SAGA-iRODS Adaptor
Version: 1.0

Yutaka Kawai, KEK
February 15, 2010

The SAGA-iRODS Adaptor Specification

This document describes the specification for the SAGA-iRODS Adaptor (SIA).

Status of This Document

This guide is still work in progress.

Contents

1 SIA Name Spaces **3**

1.1 saga::name_space::entry Specification 3

1.2 saga::name_space::entry Specification Detail 4

1.3 saga::name_space::directory Specification 14

1.4 saga::name_space::directory Specification Detail 15

1 SIA Name Spaces

1.1 saga::name_space::entry Specification

saga::name_space::entry

```
namespace saga
{
  namespace name_space
  {
    class entry
        : public saga::object,
          public saga::monitorable,
          public saga::permissions
    {
      entry  (session const & s,
              saga::url       url,
              int             mode = None);
      entry  (saga::url       url,
              int             mode = None);
      entry  (void);
      ~entry (void);

      // inspection methods
      saga::url get_url   (void) const;
      saga::url get_cwd   (void) const;
      saga::url get_name  (void) const;
      saga::url read_link (void) const;

      bool      is_dir    (void) const;
      bool      is_entry  (void) const;
      bool      is_link   (void) const;

      // management methods
      void      copy      (saga::url target,
                           int flags = saga::name_space::None);
      void      link      (saga::url target,
                           int flags = saga::name_space::None);
      void      move      (saga::url target,
                           int flags = saga::name_space::None);
      void      remove    (int flags = saga::name_space::None);
      void      close     (double timeout = 0.0);
    };
  }
}

```

1.2 saga::name_space::entry Specification Detail

1.2.1 saga::name_space::entry class

```
- entry

Purpose:  CONSTRUCTOR; create the object
Format:   entry     (session const &s,
                     saga::url     url,
                     int           mode = None)
Inputs:   s:     session handle
          url:   initial working dir
          mode:  open mode
InOuts:   N/A
Outputs:  obj:   the newly created object
PreCond:  N/A
PostCond: - the entry is opened.
          - 'Owner' of target is the id of the context
            use to perform the opereration, if the
            entry gets created.
Perms:    Exec for parent directory.
          Write for parent directory if Create is set.
          Write for url if Write is set.
          Read for url if Read is set.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          AlreadyExists
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - the default mode is 'None' (0)
          - the constructor performs an open of the
            entry - all notes to the respective open
            call (on namespace_directory) apply.


- ~entry

Purpose:  DESTRUCTOR; destroy the object
Format:   ~entry (ns_entry obj)
```

```
Inputs:   obj:   the object to destroy
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - the entry is closed.
Perms:    N/A
Throws:   N/A
Notes:    - if the instance was not closed before, the
            destructor performs a close() on the instance,
            and all notes to close() apply.
```

1.2.2 saga::name_space::entry::get_url

```
- saga::url get_url (void)

Purpose:  obtain the complete url pointing to the entry
Format:   saga::url get_url (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  url:   url pointing to the entry
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
          IncorrectState
          Timeout
          NoSuccess
Notes:    N/A
```

1.2.3 saga::name_space::entry::get_cwd

```
- saga::url get_cwd (void)

Purpose:  Not Implemented
```

```
Format:   saga::url get_cwd (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    N/A
```

1.2.4 saga::name_space::entry::get_name

```
- saga::url get_name (void)

Purpose:  obtain the name part of the url path element
Format:   saga::url get_name (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  name:   last part of path element
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
          IncorrectState
          Timeout
          NoSuccess
Notes:    N/A
```

1.2.5 saga::name_space::entry::read_link

```
- saga::url read_link (void)

Purpose:  Not Implemented
Format:   saga::url read_link (void);
```

```
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    - iRODS v2.2 does not support link.
```

1.2.6 saga::name_space::entry::is_dir

```
- bool is_dir (void)

Purpose:  tests the entry for being a directory
Format:   bool is_dir (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  test:   boolean indicating if entry
                  is a directory
PreCond:  N/A
PostCond: N/A
Perms:    Query
          Query for parent directory
Throws:   NotImplemented
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - returns true if entry is a directory, false
            otherwise
          - similar to ftest -df as defined by POSIX.
```

