CrystalPM API Documentation

Version Info:

CrystalPM Version: 5.0
API Version: 1.0
API Documentation Version: 1.0
API Documentation Last Updated: 05/17/2017

Overview:

The CrystalPM API enables third-party software vendors to securely retrieve patient data through a command based interface. It is possible to only retrieve patient data for a specific category (i.e. name, date of birth, medications, procedures, etc...) or a CCD containing all data for a desired time period.

The retrieved patient data is exported in a standardized XML format. Progress is logged through two files (TXT and XML). The output paths and file names can be customized.

Setup:

CrystalPM needs to be installed. Make sure you're running the latest version available.

The CPM_API_Client.exe is in the CrystalPM installation folder (i.e. "C:\Program Files (x86)\CrystalPM"). You can run it directly through the Windows command line with arguments (i.e. "C:\Program Files (x86)\CrystalPM\CPM_API_Client.exe -command getUsers..."). The same process can be performed programmatically. An example of this is in the CPM_API_Demo project, which is included with this documentation. You will need Microsoft Visual Studio 2015 to open the project.

It's recommended that you convert the provided XSD schema files into code for your application. This will generate the data types necessary to deserialize the patient data and log XML files. If your application is built on Microsoft .Net, you would used the xsd.exe for the target framework version (i.e. 4.0).

If the installation of CrystalPM you're working with is up-to-date, there will be a user with the first and last name, "API Access." You can get the user ID by using the "getUsers" command, and the password is "M3XahAcespuprewa" by default. Any master administrator user can change this password or limit the permissions for this user.

Data Retrieval:

<u>TXT log file OR outputFileName_log.txt</u>

This is the raw text log file. It's only meant to be a human readable log.
Any line beginning with "Error:" is a critical error.
Any line beginning with "Warning:" is due to the lack of data retrieved.

XML log file OR outputFileName_log.xml

This is the xml log file. It's meant to be a computer readable log.
The XML contained can be deserialized using the OutputCollection data type.
Any normal (non-error) logs are stored in the LogList of an Output.
Any error logs are stored in the ErrorList of an Output.
The ErrorEncountered of an Output represents whether or not an error was encountered.
Any generated patient data files are stored in the OutputFileList of an Output.
The Id of an output represents a unique GUID that was generated for the command.
The TimeStamp of an Output is the date and time the command was run.

OutputFileName

It is highly recommended that you have a different output file name each time you run a command, especially in an asynchronous environment. Otherwise you will completely clear and overwrite the previous XML file (outputFileName.xml). However, it will not clear the TXT or XML log file. In the TXT log file, it will add log lines. In the XML log file, it will add an Output to OutputCollection for each command.

XML file OR outputFileName.xml

This is XML file that will contain the data requested by the command.
The data type is different for each command. When retrieving patient data by category, each category will has its own data type.

If this file exists at the end of a command's execution, which can be verified by checking the OutputCollection->OutputList->Output[X]->OutputFileList), deserialize the contained XML into an instance of the correct data type.

Data Types:

Most data types are nullable. Int, long, and char are examples of data types that aren't by default. Even those types might be nullable, depending on the containing data type.

The names of base data types and their members are not ambiguous and are self-explanatory. If you're not sure what a base data type or member relates to, please contact our main support line.

The structure of the base data types is implied by the schemas and the code that can be generated by them.

Collections:

Some data types have “Collection” in the name, and this means they contain a list of items of a data type. If the list contained is empty, you can assume that no values were found.

List OR Array:

This is the main collection data type. Like any nullable data type, it needs to be checked if the value contained is null. It also needs to be checked for being empty. It can contain multiple items of a specific data type.

Integer, Long, Char:

Unless specified otherwise (as nullable), these types will always have a value.

Nullable Data Types:

Most data types are nullable. Instances and members that are nullable need to be checked if they contain a value (not null) before the members inside them can be accessed.

Errors and Exceptions:

Error and exception messages are placed in the TXT and XML (Output->ErrorList) log files. If an exception is thrown, it will be caught by the API.

Possible Errors:

<u>**“**Error: failed to connect to CrystalPM database”</u>

This error occurs when the API can't connect to and authenticate with the CrystalPM MySQL database.

<u>“Error: Failed to run command”</u>

This means the API failed to run a command. It might be a sign that the connection with the database was lost.

<u>“Error: Argument or value is malformed or missing”</u> or other error related to parameter

The command passed to the API is missing a parameter or the parameter syntax is off.

<u>“Error: Argument is missing and is required for the command”</u>

The command is missing a required parameter.

<u>“Error: Could not find use with ID”</u>

The user ID passed with the command is not associated with a user.

<u>“Error: User password does not match.”</u>

The user password passed with the command does not match the user's password.

“Error: Failed to get <data type>”

The API failed to retrieve the desired data from the database. This could be related to a connection loss with the database.

“Error: Failed to create <data type> XML file”

This might relate to a lack of permissions of your user account to create files in the specified output path.

Warnings:

Warnings are not critical errors. They're due to the lack of data retrieved for the parameters passed. They're placed in the TXT and XML (Output->ErrorList) log files.

Possible Warnings:

"Warning: user doesn't have permissions to access..."

This warning occurs when the user you used to generate a patient security token does not have the permissions to perform data retrieval.

