TYBERA

eFlex Electronic Filing – Batch Filing

Kansas Office of Judicial Administration

April 2014 v8

Contents

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

Introduction

As the Kansas Office of Judicial Administration, 'OJA,' continues expanding efiling throughout the state, an additional feature for batch filing is being added. Shawnee County has been using a type of batch filing process for several years. The batch filing feature is beneficial for certain for high volume case processors. This feature reduces the time required to enter data in the web interface referred to as the Filers Interface.

With batch filing, the filer still uploads a document, but the format of the document is in XML format. Embedded within the XML document is all the case initiation information, documents, and additional document-related information needed to submit one or more filings.

Batch filing provides a faster method of uploading information. The XML document can contain information for multiple cases. Batch filing can generate several documents so that the user only enters data rather than creating PDF documents.

The reason most users don't use the batch filing feature is because the XML documents that are required for batch filing are generally programmatically prepared. In addition, the filers that use batch filing must first work with OJA to be approved to use the batch filing feature by verifying that the XML documents they programmatically prepare are valid.

To contact OJA: efilingadministrator@kscourts.org;

To contact a Shawnee County representative: efilehelp@shawneecourt.org

Following is a list of documents and locations to be used as part of the development of XML batch documents:

Shawnee County District Court Website:
http://www.shawneecourt.org/DocumentCenter/View/478 - Kansas Batch Filing Guide
http://www.shawneecourt.org/DocumentCenter/View/477 - Kansas Batch Schema
http://www.shawneecourt.org/DocumentCenter/View/480 - Kansas Batch Codes
http://www.shawneecourt.org/DocumentCenter/View/512 - Case Initiation Example
http://www.shawneecourt.org/DocumentCenter/View/511 - Follow up Filing Example
http://www.shawneecourt.org/DocumentCenter/View/479 - Kansas Matter Examples*

*Matter Codes are included in this document

Judicial Branch Website:
filer.kscourts.org/KansasBatchFilingGuide_v8.docx
filer.kscourts.org/KansasBatchSchema_v8.xsd
filer.kscourts.org/KansasBatchCodes_v8.xlsx
filer.kscourts.org/Kansas/BatchMatterCodes_v8.xml
filer.kscourts.org/KansasBatchExampleCaseInit_v8.xml

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

filer.kscourts.org/KansasBatchExampleFollowup_v8.xml
filer.kscourts.org/KansasMatterExamples.xml

Alternate site

www.tybera.com/KansasBatchFilingGuide_v8.docx
www.tybera.com/KansasBatchSchema_ v8.xsd
www.tybera.com/KansasBatchCodes_ v8.xlsx
www.tybera.com/KansasBatchExampleCaseInit_v8.xml
www.tybera.com/KansasBatchExampleFollowup_v8.xml
www.tybera.com/KansasMatterExamples.xml

Batch Filing Considerations

Batch filing is a process that by-passes many of the web pages that ask for information. Most filers log into the Filers Interface with a web browser and identify whether they are creating a new case or filing to an existing case. The filer is then guided through a series of web pages that allow them to add parties, add documents, and submit the information to the courts.

In the batch filing process, the filer still logs into the Filer Interface using a web browser, but rather than preparing a submission for a single case, the filer can upload an XML document that represents submissions on multiple cases.

Batch filing requires that the law firm be able to programmatically create the XML files used for batch filing. It is anticipated that the law firm will have a case management system or an internal process they use to track their cases, and an internal process that will generate the XML document. It is not anticipated that anyone would create an XML document for batch filing by hand, even though it is possible. The challenge with creating an XML document by hand is that there is a high possibility of errors.

Many of the documents that are filed in the batch filing process will be automatically generated after the XML batch document is uploaded in the efiling system. This guide lists what documents can be automatically generated. Within the XML document used in the batch filing process, some of data required to generate the documents will be supplied by the filer, some of the data will be supplied by the court location, and some of the data will be supplied by the FullCourt Case Management System used by the county selected by the filer in initiating the batch process.

When creating an XML document for batch filing, all the submissions within that batch must be associated to a single county. Multiple counties cannot be included in a single batch XML document.

Because of the complexity of payments, the batch filing process will not collect court fees for the first release of the batch filing feature. Later, an updated version of the batch filing functionality will include payment methods. For the first release, the county will be required to bill the filers for court fees on all batch filed cases after the information is recorded in the

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

FullCourt Case Management System. For this reason, many of the counties will not participate in batch filing until after the payment methods are included in the batch filing process.

