

State of Kansas

Kansas Bureau of Investigation

Kansas Department of Transportation

**Integration of Kansas DUI Tracking System Report and Police Impaired Drivers (RAPID)**

<Contract Number 10713, Group 8>

Electronic Court Disposition/Filing Interface Implementation Description Document

Analysts International Corporation (AIC)

12/02/2013

Version 1.2.3

Analysts International Corporation
7700 France Avenue, Suite 200 ∙ Minneapolis, MN 55435
Phone: (952) 835-5900 ∙ Fax: (952) 897-4555 ∙ [www.analysts.com](http://www.analysts.com)

**Contact(s):**
Joe Mandala∙ Project Manager ∙ 785.296.7724 ∙ Joe.Mandala@KBI.STATE.KS.US
Trevor Scoggins ∙ Project Manager ∙ 859 219 4218∙TScoggins@analysts.com Steve Sutton ∙ Business Analyst ∙ 248.345.2679 ∙ SSutton@analysts.com

# Table of Contents

[Table of Contents 2](#_Toc386121287)

[Scope 1](#_Toc386121288)

[Document Overview 1](#_Toc386121289)

[Interface Overview 1](#_Toc386121290)

[Electronic Submission Overview 1](#_Toc386121291)

[Detailed Interface Requirements 6](#_Toc386121292)

[Interface Processing Time Requirements 6](#_Toc386121293)

[Interface Initiation 6](#_Toc386121294)

[Message Requirements 6](#_Toc386121295)

[Flow Control 6](#_Toc386121296)

[Appendices 8](#_Toc386121297)

[Kansas Disposition Report IEPD 8](#_Toc386121298)

[KADR-KJDR Business Rules 8](#_Toc386121299)

[Filings and Disposition Comprehensive Field Mapping Document 8](#_Toc386121300)

[Document Management 8](#_Toc386121301)

[History 8](#_Toc386121302)

# Scope

This Interface Implementation Description Document describes the interface between individual court systems and the Filings and Dispositions Web Service.

# Document Overview

The purpose of this document is to specify interface requirements to be met by the participating systems. It defines the message structure and protocols which govern the interchange of data, and identifies the communication paths along which the data are expected to flow.

# Interface Overview

The following diagram describes the flow for SharePoint-based submission & reporting and electronic-based submission of filings and dispositions. The electronic submission, coming from the district and municipal courts, is the focus of this document.

This diagram illustrates that the same Filings & Disposition web service is utilized by electronic submission, form-based submission and reporting but different methods are invoked for different functions. The detailed flows for the electronic submission will be illustrated in the following sub-sections.



Figure 1: Filings and Dispositions Web Service Process Diagram

## Electronic Submission Overview

The following diagram describes the electronic submission process whereby Filing & Disposition data are electronically submitted through the Filings & Disposition Web Service from the Courts using a standard data interface.

 

Figure 2: Filings and Dispositions Electronic Transmission Process Diagram

### Filings and Dispositions Web Service Description

The Filings & Disposition Web Service allows courts to submit filings and disposition data in XML format and get a list of submissions that failed.

#### Methods

CreateDispositionAuto()

Description

This method allows a court to submit an electronic disposition to KBI. This method will only provide a return code based on initial XML validation against the Kansas Disposition Report IEPD schema. Once the XML passes validation against the schema, the submitting ORI/submitting On Behalf Of ORI will be checked for authorization. Once the XML passed both check points, the XML is placed in a processing queue otherwise rejected. From the processing queue, the XML will be processed according to the defined disposition processing rules and either added to CCH or moved to the Errant Disposition queue for manual intervention.

Parameters

*XMLDoc* DispositionXML - XML string containing disposition data. Submitting ORI and Submitting on Behalf of ORI must be included in this input XML. See the *Kansas Disposition Report IEPD* for additional details.

Return Values/Error Codes

The following codes are returned to the caller:

* 1 for SUCCESS - XML passed validation against Kansas Disposition Report IEPD schema.
* -1 for FAILED\_INVALIDXML - XML failed Kansas Disposition Report IEPD schema validation.
* -2 for FAILED\_UNAUTHORIZEDSUBMISSION – Submitting ORI/Submitting on Behalf Of ORI is not authorized based on KBI’s defined criteria.
* -3 for FAILED\_ADDTOPROCESSQUEUE - Failure to add to the process queue

Conditional Handling for Electronic Submission

* If a disposition passes XML validation but fails Hard Rules, electronic submissions go into errant queue
* If a disposition fails Soft Rules, electronic submissions go into errant queue
* If a disposition is a Summons (it won’t have an arrest record), electronic submissions go into errant queue (orphan)
* Any other error condition, electronic submissions go into errant queue.

