# Batch Upload Screen 2

## Introduction

The batch upload screen provides functionality to allow users to upload, download, and delete **XML** files to be processed for batch processes. This guide will describe the process how to implement a screen for a specific job.

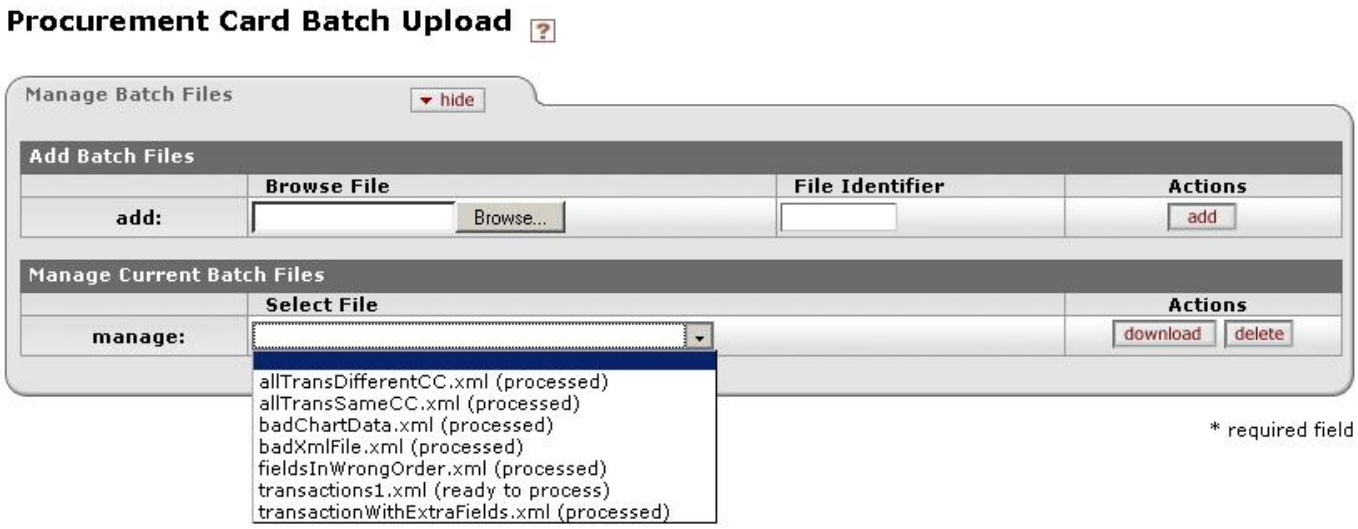
The XML files that may be easily incorporated into the batch upload screen having the following properties:

- Can be validated with an XSD file
- Can be converted into a java object using an Apache Digester script

## Batch upload screen screenshot

The following is a screenshot of the Procurement Card Batch Upload screen. Note that it allows the following actions:

- To upload (via the "add" button): The screen prompts for a file to upload and a file identifier. A file identifier is a string consisting of only letters and digits that serves as a note about file. It is used to generate the file name when the uploaded file is stored on the server, and is not likely to be used for any other purposes.
- To download and delete: There is a drop-down box indicating which files may be downloaded or deleted by the user. Note that next to each file name, "processed" or "ready to process" is displayed. Generally, the former indicates that the file has already been processed by the batch job. For exceptions to this rule, see the discussion of done files below. The latter indicates that the file is ready to be processed by the batch job.




## About KFS batch files

KFS batch data files may exist on the filesystem with a "done" file in the same directory. The done file has an extension of ".done", and its presence indicates that the data file has been fully written to disk and is ready to be processed. The "done" file is not to be interpreted as the system is "done" processing the file.

The done filename prefix must match the data file name prefix. For example, if the data file name is "pcdoTrans.xml", then the done file must be named "pcdoTrans.done".

When uploading a batch data file, the data file is usually created first, and then the done file is created. Batch jobs are required to only process a data file if the done file exists on disk. Upon completion of a batch job, it should delete the done file so that future executions of the job will not process the same data file.

The absence of the done file indicates one of two conditions, both of which will cause the batch upload screen to indicate "processed" in the drop down boxes of files:

- A vast majority of the time, it indicates that a batch job has completed processing a file
- For a very small period of time, it could indicate that a data file is being uploaded and a done file has not yet been created.

## Steps to implement the batch upload screen

This section will describe how the procurement card upload screen is implemented using the batch file upload framework.

### Create a new class that implements `BatchInputFileType`, or extends `BatchInputFileTypeBase`

Instances of `org.kuali.kfs.batch.BatchInputFileType` provide metadata about the data files appropriate for a batch process. Its base implementation `org.kuali.kfs.batch.BatchInputFileTypeBase` provides default implementations for the common operations defined by the interface.

