

---

---

**Information technology — Coding  
of audio-visual objects —**

Part 5:

**Reference software**

**AMENDMENT 16: Symbolic Music  
Representation reference software**

*Technologies de l'information — Codage des objets audiovisuels —*

*Partie 5: Logiciel de référence*

*AMENDEMENT 16: Logiciel de référence pour la représentation  
musicale symbolique*

**PDF disclaimer**

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



**COPYRIGHT PROTECTED DOCUMENT**

© ISO/IEC 2008

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office  
Case postale 56 • CH-1211 Geneva 20  
Tel. + 41 22 749 01 11  
Fax + 41 22 749 09 47  
E-mail [copyright@iso.org](mailto:copyright@iso.org)  
Web [www.iso.org](http://www.iso.org)

Published in Switzerland

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Amendment 16 to ISO/IEC 14496-5:2001 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This Amendment adds the Reference Software of Symbolic Music Representation as defined in ISO/IEC 14496-23 and ISO/IEC 14496-11:2005/Amd.5.

There are two parts to the SMR reference software — MPEG-4 Systems dependent and MPEG-4 Systems independent parts. The MPEG-4 Systems dependent part is implemented and made available as a revision to the IM1 reference software for MPEG-4 Systems. The MPEG-4 Systems independent part is available as an independent implementation.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-5:2001/Amd 16:2008

# Information technology — Coding of audio-visual objects —

## Part 5: Reference software

### AMENDMENT 16: Symbolic Music Representation reference software

*In Clause 3: "Audio reference software", add a row to the table:*

|                     |                                                 |
|---------------------|-------------------------------------------------|
| audio/SNHC/mpeg4smr | Symbolic Music Representation decoder and tools |
|---------------------|-------------------------------------------------|

*In Clause 5: "Systems reference software", add a row to the table:*

|                 |                          |
|-----------------|--------------------------|
| systems/IM1-SMR | IM1 supporting SMR nodes |
|-----------------|--------------------------|

*Add Annex D: "Guidance on SMR reference software" and rename Annexes D and E as Annexes E and F respectively.*

## Annex D (informative)

### Guidance on SMR reference software

#### D.1 IM1 Reference Software update

In Systems/IM1-SMR directory the IM1 project has been enhanced to provide support for SMR, in more details:

- the Nodes.cpp, Nodes.h file has been updated to support the two new nodes (MusicScore and ScoreShape);
- the new *MusicScoreProxy* and *ScoreShapeProxy* classes have been created;
- the new SMRDecoder project to build the decoder dll has been created, it provides the interface needed for the integration with IM1, it uses the SMRDecoderLib to process the SMR bitstream, to generate a bitmap representing the score and it is used also to manipulate the SMR content itself.
- The MP4Enc application has been enhanced to allow the encoding in MP4 files of SMR information together with the BIFS scene description.

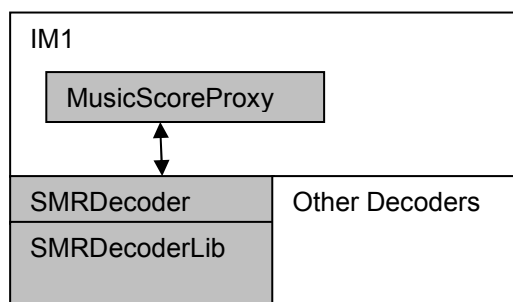


Figure D.1 — Structure of IM1 with SMR

In Figure D.2 a UML diagram describes the relations of the new classes with the IM1 existing classes.

The *ScoreShape* and *MusicScore* classes represent the data model, they have as attributes the fields defined in the nodes specification. The *MusicScore* class is derived from the *StreamConsumer* class since it has to present the data coming from SMR decoder. The *ScoreShapeProxy* and the *MusicScoreProxy* are used to render the nodes in the multimedia scene. The *MusicScoreProxy* class uses the *SMRDecoder* object to interact with the *Partitura* object that contains the music score coming from the SMR stream.

