Configuration versioning with Xoom

Prerequisites

The following software needs to be installed in order to set up Xoom versioning:

- A recent version of the Xoom client or server installation (version 3.1 or later). The client version is only suitable if the versioning is being set up on a remote system, and there is a running Xoom server on the actual managed system(s).
- Windows PowerShell v2.0. The installation can be downloaded for all current Windows systems, including Windows Server 2003 or Windows Server 2008 x64.
- A supported revision control system. Xoom currently supports Git, Mercurial and Subversion.

There are many other revision control systems out there, and Zany Ants will be happy to add support for additional revision control systems should there be customer demand.

The command line tools for the chosen revision control system need to be installed for the versioning script to work (which is typically, but not always, as for example with TortoiseSvn, the default), so we suggest some degree of attention to options when the revision control system is being installed. The system also needs to be configured for successful command line use, such as the authentication information and/or certificates need to be in place etc. The documentation of the revision control system in use should be consulted to achieve that, if necessary.

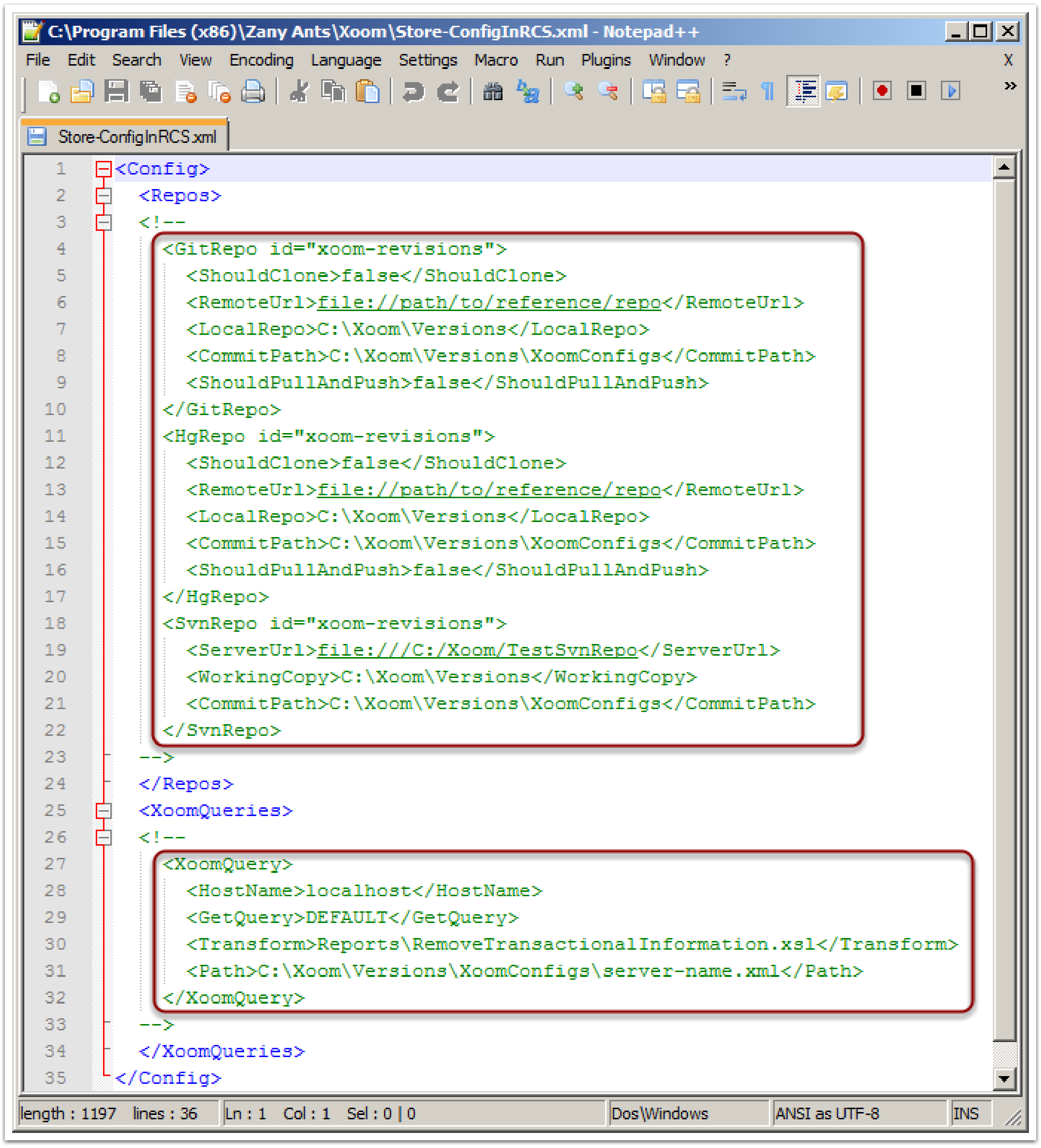
Building blocks

The Xoom versioning setup is composed of the following building blocks:

