Auf dem Weg zu einem universellen Austauschformat für Musiknotation

Dr. Edgar Huckert
Prof. Dr. Klaus Huckert
Stand 07-2007

Inhalt:

1 Einleitung

Im Laufe der Musikgeschichte hat sich im europäischen Bereich die übliche graphische Musiknotation entwickelt. Ihre erste Ausprägung wird Guido von Arezzo im Italien des 10./11. Jahrhunderts zugeschrieben. Der folgende Aufsatz konzentriert sich auf die Anforderungen an ein Musikformat, das sich per Computer verarbeiten lässt und erläutert die Vor- und Nachteile einiger gängiger Ansätze. Dabei wird vor allem der Einsatz von XML als Grundlage eines austauschbaren Musikformats diskutiert. XML wurde in den neunziger Jahren als Metaauszeichnungssprache in der Informatik entwickelt. Es deutet sich an, dass eine Variante von XML – MusicXML – weltweit von Herstellern und Benutzern als ein Standard akzeptiert wird. Damit besteht Hoffnung, dass in Zukunft Musikdokumente (Partituren) ohne Rücksicht auf das Erzeugerprogramm (z.B. Sibelius, Finale oder Capella) ausgetauscht werden können.

Drei Varianten von notationsnahen Formaten - ABC, MusicXML und MIDI - werden hier näher untersucht. Dabei wird diskutiert, warum die jeweilige Variante mehr oder weniger als Grundlage für ein universelles Notationsformat brauchbar ist.

Ein auf XML aufbauendes Notationsformat eignet sich auch als Grundlage für computergestützte musikalische Analysen. Wir zeigen in einem weiteren Aufsatz wie mit Hilfe von XML beispielsweise Akkordanalysen im Jazz wirkungsvoll unterstützt werden können.

2 Geschichtlicher Hintergrund

Musiknotation bedeutet: graphisches Festhalten von Größen wie etwa der Tonhöhe, der Tondauer und Lautstärke in einer speziellen Notenschrift**.** Die üblicherweise verwendete klassische Notation ist eigentlich eine Mischung aus graphischen Bestandteilen (Linien, Notenköpfe, Fahnen etc.) und textuellen Bestandtteilen (z.B. für Tempobezeichnungen, Akkordsymbole). Daneben gibt es auch alternative graphische Notationen wie z.B. die Gitarrentabulaturen oder Neumen (in gregorianischen Chorälen), die wir hier nicht betrachten.

Die in der westlichen Welt vorhandenen Notationsformate sind meist am Format klassischer Partituren orientiert. Klassische Partituren verwenden Konzepte wie:

- System (Anordnung von Zeilen)
- mehrere Stimmen in einem System
- Schlüssel (Violinschlüssel, Bassschlüsel, Bratschenschlüssel, Tenorschlüssel)
- Taktarten (3/4, 4/4, 6/8 etc.)
- zeilenbezogene Vorzeichen (Grundtonart pro Zeile)
- notenbezogene Vorzeichen (#, b, Auflösungszeichen)
- klassische Notendauer (Ganze, Halbe, Achtel, Sechzehntel)
- Modifikation der Notendauer (Punkt hinter Note, Triolenklammer)
- Verzierungen (z.B. Mordent)

- Metasymbole für Dynamik (forte, piano, crescendo)
- Metasymbole als Vortragsvorschriften (ritardando, Fermate)
- Schleifensymbole (Wiederholungszeichen, 1. und 2. Schluss)
- Tempoangaben in Textform (z.B. "Andante")
- Allgemeine Ausführungserläuterungen (z.B. "moriendo").

Dies alles und einiges mehr muss also ein Notationsformat festhalten können. Die derzeitig führenden Notationseditoren leisten dies - wenn auch nicht immer ganz einfach und ohne gelegentliche Probleme.

3 Wozu ein universelles elektronisches Notationsformat?

Es gibt nicht viele **allgemein akzeptierte elektronische Austauschformate**. Das bekannteste ist wahrscheinlich EDI/EDIFact zum Austausch von Rechnungsdaten im kommerziellen Umfeld. Jeder Musikeditor benutzt natürlich ein **internes Notationsformat** (proprietäres Format) und ein Ablageformat, deren Strukturen allerdings meist nicht offengelegt sind. Im musikalischen Umfeld ist ein elektronisches Austauschformat in den folgenden Situationen erforderlich:

- Musikstücke sollen gemeinsam bearbeitet (z.B. arrangiert) werden
- Musikstücke sollen auf unterschiedlichen Geräten angezeigt oder gedruckt werden ("Rendering")
- Musikstücke sollen zukunftssicher archiviert werden
- Musikstücke sollen produziert (veröffentlicht) werden
- Musikstücke sollen wiedergegeben (abgespielt) werden
- Musikstücke sollen elektronisch analysiert werden.

Es fällt wahrscheinlich auf, dass eine Situation hier weggelassen wurde: ein Musikstück wird erstellt (komponiert). Ähnlich wie in der Textverarbeitung - die Musikbearbeitung hat sehr viele Parallelen zur Textverarbeitung - kann man realistischerweise nicht davon ausgehen, dass die im Markt verbreiteten Notationseditoren wie Capella, Sibelius, Finale etc. als internes Format oder als Ablageformat ein universelles Notationsformat wählen werden. Dies mag viele musikalisch begründbare Ursachen haben - auf jeden Fall sind die kommerziellen Gründe nachvollziehbar. Um so wichtiger aus Sicht der Anwender ist dann allerdings die Forderung nach Konvertierbarkeit in ein universelles Austauschformat. Derzeit ist man häufig mit der unwürdigen Situation konfrontiert, dass entweder Produzent und Empfänger die gleichen Notationsprogramme besitzen müssen oder dass man sich auf den kleinsten Nenner - häufig MIDI - einigen muss, was zu hohem Nachbearbeitungsaufwand führt.

4 Anforderungen an ein universelles Notationsformat

Die Anforderungen an ein Notationsformat für die Verarbeitung von Musik ähneln stark den gängigen Anforderungen an Formate für die Textverarbeitung. Demnach müsste ein wirklich vollständiges Format diese Aspekte abdecken:

- die **logische Form** muss dargestellt werden können. Ähnlich wie ein Text in Kapitel, Abschnitte, Index, Vorwort etc. gegliedert ist, besteht ein größeres Musikstück aus Sätzen und jeder Satz wiederum aus Einleitung und Ende (Coda). Zur logischen Form gehören auch eher seltene Komponenten wie Aufführungsmaterial, Varianten (z.B. verworfene Passagen). Die logische Form kann sehr oft in grammatikähnlichen Notationen abgelegt werden, z.B. in DTDs (aus der SGML Welt) oder in XML-Schemas.

