Configuration Changes in MicroStation CONNECT Edition

With MicroStation CONNECT Edition we have made some significant changes to what was known as a workspace. Below is the information regarding these changes.

Configuration Changes in MicroStation CONNECT Edition

1. Introduction

In MicroStation CONNECT Edition (hereafter referred to as CONNECT), there have been a number of changes to the way Configuration Files are organized and processed. This document describes those changes. It assumes a knowledge of Configuration File concepts and a familiarity with how they were used in MicroStation V8i (hereafter referred to as V8i).

2. Terminology Changes

In V8i, the overall usage of Configuration Files and Configuration Variables was referred to as Workspaces. That overall usage is referred to as the Configuration in CONNECT.

In V8i, the grouping of Configuration Files and Configuration Variables that were used across an entire organization was referred to as the “Site” level. In CONNECT, it is referred to as the “Organization” level.

In V8i, a grouping of files and associated data was referred to as a “Project”. That terminology has been problematic, because MicroStation user organizations use the term Project for a business purpose that is rarely precisely correlated to the data grouping in MicroStation. Therefore that grouping is now referred to as a “WorkSet”. For example, the Configuration Variable level formerly known as Project is now the WorkSet level.

In V8i, Projects were organized by “User”. Selecting a different User had the effect of changing the list of Projects that were available, and also changing the User Preference (.upf) file and User Configuration (.ucf) file in use. In CONNECT, selection of User has been removed from the user interface. Each CONNECT user has their own personal .upf file and .ucf file, associated with their Windows login, so that configuration variables that they set and user preferences that they select are not affected by the WorkSpace or WorkSet they are working on.

3. Directory Structure differences

In MicroStation V8i, the default delivery directory was to a root directory that contained MicroStation, Workspace, and Documentation subdirectories. In keeping with the terminology change discussed above, the Workspace directory has been renamed to Configuration. The V8i WorkSpace directory included Bentley-provided system data in the System subdirectory. That was a disadvantage because some of that data is MicroStation version specific, so in CONNECT, there is no Configuration\System directory. Its contents have been moved into a subdirectory of the MicroStation directory.

4. Configuration Level differences

In V8i, there were 5 Configuration Levels, organized from lowest to highest priority as shown below. Configuration Variables defined at higher levels override definitions at lower levels:

System	0
Application	1
Site	2
Project	3
User	4

In a V8i Configuration File, the level at which Configuration Variables were defined was set using the %level directive, specifying one of the numbers in the right column above. An alternative was to put the level number at which Configuration Files were to be processed in the %include statement.

In CONNECT, there are 7 Configuration Levels. From lowest to highest priority they are:

System	0
Application	1
Organization	2
WorkSpace	3
WorkSet	4
Role	5
User	6

In a Configuration File, the level is still set using the %level directive, but now the argument is one of the text values in the left column instead of the number. That improves readability of configuration files. You can still specify a number, but a warning is generated because it is an indication that you might be processing a pre-CONNECT configuration file, and the Configuration Level number might no longer be correct. Most of the time, the %level directives (or level settings in %include statements) are in Bentley-delivered configuration files that sequence inclusion of user-authored configuration files, rather than in user-authored configuration files. However, advanced users may have used %level directives, and thus might encounter the warning.

The two additional levels in the CONNECT configuration, WorkSpace and Role, are discussed below.

5. WorkSpace

In CONNECT, a “WorkSpace” is a container grouping WorkSets, standards files, and associated Configuration Files that used in a particular broad context. Different user organizations will have different uses for the “WorkSpace” grouping mechanism. Engineering and Architectural firms might use a separate WorkSpace for each of their clients. Asset owners are likely to use a separate WorkSpace for each asset or department. For that reason, the label that appears for WorkSpace in the CONNECT user interface can be set by a configuration variable. That is done by specifying the Configuration Variable _USTN_WORKSPACELABEL in the WorkSpaceSetup.cfg file in the Configuration subdirectory. The default value is the neutral “WorkSpace”.