- The version repository in which the successive versions of the configuration snapshot will be stored. This can be a local or central (departmental, divisional, corporate) repository. It is best if the directory structure and the naming convention for the Xoom files for various servers is decided in advance. For example, if a single configuration snapshot will be obtained for each server, then the recommended configuration would be to have all configuration snapshots in the same folders, with file names equaling server names to which the files belong.
- The working copy of the relevant folder(s) used in the versioning on the Xoom client machine on which the versioning script will be run. The working copy will be created automatically when the script is run if it didnt exist previously. This working copy can then be used to view the history of the configuration on each server, and compare the configurations of different servers.
- The configuration file for the versioning script that defines what configurations will be retrieved and where they will be stored. The file is installed as part of the installation into the main Xoom folder under the name Store-ConfigInRCS.xml.
- The versioning script (requiring Windows PowerShell) that will be run either manually or through the scheduled task and will do the actual work of retrieving the configuration snapshot and storing it the revision control system if something has changed. The script is installed as part of the installation into the main Xoom folder under the name Store-ConfigInRCS.ps1.
- The Windows scheduled task that runs the versioning script at regular intervals.

Configuration

The versioning is configured by modifying the configuration file Store-ConfigInRCS.xml which is located in the main Xoom folder. The file contains the templates for configuring versioning with all three supported revision control systems in the commented-out sections (marked in red below). These templates can be copied outside the comment and modified with the configuration suitable for the local system.