"Warning: the security token is expired"

This warning occurs when the security token associated with a passed token is expired and is no longer valid. Another one will need to be generated using the "getPatientToken" command.

"Warning: Failed to find matching patient"

The provided demographics (first name, last name, etc...) did not match on a patient.

Best Practices:

Mimic the following steps when using the API in an automated and possibly asynchronous environment.

Provide a different "output file name" for each executed command. This separates output in a system that could be running multiple commands at the same time.

Check for the existence of a file before attempting to deserialize the data contained within it.

Check any instance or member that is nullable if it's null or empty, before you try to access any data contained.

Workflow:

When beginning to build and debug an integration with the API, it's best to view the TXT log file (outputFileName_log.txt at outputFilePath).

After executing a command from your own software with the API, look for the XML log file (outputFileName_log.xml at outputFilePath). Deserialize the XML in this file to an OutputCollection. The OutputCollection contains a list of Outputs. Find the Output in the list with the same OutputFileName that you passed in the command (Output->Input->OutputFileName).

If the Output->ErrorEncountered's value is true, then there was an issue with the command.

If the Output->ErrorEncountered is false, then the command executed successfully and returned data. Deserialize the XML in the outputFileName.xml to the correct data type for the data requested. The first XML element under each of the “test.xml” examples is the data type generated by the schemas that should be used to deserialize the XML (i.e. UserCollection, DataCategoryCollection, PatientName, etc...).
The outputFileName.xml file path will be listed under Output->OutputFileList[0].

Get list of users with IDs

"CPM_API_Client.exe -command getUsers -outputPath <output text file path> -outputFileName <output file name>"

Parameters:

-outputPath: required, string, the path where all output files will be placed
-outputFileName: required, string, the name that the output files will start with

Example:

"-command getUsers -outputPath C:/Users/Admin/Desktop -outputFileName test"

Output:

Description: Collection of users with IDs and names (XML file)

Example:

test_log.txt

```
Progress: Starting CPM API
Progress: Found CPM API (C:/Program Files (x86)/CrystalPM/CPM_API.dll)
Progress: Initialized CPM API
Progress: running command
DataOutput: successfully created Users XML file: C:/Users/Admin/Desktop/test.xml
```

test_log.xml

```
<OutputCollection>
  <OutputList>
    <Output>
      <Id>c316d08e-2b0c-4101-9333-0094ccc55c26</Id>
      <TimeStamp>2017-02-10T14:37:56.4041426-06:00</TimeStamp>
      <LogList>
        <string>DataOutput: successfully created Users XML file: C:/Users/Admin/Desktop/test.xml</string>
      </LogList>
      <OutputFileList />
      <ErrorEncountered>false</ErrorEncountered>
      <ErrorList />
    </Output>
  </OutputList>
</OutputCollection>
```

test.xml

```
<UserCollection>
  <Users>
    <User>
      <UserId>1</UserId>
      <Name>
        <FirstName>John</FirstName>
        <LastName>Doe</LastName>
      </Name>
    </User>
              <User>
      <UserId>2</UserId>
      <Name>
        <FirstName>Jane</FirstName>
        <LastName>Doe</LastName>
      </Name>
    </User>
</UserCollection>
```

Get Data Categories for retrieving specific data for patient

"CPM_API_Client.exe -getDataCategories -outputPath <output file path> -outputFileName <output file name>"

Parameters:

-outputPath: required, string, the path where all output files will be placed
-outputFileName: required, string, the name that the output files will start with

Example:

"-command getDataCategories -outputPath C:/Users/Admin/Desktop -outputFileName test"

Output:

Description: Collection of Data Categories with IDs and descriptions (XML File)

Example:

test_log.txt

```
Progress: Starting CPM API
Progress: Found CPM API (C:/Program Files (x86)/CrystalPM/CPM_API.dll)
Progress: Initialized CPM API
Progress: running command
DataOutput: successfully created Data Category XML file: C:/Users/Admin/Desktop/test.xml
```

test_log.xml

```
<OutputCollection>
  <OutputList>
    <Output>
      <Id>83fb5ef6-b98c-49fa-8f07-2fbf474bfcf3</Id>
      <TimeStamp>2017-02-10T14:37:57.0017177-06:00</TimeStamp>
      <LogList>
        <string>DataOutput: successfully created Data Category file:C:/Users/Admin/Desktop/test.xml</string>
      </LogList>
      <OutputFileList>
        <OutputFile>
          <FilePath>C:/Users/Admin/Desktop/test.xml</FilePath>
        </OutputFile>
      </OutputFileList>
      <ErrorEncountered>false</ErrorEncountered>
      <ErrorList />
    </Output>
  </OutputList>
</OutputCollection>
```

test.xml

```
<DataCategoryCollection>
  <DataCategories>
    <DataCategory>
      <Id>1</Id>
      <Description>Patient Name</Description>
    </DataCategory>
    <DataCategory>
      <Id>2</Id>
      <Description>Sex</Description>
    </DataCategory>
    <DataCategory>
      <Id>3</Id>
      <Description>Date of Birth</Description>
    </DataCategory>
     (not all data categories are included in this example (4-20))
</DataCategoryCollection>
```

