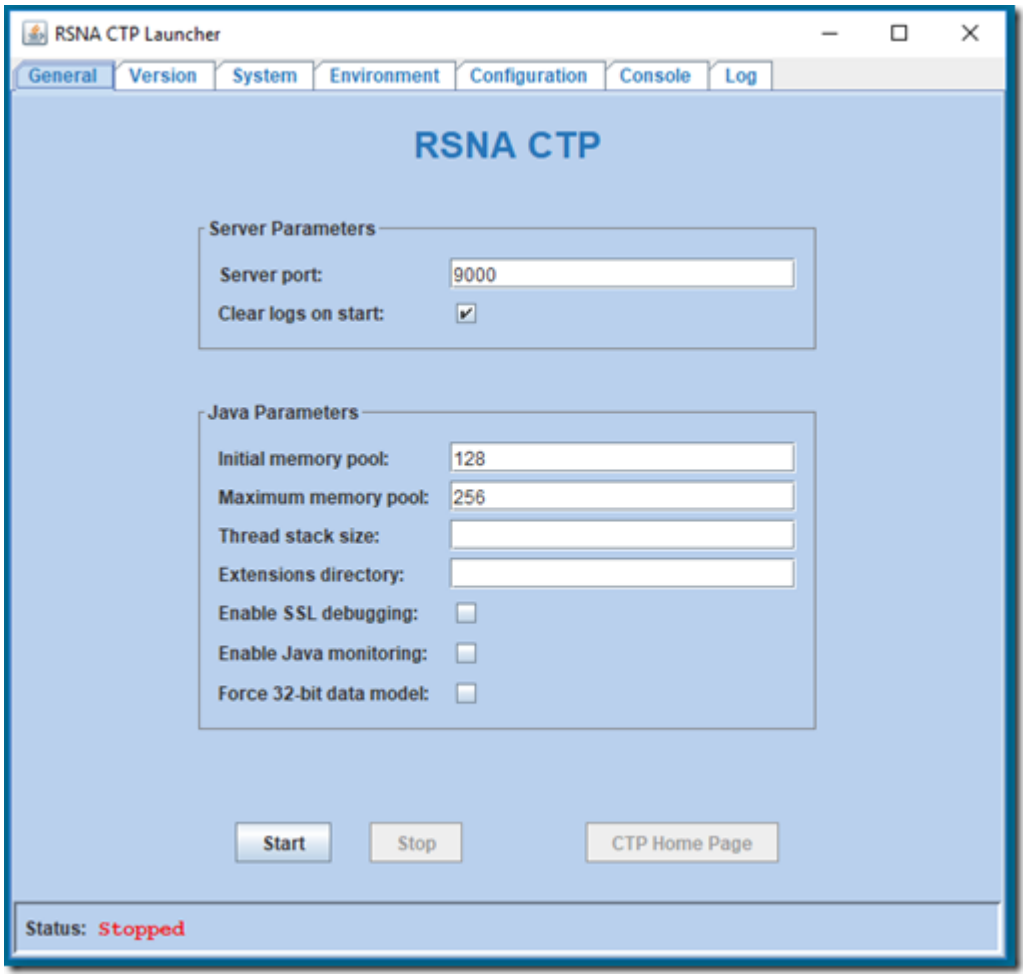
The TCIA Plugin and Servlet API

Table of Contents

1 Installing the TCIAPlugin in a CTP instance:

1. Install or upgrade to the latest CTP.
2. Put the TCIAPlugin.jar file in the **CTP/libraries** folder.
3. Put the special TCIA version of the config.xml file (see below) in the **CTP** folder.
4. Put any web pages, css files, Javascript files, and wizard files in the **CTP/ROOT** directory tree, being careful not to overwrite the index.xml file.
5. Run the Launcher program:



6. The config.xml file puts the CTP server on port 9000. It can be changed by altering the **Server port** field.
7. Click the **Start** button.
8. The **CTP Home Page** button will then be enabled. Click it.
9. The default browser will start and display the home page.
10. Log in as username=**admin**, password=**password**.
11. To shut down the server, click the **Stop** button on the Launcher window (or just close the Launcher).

