

# CTP Validation and Install Instructions

By:



# CTP Validation and Install Instructions

## Table of Contents

Table of Contents _____	2
Summary _____	2
Previewing a CTP _____	3
Validating Program/File Changes _____	4
Documenting Program/File Changes _____	5
Option 1: Document Changed Files ONLY _____	5
Option 2: Document Changed Files and Version Numbers _____	5
Option 3: Document Changed Files, Versions and Programs Affected by Library Changes _____	5
Option 4: Document Changed Files, Versions, Programs that use Changed Libraries, and Programs that use INVOKEable Programs that have Changed _____	6
Option 5: Complete Documentation Including Actual File Change Analysis _____	7
Installing the CTP _____	8
Setting the cksum Value in a File _____	10
Legend of Terms and Abbreviations _____	12

## Summary

The procedures for applying any CTP will consist of: 1) running the lasetup program in preview mode; 2) validating all program/file changes; 3) documenting all program/file changes and 4) installing the CTP.

This documentation includes how to find programs that must be compiled with the change of a library file and it includes instructions on how to find all programs that use an INVOKEable program. See Documenting Program/File Changes – [Option 3](#) and [Option 4](#).

Because the actual instructions for all of this are not documented well in the Lawson documentation this document will provide all the required information needed to perform these tasks.

# CTP Validation and Install Instructions

## Previewing a CTP

Step	Command(s)	Description
1	N/A	Download and save the CTP file (usually named something like: 8.0.3_patch_CTP#.tar) to your system.
2	Use FTP or on Windows Map a Drive...	Transfer the tar file to the Lawson server.
3	mkdir /tmp/pathes cd /tmp/patches mkdir <CTP#>	Create a directory for the patch someplace on your server. I'll use /tmp/patches for my examples. Then create a sub-directory with the CTP Number.
4	mv <filename> /tmp/patches/<CTP#>/	Move or copy the file from the upload location in Step 2 to the new directory you've just created.
5	cd /tmp/patches/<CTP#>	Change your working directory to the directory you created in Step 3.
6	tar -xvf 8.0.3_patch_<CTP#>.tar	Un-tar the .tar file. You will get three files: 1. .readme file 2. Versions file and 3. patch.tar.Z file
7	perl \$GENDIR/bin/lasetup patch -preview <PL> perl %GENDIR%/bin/... for Windows	Run Lawson's lasetup in preview mode. This will check all the versions and checksum data for each file and report the results to your screen. After the information is reported to your screen you will be asked "...commit these changes to your product line? (y or n)". Answer "n" to this question as you need to do some further validation first. Then answer "y" to the next "are you sure you want to exit" question.
8	cp \$LAWDIR/<PL>/Admin/preview.log cp %LAWDIR%/... ./ for Windows	IMMEDIATELY copy the preview.log file to the /tmp/patches/<CTP#>/ directory for analysis. If you wait on this someone else can overwrite this file with their preview. <b>DO NOT WAIT.</b>

This completes the Preview

# CTP Validation and Install Instructions

## Validating Program/File Changes

Step	Command(s)	Description
1	<code>cd /tmp/patches/&lt;CTP#&gt;</code>	Change your working directory to the directory you created in Step 3 of Previewing a CTP (see above).
2	<pre> egrep "Patch Version  Yes " preview.log or egrep "Patch Version  Yes " preview.log \ &gt; modified.log or egrep "Patch Version  Yes " preview.log \   lashow </pre> <p>Please see special notes in regards to those files that have older versions and show the "M" indicator next to the "Existing Version" of the file.</p>	<p>You can either open the preview.log file using your favorite editor or run the simple egrep command shown here.</p> <p>If you are using an editor look for the word "Yes" under the Patch In column.</p> <p>If you are using the egrep command you may want to either redirect the output to another file (like in the second command) then look through that file or pipe the output to lashow (like in the third command).</p> <p><i>Special Notes:</i></p> <p><i>Some of the files that show they will be replaced may have older versions than what is installed on the system and a capital "M" will show after the "Existing Version" in the listing. This means the file has been modified from the original and will be replaced because of this.</i></p> <p><i>Since it is not desirable to overwrite a newer version simply because the existing version has been modified I've created a way to set the cksum value in the top of the file to indicate that there have been no modifications to the file.</i></p> <p><i>There is a reason behind this and requires further explanation which can be found below under the heading of "<a href="#">Setting the cksum Value in a File</a>".</i></p>
3	N/A	Review your findings. See <a href="#">Documenting Program/File Changes</a> below on what to do with what you find.

