration file and let you either edit it or for those that know the related areas in the GUI, use the GUI.

Following are some example configurations you can use to get the general idea, combined with the Subversion book; this should be enough to setup any possible configuration. Keep in mind that one of the primary reasons to use Apache is that you get all of the power and options that it provides. So while we do not show an example of using SSL or Kerberos, or any of a number of options. Those are all just Apache features and you are free to take advantage of any or all of those options in your setup.

Sample One – Basic Configuration/iSeries Profile Authentication:

Here is a sample Apache configuration file. Additions that are relevant to Subversion are highlighted in Yellow.

```
LoadModule dav_module /QSYS.LIB/QHTTPSVR.LIB/QZSRDAV.SRVPGM
# Subversion module is always needed
LoadModule dav_svn_module /QSYS.LIB/SUBVERSION.LIB/MOD_DAVSVN.SRVPGM
Listen *:8888
DocumentRoot /www/svnbasic/htdocs
Options -ExecCGI -FollowSymLinks -SymLinksIfOwnerMatch -Includes -
IncludesNoExec -Indexes -MultiViews
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%{Cookie}n \"%r\" %t" cookie
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referrer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined
LogMaint logs/access_log 7 0
LogMaint logs/error_log 7 0
SetEnvIf "User-Agent" "Mozilla/2" nokeepalive
SetEnvIf "User-Agent" "JDK/1\.0" force-response-1.0
SetEnvIf "User-Agent" "Java/1\.0" force-response-1.0
SetEnvIf "User-Agent" "RealPlayer 4\.0" force-response-1.0
SetEnvIf "User-Agent" "MSIE 4\.0b2;" nokeepalive
SetEnvIf "User-Agent" "MSIE 4\.0b2;" force-response-1.0
AddType application/xml .xsl
<Directory />
      Order Deny,Allow
      Deny From all
</Directory>
<Directory /www/svnbasic/htdocs>
      Order Allow,Deny
      Allow From all
</Directory>
<Location /repos>
      Order Allow,Deny
      Allow From all
      Dav svn
      SVNParentPath /Subversion/repositories
      SVNPathAuthz off
      Require valid-user
      PasswdFile %%SYSTEM%%
      AuthType Basic
      AuthName Subversion Repository
</Location>
```

This would be accessed with a URL like this:

http://hostname:8888/repos/ReposName/

The top part of the configuration file causes the Apache modules to be loaded. They need to be specified in the order shown. The first module is IBM's mod_dav which provides the core WebDAV protocol support. The second module is Subversion's WebDAV provider, mod_dav_svn.

The only other thing you need to add is a location directive. The example uses a value of /repos, but you can use anything you want. Any URL that starts with /repos will be routed and handled by Subversion. The first value after /repos would be the name of your repository.

Sample Two – Advanced Configuration/LDAP Authentication

This example adds a few new wrinkles. It adds a second Subversion module, mod_authz_svn, which adds fine-grained permissions to the repository. In the first example, if you got past the Apache authentication, then you had read/write permission to the repository. In this example, we will be able to fine-tune the access you have to the repository.

```
LoadModule ibm_ldap_module /QSYS.LIB/QHTTPSVR.LIB/QZSRVLDAP.SRVPGM
LoadModule dav_module /QSYS.LIB/QHTTPSVR.LIB/QZSRDAV.SRVPGM
# Subversion modules are always needed
LoadModule dav_svn_module /QSYS.LIB/SUBVERSION.LIB/MOD_DAVSVN.SRVPGM
LoadModule authz_svn_module /QSYS.LIB/SUBVERSION.LIB/MOD_AUTSVN.SRVPGM
Listen *:8888
DocumentRoot /www/svnbasic/htdocs
Options -ExecCGI -FollowSymLinks -SymLinksIfOwnerMatch -Includes -
IncludesNoExec -Indexes -MultiViews
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
Agent}i\"" combined
LogFormat "%{Cookie}n \"%r\" %t" cookie
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referrer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined
LogMaint logs/access_log 7 0
LogMaint logs/error_log 7 0
SetEnvIf "User-Agent" "Mozilla/2" nokeepalive
SetEnvIf "User-Agent" "JDK/1\.0" force-response-1.0
SetEnvIf "User-Agent" "Java/1\.0" force-response-1.0
SetEnvIf "User-Agent" "RealPlayer 4\.0" force-response-1.0
SetEnvIf "User-Agent" "MSIE 4\.0b2;" nokeepalive
SetEnvIf "User-Agent" "MSIE 4\.0b2;" force-response-1.0
AddType application/xml .xsl
<Directory />
      Order Deny,Allow
      Deny From all
</Directory>
<Directory /www/svnbasic/htdocs>
      Order Allow,Deny
      Allow From all
</Directory>
<Location /repos>
      Order Allow,Deny
      Allow From all
      Dav svn
      SVNParentPath /Subversion/repositories
      Require valid-user
      PasswdFile %%LDAP%%
      AuthType Basic
      AuthName Subversion Repository
      LDAPConfigFile /www/sls_ad.ldap
      AuthzSVNAccessFile /Subversion/repositories/auth.conf
</Location>
```

