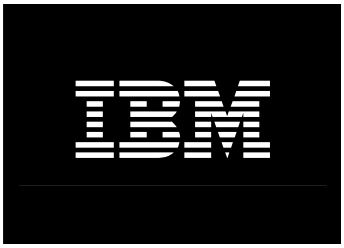
Technical paper
April 2007



UpdateXpress System Pack Advanced Customization

By

Rob Christner, Software Engineer, System x and BladeCenter Update Software Development, IBM

Peter Donovan, Senior Software Engineer, System x Update Software Development, IBM

Contents

Introduction

First released in the fourth quarter of 2006, IBM UpdateXpress System Packs (UXSP) are key deliverables that enable customers to simplify their systems management procedures when updating IBM System x and BladeCenter servers.

This paper focuses on how customers can take advantage of the modular architecture of UXSPs built on XML, and exploit the flexibility that comes with such an architecture. This paper also gives an introduction into UXSPs, the UpdateXpress System Pack Installer (UXSPI), and the associated XML files. By simply editing XML data, a customer can experience the benefit of having a product tailored to their specific environment.

This paper was written for use with the UXSPI version 1.20 and the IBM System x update XML schema version 3.0 (as defined by the *ibmSchemaVersion* field in the UXSP XML file).

Highlights

IBM UpdateXpress System Packs (UXSP) are definitions of integration-tested bundles of online Windows and Linux firmware and device driver updates for System x and BladeCenter servers.

UpdateXpress System Pack

IBM® UpdateXpress System Packs (UXSP) are definitions of integration-tested bundles of online Windows® and Linux® firmware and device driver updates for System x® and BladeCenter® servers. They facilitate the download and install of drivers and firmware for a given system and provide a complete set of updates which have been tested together by IBM. UXSPs allow customers to apply co-requisite and dependent code components together and provide a convenient methodology for keeping systems up to date.

UXSPs are provided for select machine-types. Separate UXSPs are provided for Windows and each of the Linux distributions. UXSPs are available at system GA and refreshed periodically.

UXSPs can be installed with the UXSPI, or with the Update Manager task in IBM Director. This paper focuses primarily on the use of the UXSP with the UXSPI. However, many of the same concepts apply to Update Manager as well.

To install a UXSP, the UXSP applicable to the desired system must first be downloaded from http://www.ibm.com/support. Multiple system packs can be downloaded for different machine-types into the same directory. When the installer is launched, it detects the machine-type and uses the correct content for the machine-type. In some cases, there may be common files between system packs. When this occurs, it is safe to overwrite the existing file. The files with the same name will overwrite one another as they are downloaded into the same directory.

UpdateXpress System Pack Installer

The UXSPI is used to apply UXSP updates to a system.

This information is regarding UpdateXpress System Pack Installer (UXSPI) version 1.20.

The UXSPI is used to apply UXSP updates to a system. Regardless of whether the UXSPI is run from the command-line interface (CLI) or the graphical user interface (GUI) (on Windows only), it follows the same step-by-step process:

Highlights

1. Collects an inventory of the system's drivers and firmware
2. Queries the current working directory or a specified location for a list of applicable UXSPs
3. Compares the inventory to the applicable list of updates in the UXSP and recommends a set of updates to apply
4. Deploys those updates to the system

The UXSPI tool supports two modes of operation:

- Update mode, which performs the four tasks.
- Comparison mode, which performs the first 3 tasks: inventory, query, and compare. The user then has the freedom to do the deploy step later using a method of their choice.

Note: To get more detailed information logged during a run of the UXSPI set the `DSA_LOGLEVEL` environment variable to `4`.

UpdateXpress System Pack XML definition file

The UpdateXpress System Pack (UXSP) is defined by an XML descriptor file. The name of this file is usually in a form similar to:

```
ibm_utl_uxsp_bksp38a-1.00_windows_32-64.xml
```

The goal of this XML file is to define the set of parameters for the entire UXSP as well as provide a catalog of the individual update packages included in the UXSP.

Refer to the paper entitled "New IBM xSeries firmware and device driver filename convention" for details about the filenames. The UXSPI parses this descriptor file and makes decisions based on the information contained within. The XML standard that is used for this description conforms to the DMTF CIM-XML standard. The goal of this XML file is to define the set of parameters for the entire UpdateXpress System Pack as well as provide a catalog of the individual update packages included in the UXSP.

Highlights

For UXSPs there should be only one instance of the SystemUpdateProduct class, and there should be at least one instance of the SystemUpdateElement class.

There are class definitions which describe the available fields and the possible values that can be contained in those fields. Some of the field definitions are inherited by a higher level super class that is not defined in this file, but is defined in the UXSPI application itself. There are two class definitions in the UXSP XML file. First, there is an IBMPSG_SystemUpdateProduct (referred to as SystemUpdateProduct for the remainder of this paper) defined that contains data pertinent to the overall UXSP. Second, there is an IBMPSG_SystemUpdateElement (referred to as SystemUpdateElement for the remainder of this paper) defined that contains data specific to an individual update package (e.g. BIOS).

