CARMIN

a Common web Api for ReMote pipeline INvocation
Version 0.3

Notation definitions

General

new in release 0.3

This should be discussed, not included in current specification

Comments, with only a descriptive role

Modules

Unique section used only in the common elements parts to define all the modules and their respective objects.

- **Module1** | Object1, Object2 |
- **Module2** | Object3 |

Types Definitions *(Types)*

Section that contains one or several type definitions. It can have any name that only has an informative role.

- **Type1**
  A root type. It must include a description in a comment
- **Type2:** enum {Value1, Value2}
  An enumerated type that can have several string values
- **<UnionType> :** type-enum {Type1, type2}
  Declares an union of several types that must exist. The type "UnionType" is then considered an enumerated type that can take the listed string values. More importantly, a value of the type "<UnionType>" can have any value of type "Type1" or "Type2". Any number of types is allowed.

Unnamed types uses

This does not correspond to a definition section but describe several ways to use advanced unnamed types based on existing ones.

Type1[]

Given an existing "Type1" type, this represents a list of elements that all are of ype "Type1"

{Type1 field1, Type2 field2}

Given existing "Type1" and "Type2" types, this represents an object with 2 fields of types "Type1" and "Type2" named "field1" and "field2". An object can have any number of fields.

Object1 *(Object)*

A section used to define an object named "Object1". This also declares a type "Object1". This definitions can contain the fields of the object and its related methods.

- Int64 **field1**
  A mandatory field named "field1" of type "Int64". Any type can be used.
- (Ascii **field2**)
  An optional field named "field2" of type "Ascii".
- Int64 **field3** (update)
  A mandatory field that can be updated (the field can also be optional)
- (Boolean **field4** | Module2)

A field which must be present only if the module named "Module2" is supported. In this case the field is optional

- `Ascii` **`doStuff1`**`(Int64 param1, (Ascii param2), (Boolean param3 |` **`Module3`**`))`

  A method named "doStuff1" that takes several parameters and return an element of type Ascii. Any return type can be used. A method can have any number of parameters (including none). A parameter definition has the same format than a field definition to express that it is mandatory, optional or must only be present when a given module is supported.
- **`doStuff2`**`(Ascii param)`

  A method can have no return type.
- `( Boolean` **`doStuff2`**`(UTF8 param) )`

  A optional method. It must return an explicit error message when not implemented
- `( Ascii[]` **`doStuff3`**`(Ascii param1, Ascii param2) |` **`Module4`**`)`

  A method that must only be present if a specific module is supported

Additional informations

This section contains informations that cannot be specified by other elements

- `Information description`
- `Another information description`

COMMON ELEMENTS

Root types *(Types)*

- **`UTF8`**

  A string encoded in UTF-8
- **`Ascii`**

  An UTF-8 string using only ASCII characters
- **`Boolean`**
- **`Int64`**

  A signed integer (from -2^{31} to $2^{31}-1$)
- **`Double`**

  A floating-point number encodable with IEEE-754
- **`URL`**

  A UTF-8 encoded string that represents a valid URL
- **`File`**

  An UTF-8 string (format is not specified).
- **`Base64`**

  A base64-encoded binary content.

Modules

The objects described below are grouped in modules which may depend on each other only through optional parameters and attributes.

- **`Processing`** `|Pipeline, Execution |`
- **`Data`** `| Path |`
- **`AdvancedData`** `| RequestURL |`
- **`TODO: Metadata`**
- **`TODO: Management`** `|Study, User |`
- **`TODO: Commercial, PipelineUpload`**

Common types *(Types)*

- **Module:** enum {Processing, Data, AdvancedData}

PlatformProperties *(Object)*

PlatformProperties must be supported in any case. It allows an implementing platform to inform the modules it implements and various others important informations.