GetErrantDispositions()

Description

This method allows a court to get a list of dispositions that passed the initial schema validation but ended up in the Errant Disposition queue.

Parameters

* *String* ORI - parameter containing ORI which submitted the disposition.
* *Datetime* DateProcessed - date when disposition was processed.
* *String* Status - status of data to be retrieved:
	+ ALL – when user wants a list of all dispositions in the errant queue
	+ REJECTED - when user wants a list of rejected dispositions (failed hard rules)
	+ REVIEWABLE - when user wants a list of reviewable dispositions (passed hard rules, failed soft rules).
	+ ORPHAN - when user wants a list of orphan dispositions.
	+ SUPPLEMENTAL – when user wants a list of supplemental dispositions (rejected, reviewable, or orphan)
* *Boolean* Archive - flag indicating if data is archived or not.

Return Value

XML data string matching the specified parameters with the following elements:

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Meaning** | **DataType** | **Length** |
| DispositionID | Guid internal ID for system tracking dispositions | GUID | GUID |
| TransactionNumber | Dispositions’ Transaction Number | String | 12 |
| AgencyORI | ORI of the Agency submitting the disposition | String  | 9 |
| ArrestCaseNumber\* | Arresting Agency Case Number | String | 20 |
| CourtCaseNumber\* | Court Case Number | String | xx |
| LastName | Subject Last Name | String | 30 |
| FirstName | Subject First Name | String  | 20 |
| ErrorCode | Sum of the Error Code bit values described below under “Errant Dispositions Error Codes” | Integer |  |
| ProcessedDate |  | DateTime |  |
| Archived | Flag indicating if the Errant Disposition transmission has been archived. Archived dispositions do not appear on the Errant Dispositions Report / UI on the portal. Relates to the “Archived” parameter in the GetErrantDispositions() method. | Boolean | 1 |
| \*  DetailError | Detail error string describing errors encountered on the field level validation | String | 1000 chars |

*\* Version 1.2.3 The prior CaseNumber element is renamed to ArrestCaseNumber to be explicit, the CourtCaseNumber node is added to the output and the DetailError node is added to the output XML.*

Note: If there is an error in calling the GetErrantDispositions() method or there are no data resulting from the method call per parameters supplied, the return value is an empty, serialized dataset.

Example dataset result containing errant dispositions**:**

<DocumentElement> <ErrantDispositions> <DispositionId>4</DispositionId> <TransactionNumber>3XXXXXXXXXXX</TransactionNumber> <AgencyORI>KS0870300</AgencyORI> <LastName>KENT</LastName> <FirstName>CLARK</FirstName> <CaseNumber>12345</CaseNumber> <DateProcessed>2013-11-27T21:09:22.57-05:00</DateProcessed> <ErrorCode>518</ErrorCode> <Archived>0</Archived> </ErrantDispositions></DocumentElement>

**Method Signature**

public string GetErrantDispositions(string ORI, DateTime ProcessedDateFrom, DateTime ProcessedDateTo, string Status, bool Archive)

### Examples are below.  Note that there is an Error Code field in the dataset.

### Errant Dispositions Error Codes

The following list describes the error codes that may be returned from the GetErrantDispositions() method for dispositions that erred when processed against rules validation.

  The Error column in the returned dataset represents the error code (a combination of binary bits) resulting from the processing of each disposition submitted.

public enum Flags : ulong //unsigned 64-bit

{

    /// <summary>

    /// SUCCESS - Successful.

    /// </summary>

    SUCCESS = 0x0000000000000000,

    /// <summary>

    /// FAILED\_REVIEWREJECT - Rejected when being reviewed.

    /// </summary>

    FAILED\_REVIEWREJECT = 0x0000000000000001,

    /// <summary>

    /// FAILED\_ORPHAN - Did not find a matching arrest.

    /// </summary>

    FAILED\_ORPHAN = 0x0000000000000002,

    /// <summary>

    /// FAILED\_HARDRULES - Failed hard validation rules.