Get Patient Token

“CPM_API_Client.exe -command getPatientToken -userId <user ID> -userPassword <user password> -patientFirstName <patient first name> -patientLastName <patient last name> -patientMiddleName <patient middle name> -patientSex <patient sex> -patientDateOfBirth <patient date of birth> -outputPath <output file path> -outputFileName <output file name>”

Parameters:

- -userId: required, positive integer, representing the retrieving user
- -userPassword: required (if user has password), string, user's password
- -patientFirstName: required, string, first name of queried patient
- -patientLastName: required, string, last name of queried patient
- -patientMiddleName: string, middle name of queried patient
- -patientSex: string, required, sex of patient, desired values are “M” for male, “F” for female, and “U” for unknown
- -patientDateOfBirth: required, date (MM/dd/yyyy), patient date of birth
- -outputPath: required, string, the path where all output files will be placed
- -outputFileName: required, string, the name that the output files will start with

Example:

“-command getPatientId -userId 3 -userPassword pa$$w0rd -patientFirstName John -patientLastName Doe -patientMiddleName Lee -patientSex M -patientDateOfBirth 03/15/1985 -outputPath C:/Users/Admin/Desktop -outputFileName test”

Output:

Description: Patient Security Token with expiration date (XML File)

Notes: The token to query patient data is valid for 24 hours.

Example:

test_log.txt
```
Progress: Starting CPM API
Progress: Found CPM API (C:/Program Files (x86)/CrystalPM/CPM_API.dll)
Progress: Initialized CPM API
Progress: running command
DataOutput: successfully created Patient ID XML file:  C:/Users/Admin/Desktop/test.xml
```

test_log.xml
```
<OutputCollection>
  <OutputList>
    <Output>
      <Id>4f8268e3-36ab-451f-8c6e-aa6b0e03a435</Id>
      <TimeStamp>2017-02-10T14:37:57.1002347-06:00</TimeStamp>
      <LogList>
        <string>DataOutput: successfully created Patient ID XML file: C:/Users/Admin/Desktop/test.xml</string>
      </LogList>
      <OutputFileList>
        <OutputFile>
          <FilePath>C:/Users/Admin/Desktop/test.xml</FilePath>
        </OutputFile>
      </OutputFileList>
      <ErrorEncountered>false</ErrorEncountered>
      <ErrorList />
    </Output>
  </OutputList>
</OutputCollection>
```

test.xml
```
<PatientToken>
  <Token>3d0af735e4ff4c859517c6bad59c3b5f7662bab6489947589d7f4a50400561cd</Id>
  <ExpirationDate>2017-04-12T15:23:20.3516021-05:00</ExpirationDate>
</PatientToken>
```

Get Patient Data for a specific Data Category

"CPM_API_Client.exe -command getPatientDataForCategory -patientToken <patient token> -dataCategoryId <data category ID> -startDate <retrieval start date> -endDate <retrieval end date> -outputPath <output file path> -outputFileName <output file name>"

Parameters:

-patientToken: required, string (64 characters), patient security token
-dataCategoryId: required, positive integer (1-20), data category ID
-startDate: date (MM/dd/yyyy), starting date of data to be retrieved
-endDate: date (MM/dd/yyyy), ending date of data to be retrieved
-outputPath: required, string, the path where all output files will be placed
-outputFileName: required, string, the name that the output files will start with

Notes:

To retrieve data for a single date, enter the same date for the startDate and endDate parameters.

Example:

"-command getPatientDataForCategory -patientToken <patient token> -dataCategoryId 5 -startDate 01/01/2015 -endDate 12/31/2015 -outputPath C:/Users/Admin/Desktop -outputFileName test"

Output:

Description: Patient Data Category (XML File)
Notes: The start and end dates pull data based on the date of entry.
Example:

test_log.txt

```
Progress: Starting CPM API
Progress: Found CPM API (C:/Program Files (x86)/CrystalPM/CPM_API.dll)
Progress: Initialized CPM API
Progress: running command
DataOutput: successfully created Patient Data Category XML file: C:/Users/Admin/Desktop/test.xml
```

test_log.xml

```
<OutputCollection>
  <OutputList>
    <Output>
      <Id>83fb5ef6-b98c-49fa-8f07-2fbf474bfcf3</Id>
      <TimeStamp>2017-02-10T14:37:57.0017177-06:00</TimeStamp>
      <LogList>
        <string>DataOutput: successfully created Patient Data Category XML file: C:/Users/Admin/Desktop/test.xml</string>
      </LogList>
      <OutputFileList>
        <OutputFile>
          <FilePath>C:/Users/Admin/Desktop/test.xml</FilePath>
        </OutputFile>
      </OutputFileList>
      <ErrorEncountered>false</ErrorEncountered>
      <ErrorList />
    </Output>
  </OutputList>
</OutputCollection>
```

Data Categories:

1. Patient Name:

Example:

test.xml
```
<PatientName>
  <Name>
    <FirstName>John</FirstName>
    <LastName>Doe</LastName>
    <MiddleName>Lee</MiddleName>
  </Name>
</PatientName>
```

2. Sex

Possible values:

Sex->identifier->Code: 'M' for male, 'F' for female, or 'UNK' for unknown

Example:

test.xml
```
<PatientSex>
  <Sex>
    <Identifier>
      <Code>M</Code>
    </Identifier>
  </Sex>
</PatientSex>
```