The system has been designed to the court's specifications to validate certain fields by either the efiling system or by FullCourt case management system. Filing information through the batch filing process will cause the case information and documents to bypass the clerk review process and automatically record to FullCourt. It is possible that data, such as codes, passes the validation process but some information was incorrect. Perhaps the wrong document code was included or the wrong party type was selected or the names were wrong. It is the responsibility of the filer to notify the court of the mistake. A judge will make a determination about what to do in under these circumstances.

How to Qualify to Batch File

The first step to secure permission to use the batch filing feature is to contact OJA to determine which counties accept batch filings. OJA must manually setup each account with batch filing rights before the account can use the feature. The second step is to understand the data required and how the data is entered in the XML documents. Instructions for preparing the XML documents are in this guide, including information on which codes in the KansasBatchCode.xsl spreadsheet to use. The third step is to generate your XML documents. You can download the KansasBatchExample.... documents to help guide you as well as the KansasBatchSchema.xsd.

Once a filer can generate XML documents, they can begin testing their XML by uploading the generated XML to a test server. When the filer reaches the state where they are generating all their submissions in XML format, and the test environment is accepting their XML documents without errors, the filer will contact OJA and ask OJA to verify with the filer that the filer is ready to file live filings using the batch filing feature. When OJA agrees that the filer is ready, OJA will set the filer account with batch filing rights. Once approved, the filer can use the batch filing feature to file to any county accepting batch filing.

Filers with batch filing rights will have an additional button on the Home page called 'Batch Filing'. Clicking on this button will allow the filer to upload their XML document.

Batch File Steps

A filer will log into the Filers Interface. Once OJA has set their permission to batch file, the filer will see a new button on the Home page.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

Clicking on the Batch Filings option will take the filer to the Batch Upload page as shown.

Clicking on the Browse button will allow the filer to browse and locate the XML document they have prepared for filing. Selecting the XML document will upload the XML document to the server and show the name of the document above the submit button as shown.

In the test environment, a 'Show Submissions in Draft Filing' checkbox will display. Checking this box means that, after clicking the submit button, rather than sending submission for

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

recording, eFlex will post the generated individual submissions into the draft submission state. This option will not be available in the production system at this time.

Clicking on the submit button will cause eFlex to first parse the XML document against the KansasBatchSchema.xsd file. If any errors occur in the structure of the XML, the filer will first see an error page such as the screen shot below. If an error occurs, the filer needs to fix the XML document and start over.

When the XML structure parses without error, the submission will display the status page.

Parsing the XML structure without error does not mean there are not errors in the data included. eFlex still needs to verify certain data fields. Any data field that is enumerated still needs to pass validation. This type of validation refers to the KansasBatchCode.xsl spreadsheets.

After this validation there are still format validations such as the format of a Social Security Number and validation that the Bar Number is correct. This layer of validation is not 100% tested at the eFlex level and may pass through to the FullCourt validation where it can error out.

Assuming that the XML and data is accurate, eFlex will generate individual submissions for each entry within the batch XML document. Once eFlex has generated each submission, eFlex will display a status result of what was generated and what failed. For example, if the XML contained a code to a county that did not accept batch submissions, this status result page would show that the batch submission failed.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

After the batch file has been submitted, the user will view a page that displays the results of the submission on the Batch Filing Status page. There are several identification numbers associated with Batch Filing. A Filer's Reference There is a Batch Id generated by eFlex that identifies the entire batch package. The Tracking Ids are generated by eFlex and are assigned to each submission that is created. The red boxes on the image have reference letters which are described below.

A. The Filer's Reference Number is the referenceNumber that is an attribute on the submission element supplied by the filer in the XML document. It is the number that the filer can use to check whether the individual submission information passed or failed. Submissions that pass will continue to be recorded in FullCourt. If a submission fails, the Processing Status column will indicate the failure. No information will be recorded if there is an error.

B. The Batch Id is generated by eFlex and is used by FullCourt to identify to the court which submissions did not actually collect money. Eventually this Batch Id will be used to expand the payment collections as well.

C. Tracking Ids generated and assigned to each submission. eFlex assigns two tracking ids to every submission whether they were created using batch filing or using the Filer Interface. Tracking Ids allow court employees and the filer to easily located a submission and use this reference number if something goes wrong during the recording to FullCourt.

D. Filing Status/Draft Filing Button. The first button on the batch filing result page is a link to the Filing Status page where the filer can view the updated status of each submission. If the filer selected the 'Show Submissions in Draft Filing' checkbox on the page where the XML was uploaded, the 'Filing Status' button at the bottom of the batch filing result page would change to be "Draft Submissions." Putting the submissions in draft state allows the person developing the XML documents verify, using the Filer Interface, that the data included in the XML document was translated properly into the submission. It also

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

allows the developer to see if data was left out of the XML document that should be included.