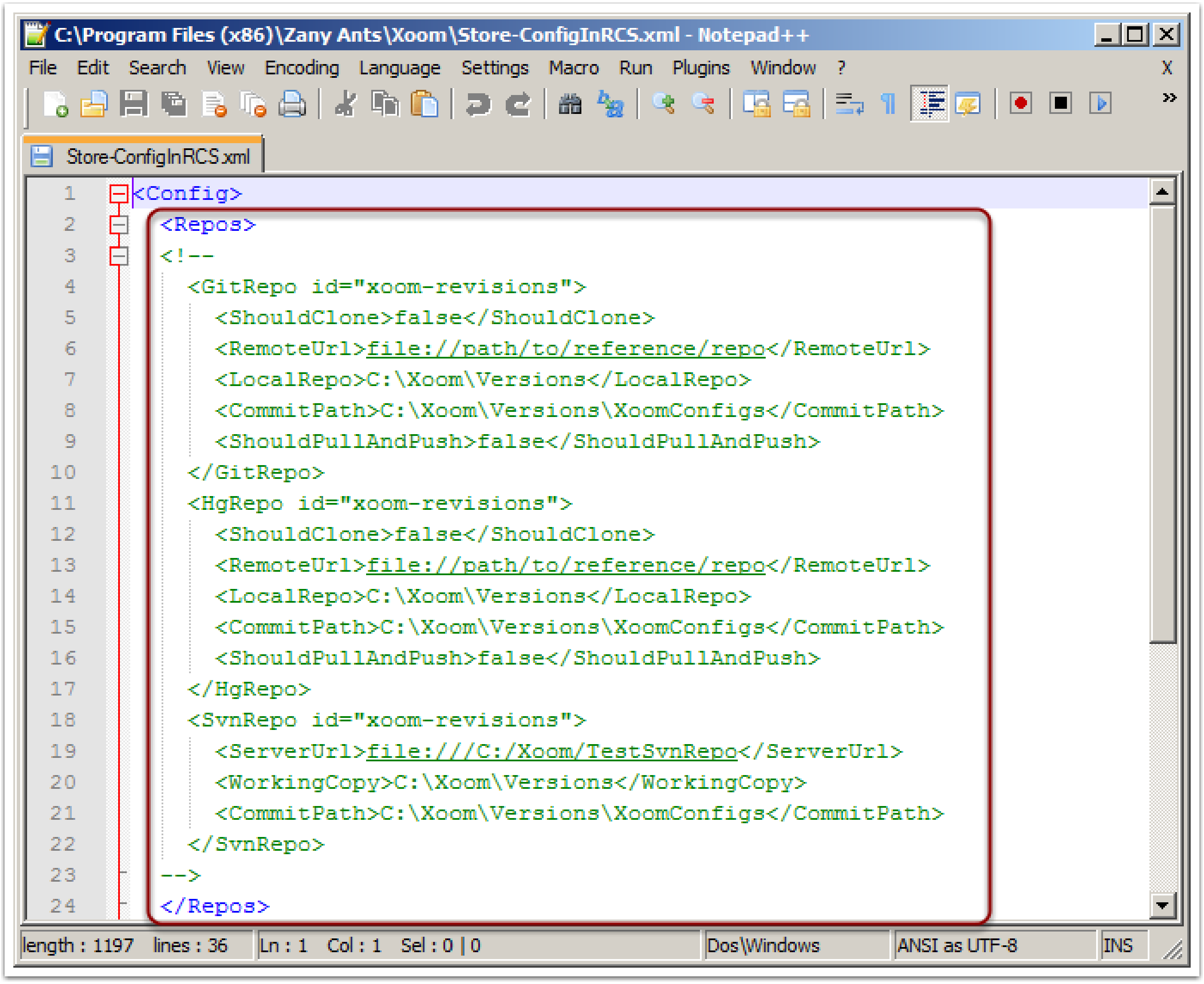

C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

```xml
<Config>
  <Repos>
  <!--
    <GitRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </GitRepo>
    <HgRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </HgRepo>
    <SvnRepo id="xoom-revisions">
      <ServerUrl>file:///C:/Xoom/TestSvnRepo</ServerUrl>
      <WorkingCopy>C:\Xoom\Versions</WorkingCopy>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
    </SvnRepo>
  -->
  </Repos>
  <XoomQueries>
  <!--
    <XoomQuery>
      <HostName>localhost</HostName>
      <GetQuery>DEFAULT</GetQuery>
      <Transform>Reports\RemoveTransactionalInformation.xsl</Transform>
      <Path>C:\Xoom\Versions\XoomConfigs\server-name.xml</Path>
    </XoomQuery>
  -->
  </XoomQueries>
</Config>
```

length : 1197   lines : 36   Ln : 1   Col : 1   Sel : 0 | 0   Dos\Windows   ANSI as UTF-8   INS

Repository and working copy configuration

The first section of the configuration file, the element called Repos (marked red), specifies which revision control repositories will be used, and where will the working copies be located. (If this is confusing, keep in mind that a repository and a working copy are one and the same in some revision control systems, such as for example with Mercurial.)
Typically there will be a single repository for storing configuration information, but the script controls any number of elements of the three kinds, corresponding to the three templates for the three supported revision control systems.



C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

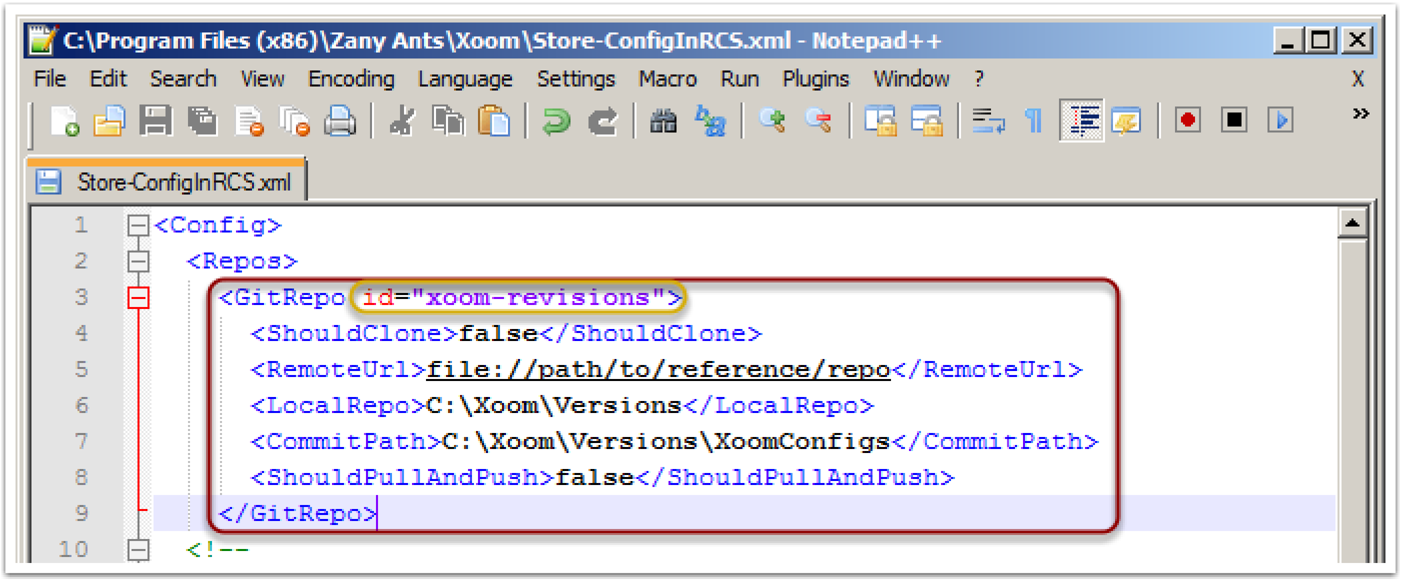
```
<Config>
  <Repos>
  <!--
    <GitRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </GitRepo>
    <HgRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </HgRepo>
    <SvnRepo id="xoom-revisions">
      <ServerUrl>file:///C:/Xoom/TestSvnRepo</ServerUrl>
      <WorkingCopy>C:\Xoom\Versions</WorkingCopy>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
    </SvnRepo>
  -->
  </Repos>
```

length : 1197 lines : 36 Ln : 1 Col : 1 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

