

# **SRM v2.2 Changes From SRM v2.1.2**

Compiled by Alex Sim at LBNL

Last updated : December 15, 2006

## **Abstract**

In regards to recent discussion with WLCG Data Management Coordination Group for the requirements of the LHC experiments of the Grid Storage Interfaces, we made changes to the SRM v2.1.2 specification. The changes are based on the 2-day meeting in May, 2006.

This SRM version 2.2 changes extend and update the SRM v2.1.2 specification.

WSDL Finalized : June 20, 2006

All changes after the above date will be grouped separately from page 5.

## **V2.2 Changes**

### **1. In general**

- If there is any discrepancy between SRM.v2.2.doc and SRM.v2.2.ws.op.doc, then lines in the SRM.v2.2.doc will take the precedence.
- For each function, status codes are defined with basic meanings for the function. Only those status codes are valid for the function. Specific cases are not stated for each status code. If other status codes need to be defined for a specific function, send an email to the collaboration to discuss the usage.
- Some attribute names were changed to be consistent throughout the paper. Attributes have an indication of multiple values by having a plural form.
- The word “Pinning” is limited to the “copies” or “states” of SURLs and the Transfer URLs (TURLs).

### **2. In the Defined Structure:**

- Space types (TSpaceType) are removed.
- TAccessLatency is added.
- TRetentionPolicy is added.
- Together with TAccessLatency and TRetentionPolicy, TRetentionPolicyInfo is added, and TRetentionPolicy is required when TRetentionPolicyInfo is provided.
- TRequestType is expanded.
- TGMTTime becomes TUTCTime.

- TFileLocality is added.
- TMetaDataPathDetail is changed slightly
- TMetaDataSpace is changed slightly
- TExtraInfo is added for key/value pairs of additional information.
- TStorageSystemInfo is removed, and is used as TExtraInfo[.]
- TAccessPattern is added.
- TConnectionType is added.
- TTransferParameters is added to replace the string array of arrayOfTransferProtocols parameter.
- “None” from TPermissionMode became “NONE”
- TRequestType has capitals: PREPARE\_TO\_GET, PREPARE\_TO\_PUT and COPY.
- TRequestType has additional BRING\_ONLINE, RESERVE\_SPACE, UPDATE\_SPACE, CHANGE\_SPACE\_FOR\_FILES, and LS.
- TLifeTimeInSeconds and TSizeInBytes had “unsigned long” in the spec, but “long” in the WSDL file. They are corrected to have “unsigned long” in WSDL.
- TGetFileRequest, TPutFileRequest, and TCopyFileRequest are simplified.
- SRM\_UNAUTHORIZED\_ACCESS is changed to SRM\_AUTHORIZATION\_FAILURE to cover any unauthorized failures.
- TStatusCode has a few more status codes: SRM\_PARTIAL\_SUCCESS, SRM\_REQUEST\_TIMED\_OUT, SRM\_LAST\_COPY, SRM\_FILE\_BUSY, SRM\_FILE\_LOST, and SRM\_FILE\_UNAVAILABLE.
- TUserPermission, TGroupPermission, TURLPermissionReturn, and TRequestTokenReturn have the required parameters. Those parameters are required when the type information is being provided.
- isExpired parameter is removed from TMetaDataSpace. SRM\_SPACE\_LIFETIME\_EXPIRED may be used instead.
- {Volatile, Durable, Permanent} becomes {VOLATILE, DURABLE, PERMANENT}
- {File, Directory, Link} becomes {FILEPATH, DIRECTORY, LINK}
- {Never, Always, WhenFilesAreDifferent} becomes {NEVER, ALWAYS, WHEN\_FILE\_ARE\_DIFFERENT}
- {TransferMode, ProcessingMode} becomes {TRANSFER\_MODE, PROCESSING\_MODE}
- TRequestSummary has "isSuspended" removed
- TRequestToken is removed to be string (xsd:string)
- TSpaceToken is removed to be string (xsd:string)
- TUserID is removed to be string (xsd:string)
- TGroupID is removed to be string (xsd:string)
- TOwnerPermission is removed and TPermissionMode is used.
- TOtherPermission is removed and TPermissionMode is used.
- Checksum types (TCheckSumType, TCheckSumValue) are removed and string (xsd:string) is used.
- TSizeInBytes is removed and unsigned long (xsd:unsignedLong) is used
- TUTCTime is removed and dateTime (xsd:dateTime) is used