- das **Layout** muss dargestellt werden können. Dazu zählen Konzepte wie der Seitenaufbau einer Partitur (Deckblatt, Inhaltsseiten), der Aufbau einer Seite (Ränder, Seitennummerierung) eines Systems (Anzahl Zeilen, Zeilenabstände) oder einer Zeile (Anzahl Linien). Das Layout wird in der Textverarbeitung und in vielen Notationsprogrammen als "Vorlage" (Musterdokument ohne Inhalte) abgelegt.

- der **musikalische Inhalt** muss korrekt erfassbar sein. Bei Texten hat man es mit Zeichen, dargestellt als Zeichencodes (ASCII, ANSI, UTF8 etc.), zu tun. Bei Musikstücken müssen Tonhöhe, Tondauer, zeitliche Abfolge etc. korrekt dargestellt werden. Erschwerend kommt bei Musikdarstellung die Ebene der Metainformationen hinzu (z.B. Dynamikanweisungen), die kaum Parallelen in der Textverarbeitung aufweist.

- die **Formatierung des Inhalts** muss darstellbar sein. Bei einer Textverarbeitung ordnet man den Textzeichen Attribute zu wie Fontangabe, Schriftschnitt, Schriftabstände. In der Darstellung von Musik müssen Formatierungen für Balken (Einzelnoten, lange Balkenfolgen), für Stichnoten, für Verzierungen etc. darstellbar sein.

- die **Aufbereitung** für die graphische Darstellung ("final form", "Rendering" für Druckausgabe, Ausgabe am Schirm) muss angebbar sein. Hier kann man sehr oft auf die gleichen Konzepte wie in der Textverarbeitung zurückgreifen, z.B. auf CSS ("cascaded style sheets") oder Postscript-Makros.

Hinzu kommen weitere, musikspezifische Forderungen:

- die **musikalische Aufführung** über elektronische Klangerzeuger (z.B. Synthesizer) sollte möglich sein - vielleicht auch nur im Sinne eines akustischen "preview", d.h. eines ersten Eindrucks darüber, wie die Musik klingt.

- die **Auswertung per EDV** (z.B. für musikwissenschaftliche Zwecke) muss möglich sein. Musikalische Strukturen müssen sich leicht per Computer auffinden und analysieren lassen.

Und schließlich die wichtigste Forderung für ein universelles Austauschformat:

- die Formatdefinition muss **publiziert, allgemein zugänglich** und im öffentlichen Besitz sein.

Nach diesen hehren Zielen sollte man einige viel praktischere und deswegen u.U. viel wichtigere Ziele nicht vergessen: jedes Notationsformat - ob universell oder nicht - muss die **alltäglichen Aufgaben** ermöglichen, z.B.:

- Erfassen eines Musikstücks
- Korrektur eines Musikstücks
- Erstellen von Auszügen (z.B. Stimmauszüge)
- Transponieren (Ändern der Tonart oder der Stimmlage)
- Abspielen (zumindest Vorhören).

Das bedeutet konkret: jedes künftige universelle Austauschformat muss prinzipiell auch als "native format" (proprietäres Format) eines Notationseditors einsetzbar sein.

Es gibt sehr viele Ansätze für EDV-orientierte Musikformate. Viele der nachstehend genannten Formate haben sich nicht durchgesetzt und sind im wesentlichen nur Spezialisten bekannt. Gerd Castan listet auf seiner Webseite www.music-notation.info viele dieser Ansätze auf:

XML	ASCII	Binär-Format
SMDL	PDF	MIDI
MNML	DARMS	RMTF
MusicML	GUIDO	Sibelius
MHTML	ABC	Finale
MML	MusiXTeX	XMF
Theta	Kern	NIFF
ScoreML	Hildegard	Capella
JscoreML	Koto	SASL
XScore	Bol	Score
MusiXML	Musedata	RMF
MusiqueXML	LilyPond	SSS
GUIDO XML	PMW	Tilia
WEDELMUSIC	TexTab	PowerTab
ChordML	Mup	
ChordQL	NoteEdit	
NeumesXML	Liszt	
MEI	ETF	
JMSL Score	CMN	
Minimusic	OMNL	
EMNML	Score	
XMusic	PMX	
SongWrite	Nightingale	
MDL	NeXT	
VMML	CHARM	
MusiCat	Muscript	
CapXML	Gaayaka	
SVG	Plaine and Easie	
SMIL	Clan	
	Parsons	

Prinzipiell lassen sich je nach gewünschtem Ziel unterschiedliche Schwerpunkte bei der Festlegung eines Notationsformat begründen:

- soll ein musikalischer Entwurf (eine entstehende Komposition) festgehalten oder ausgetauscht werden?
- soll ein reales Musikereignis festgehalten werden?
- orientiere ich mich am graphischen klassischen Notenbild?
- will ich eher Klänge beschreiben?

5 Orientierung am Notenbild

Orientierung am Notenbild könnte heißen: betrachten wir ein Musikstück als (gescanntes) Bild. In der EDV wird die Graphikverarbeitung in pixelorientierte Verarbeitung ("Bildverarbeitung") und vektororientierte Verarbeitung (z.B. CAD/CAM) aufgetrennt. Das Problem bei der Verarbeitung von pixelorientierten Informationen ist eigentlich immer das gleiche: es gibt zuviele Missinterpretationen beim Verarbeiten der graphischen Notation, da deren Grundelemente (etwa die Notenlinien) zunächst nur als Pixel (Bildpunkte) vorliegen - weitergehende Informationen (welche Punkte gehören auf die Linie?) müssen erst im Computer erzeugt werden. Das Problem ist aus der Textverarbeitung bekannt: selbst nach mehr als 30 Jahren Tradition für OCR arbeitet diese Technik noch nicht problemfrei.