For those familiar with MicroStation V8i, there is some parallel between WorkSpaces and the “User” concept in V8i, in that the User Configuration File was used to filter the displayed Projects list. However, WorkSpaces are better suited to typical user organization workflows because they appear at the correct level in the hierarchy of Configuration Levels, and because they do not affect user-specific settings.

6. Role

A frequent enhancement request has been to add a Configuration Level that allows standards and certain program behavior to be controlled based on the role or discipline of an individual user. CONNECT provides this additional Configuration Level, but leaves it to the user organization to determine how to identify the role of individual users. To use this feature, the Configuration Variable _USTN_ROLECFG is set to the full file name of a Configuration File that contains the role-specific Configuration Variable settings. That can be accomplished in any of the following ways:

- setting a system environment variable at login time,
- conditional tests in WorkSet or WorkSpace configuration files,
- setting the configuration variable in the Configuration dialog,
- some other mechanism devised by the system administrator.

The _USTN_ROLECFG Configuration File is processed after the WorkSpace, WorkSet and User Configuration Files have been processed, to allow any of them to specify _USTN_ROLECFG.

7. Configuration Fundamentals

MicroStation and associated applications define Configuration Variables at the System and Application levels in Configuration Files that are delivered with the product. Administrators generally make changes at the Organization, WorkSpace, WorkSet, and Role levels, in user-supplied Configuration Files. MicroStation provides template Configuration Files that can be used as a starting point for those Configuration Files.

Configuration Variables are organized into Framework Configuration Variables, which start with the “_USTN_” prefix, and Operational Configuration Variables, most of which start with the “MS_” prefix. In general, the Framework Configuration Variables are used in Configuration Files, while Operational Configuration Variables are used to direct program flow within MicroStation. A few of the

Framework Configuration Variables are determined by the MicroStation installation directory. Other Framework Configuration Variables default to locations relative to the installation directory, but can be (and some of them are expected to be) changed in configuration files provided by the user.

MicroStation's Configuration File processing can be regarded as interpreting a simple program, part of which is provided by System Configuration Files, which should not be modified by the user, and part of which is provided by Configuration Files that are intended to be user modified. All configuration files are simple text files that can be examined (and modified, in the case of user-modifiable Configuration Files) with any text editor.

The system Configuration Files are located in the MicroStation/config installation directory, while user-modifiable Configuration Files are in the Configuration installation directory or other user-specified directories. User-modifiable Configuration Files are included into the Configuration File processing flow at appropriate times by the System Configuration Files.

Section 9 provides a "walk through" of the processing of Configuration Files, and identifies the touch points where user-modifiable Configuration Files can specify directories and MicroStation behavior.

An essential aid to understanding Configuration File processing is the MicroStation command line argument "-debug". That tells MicroStation to write out a text file that contains the history of how every Configuration Files was processed, and to open that file in whatever editor that your Windows system has configured to handle text files (usually Notepad). When you close Notepad, MicroStation closes also.

While running MicroStation, you can see the current values of all Configuration Variables using the new "SHOW CONFIGURATION" command. That also opens Notepad with the current Configuration Variables.

8. Configuration File Syntax

Configuration files consist of statements of three types:

- Flow Directives that control the flow through Configuration Files.
- Variable Directives that control certain aspects of Configuration Variables
- Assignment statements that set the value of Configuration Variables.
- Expressions and operators that manipulate strings or Configuration Variables to yield results that can be used in directives or assignments

Configuration Variables are often defined in terms of other Configuration Variables. This can be done in two different ways:

- When plain parentheses are used, as in $(<CfgVarName>), the expression is stored verbatim in the Configuration Variable definition and evaluated later, when the value of the Configuration Variable is eventually needed during program execution. This form is much more flexible, since the definition can be set using other Configuration Variables even if they have not yet been defined. It is thus more commonly used.

• When curly braces are used, as in ${<CfgVarName>}, the value of CfgVarName is evaluated immediately, while the configuration file itself is being processed. Therefore, the Configuration Variable used in the expression must be previously defined.

9. Configuration File Processing

Configuration File processing starts with the Configuration File mslocal.cfg. It is a “bootstrap” file with only a few lines – it includes msdir.cfg, another small Configuration File that is generated at install time and identifies the MicroStation installation directory, and then includes msconfig.cfg, which contains the main “program flow” of Configuration File processing.