E. Download Status XML Button. The second button on the Batch Filing Status page allows the filer to download the status results of the batch upload. This status will tell the filer the status of each submission and whether the referenceNumber succeeded or failed.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

Building a Batch XML document

Structure Overview

A batch submission contains multiple submissions. All submissions within a batch must go to the same county. The submissions can be on different cases. The batch submission can contain information that creates a new case and information that files to an existing case.

The batch submission requires a XML batch document which uses the Extensible Markup Language (XML). XML is a language designed to structure, transport, and store data rather than to display it. The data being transported is wrapped in "tags." These tags are elements defined in a schema. Proper XML language refers to tag names as elements. In this document elements and tags may be used interchangeably.

This section of instructions assumes the reader has an understanding of XML. For more information or tutorials on XML, please refer to the World Wide Web standards on the web at www.w3c.org/standards/xml/. The schema can be downloaded as identified in the introduction.

Although the following instructions help guide the reader through creating an XML Batch document it is recommended that the reader download the KansasBatchSchema.xds file. Once a schema is understood the following information is not as important. The schema provides the structure, order, and restrictions that a developer should understand in developing the process that generates the XML Batch document.

Following is a high level view of how a batch XML document is structured.

```
<?xml version="1.0" encoding="utf-8" standalone="true"?>
<batchSubmission>
  <submissions>
    <submission referenceNumber="3000094"><!--submission 1-->
      <actors>...</actors>
      <filingInformation>...</filingInformation >
      <document>...</document>
    </submission>
    <submission referenceNumber="3000095"><!--submission 2-->
      <actors>...</actors>
      <filingInformation>...</filingInformation >
      <document>...</document>
    </submission>
    <submission referenceNumber="3000096"><!--submission 3-->
      <actors>...</actors>
      <filingInformation>...</filingInformation >
      <document>...</document>
    </submission>
  </submissions>
</batchSubmission>
```

This simple example shows that there are three individual submissions included in this batch submission. The comment tag that looks like: `<!--submission 1-->` shows where the individual submissions are.

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

The first line of the XML document, the declaration, should always be the same and should contain the following information.

```
<?xml version="1.0" encoding="utf-8" standalone="true"?>
```

Likewise, the root tag or parent element (second line of XML) will always be <batchSubmission>, with the first word in lower case, the second word capitalized, and no space between the two words.

```
<?xml version="1.0" encoding="utf-8" standalone="true"?>
<batchSubmission>
```

The third line of the XML tree, the child element of batchSubmission, is submissions. Be sure that this element is plural as the example shows. Submissions plural means that multiple submission elements can exist, or multiple filings can exist in this XML Batch document.

```
<?xml version="1.0" encoding="utf-8" standalone="true"?>
<batchSubmission>
  <submissions>
```

At this point, the user is ready to begin entering information for each individual submission. Each submission has a unique identifier. The referenceNumber shown in the example is an attribute on the submission element. This reference number is required and is assigned by the filer. In a structure example above there are three referenceNumbers, `3000094, 3000095, 3000096,` which are supplied by the filer. Each referenceNumber is for a unique filing. The referenceNumber values supplied by the filer must be unique in a submission.

Whether the referenceNumber is used in another batch is up to the filer's internal process generating the XML batch document. Tybera does not recommend reusing these reference numbers unless that submission had an error and the filer regenerates the submission for the same purpose and corrects the errors.

Adding Actors

With an understanding of the overall structure of a batch XML document we can begin building the structure for each submission. This section deals with adding parties on a new case. Party information differs between case initiation and filing to an existing case. Not all the party information is required on filing to an existing case. First, we discuss how to enter actors for new cases. Recall that for each submission you have:

```
<submission referenceNumber="3000094">
  <actors>...</actors>
  <filingInformation>...</filingInformation >
  <document>...</document>
</submission>
```

The first child element in a submission is the actors section. The actors element is plural and means it has multiple actor elements.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

All submissions in the batch will include information on persons, or actors, associated with the case. From a structural level, the submission actors follow a pattern on new cases.

```
<actors>
      <actor newActor="true" id="B_2222">...</actors>
      <actor newActor="true" id="P1">...</actors>
      <actor newActor="true" id="D3">...</actors>
      <actor newActor="true" id="D2">...</actors>
      <actor newActor="true" id="D1">...</actors>
</actors>
```