- UTF8 **platformName**
  Name of the platform.
- ({ Int64 errorCode, UTF8 errorMessage }[] **APIErrorCodesAndMessages**)
- Ascii[] **supportedTransferProtocols**
  Used in getExecutionResults and getAccessURL. Protocol names must be URL prefixes (e.g. "http", "https", "ftp", "sftp", "ftps", "scp", "webdav").
- Module[] **supportedModules**
- Int64 **defaultLimitListExecutions**
  The number of Executions returned by Execution.listExecution when limit is not specified.
- (UTF8 **email**)
- (UTF8 **platformDescription**)
- (Int64 **minAuthorizedExecutionTimeout**)
- (Int64 **maxAuthorizedExecutionTimeout**)
  0 or absent means no max timeout. max has to be greater or equal to the min.
- (Int64 **defaultExecutionTimeout**)
  0 or absent means no default timeout. defaultTimeout has to be between the min and max value.
- Ascii[] unsupportedMethods
  List of optional methods that are not supported by the platform.
- (Int64 **maxSizeDirectTransfer | Path**)
  The maximal size of files and directories that can be transferred in the Path module
- Ascii **supportedAPIVersion**
- (Ascii **defaultStudy | Management**)
- **more technical info about the platform (OS, RAM, etc)**

➢ PlatformProperties **getPlatformProperties**()

Authentication *(Object)*

- Ascii **httpHeader**
- Ascii **httpHeaderValue**

➢ Authentication **authenticate** (UTF8 username, UTF8 password)

If authentication is successful, the response payload contains an authentication object with httpHeaderValue and a httpHeader (possibly empty) that the client has to use in subsequent requests. If authentication uses cookies, they have to be set by the server using the standard HTTP header. The header line may also specify, for instance, an API Key to use in the header.

➢ (**logout**)

Destroys the session created by **authenticate**.

Additional informations

- All methods must include in their response a code to know if the execution is successful. In case of error, the response must contain an error code and an error message.This can be done by an envelope that wrap all methods responses and contain all these informations. The error codes can be described in the PlatformProperties.
- Although only the session authentication mechanism is described by methods, the authentication can also be ensured by an API key

mechanism. An API key must then be included in an header when a protected method is called.

Processing Module

Processing types *(Types)*

- **<ParameterType>:** type-enum { File, String, Boolean, Int64, Double,<ParameterType>[] }
  All the types allowed for execution input values. Correspond to basic types plus list of basic types. List of lists are not allowed. All the elements of a list must be of the same basic type.
- **ExecutionStatus:** enum { Initializing, Ready, Running, Finished, InitializationFailed, ExecutionFailed, Unknown, Killed}
  Finished, InitializationFailed, ExecutionFailed, Killed are terminal statuses. Initializing, Ready, Running, Unknown are active statuses.

Execution *(Object)*

- Ascii **identifier**
- UTF8 **name** (update)
- Ascii **pipelineIdentifier**
- (Int64 **timeout** (update))
  The timeout in seconds until which the execution is killed and deleted with all its files. 0 or absence means no timeout.
- ExecutionStatus **status**
- ({Ascii parameterName,<ParameterType> parameterValue}[] **inputValues**)
- ({Ascii parameterName,File returnedFile}[] **returnedFiles**)
  Absent when information is not available (e.g. execution is still running). Empty list when execution has not produced any returned file.
- (Ascii **studyIdentifier** | **Management**)
- (Int64 **errorCode**)
  Dependent on the platform and pipeline. May be specified in Pipeline.
- (Int64 **startDate**)
  The date when the pipeline entered status "Init", represented as a number of seconds since the Epoch.
- (Int64 **endDate**)
  The date when the pipeline entered a terminal state, represented as a number of seconds since the Epoch..

➢ Execution **getExecution** (Ascii executionIdentifier)
➢ Execution[] **listExecutions** ((Ascii studyIdentifier | **Management**), (Int64 offset), (Int64 limit))
Returns a list of Executions, ordered in decreasing submission time. offset and limit are used to request a sub-list of the Execution list containing elements with indexes ranging from offset to offset+limit-1. offset defaults to 0, i.e. the index of the first element in the list. limit defaults to the value in the PlatformProperties. All the executions that were launched by the user must be in the Execution list. It is up to the platform to return executions that the user did not launch. When studyIdentifier is present, all the executions that the user launched in the study must be returned.
➢ Int64 **countExecutions** ((Ascii studyIdentifier | **Management**))
Returns the number of Executions accessible by the user.
➢ **updateExecution** (Ascii executionId, (UTF8 name), (Int64 timeout))

Updates an execution

- Ascii **createExecution** (Ascii pipelineId, {Ascii parameterName, <ParameterType> parameterValue}[] inputValues, (Int64 timeoutInSeconds), (UTF8 executionName), (Ascii studyId | **Management**))

Creates an execution and returns its identifier. If status "Initializing" is returned, playExecution must be called to start the execution.