The *SMRDecoder* class derived from the *DecoderImp* class implements the interface to decode data coming from the SMR stream. The real decoding work is performed by the *SMRInternalDecoder* class that is a façade class defined in the strictly SMR related software that hides the interaction the SMR data structures (this was done to avoid to add to IM1 the dependency to SMR).

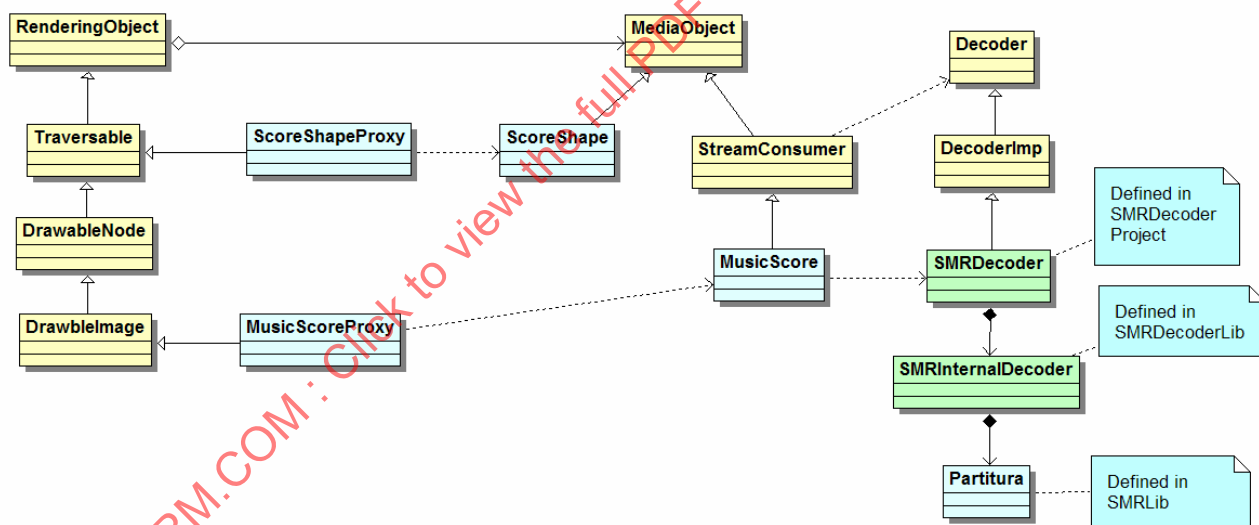


Figure D.2 — UML diagram of new classes

In Figure D.3 a snapshot of IM1 using *MusicScore* node is presented.