1.2.7 saga::name_space::entry::is_entry

```
- bool is_entry (void)

Purpose:  tests the entry for being an ns_entry
Format:   bool is_entry (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  test:   boolean indicating if entry
                  is an ns_entry
PreCond:  N/A
PostCond: N/A
Perms:    Query
          Query for parent directory
Throws:   NotImplemented
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - the method returns false if the entry is a
            link or a directory (although an ns_directory
            IS_A ns_entry, false is returned on a test on
            an ns_directory) -otherwise true is returned.
          - similar to ftest -ff as defined by POSIX.
```

1.2.8 saga::name_space::entry::is_link

```
- bool is_link (void)

Purpose:  Not Implemented
Format:   bool is_link (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
```

```
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    - iRODS v2.2 does not support link.
```

1.2.9 saga::name_space::entry::copy

```
- void copy (saga::url target, int flags = None);

Purpose:  copy the entry to another part of the name space
Format:   void copy (saga::url target, int mode = None);
Inputs:   target: name to copy to
          flags:  flags defining the operation mode
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - an identical copy exists at target.
          - fOwnerf of target is the id of the context
            use to perform the opereration, if target gets
            created.
Perms:    Query
          Exec for parent directory.
          Query for target.
          Query for targetfs parent directory.
          Exec for targetfs parent directory.
          Write for target
                if target does exist.
          Write for targetfs parent directory
                if target does not exist.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          AlreadyExists
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
```

```
Notes:    - if the target is a directory, the source entry
            is copied into that directory
          - a fBadParameterf exception is thrown if the
            source is a directory and the fRecursivef flag
            is not set.
          - a fBadParameterf exception is thrown if the
            source is not a directory and the fRecursivef
            flag is set.
          - if the target lies in a non-existing part of
            the name space, a fDoesNotExistf exception is
            thrown, unless the fCreateParentsf flag is
            given -then that part of the name space must
            be created.
          - if the target already exists, it will be
            overwritten if the fOverwritef flag is set,
            otherwise it is an fAlreadyExistsf exception.
          - if a directory is to be copied recursively,
            but the target exists and is not a directory,
            and not a link to a directory, an
            fAlreadyExistsf exception is thrown even if
            the fOverwritef flag is set.
          - if the instance points at an symbolic link,
            the source is deeply dereferenced before copy.
            If derefencing is impossible (e.g. on a broken
            link), an fIncorrectStatef exception is thrown.
          - other flags are not allowed, and cause a
            fBadParameterf exception.
          - the default flags are fNonef (0).
          - similar to fcpf as defined by POSIX.
```

1.2.10 saga::name_space::entry::link

```
- void link (saga::url target, int flags = None);

Purpose:  Not Implemented
Format:   void link (saga::url target, int flags = None);
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
```

```
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    - iRODS v2.2 does not support link.
```

1.2.11 saga::name_space::entry::move

```
- void move (saga::url target, int flags = None);

Purpose:  rename source to target, or move source to
          target if target is a directory.
Format:   void move (saga::url target, int flags = None);
Inputs:   target: name to move to
          flags:  flags defining the operation mode
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - an identical copy exists at target.
          - the original entry is removed.
          - fOwnerf of target is the id of the context
            use to perform the opereration, if target gets
            created.
Perms:    Query
          Write
          Exec  for parent directory.
          Write for parent directory.
          Query for target.
          Exec  for targetfs parent directory.
          Write for target
                if target does exist.
          Write for targetfs parent directory
                if target does not exist.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          AlreadyExists
          IncorrectState
          PermissionDenied
          AuthorizationFailed
```

```
            AuthenticationFailed
            Timeout
            NoSuccess
Notes:      - if the target is a directory, the source entry
              is moved into that directory
            - a fBadParameterf exception is thrown if the
              source is a directory and the fRecursivef flag
              is not set.
            - a fBadParameterf exception is thrown if the
              source is not a directory and the fRecursivef
              flag is set.
            - if the target lies in a non-existing part of
              the name space, a fDoesNotExistf exception is
              thrown, unless the fCreateParentsf flag is
              given - then that part of the name space must
              be created.
            - if the target already exists, it will be
              overwritten if the fOverwritef flag is set,
              otherwise it is an fAlreadyExistsf exception.
            - if the instance points at an symbolic link,
              the source is not dereferenced before moving,
              unless the fDereferencef flag is given.
              If derefencing is impossible (e.g. on a broken
              link), an fIncorrectStatef exception is thrown.
            - other flags are not allowed, and cause a
              fBadParameterf exception.
            - the default flags are fNonef (0).
            - similar to fmvf as defined by POSIX.
```