Most of the changes in this example are just to add the LDAP configuration and this is just standard Apache stuff, not specific to Subversion. However, note that we also loaded a new module for the mod_authz_svn support and added a line to our location directive to point to a configuration file (be sure that the file has an ASCII (819) or UTF-8 (1208) CCSID).

The way that this module works, and the format of this file are all explained in the Subversion book. Just for the basic idea, here is a sample file:

```
[groups]
java-developers = ed, ray, paul
web-designers = susan, tim, george
developers = ed, ray, paul, susan
managers = mark, mary, joe
build = mark, steve

[web:/]
@web-designers = rw
@developers = r
@managers = r
@build = r

[web:/tags]
@web-designers = r
@developers = r
@build = rw

[plugins:/]
@java-developers = rw
@developers = r
mark = r

[plugins:/tags]
@java-developers = r
@developers = r
mark = rw

[dev:/]
@developers = rw
* = r
```

Again, refer to the Subversion book for a detailed explanation, but here is a quick overview:

The first part of the file lets you define [groups]. The username comes from what the user authenticates to Apache with, but there is no way in Apache to access any underlying groups, such as in an LDAP directory. So you have to make your own groups in this file.

You then just have sections for repository locations. They take the format:

```
[repos_name:/path]
username1 =
username2  r
username3 = rw
@groupname = rw
```

You can have as many of these as you need to define all of the permissions you want.

NOTE: Username matching is case-sensitive and if you use the iSeries user profiles, your usernames will be all UPPERCASE.

Start the Apache Server

Once you have your configuration in place, you should be able to just start your server instance.

```
STRTCPSVR SERVER(*HTTP) HTTPSVR(instance_name)
```

One nicety of the Apache server is that you get some basic web browsing of your repository for free. Try opening your web browser to one of your repository URL's:

Getting Help

The Subversion community is very active and helpful. Generally, everything is managed with mailing lists, which you can subscribe to here:

http://subversion.tigris.org/servlets/ProjectMailingListList

For general usage questions, you should post to the users@ list. That will get your question in front of the most people. There are good searchable archives available here:

http://svn.haxx.se/

For questions specific to the OS/400 port, there is an os400@ mailing list. This would be the best place to ask questions that are likely to be specific to OS/400 such as installing or configuring. We will also use this list to discuss development activities and receive patches.

Thanks. Please join the os400@ mailing list and let us know what you think!

Building from Source

Subversion is an open source application, so some people may want to build it from the source code. Even better is that some people may want to look at the code and possibly contribute to the port. In either case, we have made building Subversion on OS/400 from its source code a very easy task. All you need is to have the ILE C Compiler installed.

NOTE: You still need to install the binary distribution first as it includes the "SVNMAKE" command that we wrote that builds Subversion from source. All of the source code for the "OS/400 commands" we provide is included in source files in the binary distribution.

Download the Source

As of Subversion 1.2, the source code for the port is available in a zip file on the same page as the download of the binary. Download the zip file and unzip the source code into the IFS of your OS/400 server. For the sake of this documentation, we will assume you unzipped it directly in the root folder. This would result in a path of something like:

/subversion-1.4.0_os400 _v6r1

The source for Subversion is in a sub-folder named "subversion".

