```
/*
** RELEASE STATEMENT(s):
**                          UNLIMITED RIGHTS
**  The Government has the right to use, modify, reproduce, release, perform,
**  display, or disclose this application programmable interface in whole or in
**  part, in any manner and for any purpose whatsoever, and to have or
**  authorize others to do so.
**
** Distribution Statement A - Approved for public release; distribution is
** unlimited (27 August 2015).
*/

/*
** JTNC Standard:
**  Software Communications Architecture
**  Appendix C: Core Framework Interface Description Language (IDL)
**  Version: 4.1, 20 August 2015
*/

//Source file: CFFileSystem.idl

#ifndef __CFFILESYSTEM_DEFINED
#define __CFFILESYSTEM_DEFINED

#include "CFProperties.idl"
#include "CFFile.idl"

module CF {

   /* This interface defines the operations to enable
      remote access to a physical file system. */
   interface FileSystem {

      /* This exception indicates a set of properties unknown by
         the FileSystem object. */
      exception UnknownFileSystemProperties {
         CF::Properties invalidProperties;
      };

      /* This constant indicates file system size. */
      const string SIZE = "SIZE";

      /* This constant indicates the available space on the file system. */
      const string AVAILABLE_SPACE = "AVAILABLE_SPACE";

      /* This enumerations indicates the type of file entry. A file system can
         have PLAIN or DIRECTORY files and mounted file systems contained
         in a FileSystem. */
      enum FileType {
         PLAIN,
         DIRECTORY,
         FILE_SYSTEM
      };

      /* This structure indicates the information returned
         for a file. */
      struct FileInformationType {
         string name;
         CF::FileSystem::FileType kind;
         unsigned long long size;
         CF::Properties fileProperties;
      };

      typedef sequence <FileInformationType> FileInformationSequence;

      /* The CREATED_TIME_ID is the identifier for the created time file
```

```
      property. */
   const string CREATED_TIME_ID = "CREATED_TIME";

   /* The MODIFIED_TIME_ID is the identifier for the modified time file
      property. */
   const string MODIFIED_TIME_ID = "MODIFIED_TIME";

   /* The LAST_ACCESS_TIME_ID is the identifier for the last access time
      file property. */
   const string LAST_ACCESS_TIME_ID = "LAST_ACCESS_TIME";

   /* This operation removes the file with the given filename. */
   void remove (
         in string fileName
         )
      raises (CF::FileException, CF::InvalidFileName);

   /* This operation copies the source file with the specified
      sourceFileName to the destination file with the specified
      destinationFileName. */
   void copy (
         in string sourceFileName,
         in string destinationFileName
         )
      raises (CF::InvalidFileName, CF::FileException);

   /* This operation checks to see if a file exists based
      on the filename parameter. */
   boolean exists (
         in string fileName
         )
      raises (CF::InvalidFileName);

   /* This operation provides the ability to obtain a list
      of files along with their information in the file system according
      to a given search pattern. */
   CF::FileSystem::FileInformationSequence list (
         in string pattern
         )
      raises (CF::FileException, CF::InvalidFileName);

   /* This operation creates a new File based upon the provided
      file name and returns a File to the opened file. */
   CF::File create (
         in string fileName
         )
      raises (CF::InvalidFileName, CF::FileException);

   /* This operation opens a file for reading or writing based
      upon the input fileName. */
   CF::File open (
         in string fileName,
         in boolean read_Only
         )
      raises (CF::InvalidFileName, CF::FileException);

   /* This operation creates a file system directory based on
      the directoryName given. */
   void mkdir (
         in string directoryName
         )
      raises (CF::InvalidFileName, CF::FileException);

   /* This operation removes a file system directory based
      on the directoryName given. */
   void rmdir (
```

```
            in string directoryName
            )
         raises (CF::InvalidFileName, CF::FileException);

      /* This operation returns file system information to the
         calling client based upon the given fileSystemProperties' ID. */
      void query (
            inout CF::Properties fileSystemProperties
            )
         raises (CF::FileSystem::UnknownFileSystemProperties);
   };
};
#endif
```