1.2.12 saga::name_space::entry::remove

```
- void remove (int flags = None);

Purpose:  removes this entry, and closes it
Format:   void remove (int flags = None);
Inputs:   flags:  flags defining the operation mode
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
```

```
PostCond: - the original entry is closed and removed.
Perms:    Query
          Write
          Exec  for parent directory.
          Write for parent directory.
Throws:   NotImplemented
          BadParameter
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - a fBadParameterf exception is thrown if the
            source is a directory and the fRecursivef flag
            is not set.
          - a fBadParameterf exception is thrown if the
            source is not a directory and the fRecursivef
            flag is set.
          - the source will not be dereferenced unless the
            fDereferencef flag is given. If derefencing is
            impossible (e.g. on a broken link), an
            fIncorrectStatef exception is thrown.
          - other flags are not allowed, and cause a
          fBadParameterf exception.
          - the default flags are fNonef (0).
          - if the instance was not closed before, this
            call performs a close() on the instance, and
            all notes to close() apply.
          - similar to frmf as defined by POSIX.
```

1.2.13 saga::name_space::entry::close

```
- void close (float timeout = 0.0);

Purpose:  closes the object
Format:   void close (float timeout = 0.0);
Inputs:   timeout: seconds to wait
InOuts:   N/A
```

```
Outputs:  N/A
PreCond:  N/A
PostCond: - the entry instance is closed.
Perms:    N/A
Throws:   NotImplemented
          IncorrectState
          NoSuccess
Notes:    - any subsequent method call on the object
            MUST raise an fIncorrectStatef exception
            (apart from DESTRUCTOR and close()).
          - close() can be called multiple times, with no
            side effects.
          - if close() is implicitely called in the
            DESTRUCTOR, it will never throw an exception.
          %- for resource deallocation semantics, see Section 2.
          %- for timeout semantics, see Section 2.
```

1.3 saga::name_space::directory Specification

saga::name_space::directory

```
1  namespace saga
2  {
3    namespace name_space
4    {
5      class directory
6          : public saga::name_space::entry
7      {
8        public:
9          directory  (session const & s,
10                     saga::url       url,
11                     int             mode = None);
12         directory  (saga::url       url,
13                     int             mode = None);
14         directory  (void);
15         ~directory (void);
16
17         void        change_dir (saga::url    target)
18         std::vector <saga::url>
19                     list       (std::string  pattern = "*",
20                                 int          flags   = None) const;
21         std::vector <saga::url>
22                     find       (std::string  pattern,
```

```
23                                int          flags   = Recursive) const;
24
25          saga::url   read_link  (saga::url    url) const;
26          bool        exists     (saga::url    url) const;
27          bool        is_dir     (saga::url    url) const;
28          bool        is_entry   (saga::url    url) const;
29          bool        is_link    (saga::url    url) const;
30          unsigned    int get_num_entries
31                                 (void) const;
32          saga::url   get_entry  (unsigned int entry) const;
33          void        copy       (saga::url    source_url,
34                                  saga::url    dest_url,
35                                  int          flags = None);
36          void        link       (saga::url    source_url,
37                                  saga::url    dest_url,
38                                  int          flags = None);
39          void        move       (saga::url    source_url,
40                                  saga::url    dest_url,
41                                  int          flags = None);
42          void        remove     (saga::url    url,
43                                  int          flags = None);
44          void        make_dir   (saga::url    url,
45                                  int          flags = None);
46          entry       open       (saga::url    url,
47                                  int          flags = None);
48          directory   open_dir   (saga::url    url,
49                                  int          flags = None);
50        };
51      } // namespace_dir
52    } // namespace saga
53
```