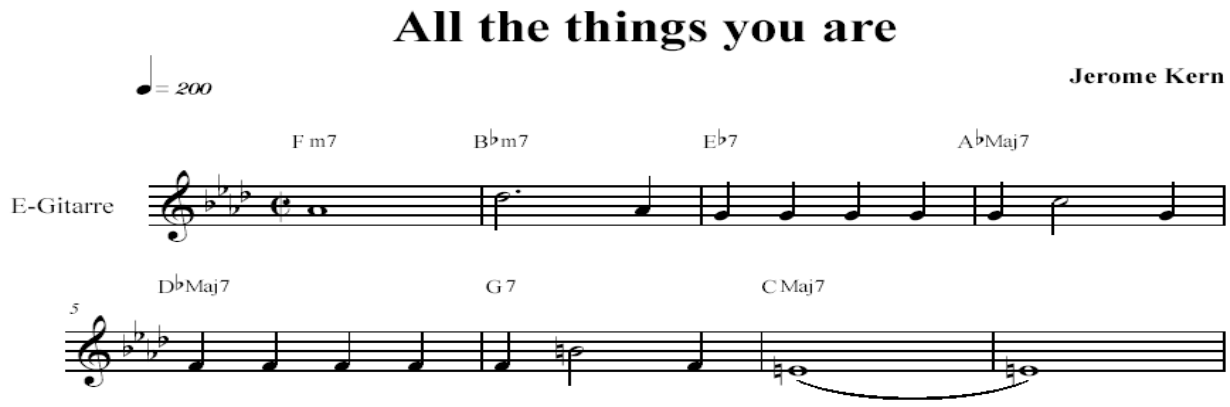
Jeder Informatiker weiß nun, dass höhere Organisationsformen für Graphikelemente (z.B. Vektoren) als Grundlage der Weiterverarbeitung besser geeignet sind als Pixel. Menschen erfassen solche höheren Organisationsformen (also z.B. Notenlinien) intuitiv - Computer haben damit ihre Probleme. Eine wesentliche Forderung an ein Musikformat besteht also darin, über den reinen Bildbereich hinauszugehen und möglichst viel bereits bekannte höhere Interpretation unterzubringen.

Ein am Notenbild orientiertes Notationsformat muss also höhere Graphikkonzepte enthalten. Dies soll nicht bedeuten, dass eine Graphiksprache wie z.B. GKS oder SVG als Notation verwendet wird - obwohl diese Sprachen als Bestandteile eines Notationsformats denkbar sind.

5. 1 Das Format ABC

Das Format ABC wurde vom englischen Mathematiker und Musiker Chris Walshaw entwickelt, in erster Linie, um traditionelle einstimmige Melodien aus Westeuropa zu notieren. 1991 erfolgte erstmals die Publikation des Formates. ABC wurde danach entscheidend weiterentwickelt. So kann man die ABC-Musiknotation auch dazu einsetzen, um komplexere klassische Kompositionen zu notieren. Das Format zählt zu den ASCII-basierten Notationsformaten. Es orientiert sich allerdings am Notenbild und wird deshalb parallel zu MusicXML behandelt.

Das folgende notierte Beispiel wird durchgehend in diesem Aufsatz verwendet:



In ABC wird dieser Teil eines Jazz-Standards wie folgt notiert:

```
C:JeromeKern
T:All The Things You Are
L:1/4
M:4/4
K:Ab
Q:120
V:M clef=treble
[V:M]A4| D3A| GGGG| GC2G|FFFF| F^B2F| ^E4| ^E4|
```

Die ersten 7 Zeilen enthalten **Metainformationen**. Es bedeuten:

```
C: Komponist
T: Titel
L: Grundeinheit Viertel
M: Takt 4/4
K: Tonart
Q: Tempo in BPM
V:M Benamung eines Systems
```

Die letzte Zeile ("[V:M]...") enthält die ersten 8 Takte der Melodie von "All The Things You Are". Die senkrechten Striche ("|") sind Taktenden. Die Notennamen entsprechen den in Deutschland üblichen. Die Zahlen ("4, 3, 2") sind Multiplikatoren der unter "L:" genannten Grundeinheit - hier also von Vierteln. Das Zeichen **^** steht für "Erhöhung um einen Halbton".

In ABC sind als Vorzüge die Übersichtlichkeit, der Klartext und die kompakte Darstellung zu nennen. Zudem existieren einige ABC-Interpreter, die den Austausch in andere Formate ermöglichen.

ABC eignet sich zum Festhalten von Musikstücken, wenn keine graphische Darstellung möglich ist. In der EDV kann das Format dazu benutzt werden, Mustersuche auf einfache Art zu betreiben: als Suche über textuellen Informationseinheiten (z.B. mittels regulärer Ausdrücke).

ABC hat sich nicht wirklich durchgesetzt. Die Gründe dafür sind wahrscheinlich die folgenden:

- ABC wurde bisher vorwiegend im Folklore-Bereich eingesetzt: dort sind die aufgezeichneten Musikstücke wenig komplex. Die überzeugenden komplexen Beispiele fehlen.
- Die wenigen vorhandenen komplexeren Beispiele sind letztlich doch nicht einfach zu überschauen.
- ABC kann Vererbungsbeziehungen (dazu weiter unten) kaum darstellen
- ABC ist nur locker beschrieben und normiert. Selbst beim derzeitigen geringen Verbreitungsgrad existieren Varianten.
- Es gibt nur wenige Konverter, die ABC in andere Formate umwandeln und umgekehrt
- ABC erschien zu einer Zeit auf der Szene, als die ersten brauchbaren graphischen Musikeditoren erschienen.

5.2 XML und MusicXML

XML ist die Abkürzung für **Extensible Markup Language**. XML dient u.a. dazu, Dokumente auszuzeichnen, d.h. die Informationen (Inhalte, Strukturen) innerhalb von Dokumenten durch sogenannte Tags (Etikett, Schildchen) zu markieren. Alle Daten innerhalb von XML-Dokumenten müssen als Klartext-Daten (Strings) dargestellt werden. XML ist eine Meta-Auszeichnungssprache, d.h. XML besitzt keine fest definierte Menge von Tags. Es erlaubt vielmehr, für einen spezielle Zwecke (z.B. Musik, Mathematik, E-Business, Chemie etc.) eine Menge von Tags festzulegen, deren Beziehung untereinander durch eine Syntaxbescheibung (eine Grammatik) festgelegt wird. XML ermöglicht es damit, Daten bzw. Dokumente so zu beschreiben und zu strukturieren, daß sie zwischen einer Vielzahl von Anwendungen (Internet, in der Musik z.B. zwischen verschiedenen Notations-/Scan-/Midi-Programmen) ausgetauscht und weiterverarbeitet werden können. Die bereits erwähnte Syntaxbeschreibung (Grammatik) für eine konkrete Dokumentklasse wird durch eine Dokument-Typ-Definition (DTD, noch aus der SGML Welt stammend) oder durch ein XML-Schema festgelegt.

Geht man von der graphischen Notation (dem Notenbild im obigen Beispiel) aus, dann kann man die folgenden Informationseinheiten erkennen:

- Titel des Stückes
- Name Komponist
- Metronomangabe
- Stimmenbezeichnung/Instrument
- Notenzeile
- Notenschlüssel
- Tonart
- Taktart
- Takte.

XML ist in der Tat gut für die Darstellung musikalischer Informationen der gezeigten Art geeignet, weil es die genannten musikalischen Konzepte darstellen und insbesondere die dazwischen bestehenden Zusammenhänge darstellen kann. Dazu zählen insbesondere die hierarchischen Beziehungen.