- TLifeTimeInSeconds is removed and int (xsd:int) is used
- “infinite” lifetime is agreed.
- TSURL is removed and anyURI (xsd:anyURI) is used
- TTURL is removed and anyURI (xsd:anyURI) is used
- SURLInfo is removed. anyURI is used instead for SURLs and storage system info is moved at the request level.
- TSURLLifetimeReturnStatus is added for srmExtendFileLifeTime

### **3. In the Space Reservation Functions:**

- Retention policy is introduced as a way of indicating quality of the space where files are located.
- Access latency is also introduced to describe how latency of files is improvable.
- srmReserveSpace has new input parameters.
- srmReserveSpace is now asynchronous, and srmStatusOfReserveSpaceRequest is added for checking status of the asynchronous srmReserveSpace.
- TMetaDataSpace includes retention policy information instead of previous space types.
- srmChangeSpaceForFiles is added to change the storage token of files. Since it may take a long time to complete the request, it may be an asynchronous operation, and srmStatusOfChangeSpaceForFilesRequest is added.
- srmExtendFileLifetimeInSpace is added to extend lifetime for all files in a space that is associated with a space token.
- srmCompactSpace is removed
- srmPurgeFromSpace is added.
- srmStatusOfUpdateSpaceRequest is added for an asynchronous operation for srmUpdateSpace.
- In srmExtendFileLifetimeInSpace, new file lifetime must not exceed the remaining lifetime of the space.
- srmChangeSpaceForFiles (2.7.2, 2.7.3) has a note for target space when it cannot hold all files
- 

### **4. In the Permission Functions:**

- srmReassignToUser is removed.
- srmSetPermission (3.1.2) has a note about CHANGE permission type for non-existing user/group.
- srmCheckPermission (3.2.2) has "localCheckOnly" removed.
- srmGetPermission is added.

### **5. In the Directory Functions:**

- TMetaDataPathDetail includes the assigned retention policy and file locality.

- srmLs has TSURL and TStorageSystemInfo separately from the previously combined TSURLInfo as input parameters.
- srmLs may be an asynchronous operation, and srmStatusOfLsRequest is added.
- srmLs has limited permission returns.

## **6. In the Data Transfer Functions:**

- srmReleaseFiles has an optional remove flag to remove the “copy” or “state” from the space.
- srmRemoveFiles has been removed.
- Client access pattern is added to indicate the possible usage pattern of the TURL.
- Client connection type is added to indicate the possible connection to the TURL.
- TTransferParameters is added to combine the client input parameters for array of client supported transfer protocol list, client access pattern, client connection type and array of client networks.
- Array of client network indicates IP addresses that client has a possible access to.
- TExtraInfo is added for additional information as a key/value pair. It may be used for the returned transfer protocol of TURL. It may indicate the properties of the transfer protocol so that the client can optimize the data transfer.
- srmPrepareToGet and srmStatusOfGetRequest have updated input parameters.
- srmPrepareToPut and srmStatusOfPutRequest have updated input parameters.
- srmCopy and srmStatusOfCopyRequest have updated input parameters.
- srmBringOnline and srmStatusOfBringOnlineRequest are added.
- srmGetRequestTokens is changed from srmGetRequestID.
- srmExtendFileLifeTime has two newLifetime input parameters, one for pin lifetime for TURL, and another for file lifetime for SURL.
- srmExtendFileLifeTime has arrayOfSURLs as required.
- srmPrepareToGet/BringOnline/Put/Copy has notes changed on totalRequestTime to indicate default time and trying at least once.
- Total request time “0” (zero) indicates that SRM will try at least once for each file in the request.
- srmPrepareToGet/BringOnline/Put/Copy has notes on timed-out requests.
- srmPrepareToPut (5.5.2-h) adds "Some SRM implementation may require targetSURL."
- srmExtendFileLifeTime has a lifetime paired with the SURLs in return.

## **7. In the Information Discovery Functions:**

- srmGetTransferProtocols is added for clients to discover the supported transfer protocols by SRM.
- srmPing is added for clients to check the status of the SRM.