1.4 saga::name_space::directory Specification Detail

1.4.1 saga::name_space::directory class

```
- directory

Purpose:  CONSTRUCTOR; create the object
Format:   directory (session const &s,
                     saga::url     url,
                     int           mode = None)
Inputs:   s:     session handle for
          url:   initial working dir
```

```
          mode:  open mode
InOuts:   N/A
Outputs:  obj:   the newly created object
PreCond:  N/A
PostCond: - the directory is opened.
          - 'Owner' of target is the id of the context
            use to perform the opereration, if the
            directory gets created.
Perms:    Exec for parent directory.
          Write for parent directory if Create is set.
          Write for url if Write is set.
          Read for url if Read is set.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - the semantics of the inherited constructors
            apply
          - the constructor performs an open of the
            entry
          - all notes to the respective open call apply.
          - the default flags are fNonef (0).


- ~directory

Purpose:  DESTRUCTOR; destroy the object
Format:   ~directory (ns_directory obj)
Inputs:   obj:   the object to destroy
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - the directory is closed.
Perms:    N/A
Throws:   N/A
Notes:    - the semantics of the inherited destructors
          apply
```

1.4.2 saga::name_space::directory::change_dir

```
- void change_dir (saga::url dir);

Purpose:  change the working directory
Format:   void change_dir (saga::url dir);
Inputs:   dir:   directory to change to
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - dir is the directory the instance represents.
Perms:    Exec for dir.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - if fdirf can be parsed as URL, but contains an
            invalid directory name, a fBadParameterf
            exception is thrown.
          - if fdirf does not exist, a fDoesNotExistf
            exception is thrown.
          - similar to the fcdf command in the POSIX
            shell.
```

1.4.3 saga::name_space::directory::list

```
- std::vector<saga::url> list (string name_pattern = ".",
                               int flags = None);

Purpose:  list entries in this directory
Format:   std::vector<saga::url>
                        list (string name_pattern = ".",
```

```
                                int flags = None);
Inputs:   flags:        flags defining the operation modus
          name_pattern: name or pattern to list
InOuts:   N/A
Outputs:  names:  array of names matching the name_pattern
PreCond:  N/A
PostCond: N/A
Perms:    Query for entries specified by name_pattern.
          Exec for parent directories of these entries.
          Query for parent directories of these entries.
          Read for directories specified by name_pattern.
          Exec for directories specified by name_pattern.
          Exec for parent directories of these directories.
          Query for parent directories of these directories.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - if name_pattern is not given (i.e. is an empty
            string), all entries in the current working
            directory are listed.
          - if name_pattern is given and points to a
            directory, the contents of that directory
            are listed.
          - the name_pattern follows the standard POSIX
            shell wildcard specification, as described
            above.
          - list does not follow symbolically linked
            directories, unless the fDereferencef flag
            is specified - otherwise list lists symbolic
            link entries with a matching name.
          - if the fDeReferencef flag is set, list
            returns the name of link targets, not of the
            link entry itself.
          - the default flags are fNonef (0).
          - other flags are not allowed, and cause a
            fBadParameterf exception.
          - if the name_pattern cannot be parsed, a
            fBadParameterf exception with a descriptive
            error message is thrown.
```

```
          - if the name_pattern does not match any entry,
            an empty list is returned, but no exception is
            raised.
          - similar to flsf as defined by POSIX.
```