Any number of instances of a specified class can then be subsequently described. For UXSPs there should be only one instance of the SystemUpdateProduct class, and there should be at least one instance of the SystemUpdateElement class. Figure 1 shows a typical overall layout of the XML file.

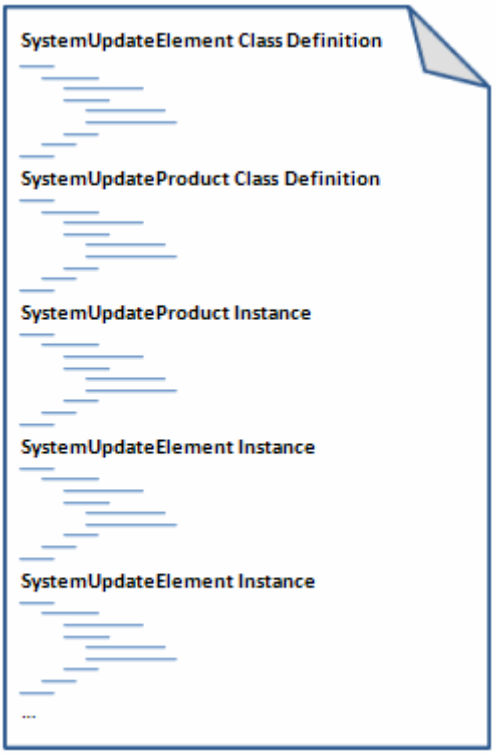


Figure 1: Layout of UXSP XML file

Highlights

The class definition sections of the XML file should not be modified for either the SystemUpdateProduct or the SystemUpdateElement. The sections can be identified by these two sequences of XML tag s.

```
<IPARAMVALUE NAME="NewClass">
        <CLASS NAME="IBMPSG_SystemUpdateProduct"
SUPERCLASS="CIM_Product">

<IPARAMVALUE NAME="NewClass">
        <CLASS NAME="IBMPSG_SystemUpdateElement"
SUPERCLASS="CIM_Product">
```

The SystemUpdateProduct instance can be identified by this sequence of XML tags.

```
<IPARAMVALUE NAME="NewInstance">
        <INSTANCE CLASSNAME="IBMPSG_SystemUpdateProduct">
```

It contains the support information for the UXSP. Examples of this support information are applicable machine types, operating systems, version, and filename. The key values that lend to customization are identified by the XML tags below.

- *Filename* – a string that represents the filename of this XML file (for example, `ibm_utl_uxsp_bksp38a-1.00_windows_32-64.xml`).
- *Version* – a string that represents the Version of this UXSP (for example, `1.2.3.4`).
- *applicableMachineTypes* – an array of strings that represent the supported machine types. The strings must conform to a particular string format, `System Name-[MachineType]-`, where `MachineType` is the 4 digit numerical machine type (for example, `IBM BladeCenter HS20-[8843]`).
- *applicableOperatingSystems* – contains an array of numerical values that correspond to the supported operating systems listed in Table 1. Note that this data is in a different format than the *applicableOperatingSystems* field in the SystemUpdateElement definition described in the next section.

Highlights

Description	Enumeration Value
Windows 2000	102
Windows 2003	103
Windows XP	105
RHEL 3	201
RHEL 4	202
SLES 9	205
SLES 10	206

Table 1: Operating system enumeration values

…these instances should be copies of the XML files included with each individual update package. The UXSPI does not read the individual package XML files, but instead relies solely on the duplicated information in the UXSP XML file.

The SystemUpdateElement instances can be identified by this sequence of XML tags

```
<IPARAMVALUE NAME="NewInstance">
        <INSTANCE CLASSNAME="IBMPSG_SystemUpdateElement">
```

They contain the support information for each of the individual update packages included in the UXSP. It is important to note that these instances should be copies of the XML files included with each individual update package. The UXSPI does not read the individual package XML files, but instead relies solely on the duplicated information in the UXSP XML file. Any customizations made to individual update packages XML files will not be seen by the UXSPI. The Update Manager task in IBM Director can interpret the individual update package XML file as well as the UXSP XML file. Therefore, if a customized UXSP is expected to be used in Update Manager, it is recommended to keep these XML files in sync.

Update package XML definition file

Since each update package XML is duplicated in the UXSP XML, the information in this section applies to both the individual update package XML file and the SystemUpdateElement instance in the UXSP XML file.

The key values that lend to customization are identified by the XML tags below.