2 The config.xml file

The config.xml for the TCIA project is:

```
<Configuration>
    <Server
        maxThreads="20"
        port="9000"/>
    <Plugin
        anonymizerID="CollectionDicomAnonymizer"
        anonymizerInputID="CollectionAnonymizerInput"
        anonymizerStorageID="CollectionAnonymizerStorage"
        class="edu.uams.tcia.TCIAPlugin"
        exportInputID="CollectionExportInput"
        exportManifestLogID="CollectionManifestLog"
        exportOutputID="CollectionExportOutput"
        id="Collection"
        importInputID="CollectionImportInput"
        importStorageID="CollectionImportStorage"
        name="CollectionTCIAPlugin"
        root="Collection/TCIAplugin"/>
    <Plugin
        class="edu.uams.tcia.ManifestLogPlugin"
        exportURL="http://192.168.0.225:7778"
        id="CollectionManifestLog"
        manifestLogTags="PatientID;StudyDate;SeriesInstanceUID"
        name="CollectionManifestLog"
        root="Collection/ManifestLog"/>
    <Pipeline
        name="CollectionImport"
        root="Collection/import">
        <DirectoryImportService
            class="org.rsna.ctp.stdstages.DirectoryImportService"
            id="CollectionImportInput"
            import="roots/DirectoryImportService/import"
            name="DirectoryImportService"
            quarantine="quarantines/DirectoryImportService"
            root="roots/DirectoryImportService"/>
        <DicomImportService
            class="org.rsna.ctp.stdstages.DicomImportService"
            logConnections="no"
            name="DicomImportService"
            port="104"
            quarantine="quarantines/DicomImportService"
            root="roots/DicomImportService"/>
        <DirectoryStorageService
            acceptDuplicates="yes"
            class="org.rsna.ctp.stdstages.DirectoryStorageService"
            defaultString=""
            id="CollectionImportStorage"
            logDuplicates="no"
            name="DirectoryStorageService"
            quarantine="quarantines/DirectoryStorageService"
            root="roots/DirectoryStorageService"
            setStandardExtensions="no"
            structure="(0010,0020)-(0010,0010)/(0008,0020)"
            whitespaceReplacement="_"/>
```

```
    </Pipeline>
    <Pipeline
        name="CollectionAnonymization"
        root="Collection/anonymization">
        <DirectoryImportService
            class="org.rsna.ctp.stdstages.DirectoryImportService"
            id="CollectionAnonymizerInput"
            import="roots/DirectoryImportService/import"
            name="DirectoryImportService"
            quarantine="quarantines/DirectoryImportService"
            root="roots/DirectoryImportService"/>
        <ObjectCache
            class="org.rsna.ctp.stdstages.ObjectCache"
            id="CollectionObjectCache"
            name="ObjectCache"
            root="roots/ObjectCache"/>
        <DicomAnonymizer
            class="org.rsna.ctp.stdstages.DicomAnonymizer"
            id="CollectionDicomAnonymizer"
            lookupTable="scripts/LookupTable.properties"
            name="DicomAnonymizer"
            quarantine="quarantines/DicomAnonymizer"
            root="roots/DicomAnonymizer"
            script="scripts/DicomAnonymizer.script"/>
        <ManifestLogger
            class="edu.uams.tcia.ManifestLogger"
            id="CollectionManifestLogger"
            manifestLogID="CollectionManifestLog"
            name="ManifestLogger"
            root="roots/CollectionManifestLogger"/>
        <DirectoryStorageService
            acceptDuplicates="yes"
            cacheID="CollectionObjectCache"
            class="org.rsna.ctp.stdstages.DirectoryStorageService"
            defaultString=""
            id="CollectionAnonymizerStorage"
            logDuplicates="no"
            name="DirectoryStorageService"
            quarantine="quarantines/DirectoryStorageService"
            root="roots/DirectoryStorageService"
            setStandardExtensions="no"
            structure="(0010,0020)-(0010,0010)/(0008,0020)"
            whitespaceReplacement="_"/>
    </Pipeline>
    <Pipeline
        name="CollectionExport"
        root="Collection/export">
        <DirectoryImportService
            class="org.rsna.ctp.stdstages.DirectoryImportService"
            id="CollectionExportInput"
            import="roots/DirectoryImportService/import"
            name="DirectoryImportService"
            quarantine="quarantines/DirectoryImportService"
            root="roots/DirectoryImportService"/>
        <HttpExportService
            class="org.rsna.ctp.stdstages.HttpExportService"
            id="CollectionExportOutput"
```

```
            name="HttpExportService"
            quarantine="quarantines/HttpExportService"
            root="roots/HttpExportService"
            sendDigestHeader="no"
            url="http://192.168.0.225:7777"/>
    </Pipeline>
    <Pipeline
        name="DummyExportReceiver"
        root="Collection/dummy/dicom">
        <HttpImportService
            class="org.rsna.ctp.stdstages.HttpImportService"
            logConnections="no"
            name="HttpImportService"
            port="7777"
            quarantine="quarantines/HttpImportService"
            root="roots/HttpImportService"/>
    </Pipeline>
    <Pipeline
        name="DummyManifestReceiver"
        root="Collection/dummy/manifest">
        <HttpImportService
            class="org.rsna.ctp.stdstages.HttpImportService"
            logConnections="no"
            name="HttpImportService"
            port="7778"
            quarantine="quarantines/HttpImportService"
            root="roots/HttpImportService"/>
    </Pipeline>
</Configuration>
```