Hierarchische Beziehungen werden in der objektorientierten Sprechweise der Informatik als Vererbungsbeziehungen bezeichnet. In der Tat lassen sich solche Vererbungsbeziehungen schon an diesem Beispiel aufzählen:

- der Titel gilt das ganze Stück
- der Schlüssel gilt für alle Zeilen und Takte
- die Tempobezeichnung gilt für alle Zeilen und Takte
- die Tonart gilt für alle Zeilen und Takte
- die Taktart gilt für alle Zeilen und Takte.

Man kann in objektorientierter Sprechweise beispielsweise sagen: die Takte des obigen Beispiels erben Schlüssel, Tonart und Taktart.

Neben den hierarchischen Beziehungen gibt es in der Musik und in der üblichen Musiknotation noch die besonders wichtige **linear - zeitliche** Beziehung: Noten und Akkorde folgen einander zeitlich mit definierten Abständen. Sie sind zeitlich geordnet. Üblicherweise stellt man diese Ordnung durch eine graphische Links-Rechts Beziehung dar. In XML kann man sowohl ungeordnete Mengen (ein Knoten kann beliebig viele Unterknoten haben, deren Reihenfolge nicht zwingend festgelegt ist) als auch Folgen (geordnete Mengen) wiedergeben, wobei natürlich nur dieser Aspekt (Notenereignisse sind linear geordnet) für die Musikwiedergabe wesentlich ist.

Neben der Wiedergabe von hierarchischen und linearen Beziehungen gibt es eine dritte Eigenschaft, die XML für die Wiedergabe von Musikereignissen besonders geeignet erscheinen lässt: XML-Knoten können **Attribute** enthalten, deren Zahl und Typ frei definierbar ist. Für einen Knoten des (frei definierbaren) Typs "Note" lassen sich z.B. die folgenden Attribute definieren:

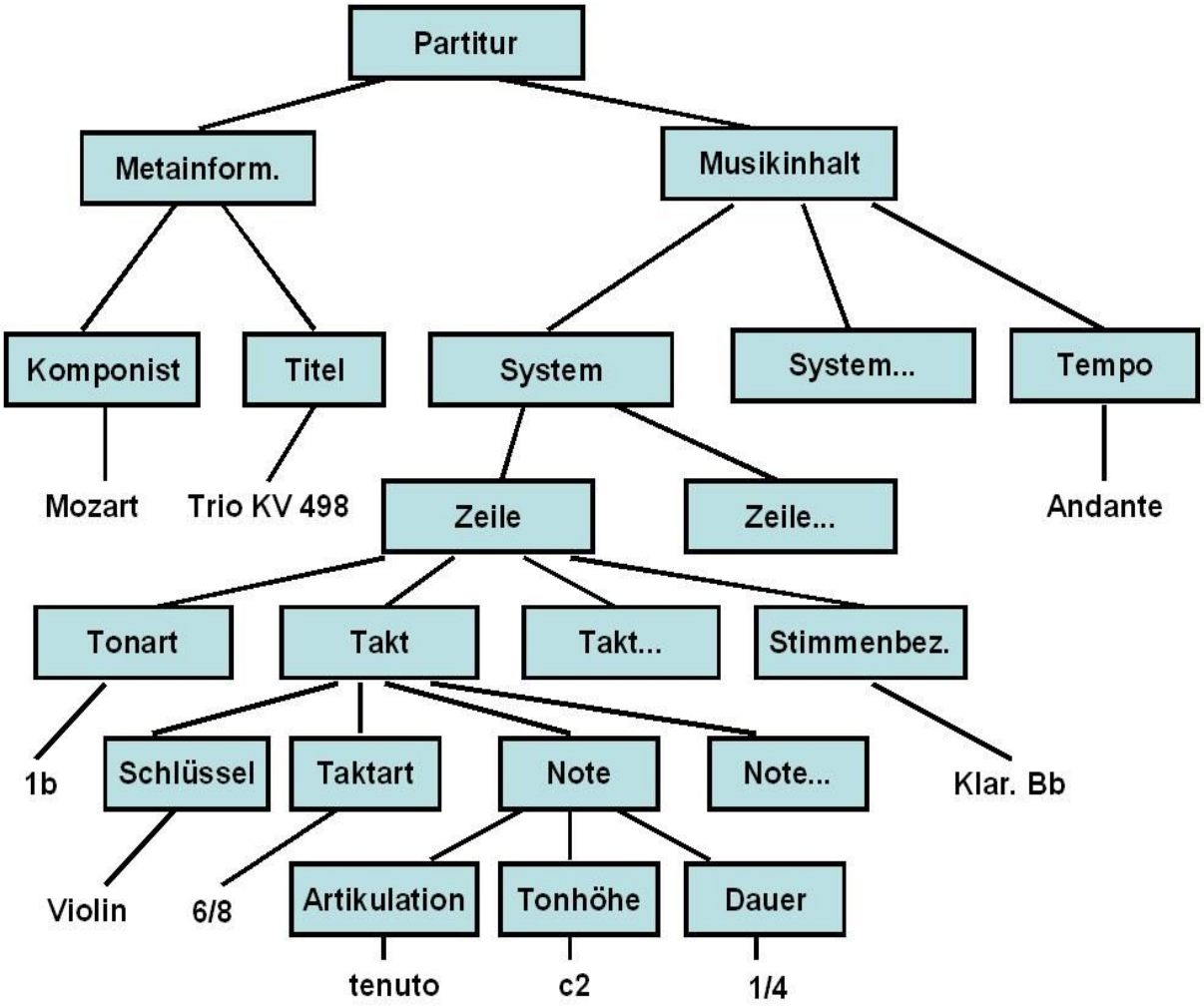
- Tonhöhe
- Name
- Dauer (z.B. absolut Millisekunden oder relativ in 64tel)
- Vortragsart (legato, staccato, akzentuiert).

In XML kann man dies z.B. so ausdrücken

```
TON name="c" hoehe="2" dauer="16/64" vortrag="staccato"
```

Technisch gesehen könnte man auch ohne solche Attribute auskommen - das würde jedoch zu einer extrem großen und deshalb unintuitiven Anzahl von Knotentypen führen. Gerade die XML Attribute führen zu einer relativ kompakten und intuitiven Darstellung.

Einen typischen Auszug aus einem Baum als Darstellung für den Anfang einer kleinen Partitur zeigt die nachstehende Illustration:



Es wird zwar allgemein von **einem** XML-Dokument gesprochen, in der Regel braucht man allerdings mindestens **zwei Dateien**, um ein XML-Dokument vernünftig verarbeiten zu können. Diese sind:

- das **XML-Dokument**, das die Dokumentdaten enthält. In unserem Fall wären dies alle musikalischen Daten (die Noten), die Ausführungsvorschriften (Tempo etc.) und die Metainformationen (Komponist etc.), die im Notenbild erscheinen.