## 8. All Changes after June 20, 2006

### 8.1. July 3, 2006

- TMetaDataPathDetails as an output of srmLs includes unix-like permission returns: ownerPermission, groupPermission, and otherPermission. {owner,group}Permission must show {owner,group} id and the {owner,group} permission.

### 8.2. July 6, 2006

- Behavior on srmPurgeFromSpace is changed for clarity. (more to come later)
- SRM\_FILE\_BUSY status is removed from srmRm for it is a hard remove.

### 8.3. September 27, 2006

- srmLs: TMetaDataPathDetail has path as string, instead of SURL as anyURI. Path reflects an absolute path of a file or a directory.
- srmMkdir, srmRmdir: input parameter directoryPath becomes SURL.
- srmReleaseFiles: input parameter arrayOfSURLs becomes not required. When request token is provided and SURLs are not provided, all files in the request will be released.
- WSDL : extra output parameter in srmExtendFileLifeTimeInSpace is removed. The extra output parameter is not in the spec.

### 8.4. December 15, 2006 (includes contributions from Flavia Donno)

- Typos fixed
  - From srmGetRequestID to srmGetRequestTokens
- srmLs and srmStatusOfLsRequest
  - SRM\_FILE\_BUSY is added
  - SRM\_FILE\_LIFETIME\_EXPIRED is added
  - SRM\_REQUEST\_INPROGRESS is returned at the request level and file level.
  - A comment of “Operation on the path such as browsing the top directory may be prohibited” is added to the SRM\_INVALID\_PATH.
- srmStatusOfLsRequest
  - SRM\_REQUEST\_INPROGRESS is returned at the request level and file level.
- srmReleaseSpace
  - b) will be changed to OUTPUT or CUSTODIAL retention quality space, from durable or permanent space which no longer exists as definitions.
  - If space is being released with *forceFileRelease* option while SURLs are being created with *srmPrepareToPut* or *srmCopy*, the file is removed and SRM\_INVALID\_PATH must be returned by the

*srmPutDone*, *srmStatusOfPutRequest*, or *srmStatusOfCopyRequest* when the file is volatile. If the file is permanent type, the file is moved to the default space, and the space would be successfully released. The subsequent *srmPutDone*, *srmStatusOfPutRequest*, or *srmStatusOfCopyRequest* would be successful.

- If space is being released without *forceFileRelease* option while SURLs are being created with *srmPrepareToPut* or *srmCopy*, SRM\_FAILURE must be returned in *srmReleaseSpace*.
- *srmReleaseFiles*: clarified the comments;
  - If requestToken is not provided and SURLs are provided, then the SRM will release all the files specified by the SURLs owned by the caller, regardless of the requestToken.
  - If requestToken is provided and SURLs are not provided, then the SRM will release all the files in the request that is associated with the requestToken.
  - At least one of requestToken and SURLs must be provided.
  - If requestToken is not provided, then authorizationID may be needed as an additional verification method for the client authorization to release files. It may be inferred or provide in the call.
  - *srmReleaseFiles* is only valid after *srmPrepareToGet* or *srmBringOnline* operations. To release TURLs after a *srmPrepareToPut*, *srmAbortRequest* or *srmAbortFiles* must be used. If a client submits *srmReleaseFiles* after *srmPrepareToPut* or *srmPutDone*, then the SRM server returns SRM\_INVALID\_REQUEST.
- *srmExtendFileLifeTime*: behavior gets clarified:
  - This method allows to change only one lifetime at a time (either SURL lifetime by the *newFileLifetime* or pin lifetime by the *newPinLifetime*), depending on the presence or absence of the request token. SURL lifetimes are on SURLs that resulted from the successful *srmCopy* or *srmPrepareToPut* followed by *srmPutDone*, and pin lifetimes are on TURLs or file copies that resulted from *srmPrepareToGet*, *srmPrepareToPut* or *srmBringOnline*
  - When the *requestToken* is provided, only pin lifetime is extended with *newPinLifetime*.
  - When SURL lifetime is extended with *newFileLifetime*, the request token must not be specified.
  - When lifetime input parameters (*newPinLifetime* or *newFileLifetime*) are not specified, the SRM server uses its default value.
  - Lifetime cannot be extended on the released files, aborted files, expired files, and suspended files. For example, pin lifetime cannot be extended after *srmPutDone* is requested on SURLs after *srmPrepareToPut*. In such case, SRM\_INVALID\_REQUEST at the file level must be returned, and SRM\_PARTIAL\_SUCCESS or SRM\_FAILURE must be returned at the request level.