- ExecutionStatus status **playExecution** (Ascii executionId)
- **killExecution** (Ascii executionId)
- **deleteExecution** (Ascii executionId, Boolean deleteFiles)

Active executions cannot be deleted (they must be killed first). If deleteFiles is true, deletes all the files produced by the execution, including temporary files. When deleteExecution is called, the execution completely disappears from the platform, i.e. it is not returned by getExecution or listExecutions.

- URL[] **getExecutionResultsURL**(Ascii executionId, (Ascii protocol))

Return a list of urls to download the results. The default protocol is https.

- (Path[] **getExecutionResults**(Ascii executionId) | **Path**)

Returns a list of Paths for the results.

- (UTF8 **getStdout** (Ascii executionIdentifier))

Returns the standard output of the execution.

- (UTF8 **getStderr** (Ascii executionIdentifier))

Returns the standard error of the execution.

PipelineParameter *(Object)*

- Ascii **name**
- ParameterType **type**
- Boolean **isOptional**
- Boolean **isReturnedValue**

True if the parameter is a returned value. False if the parameter is an input value.

- (<ParameterType> **defaultValue**)
- (UTF8 **description**)

Pipeline *(Object)*

- Ascii **identifier**
- UTF8 **name**
- Ascii **version**
- (UTF8 **description**)
- (Boolean **canExecute**)

True if user who requested the pipeline can execute it.

- ([PipelineParameter] **parameters**)
- ({ Int64 errorCode, UTF8 errorMessage }[] **errorCodesAndMessages**)

- Pipeline[] pipelines **listPipelines** ((Ascii studyIdentifier | **Management**))

List pipelines. All the pipelines that the user can execute must be returned. It is up to the platform to return pipelines that the user cannot execute. When studyIdentifier is present, all the pipelines that the user can execute in the study must be returned. In this case, execution rights denote the rights to execute the pipeline in the study.

- Pipeline **getPipeline** (Ascii pipelineId)

Data module

Data types *(Types)*

- **TransferStatus:** enum {Transferring, Done, Failed}
- **ArchiveType:** enum {Zip, Tar}

Path *(Object)*

A Path may represent a Directory.

- UTF8 **platformURI** (update)
  A valid URI, complying to the following pattern:
  scheme:[//[user:password@]host[:port]][/]path[?query][#fragment].
  For instance,
  shablanc://shablanc.somedomain.org/home/foo/file.txt and
  sftp://foo@server1.localhost/home/foo/file.txt
  are valid platform URIs. In addition, the path in this URI must be consistent with the path of files and directories uploaded and downloaded by clients. For instance, if a user uploads a directory structure "dir/{file1.txt,file2.txt}", it is expected that the path of the first file will be "[prefix]/dir/file1.txt" and that the path of the second file will be "[prefix]/dir/file2.txt" where [prefix] depends on the upload parameters, in particular destination directory.
- Int64 **lastModificationDate**
  Date of last modification, in seconds since the Epoch (UNIX timestamp).
- Boolean **isDirectory**
  True if the path represents a directory.
- (Boolean **exists)**
  True if the path exists
- (Int64 **size**)
  For a file, size in bytes. For a directory, sum of all the sizes of the files contained in the directory (recursively).
- (Execution **producedBy | Processing**)
  The Execution that produced the Path.

➢ UTF8 **getMimeType**
For a file returns the mime type based on RFC 6838.

➢ Path **getPath** (UTF8 platformURI)
Gets the Path object (does not download the Path).