Configuring a Git repository

Each Git repository is configured in a separate GitRepo element (marked in red below). There is an attribute called id that gives the repo a unique name (marked in beige), and the body contains the following elements:

- ShouldClone: if true, the local repository will be cloned from the remote repository if it doesn't exist at the time that the script is run.
- RemoteUrl: specifies the URL of the remote repository from which the local repository will be cloned (if cloning is enabled) and which will be used to pull and push the configuration information (if pulling and pushing are enabled, see below).
- LocalRepo: the path to the location of the local repository. If cloning is not enabled, then the repository already has to have been created before the script is run.
- CommitPath: the path that will be commited after all Xoom queries have been run. It is important that this path (or any folder under it) doesn't contain any other information that may be in development locally in order to avoid accidental commits of those changes.
- ShouldPullAndPush: if true, the information will be pulled from the remote repository (see RemoteUrl above) before Xoom queries are run, and pushed to the remote repository after the commit.

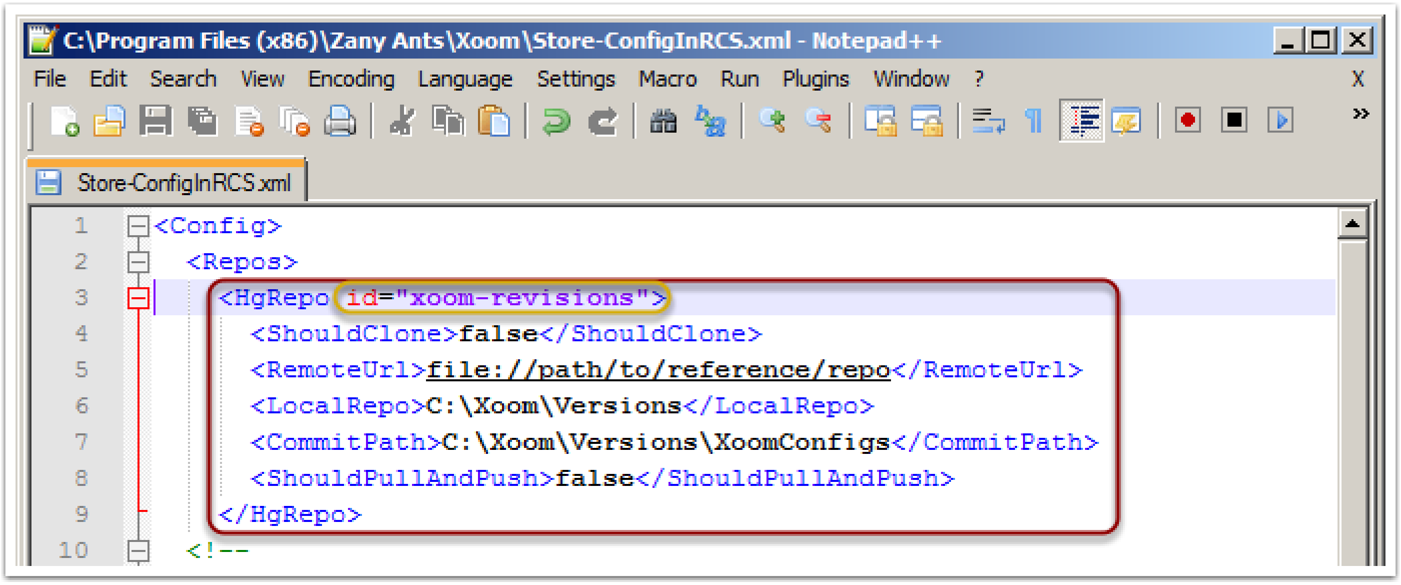

C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

```
<Config>
  <Repos>
    <GitRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </GitRepo>
  <!--
```

Configuring a Mercurial repository

Each Mercurial repository is configured in a separate HgRepo element (marked in red below). There is an attribute called id that gives the repo a unique name (marked in beige), and the body contains the following elements:

- ShouldClone: if true, the local repository will be cloned from the remote repository if it doesn't exist at the time that the script is run.
- RemoteUrl: specifies the URL of the remote repository from which the local repository will be cloned (if cloning is enabled) and which will be used to pull and push the configuration information (if pulling and pushing are enabled, see below).
- LocalRepo: the path to the location of the local repository. If cloning is not enabled, then the repository already has to have been created before the script is run.
- CommitPath: the path that will be commited after all Xoom queries have been run. It is important that this path (or any folder under it) doesn't contain any other information that may be in development locally in order to avoid accidental commits of those changes.
- ShouldPullAndPush: if true, the information will be pulled from the remote repository (see RemoteUrl above) before Xoom queries are run, and pushed to the remote repository after the commit.