1.4.4 saga::name_space::directory::find

```
- std::vector<saga::url> find (string name_pattern,
                               int flags = Recursive);

Purpose:  find entries in the current directory and below
Format:   std::vector<saga::url>
                        find (string name_pattern,
                              int flags = Recursive);
Inputs:   flags:        flags defining the operation modus
          name_pattern: pattern for names of entries to be
                        found
InOuts:   N/A
Outputs:  names:  array of names matching the name_pattern
PreCond:  N/A
PostCond: N/A
Perms:    Read for cwd.
          Query for entries specified by name_pattern.
          Exec for parent directories of these entries.
          Query for parent directories of these entries.
          Read for directories specified by name_pattern.
          Exec for directories specified by name_pattern.
          Exec for parent directories of these directories.
          Query for parent directories of these directories.
Throws:   NotImplemented
          BadParameter
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - find operates recursively below the current
            working directory if the fRecursivef flag is
            specified (default)
```

```
  - find does not follow symbolically linked
    directories, unless the fDereferencef flag
    is specified - otherwise find lists symbolic
    link entries with a matching name.
  - the default flags are fRecursivef (1).
  - other flags are not allowed, and cause a
    fBadParameterf exception.
  - the name_pattern follows the standard POSIX
    shell wildcard specification, as described above.
  - the matching entries returned are path names
    relative to cwd.
  - similar to ffindf as defined by POSIX, but
    limited to the -name option.
```

1.4.5 saga::name_space::directory::read_link

```
- saga::url read_link (saga::url name);

Purpose:  NotImplemented
Format:   saga::url read_link (saga::url name);
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    - iRODS v2.2 does not support link.
```

1.4.6 saga::name_space::directory::exists

```
- bool exists (saga::url name);

Purpose:  returns true if entry exists, false otherwise
```

```
Format:   bool exists (saga::url name);
Inputs:   name:   name to be tested for existence
InOuts:   N/A
Outputs:  exists: boolean indicating existence of name
PreCond:  N/A
PostCond: N/A
Perms:    Query for name.
          Exec for namefs parent directory.
          Read for namefs parent directory.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - if fnamef can be parsed as URL, but contains
            an invalid entry name, an fBadParameterf
            exception is thrown.
          - note that no exception is thrown if the entry
            does not exist - the method just returns
            ffalsef in this case.
          - similar to ftest -ef as defined by POSIX.
```

1.4.7 saga::name_space::directory::is_dir

```
- bool is_dir (saga::url name);

Purpose:  tests name for being a directory
Format:   bool is_dir (saga::url name);
Inputs:   name: name to be tested
InOuts:   N/A
Outputs:  test: boolean indicating if name is a directory
PreCond:  N/A
PostCond: N/A
Perms:    Query for name.
          Exec for namefs parent directory.
          Read for namefs parent directory.
```

```
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - returns true if the instance represents
            a directory entry, false otherwise
          - all notes to the ns_ntry::is_dir() method apply.
          - if fnamef can be parsed as URL, but contains
            an invalid entry name, an fBadParameterf
            exception is thrown.
          - if fnamef is a valid entry name but the entry
            does not exist, a fDoesNotExistf exception is
            thrown.
          - similar to ftest -df as defined by POSIX.
```

1.4.8 saga::name_space::directory::is_entry

```
- bool is_entry (saga::url name);

Purpose:  tests name for being an ns_entry
Format:   bool is_entry (saga::url name);
Inputs:   name: name to be tested
InOuts:   N/A
Outputs:  test: boolean indicating if name is a non-directory
                entry
PreCond:  N/A
PostCond: N/A
Perms:    Query for name.
          Exec for namefs parent directory.
          Read for namefs parent directory.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
```

```
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - all notes to the ns_ntry::is_entry() method
            apply.
          - if fnamef can be parsed as URL, but contains
            an invalid entry name, a fBadParameterf
            exception is thrown.
          - if fnamef is a valid entry name but the entry
            does not exist, a fDoesNotExistf exception is
            thrown.
          - similar to ftest -ff as defined by POSIX.
```

1.4.9 saga::name_space::directory::is_link

```
- bool is_link (saga::url name);

Purpose:  tests name for being a symbolic link
Format:   bool is_link (saga::url name);
Inputs:   name: name to be tested
InOuts:   N/A
Outputs:  test: boolean indicating if name is a link
PreCond:  N/A
PostCond: N/A
Perms:    Query for name.
          Exec for namefs parent directory.
          Read for namefs parent directory.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
```

```
Notes:    - all notes to the ns_ntry::is_link() method
            apply.
          - if fnamef can be parsed as URL, but contains
            an invalid entry name, a fBadParameterf
            exception is thrown.
          - if fnamef is a valid entry name but the entry
            does not exist, a fDoesNotExistf exception is
            thrown.
          - similar to ftest -Lf as defined by POSIX.
```