- eine **Syntaxdefinition** (Grammatik) in Form eines XML-Schemas oder in Form einer document type definition (DTD). In dieser Datei sind Regeln festgelegt, wie XML-Elemente, Attribute und andere Daten definiert und mit einem logischen Bezug zueinander in einem XML-Dokument dargestellt werden können. Und noch wichtiger: anhand eines DTDs oder eines XML-Schemas kann ein Empfängerprogramm prüfen, ob das XML-Dokument wohlgeformt ist (syntaktisch ok) und somit gefahrlos verarbeitet werden kann. Name und Pfad des DTDs oder XML-Schemas werden übrigens im Dokumentkopf von XML-Dateien festgehalten.

Hinzu kommt sehr oft ein dritte Datei, eine sogenannte **Stylesheet-Datei** (z.B. im Format CSS, = "cascaded style sheet" aus dem WEB-Kontext). Sie legt fest, wie die

Dokumentelemente am Ausgabegerät (Bildschirm, Drucker, Sequenzer etc.) dargestellt werden sollen. Diese Stylesheet-Datei kann für verschiedene Ausgabegeräte unterschiedliche Formatvorschriften enthalten, also etwa für Farbdrucker andere Formatierungsvorschriften wie für Schwarz-Weiß-Drucker. Stylesheet-Dateien sorgen also für die Trennung von Form und Inhalt, auf die man sich in der Welt der Textverarbeitung seit langem geeinigt hat.

Im Prinzip kann jeder Hersteller und sogar jeder Musiker seine eigene Syntaxdefinition aufstellen. Der gleiche musikalische Sachverhalt kann also durch beliebig viele unterschiedliche XML-Dokumente angemessen dargestellt werden. Dies macht natürlich in der Praxis wenig Sinn. Deswegen hat sich nach einer kurzen Zeit der babylonischen DTD-Verwirrung (es gibt viele DTDs für Notation in der Musik, vgl. dazu auch die vorstehende Tabelle von CASTAN in Kapitel 4) eine DTD durchgesetzt, die von vielen Musikern und Herstellern von Notations-/Scan-/Midisoftware akzeptiert wird, die DTD der Firma Recordare (www.recordare.com/downloads.html#DTD). Dieser Ansatz wird im folgenden immer als MusicXML bezeichnet.

In Deutschland hat u.a. der Hersteller der Notationssoftware von CAPELLA eine eigene DTD (www.capella.de) vorgestellt. CAPELLA bietet mit der Software **capella Media Producer** u.a. einen Konverter an, der XML von Capella (capXML) nach MusicXML transformiert.

Ein weiterer Vorteil von XML liegt darin, dass im Public Domain Bereich und im kommerziellen Umfeld eine Vielzahl von Werkzeugen zu finden sind, mit denen XML-Dokumente bearbeitet werden können. So gibt es **Parser** (Beispiel: Xerces in JAVA und C++) mit denen XML-Dokumente geparst werden können und die es auch erlauben, XML Dokumente zu schreiben. Andere XML-Werkzeuge wie XPath und XQuery erlauben das Auffinden von Daten (Navigation) innerhalb eines XML-Dokuments.

XPath wurde entwickelt, um einfach auf bestimmte Einträge in einem XML-Dokument zugreifen zu können. In Musikstücken kann dies beispielsweise der Zugriff auf einen Takt oder auf alle Akkorde eines Werkes sein. Eine Fortentwicklung mit erheblich komplexeren Leistungsumfang hat zur Entwicklung von **XQuery** geführt (XML-Query). Im wesentlichen ist XQuery eine standardisierte Abfragesprache, die vom World Wide Web Consortium entwickelt wurde. XQuery ist flexibel genug, um Abfragen für eine breite Palette an Datenquellen durchführen zu können, darunter relationale Datenbanken, XML-Dokumente, Web Services oder kommerzielle Softwarelösungen. Weitere Einzelheiten zu XQUERY sind beispielsweise dem Buch von Lehner / Schöning zu entnehmen.

5.3 Ein Beispiel in MusicXML

Nach diesen theoretischen Ausführungen sei ein Beispiel betrachtet. Damit dieses Beispiel nicht ausufert, sei nur Takt 1 von „All the things you are" (Notenbild s. oben) betrachtet. Erfasst man diese Noten mit einem gängigen Programm (hier Finale Print Music) und exportiert es als MusicXML-Datei, so erhält man folgende XML-Datei als Ergebnis:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 1.0 Partwise//EN"
                        "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise>
  <movement-title>All the things you are</movement-title>
  <identification>
    <creator type="composer">Jerome Kern</creator>
    <encoding>
      <software>PrintMusic 2006 for Windows</software>
      <software>Dolet Light for PrintMusic 2006</software>
      <encoding-date>2007-05-17</encoding-date>
    </encoding>
  </identification>
  <part-list>
    <score-part id="P1">
      <part-name>E-Gitarre</part-name>
      <score-instrument id="P1-I1">
        <instrument-name>Electric Guitar (jazz)</instrument-name>
      </score-instrument>
      <midi-instrument id="P1-I1">
        <midi-channel>1</midi-channel>
        <midi-program>27</midi-program>
      </midi-instrument>
    </score-part>
  </part-list>
  <!--=========================================================-->
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>-4</fifths>
          <mode>major</mode>
        </key>
        <time symbol="cut">
          <beats>2</beats>
          <beat-type>2</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
        <transpose>
          <diatonic>0</diatonic>
          <chromatic>0</chromatic>
          <octave-change>-1</octave-change>
        </transpose>
      </attributes>
      <direction placement="above">
        <direction-type>
          <metronome default-y="95" relative-x="-114">
            <beat-unit>quarter</beat-unit>
            <per-minute>200</per-minute>
          </metronome>
        </direction-type>
        <sound tempo="200"/>
```