For each actor, the structure follows a pattern. Not all fields are required, but if the filer has accurate information, it is best to provide it to the court.

```
<actor newActor="true" id="B2222">
      <name type="person"> ... </name>
      <personDescription> ... </personDescription>
      <roles> ... </roles>
      <aliases> ... </aliases>
</actor>
```

Adding Attorneys

The actors section begins with the <actor> element, and the first actor Tybera recommends to enter is the attorney(s) who will be representing the plaintiff(s). You should always add the filer as the first attorney. In the following example, there are several critical sections of information.

```
<actor newActor="true" id="B2222">
      <name type="person">
            <firstName/>
            <lastName>Cox</lastName>
      </name>
      <personDescription>
            <personalIDNumber type="Bar">
                  <idNumber>2222</idNumber>
                  <state>KS</state>
            </personalIDNumber>
      </personDescription>
      <roles>
           <role>
              <roleName>A</roleName>
           </role>
      </roles>
</actor>
```

First, the id attribute on the actor is required. This id gives a reference number that is used later when associating this attorney to the plaintiff or to other parties on the case. In the following example, there is a person to whom we have given the id="B2222". In XML you should always start an attribute value with a letter and not a number. You can give an actor an id value that you generate. In this example, we simply used the attorney's (fake) Kansas Bar with a leading

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

B. Since the Kansas Bar number does not start with a letter, we added a B to the front of it. This reference number as an easy reference id to remember.

The second attribute on an actor is whether the actor is being added to the case or not. newActor="true" means this attorney is being added to the case. If the submisison is being filed on an existing case, and there are no new attorneys being added, newActor="false" would be entered.

The first child element in actor is the name element. The name element has an attribute called 'type.' The value for type is always 'person' for an attorney. Other actors, such as plaintiffs or defendants, possibly could be a type='organization'. In the example above, the attorney's last name is Cox. . For attorneys the first name is not required. Later if you are adding an organization or business then the first name is not required. The last name is use to identify the business or organization or an attorney if applicable.

Following the name element is a sibling element called personDescription. This section is like an id issued by a government agency. This information is not included in an attribute field because government id numbers do not start with a letter and attributes should. The first child element called personalIDNumber. The personalIDNumber has an attribute called type. For an attorney, the type will always be type="BAR". Notice the capitalization.

The personalIDNumber has two child elements called idNumber and state. The element idNumber does not have attribute. The value of idNumber for an attorney is always the Kansas Bar number. The attorney bar number must be a valid bar number previously recorded in the FullCourt CMS at the county the submission is going to. FullCourt will return an error if the attorney bar number is not in the FullCourt installation in the county to which the filer is sending the submissions. Kansas only allows attorneys licensed in Kansas to efile, so the state element value for an attorney is always KS. For an attorney, the personal description, which includes the Kansas Bar number and the bar state, is required. The KansasBatchCodes.xls has a list of the state codes that you can use however it is anticipated that the majority of Ids that you enter will be from KS.

Unlike other participants on the case, the actor information for the attorney does not need to include the address and phone number. Other actors may not have personDescription element at all or may have a more complex personDescription with address information.

The next element, roles, is a sibling to name and personalDescription. The roles element is plural because some courts allow an actor to have more than one role, such as when a defendant counter sues the initial plaintiff, and the defendant then becomes a plaintiff.

The first child of roles is the role element. The first child of role is the roleName element. The roleName element for an attorney is "A". The filer of the submission is automatically entered by eFlex as the user that logs in to upload the XML Batch document. Tybera recommends that the attorney and the roleName of filer always be included for each submission.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

Plaintiff, Defendant, and other Actors

Regardless of the case type, the submission structure will follow the same pattern in terms of adding the parties on the case.

```
<actor newActor="true" id="P1">
      <name type="person"> ... </name>
      <personDescription> ... </personDescription>
      <roles> ... </roles>
</actor>
```

The Plaintiff is an actor in this submission information. Whether the actor is a plaintiff, defendant, or some other role, the pattern is the same.

The id="P1" is a reference to the actor that is required. Each actor needs an id and the ids must be unique in the submission. The id reference is used to reference this party in the matter items. The matter item section is a special section that is used for document generation and will be discussed later.

The newActor="true" means that eFlex needs to tell FullCourt to add this party ot the case.

The name element has an attribute like the attorney to identify it as either a person or an organization. If the actor is a person, the first name is required, but if the actor is an organization, then only the last name is required.