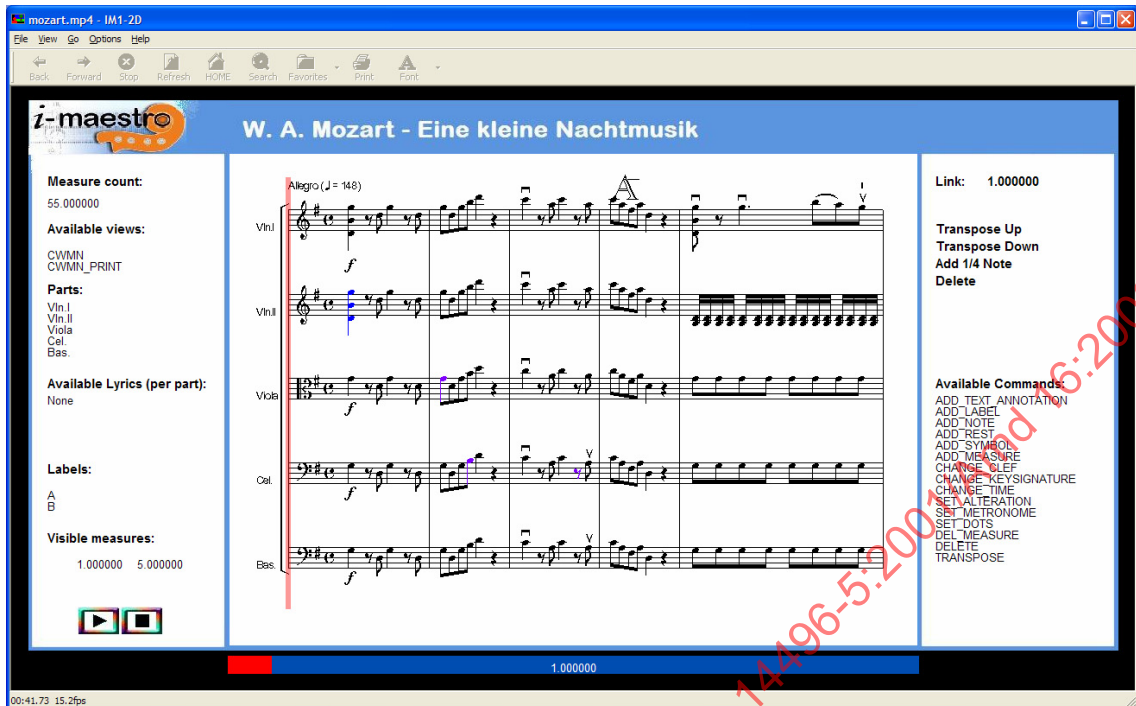


Figure D.3 — IM1 with SMR

## D.2 SMR source code

### D.2.1 Source files organization

In audio/SNHC/mpeg4smr is present the source code needed to build an SMR decoder to be used in IM1, a SMR viewer application and a SMR editor application.

| Location      | Content                                                                                                                                                                  |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| errmsg        | contains error messages used at runtime                                                                                                                                  |
| errmsg/xsd    | contains XML schemas used for validation                                                                                                                                 |
| font          | contains fonts, font tables and the SMFL rules file (smfl.xml) used for rendering the score                                                                              |
| lib           | contains the produced SMR library                                                                                                                                        |
| samples       | contains the source code for a music editor (wmusiceditor) and for a score viewer (wmusicviewer) using the SMR library (open the smr.dsw file with MS Visual Studio 6.0) |
| smrdecoderlib | contains the source code to implement the SMRDecoderLib used in IM1, it is built on top of the smr library                                                               |
| source        | contains the source code used to build the SMR library, open the smr.dsw with MS Visual Studio 6.0 to build the library                                                  |
| source/abb    | contains classes for ornaments (from italian abbellimenti)                                                                                                               |
| source/alt    | contains classes for accidentals (from italian alterazioni)                                                                                                              |

|                |                                                                                                    |
|----------------|----------------------------------------------------------------------------------------------------|
| source/aud     | contains interface classes to link with an audio player                                            |
| source/bat     | contains classes for measure representation (from italian battuta)                                 |
| source/bit     | contains classes for SMR bitstream management                                                      |
| source/bitmaps | contains bitmaps resources                                                                         |
| source/bra     | contains classes for brackets connecting scores in the main score                                  |
| source/drw     | contains classes for basic drawing                                                                 |
| source/fig     | contains classes for musical figures representation (notes, rests, chords, beams)                  |
| source/fon     | contains classes for font management                                                               |
| source/gui     | contains classes for high level gui                                                                |
| source/img     | contains classes for image management                                                              |
| source/imp     | contains classes for MIDI import                                                                   |
| source/ind     | contains classes for indication symbols (expression symbols, fingering, mute, accents, etc.)       |
| source/int     | contains classes for horizontal symbols or interval symbols like slurs, crescendo, diminuendo etc. |
| source/kor     | contains classes for Korean ornaments                                                              |
| source/lst     | contains a class to manage lists                                                                   |
| source/lyr     | contains classes for lyric management                                                              |
| source/mdo     | contains classes for MIDI production                                                               |
| source/mil     | contains classes for SMFL rules interpretation                                                     |
| source/net     | contains classes for network access                                                                |
| source/pag     | contains classes for page objects (textbox, imagebox)                                              |
| source/par     | contains classes for main score management (from italian partitura)                                |
| source/sca     | contains classes for jump symbols, metronome, labels (from italian scansione)                      |
| source/smr     | contain classes to manage SMR content                                                              |
| source/spa     | contains classes for single part management (from italian spartito)                                |
| source/str     | contains classes for instrumental symbols (from italian strumenti)                                 |
| source/txt     | contains classes for textual indication                                                            |
| source/var     | contains classes for duration variation (augmentation dots, corona)                                |
| source/xml     | contains classes for xml loading using sax and dom                                                 |