➢ {Boolean exists} **doesPathExist** (UTF8 platformURI)
Test if the path exists

➢ **deletePath** (UTF8 platformURI)
Returns an error if user is not allowed to delete Path.

➢ Path[] **listDirectory** (UTF8 platformURI)
When platformURI refers to a directory, returns a list of Paths in this directory.

➢ Path **mkdir** (UTF8 platformURI)
Creates a directory at platformURI and returns the corresponding Path object.

➢ Path **uploadPath** (UTF8 platformURI, Base64 pathContent, (ArchiveType archiveType))

Path content is passed in the HTTP request, encoded in base64. If Path is a directory, content has to be an archive of type `archiveType`*. Archive must contain the exact directory content (e.g., no absolute path). File names in archive must be encoded in UTF8. Method returns the newly created Path object. Platform may impose restrictions on the max size of files that can be uploaded with this method. For larger files, use the Advanced Data.module.*

➢ `{Path path, base64 pathContent}` **`downloadFile`** `( UTF8 platformURI )`
Returns the Path content and the Path object in the API response envelope. Path has to be a file. Platform may impose restrictions on the max size of files that can be downloaded with this method.

➢ `TransferStatusObject` **`copyPath`** `( UTF8 sourcePlatformURI, UTF8 destinationPlatformURI )`

➢ `TransferStatusObject` **`movePath`** `( UTF8 sourcePlatformURI, UTF8 destinationPlatformURI )`

➢ `( Path` **`computeDirectorySize`** `( UTF8 platformURI ) )`
Computes the size of the directory and returns the corresponding Path object. Returns an error if platformPath does not relate to a directory.

TransferStatusObject

Status of a Path transfer initiated by `copyPath` *or* `movePath`*.*

- `Ascii` **`identifier`**
- `UTF8` **`sourcePlatformURI`**
- `UTF8` **`destinationPlatformURI`**
- `TransferStatus` **`transferStatus`**

➢ `TransferStatusObject` **`getTransferStatus`** `( Ascii identifier )`

Advanced Data Module

Advanced data types *(Types)*

- **`FileURLStatus:`** `enum {NotAvailable, Transferring, Available, Finalizing}`

AccessURL *(Object)*

A URL that the user can access in read or read/write mode.

- Ascii **identifier**
- URL **url**
  The URL.
- `FileURLStatus` **`urlStatus`**
  Status of this URL.
- `(Int64` **`urlExpirationDate`**`)`
  Date until which the URL will be available.
- `Boolean` **`urlIsWritable`**
  True if user can write to the URL.

➢ `AccessURL` **`createAccessURL`** `(UTF8 platformURI, (Ascii protocol), (Boolean write), (Int64 releaseDelay) )`
Returns an AccessURL object where the Path will be accessible and sets the corresponding attribute in Path (readAccessURL or writeAccessURL). In the returned AccessURL object, sets urlStatus to

Transferring until the URL is available. This method is also used to monitor the status of the AccessURL.

- *protocol: sftp should be supported and is the default protocol for read and write URLs. sftp allows to upload/download*
- *write: if true then write access is granted on the URL and it will be copied back to the platform when released.*
- *releaseDelay: delay in seconds after which the URL may not be accessible any more.*

Once getAccessURL was called, getPath should be called until urlStatus is Available.

➢ `AccessURL getAccessURL (Ascii identifier)`

➢ **`releaseAccessURL`** (`Ascii identifier`)

Releases file URL and any associated resource. Commits modifications made on the Path. Destroys the corresponding AccessURL object.

Related APIs

	Data	Computing
CMIS https://en.wikipedia.org/wiki/Content_Management_Interoperability_Services	X	
FHIR https://www.hl7.org/fhir/	X	
CBRAIN https://github.com/aces/cbrain-apis		X
AGAVE http://agaveapi.co/live-docs	X	X
XNAT REST API https://wiki.xnat.org/display/XNAT16/XNAT+REST+AxPI+Directory	X	?
LORIS https://github.com/driusan/Loris/tree/InstrumentAPI/docs/API	X	
APACHE Airavata API		
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4362911/		