- If input parameters *newFileLifetime* or *newPinLifetime* request exceeds the remaining lifetime of the space, then SRM\_SUCCESS is returned at the request and file level, and *TSURLLifetimeReturnStatus* contains the remaining lifetime.
  - Lifetime extension must fail on SURLs when their status is SRM\_FILE\_BUSY.
  - SRM\_INVALID\_REQUEST is added at the file level.
- srmExtendFileLifeTimeInSpace: added comments
  - If input parameters *newLifetime* request exceed the remaining lifetime of the space, then SRM\_SUCCESS is returned at the request and file level, and *TSURLLifetimeReturnStatus* contains the remaining lifetime.
  - Lifetime extension must fail on SURLs when their status is SRM\_FILE\_BUSY.
  - *arrayOfSURLs* are optional. When SURLs are not provided, all files in the space must have the new extended lifetimes.
  - This method applied only to SURLs, and output parameter *pinLifetime* in *TSURLLifetimeReturnStatus* must be null
- srmUpdateSpace
  - Output parameter, *lifetimeGranted* is clarified as it is relative to the calling time.
- srmReserveSpace
  - includes SRM\_REQUEST\_INPROGRESS as a valid return status.
  - Data type of input parameter *desiredLifetimeOfReservedSpace* is corrected to be int.
  - Optional input parameters in *TTransferParameters* may collide with the characteristics of the space specified. In this case, *TTransferParameters* as an input parameter must be ignored.
- srmGetSpaceMetaData
  - description about the *unusedSize* is added.
  - SRM\_EXCEED\_ALLOCATION is added at the space level.
- srmRm
  - *srmRm* will remove SURLs even if the statuses of the SURLs are SRM\_FILE\_BUSY. In this case, operations such as *srmPrepareToPut* or *srmCopy* that holds the SURL status as SRM\_FILE\_BUSY must return SRM\_INVALID\_PATH upon status request or *srmPutDone*.
- srmSetPermission
  - *srmSetPermission* will modify permissions on SURLs even if the statuses of the SURLs are SRM\_FILE\_BUSY.
- srmMv
  - *srmMv* must fail on SURL that its status is SRM\_FILE\_BUSY, and SRM\_INVALID\_REQUEST must be returned.
- srmPrepareToGet
  - If input parameter *desiredTotalRequestTime* is 0 (zero), each file request will be tried at least once. Negative value is invalid.
  - Output parameter *remainingTotalRequestTime* indicates how long the *desiredTotalRequestTime* is left. If *remainingTotalRequestTime* is 0

(zero), the request has been timed out. If *remainingTotalRequestTime* is a negative value (-1), it would mean that each file request will be tried at least once.

- srmBringOnline
  - If input parameter *desiredTotalRequestTime* is 0 (zero), each file request will be tried at least once. Negative value is not valid
  - Output parameter *remainingDeferredStartTime* indicates how long the *deferredStartTime* is left, if supported. Negative value is not valid.
  - Output parameter *remainingTotalRequestTime* indicates how long the *desiredTotalRequestTime* is left. If *remainingTotalRequestTime* is 0 (zero), the request has been timed out. If *remainingTotalRequestTime* is a negative value (-1), it would mean that each file request will be tried at least once.
- srmPrepareToPut
  - comments added: TURLs will not be valid any more after the *desiredPinLifetime* is over if *srmPutDone* or *srmAbortRequest* is not submitted on the SURL before expiration.
  - Upon *srmPrepareToPut*, SURL entry is inserted to the name space, and any methods that access the SURL such as *srmLs*, *srmBringOnline* and *srmPrepareToGet* must return SRM\_FILE\_BUSY at the file level. If another *srmPrepareToPut* or *srmCopy* were requested on the same SURL, SRM\_FILE\_BUSY must be returned if the SURL can be overwritten, otherwise SRM\_DUPLICATION\_ERROR must be returned at the file level.
  - Input parameter *desiredFileLifetime* is the lifetime of the SURL when the file is put into the storage system. It does not refer to the lifetime (expiration time) of the TURL. Lifetime on SURL starts when successful *srmPutDone* is executed.
  - If input parameter *desiredTotalRequestTime* is 0 (zero), each file request will be tried at least once. Negative value is invalid.
  - Output parameter *remainingTotalRequestTime* indicates how long the *desiredTotalRequestTime* is left. If *remainingTotalRequestTime* is 0 (zero), the request has been timed out. If *remainingTotalRequestTime* is a negative value (-1), it would mean that each file request will be tried at least once.
- srmStatusOfPutRequest
  - comments added: TURLs will not be valid any more after the pin lifetime is over if *srmPutDone* or *srmAbortRequest* is not submitted on the SURL before expiration.
  - Lifetime on SURL starts when successful *srmPutDone* is executed
  - SRM\_ABORTED is returned at the request level at the successful abort of the request.
  - SRM\_NO\_USER\_SPACE, SRM\_NO\_FREE\_SPACE and SRM\_FILE\_BUSY are added at the file level status.
  - *srmRm* may remove SURLs even if the statuses of the SURLs are SRM\_FILE\_BUSY. In this case, the status for *srmPrepareToPut*