### D.2.2 Building dependencies

Other components needed to build the SMR library are:

- wxWindows v. 2.2.6 (<http://www.wxwindows.org>)
- xerces v. 2.6.0 (<http://xml.apache.org/dist/xerces-c>)

To build the library and the examples, use MS Visual Studio 6.0 and SET the following environment variables:

- WXWIN set to the path where the wxWindows home is (i.e. c:\wx2-226)
- XERCES set to the path where the xerces library is (i.e. c:\xerces-2\_6\_0)

### D.2.3 Runtime dependencies

The SMR Editor, SMR Viewer and SMRDecoderLib need the following registry keys (normally are set by the SMR Editor)

HKEY\_CURRENT\_USER\Software\SMR\paths

- errpath contains the errpath directory (important for validation)
- fontpath contains the font path
- imgscore not used
- midipath not used
- smrpath path where to start searching for SMR files

### D.2.4 SMR core classes

In Figures D.4 and D.5 some of the core classes defined to model the music score are reported

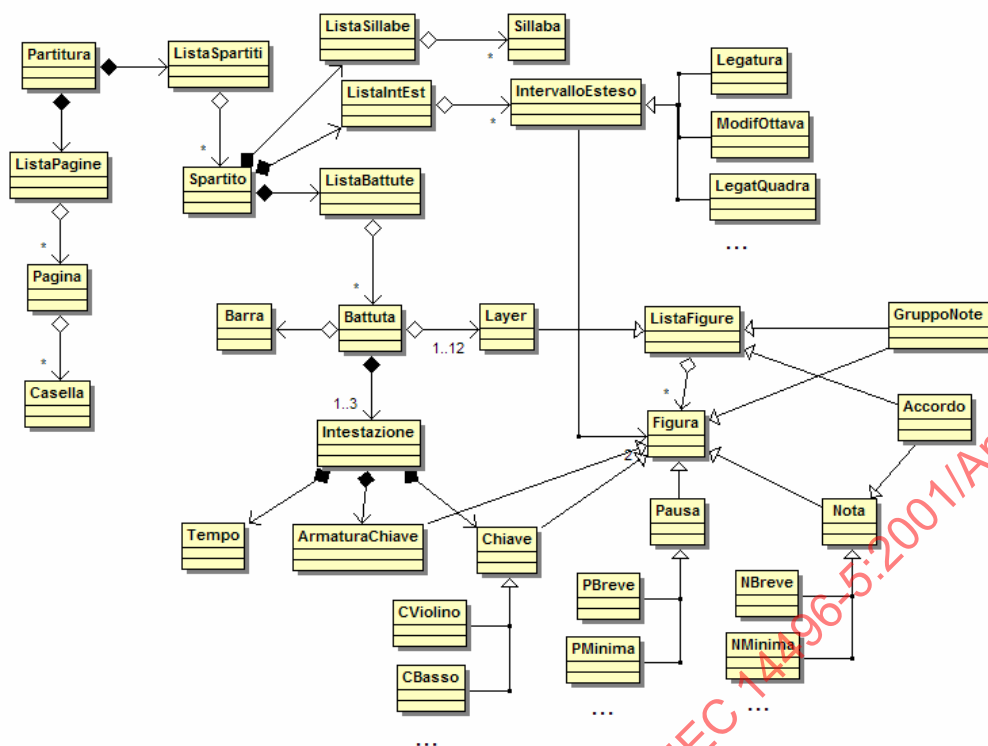


Figure D.4 — Core SMR classes, part 1

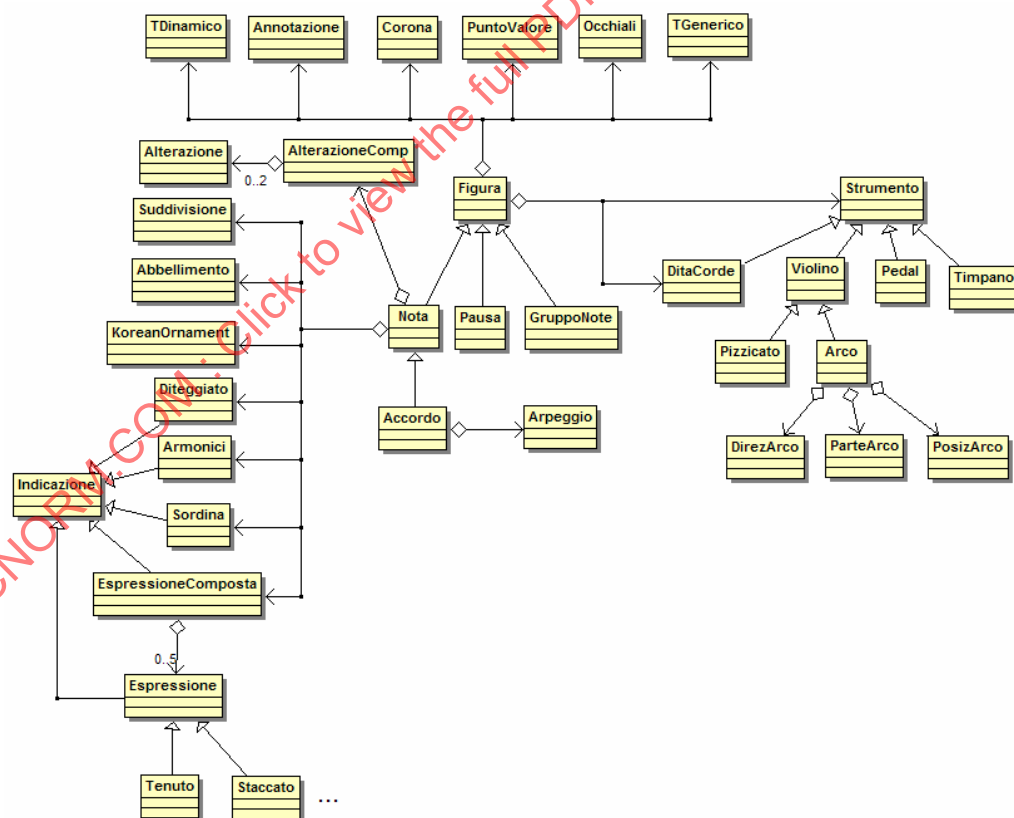


Figure D.5 — Core SMR classes, part 2

These classes support:

- score positioning and drawing (as main score or as a single part) the score positioning is performed using the SMFL rules (loaded from file);
- score editing;
- load and save as XML;
- MIDI generation;
- generation of MP4 files containing the plain XML, gzipped XML or BiM encoded XML;
- loading from MP4 files;

### D.3 SMR Music Editor

Using the SMR library a simple music editor has been implemented, in Figure D.6 the user interface is presented.

