W3C Music Notation Community Group Meeting

MICHAEL GOOD
VICE PRESIDENT OF MUSICXML TECHNOLOGIES

APRIL 7, 2017

makemusic®

Agenda

- 2:30 – Introduction
- 2:35 – SMuFL 1.2 update
- 2:40 – MusicXML 3.1 discussion
- 3:05 – MNX discussion
- 4:20 – Encoding graphics and performance data
- 4:30 – Reception sponsored by Steinberg

steinberg

SMuFL 1.2

- Currently 78% complete – 33 of 42 issues intended for version 1.2 are closed
- Remaining issues are largely small additions to existing ranges
- Largest addition currently in scope is for Kahnotation for transcribing dance alongside music notation
- Largest proposal not currently in scope is for adding Spacialization Symbolic Music Notation (SSMN) to SMuFL – interested in discussion about this
- Are there any other needs not met by this release?

MusicXML 3.1

- What’s new in MusicXML 3.1
- Status of open issues
- Is the V3.1 milestone really what we want?
- Are there issues not in 3.1 that should be there?
- Currently 88% complete (60 of 68 issues)
- Timetable for release

What’s New in MusicXML 3.1 – Part 1

- Many more SMuFL symbols are now represented directly by MusicXML elements (e.g. <flip/>)
- Other symbols can be specified by SMuFL canonical glyph name in MusicXML extension elements, using the smufl attribute (e.g. <other-technical smufl=“brassValveTrill”/>)
- Unique id attribute for many elements
- Musical symbols interspersed in text using new <symbol> and <credit-symbol> elements
- Grace cue notes

What’s New in MusicXML 3.1 – Part 2

- Ties in metronome marks
- Image height and width for scaling
- New time-only attribute for <lyric> element
- New let-ring type value for <tied> element
- Non-unique measure number text
- More polygonal enclosure shapes
- Highest / lowest notes without leger lines
- Several documentation clarifications

Status of Open Issues

- Media type
  - Change from application/vnd/recordare/musicxml to application/musicxml?
- Uniform type identifier for macOS / iOS
  - org.w3c.musicxml or org.recordare.musicxml?
- Add a mimetype file to compressed MusicXML files for easier magic number detection?
- Change recommended file extension for uncompressed MusicXML files?
  - umxl? xmlm? mxlt? mxlu?
- Also finishing up three SMuFL range issues

What About Issues Not in 3.1?

- SMuFL ranges not in 3.1 that could be:
  - Plucked techniques
  - String techniques
  - Wind techniques
- OK to leave these to extension elements, most likely <other-technical>?
- Any important documentation issues?
- Anything else?

Timetable for Release

- We would like to have multiple implementations done before release
- Aim for end of Q2 / start of Q3?
- MakeMusic can add MusicXML 3.1 to our Dolet for Finale and Dolet for Sibelius plug-ins
- Who else would be interested in doing an early implementation to help find problems in these additions and changes?

MNX Update and Discussion

Joe Berkovitz (joe@noteflight.com)
Founder, Noteflight LLC
Co-chair, W3C Music Notation Community Group
W3C Advisory Committee Rep., Hal Leonard Corporation

Topics

- MNX Status
- Roadmap, Tradeoffs, Rationales
- Interoperability
- CSS
- Cursors, syntaxes, system notations
- Next steps
- Generalized notation

“Those […] who have no taste for this sort of speculative exercise will just have to stay in the trenches and do without it, while the rest of us risk embarrassing mistakes and have a lot of fun.”

–Daniel Dennett

“The world needs another notation standard like it needs another hole in the ozone layer.”