Highlights

- *Name* – a string that defines the name of the update (for example, `Flash BIOS Update`).
- *TargetOperatingSystem* – a numerical value that represents the generic target operating system (for example, `58`). The *applicableOperatingSystems* and *orderedOSList* specify in detail the operating system supported.
    - 36 – Linux
    - 58 – Windows
- *Version* – a string that represents the architected System x version string in the format, `Version version -[build-version]-`. The version format embeds the update version and the build number in a string (for example, `Version 1.08 –[ZUE147B-1.08]-`).
- *applicableMachineTypes* – an array of strings representing the supported machine types and conforms to a particular string format, `System Name-[MachineType]-`, where `MachineType` is a 4 digit numerical machine type (for example, `IBM eServer xSeries 200-[8478]-`).
- *applicableOperatingSystems* – an array of strings that identifies which operating systems are supported by this update. These values must be from the following list:
    - Windows 2000
    - Windows 2003
    - Windows 2003 x64
    - Linux
- *Description* – a string that represents the description of the update. Usually this is the same as the *Name* field (for example, `Flash BIOS Update`).
- *rebootRequired* – a numerical enumeration (for example, `3`) corresponding to the following values:
    0. No reboot required -- update effective immediately
    1. Reboot forced by update
    2. Reboot required immediately after update
    3. Reboot required to take effect
    4. Logoff required to take effect
- severity – a numerical enumeration (for example, `2`) corresponding to the following values:

Highlights

0. Initial release
1. Critical
2. Recommended
3. Non-Critical

- *unattendedInstallCommandLine* – a string that represents the command line for installing this update in an unattended fashion (for example, `ibm_fw_bmc_g9et14a_windows_i386.exe –s –a –s`).
- *type* – a numerical value that represents the update type corresponding to the values below. The UXSPI filters updates based on this value, so that the user can select on updates of a certain type to display and apply.
    1. Online firmware
    2. Offline firmware
    3. Device driver
- *xmlFilename* – a string that represents the name of the update XML file (for example, `ibm_fw_bmc_g9et14a_windows_i386.xml`).
- *orderedMachineTypeList* – an array of strings that represent an ordered list of applicable machine types. This list is used in conjunction with the *orderedOSList* and *orderedOSArchitectureList* to identify the combination of applicable machine types and operating systems. Unlike *applicableMachineTypes*, only the machine type is listed (for example, `8678`).
- *orderedOSList* – an array of numerical values that represent an ordered list of applicable operating systems. This list is used in conjunction with the *orderedMachineTypeList* and *orderedOSArchitectureList* to identify the combination of applicable Machine Types and Operating Systems. See Table 1 for a list of possible values.
- *orderedOSArchitectureList* – an array of numerical values that represent an ordered list of operating system architecture types. This list is used in conjunction with the *orderedMachineTypeList* and *orderedOSList*. Appropriate values include:
    1. x32 bit (i386)
    2. x64 bit (EM64T/AMD64)
- *payload* – a string that identifies the update filename (for example, `ibm_fw_bmc_g9et14a_windows_i386.exe`).
- *updateSelection* – a string that defines the selection for installation of the update by default. Appropriate values include:
    - Never

Highlights

- Always
- Auto

Several of the XML fields are defined as ordered qualified lists. An ordered qualified list refers to two or more array fields that qualify each other. Each array element will have a corresponding element in each other array. Thus, the size of each array in an ordered qualified list will be the same. Figure 2 shows this relationship graphically.

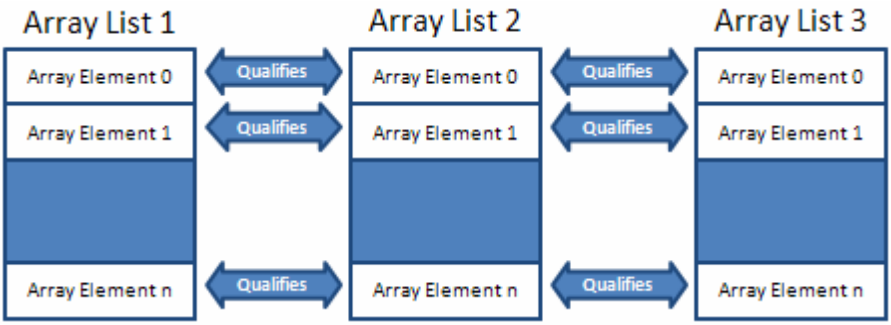


Figure 2: An ordered qualified list composed of three arrays

As an example, an update is supported on machine type “1234” running operating system “100” in a 32-bit architecture and it also supports machine type “4321” running operating system “200” both in a 32 and 64-bit architecture, where “1” represents 32-bit and “2” represents 64-bit. This is an example of a representation in an ordered qualified list.

An ordered qualified list refers to two or more array fields that qualify each other. Each array element will have a corresponding element in each other array.

```
<PROPERTY.ARRAY NAME="orderedMachineTypeList"
TYPE="string">
  <VALUE.ARRAY>
    <VALUE>1234</VALUE>
    <VALUE>4321</VALUE>
    <VALUE>4321</VALUE>
  </VALUE.ARRAY>
</PROPERTY.ARRAY>
<PROPERTY.ARRAY NAME="orderedOSList" TYPE="uint32">
  <VALUE.ARRAY>
    <VALUE>100</VALUE>
```

Highlights

```
      <VALUE>200</VALUE>
      <VALUE>200</VALUE>
    </VALUE.ARRAY>
  </PROPERTY.ARRAY>
  <PROPERTY.ARRAY NAME="orderedOSArchitectureList"
  TYPE="uint32">
    <VALUE.ARRAY>
      <VALUE>1</VALUE>
      <VALUE>1</VALUE>
      <VALUE>2</VALUE>
    </VALUE.ARRAY>
  </PROPERTY.ARRAY>
```

Customization instructions and examples

The following sections describe various useful examples of how a UXSP can be customized.