    /// </summary>

    FAILED\_HARDRULES = 0x0000000000000004,

    /// <summary>

    /// FAILED\_SOFTRULES - Failed Soft validation rules.

    /// </summary>

    FAILED\_SOFTRULES = 0x0000000000000008,

    /// <summary>

    /// FAILED\_INVALIDSTATUTE - Invalid Statute.

    /// </summary>

    FAILED\_INVALIDSTATUTE = 0x0000000000000010,

    /// <summary>

    /// FAILED\_DUPLICATE - A duplicate segment already exist in CCH Database.

    /// </summary>

    FAILED\_DUPLICATE = 0x0000000000000020,

    /// <summary>

    /// FAILED\_WAIVEDTOADULT - KJDR Form is used to submit disposition for a juvenile that was waived to adult status.

    /// </summary>

    FAILED\_WAIVEDTOADULT = 0x0000000000000040,

    /// <summary>

    /// FAILED\_MISSINGTRANSNO - Transaction number is missing.

    /// </summary>

    FAILED\_MISSINGTRANSNO = 0x0000000000000080,

    /// <summary>

    /// FAILED\_SAVEDASORPHAN - An orphan disposition is forced to be saved as orphan.

    /// </summary>

    FAILED\_SAVEDASORPHAN = 0x0000000000000100,

    /// <summary>

    /// FAILED\_SUPPLEMENTAL - Disposition is submitted as supplemental but no existing segments are found on the CCH Database.

    /// </summary>

    FAILED\_SUPPLEMENTAL = 0x0000000000000200,

    FAILED\_MISSINGCHARGES = 0x0000000000000400,

    FAILED\_INVALIDFIELDVALUE = 0x0000000000000800,

    FAILED\_SUPPLEMENTAL\_SEGMENT = 0x0000000000001000,

    FAILED\_ADDDISPOSITION = 0x0000000000002000,

    FAILED\_ADDERRANT = 0x0000000000004000,

    zFAILED\_Reserve6 = 0x0000000000008000,

    zFAILED\_Reserve7 = 0x0000000000010000,

    zFAILED\_Reserve8 = 0x0000000000020000,

    zFAILED\_Reserve9 = 0x0000000000040000,

    zFAILED\_Reserve10 = 0x0000000000080000,

    zFAILED\_Reserve11 = 0x0000000000100000,

    zFAILED\_Reserve12 = 0x0000000000200000,

    zFAILED\_Reserve13 = 0x0000000000400000,

    zFAILED\_Reserve14 = 0x0000000000800000,

    zFAILED\_Reserve15 = 0x0000000001000000,

    zFAILED\_Reserve16 = 0x0000000002000000,

    zFAILED\_Reserve17 = 0x0000000004000000,

    zFAILED\_Reserve18 = 0x0000000008000000,

    zFAILED\_Reserve19 = 0x0000000010000000,

    zFAILED\_Reserve20 = 0x0000000020000000,

    zFAILED\_Reserve21 = 0x0000000040000000,

    zFAILED\_Reserve22 = 0x0000000080000000

}

\*The <DetailError> node will contain the detail such as “<DetailError> Hard Rule Violation[Sex]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;</DetailError>”

|  |
| --- |
| **Detail Error Code Legend:*** Hard Rule violation
* Duplicate
* Soft Rule violation
* Invalid Field
* Missing Charges
* Waived to Adult
* Orphan because no matching arrest was found
* Orphan due to blank transaction number
* Attempt to add disposition to CCH failed.
* Supplemental flag is true plus no matching arrest found
* Supplemental flag is true and prosecution and court segments exist and data diversion revoked has a value.
* Invalid Statute