Then for each actor element, the pattern may include name, postalAddress, phoneNumber, faxNumber, email, personalDescription, and role.

```
<actor newActor="true" id="xxx">
      <name type="person">...</name>
      <postalAddress>...</postalAddress>
      <telephoneNumber>...</phoneNumber>
      <faxNumber>...</faxNumber>
      <email>...</email>
      <personDescription>...</personDescription >
      <roles>...</roles>
</actors>
```

The name has the following structure:

```
<name type="organization">
      <firstName> ... </firstName>
      <middleName> ... </middleName>
      <lastName> ... </lastName>
      <nameSuffix> ... </nameSuffix>
      <aliases> ... </aliases>
</name>
```

The firstName, middleName, and nameSuffix elements are only used for a actor of type="person." For a business the name must go in the lastName element. The aliases element is plural which means that there can be multiple alias elements.

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

```
<name type="person">
      <firstName>George</firstName>
      <middleName>W</middleName>
      <lastName>Bush</lastName>
      <nameSuffix>JR</nameSuffix>
      <aliases type="DBA">
            <name type="person">
                  <lastName>President of United States</lastName>
            </name>
      </aliases>
</name>
```

The alias element has an attribute of type. Following are a listing of the types that are available.

- AKA <!-- means Also Known As -->
- DBA <!-- means Doing Business As -->
- FDBA <!-- means Frequently Doing Business As -->
- FKA <!-- means Frequently Known As -->
- NKA <!-- means Now Known As -->
- OBO <!-- means On Behalf Of -->

Following is an example of a business that is being added to the case. In this case, the firstName element is not used.

```
<actor newActor="true" id="PF_1">
      <name type="organization">
            <lastName>THIRD FIFTH BANK</lastName>
      </name>
      <personDescription>
            <personalIDNumber type="EIN">
                  <idNumber>123412343</idNumber>
            </personalIDNumber>
      </personDescription>
      <roles>
           <role>
              <roleName>P</roleName>
              <roleWiths>
                  <roleWith>B2222</roleWith>
              </roleWiths>
           </role>
      </roles>
</actor>
```

For each party other than an attorney added to a submission, a street address, city, state, and postal (zip) code must be included. Additional information such as telephone number, fax number, or email address may also be included but are optional. If the filer does not have that information, use the empty begin tag structure to indicate no information is available such as <email />. The slash at the end of an element represents an empty begin tag and no data or end tag exists.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

Notice how the structure is the same as for an attorney actor, but the plaintiff actor has additional information. The most critical item that is required is the address information for the Plaintiff role. The address information was not added for the attorney.

The personDescription for a Plaintiff may include either the Social Security Number (SSN), the Driver's License Number (DL), or, for an organization, the Federal Employee Identification Number (EIN) as the idNumber attribute value. In addition, if an actor is a frequently referenced actor such as the Director of Taxation Department of Revenue the CMS identification (CMSID) can be used in the type field and the FullCourt CMS ID published by the county can be inserted.

If an individual does not have a Driver's License but does have a State Identification Number, the State Identification Number should be entered where a Driver's License number would appear.

Refer to the filer.kscourts.org/KansasBatchCodes_2014_0426.xsl Excel spreadsheet for the values that can be used.

The role information section is where we identify if this actor is a Plaintiff, Defendant, Witness, or other role. In this example, the roleName 'P' means FullCourt will enter this party as a Plaintiff. Reference the KansasBatchCodes.xls sheet for the codes of all roles that can be used.

The rolesWiths is plural and may have multiple references. If a plaintiff has multiple attorneys, then each attorney would be referenced with a roleWith individually. In this case the business is referencing the attorney in the example above by including the id of the attorney actor as a reference. For this reason, the B2222 is included here. If there were two attorneys representing this actor, first this submission would require that two attorneys with different id values be added. The actor could then have two separate roleWith lines to reference each attorney.

Additional plaintiffs can be added as a separate actor entries. The number of parties the filer adds is not limited.

When adding a defendant, most often it is not known who will represent that party. Because of that, the role section will be simplified and only contain the role as shown in this example.

```
<roles>
   <role>
      <roleName>D</roleName>
   </role>
</roles>
```


Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

Adding Filing Information

Each individual submission will include filing information. The data provided for filing information varies between case initiation and follow-up filings. In both conditions the overall structure is as follows:

```
    <submission referenceNumber="3000094">
      <actors>...</actors>
      <filingInformation>
            <courtInformation>...</courtInformation>
            <caseInformation newCase="" >...</caseInformation>
      </filingInformation >
      <document>...</document>
    </submission>
</filingInformation>
```

Court Information

The filing information always begins with the court information element. Each submission must clearly identify the court to which the submission is routed by including the court location code as well as the court code that identifies the submission. This is required for new and existing cases. The court location code will be a two or three digit number unique to each district court. For batch filing, the courtCode is always DISTRICT because the appellate court does not accept batch filings.