Remove an update package from a UXSP

Removing specific firmware and device driver updates from the UXSP streamlines the update process and provides assurance that the necessary updates are applied.

This scenario describes how to remove an update package from a UXSP. By removing the update from the UXSP it will no longer be displayed in the user interface of the UXSPI, and will not be installed. This scenario can be useful if it is known that no system in the environment has a particular piece of hardware. For example, if there are no Remote Supervisor Adapters in the environment, removing the Remote Supervisor Adapter firmware and device driver updates from the UXSP streamlines the update process and provides assurance that the necessary updates are applied. Use the following steps to complete this scenario.

1. Optionally delete the update and its corresponding files from the directory that contains the UXSP that requires customization.
2. Locate the XML description for the UXSP and open it in a text editor.
3. Locate the SystemUpdateElement instance in the UXSP XML that corresponds to the update that is being removed. One way to locate the update in the XML is to search for the filename.
4. Delete the entire SystemUpdateElement instance for the update that is being removed. This is accomplished by removing everything contained within the <SIMPLEREQ> XML tags for the particular

Highlights

instance that is being replaced. Make sure to also delete the <SIMPLEREQ> tags.

```
<SIMPLEREQ>
  <IMETHODCALL NAME="CreateInstance">
    <LOCALNAMESPACEPATH>
      <NAMESPACE NAME="root"/>
      <NAMESPACE NAME="cimv2"/>
    </LOCALNAMESPACEPATH>
    <IPARAMVALUE NAME="NewInstance">
      <INSTANCE CLASSNAME="IBMPSG_SystemUpdateElement">
        <PROPERTY NAME="Name" TYPE="string">
          <VALUE>IBM BIOS Flash Update</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementID" TYPE="string">
          <VALUE>BKE1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementState"
        TYPE="uint16">
          <VALUE>1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="TargetOperatingSystem"
        TYPE="uint16">
          <VALUE>58</VALUE>
        </PROPERTY>
        <PROPERTY NAME="Version" TYPE="string">
          <VALUE>Version 1.22 -[BKE121N-1.22]-</VALUE>
        </PROPERTY>
        …
      </INSTANCE>
    </IPARAMVALUE>
  </IMETHODCALL>
</SIMPLEREQ>
```

5. Close and save the UXSP XML file.

Highlights

While this scenario diminishes the benefits of having an integration-tested bundle of updates, it can prove useful if a critical update has released, but has yet to be included in a new UXSP version.

Replace an update package in a UXSP with a newer version

This scenario describes how to replace an update package in a UXSP with a newer version of the same package. While this scenario diminishes the benefits of having an integration-tested bundle of updates, it can prove useful if a critical update has released, but has yet to be included in a new UXSP version. Use the following steps to complete this scenario.

1. Copy the new version of the update package, and its corresponding XML file, change history (.chg) file, and readme file into the directory that contains the UXSP that requires customization.
2. Optionally delete the previous version of the update and its corresponding files from the directory.
3. Locate the XML description for the UXSP and open it in a text editor.
4. Locate the SystemUpdateElement instance in the UXSP XML that corresponds to the version of the update that is being replaced. One way to locate the update in the XML is to search for the filename.
5. Delete the entire SystemUpdateElement instance for the update that is being replaced. This is accomplished by removing everything contained within the `<SIMPLEREQ>` XML tags for the particular instance that is being replaced. Make sure to also delete the `<SIMPLEREQ>` tags.

```
<SIMPLEREQ>
  <IMETHODCALL NAME="CreateInstance">
    <LOCALNAMESPACEPATH>
      <NAMESPACE NAME="root"/>
      <NAMESPACE NAME="cimv2"/>
    </LOCALNAMESPACEPATH>
    <IPARAMVALUE NAME="NewInstance">
      <INSTANCE CLASSNAME="IBMPSG_SystemUpdateElement">
        <PROPERTY NAME="Name" TYPE="string">
          <VALUE>IBM BIOS Flash Update</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementID" TYPE="string">
          <VALUE>BKE1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementState"
        TYPE="uint16">
```

Highlights

```
            <VALUE>1</VALUE>
          </PROPERTY>
          <PROPERTY NAME="TargetOperatingSystem"
          TYPE="uint16">
            <VALUE>58</VALUE>
          </PROPERTY>
          <PROPERTY NAME="Version" TYPE="string">
            <VALUE>Version 1.22 -[BKE121N-1.22]-</VALUE>
          </PROPERTY>
          …
        </INSTANCE>
      </IPARAMVALUE>
    </IMETHODCALL>
  </SIMPLEREQ>
```

6. Locate the XML description for the new update package to include in the UXSP, and open it in a text editor.
7. Copy the entire contents of the file starting with the `<SIMPLEREQ>` XML tag and ending with the `</SIMPLEREQ>` tag. Make sure to also include both `SIMPLEREQ` tags.
8. Paste the contents from the individual update XML file into the UXSP XML file in the location where the instance of the previous version of the update existed. Make sure there is one and only one set of `<SIMPLEREQ>` tags for this instance.
9. Close and save the UXSP XML file. Close the individual update XML file.