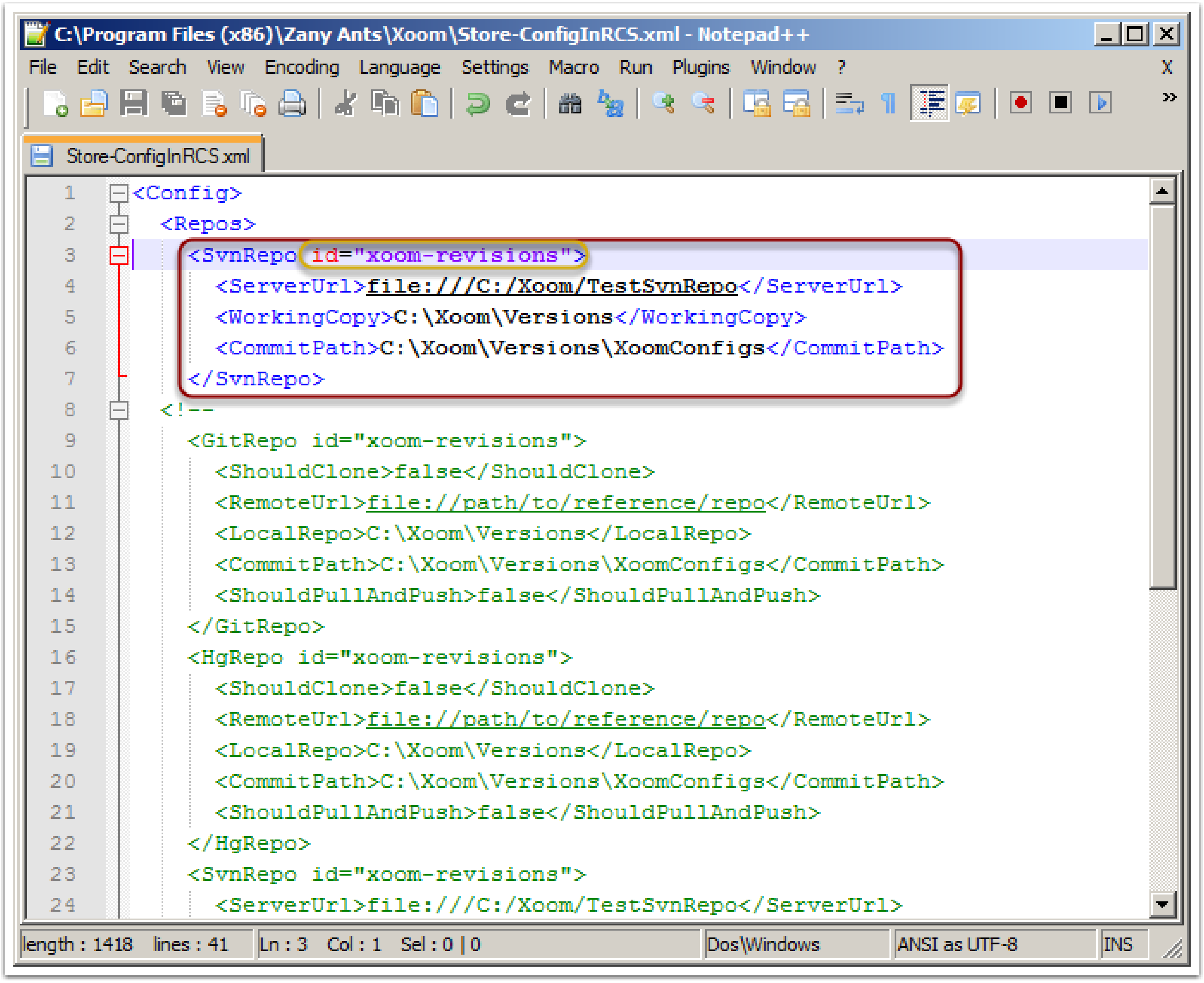

C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

```
<Config>
  <Repos>
    <HgRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </HgRepo>
  <!--
```

Configuring a Subversion repository and working copy

Each Subversion repository is configured in a separate SvnRepo element (marked in red below). There is an attribute called id that gives the repo a unique name (marked in beige), and the body contains the following elements:

- ServerUrl: specifies the URL of the repository from which the local working copy should be or has been checked out.
- WorkingCopy: the path to the location of the local working copy.
- CommitPath: the path that will be commited after all Xoom queries have been run. It is important that this path (or any folder under it) doesn't contain any other information that may be in development locally in order to avoid accidental commits of those changes.