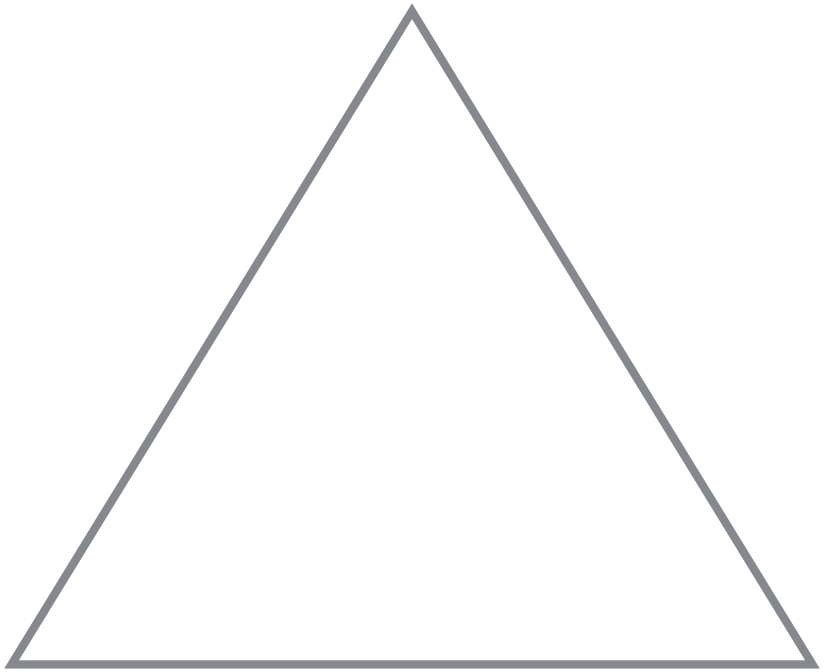
–*Anonymous Fan*

MNX Status

- Rough overview with lots of gaps
- A few examples
- Good food for thought
- Many important issues have been raised

Semantics, interoperability, generality: pick any two!

+strong semantics
+interoperable
-less general

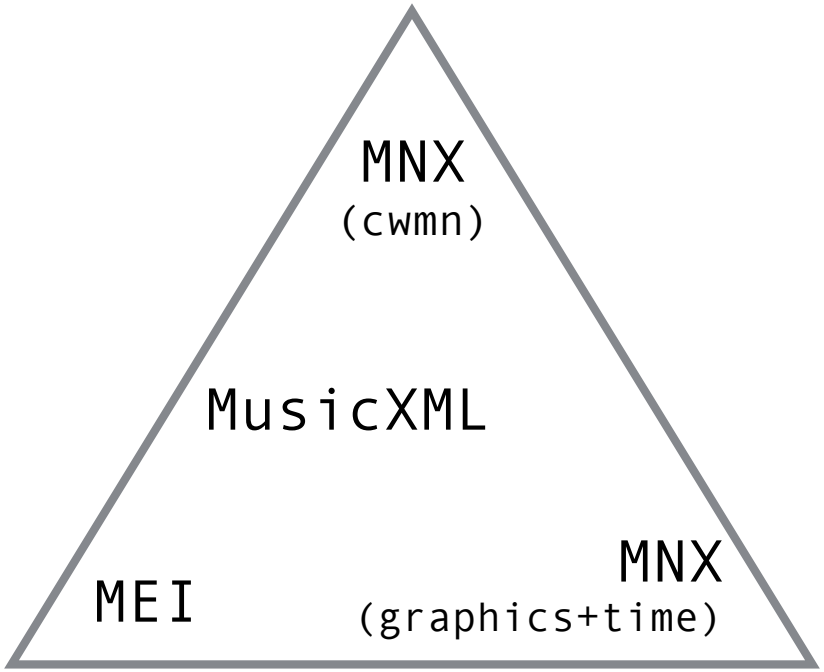


+strong semantics
+more general
-less interoperable

+very general
+interoperable
-less semantics

Fundamental Tradeoffs

+strong semantics
+interoperable
-less general



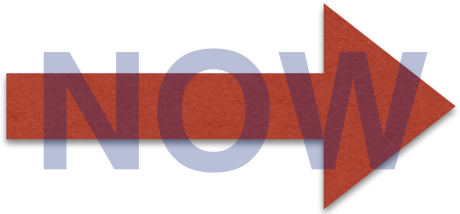
+strong semantics
+more general
-less interoperable

+very general
+interoperable
-less semantics

The Roadmap:
2 approaches to encoding

+strong semantics
-less general

Idiomatic CWMN



MNX (cwmn)