Add a compatible update package to a UXSP

The update that is being added must already be supported by the UXSPI in order for this scenario to be effective.

This scenario describes how to add a new compatible update package to a UXSP that does not contain a previous version of the update. The update that is being added must already be supported by the UXSPI in order for this scenario to be effective. Use the following steps to complete this scenario.

1. Copy the new update package, and its corresponding XML file, change history (.chg) file, and readme file into the directory that contains the UXSP that requires customization.

Highlights

2. Locate the XML description for the new update package to include in the UXSP, and open it in a text editor.
3. Copy the entire contents of the file starting with the `<SIMPLEREQ>` XML tag and ending with the `</SIMPLEREQ>` tag. Make sure to also include both `SIMPLEREQ` tags.

```
<SIMPLEREQ>
  <IMETHODCALL NAME="CreateInstance">
    <LOCALNAMESPACEPATH>
      <NAMESPACE NAME="root"/>
      <NAMESPACE NAME="cimv2"/>
    </LOCALNAMESPACEPATH>
    <IPARAMVALUE NAME="NewInstance">
      <INSTANCE CLASSNAME="IBMPSG_SystemUpdateElement">
        <PROPERTY NAME="Name" TYPE="string">
          <VALUE>IBM BIOS Flash Update</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementID" TYPE="string">
          <VALUE>BKE1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementState"
        TYPE="uint16">
          <VALUE>1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="TargetOperatingSystem"
        TYPE="uint16">
          <VALUE>58</VALUE>
        </PROPERTY>
        <PROPERTY NAME="Version" TYPE="string">
          <VALUE>Version 1.22 -[BKE121N-1.22]-</VALUE>
        </PROPERTY>
        …
      </INSTANCE>
    </IPARAMVALUE>
  </IMETHODCALL>
</SIMPLEREQ>
```

4. Locate the XML description for the UXSP and open it in a text editor.

Highlights

5. Find the end of the last update instances. Each update instance is completely enclosed in a set of <SIMPLEREQ> tags. Therefore, the location to insert a new update instance will be after the last </SIMPLEREQ> tag.

```
…
        </SIMPLEREQ>
        Insert new update package definition here.
      </MULTIREQ>
    </MESSAGE>
</CIM>
```

6. Paste the contents from the individual update XML file into the UXSP XML file into this location. Make sure there is one and only one set of <SIMPLEREQ> tags for each update instance.
7. Close and save the UXSP XML file. Close the individual update XML file.

An offline update is an update package that is not applied inside of the native operating system environment of the intended system.

Add a non-compatible update package to a UXSP

Even though an update package conforms to the IBM specification (an unattended, command line, self contained package that can be completely described by an IBM XML file), it may not be compatible with the UXSPI, and thus is not included in a UXSP definition release. A common set of packages that fall in this category are termed “offline” updates. An offline update is an update package that is not applied inside of the native operating system environment of the intended system. One reason for an “offline” update is to support an update that must be applied in an operating environment (for example, DOS) other than what is currently running on the system. This is accomplished by wrapping a DOS floppy image that contains the DOS update utility and the firmware file inside of a specific shell that runs in the native operating system (for example, Windows). This specific shell includes a virtual DOS environment that it loads onto the file system, and then targets that location to boot into upon the next restart of the system. When the system restarts, it loads the DOS environment which has the update program scripted. Once the update is complete, the system is restarted again back into the native operating system. Because of limitations with this method (only one offline update can be scheduled at a time), these updates are intentionally not included in a UXSP.

Highlights

…only one offline update can be scheduled at a time.

However, it is possible to trick the UXSPI into thinking the update is a compatible update. This scenario describes how to add a single offline update to a UXSP. Adding more than one offline update is possible, but the second update will generate an error when it tries to schedule itself. This is due to the limitation that only one offline update can be scheduled at a time.

Since offline updates are not compatible with the UXSPI it is very likely that the UXSPI will not know how to determine if the update is applicable to the system. Starting with version 1.20 of the installer, if the installer is unable to find a particular firmware on the system, it assumes that the firmware is not installed, but it will show the firmware on the user interface and select it to be installed by default. The user interface will show the installed version of the firmware as "undetectable". Use the following steps to complete this scenario.