C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

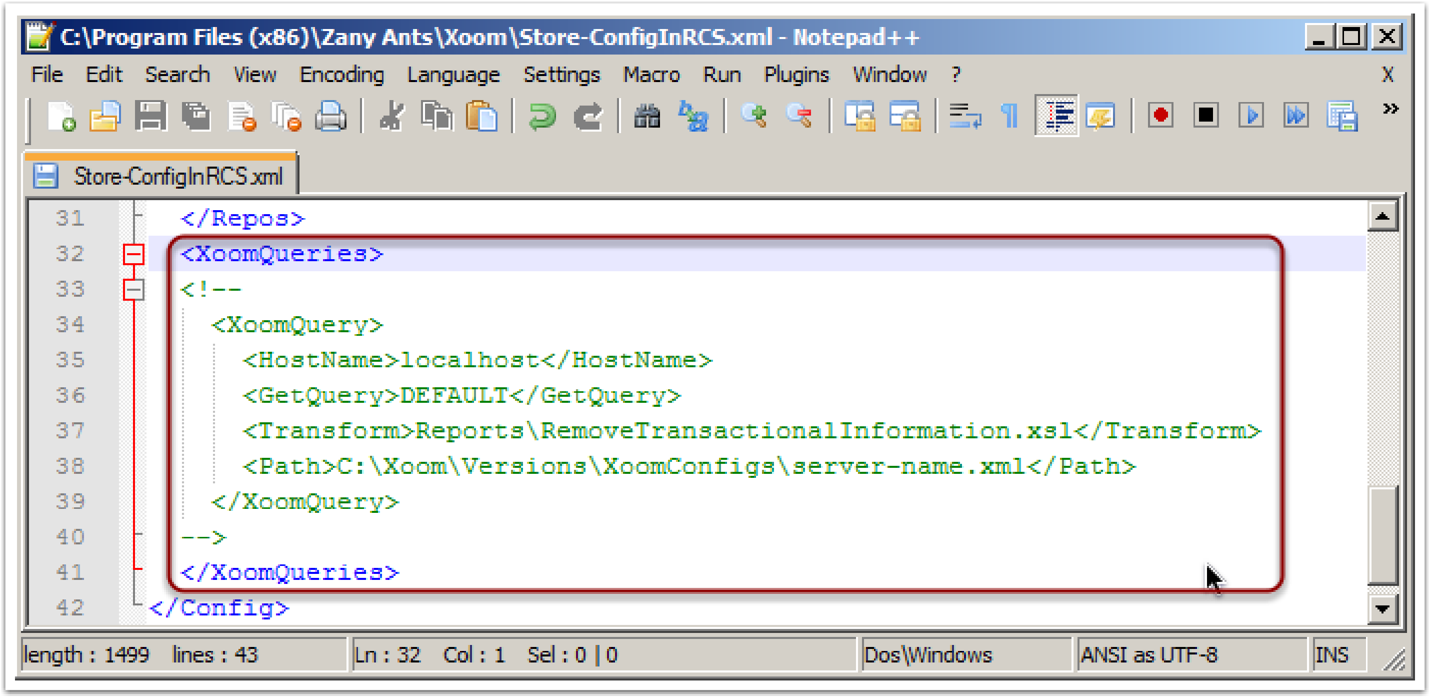
```
<Config>
  <Repos>
    <SvnRepo id="xoom-revisions">
      <ServerUrl>file:///C:/Xoom/TestSvnRepo</ServerUrl>
      <WorkingCopy>C:\Xoom\Versions</WorkingCopy>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
    </SvnRepo>
  <!--
    <GitRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </GitRepo>
    <HgRepo id="xoom-revisions">
      <ShouldClone>false</ShouldClone>
      <RemoteUrl>file://path/to/reference/repo</RemoteUrl>
      <LocalRepo>C:\Xoom\Versions</LocalRepo>
      <CommitPath>C:\Xoom\Versions\XoomConfigs</CommitPath>
      <ShouldPullAndPush>false</ShouldPullAndPush>
    </HgRepo>
    <SvnRepo id="xoom-revisions">
      <ServerUrl>file:///C:/Xoom/TestSvnRepo</ServerUrl>
```

length : 1418 lines : 41 Ln : 3 Col : 1 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

Xoom query configuration

The second part of the configuration file, the element called XoomQueries, contains the specific queries that will be sent to specific Xoom servers, processed using specific transformation and stored into specific files. The section can contain any number of XoomQuery sections, making it possible to version configurations for a number of environments from a single client machine, and also to use different kinds of queries, such for example Service Optimization internal configuration separately from the file information.
Each XoomQuery contains the following elements:

- HostName: the host name of the Xoom server on which the query will be run.
- GetQuery: the name of the query (as defined on that Xoom server) that will be used to get the relevant information.
- Transform: the file name of the XSLT file that will be used to transform the result of the query after it was retrieved from Xoom. By default this would remove transactional information (i.e. information subject to routine changes as opposed to more stable configuration information). The transformation is optional, so if there is no need for transformation this element can be omitted.
- Path: The full path (including the file name) where the final file will be stored. This path would typically be in the right place in the repository / working copy, so that it can be versioned.