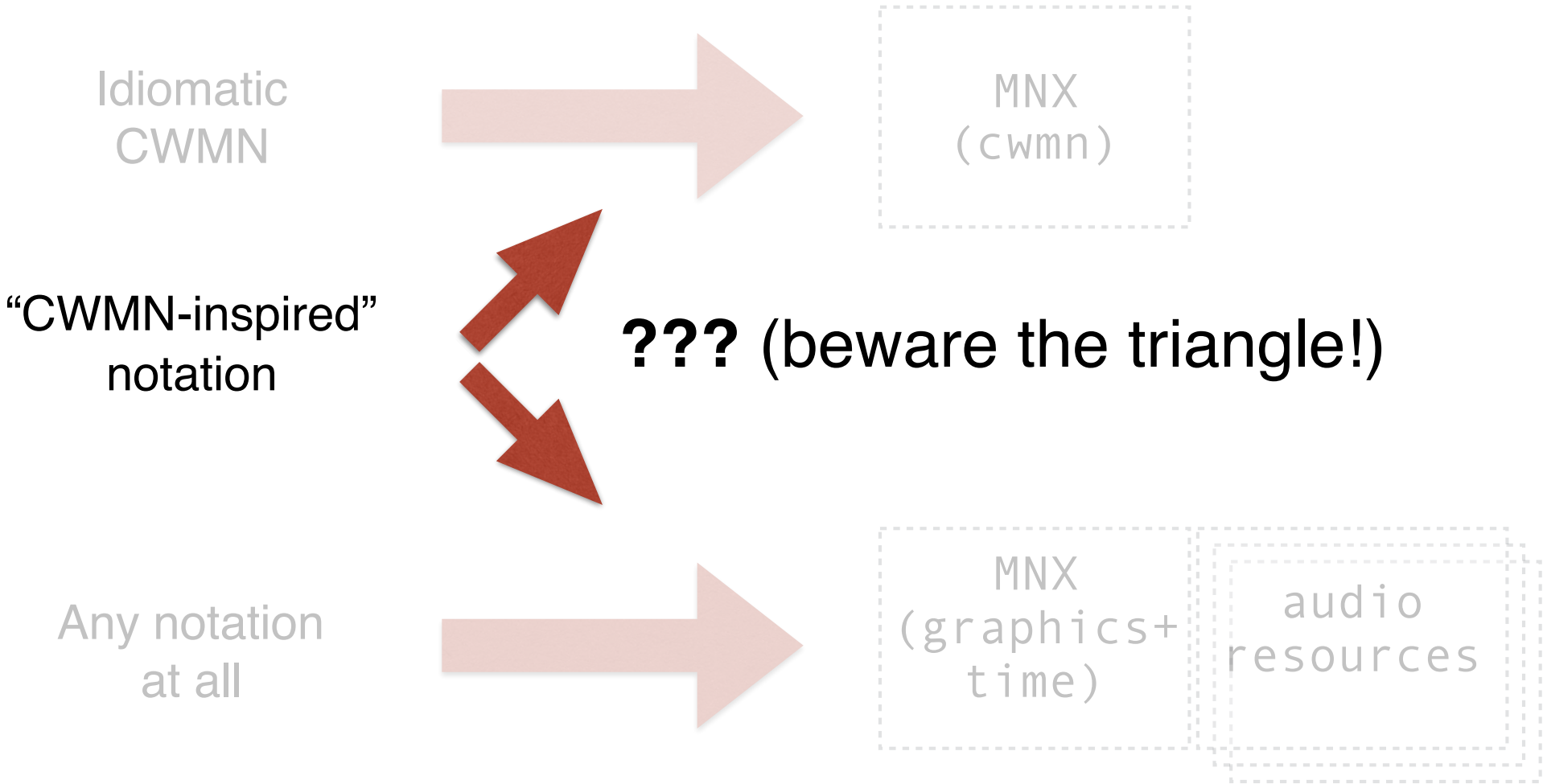
+very general
-less semantics

Any notation at all

MNX (graphics+ time)

audio resources

The middle ground



The middle ground

+strong semantics
-less general

Idiomatic CWMN

MNX (cwmn)

"CWMN-inspired" notation

preserve interoperability

+very general
-less semantics

Any notation at all

MNX (graphics+ time)

audio resources

Why another CWMN encoding?