The configuration has three main pipelines:

1. CollectionImport receives and stores DICOM files.
2. CollectionAnonymization anonymizes and stores files.
3. CollectionExport transmits files to the principal investigator.

The configuration has two additional pipelines just for testing:

1. DummyExportReceiver receives DICOM files from the CollectionExport pipeline.
2. DummyManifestReceiver receives manifest submissions.

The configuration has two Plugins:

- The TCIAPlugin provides the interface into the pipelines for the wizard.
- The ManifestLogPlugin accumulates manifest information for submission to the principal investigator.

3 User Accounts

When CTP starts, the TCIAPlugin changes the password of the **admin** user to **tcia** and grants it these roles:

- The **admin** role grants access to the administrative roles on the server.
- The **TCIA** role grants access to the TCIAServlet.

- The **qadmin** role grants access to the QuarantineServlet.
- The **shutdown** role allows the wizard to shut down CTP.

Additional users can be created through the User Manager on the CTP home page by logging in as a user with the **admin** role.

The wizard can automatically log in the user is as tcia/tcia, or it can provide a UI that gets the username and password from the user and then make the call to the LoginServlet. The URL of the LoginServlet is:

/login/ajax?username=...&password=...

The login returns either a 200 or a 403 response code.

To provide a logout feature, the wizard can use this URL:

/login/ajax?logout

This call always returns 200.

4 The TCIAServlet API

This section describes the functions provided to the wizard by the TCIAServlet. All functions are accessed by making an HTTP connection to the CTP server. Unless otherwise indicated, all functions return an HTTP response with **Content-Type: text/xml;charset=UTF-8**.

4.1 Updating the DicomAnonymizer Lookup Table

To update the anonymizer lookup table from an Excel spreadsheet, the wizard does an HTTP POST of the file (with Content-Type **multipart/form-data)** to the URL:

/Collection

The spreadsheet must be an xlsx file with this structure:

	A	B	C	D
1		ptid	cname	ddate
2	**Local Patient ID**	**Anonymized Patient ID**	**Collection Name**	**Diagnosis Date (M/D/YYYY)**
3	657662	12345	A Collection Name	4/15/2014
4	5555	4444	Another Collection Name	2/1/2011
5	etc.			
6				

The first row contains the KeyTypes used in the lookup table for the data in each spreadsheet column. They must not be modified.

The second row provides a human-readable label for the column. The TCIA servlet needs the first row. It ignores the second row and starts processing rows at the third row.