The msconfig.cfg Configuration File

You should never modify msconfig.cfg itself (or any of the other configuration files in the MicroStation program directory). As you will see in the discussion below, there are a number of well-defined places where msconfig.cfg includes user-modifiable Configuration Files. It is in those user-modifiable Configuration Files that Configuration Variables should be modified to provide all the flexibility necessary to meet your organization’s requirements for data location.

The msconfig.cfg Configuration File begins by setting the _USTN_BENTLEYROOT Configuration Variables and a number of Framework Configuration Variables that point to directories where program data is delivered. Those are necessary for program operation, but do not define locations for any user data or files. It then includes the system and application Configuration Files that are shipped with MicroStation.

```
#----------------------------------------------------------------------*/

# Include all the delivered system configuration files.

# These define System level configuration variables.

#----------------------------------------------------------------------

%include $(_USTN_SYSTEM)*.cfg level System



#----------------------------------------------------------------------

# Include the delivered application configuration files.

# These define Application level configuration variables.

#----------------------------------------------------------------------

%include $(_USTN_APPL)*.cfg level Application
```

The part relevant to setting up a customized Configuration begins where msconfig.cfg defines the _USTN_CONFIGURATION Configuration Variable:

```
#----------------------------------------------------------------------

# Define the root directory for the Configuration data.

#----------------------------------------------------------------------

_USTN_CONFIGURATION     : ${_USTN_BENTLEYROOT}Configuration/
```

Since the curly braces are used, this is immediately evaluated to the Configuration subdirectory of the installation directory.

By default, many other Configuration Variables are set to subdirectories of the directory defined by _USTN_CONFIGURATION. The definition of those variables (_USTN_ORGANIZATION, _USTNWORKSPACESROOT, _USTNWORKSETSROOT, etc.) follows the definition of _USTN_CONFIGURATION.

The WorkSpaceSetup.cfg Configuration File

The first opportunity for customization of your configuration is where msconfig.cfg includes WorkSpaceSetup.cfg:

```
%if exists ($(_USTN_CONFIGURATION)WorkSpaceSetup.cfg)

% include $(_USTN_CONFIGURATION)WorkSpaceSetup.cfg

%endif
```

As you can see WorkSpaceSetup.cfg is located in the Configuration directory of the delivery, and is intended to be customized by users. Here is the contents as it shipped:

```
#----------------------------------------------------------------------

# WorkSpaceSetup.cfg - Configures WorkSpace for Your Organization

#

# The main function of this configuration file is to set the label that

# your organization wants to use for WorkSpaces. WorkSpaces are the

# grouping mechanism for WorkSets. The label for this level of grouping

# could be Client, Facility, Department, Owner, or whatever you would

# like it to be. The default label is the neutral "WorkSpace". Uncomment

# the definition below, and set it to your preferred label.

#----------------------------------------------------------------------

# _USTN_WORKSPACELABEL   : WorkSpace
```

```
#----------------------------------------------------------------------

# A second possible use for this configuration file is to redirect the

# root directory where your Organization-wide standards are stored to somewhere

# other than the default. The default is $(_USTN_CONFIGURATION)Organization/.

# It can be changed by redefining _USTN_ORGANIZATION.

#----------------------------------------------------------------------


#----------------------------------------------------------------------

# A third possible use for this configuration file is to redirect the

# root directory where your WorkSpaces are stored to somewhere other

# than the default. The default is $(_USTN_CONFIGURATION)WorkSpaces/.

# It can be changed by redefining _USTN_WORKSPACESROOT.

#----------------------------------------------------------------------
```

The comments in the file explain its three principle uses – setting the WorkSpace label to the desired value for your organization, and possibly redirecting your organization-wide standards data and/or all of your other WorkSpace data to another location (for example, to a network share). See the "Typical Configuration Scenario" section below for an example of usage.