1.4.10 saga::name_space::directory::get_num_entries

```
- const unsigned int get_num_entries (void);

Purpose:  gives the number of entries in the directory
Format:   const unsigned int get_num_entries (void);
Inputs:   N/A
InOuts:   N/A
Outputs:  num: number of entries in the directory
PreCond:  N/A
PostCond: N/A
Perms:    Query for cwd.
          Exec for cwd.
          Read for cwd.
Throws:   NotImplemented
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - at the time of using the result of this call,
            the actual number of entries may already have
            changed (no locking is implied)
          - vaguely similar to fopendirf/freaddirf (2) as
            defined by POSIX.
```

1.4.11 saga::name_space::directory::get_entry

```
- const saga::url get_entry (unsigned int entry);

Purpose:  gives the name of an entry in the directory
          based upon the enumeration defined by
          get_num_entries
Format:   const saga::url get_entry (unsigned int entry);
Inputs:   entry: index of entry to get
InOuts:   N/A
Outputs:  name:  name of entry at index
PreCond:  N/A
PostCond: N/A
Perms:    Query for cwd.
          Exec for cwd.
          Read for cwd.
Throws:   NotImplemented
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - f0f is the first entry
          - there is no sort order implied by the
            enumeration, however an underlying
            implementation MAY choose to sort the entries
          - subsequent calls to get_entry and/or
            get_num_entries may return inconsistent data,
            i.e. no locking or state tracking is implied.
            In particular, an index may be invalid - a
            fDoesNotExistf exception is then thrown (not a
            fBadParameterf exception).
          - vaguely similar to fopendirf/freaddirf (2) as
            defined by POSIX.
```

1.4.12 saga::name_space::directory::copy

```
- void copy (saga::url source, saga::url target,
             int flags = None);

Purpose:  copy the entry to another part of the name space
Format:   void copy (saga::url source, saga::url target,
                     int flags = None);
Inputs:   source: name to copy
          target: name to copy to
          flags:  flags defining the operation modus
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - an identical copy of source exists at target.
          - fOwnerf of target is the id of the context
            used to perform the opereration if target gets
            created.
Perms:    Query for source.
          Exec for sourcefs parent directory.
          Query for target.
          Query for targetfs parent directory.
          Exec for targetfs parent directory.
          Write for target
                if target does exist.
          Write for targetfs parent directory
                if target does not exist.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          AlreadyExists
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - all notes to the ns_entry::copy() method
            apply.
          - the default flags are fNonef (0).
          - if fnamef can be parsed as URL, but contains
            an invalid entry name, a fBadParameterf
            exception is thrown.
          - if fnamef is a valid entry name but the entry
            does not exist, a fDoesNotExistf exception is
            thrown.
```

1.4.13 saga::name_space::directory::link

```
- void link (saga::url source, saga::url target,
             int flags = None);

Purpose:  NotImplemented
Format:   void link (saga::url source, saga::url target,
                     int flags = None);
Inputs:   N/A
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: N/A
Perms:    N/A
Throws:   NotImplemented
Notes:    - iRODS v2.2 does not support link.
```