3. Date of Birth:

Example:

test.xml
```
<PatientDateOfBirth>
  <DateOfBirth>
    <Date>1976-10-15T00:00:00</Date>
  </DateOfBirth>
</PatientDateOfBirth>
```

4. Race

Example:

test.xml
```
<PatientRaceCollection>
  <Races>
    <PatientRace>
      <Race>
        <DateSet>true</DateSet>
        <Date>2017-05-22T19:02:50</Date>
        <Identifer>
          <Code>1002-5</Code>
          <Name>American Indian or Alaska Native</Name>
          <CodingSystem>2.16.840.1.113883.6.238</CodingSystem>
        </Identifer>
      </Race>
    </PatientRace>
    <PatientRace>
      <Race>
        <DateSet>true</DateSet>
        <Date>2017-05-22T19:02:50</Date>
        <Identifer>
          <Code>1579-2</Code>
          <Name>Absentee Shawnee</Name>
          <CodingSystem>2.16.840.1.113883.5.104</CodingSystem>
        </Identifer>
      </Race>
    </PatientRace>
  </Races>
</PatientRaceCollection>
```

5. Ethnicity

Example:

test.xml
```
<PatientEthnicityCollection>
  <Ethnicities>
    <PatientEthnicity>
      <Ethnicity>
        <DateSet>true</DateSet>
        <Date>2017-05-22T19:02:50</Date>
        <Identifier>
          <Code>2135-2</Code>
          <Name>Hispanic or Latino</Name>
          <CodingSystem>2.16.840.1.113883.6.238</CodingSystem>
        </Identifier>
      </Ethnicity>
    </PatientEthnicity>
    <PatientEthnicity>
      <Ethnicity>
        <DateSet>true</DateSet>
        <Date>2017-05-22T19:02:50</Date>
        <Identifier>
          <Code>2138-6</Code>
          <Name>Andalusian</Name>
          <CodingSystem>2.16.840.1.113883.6.238</CodingSystem>
        </Identifier>
      </Ethnicity>
    </PatientEthnicity>
  </Ethnicities>
</PatientEthnicityCollection>
```

6. Preferred Language

Example:

test.xml
```
<PatientPreferredLanguage>
  <PreferredLanguage>
    <DateSet>true</DateSet>
    <Date>2017-02-08T15:33:13</Date>
    <Identifier>
      <Code>eng</Code>
      <Name>English</Name>
      <CodingSystem>ISO 639-2</CodingSystem>
    </Identifier>
  </PreferredLanguage>
</PatientPreferredLanguage>
```

7. Smoking Status

Example:

test.xml
```
<SmokingStatusCollection>
  <SmokingStatuses>
    <SmokingStatus>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>428061000124105</Code>
        <Name>Light smoker (&lt;10 cigs/day)</Name>
        <CodingSystem>2.16.840.1.113883.6.96</CodingSystem>
      </Identifier>
    </SmokingStatus>
  </SmokingStatuses>
</SmokingStatusCollection>
```

8. Problems

Example:

test.xml
```
<ProblemCollection>
  <Problems>
    <Problem>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>A18.52</Code>
        <Name>Tuberculous keratitis</Name>
        <CodingSystem>2.16.840.1.113883.6.3</CodingSystem>
      </Identifier>
      <StartDate>2017-02-08T15:33:13</StartDate>
      <StartDateSet>true</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </Problem>
    <Problem>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>271860004</Code>
        <Name>Abdominal mass</Name>
        <CodingSystem>2.16.840.1.113883.6.96</CodingSystem>
      </Identifier>
      <StartDate>2017-02-08T15:33:13</StartDate>
      <StartDateSet>true</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </Problem>
  </Problems>
</ProblemCollection>
```

9. Medications

Example:

test.xml

```
<MedicationCollection>
  <Medications>
    <BasicOrder>
      <Lab>0</Lab>
      <ProviderEmpID>0</ProviderEmpID>
      <Type>Medication</Type>
      <Status>55561003</Status>
      <MedicationOrder>
        <Refills>0</Refills>
        <FillAmount>
          <Value>0</Value>
        </FillAmount>
        <Drug>
          <BrandName>Tylenol</BrandName>
          <RxCode>209387</RxCode>
          <Ingredients>
            <Ingredient>
              <Name>Acetaminophen</Name>
              <RxCode>161</RxCode>
            </Ingredient>
          </Ingredients>
          <Form>TABLET</Form>
          <Route>B</Route>
          <Generic>Acetaminophen</Generic>
          <Strength>
            <Value>325</Value>
            <Units>MG</Units>
          </Strength>
          <FreeText>Acetaminophen 325 MG Oral Tablet </FreeText>
          <NotAClass>true</NotAClass>
        </Drug>
        <ID />
        <Type />
        <Site />
        <Form />
        <Directions>
          <Rate>
            <Occurances>0</Occurances>
          </Rate>
          <Route2 />
          <Duration>
            <Occurances>0</Occurances>
          </Duration>
          <PeriodicAmount>
            <Value>0</Value>
          </PeriodicAmount>
        </Directions>
        <AsWritten>false</AsWritten>
      </MedicationOrder>
      <Date>2017-02-08T00:00:00-06:00</Date>
      <EndDate>2017-02-08T15:36:15.1550588-06:00</EndDate>
      <LastModifiedDate>2017-02-08T15:37:23.5460499-06:00</LastModifiedDate>
      <Notes />
      <ObsRequests />
    </BasicOrder>
  </Medications>
</MedicationCollection>
```