Organization Configuration Files

Upon returning from including WorkSpaceSetup.cfg, msconfig.cfg includes your organization-wide Configuration Files (if any):

```
#----------------------------------------------------------------------

# Include the Organization specific configuration files.

# The configuration files in the _USTN_ORGANIZATION directory are intended to

# set configuration variables that point to organization-wide standards

# such as level libraries, cell libraries, etc. Those settings can be

# augmented or overridden at the WorkSpace or WorkSet level.

#----------------------------------------------------------------------

%level Organization
```

```
%if exists ($(_USTN_ORGANIZATION)*.cfg)

% include $(_USTN_ORGANIZATION)*.cfg

%endif
```

As delivered, _USTN_ORGANIZATION points to the Configuration\Organization directory, which contains one Configuration File, standards.cfg, which sets Configuration Variables assuming the generic directory structure in the delivery. If you have redirected _USTN_ORGANIZATION, you can either adopt the same directory structure or change it to meet your needs. Since the %include statement above includes all Configuration Files in the _USTN_ORGANIZATION directory, you can call your Configuration File something other than standards.cfg, and you can use more than one Configuration File in that directory if you prefer. They will be included in alphabetical order.

The User Configuration File

The next step is to include the User Configuration file. The User Configuration file is stored in the preferences directory as Personal.ucf, and stores the individual users preferences and state. It is included here, because the most recently used WorkSpace and WorkSet are stored in the User Configuration file.

The WorkSpace Configuration File

The next step in the configuration process is to determine the WorkSpace Configuration File. Each WorkSpace has a Configuration File named <WorkSpaceName>.cfg that must be located in the directory pointed to by _USTN_WORKSPACESROOT. As delivered, _USTN_WORKSPACESROOT is defined as $(_USTN_CONFIGURATION)WorkSpaces/, but that can be changed in the WorkSpaceSetup.cfg file as discussed above.

One and only one WorkSpace Configuration File is processed. There is some logic in msconfig.cfg that determines which WorkSpace Configuration File to load by setting the _USTN_WORKSPACENAME, but that can be ignored for our current purposes – MicroStation is responsible for remembering the most recent WorkSpace name and using it to set _USTN_WORKSPACENAME.

The WorkSpace Configuration File is then included from msconfig.cfg using this construction:

```
%if defined (_USTN_WORKSPACENAME)

% if exists ($(_USTN_WORKSPACESROOT)$(_USTN_WORKSPACENAME).cfg)

   _USTN_WORKSPACECFG = $(_USTN_WORKSPACESROOT)$(_USTN_WORKSPACENAME).cfg

%   include $(_USTN_WORKSPACECFG) level WorkSpace

% endif

%endif
```

User organizations will frequently customize WorkSpace Configuration Files. When a new WorkSpace is created, it starts with a template WorkSpace Configuration File like the following:

```
#----------------------------------------------------------------------
# WorkSpace.Template - Template for new WorkSpaces
#
# When MicroStation runs, one and only one WorkSpace configuration file
# is chosen and included.
#
# The function of the WorkSpace configuration file is to define the location
# of _USTN_WORKSPACEROOT, _USTN_WORKSPACESTANDARDS, and/or _USTN_WORKSETSROOT
# for this WorkSpace. Those are the root directory, the standards directory,
# and the WorkSets root directory, respectively.
#
# Default locations are defined in msconfig.cfg:
# _USTN_WORKSPACEROOT is $(_USTN_WORKSPACESROOT)$(_USTN_WORKSPACENAME)/
# _USTN_WORKSPACESTANDARDS is defined as $(USTN_WORKSPACEROOT)Standards/
# _USTN_WORKSETSROOT is defined as $(USTN_WORKSPACEROOT)WorkSets/
# If those defaults are acceptable, this file need not make any definitions.
# To move all WorkSpace data to a separate directory (e.g., to a network share)
# _USTN_WORKSPACEROOT can be redefined and the default values retained for
# _USTN_WORKSPACESTANDARDS and _USTN_WORKSETSROOT
#----------------------------------------------------------------------
```

As you can see, any combination of the WorkSpace root, standards, or WorkSets root directory can be redirected according to the user’s requirements.