1. Follow the steps in the scenario "Add a compatible update package to a UXSP".
2. Offline firmware updates have a value for the "type" field defined in the XML that describes them as being offline firmware. This value must be changed in order for the update to work correctly. Locate the instance of the SystemUpdateElement for this particular update in the UXSP XML file (not the individual update XML file).
3. Within this particular instance locate the "type" XML field.

```
<PROPERTY NAME="type" TYPE="uint16">
  <VALUE>2</VALUE>
</PROPERTY>
```

4. Change the value to "1" to indicate this update is an online firmware update.

```
<PROPERTY NAME="type" TYPE="uint16">
  <VALUE>1</VALUE>
</PROPERTY>
```

Example: Add a SAS hard drive update to a Windows UXSP for IBM BladeCenter LS21

A practical example in which the previous scenario can be used is to add a hard disk drive microcode offline update package to a UXSP. Here are the steps that

Highlights

can be taken to accomplish this scenario for an IBM BladeCenter LS21 running Microsoft Windows 2003 Server:

1. Download the Windows UXSP for LS21 from http://www.ibm.com/support into a directory (e.g. `C:\uxsp\ls21_custom\`).
2. Download the UXSPI for Windows from http://www.ibm.com/support into `C:\uxsp\ls21_custom\`.
3. Download the SAS hard drive update program package (includes the .exe, .xml, .txt, and .chg files) from http://www.ibm.com/support into `C:\uxsp\ls21_custom\`.
4. Locate the XML description (`ibm_fw_sas_hdd_v101_windows-pq_i386.xml`) for the new update package to include in the UXSP, and open it in a text editor.
5. Copy the entire contents of the file starting with the `<SIMPLEREQ>` XML tag and ending with the `</SIMPLEREQ>` tag. Make sure to also include both `SIMPLEREQ` tags.

```
<SIMPLEREQ>
  <IMETHODCALL NAME="CreateInstance">
    <LOCALNAMESPACEPATH>
      <NAMESPACE NAME="root"/>
      <NAMESPACE NAME="cimv2"/>
    </LOCALNAMESPACEPATH>
    <IPARAMVALUE NAME="NewInstance">
      <INSTANCE CLASSNAME="IBMPSG_SystemUpdateElement">
        <PROPERTY NAME="Name" TYPE="string">
          <VALUE>IBM SAS Hard Disk Drive Update
          Program</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementID" TYPE="string">
          <VALUE>SSHD</VALUE>
        </PROPERTY>
        <PROPERTY NAME="SoftwareElementState"
        TYPE="uint16">
          <VALUE>1</VALUE>
        </PROPERTY>
        <PROPERTY NAME="TargetOperatingSystem"
```

Highlights

Offline firmware updates have a value for the "type" field defined in the XML that describes them as being offline firmware. This value must be changed in order for the update to work correctly.

```
          TYPE="uint16">
            <VALUE>58</VALUE>
          </PROPERTY>
          <PROPERTY NAME="Version" TYPE="string">
            <VALUE>Version 1.01 -[-1.01]-</VALUE>
          </PROPERTY>
          …
        </INSTANCE>
      </IPARAMVALUE>
    </IMETHODCALL>
  </SIMPLEREQ>
```

6. Locate the XML description for the UXSP (`ibm_utl_uxsp_basp03a-1.00_windows_32-64.xml`) and open it in a text editor (e.g. Notepad, Vi, or Emacs).
7. Find the end of the last update instances. Each update instance is completely enclosed in a set of `<SIMPLEREQ>` tags. Therefore, the location to insert a new update instance will be after the last `</SIMPLEREQ>` tag.

```
      …
      </SIMPLEREQ>
      Insert new update package definition here.
    </MULTIREQ>
  </MESSAGE>
</CIM>
```

8. Paste the contents from the individual update XML file into the UXSP XML file into this location. Make sure there is one and only one set of `<SIMPLEREQ>` tags for each update instance.
9. Close and save the UXSP XML file. Close the individual update XML file. Make sure to retain the .xml extension and format of these files.
10. Offline firmware updates have a value for the "type" field defined in the XML that describes them as being offline firmware. This value must be changed in order for the update to work correctly. Locate the instance of the SystemUpdateElement for this particular update in the UXSP XML file (not the individual update XML file).
11. Within this particular instance locate the "type" XML field.

Highlights

```
<PROPERTY NAME="type" TYPE="uint16">
  <VALUE>2</VALUE>
</PROPERTY>
```

12. Change the value to “1” to indicate this update is an online firmware update. This is how we trick the UXSPI into thinking this is a compatible update package.

```
<PROPERTY NAME="type" TYPE="uint16">
  <VALUE>1</VALUE>
</PROPERTY>
```

13. When the UXSPI is launched, the SAS hard drive update should now appear in the list of applicable updates as shown in Figure 3.