Currently, the only court to which batch filings can be submitted is the Shawnee District Court. The Shawnee District Court location code is “89.” For other court locations, refer to the Kansas Batch Codes Excel spreadsheet.

The Court Information is the same for case initiation as it is for follow-up filings.

```
<filingInformation>
      <courtInformation>
            <courtLocationCode>89</courtLocationCode>
            <courtCode>DISTRICT</courtCode>
      </courtInformation>
```

Case Information

The Case Information is different between case initiation and follow-up filings. The case information must indicate whether the submission is to a new case or to an existing case.

The case category is identified by a two digit code and is consistent for both case initiation and follow-up filings. Both case initiation and follow-up filings include the filersCaseNumber. This can be the same as the submission referenceNumber, which is replicated here for internal use. The filersCaseNumber could also be another number entered by the filer. This number is used in the receipt that is returned in the status updates on the Filing Status web page.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

Notice on the caseInformation tag the attribute newCase="true" for new cases or newCase="false" for existing cases.

For New Cases:

```
<filingInformation>
      <courtInformation>...</courtInformation>
      <caseInformation newCase="true">
            <fullCaseNumber/>
            <caseTypeCode>CV_CF</caseTypeCode>
            <caseCategoryCode>CV</caseCategoryCode>
            <totalAmount>2344</totalAmount><!-- not required on all -->
            <filersCaseNumber>300095</filersCaseNumber>
     </caseInformation>
</filingInformation>
```

The FullCourt CMS will actually generate the case title. Notice how the fullCaseNumber tag is an empty begin tag. The empty begin tag means this is a new case and the case number is not yet assigned.

The totalAmount is only required on case initiation and only for certain types of cases. This is the field that identifies the Prayer Amount being soughtin the complaint.

For follow-up filings, the fullCaseNumber is filled in, and the totalAmount, caseCategoryCode, and caseTypeCode are empty begin tags.

```
<filingInformation>
      <courtInformation>...</courtInformation>
      <caseInformation newCase="false">
            <fullCaseNumber>2014-CV-2345</fullCaseNumber>
            <caseTypeCode/>
            <caseTitle>JONES VS JOHNSON</caseTitle>
            <caseCategoryCode/>
            <totalAmount/>
            <filersCaseNumber>300095</filersCaseNumber>
      </caseInformation>
```

The case title will be generated by FullCourt for case initiation submissions so the caseTitle could be an empty begin tag or the filer could fill in the caseTitle. Filling it in does not hurt anything, and later it may be used for document generation. However, FullCourt will ignore a populated caseTitle at this time.

Refer to the Kansas Batch Codes Excel spreadsheet for the codes that are available for caseTypeCode and caseCategoryCode.

Adding Document Information

Once the court and case information is complete, the filer is ready to work with the document associated with the submission. A submission can have multiple documents. Because of this the schema has the documents element which is plural meaning there can be more than one document in each submission. The documents element is a sibling to caseInformation.

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

The documents element has document, singular, that are children of documents element. The structure is as follows:

```
<submission referenceNumber="3000094">
      <actors>...</actors>
      <filingInformation>...</filingInformation >
      <documents>
            <document id="D1">...</document>
            <document id="D2">...</document>
      </documents>
</submission>
```

The structure for each document element will follow this pattern:

```
<document id="D1">
      <documentInformation>
            <documentTitleSuffix>...</documentTitleSuffix>
            </documentTypeCode>...<documentTypeCode>
      </documentInformation>
      <documentContent>
            <content contentEncoding="..." mimeType="....">
                  ...
            </content>
      </documentContent>
      <mattersDocument>
            <documentContent>
                  <content contentEncoding="..." mimeType="....">
                        ...
                  </content>
            </documentContent>
      </mattersDocument>
</document>
```

Every document element will have an id attribute. The id attribute must be unique in the XML Batch document. The id must start with a letter.

For every document element the first child is the documentInformation element. This section is required for every document included in the submission. To understand the children of the documentInformation, the documentTitleSuffix and the documentTypeCode a screen shot of the web Filer Interface is presented.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

From this screen shot, the first field is the Action Type field. This field is a filter for the next field which is Action. The Action Type field is not included in the documentInformation structure becaue it is just a filter and does not provide any necessary data to FullCourt. When a court has hundreds of Actions to choose from, the Action Type field filters the number of documents displayed for the filer to choose from. If the Action Type is left blank in the Filer Interface, then the Action field will display all possible documents that could be filed.