The next step in msconfig.cfg is to include any Configuration Files that are stored in the directory pointed to by _USTN_WORKSPACEROOT:

```
#----------------------------------------------------------------------
```

```
# When we get to this point, we have a WorkSpace defined.

# There may be .cfg files within the WorkSpace. Process those here.

#----------------------------------------------------------------------

%if exists ($(_USTN_WORKSPACEROOT)*.cfg)

% include $(_USTN_WORKSPACEROOT)*.cfg level WorkSpace

%endif
```

These Configuration File(s) are optional, and can contain whatever Configuration Variable definitions that are appropriate in the user's workflow. Often, no additional Configuration Files are needed. The Example WorkSpace delivered with MicroStation doesn't have any.

The WorkSet Configuration File

After the WorkSpace Configuration Files have been processed, msconfig.cfg attempts to load one (and only one) WorkSet Configuration File. Each WorkSet within a WorkSpace has a Configuration File named <WorkSetName>.cfg that must be located in the directory pointed to by _USTN_WORKSETSROOT. By default, _USTN_WORKSETSROOT is defined as $(_USTN_WORKSPACEROOT)WorkSets/, but that can be changed in the WorkSet Configuration File as discussed above. MicroStation is responsible for remembering the most recently used WorkSet, and it sets the _USTN_WORKSETNAME configuration variable accordingly. The WorkSet Configuration File is included like this:

```
%if exists ($(_USTN_WORKSETCFG))

% include $(_USTN_WORKSETCFG) level WorkSet

%endif
```

The Role Configuration File

After WorkSet Configuration files have been processed, msconfig.cfg checks to see whether _USTN_ROLECFG has been defined. If it has, the Configuration File that it is defined in _USTN_ROLECFG is processed. As mentioned above, MicroStation has no default for _USTN_ROLECFG – it must be set as a system environment variable or defined in one of the user-modifiable Configuration Files that are processed prior to reaching this part of msconfig.cfg

The final part of msconfig.cfg handles the database configuration file, if that feature is in use.

When msconfig.cfg has been fully processed, all the initial Configuration Variable definitions are complete

10. Configuration Changes during MicroStation execution

Whenever a different WorkSet is selected, MicroStation takes the following Configuration Variable actions:

- If the WorkSet belongs to a WorkSpace that is different from the WorkSpace to which the currently active WorkSet belongs, all WorkSpace level Configuration Variables are discarded.
- All WorkSet level Configuration Variables are discarded.
- If the WorkSet belongs to a different WorkSpace, that WorkSpace's Configuration File(s) are processed.
- The WorkSet Configuration Files are processed.
- The new WorkSpace and WorkSet are written to the user's Personal.ucf file.

A different WorkSet can be selected in a number of ways:

- From File->Browse, a different WorkSpace/WorkSet can be selected in the File Browser dialog.
- From the Start Page (reached using File->Close) a different WorkSpace/WorkSet can be selected either directly or by selecting one of the Recent WorkSets.
- In CONNECT, design files have their WorkSpace / WorkSet recorded in their metadata. When a design file with such metadata is opened, and it is from a different WorkSet, the user can opt to change to the WorkSpace / WorkSet recorded in its metadata.
- Using the "-WW<workset> command line argument when starting MicroStation.

11. Configuration Variable Changes between V8i and CONNECT

The following table lists the framework Configuration Variables in CONNECT that are either new or replace Configuration Variables in V8i. In the case of a replacement, the second column lists the V8i Configuration Variable.

12. Typical Configuration Scenario

Tensor Engineering performs services for a number of clients. The firm has its own internal standards for drawing borders, cell libraries, levels, materials, etc. However, some of its clients require that their own standards be used in addition or instead. The firm's Information Technology department uses one server 'Tensor' for internal standards. One of its major clients is Imperial Biotechnology Company, who has their own standards. The IT department maintains a server 'Imperial' for all of the work that it does for Imperial Biotechnology. The Administrator is tasked with defining the appropriate Configuration for Tensor Engineering and for the work it does on behalf of Imperial Biotechnology, and getting all MicroStation users to share that configuration. Here are steps that can be taken.