1.4.14 saga::name_space::directory::move

```
- void move (saga::url source, saga::url target,
             int flags = None);

Purpose:  rename source to target, or move source to
          target if target is a directory.
Format:   void move (saga::url source, saga::url target,
                     int flags = None);
Inputs:   source: name to move
          target: name to move to
          flags:  flags defining the operation modus
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - an identical copy of source exists at target.
          - fOwnerf of target is the id of the context
            used to perform the opereration if target gets
```

```
            created.
Perms:      Query for source.
            Exec for sourcefs parent directory.
            Query for target.
            Query for targetfs parent directory.
            Exec for targetfs parent directory.
            Write for target
                  if target does exist.
            Write for targetfs parent directory
                  if target does not exist.
Throws:     NotImplemented
            IncorrectURL
            BadParameter
            AlreadyExists
            DoesNotExist
            IncorrectState
            PermissionDenied
            AuthorizationFailed
            AuthenticationFailed
            Timeout
            NoSuccess
Notes:      - all notes to the ns_entry::move() method
              apply.
            - if the fRecursivef flag is defined, the source
              is recursively copied if it is a directory;
              otherwise this flag is ignored.
            - if the fDereferencef flag is specified, the
              method applies to the link target of source.
              The flag causes a fBadParameterf exception
              if source is not a link.
            - if the the target already exists, the
              fOverwritef flag must be specified, otherwise
              an fAlreadyExistsf exception is thrown.
            - the default flags are fNonef (0).
            - other flags are not allowed on this method,
              and cause a fBadParameterf exception.
            - if fsourcef can be parsed as URL, but contains
              an invalid entry name, a fBadParameterf
              exception is thrown.
            - if fsourcef is a valid entry name but the entry
              does not exist, a fDoesNotExistf exception is
              thrown.
            - moving any parent or the current directoy
              (e.g. f.f, f..f etc.) is not allowed,
              and throws a fBadParameterf exception
```

1.4.15 saga::name_space::directory::remove

```
- void remove (saga::url target, int flags = None);

Purpose:  removes the entry
Format:   void remove (saga::url target, int flags = None);
Inputs:   target: entry to be removed
          flags:  flags defining the operation modus
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - source is removed.
          - source is closed if it refers to the cwd.
Perms:    Query for source.
          Write for source.
          Exec for sourcefs parent directory.
          Write for sourcefs parent directory.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          AlreadyExists
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - all notes to the ns_entry::remove() method
            apply.
          - if the fRecursivef flag is defined, the source
            is recursively removed if it is a directory;
            otherwise this flag is ignored.
          - if the fDereferencef flag is specified, the
            method applies to the link target of source.
            The flag causes a fBadParameterf exception if
            source is not a link.
          - the default flags are fNonef (0).
          - other flags are not allowed on this method,
            and cause a fBadParameterf exception.
          - if fsourcef can be parsed as URL, but contains
```

```
            an invalid entry name, a fBadParameterf
            exception is thrown.
          - if fsourcef is a valid entry name but the entry
            does not exist, a fDoesNotExistf exception is
            thrown.
          - removing any parent or the current directoy
            (e.g. f.f, f..f etc.) is not allowed, and
            throws a fBadParameterf exception
```

1.4.16 saga::name_space::directory::make_dir

```
- void make_dir (saga::url target, int flags = None);

Purpose:  creates a new directory
Format:   void make_dir (saga::url target, int flags = None);
Inputs:   target: directory to create
          flags:  flags defining the operation modus
InOuts:   N/A
Outputs:  N/A
PreCond:  N/A
PostCond: - fOwnerf of target is the id of the context
            used to perform the opereration if target gets
            created.
Perms:    Exec for targetfs parent directory.
          Write for targetfs parent directory.
          Write for target if Write is set.
          Read for target if Read is set.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          AlreadyExists
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - if the parent directory or directories do not
            exist, the fCreateParentsf flag must be set
```

```
            or a fDoesNotExistf exception is thrown.
            If set, the parent directories are created as
            well.
          - an fAlreadyExistsf exception is thrown if the
            directory already exists and the fExclusivef
            flag is given.
          - the default flags are fNonef (0).
          - other flags are not allowed on this method,
            and cause a fBadParameterf exception.
          - if ftargetf can be parsed as URL, but contains
            an invalid entry name, a fBadParameterf
            exception is thrown.
          - similar to fmkdirf (2) as defined by POSIX.
```