request must return SRM\_INVALID\_PATH upon status request or srmPutDone.

- srmPutDone
  - *srmRm* may remove URLs even if the statuses of the URLs are SRM\_FILE\_BUSY. In this case, SRM\_INVALID\_PATH must be returned upon *srmPutDone* request.
  - If any additional *srmPutDone* is requested on the same URL, SRM\_DUPLICATION\_ERROR must be returned at the file level.
- srmCopy
  - Upon srmCopy, URL entry is inserted to the target name space, and any methods that access the target URL such as srmLs, srmBringOnline and srmPrepareToGet must return SRM\_FILE\_BUSY at the file level. If another srmPrepareToPut or srmCopy were requested on the same target URL, SRM\_FILE\_BUSY must be returned if the target URL can be overwritten, otherwise SRM\_DUPLICATION\_ERROR must be returned at the file level.
  - If input parameter *desiredTotalRequestTime* is 0 (zero), each file request will be tried at least once. Negative value is invalid.
  - Output parameter *remainingTotalRequestTime* indicates how long the *desiredTotalRequestTime* is left. If *remainingTotalRequestTime* is 0 (zero), the request has been timed out. If *remainingTotalRequestTime* is a negative value (-1), it would mean that each file request will be tried at least once.
- srmStatusOfCopyRequest
  - *srmRm* may remove URLs even if the statuses of the URLs are SRM\_FILE\_BUSY. In this case, the status for srmCopy request must return SRM\_INVALID\_PATH upon status request.
- Request Token
  - Includes a statement about the lifetime of the request token.
- User request/space descriptions
  - Statement that *userRequest[Space]Description* may be null, and it is case-sensitive when provided. SRM server is expected to keep it as client provides. It can be reused by the client. It can be used in the *srmGetRequest[Space]Tokens* function to get back the system assigned request or space tokens.
- Streaming mode
  - srmPrepareToGet, srmStatusOfGetRequest, srmBringOnline, srmStatusOfBringOnlineRequest, srmPrepareToPut, srmStatusOfPutRequest, srmCopy, srmStatusOfCopyRequest
  - added comments
  - SRM\_NO\_USER\_SPACE and SRM\_NO\_FREE\_SPACE are added at the file level status.
- estimatedWaitTime
  - TGetRequestFileStatus, TBringOnlineRequestFileStatus, TPutRequestFileStatus, TCopyRequestFileStatus

- Negative value, -1, for unknown.
- Transfer parameters
  - TTransferParameters may be provided optionally in the methods such as srmPrepareToGet, srmBringOnline, srmPrepareToPut and srmReserveSpace. Optional input parameters in TTransferParameters may collide with the characteristics of the space specified. In this case, TTransferParameters as an input parameter must be ignored.
- TMetaDataPathDetail: added comments
  - *lifetimeAssigned* is the total lifetime that is assigned on the SURL. It includes all SURL lifetime extensions if extended.
  - *lifetimeLeft* is the remaining lifetime that is left on the SURL.
- TMetaDataSpace: added comments
  - *lifetimeAssigned* is the total lifetime that is assigned to the space. It includes all space lifetime extensions if extended.
  - *lifetimeLeft* is the remaining lifetime that is left on the space.
-