- Strong interoperability for expanded set of use cases
- Let developers build simpler, smaller apps
- Standardize agreed subset of CWMN
- Clearly separate semantics, appearance, interpretation
- Make structure constrain content, naturally
- More clarity/terseness, meaningful container elements, ease of parsing
- Leverage existing MusicXML adoption

Achieving Interoperability

- MNX profiles and content types…
  - save us from "must be able to do anything"
  - establish core features
  - establish core usages
- MNX CWMN rendering model…
  - must define key aspects of appearance
  - defines glyph registration

CSS: The good

- It’s not about the web, really!
- Separates semantics from appearance, interpretation
- Supports simple, modular rules that apply properties to elements and their descendants
- Known, established syntax
- Orthogonal to XML structure, so easy to ignore
- Parsing libraries available

CSS for Interpretation

- Styling can affect performance interpretation, not just appearance
- Replaces various sound-related elements in MusicXML
- Permits “MIDI-like” specification of arbitrary note events to be played back for any given element in MNX

CSS: The not-as-good

- One more syntax you have to parse
- Maybe better for publishing than exchange
- Selector syntax is pretty open-ended, may need restriction (via profiles?)
- alternatives…
  - MusicXML-style attribute mix
  - Separate styling into dedicated `<appearance>` and `<interpretation>` elements

Compact Syntaxes

- The proposal includes some compact syntaxes like `Bb7 and 8*`
- Can be used across the board as XML attributes, elements or CSS properties
- Readability and compactness are good: documentation, hand encoding and debugging will never die!

Cursors and Positioning

- How can we position directions in arbitrary places? Some choices:
  - Require a position for every event/direction
  - Allow position specification, for directions only
  - Allow the cursor to be advanced with empty elements
  - Use `offset` from cursor, like MusicXML

System Notations

- We need a way to describe directions that belong to all parts in the system
- Alternatives:
  - A `<system>` “pseudo-part”
  - A means of tagging such directions, when they occur within some part

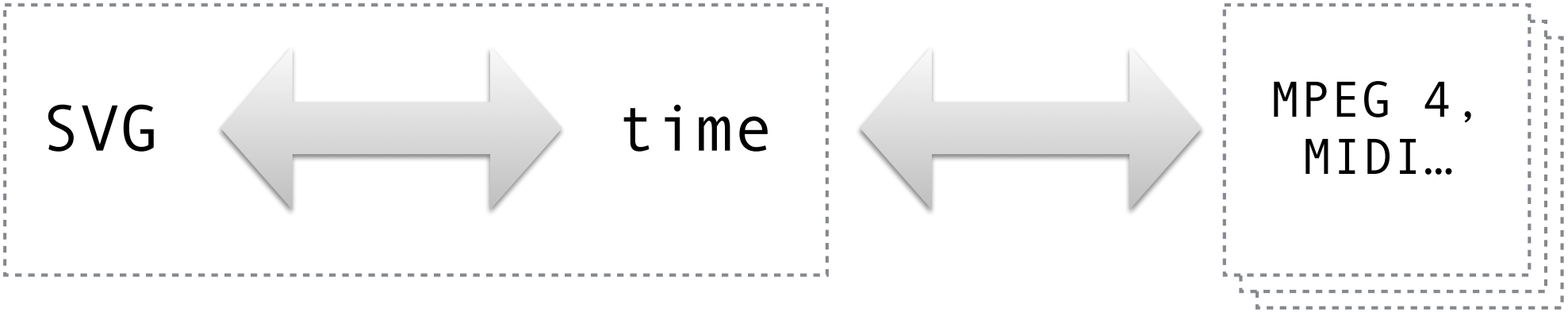
MNX: next steps

- apply course corrections
- more MNX examples
- create and approve roadmap document
- establish "beachhead" specs for MNX(cwmn) elements, style properties
- begin reference implementations
- continue open design discussion of MNX(graphics+time) in background