10. Medication Allergies

Example:

test.xml

```xml
<MedicationAllergyCollection>
  <DrugAllergies>
    <DrugAllergy>
      <Drug>
        <RxCode>10318</RxCode>
        <Ingredients>
          <Ingredient>
            <Name>Tacrine</Name>
            <RxCode>10318</RxCode>
          </Ingredient>
        </Ingredients>
        <Generic>Tacrine</Generic>
        <NotAClass>false</NotAClass>
      </Drug>
      <Reaction />
      <Status>55561003</Status>
      <ID />
      <Severity />
      <StartDateNotSet>false</StartDateNotSet>
      <StartDate>2017-02-08T00:00:00-06:00</StartDate>
      <EndDateNotSet>true</EndDateNotSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <LastModifiedDate>2017-02-08T15:38:27.5733168-06:00</LastModifiedDate>
    </DrugAllergy>
  </DrugAllergies>
</MedicationAllergyCollection>
```

11. Laboratory Tests

Example:

test.xml

```
<LabTestCollection>
  <LabTests>
    <BasicOrder>
      <Lab>0</Lab>
      <ProviderEmpID>1</ProviderEmpID>
      <Type>Laboratory</Type>
      <Status>CM</Status>
      <OrderID>248</OrderID>
      <FillerOrderNumber>123</FillerOrderNumber>
      <Name>Indirect antiglobulin test.complement specific reagent [Presence] in Serum or Plasma</Name>
      <LabOrder />
      <Date>2017-02-08T00:00:00-06:00</Date>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <LastModifiedDate>2017-02-08T15:39:29.1700984-06:00</LastModifiedDate>
      <Code>1003-3</Code>
      <Notes />
      <ObsRequests>
        <ObservationRequest>
          <OrderNumber xmlns="http://tempuri.org/LabResults.xsd">
            <EntityID />
          </OrderNumber>
          <FillerOrderNumber xmlns="http://tempuri.org/LabResults.xsd">
            <EntityID>123</EntityID>
          </FillerOrderNumber>
          <ServiceID xmlns="http://tempuri.org/LabResults.xsd">
            <Code>1003-3</Code>
            <Name>Indirect antiglobulin test.complement specific reagent [Presence] in Serum or Plasma</Name>
          </ServiceID>
          <reasonsForStudy/>
          <Results/>
          <Notes/>
          <Diags/>
          <Collecters/>
          <Technicians/>
        </ObservationRequest>
      </ObsRequests>
    </BasicOrder>
  </LabTests>
</LabTestCollection>
```

12. Laboratory Results
Format:

test.xml

```
<LabResultCollection>
  <LabResults>
    <LabResult>
      <OrderRequest>
        <OrderControl>
          <complete>false</complete>
          <value>RE</value>
        </OrderControl>
        <Provider>
          <Provider>
            <Name>
              <FamilyName>HL</FamilyName>
              <GivenName>Seven</GivenName>
              <Prefix>Dr.</Prefix>
            </Name>
            <Address>
              <Street>8943 Database Row</Street>
              <City>Excessdee</City>
              <State>NV</State>
              <Zip>89086</Zip>
              <Type>M</Type>
            </Address>
            <PhoneNumber>
              <AreaCode />
              <Number />
            </PhoneNumber>
            <Notes />
          </Provider>
          <AssigningAuthority />
          <Empid>0</Empid>
        </Provider>
        <Facility>
          <Name>Ex Path Clinic</Name>
          <TypeCode>D</TypeCode>
          <AssigningAuthority />
          <Address>
            <Street>21 North Ave.</Street>
            <City>Burlington</City>
            <State>MA</State>
            <Zip>02368</Zip>
            <Type>M</Type>
          </Address>
          <Phone>
            <AreaCode>555</AreaCode>
            <Number>5551003</Number>
          </Phone>
        </Facility>
        <PlacerOrderNumber>248</PlacerOrderNumber>
        <FillerOrderNumber>123</FillerOrderNumber>
        <Status />
        <Date>2017-02-08T15:40:22.0968211-06:00</Date>
        <Notes />
      </OrderRequest>
      <ObservationRequests>
        <ObservationRequest>
          <OrderNumber>
            <EntityID />
          </OrderNumber>
          <Status />
          <reasonsForStudy />
          <Results>
            <Result>
              <ParentType>None</ParentType>
              <AssignedCodeType>None</AssignedCodeType>
              <IdentifierSelectionType xsi:nil="true" />
              <ObsID>
                <Name />
              </ObsID>
              <Notes />
              <AbnormalFlags />
              <DataLastObsNormalValuesSpecified>false</DataLastObsNormalValuesSpecified>
            </Result>
          </Results>
          <Notes />
          <SpecimenActionCode />
          <Specimen>
            <RejectReasons />
            <Conditions />
          </Specimen>
          <Diags />
          <Collecters />
          <Technicians />
        </ObservationRequest>
      </ObservationRequests>
      <Lab>0</Lab>
    </LabResult>
  </LabResults>
</LabResultCollection>
```