This completes the Validating Program/File Changes

# CTP Validation and Install Instructions

## ***Documenting Program/File Changes***

Once you have identified the File(s) that will be changed with the application of the CTP you can do a few things:

1. You can simply document that these are the files that will change
2. You can document the old and new version numbers of the files
3. You can do either of the above AND note any programs that must be recompiled because of changes to PD or WS library files.
4. In addition to number 3 you can also document which programs use the modified program as an “Invoked” program (if the modified program is “INVOKEable”).
5. You can do all of the above and take it one more step and actually extract the files out of the patch.tar file that show they have been changed and compare these “new” files with the existing files on your system to see the exact impact.

Since performing option number 5 (in essence all of the above options) really does not take too much effort if you have the instructions. And the fact that it would be good for your QA person to know exactly what the impact of the CTP will be to their unit testing and the effect this CTP will have on the “End-To-End” test. We’ll let the “volume” of changed files make that decision for now. As we will be applying CTPs we may want to ask your QA person what he wants to see.

At a minimum you should include all the items of option number 4 in the documentation. And optionally include the actual changes derived from the additional tasks of option number 5.

Here are the steps required for each of the options as found above:

### **Option 1: Document Changed Files ONLY**

Simply list the files that you show have changed in the documentation

### **Option 2: Document Changed Files and Version Numbers**

Simply list the files changed and the old/new versions in the documentation

### **Option 3: Document Changed Files, Versions and Programs Affected by Library Changes**

List the files changed with the old/new versions and then to find the programs that are affected by the changes to library (pdlib and wslib) files you can use the following command to extract the information:

```
rngdbdump gen pgmlib -f systemcode programcode -v productline=<PL> \  
libname=<LIB>
```

This will show you a list of System Codes and Program Codes that are used by the modified library.

# CTP Validation and Install Instructions

## Option 4: Document Changed Files, Versions, Programs that use Changed Libraries, and Programs that use INVOKEable Programs that have Changed

Do all what is suggested in Option #3 to get all the required information *except* the information on INVOKEable Programs.

First you'll want to check if the program is an INVOKEable On Line Update program. You can do this one of two ways:

1. Run Lawson's pgmdef for the product line, system code and program code. When you have highlighted the program code in this form press the Define key (<f6>) and select "A. Program" from the menu. The second line from the bottom has the label "Can Be Accessed Via INVOKE". If the value to the right of this label is "Yes" then this is an INVOKEable program.
2. Run this command:  

```
rngdbdump gen program -f systemcode programcode iscalled -v \  
productline=<PL> programcode=<PROGRAM> iscalled=1
```

The output will show you the system code, program code and either a zero (0) or one (1). The display of the system code and program code are just to be sure you entered it correctly. If the IsCalled value is 1 then this is an INVOKEable Program.

After you know that the program is INVOKEable you can then run Lawson's command to produce a listing of all the programs that INVOKE this program. Enter this command:

```
lstinvk <PL> <PROGRAM>
```

And you will see a listing of "qcompile" commands for all programs that INVOKE this program. You can then provide this information in the testing documentation.

The reason you would want to know the programs that INVOKE the modified program is that if an INVOKEable program is changed then all programs that use this program "should" be tested.

# CTP Validation and Install Instructions

## Option 5: Complete Documentation Including Actual File Change Analysis

Do all what is suggested in Option #4 to get all the required information to document all files/programs affected by the change to the source files in this CTP.

Then you will extract the changed files from the patch.tar file and run the diff command on the old and new files to find the changes. You should be familiar with the diff command and how it works to perform this Option.

Here are the steps to extract the files from the tar file:

Step	Command(s)	Description
1	<code>cd /tmp/patches/&lt;CTP#&gt;</code>	Change your working directory to the directory you created in Step 3 of Previewing a CTP (see above).
2	<pre>1. tar -xvf patch.tar 2. tar -xvf patch.tar &lt;FILE&gt; &lt;FILE&gt; ...</pre>	You can either: 1. Extract ALL the files from patch.tar or 2. Extract ONLY the files you need from the patch.tar file Remember to include in <FILE> the directory/filename for the extract.