\*\* Fields will be indicated in [square brackets]\*\*NOTE: Detail error string will be truncated to 1000 chars**Sample detail error messages:**C1\_4 TN:3000L0000002  ORPHAN; Result=Added to Errant;C1\_5 TN:3000L0000002  Hard Rule Violation[Sex]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_6 TN:3000L0000002  Hard Rule Violation[Race]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_7 TN:3000L0000002  Hard Rule Violation[ProsecutorActionDate]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_8 TN:3000L0000002  Hard Rule Violation[ProsecutorActionDate]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_9 TN:3000L0000002  Hard Rule Violation[ProsecutorActionType]; Missing Charges[R=1;P=0;C=0]; Duplicate[TransNo]=3000L0000002; Result=Added to Errant;C1\_13 TN:3000L0000002  Missing Charges[R=1;P=0;C=0]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_15 TN:3000L0000002  Hard Rule Violation[TypeofCounsel]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_16 TN:3000L0000002  Hard Rule Violation[CourtORI]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_17 TN:3000L0000002  Invalid Field[DefenseAttorney]; Duplicate[TransNo=3000L0000002]; Result=Added to Errant;C1\_18 TN:3000L0000002  Duplicate[TransNo=3000L0000002]; Result=Added to Errant; |

### Disposition Processor Description

The Disposition Processor Assembly is a reusable assembly that is utilized both by SharePoint-based submission and electronic submission of filings and dispositions. This assembly validates the input XML's data contents against a set of business rules. Dispositions passing the business rules validation are processed into CCH, while dispositions failing the validation are processed as Errant Dispositions.

# Detailed Interface Requirements

## Interface Processing Time Requirements

Electronic Filing and Disposition submissions may be submitted to the web service any time. After an initial validation, submissions will be placed within a queue for processing into the CCH system and the next submission will be evaluated for acceptance. Next, a Disposition Processor Console Job, which runs on a regular interval, will loop through the contents of the process queue and validate the XML data contents against the KADR-KJDR Business Rules. Submissions passing the validation are then processed into CCH while submissions which fail are processed as Errant Dispositions.

## Interface Initiation

An XML string will be passed to the CreateDispositionAuto() method of the web service. The location of the web service and full URL will be communicated closer to implementation.

## Message Requirements

* XML string meets the Kansas Disposition Report IEPD schema requirements
* One filing & disposition record per XML string

## Flow Control

When an XML string is received, it is validated against the message requirements listed above. If the XML string meets the requirements:

* The XML string is added to the Disposition Queue for processing by a separate Disposition Processor Console Job
* A response of 1 (positive 1) for “SUCCESS” is returned to the submitting process

If the XML string does not meet the requirements:

* A response of -1 (negative 1) for “FAILED\_INVALIDXML” is returned to the submitting process if the XML does not comply with the schema
* A response of -2 (negative 2) for “FAILED\_UNAUTHORIZEDSUBMISSION” is returned to the submitting process if the Submitting ORI/Submitting on Behalf of ORI is unauthorized based on KBI’s criteria.

# Appendices

## Kansas Disposition Report IEPD

The latest version of the Kansas Disposition Report IEPD can be found on the project portal in the Deliverables section.

## KADR-KJDR Business Rules

The latest version of the KADR-KJDR Business Rules can be found on the project portal in the Deliverables section.

## Filings and Disposition Comprehensive Field Mapping Document

This document shows the association of each field/element of a disposition between the:

* Online KADR/KJDR form
* Kansas Disposition Report IEPD
* KADR/KJDR Business Rules
* Internal Disposition XML Schema

The latest version of the Filings and Dispositions Comprehensive Field Mapping Document can be found on the project portal in the Deliverables section.

# Document Management

## History

The following table gives a record of major changes to the document. The version number listed does not include the revision number, which indicates minor version numbers.

|  |  |  |  |
| --- | --- | --- | --- |
| Version  | Date | Author | Change |
| 1.0 | 9/3/2013 | Steve Sutton | Initial Version |
| 1.1 | 12/2/2013 | Cindy Isip | Updated to conform to design updates |
| 1.2 | 12/18/2013 | Trevor Scoggins, Cindy Isip | Updated with more details on GetErrantDispositions() method including list of error return codes |
| 1.2.1 | 12/24/2013 | Steve Sutton | Updated GetErrantDisposition. Returns XML data string instead of data table. |
| 1.2.2 | 3/13/2014 | Cindy Isip, Trevor Scoggins | Updated   GetErrantDispos() Return Values FAILED\_ADDDISPOSITION = 0x0000000000002000, FAILED\_ADDERRANT = 0x0000000000004000, |
| 1.2.3 | 4/23/2014 | Trevor Scoggins | XML Output: The prior CaseNumber element is renamed to ArrestCaseNumber, the CourtCaseNumber node is added to the output and the DetailError node is added to the output XMLDetail Error values no longer use a legend, but explicit text describing errant fields. |