1.4.17 saga::name_space::directory::open

```
- entry open (saga::url name, int flags = None);

Purpose:  creates a new ns_entry instance
Format:   entry open (saga::url name, int flags = None);
Inputs:   name:    entry
          flags:   flags defining the operation modus
InOuts:   N/A
Outputs:  entry:   opened entry instance
PreCond:  N/A
PostCond: - the session of the returned instance is that
            of the calling instance.
          - fOwnerf of name is the id of the context
            used to perform the opereration if name gets
            created.
          - the namespace entry is created if it does not
            yet exist, and the CREATE flag is specified.
Perms:    Exec for namefs parent directory.
          Write for namefs parent directory if Create is set.
          Write for name if Write is set.
          Read for name if Read is set.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          AlreadyExists
```

```
            DoesNotExist
            IncorrectState
            PermissionDenied
            AuthorizationFailed
            AuthenticationFailed
            Timeout
            NoSuccess
Notes:      - a fBadParameterf exception is thrown if fnamef
              points to a directory, or is an invalid entry
              name.
            - a fDoesNotExistf exception is thrown if fnamef
              does not exist, and the fCreatef flag is not
              given.
            - a fAlreadyExistsf exception is thrown if fnamef
              does exist, and the fCreatef and fExclusivef
              flags are given.
            - fnamef is always deeply dereferenced, the cwd,
              however, is not changed to the link targets cwd.
            - parent directories are created on the fly if the
              fCreateParentsf and fCreatef flag are both
              given, if they donft exist.
            - the entry is locked on open if the fLockf flag
              is given. If the entry is already in a locked
              state, the open will fail and a descriptive
              error will be issued. If a entry is opened in
              locked mode, any other open on that entry MUST
              fail with a fNoSuccessf exception if the fLockf
              flag is given. Note that a entry can be opened
              in unlocked mode, and then in locked mode,
              without an error getting raised.
              The application programmer must take precautions
              to avoid such situations. The lock will get
              removed on destruction of the entry object, and
              also on close. If an implementation does not
              support locking, a descriptive fBadParameterf
              exception MUST get thrown if the fLockf flag
              is given. Read-locks and Write-locks are not
              distinguished.
            - the default flags are fNonef (0).
            - other flags are not allowed on this method,
              and cause a fBadParameterf exception.
            - similar to fopenf (2) as defined by POSIX.
```

1.4.18 saga::name_space::directory::open_dir

```
- directory open_dir (saga::url name, int flags = None);

Purpose:  creates a new ns_directory instance
Format:   directory open_dir (saga::url target, int flags = None);
Inputs:   name:  directory to open
          flags: flags defining the operation modus
InOuts:   N/A
Outputs:  dir:   opened directory instance
PreCond:  N/A
PostCond: - the session of the returned instance is that of
            the calling instance.
          - fOwnerf of name is the id of the context
            used to perform the opereration if name gets
            created.
          - the namespace directory is created if it
            does not yet exist, and the Create is set.
Perms:    Exec for namefs parent directory.
          Write for namefs parent directory if Create is set.
          Write for name if Write is set.
          Read for name if Read is set.
Throws:   NotImplemented
          IncorrectURL
          BadParameter
          AlreadyExists
          DoesNotExist
          IncorrectState
          PermissionDenied
          AuthorizationFailed
          AuthenticationFailed
          Timeout
          NoSuccess
Notes:    - the cwd of the new dir object instance is set
            to fnamef
          - a fDoesNotExistf exception is thrown if fnamef
            does not exist and the fCreatef flag is not
            given.
          - a fAlreadyExistf exception is thrown if fnamef
            does exist and the fCreatef flag and the
            fExclusivef flag are given.
          - no exception is thrown if fnamef does exist and
            the fCreatef flag is given, and the
            fExclusivef flag is not given.
```

```
- if the fCreatef flag is given, all notes to the
  ns_directory::make_dir() method apply.
- the default flags are fNonef (0).
- other flags are not allowed on this method,
  and cause a fBadParameterf exception.
- fnamef is always deeply dereferenced, however,
  the cwd is still set to fnamef, and not to the
  value of the link target.
- parent directories are created on the fly if
  the fCreateParentsf and fCreatef flag are both
  given, if they donft exist.
- if fnamef can be parsed as URL, but contains
  an invalid directory name,
```