After you have extracted the files you would then run a “diff” on each file that shows to be modified. An example of the command would be (assuming you’re working directory is set still from Step 1 just above):

```
diff ./acsrc/AC10PD $LAWDIR/test/acsrc/AC10PD | lashow
      (use %LAWDIR%/... for Windows)
```

Then you can do a base analysis on the changes to understand the impact.

This completes the Documenting Program/File Changes

# CTP Validation and Install Instructions

## Installing the CTP

Lawson's documentation, while it works, is too long for the simplicity of this process. Here are the exact steps required.

I'm including the steps 1 through 6 from "[Previewing a CTP](#)" from above just in case you use this page ONLY for installing the CTP. If you have already previewed the CTP you can start at step number 7 here.

Step	Command(s)	Description
1	N/A	Download and save the CTP file (usually named something like: 8.0.3_patch_CTP#.tar) to your system.
2	Use FTP or on Windows Map a Drive...	Transfer the tar file to the Lawson server.
3	<pre>mkdir /tmp/pathes cd /tmp/pathes mkdir &lt;CTP#&gt;</pre>	Create a directory for the patch someplace on your server. I'll use /tmp/pathes for my examples. Then create a sub-directory with the CTP Number.
4	<pre>mv &lt;filename&gt; /tmp/pathes/&lt;CTP#&gt;/</pre>	Move or copy the file from the upload location in Step 2 to the new directory you've just created.
5	<pre>cd /tmp/pathes/&lt;CTP#&gt;</pre>	Change your working directory to the directory you created in Step 3.
6	<pre>tar -xvf 8.0.3_patch_&lt;CTP#&gt;.tar</pre>	Un-tar the .tar file. You will get three files: 4. .readme file 5. Versions file and 6. patch.tar.Z file
7	<pre>perl \$GENDIR/bin/lasetup patch -preview &lt;PL&gt; perl %GENDIR%/bin/... for Windows</pre>	Run Lawson's lasetup in non-preview mode. This will check all the versions and checksum data for each file and if the file has changed it will backup the original file and replace the existing file with the file from this patch then it will report the summary results to your screen.
8	<pre>cp \$LAWDIR/&lt;PL&gt;/Admin/preview.log cp %LAWDIR%/... ./ for Windows</pre>	The appmetaload command will load any library definitions, menus, messages and other items that are NOT source code related. The appmetaload WILL NOT make any modification to the database.
9	<pre>perl \$GENDIR/bin/patchcompile \ -s compile.sh -u &lt;PL&gt; &lt;CTP#&gt; perl %GENDIR%/bin/... for Windows</pre>	The patchcompile command (with the -s option) will produce a shell script containing all the qcompile commands needed to compile all programs affected by the install of this CTP.

After you have completed all the steps above and before you actually execute the contents of the compile.sh file created in step 9 you'll need to check the "readme.html" file to see what database (and "other") changes are required, make the changes, build and reorg the dictionary (if necessary).

# CTP Validation and Install Instructions

Here are the instructions to perform this task:

You can either download the 8.0.3\_patch\_<CTP#>.readme.html file from the patches directory (see step 5 above) and view the file with your browser or you can go back to Lawson's support site and in the area where you downloaded the CTP file there is a link for "Info" under the "Read Me" column heading. This will display the same html document in your browser and is a bit easier than downloading the html file from the Lawson machine.

Page down (or find) to section "5.2 Special Notes" and follow all the instructions contained therein. These are the dbdef changes as well as other changes (for example: in CTP 22694 there is a requirement to create a "touch" file) for this CTP.

You may notice that a lot of these changes have already been done – that is because if any of the CTPs listed have been applied or if any of them are included in the latest MSP on your system then this task will have already been completed.

After you have made the required changes you may need to perform a "blddbdict" and "dbreorg" on the product line dictionary. I'm not going to provide those instructions here – if you need help contact someone who has done this before.

Finally...

After you have made all of the changes listed in the Special Instructions section and have performed a dbreorg, if needed, you can submit the compile commands in the compile.sh file created in step 9 above:

```
. ./compile.sh
```

The "dot-space-dot-slash" before the "compile.sh" will cause each command within the compile.sh file to be executed one at a time without having to make the .sh file executable.