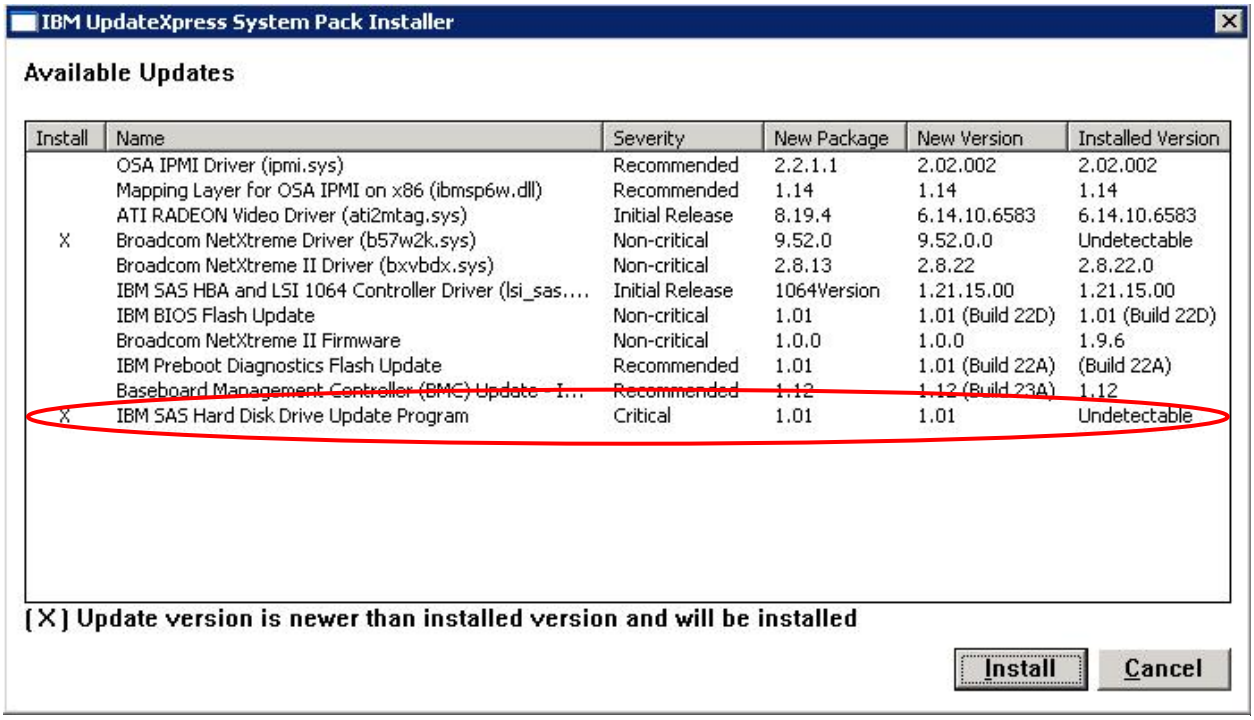


Figure 3: User interface of the UpdateXpress System Pack Installer showing the SAS hard drive update

Highlights

Modify the behavior or metadata of an update package

There are several key pieces of metadata described in the instances of the SystemUpdateProduct and SystemUpdateElement that can modify the default behavior when installing UXSPs. Some modifications will change how things are displayed in the user interface of the UXSPI, while others will actually change how the application behaves.

Some of the most useful modifications are those that change the support data.

Some of the most useful modifications are those that change the support data. These changes actually affect how the application behaves, because the installer ignores UXSPs or individual updates that are not supported on the system (machine type and operating system) that the installer is launched from. These modifications can include adding new system machine types that were not previously supported by the update or adding a different operating system. These modifications can either be made to the UXSP definition in the instance of the SystemUpdateProduct, or to individual updates in the instances of SystemUpdateElements. The installer first checks the SystemUpdateProduct data to make sure the UXSP XML file that it is reading is appropriate for the hardware it is running on. The installer will then proceed to the steps that read the individual update metadata. This metadata that is described in the SystemUpdateElement instances in the UXSP XML is used to determine if the individual update is intended for the system that the installer is running on. If it is not, the installer will ignore that update and continue to the next update. This is a common reason why an update may be included in the UXSP, but for some reason the update is not being displayed in the user interface of the UXSPI. If this data is being modified, and the update is still not showing on the user interface, it is likely that not all of the fields have been properly updated. For example, to add a new operating system the following fields must be modified according to the specification described in the previous sections.

- *targetOperatingSystem*
- *applicableOperatingSystems* (in both SystemUpdateProduct and SystemUpdateElement) Note that the format of this field differs between the SystemUpdateProduct definition and the SystemUpdateElement definition. Refer to the appropriate XML definition file sections in this paper for more information.
- *orderedOSList*

Highlights

The* unattendedInstallCommandLine *field is probably the single most important field described in the XML.

- *orderedOSArchitectureList*
- *orderedMachineTypeList*

Adding a new machine type to the supported list is similar to adding a new operating system in that fields will need to be modified in the SystemUpdateProduct as well as the SystemUpdateElement definitions.