```
57        </direction>
58        <harmony default-y="40">
59          <root>
60            <root-step>F</root-step>
61          </root>
62          <kind>minor-seventh</kind>
63        </harmony>
64        <note>
65          <pitch>
66            <step>A</step>
67            <alter>-1</alter>
68            <octave>4</octave>
69          </pitch>
70          <duration>4</duration>
71          <voice>1</voice>
72          <type>whole</type>
73        </note>
74      </measure>
75      <!--=========================================================-->
76      <measure number="2">
```

Das Ergebnis wird den Laien verwundern. Zur Beschreibung des ersten Taktes werden über 40 Zeilen Text benötigt (Zeile 28 – 74). Alle wesentlichen Eigenschaften des Taktes 1 stehen zwischen den Tags <measure> (Zeile 28) und </measure> (Zeile 74)

Sehen wir uns jedoch Einzelheiten an. Ein MusicXML-Dokument beinhaltet zunächst einige Informationen über die verwendete DTD, die zum Entschlüsseln benötigt wird, die verwendete Software und allgemeine Angaben zum Notenwerk.

Konkret wird in Zeile 1 ein Standardheader definiert, der von allen XML-Dokumenten verwendet wird. Es wird eingestellt, welche Zeichencodierung (hier UTF-8, eine Codierungsform für sogenannten Unicode) verwendet wird. Ebenfalls wird festgelegt, dass eine externe DTD eingebunden werden soll (standalone=”no”). Die exakte inhaltliche Beschreibung dieser DTD ist unter Internet-Adresse www.recordare.com/dtds/partwise.dtd zu finden.

Zeile 2 legt den Typ der verwendeten DTD fest, da Recordare zwei unterschiedliche Typen verwenden kann. Wenn zu Beginn des Dokuments als DOCTYPE die DTD PARTWISE angegeben ist, wird nun das erste Instrument von Beginn bis Ende des Stückes notiert, dann das zweite und so weiter. Wäre als DOCTYPE die DTD TIMEWISE angegeben, so würde das Stück von Beginn bis Ende in zeitlicher Abfolge notiert. Die Noten sind dann nicht nach Instrumenten sortiert, sondern nach Zählzeiten und Takten. Diese Version ist für die Echtzeitwiedergabe von Musikstücken sicherlich interessanter, während die PARTWISE-Version leichter lesbar ist.

Zeile 13 bis 24 charakterisiert für die (einzige) vorhandene Stimme P1 das Instrument E-Gitarre und macht Angaben zur MIDI-Ansteuerung.

Ab Zeile 26 beginnt dann die Beschreibung der einzelnen Takte. (**measure number = ”1”**). Anschließend werden die Taktattribute gesetzt. Diese müsen nur einmal angegeben werden und gelten dann bis zum Ende des Dokuments, oder bis ein neues

”attributes“-Element vorkommt.

In MusicXML hat jede Note ein ”duration“-Element. Dieses gibt die Tonlänge in Relation zu einem ”divisions“-Element (siehe Zeile 30) an. Die Tonlänge einer Note berechnet sich somit aus dem Wert aus dem “duration“-Eintrag geteilt durch den Wert von ”divisions“. Da wir in unserem Beispiel keine Notenwerte kleiner als ”Viertel“ benötigen, ist der Teiler auf 1 gestellt. Würden Achtel oder Sechszehntel-Noten im Notenstück benötigt, wären der Wert von “duration“ 2 bzw. 4.

Das ”key“-Element in Zeile 31 gibt die Tonart an, in der sich das Notendokument befindet. Ist das Werk in C-Dur geschrieben, wäre dieser Wert auf 0 gestellt. Würde unser Stück z.B. in D-Dur stehen (2 Kreuze) so müssten wir das ”key“-Element auf 2 stellen. Da unser Stück in Ab-Dur steht ist hier -4 einzustellen. Die Angabe der Tonart ergibt sich aus dem Quinten- und Quartenzirkel (> 0: Quintenzirkel, Kreuztonarten, < 0: Quartenzirkel, Bb-Tonarten)

```
35 <time symbol =”cut” >
36 <beats >2</ beats >
37 <beat - type >2</beat - type >
38 </ time >
```

In dieser Sektion wird nun die Taktart des Stückes eingestellt. Mit time **symbol =“cut“** ist die Takteinstellung halbtaktig gemeint. Zeile 37 stellt die Anzahl der ”Beats“ in einem Takt ein. der zweite Wert die Länge des ”Beats”. In unserem Fall haben wir es mit einem 4/4 Takt zu tun, der halbtaktig zu behandeln ist, d.h. wir haben einen Beat-Typ von 2 und 2 Beats pro Takt.