Monitor your compile and check for .err files like you would do with any program recompile. If there are issues you may need to contact Lawson.

This completes the Installing the CTP

# CTP Validation and Install Instructions

## ***Setting the cksum Value in a File***

Overview:

When either a Preview or Install is run for a CTP the system will check each file for two things:

1. The Difference between the Existing Version of the file and the Version of the file delivered with the CTP. By Default the newer version should be applied and
2. If the file has been modified since it was installed. This is done with a file “cksum” that is stored in the file header.

Then if the delivered file contains a newer version or if the Existing File has been modified (no matter what version) it will be replaced by the lasetup program.

In the cases where you have changed the source code of a file directly (as we have with OEOIL) we do not want an older version overwriting a newer version simply because we have modified it.

In order to prevent this I have created a perl script that will check and set the cksum on a file to so that the lasetup program does not know it has been modified and will then look at only the version differences (the way it should be).

Details:

The header of each Lawson file contains the file name, version number and a multi-digit value between “<” and “>”. This value is the cksum which is calculated by the lasetup program and applied to each file at the time of install. The algorithm used to create this number is stored in a function inside lasetup called “cksum”.

lasetup will: 1) remove the cksum number from the file and then run the rest of the file through the function that calculates the cksum. After that it will compare the two numbers and if they are different it will know there have been changes to the originally installed file.

A custom perl script was created that will calculate a new cksum, compare with the existing cksum and if they are different will ask if you’d like to update the file with the new cksum.

The ONLY reason this would be done would be in the event that you have modified a standard Lawson delivered file and you do not want it to come up as a file that will be updated with the install of a CTP unless the file on the CTP media has a newer version.

So now what?

The step of setting the cksum value should be moved up into the process of the code modification, but since we’re just introducing that at this point we’ll need the “what to do” explained a bit here.

# CTP Validation and Install Instructions

(So now what – continued)

When you perform a preview and notice files (mostly library files) that show an Existing Version that is newer than the Patch Version and there is an “M” indicator next to the Existing Version you’ll need to check if the file has been modified by you.

Somewhere in the header of the file should be documentation of the change by you. Or you may just know that it was changed because you changed it. In any event if the file has been modified and the ONLY reason it is showing up on the list of files to be replaced is because it was modified then you’ll want to set the cksum value on the file to prevent it from showing up in the “to-be-replace” list of files.

Step	Action/Command(s)	Description
1	<pre>From the command line type:  perl \$GENDIR/bin/updcksum productline \ directory filename  (Use %GENDIR%/bin... for Windows)  and press Enter</pre>	This will perform the same cksum validation that lasetup performs and if there are differences will as it you want to update the cksum value in the file (see example below).

Here is an example on the OEOIL file (on Unix):

```
/export/home/tkastle/> perl $GENDIR/bin/updcksum test pdlib oeoil
```

```
For the file: /lawson/law/test/pdlib/OEOIL *****
cksum value in header of file: 3638786872
cksum value calculated for file: 269116563
```

```
Do you want to update the file with the calculated cksum? (y/n) _
```

You then have a choice to change the header of the file or not. If your desire is to change the file then enter “y” otherwise enter “n”. You will receive a message letting you know the file was changed or not changed.

If the cksum on the file is not different you will see this:

```
/export/home/tkastle/> perl $GENDIR/bin/updcksum test pdlib oeoil
cksum does not need to be changed
```

This completes Setting the cksum Value in a File

# CTP Validation and Install Instructions

## *Legend of Terms and Abbreviations*

<b>Term/Abbreviation</b>	<b>Description</b>
CTP	Critical Transfer Pack (a Lawson term)
<CTP#>	Means to replace this entire string, including “<” with the actual CTP # in the command
<PL>	Means to replace this entire string, including “<” with the Product Line name in the command
<LIB>	Means to replace this entire string, including “<” with the library (pdlib or wslib) file name in the command
<FILE>	Means to replace this entire string, including “<” with the full file name (which includes the directory if one exists for this file) in the command
<PROGRAM>	Means to replace this entire string, including “<” with the actual program code in the command. This is NOT the filename (which would include a suffix like PD or WS or .scr) this is ONLY the program code. For example if the filename showing it has been modified is AC10PD then the program code would be AC10.
\	A “backslash” in a Unix command means the command continues on the next line.