DCMTK - Feature #226

Support für Dateien > 2 GByte in DCMTK

2005-02-15 00:00 - Marco Eichelberg

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0:00 hour
Target version:			
Module:	dcmdata, andere?	**Compiler:**	
Operating System:			

Description

Status: erledigt? [JR]

DCMTK unterstützt zur Zeit keine Dateien > 2 GByte. Unter Solaris gibt
es zwei Möglichkeiten, wie man auf größere Dateien zugreifen kann (man
lfcompile, man lfcompile64), unter Linux scheinbar nur die erste Variante.

```
- Long File Support sollte wohl eine Configure-Option sein, default: an wenn
  vorhanden(?)
- In Configure mit "getconf LFS_CFLAGS" die Compiler-Optionen herausfinden,
  die für "Long File Support" gesetzt werden müssen, dito
  "getconf LFS_LDFLAGS" und "getconf LFS_LIBS"
- Auf jeden Fall muss fseeko() statt fseek(), ftello statt ftell verwendet
  werden
- Eine Liste aller Systemfunktionen, die "betroffen" sind, findet man unter
  Solaris mit "man lf64". Für jede dieser Funktionen muss geprüft werden, ob
  DCMTK den "offiziellen" Datentyp verwendet, der ggf. 64-bittig definiert
  ist. Das betrifft neben fseeko/ftello auch lockf, readdir, open, mmap,
  get/setrlimit, fstat, lseek. Nutzung im Toolkit:
  - fseek: dcmdata/apps/pdf2dcm.cc, dcmdata/apps/dcmftest.cc, dcmdata/libsrc/dcistrmf.cc,
           dcmmisc/apps/pdf2dcm.cc, dcmmisc/apps/dcmmkmg.cc, dcmmisc/apps/dcmmkmg.cc,
           dcmsign/apps/dcmsign.cc
  - ftell: dcmdata/libsrc/dcistrmf.cc, dcmmisc/apps/dcmmkmg.cc, dcmsign/apps/dcmsign.cc
  - lockf (unused)
  - readdir: ofstd/libsrc/ofstd.cc, dcmpstat/apps/dcmprscu.cc, dcmpps/libsrc/ppsmgram.cc,
           dcmwlm/libsrc/wlfsim.cc
  - open: dcmqrdb/libsrc/dcmqrcbg.cc, dcmqrdb/libsrc/dcmqrcbm.cc, dcmqrdb/libsrc/dcmqrdbi.cc,
           dcmqrdb/libsrc/dcmqrsrv.cc, dcmqrdb/libsrc/dcmqrtis.cc, dcmpstat/apps/dcmpsrcv.cc,
           dcmpstat/apps/dcmpssnd.cc, dcmnet/apps/storescp.cc, dcmpps/libsrc/ppsmgram.cc,
           dcmpps/libsrc/ppsfsim.cc, dcmwlm/libsrc/wldsfs.cc
  - mmap (unused)
  - get/setrlimit (unused)
  - fstat (unused)
  - lseek: dcmqrdb/libsrc/dcmqrdbi.cc, dcmnet/libsrc/dcompat.cc

- Insbesondere die Stream-Klassen in dcmdata verwenden derzeit explizit
  Uint32, um die aktuelle Position im Stream zu beschreiben. Das muss geändert
  werden. Problematisch ist, dass es Uint64 vermutlich nicht auf jeder
  Plattform gibt. Geschickter wäre es vielleicht, off_t usw. selbst zu
  definieren falls nicht vorhanden.

- Aus der Linux-fseeko-Manpage:
   On many architectures both off_t and long are 32-bit types, but compilation with
     #define _FILE_OFFSET_BITS 64
   will turn off_t into a 64-bit type.

- Ein anderes Makro in diesem Zusammenhang scheint __USE_LARGEFILE64 zu sein,
  dessen Bedeutung ist mir aber nicht ganz klar.

- Implementierungsstrategie:
```

```
  - in configure prüfen, ob es off_t gibt (wenn ja, dann benutzen)
  - in configure prüfen, ob es fseeko/ftello gibt (wenn ja, dann benutzen)
  - in configure prüfen, ob es getconf gibt, wenn ja, LFS_CFLAGS, LFS_LDFLAGS und LFS_LIBS ermitte
ln
  - in configure prüfen, ob es die lfcompile64-Funktionen gibt
  - in configure prüfen, wie man lf-Support bekommt:
    - 1. sizeof(long) == 8 -> nothing to do.
    - 2. off_t und fseeko existieren und sizeof(off_t)==8 (mit #define _FILE_OFFSET_BITS 64) -> lf
compile nutzen
    - 3. lfcompile64 existiert -> lfcompile64 nutzen
```

History

#1 - 2013-09-12 21:55 - Jörg Riesmeier

- % Done changed from 0 to 50

#2 - 2013-09-14 19:06 - Jörg Riesmeier

- Status changed from New to Closed

- % Done changed from 50 to 100