The second field in the screenshot is the Action field. This field represents the type of document being included. The documentTypeCode in the document structure is what the Action field refers to. The KansasBatchCodes_v8.xlsx contains a worksheet that lists the documents and their codes that will be used to populate this tag's value.

```
<documentInformation>
      <documentTitleSuffix>Additional Text field</documentTitleSuffix>
      </documentTypeCode>EBPET<documentTypeCode> <!-- Action -->
</documentInformation>
```

The third field in the screenshot above is the Additional Text field. Notice that this is a required field in the Filer Interface. The documentTitleSuffix element is populated by the additional text that may be needed to further describe the document. The additional text information is displayed in the FullCourt case history with the associated document. If you look at a case history in eFlex, for each document recorded to the case, you will see the additional text which in the structure is the documentTitleSuffix.

Document Content

There are two methods for including a document in a submission. The first method is to embed the actual document and the second is to include just the data and let eFlex generate the document. The generated documents will use a standard template defined by Kansas OJA. There may be times when the standard templates do not satisfy the need of the filer. In this

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

case the filer will use the first method by creating a pdf document, base64 encode the pdf document, and wrtting the base64 character stream into the XML Batch document.

There are two children to the document element that are used for these methods. The documentContent element is always used for method one and the mattersDocument is used for the second method.

The content element is the only child of documentContent. The content element values are always base64 encoded. For method one of embedding the document character stream the base64 encoded stream is stored in the content element.

The mattersDocument always exists (as of this writting) whether you use method one or not. If the documentConent child of document is not used then the mattersDocument contains the data required to generate the document. The mattersDocument element has as a child the same element used in method one but this time rather than embedding a pdf document that base64 encoded an XML document that contains data specific to the document is encoded and the value is stored in the content element. Following are two examples. The first is what the structure will look like for method one if you create a pdf document and embed it.

```
<document id="D1">
    <documentInformation>...</documentInformation>
    <documentContent> <!-- used only when embedding non-generated doc -->
          <content contentEncoding="" id="" mimeType="">
          ...base64 encoded pdf document...
          </content>
    </documentContent>
    <mattersDocument> <!-- used for data to generate document -->
          <documentContent> <!-- same as above -->
                <content contentEncoding="" id="" mimeType="">
                ...base64 encoded xml data used to generate documents -->
                </content>
          </documentContent>
    </mattersDocument>
</document>
```

This next example is what the structure will look like if you are generating a document.

```
<document id="D1">
    <documentInformation>...</documentInformation>
    <mattersDocument> <!-- used for data to generate document -->
          <documentContent> <!-- same as above -->
                <content contentEncoding="" id="" mimeType="">
                ...base64 encoded xml data used to generate documents -->
                </content>
          </documentContent>
    </mattersDocument>
</document>
```

The process of base64 encoding is a standard method available in most programming languages and should not be difficult for those preparing the batch XML documents to generate.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

The content element has two attributes which are `contentEncoding` and `mimeType.` The attribute value for `contentEncoding="base64"` will always be base64. The value for mimeType can have multiple values and is based on the type of document that is base64 encoded. These values are:

1. mimeType="application/pdf" (embedding a PDF document)
2. mimeType="application/xml" (embedding a XML document)
3. mimeType="application/rtf" (embedding a RTF document)
4. mimeType="application/msword" (embedding a Microsoft Word document)
5. mimeType="application/wordperfect" (embedding a Corel WordPerfect document)
6. mimeType="text/plain " (embedding a text document)

The first mimeType="application/pdf" is used with method one when embedding a PDF document.

The second mimeType="application/xml" is always used for the embedded document stream that is stored in the mattersDocument.documentConent.content element.

At the time of writting no documents can be submitted in the other formats. These formats are used for filing proposed orders which are not included in the documentTypeCodes supported by batch filing. The mimeType="plain/txt" can be useful during the development and testing of method one so that it is easy to see if the encoding process is working and you are manually creating the XML Batch document.

One of the problems with generating documents is that the documents change from time to time and the data required changes. By base64 encoding the document data as well as the documents, the eFlex system can minimize changing the schema and major portions of the program each time the documents and the data need to change.