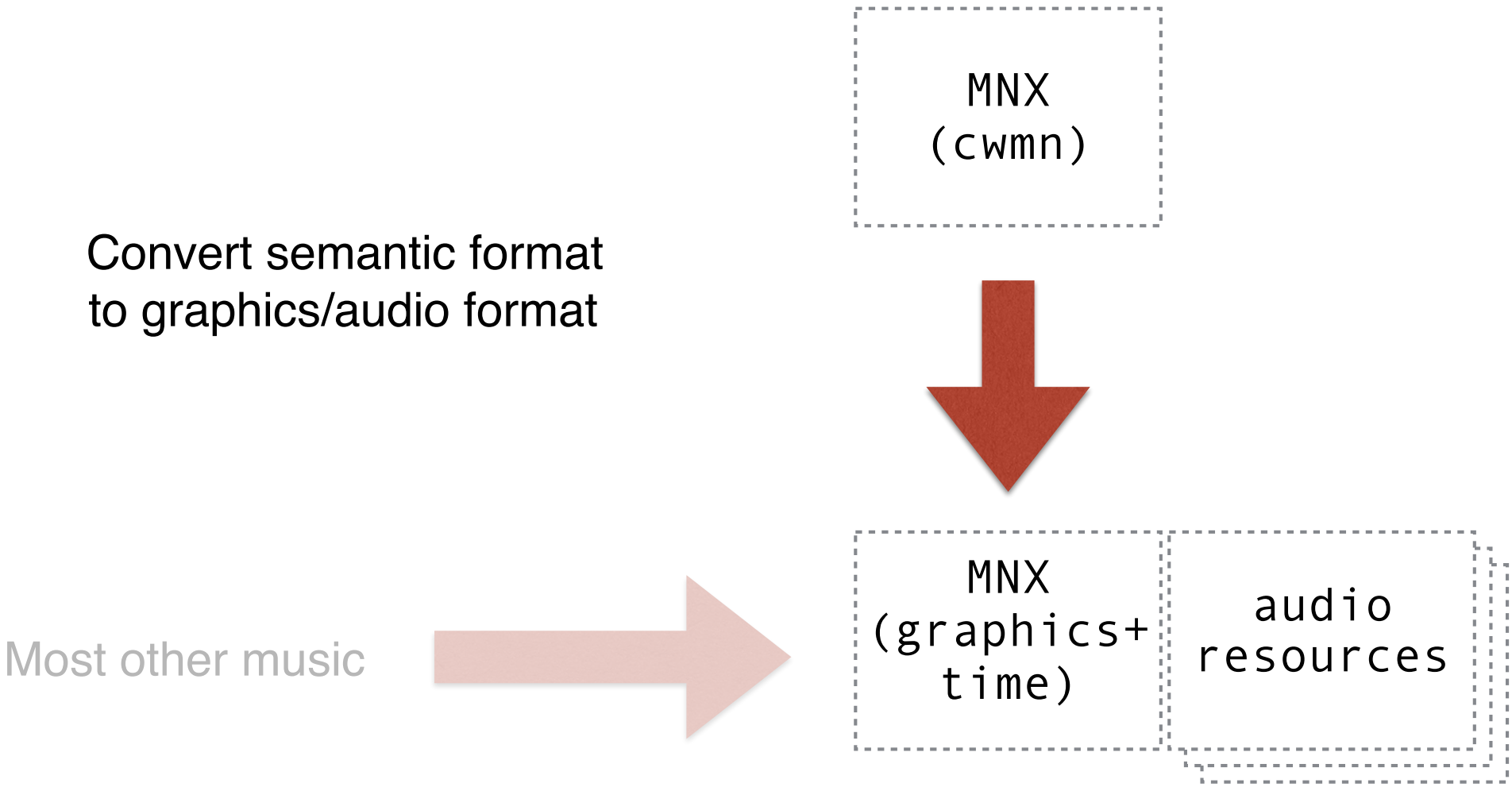
What's another take on a "general" approach?

MNX (graphics+time) format

standard audio resources:



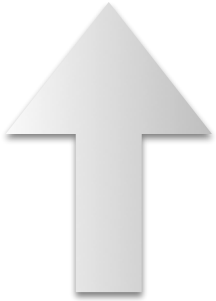
Possibility: convert CWMN for non-semantic clients



Connecting graphics, audio with original semantics

MNX
(cwmn)

references from rendering
point back to elements
in semantic document



MNX
(graphics+
time)