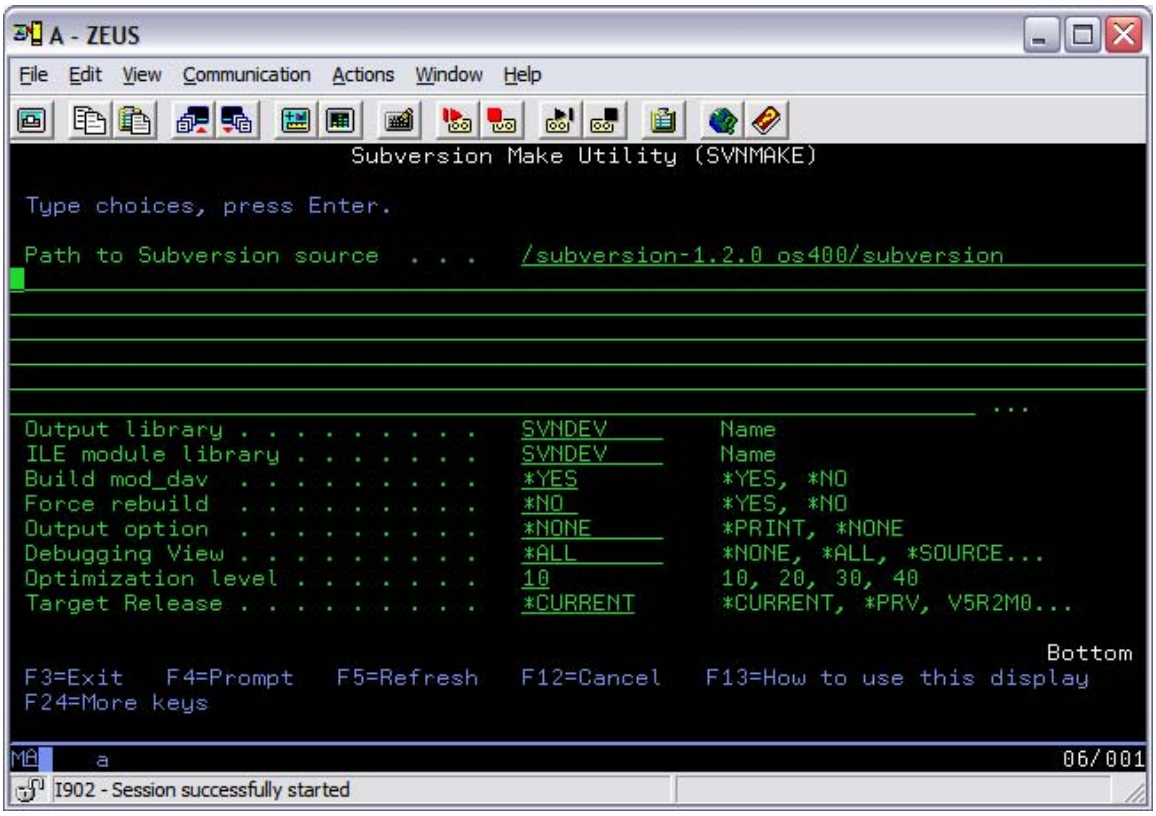
Checkout the Source

Since the source code for the port is stored in the Subversion repository, you could also use a Subversion client to checkout the source code for the port from the repository. This is likely what you would do if you wanted to contribute to the project or get the very latest code. The URL for the repository is:

```
http://svn.collab.net/repos/svn/branches/ebcdic/
```

Compile

In the SUBVERSION library of the binary distribution resides the command SVNMAKE. Just use SBMJOB to submit that command to batch.



The above is an example of the command you might run. All of the parameters have help associated with them.

NOTE: The command has a parameter to build mod_dav from source. If you specify *YES, then the source for mod_dav has to exist in a folder of that name inside the "subversion" folder. The zipped source distribution includes this source code, but a checkout from the repository would not. Once IBM has a PTF for their mod_dav available, and you have the PTF applied, you can just specify *NO for this parameter. You can also specify *NO if you just do not care about mod_dav_svn.

SVNMAKE produces a spooled file log of the build. Always review it to see if there were any errors. Also, note that SVNMAKE is just a "poor-man's" make utility. If a *.c source file is newer than the *MODULE it will recompile the *MODULE. However, if you change a *.h file it is not intelligent enough to rebuild any *.c files that use it. In that situation specify FORCE(*YES) to rebuild everything. Or manually delete the *MODULE objects that you need to rebuild.

Feel free to post build questions to the os400@subversion.tigris.org mailing list.

Using the Subversion Client

Subversion 1.3.0 for OS/400 adds support for the Subversion client, although that support is limited to the svn:// and file:// protocols. The easiest way to use the client is from QShell. This allows you to enter the command in essentially the same format as you would from a Unix or Windows command line. For example, to checkout a project, you could enter something like this in QShell:

```
/QSYS.LIB/SUBVERSION.LIB/SVN.PGM checkout
file://Subversion/repositories/repos1/Project1/trunk
/home/USER/dev/Project1
```

To commit you could enter:

```
/QSYS.LIB/SUBVERSION.LIB/SVN.PGM commit –m “Commit message”
```

If you are going to use the client a lot you can make a symbolic link to the program and name the link “svn” and put it somewhere in your PATH. Then you can enter commands just as you would on other platforms:

```
svn checkout file://Subversion/repositories/repos1/Project1/trunk
/home/USER/dev/Project1
```