13. Vital Signs

Example:

test.xml

```
<VitalSignCollection>
  <VitalSigns>
    <VitalSign>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>8302-2</Code>
        <Name>Height</Name>
        <CodingSystem>2.16.840.1.113883.6.1</CodingSystem>
      </Identifier>
      <NumericValue>6</NumericValue>
      <UnitType>
        <Identifier>
          <Code>[ft_i]</Code>
          <Name>Feet [English Length Units]</Name>
        </Identifier>
      </UnitType>
      <StartDate>2017-02-08T15:33:13</StartDate>
      <StartDateSet>true</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </VitalSign>
    <VitalSign>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>3141-9</Code>
        <Name>Weight Measured</Name>
        <CodingSystem>2.16.840.1.113883.6.1</CodingSystem>
      </Identifier>
      <NumericValue>200</NumericValue>
      <UnitType>
        <Identifier>
          <Code>[lb_av]</Code>
          <Name>Pound [English Mass Units]</Name>
        </Identifier>
      </UnitType>
      <StartDate>2017-02-08T15:33:13</StartDate>
      <StartDateSet>true</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </VitalSign>
  </VitalSigns>
</VitalSignCollection>
```

14. Procedures

Example:

test.xml

```
<ProcedureCollection>
  <Procedures>
    <Procedure>
      <DateSet>true</DateSet>
      <Date>2017-02-08T16:29:23</Date>
      <Identifier>
        <Code>108267006</Code>
        <Name>Immunologic procedure</Name>
        <CodingSystem>SNOMEDCT_US</CodingSystem>
      </Identifier>
      <StartDate>0001-01-01T00:00:00</StartDate>
      <StartDateSet>false</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </Procedure>
    <Procedure>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Identifier>
        <Code>2019F</Code>
        <Name>AMD - dilated macular exam</Name>
        <CodingSystem>2.16.840.1.113883.6.12</CodingSystem>
      </Identifier>
      <StartDate>2017-02-08T15:33:13</StartDate>
      <StartDateSet>true</StartDateSet>
      <EndDate>0001-01-01T00:00:00</EndDate>
      <EndDateSet>false</EndDateSet>
    </Procedure>
  </Procedures>
</ProcedureCollection>
```

15. Care Team Members

Example:

test.xml

```
<CareTeamMemberCollection>
  <CareTeamMembers>
    <CareTeamMember>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <PersonInfo>
        <Name>
          <FirstName>Jack</FirstName>
          <LastName>Thompson</LastName>
          <MiddleName />
          <Title>Dr</Title>
        </Name>
      </PersonInfo>
      <IsDataEnterer>false</IsDataEnterer>
      <ParticipationFunction>
        <Code>PCP</Code>
        <Name>primary care physician</Name>
        <CodingSystem>2.16.840.1.113883.5.88</CodingSystem>
      </ParticipationFunction>
      <Taxonomy>
        <Code>363LA2100X</Code>
        <Name>Acute Care</Name>
        <CodingSystem>2.16.840.1.113883.6.101</CodingSystem>
      </Taxonomy>
    </CareTeamMember>
    <CareTeamMember>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <PersonInfo>
        <Name>
          <FirstName>Frank</FirstName>
          <LastName>Miller</LastName>
          <MiddleName />
          <Title />
        </Name>
      </PersonInfo>
      <IsDataEnterer>true</IsDataEnterer>
      <ParticipationFunction>
        <Code>FASST</Code>
        <Name>first assistant</Name>
        <CodingSystem>2.16.840.1.113883.5.88</CodingSystem>
      </ParticipationFunction>
      <Taxonomy>
        <Code>363LA2100X</Code>
        <Name>Acute Care</Name>
        <CodingSystem>2.16.840.1.113883.6.101</CodingSystem>
      </Taxonomy>
    </CareTeamMember>
  </CareTeamMembers>
</CareTeamMemberCollection>
```

16. Immunizations

Example:

test.xml

```
<ImmunizationCollection>
  <Immunizations>
    <Immunization>
      <Existing>true</Existing>
      <Fid>252</Fid>
      <ContainsErrors>false</ContainsErrors>
      <ErrorMessage />
      <ErrorCount>0</ErrorCount>
      <PlaceOrderNumber>
        <complete>false</complete>
        <Code>123</Code>
      </PlaceOrderNumber>
      <FillerOrderNumber>
        <complete>false</complete>
        <Code>456</Code>
      </FillerOrderNumber>
      <StartTime xsi:nil="true" />
      <EndTime xsi:nil="true" />
      <ID>
        <complete>false</complete>
        <Code>54</Code>
        <Name>adenovirus, type 4</Name>
        <CodingSystem>2.16.840.1.113883.12.292</CodingSystem>
      </ID>
      <Amount>1</Amount>
      <Units>
        <complete>false</complete>
        <Code>mg</Code>
        <Name>MilliGram [SI Mass Units]</Name>
        <CodingSystem>UCUM</CodingSystem>
      </Units>
      <SubstanceExpirationDate xsi:nil="true" />
      <RefusalCode>
        <complete>false</complete>
      </RefusalCode>
      <Refusal2Code>
        <complete>false</complete>
      </Refusal2Code>
      <ActionCode>A</ActionCode>
      <RelatedObs />
      <NextOfKin />
      <Index>0</Index>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:45:10</Date>
    </Immunization>
  </Immunizations>
</ImmunizationCollection>
```