C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml

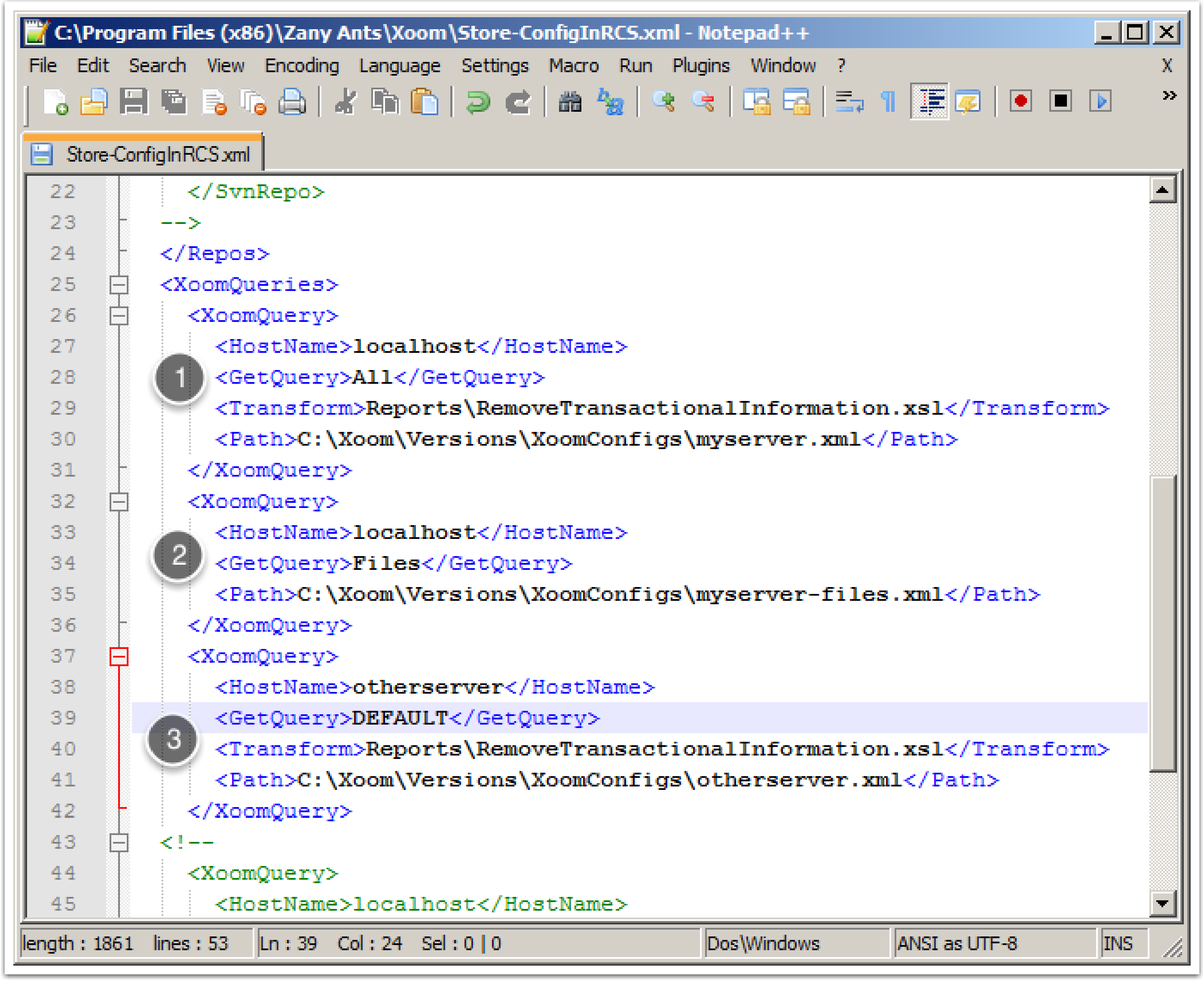
```
  </Repos>
  <XoomQueries>
  <!--
    <XoomQuery>
      <HostName>localhost</HostName>
      <GetQuery>DEFAULT</GetQuery>
      <Transform>Reports\RemoveTransactionalInformation.xsl</Transform>
      <Path>C:\Xoom\Versions\XoomConfigs\server-name.xml</Path>
    </XoomQuery>
  -->
  </XoomQueries>
</Config>
```

length : 1499 lines : 43
Ln : 32 Col : 1 Sel : 0 | 0
Dos\Windows
ANSI as UTF-8
INS

Example Xoom query configuration

Below we show examples of three configured queries:

1. In this query, we use the All named query to obtain the Service Optimization internal information from the local Xoom server (localhost). After the retrieval, we remove transactional information and store the result as myserver.xml in the versioning folder intended for Xoom files.
2. In this query, we use the Files named query to retrieve file information from the local Service Optimization server. No transformation is required after the retrieval, and we store the result as myserver-files.xml in the same versioned location.
3. Finally, in this query we use the DEFAULT query on the remote Xoom server (host name otherserver in this example) to obtain the default subset of the internal configuration. We remove the transactional information after the retrieval, and store the result as otherserver.xml.