4.2 Submitting Files to the Import Pipeline

4.2.1 Listing Files on the Server

To list the files in a directory on the server, the wizard does an HTTP GET to:

/Collection/listFiles?dir=path

The servlet returns an XML structure like this:

```
<dir
    name="...directory name..."
    parent="...absolute path to the parent directory...">
    <dir name="...child directory 1 name..."/>
    <dir name="...child directory 2 name..."/>
    ...
    <dir name="...child directory n name..."/>
    <file name="...child file 1 name..."/>
    <file name="...child file 2 name..."/>
    ...
    <file name="...child file n name..."/>
</dir>
```

4.2.2 Submitting Files to the DirectoryImportService

To submit a list of files to the DirectoryImportService in the import pipeline, the wizard does an HTTP GET to:

/Collection/submitFile?file="path sequence"

where **path sequence** is a string of paths separated by pipe characters, like this:

absolutePath-1|absolutePath-2|...|absolutePath-n

The servlet copies the files to the import pipeline. An absolutePath that points to a file imports that file. An absolutePath that points to a directory imports all the files in the directory and all of its child directories recursively.

The servlet returns an XML structure indicating whether the submissions all succeeded (<OK/>) or at least one failed (<NOTOK/>).

4.2.3 Submitting Files to the DicomImportService

The DicomImportService receives DICOM transfers on port 104. No configuration of the AE Titles is necessary; the SCP accepts all AE Titles.

4.3 Viewing the List of Files Ready for Anonymization

To obtain the list of files that have been received but not yet anonymized, the wizard does an HTTP GET to the URL:

/Collection/listImport

The AJAX call returns an XML structure like this:

The XML structure mimics the directory structure in which the files are stored. Each **dir** element in the XML structure represents a directory. The top-level directory is the root directory of the DirectoryStorageService pipeline stage that contains the files received by the import pipeline. Under the top-level directory, there is one directory for each patient. Under a patient's directory is one directory for each date on which the patient had a study. The study date directory contains all the images for studies done on that date. Individual image files are represented in the XML structure by **DicomObject** elements.

The relative path to a directory can be obtained by walking the tree from the point in question back to the top. Thus, the relative path to the **20010312** directory is:

DirectoryStorageService/1200824338-Bunny,Bugs/20010312

The base of the relative path is the root of the pipeline stage.

At this point, the wizard can display some representation of the XML structure, for example:

Date	Patient	Modality	Number of Images
20010312	**1200824338-Bunny,Bugs**	**CT**	**40**
20151017	**33367-Duck,Daisy**	**MR**	**255**

[The wizard might also allow the user to expand a line to list the images and then click an image to see the image itself and/or a listing of the DICOM elements in the image. The TCIA servlet doesn't currently provide support for either of those functions.]

4.4 Submitting Files for Anonymization

To anonymize studies that were listed in 4.2, above, the wizard passes some level in the hierarchy (a patient, a single study, or the top-level directory) to the servlet in an HTTP GET to the URL:

/Collection/anonymize?file=filepath

where **filepath** is the relative path. If the filepath is a directory, the servlet processes all the files in that directory and all its child directories. For example, to process everything that has been received, the URL would be:

/Collection/anonymize?file=DirectoryStorageService

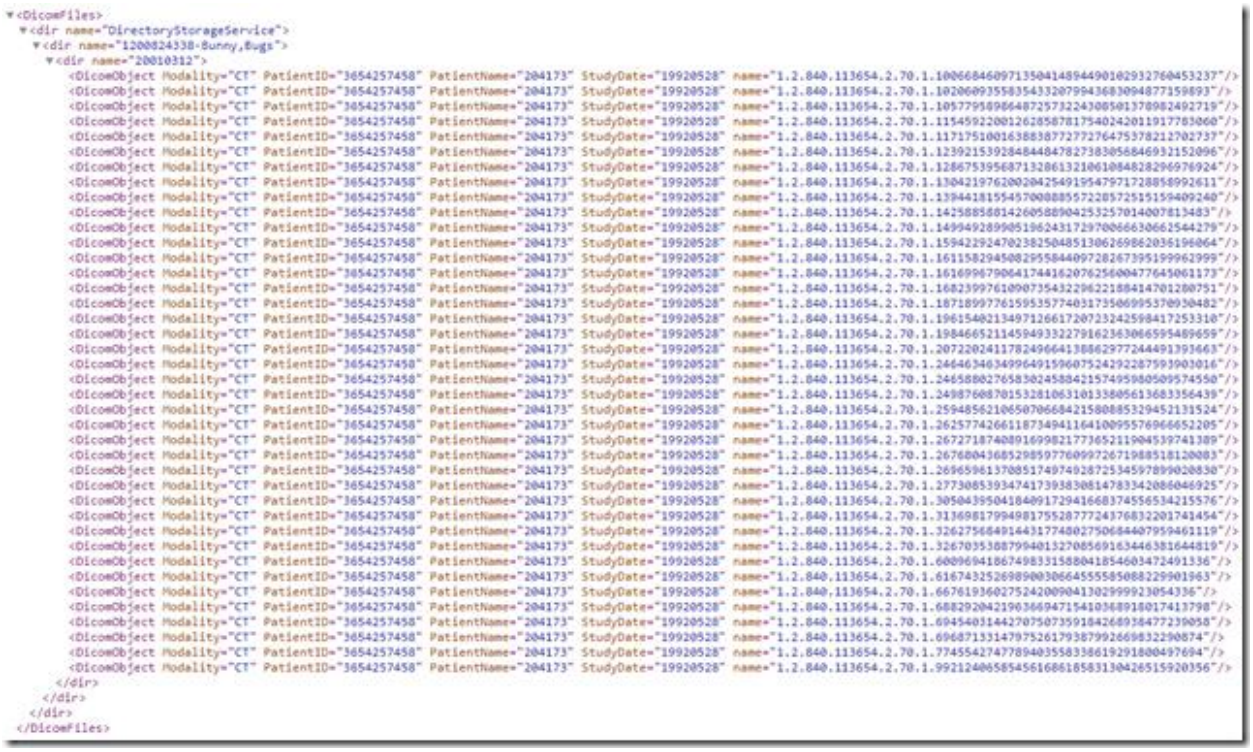
The servlet moves the files from the import pipeline to the anonymizer pipeline to start the anonymization. The servlet returns an XML structure with one element <OK/> or <NOTOK/> to indicate whether all the moves were successful.

4.5 Viewing the List of Files Ready for Export

To obtain the list of files that have been anonymized but not yet exported, the wizard does an HTTP GET to the URL:

/Collection/listAnonymized

The AJAX call returns an XML structure like this:



The XML structure exactly mimics the one that listed the imported files, but it references a different DirectoryStorageService pipeline stage, so although the directory names are the same, they are relative to a different root directory.

[This is probably also a good place to provide file viewing capabilities to reassure the user that they are not exporting PHI.]

4.6 Exporting Files

To export studies that were listed in 4.4, above, the wizard passes some level in the hierarchy (a patient, a single study, or the top-level directory) to the servlet in an HTTP GET to the URL:

/Collection/export?file=filepath

where **filepath** is again the relative path.

The servlet moves the files from the anonymizer pipeline to the export pipeline to start the export. The servlet returns an XML structure with one element <OK/> or <NOTOK/> to indicate whether all the moves were successful.

4.7 The Manifest

As files are anonymized, a manifest listing key identifiers from each file is created. The TCIAServlet provides four functions for controlling the process:

4.7.1 Clearing the Manifest

The wizard can clear the manifest by an HTTP GET to the URL:

/Collection/clearManifest

The servlet returns an XML structure with one element <OK/> or <NOTOK/> to indicate whether all the operation was successful.

4.7.2 Listing the Manifest as XML

The wizard can obtain an XML structure containing the manifest by an HTTP GET to:

/Collection/listManifest/xml

This AJAX call returns an XML structure like this:


```
▼<Manifest>
 ▼<Entry SOPInstanceUID="1.2.840.113654.2.70.1.313698179949817552877724376832201741454">
    <Element name="PatientID" value="3654257458"/>
    <Element name="StudyDate" value="19920528"/>
    <Element name="SeriesInstanceUID" value="1.2.840.113654.2.70.1.113479017276406998463382173796105434129"/>
  </Entry>
 ▼<Entry SOPInstanceUID="1.2.840.113654.2.70.1.187189977615953577403173506995370930482">
    <Element name="PatientID" value="3654257458"/>
    <Element name="StudyDate" value="19920528"/>
    <Element name="SeriesInstanceUID" value="1.2.840.113654.2.70.1.113479017276406998463382173796105434129"/>
  </Entry>
 ▼<Entry SOPInstanceUID="1.2.840.113654.2.70.1.616743252698900306645555585088229901963">
    <Element name="PatientID" value="3654257458"/>
    <Element name="StudyDate" value="19920528"/>
    <Element name="SeriesInstanceUID" value="1.2.840.113654.2.70.1.113479017276406998463382173796105434129"/>
  </Entry>
 ▼<Entry SOPInstanceUID="1.2.840.113654.2.70.1.262577426611873494116410095576966652205">
    <Element name="PatientID" value="3654257458"/>
    <Element name="StudyDate" value="19920528"/>
    <Element name="SeriesInstanceUID" value="1.2.840.113654.2.70.1.113479017276406998463382173796105434129"/>
  </Entry>
 ▼<Entry SOPInstanceUID="1.2.840.113654.2.70.1.123921539284844847827383056846932152096">
    <Element name="PatientID" value="3654257458"/>
```

The elements listed in the manifest object are determined by the manifestLogTags attribute of the ManifestLogPlugin element in the CTP config.xml file.

4.7.3 Listing the Manifest as CSV

The wizard can obtain a text string containing the manifest as a spreadsheet by making an AJAX call to:

/Collection/listManifest/csv

This AJAX call returns a CSV text response with **Content-Type: text/csv;charset=UTF-8**.

4.7.4 Exporting the Manifest

The wizard can sent the manifest to the principal investigator by an HTTP GET to:

/Collection/exportManifest

The servlet returns an XML structure with one element <OK/> or <NOTOK/> to indicate whether the transmission was successful.

4.8 Image Functions

4.8.1 Listing the Elements of a DICOM File

The wizard can obtain an HTML page containing a table showing the values of the elements in a DICOM file by an HTTP GET to the URL:

/Collection/listElements?file=filepath

If the servlet cannot find the file, or if it cannot be parsed as a DICOM object, the servlet returns a 404 response code.

4.9 Special Functions

4.9.1 Getting the Available Space on the Server

The wizard can obtain an XML structure indicating the available space on the partition on which the CTP instance is located by an HTTP GET to the URL:

/Collection/getAvailableSpace

This servlet returns an XML structure like this:

```
<space partition="D:\" available="434932" units="MB"/>
```

4.9.2 Getting the Export Queue Size

The wizard can obtain an XML structure indicating the current size of the export queue by an HTTP GET to the URL:

/Collection/getExportQueueSize

This servlet returns an XML structure like this:

```
<queue stage="HttpExportService" size="10"/>
```

4.9.3 Getting the URL of the Quarantine Manager Servlet

The wizard can obtain an XML structure containing the URL of the CTP Quarantine Manager for the DicomAnonymizer quarantine by an HTTP GET to the URL:

/Collection/getQuarantineURL

This servlet returns an XML structure like this:

```
<quarantine stage="DicomAnonymizer" url="/quarantines?p=1&s=2"/>
```

4.9.4 Getting a Summary of the DicomAnonymizer Quarantine

The wizard can obtain an XML structure containing a summary of the objects in the DicomAnonymizer quarantine by an HTTP GET to the URL:

/Collection/getQuarantineSummary

This call returns an XML structure like this:

… TBD …

4.9.5 Shutting Down the Server

The wizard can shut down the CTP server by an HTTP GET to:

/shutdown

The system returns either a 200 or 403 response code. It fails if the user does not have the **shutdown** role. This URL accesses the CTP ShutdownServlet directly. It returns a web page like this:

> Shutdown request received from admin at 192.168.0.225.
> Goodbye.

4.10 Testing Functions

During testing, it may be convenient to clear all the import/export directories, the quarantines, the queues, and the manifest. This can be done by an HTTP GET to the URL:

/Collection/reset

The servlet does all it can and returns an XML structure with one element <OK/>. It never returns <NOTOK/>.