Another field that when modified changes behavior of the update is the *unattendedInstallCommandLine*. This field is probably the single most important field described in the XML, as it defines the command line that the update package must be called with in order to perform the update. Modifying this field allows for specifying additional command line options that would result in a different behavior from the update package. An example of this is described in the section "Override the default behavior for Linux device driver update packages".
Other XML fields change how certain things are displayed in the UXSPI user interface. Examples of these fields are:

- *rebootRequired*
- *severity*
- *Name*
- *description*

Multiple UXSPs in a common directory

The UXSPI searches through the defined UXSP location (by default this is the current working directory, but it can be overridden by the `--local` command line option) for a UXSP XML file that supports the system that the installer is running from. The installer queries the system for its machine type and operating system and compares that data to what it reads from the SystemUpdateProduct *applicableMachineTypes* and *applicableOperatingSystems* fields in the UXSP XMLs in the search location. The installer will search the UXSP XML files in the search location and choose the most recent (determined by the *version* field in the SystemUpdateProduct) that supports the system. This allows a user to copy multiple UXSPs into the same directory, and the installer will automatically choose the appropriate one.

Highlights

The default operation for Linux device drivers included in a UXSP is to maintain the Linux distribution certified drivers whenever possible.

Override the default behavior of Linux device driver update packages

The default operation for Linux device drivers included in a UXSP is to maintain the Linux distribution certified drivers whenever possible. The command specified in the *unattendedInstallCommandLine* field in the XML is similar to

```
<PROPERTY NAME="unattendedInstallCommandLine" TYPE="string">
  <VALUE>brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz; tar xzf
brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz;
  ./install.sh --update -force-if-overridden;</VALUE>
</PROPERTY>
```

Running this command installs the driver that was previously overridden in a past installation. However, this default behavior does not allow an UXSP driver to install over a newer UXSP driver that is already installed. This field in the XML can be modified to change the behavior of Linux device driver packages. The following command can be used to override the Linux distribution version of the driver:

```
<PROPERTY NAME="unattendedInstallCommandLine" TYPE="string">
  <VALUE>brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz; tar xzf
brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz;
  ./install.sh --update --override;</VALUE>
</PROPERTY>
```

Once a driver has been overridden, the UXSPI will detect that the installed driver was not provide by the distribution. Thus, future UXSP installations can update the driver. To perform an unconditional install of the device driver (assuming a compatible device driver is available), change the *unattendedInstallCommandLine* in the XML to resemble the following command:

```
<PROPERTY NAME="unattendedInstallCommandLine" TYPE="string">
  <VALUE>brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz; tar xzf
brcm_dd_nic_tg3-3.58b_rhel3_32-64.tgz;
  ./install.sh --update --force;</VALUE>
</PROPERTY>
```

Highlights

This technique is useful for cases in which an updated device driver is a prerequisite for a firmware update. An example of this is the network interface card updates. In some Linux operating system versions, the driver that is supplied by the distribution does not support the firmware update process. Therefore, in order to successfully apply the firmware update package, the device driver must be migrated from the distribution provide driver.

Override the default behavior of update selection

The* updateSelection *field specifies whether this default behavior of selecting an update should be overridden.

Several types of firmware updates have been regarded as more sensitive than others. An example of this type of update is that of a Fibre Channel host bus adapter (HBA). To accommodate this scenario the *updateSelection* field was introduced in the update XML file. The default behavior of the UXSPI is to select updates if they are either at an older version than what is included in the UXSP, or if the installer was unable to determine the current version of the firmware or device driver. The *updateSelection* field specifies whether this default behavior of selecting an update should be overridden. It has three possible values:

- *Never* – this particular update will never be selected automatically. The user must use the interactive modes of the UXSPI to select this update and install it on the system.
- *Always* – this particular update will always be selected automatically. The user must use the interactive modes of the UXSPI to unselect this update so that it is not installed on the system.
- *Auto* – this particular update will follow the defaults rules as to whether or not it will be selected. Since *updateSelection* is an optional field in the update XML, this value has identical behavior as if the field was not specified.

The UXSPI also has a command line option that allows the user to ignore this field for the updates in the UXSP. By using the `-n` or `--new` options from the command line, the UXSPI will treat the updates as if the *Auto* value was defined for each *updateSelection*.

Highlights

Conclusion

This paper demonstrated how modifying UXSP XML files can allow for advanced customization of the UXSP tailored to customer's specific environments. The paper described how a customer can add or remove updates from a UXSP, or modify the behavior of updates included in a UXSP.

Highlights

For more information

For more information about the UpdateXpress System Pack Installer, please visit http://publib.boulder.ibm.com/infocenter/toolsctr/v1r0/index.jsp?topic=/com.ibm.xseries.tools.doc/update_tools_uxspi.html

For information regarding initial deployment using IBM systems management tools, please visit http://www-03.ibm.com/systems/management/sgstk.html

There is an online tools center available that provides an overview of IBM systems management tools. It can be found at http://publib.boulder.ibm.com/infocenter/toolsctr/v1r0/index.jsp

Information regarding the DMTF CIM-XML standard can be obtained from http://www.dmtf.org/standards/wbem/CIM-XML/

The paper entitled “New IBM xSeries firmware and device driver filename convention” describes the naming convention used for updates. It can be found on http://www.ibm.com/support by entering the title of the paper in the search box, and clicking “search”. The result “IBM Support - New firmware and driver file naming convention – Servers” will link to the download page.

IBM Corporation
Systems and Technology Group
Route 100
Somers, NY 10589
U.S.A.
Produced in the United States of America
01-07

.