C:\Program Files (x86)\Zany Ants\Xoom\Store-ConfigInRCS.xml - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Store-ConfigInRCS.xml
1
2
3
length : 1861 lines : 53 Ln : 39 Col : 24 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

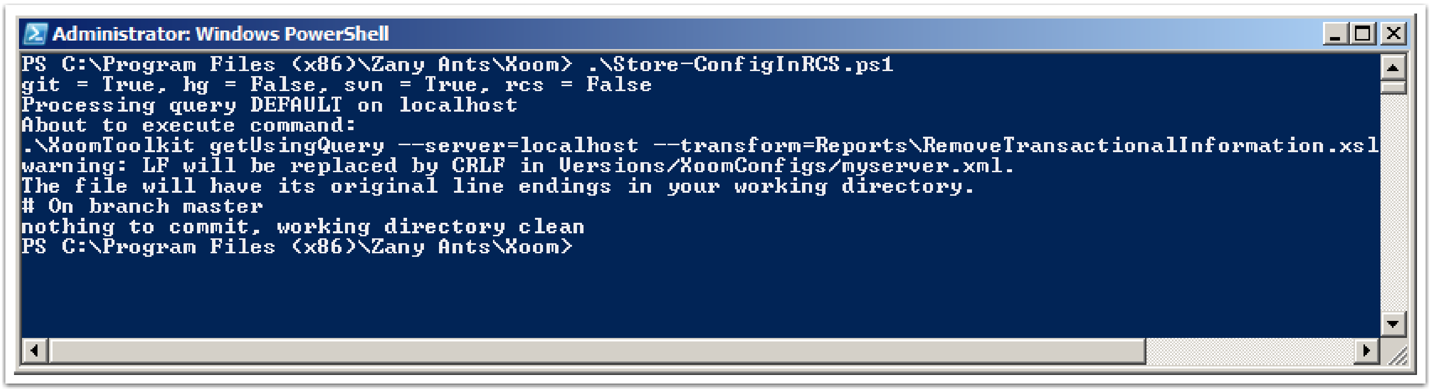
```
    </SvnRepo>
  -->
  </Repos>
  <XoomQueries>
    <XoomQuery>
      <HostName>localhost</HostName>
      <GetQuery>All</GetQuery>
      <Transform>Reports\RemoveTransactionalInformation.xsl</Transform>
      <Path>C:\Xoom\Versions\XoomConfigs\myserver.xml</Path>
    </XoomQuery>
    <XoomQuery>
      <HostName>localhost</HostName>
      <GetQuery>Files</GetQuery>
      <Path>C:\Xoom\Versions\XoomConfigs\myserver-files.xml</Path>
    </XoomQuery>
    <XoomQuery>
      <HostName>otherserver</HostName>
      <GetQuery>DEFAULT</GetQuery>
      <Transform>Reports\RemoveTransactionalInformation.xsl</Transform>
      <Path>C:\Xoom\Versions\XoomConfigs\otherserver.xml</Path>
    </XoomQuery>
  <!--
    <XoomQuery>
      <HostName>localhost</HostName>
```

Making sure everything works

Before attempting to set up a scheduled task, the versioning script should be run from the Windows PowerShell command prompt in order to make sure that the revision control system is configured correctly, that the user has all the required rights to access and modify the repositories, and that all relevant Xoom servers are accessible and operational. In order to do that, we start Windows PowerShell, change to the Xoom folder and run the versioning script:

```
.\Store-ConfigInRCS.ps1
```

If the run completed successfully without any errors, such as in the example in the screenshot, then we are ready to set up a scheduled task. Otherwise any errors should be resolved first.



```
PS C:\Program Files (x86)\Zany Ants\Xoom> .\Store-ConfigInRCS.ps1
git = True, hg = False, svn = True, rcs = False
Processing query DEFAULT on localhost
About to execute command:
.\XoomToolkit getUsingQuery --server=localhost --transform=Reports\RemoveTransactionalInformation.xsl
warning: LF will be replaced by CRLF in Versions/XoomConfigs/myserver.xml.
The file will have its original line endings in your working directory.
# On branch master
nothing to commit, working directory clean
PS C:\Program Files (x86)\Zany Ants\Xoom>
```

Setting up a Windows scheduled task

Once we know that the versioning script is working correctly with our configuration, we want to make it run automatically at regular intervals. We do this by creating a Windows scheduled task. The specifics of how to configure a scheduled task are different in different versions of Windows, but they are all well documented elsewhere. The desired interval and other settings are subject to specific wishes of each customer and possibly environment, but the important thing is that the scheduled task runs the following command line:

```
C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.exe -command "& 'C:\Program Files
(x86)\Zany Ants\Xoom\Store-Config-In-Svn.ps1'"
```

The specific paths may vary depending on the version of Windows and where PowerShell and Xoom are installed, but the parameters should all be present in the same way and in the same order.