17. Implantable Devices

Example:

test.xml

```
<ImplantableDeviceCollection>
  <ImplantableDevices>
    <MedRecImplantableDevice>
      <implantableDevice>
        <udi>(01)00850252006004(11)170208(17)170208(10)123(21)456</udi>
        <deviceIdIssuingAgency>GS1</deviceIdIssuingAgency>
        <isBloodBag>false</isBloodBag>
        <deviceIdentifier>00850252006004</deviceIdentifier>
        <serialNumber>456</serialNumber>
        <lotNumber>123</lotNumber>
        <expirationDate>2017-02-08T00:00:00</expirationDate>
        <expirationDateDaySpecified>true</expirationDateDaySpecified>
        <expirationDateHourSpecified>false</expirationDateHourSpecified>
        <manufacturingDate>2017-02-08T00:00:00</manufacturingDate>
        <lookupResult>
          <deviceRecordStatus>Published</deviceRecordStatus>
          <devicePublishDate>2015-01-23</devicePublishDate>
          <deviceCommDistributionEndDate/>
          <deviceCommDistributionStatus>In Commercial Distribution</deviceCommDistributionStatus>
          <brandName>Avery Breathing Pacemaker System</brandName>
          <versionModelNumber>Neurostimulator Transmitter </versionModelNumber>
          <catalogNumber>MARK IV</catalogNumber>
          <companyName>Avery Biomedical Devices Inc</companyName>
          <deviceCount>1</deviceCount>
          <deviceDescription/>
          <DMExempt>false</DMExempt>
          <premarketExempt>false</premarketExempt>
          <deviceHCTP>false</deviceHCTP>
          <deviceKit>false</deviceKit>
          <deviceCombinationProduct>false</deviceCombinationProduct>
          <singleUse>true</singleUse>
          <lotBatch>false</lotBatch>
          <serialNumber>true</serialNumber>
          <manufacturingDate>true</manufacturingDate>
          <expirationDate>false</expirationDate>
          <donationIdNumber>false</donationIdNumber>
          <labeledContainsNRL>false</labeledContainsNRL>
          <labeledNoNRL>false</labeledNoNRL>
          <MRISafetyStatus>Labeling does not contain MRI Safety Information</MRISafetyStatus>
          <rx>true</rx>
          <otc>false</otc>
          <deviceSizes/>
          <identifiers>
            <identifier>
              <deviceId>00850252006004</deviceId>
              <deviceIdType>Primary</deviceIdType>
              <deviceIdIssuingAgency>GS1</deviceIdIssuingAgency>
              <containsDINumber />
              <pkgQuantity />
              <pkgDiscontinueDate />
              <pkgStatus />
            </identifier>
          </identifiers>
          <contacts>
            <customerContact>
              <phone>631-864-1600</phone>
              <phoneExtension />
              <email>info@averybiomedical.com</email>
            </customerContact>
          </contacts>
          <gmdnTerms>
            <gmdn>
              <gmdnPTName>Extramuscular diaphragm/phrenic nerve electrical stimulation system</gmdnPTName>
              <gmdnPTDefinition>An assembly of battery-powered devices designed to provide ventilatory support to a patient with diaphragm dysfunction of
neuromuscular origin through electrical stimulation of the phrenic nerve, to contract the diaphragm rhythmically and cause the patient to draw breath in a
manner similar to natural breathing. It typically consists of an implanted receiver with electrodes that are placed around the patient's phrenic nerve, and an
external transmitter for transmitting the stimulating pulses across the patient's skin to the implanted receiver.</gmdnPTDefinition>
            </gmdn>
        </gmdnTerms>
        <productCodes>
          <fdaProductCode>
            <productCode>GZE</productCode>
            <productCodeName>Implanted Diaphragmatic/Phrenic Nerve Stimulator</productCodeName>
          </fdaProductCode>
        </productCodes>
        <environmentalConditions/>
        <sterilization>
          <deviceSterile>false</deviceSterile>
          <sterilizationPriorToUse>false</sterilizationPriorToUse>
          <methodTypes />
        </sterilization>
      </lookupResult>
      <parsedResult>
        <udi>(01)00850252006004(11)170208(17)170208(10)123(21)456</udi>
        <issuing-agency>GS1</issuing-agency>
        <di>00850252006004</di>
        <serial-number>456</serial-number>
        <expiration-date-original-format>YYMMDD</expiration-date-original-format>
        <expiration-date-original>170208</expiration-date-original>
        <manufacturing-date-original-format>YYMMDD</manufacturing-date-original-format>
        <manufacturing-date-original>170208</manufacturing-date-original>
        <lot-number>123</lot-number>
        <expiration-date type="date">2017-02-08</expiration-date>
        <manufacturing-date type="date">2017-02-08</manufacturing-date>
      </parsedResult>
    </implantableDevice>
    <statusCode>0</statusCode>
    <startDateTime>2017-02-08T16:25:39</startDateTime>
    <endDateTime xsi:nil="true" />
    <DateSet>true</DateSet>
    <Date>2017-02-08T16:27:21</Date>
  </MedRecImplantableDevice>
 </ImplantableDevices>
</ImplantableDeviceCollection>
```