Following is an example of the content tag and a base64 encoded text stream. Base64 encoding takes a stream of information from characters represented as eight bits and converts them to six bit characters. This increases the size of the original stream. The purpose of this is to shift the characters embedded in the XML stream to the lower ascii characters which do not include charactrs that can be interpreted as XML markup. The following example is the base64 encoding of the following text stream:

This is a Kansas Batch filing test to show how the </content> end tag is not interpreted in the stream of data when base64 encoded.

```
<content contentEncoding="base64" mimeType="plain/txt">
VGhpcyBpcyBhIEthbnNhcyBCYXRjaCBmaWxpbmcgdGVzdCB0byBzaG93IGhvdyB0aGUgPC9jb250ZW50PiBlbmQgdGFnIGlzI
G5vdCBpbnRlcnByZXRlZCBpbiB0aGUgc3RyZWFtIG9mIGRhdGEgd2hlbiBiYXNlNjQgZW5jb2RlZC4=
</content>
```


Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

MattersDocument

For each document whether it is embedded or generated there must be a mattersDocument element included and the appropriate XML data base64 encoded as the content of the tag. The structure for the XML data that is follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<Form>
        <Matter_X_Matter_Name>KEYWORD</Matter_X_Matter_Name>
        <Matter_X_keySubWord>value</ Matter_X_keySubWord >
</Form>
```

To embed this data in the document first create the XML form, then base64 encode the form, then write the base64 stream as the value of the tag.

This XML data is not included in the schema because the data required to generate documents varies greatly, is not constant, and changes as the document templates change. By embedding these XML documents as a base64 document the schema that controls the overall structure is less complicated and does not have to be updated each time the templates to generate documents change.

To understand the form and how it is constructed it is first important to understand that these forms function as a name value pair and is not controlled with structure but heavily depends on parsing the actual element or tag name. The tag name is read, interpreted, and then a value associated to the tag is extracted for template generation.

Looking at the example above all form data will begin with the XML declaration

<?xml version="1.0" encoding="UTF-8"?> followed by the root element of the form which is <Form> </Form>.

Within the form there are what are referred to as matter items. As mentioned earlier this XML data is not controlled by a structure but is controlled by the names and key words.

The naming convension for matter items is as follows:

```
        <Matter_1_Matter_Name>Filing</Matter_1_Matter_Name>.
```

It is a declaration that anything starting with Matter_1_??? within the <Form>... </Form> is associated to "Filing."

In the following example the matter item that is being declared is word DocumentService and the Matter_1_FullNm_DF_1 is interpreted as the Full Name of the party being served by this Summon that is being generated.

```
<form>
        <Matter_1_Matter_Name>DocumentService</Matter_1_Matter_Name>
```


NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.

<Matter_1_FullNm_DF_1>Janet A Wilson</Matter_1_FullNm_DF_1>
<Matter_1_IssueDate>2014-04-12</Matter_1_IssueDate>
.....

</Form>

In this example Matter_1_IssueDate is interpreted as the issue date of the document being served is April 12, 2014.

Some documents that are generated refer to parties. When a summons is generated FullCourt needs to know who is being served. The reference to a party can be a new party being added as part of a submission for a new case or it could be a summon on a party that was previously recorded on an existing case.

Where there are two types of references, one for a new party that is being added as part of the submission and one for a party that already exists on the case there must be two different methods to reference these parties.

Referencing a new party included in the submission means that there is a reference in the matter items to an actor that is in the main body of the XML. Following is an example of what this will look like: <Matter_1_IdRef_DF_1>D1</Matter_1_IdRef_DF_1>. This example can be interpreted as the document being served on the first defendant that will be recorded in FullCourt is a new actor to the case that was defined in the XML and given an id="D1".

Referencing a party that was previously recorded to a case means that there is a reference in the matter items to a CMS Id. Following is an example of what this will look like: <Matter_1_CMSId_DF_1>24134</Matter_1_IdRef_DF_1>. This example can be interpreted as the document being served on the first defendant that already exists in FullCourt and FullCourt CMS Id is 24134.

The challenge with referencing CMS Ids is getting access to and storing those Ids. The Kansas OJA decided that summons on existing parties needs to be supported in batch filing. In order to achieve this requirement the FullCourt CMS Ids must be included in the submission.

The CMS Ids are critical to communicate back to FullCourt so that FullCourt can track who the summon was issued for. In addition, the summon itself will have a FullCourt Id associated to it. When a return of service is efiled this CMS ID for the summon must be included in the data and eFlex will pass this to FullCourt. With the summon Id FullCourt will be able to identify and record which summon was perfected or failed.

If for some reason a filer does not want to retrieve and store the CMS Ids for use in the batch filing feature the alternative is to use the Filer Interface and manually submit the request. If this type of summon is not used that often then using the Filer Interface is not an unreasonable option.

Copyright, December 2013 – Tybera Development Group, Inc. NOT FOR DISTRIBUTION

For definitions of all data fields required in the mattersDocument section refer to the KansasMattersExample_v8.xml.

NOT FOR DISTRIBUTION Copyright, March 2014 – Tybera Development Group, Inc.