The following is the implementation for the batch input file type for the procurement card. Further explanation is provided in the comments.

```
package org.kuali.module.financial.batch.pcard;

import java.io.File;
import java.sql.Timestamp;

import org.apache.commons.lang.StringUtils;
import org.kuali.core.bo.user.UniversalUser;
import org.kuali.core.service.DateTimeService;
import org.kuali.kfs.KFSConstants;
import org.kuali.kfs.KFSKeyConstants;
import org.kuali.kfs.batch.BatchInputFileTypeBase;

/**
 * Batch input type for the procurement card job.
 */
public class ProcurementCardInputFileType extends BatchInputFileTypeBase {
    private static org.apache.log4j.Logger LOG =
org.apache.log4j.Logger.getLogger(ProcurementCardInputFileType.class);

    private DateTimeService dateTimeService;

    public String getFileTypeIdentifer() {
        // The name of the Spring bean that is an instance of this class
        return KFSConstants.PCDO_FILE_TYPE_INDENTIFIER;
    }

    public Class getUploadWorkgroupParameterComponent() {
        // used to determine which parameter defines the workgroup that is allowed to
upload/delete/download files of this type.  see subsection titled "Create a system
parameter to store the workgroup that is allowed to upload/download/delete files"
below
        return ProcurementCardLoadStep.class;
    }

    /**
     * Generates the name of the file
     *
     * The parsedFileContents parameter refers to the java object that was generated
by invoking the digester on the given file.
     * The userIdentifier parameter is a user entered string that provides a human
readable annotation about the file (i.e. like a comment) with only characters and
digits
     */
    public String getFileName(UniversalUser user, Object parsedFileContents, String
```

```
userIdentifier) {
        Timestamp currentTimestamp = dateTimeService.getCurrentTimestamp();

        String fileName = "pcdo_" + user.getPersonUserIdentifier().toLowerCase();
        if (StringUtils.isNotBlank(userIdentifier)) {
            fileName += "_" + userIdentifier;
        }
        fileName += "_" + dateTimeService.toString(currentTimestamp,
"yyyyMMdd_HHmmss");

        // remove spaces in filename
        fileName = StringUtils.remove(fileName, " ");

        return fileName;
    }


    /**
     * Allows the implementation of additional logic to determine whether a user is
allowed to upload/download/delete a given file
     */
    public boolean checkAuthorization(UniversalUser user, File batchFile) {
        return true;
    }

    /**
     * Given the object that was constructed from the file using the digester, is this
object valid?
     */
    public boolean validate(Object parsedFileContents) {
        return true;
    }

    /**
     * The name of the property accessible via the KualiConfigurationService that
returns the title to be displayed on the batch input file screen
     */
    public String getTitleKey() {
        return KFSKeyConstants.MESSAGE_BATCH_UPLOAD_TITLE_PCDO;
    }

    public DateTimeService getDateTimeService() {
        return dateTimeService;
    }

    public void setDateTimeService(DateTimeService dateTimeService) {
        this.dateTimeService = dateTimeService;
    }
```

```
}
```

## Configure the bean from the previous step in Spring, and set any desired properties

The next step is to define a new Spring bean that is an instance of the class implemented in the previous step. Consult this page for instructions on how to incorporate a new Spring definition file into KFS.

The following is the bean for the Procurement Card Batch Upload screen. Note that the name of this bean is identical to the value returned by the `getFileTypeIdentifer()` method in the class above.

```
<bean id="procurementCardInputFileType"
class="org.kuali.module.financial.batch.pcard.ProcurementCardInputFileType">
    <property name="directoryPath">
       <!-- the full local path in which batch input files for this type should be
stored. This path should be independent of a particular file operating system. For
example, use '/' for the folder separator and '/' to refer to the root of the file
system, as opposed to 'c:\'. -->
       <value>${staging.directory}/PCDO</value>
    </property>
    <property name="fileExtension">
       <!-- the file extension for the files handled by this batch file type -->
       <value>xml</value>
    </property>
    <property name="digestorRulesFileName">
       <!-- the full path and name of the digester rules file (that contains the
parsing rules for this file type) -->
       <value>org/kuali/module/financial/batch/pcard/pcdoDigesterRules.xml</value>
    </property>
    <property name="schemaLocation">
       <!-- the classpath to the schema file -->
       <value>org/kuali/module/financial/batch/pcard/procurementCard.xsd</value>
    </property>
    <property name="dateTimeService">
       <!-- used for making date/time part of the file name to ensure unique names -->
       <ref bean="dateTimeService"/>
    </property>
</bean>
```

Note that property substitution was used (e.g. the `${staging.directory}`), and the value for properties can be overridden by following the instructions here.

## Create a system parameter to store the workgroup that is allowed to upload/download/delete files

First, determine the step class associated with using files of the batch upload type being implemented. For the procurement card example, the `org.kuali.module.financial.batch.pcard.ProcurementCardLoadStep` step will process the files.

A parameter consists of three primary keys: namespace, component, parameter name. For KFS, we normally map certain classes (including Step classes) to namespace and component codes, and the process for doing so is defined here. This allows us to specify parameters using a Class and a parameter name.

For a batch input file type, a parameter is needed that specifies which workgroup is allowed to operate on files of a given type. The parameter name must be "UPLOAD_GROUP". The step class determines the namespace and component of the parameter. Because `ProcurementCardLoadStep` is associated with the financial module (because it's in one of the packages managed by the financial module) and the financial module has a module code of "FP", the namespace code is "KFS-" appended with the moduleCode property of the financial module bean ("FP"), which forms KFS-FP. Because the class is a Step class, the simple class name (i.e. no package) is used for the "component" of the parameter name.

The following is the definition of the procurement card parameter.

| Field Name | Value | Explanation |
|---|---|---|
| Module: | KFS-FP | Part if parameter primary key; derived from the module managing the `ProcurementCardLoadStep` class |
| Component: | ProcurementCardLoadStep | Part if parameter primary key; the component is the simple class name because the class is a Step type |
| Parameter Name: | UPLOAD_GROUP | Part if parameter primary key; required parameter name for the batch upload type |
| Parameter Value: | KUALI_FMSOPS | The group allowed to upload/download/delete files |
| Parameter Constraint Code: | A | A for allow, just use A for this type of parameter |
| Parameter Description: | The workgroup to which a user must belong to have permission to do batch uploads for the Procurement Card Document. | A human-readable description |
| Parameter Type: | AUTH | Indicates that the parameter is used for authorization |
| Workgroup: | KUALI_ROLE_ADMINISTRATOR | The workgroup allowed to edit this parameter, is NOT related to batch upload functionality. |