18. Assessment and Plan of Treatment

Example:

test.xml

```
<AssessmentAndPlanOfTreatment>
  <Assessments>
    <Assessment>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Narrative>test</Narrative>
    </Assessment>
  </Assessments>
  <Reasons>
    <TreatmentPlanReason>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Code>
        <Code>28764-9</Code>
        <Name>Heterophoria:Prid:Pt:Eyes:Nom:Worth test</Name>
        <CodingSystem>LNC</CodingSystem>
      </Code>
      <Narrative>test</Narrative>
      <EffectiveStartDateSet>true</EffectiveStartDateSet>
      <EffectiveStartDate>2017-02-08T16:22:30.2411834-06:00</EffectiveStartDate>
      <EffectiveEndDateSet>false</EffectiveEndDateSet>
      <EffectiveEndDate>0001-01-01T00:00:00</EffectiveEndDate>
    </TreatmentPlanReason>
  </Reasons>
  <Referrals />
  <LabTests />
  <ImagingTests />
  <Procedures>
    <TreatmentPlanProcedure>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Procedure>
        <Code>28764-9</Code>
        <Name>Heterophoria:Prid:Pt:Eyes:Nom:Worth test</Name>
        <CodingSystem>LNC</CodingSystem>
      </Procedure>
      <Narrative>test</Narrative>
      <EffectiveDateSet>true</EffectiveDateSet>
      <EffectiveDate>2017-02-08T00:00:00-06:00</EffectiveDate>
    </TreatmentPlanProcedure>
  </Procedures>
  <Medications />
</AssessmentAndPlanOfTreatment>
```

19. Goals

Example:

test.xml

```
<GoalCollection>
  <Goals>
    <Goal>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Narrative>test</Narrative>
    </Goal>
  </Goals>
</GoalCollection>
```

20. Health Concerns

Example:

test.xml

```
<HealthConcernCollection>
  <HealthConcerns>
    <HealthConcern>
      <DateSet>true</DateSet>
      <Date>2017-02-08T15:33:13</Date>
      <Narrative>test</Narrative>
      <Status>
        <Code>55561003</Code>
        <Name>Active</Name>
        <CodingSystem>2.16.840.1.113883.6.96</CodingSystem>
      </Status>
    </HealthConcern>
  </HealthConcerns>
</HealthConcernCollection>
```

Get all Data (CCD) for a Patient

"CPM_API_Client.exe -command getAllDataForPatient -patientToken <patient token> -startDate <retrieval start date> -endDate <retrieval end date> -outputPath <output path> -outputFileName <output file name>"

Parameters:

-patientToken: required, string (64 characters), patient security token
-startDate: date (MM/dd/yyyy), starting date of data to be retrieved
-endDate: date (MM/dd/yyyy), ending date of data to be retrieved
-outputPath: required, string, the path where all output files will be placed
-outputFileName: required, string, the name that the output files will start with

Notes:

To retrieve data for a single date, enter the same date for the startDate and endDate parameters

Example:

```
"-command getAllDataForPatient -patientToken <patient token>
-startDate 01/01/2015 -endDate 12/31/2015
-outputPath C:/Users/Admin/Desktop -outputFileName test"
```

Output:

Description: Patient CCD Container (XML File)
Notes: The start and end dates pull data based on the date of entry.
Example:

test_log.txt

```
Progress: Starting CPM API
Progress: Found CPM API (C:/Program Files (x86)/CrystalPM/CPM_API.dll)
Progress: Initialized CPM API
Progress: running command
DataOutput: successfully created CCD XML file:  C:/Users/Admin/Desktop/test.xml
```

test_log.xml

```
<OutputCollection>
  <OutputList>
    <Output>
      <Id>4f8268e3-36ab-451f-8c6e-aa6b0e03a435</Id>
      <TimeStamp>2017-02-10T14:37:57.1002347-06:00</TimeStamp>
      <LogList>
        <string>DataOutput: successfully created CCD XML file:  C:/Users/Admin/Desktop/test.xml</string>
      </LogList>
      <OutputFileList>
        <OutputFile>
          <FilePath>C:/Users/Admin/Desktop/test.xml</FilePath>
        </OutputFile>
      </OutputFileList>
      <ErrorEncountered>false</ErrorEncountered>
      <ErrorList />
    </Output>
  </OutputList>
</OutputCollection>
```

test.xml

```
<CcdContainer>
  <CcdDocument moodCode="EVN">
          ...
  </CcdDocument>
</CcdContainer>
```

Terms and Conditions:

Last Updated: 05/17/2017

Please read the Terms and Conditions (“Terms”, “Terms and Conditions”) carefully before using the API.

Your access to use the API is conditioned on your acceptance of and compliance with these terms. These terms apply to all customers, users, and third-party integrators who use CrystalPM or its API.

By accessing or using the API, you agree to be bound by these Terms. If you disagree with any part of the terms, then you may not access CrystalPM or its API.

Terms and Conditions

http://crystalpm.com/business-associate-agreement/

http://crystalpm.com/notice-of-privacy-practices/

Contact

If you have any questions, please contact our support.

http://crystalpm.com/support-request/