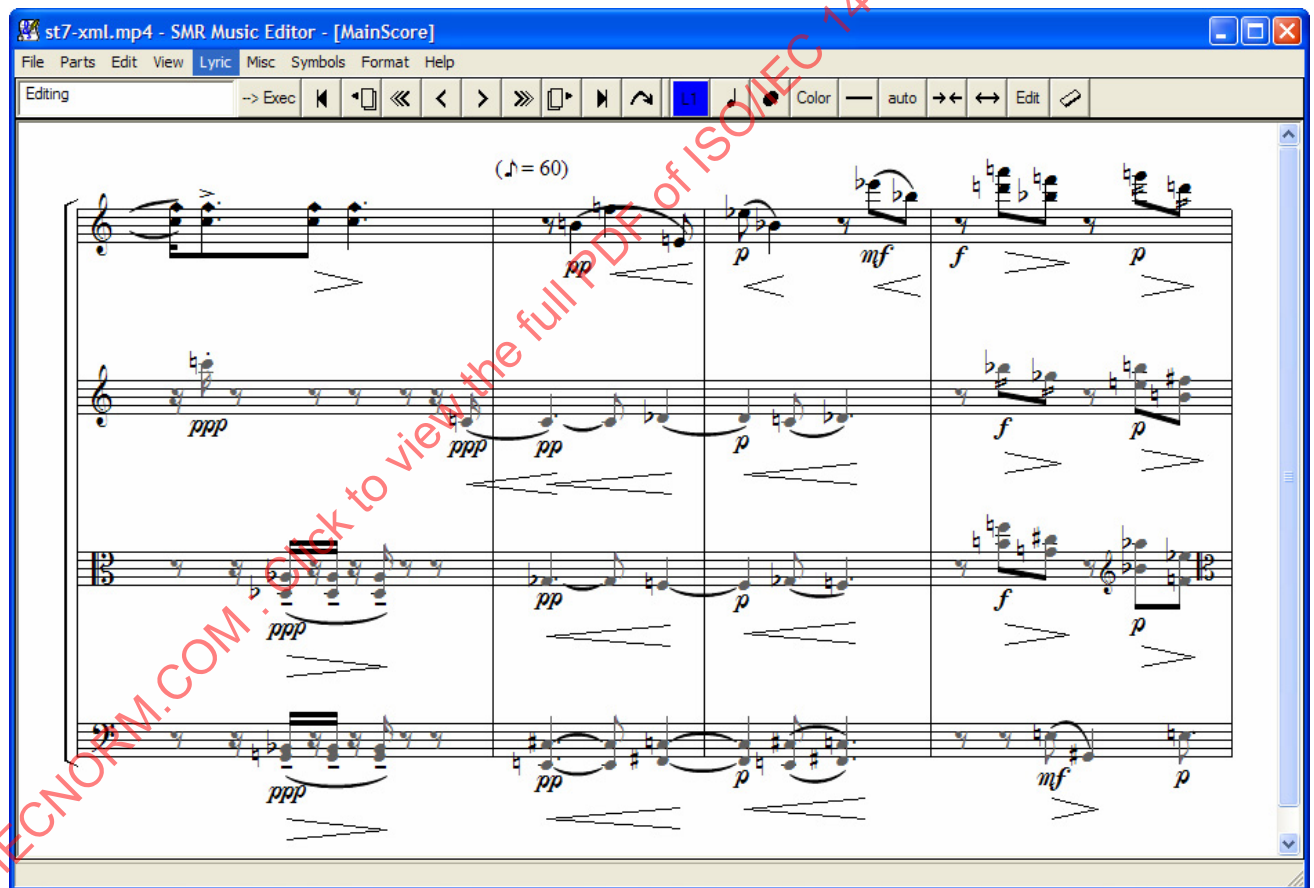


Figure D.6 — SMR Music Editor

The main functionalities provided are:

- create a new score from scratch;
- load and save from/to the XML format and MP4 files;