Tensor's MicroStation administrator configures the MicroStation installation to install the MicroStation desktop shortcut to include the command line argument "-

WR\\Tensor\MicroStation\Configuration\”. This defines the _USTN_CONFIGURATION variable when MicroStation is started.

The administrator copies the delivered WorkSpaceSetup.cfg to the \\Tensor\MicroStation\Configuration directory (to which _USTN_CONFIGURATION is defined at startup) and edits it to read as follows:

```
#-------------------------------------------------------------------

# WorkSpaceSetup.cfg - Configures WorkSpace for Tensor Engineering

#

# Set the WorkSpace label to Client

#-------------------------------------------------------------------

_USTN_WORKSPACELABEL   = Client



#-------------------------------------------------------------------

# Tensor-wide standards are located on the ‘Tensor’ server

#-------------------------------------------------------------------

_USTN_ORGANIZATION      = //Tensor/MicroStation/TensorStandards/



#-------------------------------------------------------------------

# Tensor Client Configuration files are located on the server.

#-------------------------------------------------------------------

_USTN_WORKSPACESROOT    = //Tensor/MicroStation/Clients/
```

Tensor Engineering decides to use the Bentley-suggested subdirectory structure for its internal standards. Therefore the administrator creates the same subdirectories as are in the delivered example Configuration\Organization directory, and populates those directories with cell libraries, DGNLIBS, materials, fonts, etc. that match Tensor standards. Since the subdirectory structure is the same as the example, the administrator simply copies the delivered Standards.cfg to the \\Tensor\MicroStation\TensorStandards\ directory. Since all Configuration Variable definitions in standards.cfg are relative to _USTN_ORGANIZATION, no changes are needed.

The administrator sets up the Imperial Biotechnology configuration. The starting point is the Example.cfg file delivered in Configuration\WorkSpaces, so he or she copies that file to “Imperial BioTechnology.cfg” in the \\Tensor\MicroStation\Clients\ directory. Since all of the Imperial Biotechnology data is located on the Imperial server (in the share called MicroStation), the easiest thing to do is to simply redirect _USTN_WORKSPACEROOT to a share on that server. The

‘MicroStation’ share contains Standards and WorkSets subdirectories, which match the defaults for _USTN_WORKSPACESTANDARDS and _USTN_WORKSETSROOT, so there is no need to set those. Imperial Biotechnology supplies some additional DGNLIB files and cell libraries used for Tensor to use, so those are appended to the appropriate Configuration Variables. So the completed “Imperial Biotechnology.cfg” WorkSpace Configuration File looks like this.:

```
#--------------------------------------------------------------------

# Imperial Biotechnology.cfg – Configuration for Imperial Biotechnology

#

# All of the standards and WorkSets for Imperial are located on the

# ‘Imperial’ server.

#--------------------------------------------------------------------

_USTN_WORKSPACEROOT     =     //Imperial/MicroStation/


#--------------------------------------------------------------------

# Imperial Biotechnology supplies several DgnLibs and GUI DgnLib files:

#--------------------------------------------------------------------

MS_DGNLIBLIST          > $(_USTN_WORKSPACESTANDARDS)DgnLib/*.dgnlib

MS_GUIDGNLIBLIST       > $(_USTN_WORKSPACESTANDARDS)DgnLib/GUI/*.dgnlib


#--------------------------------------------------------------------

# Imperial Biotechnology supplies several Cell libraries, so

# add their subdirectory to the Cell library search path.

#--------------------------------------------------------------------

MS_CELL                > $(_USTN_WORKSPACESTANDARDS)Cell/
```

Tensor Engineering secures a new contract with Imperial Biotechnology for renovation of their laboratory in Taos, New Mexico. In the “Imperial Biotechnology” WorkSpace, they create a new WorkSet called ‘Taos’. The WorkSet creation tool copies the WorkSet template to _USTN_WORKSETSROOT, creating \\Imperial\MicroStation\WorkSets\Taos.cfg. The administrator retains the defaults for the Standards and Data location, so the Create WorkSet tool creates subdirectories \\Imperial\MicroStation\WorkSets\Taos\Standards and \\Imperial\MicroStation\WorkSets\Taos\Dgn to hold the standards and design files for this WorkSet.