```
39 <clef >
40 <sign >G</ sign >
41 <line >2</ line >
42 </ clef >
```

Als nächstes definieren wir den Notenschlüssel. In der gängigen Musiknotation gibt es eine Vielzahl von Notenschlüsseln. Schlüssel dienen der Erleichterung der Musikdarstellung - damit nicht zu viele Hilfslinien erforderlich sind. Der Violinschlüssel wird hier als G“-Schlüssel dargestellt, der von der 2. Linie (Note G) ausgeht.

Desweiteren folgen dann Angaben zum Metronom und zum Abspielen des Sounds. (Zeile 48 – 57). Der zu Takt 1 gehörendene Fm7-Akkord wird in der XML-Datei in Zeile 58-63 definiert. Zwischen den Tags <note> und </note> (Zeile 64 – 73) ist die zu Takt 1 gehörende Ganznote Ab dargestellt. Dies erfolgt durch die Angabe des Tones A in Tonlage 4 vermindert um eine Tonhöhe (also Ab). Kreuze würde mit + 1 alteriert. Dies leistet der <alter>-Tag.

In MusicXML sind viele weitere Möglichkeiten gegeben, um Musikwerke zu beschreiben. In diesem Zusammenhang sei beispielsweise Punktierung, Triolen, Bindebögen, Gruppierung von Noten, Pausen, Dynamikangaben, Akzentuierungszeichen, Verwendung von Liedtexten etc. erwähnt. Der geneigte Leser kann unter der Internetadresse www.recordare.com/xml/musicxml-tutorial.pdf dazu Einzelheiten erfahren.

Als **Vorteile** von MusicXML sind zu erwähnen:

- frei verfügbar

• gute Lesbarkeit
• schnell zu erlernen
• leicht erweiterbar
• durch verschiedene Parser leicht in andere Formate umzuwandeln (z.B. MIDI)
• plattformunabhängig
• leicht komprimierbar (durch ZIP beispielsweise)
• wird von sehr vielen Softwareprogrammen unterstützt
• das MusicXML-Dokument kann in gängigen relationalen Datenbanken gespeichert werden.

Das letzte Argument bedarf einiger Erläuterung. Unterstellt man, daß herkömmlich ein Musikstück eine Datei bildet, so hat man häufig mehrere hundert Dateien mit Musikstücken, die für den Nutzer interessant sind. Diese sind dann in der Regel über mehrere Dateiverzeichnisse und Platten verteilt. Speichert man das Musikstück zunächst in einem proprietären Format und wandelt dies dann mit einem Konverter in MusicXML um, kann die XML-Datei in eine Datenbanktabelle überführt werden, in der alle benötigten Musikstücke liegen. Dann sind alle gespeicherten Musikwerke zentral an einer Stelle verfügbar. Dort sind diese mit entsprechenden Datenbankbefehlen wartbar. Datensicherungsroutinen, die im Datenbanksystem implementiert sind, übernehmen die Arbeit des Backups dieser Musikstücke. Mit Sprachen wie XSL, XPATH und XQUERY sind eine Vielzahl von Konvertierungsroutinen und Analysemöglichkeiten gegeben.

Allerdings bringt das Format auch einige **Nachteile** mit sich:

- sehr ausführliche textuelle Darstellung (ein Takt benötigt schnell mehr als 30 Zeilen)
- schlechte Übersicht über das Dokument, sofern kein geeigneter Editor (z.B.XMLSpy) bzw. Parser (z.B. Xerces) verwendet wird
- wenige Hardwarekomponenten (z.B. Synthesizer) verstehen im Moment MusicXML.

6 Orientierung am Klang: MIDI

Wie der Name bereits sagt, gehen klangorientierte Ansätze davon aus, dass die zu erfassende Musik bereits gespielt wurde oder gerade gespielt wird. Die klangorientierten Ansätze lassen sich in zwei Gruppen aufspalten:

- wissenschaftlich-physikalische Ansätze
- praktische geräteorientierte Ansätze.

Die wissenschaftlich-physikalischen Ansätze versuchen meist, Klänge physikalisch (z.B. als Schichtungen von Sinuswellen, Fourier-Analyse) zu beschreiben oder zu analysieren. Ein irgendwie normierter Ansatz hat sich dabei nicht herausgebildet. Da es sehr schwer

ist, aus Klängen zu eher kognitiven Strukturen wie Stimmen, Dauer einzelner Noten etc. zu kommen, ist dieser Ansatz für den breiten Einsatz nicht geeignet. Er wird hier weiterhin nicht betrachtet.

Der geräteorientierte Ansatz ist weniger wissenschaftlich als praktisch: Klänge sind das Ergebnis eines elektronischen Klangerzeugers, z.B. eines Synthesizers. Die eigentliche Klangerzeugung ist dann meist uninteressant. Der bekannteste Ansatz ist **MIDI.**

MIDI ist ein um 1983 konzipiertes Format, das ursprünglich nur für die Ansteuerung von Synthesizern gedacht war. Es stammt nicht etwa von einem Normierungsgremium wie DIN, ISO, ECMA o.ä. sondern von einem Konsortium von Herstellern elektronischer Musikinstrumente, darunter Roland und Yamaha.

Das ursprüngliche Leitungsprotokoll ging von einer seriellen Stromschnittstelle mit 31,25 KBaud Übertragungsrate aus. Heute mutet es archaisch an, dass außer den üblichen Synchroniserungsmechanismen wie Startbit und Stopbit keinerlei Datensicherungs-mechanismen (Paritätsbits, Hamming-Codes, CRC-Checks etc.) eingebaut wurden. Das erklärt, warum viele MIDI-Synthesizer immer noch eine "Panic" Taste aufweisen, die im Falle heilloser Datenverwirrung gedrückt werden kann. Inzwischen werden statt des Original-Leitungsprotokolls auch oft andere Protokolle verwendet, z.B. USB oder TCP/IP.

Bei der damals üblichen niedrigen Durchsatzrate von ca. 32 KBaud (max. ca. 3000 Bytes pro Sekunde) musste mit jedem Byte gegeizt werden, um nicht deutliche hörbare Verzögerungen zu riskieren. Deshalb verwendet MIDI früher übliche Kompressionsmechanismen wie variable Längen in Zahlenwerten oder die Kombination zweier Informationen (z.B. Status und Kanal) in einem Byte. Die häufigsten MIDI Ereignisse selbst können ein, zwei oder 3 Bytes lang sein aber auch (z.B. Textereignisse) länger sein.

Jede MIDI Information heißt "Ereignis" ("event") oder auch "Botschaft" ("message"). Jedes Ereignis besteht aus einer variabel langen Zeitinformation (relative Zeiten, d.h. die Differenz in Ticks zum vorhergehenden Ereignis) und einer Steuerinformation.

MIDI geht von einem Tasteninstrument als Denkmodell aus. Demzufolge bedeuten die beiden wichtigsten Steuerinformationen:

- Taste gedrückt
- Taste losgelassen.

Diese beiden Ereignistypen transportieren weitere wichtige Informationen wie Anschlagstärke, Tastennummer und Kanal. Im MIDI Standard werden 16 Kanäle unterstützt. Jedem Kanal kann ein Instrument aus einer Liste von 128 Instrumenten zugeordnet werden.

Aus diesen beiden Ereignistypen kann schon ein wohlgeformter MIDI-Strom aufgebaut werden. Alle weiteren Ereignistypen sind eigentlich optional, d.h. sie müssen nicht auftreten. Zu den weiteren Ereignistypen zählen die "controller events", darunter insbesondere die klaviertypische Pedalinformation ("Hold") oder auch Hall, Vibrato, Pitch Bend etc.

Hinzu kommen musikalische Metainformationen wie Tempo, Taktvorgabe (4/4, 6/8 etc.) oder Schlüsselangaben. Alle diese Angaben sind optional - in kaum 50% aller MIDI-

Dateien wird man solche Metainformationen finden. Sie sind ja für die Produktion von Klangereignissen über einen Synthesizer nicht wichtig. Auch Textinformationen (Copyright, Spurname, Begleittexte) können über MIDI transportiert werden. Das Karaoke-Format (Extension .kar) ist nichts anderes als eine Ableitung aus dem MIDI-Format.

Über dem MIDI-Leitungsformat, das Regeln zur Formulierung des musikalischen Gehalts festlegt, wurde im MIDI-Standard ein Dateiformat spezifiziert. Dieses Format legt z.B. fest, dass ein MIDI-Dateivorspann ein "magic word" als Kennung haben muss, dass die MIDI-Formatvariante dort verzeichnet sein muss und dass die zeitliche Grundeinheit ("division") dort festgehalten werden muss. Diese zeitliche Grundeinheit legt fest, wieviele "Ticks" (Pulse eines Timers) pro Viertelnote den Zeitstempeln der Ereignisse zugrunde gelegt werden. Damit wird noch nicht das Tempo festgelegt! Aber das ist schon eines der schwierigeren Themen des ansonsten nicht so komplizierten MIDI-Formats.

Das MIDI-Dateiformat kennt das Konzept der "Spur" - was nicht mit "Kanal" verwechselt werden darf. Spuren sind eigentlich nur Container für Ereignisse (vielleicht mit den Systemen einer Partitur vergleichbar), die inhaltlich zusammengehören. Das Konzept Spur hat kein Äquivalent im Leitungsprotokoll, d.h. zur Abspielzeit bilden alle Ereignisse einen sequentiellen Strom, der nur durch die zeitliche Abfolge geordnet ist - nicht durch Spuren.

MIDI war eigentlich nie als Notationsformat gedacht, sondern als Format zur Produktion von Klangereignissen in Echtzeit. Deshalb kennt MIDI zwar das Konzept "Viertelnote" - aber nur als Hilfskonzept zur Festlegung der Zeitstempel. Notenereignisse können beliebig lang sein - es gibt keine Markierung für ganze Noten, halbe Note etc. Pausen werden nicht markiert - sie müssen aus den Zeitstempeln rekonstruiert werden. Konzepte wie Triolen, Quintolen oder andere Gruppierungen sind unbekannt: auch solche Informationen müssen aus den Zeitstempeln rekonstruiert werden. Jeder Benutzer eines MIDI-Konverters kennt die daraus resultierenden **falschen** Ergebnisse.

Obwohl nie als Notationsformat gedacht, wird MIDI oft mit Notationsformaten verglichen. MIDI kennt z.B. die folgenden Konzepte **nicht**:

- Auftakt
- Pausen
- Notengruppierungen wie Triole, Sextole etc. Systeme (Kombinationen von Notenzeilen: das Konzept "Spur" ist nicht damit vergleichbar!)
- Dynamikänderungen wie Crescendo, Diminuendo
- Änderungen der Geschwindigkeit wie Ritardando, Accelerando
- Ausführungsvarianten wie Legato, Fermate
- Notationsfeinheiten wie aufrechter Notenhals, hängender Notenhals, Balkengruppe, Konzepte wie "transponierendes Instrument" (MIDI Sequenzer kennen das - nicht aber das Protokoll).
- enthält keine Layoutinformationen.

Das obige Beispiel soll in dem Binärformat MIDI dargestellt werden. Dieser Auszug aus einem MIDI-Dump zeigt die ersten 3 Takte von "All the things you are" (Notenbild s. oben) für eine Melodiespur. Die MIDI-Datei wurde über einen Noteneditor (Capella) erzeugt und als Datei in einem eigenen Dump-Format abgelegt. Anmerkung: Dies ist

nicht das "echte" MIDI-Dateiformat: das MIDI-Format ist binär und somit nicht direkt lesbar. Dieses Dump Format ist aber eng an das MIDI-Dateiformat angelehnt.

Eine MIDI-Datei besteht aus einem Dateivorspann ("Header") und den Ereignissen, die meist in "Spuren" aufgezeichnet sind. Der Dateivorspann fehlt natürlich komplett beim Spielen über MIDI-Synthesizer in Echtzeit. Er ist Bestandteil des Dateiformats, nicht des Echtzeitprotokolls.

Informationen aus dem Dateivorspann:

```
format=1
no_tracks=2
division=480
```

Aus dem Dateivorspann sind hier nur drei Informationen extrahiert: das Format der Datei -1- bedeutet dass die Datei in "Spuren" organisiert ist, wobei Spuren nicht mit Kanälen (Instrumenten) verwechselt werden dürfen. Die Datei enthält zwei Spuren – das sind eher Aufzeichnungsblöcke. Die wichtigste Information ist "division": hier wird festgelegt, wieviele Ticks (Schläge der internen Uhr) eine Viertelnote hat.

Die MIDI Ereignisse der ersten Spur:

```
00000000 ff5103070ae2  // settempo: Zeiteinheit für Viertelnote in Ticks
00000000 ff5902fc00    // keysign: -4 = 4B
00000000 ff030a756e62656e616e6e7431 //trackname, Länge 0x0a=10,"unbenannt1"
00000000 c000    // Instrumentzuordnung: Kanal 0 = Piano
00000000 b00750  // Control change: Controller No.7, Lautstärke=0x50=80 00000000 904464
00000000 904464  // Takt 1: as ON
00001900 804464  // as OFF
00001920 904964  // Takt 2: des ON
00003345 804964  // des OFF
00003360 904464  // as ON
00003835 804464  // as OFF
00003840 904364  // Takt 3: g ON
00004315 804364  // g OFF
00004320 904364  // g ON
00004795 804364  // g OFF
00004800 904364  // g ON
00005275 804364  // g OFF
00005280 904364  // g ON
00005755 804364  // g OFF
```

Jede variabel lange Übertragungseinheit heißt in MIDI "Ereignis". Ein Ereignis besteht immer aus einer variabel langen, relativen Zeitangabe (hier als absolute Zeit mit fester Länge, 8-stellig dezimal kodiert dargestellt) und der Sachinformation, die ebenfalls variabel lang sein kann. Wenn man genau hinschaut, erkennt man dass die Notenanfänge immer auf Vielfache von 480 (Parameter "division") fallen. Durch die variablen Längen, durch die Binärkodierung und durch Optimierungsmöglichkeiten ist MIDI ein nicht ganz einfach zu parsendes Format.

Die ersten 5 Zeilen enthalten (vereinfacht dargestellt) **Metaereignisse** - in diesem Beispiel eine Tempoangabe, eine Tonartangabe, ein Spurname, eine Instrument-zuordnung und eine globale Lautstärkeeinstellung. Metaereignisse sind optional. Man kann sich also nicht darauf verlassen, dass eine MIDI-Datei Metainformationen wie Tonartangaben oder Tempoangaben enthält.

Ab Zeile 6 beginnen die **Notenereignisse**. MIDI-Notenereignisse können als **Tastenanschläge** interpretiert werden. Jedes Notenereignis besteht aus 3 Bytes und trägt vier Informationen:

- Taste gedrückt (0x9...) oder losgelassen (0x9...): das erste Halbbyte

- Kanalinformation: in diesem Beispiel immer 0: das zweite Halbbyte

- Tastennummer: Beispiel: 0x44=68 steht für die Note "as eingestrichen".

- Anschlagstärke: hier immer 0x64=100. Beim "Note Off" Ereignis ist die Anschlagstärke bedeutungslos.

MIDI enthält bis auf eine optionale Taktangabe keine Informationen über Taktanfänge oder Taktnummern! Solche Informationen muss ein MIDI-Player oder Sequencer mühsam rekonstruieren. Man stelle sich die ständig wechselnden Taktarten z.B. bei Stravinsky vor: Fehler sind vorprogrammiert. Auffällig ist auch die ziemlich komplexe und intuitiv nicht direkt nachvollziehbare Darstellung zeitlicher Informationen (Tempo).

Zusammenfassend sind über MIDI die nachstehenden Aussagen zu treffen:

- Binäres Datenformat, nur für Spezialisten lesbar, für Analysen nicht besonders gut geeignet

- Variable Interpretation durch unterschiedliche Hersteller. Der Mindestumfang der dargestellten

- Informationseinheiten ist gering - viele Konzepte (z.B.Taktangabe) werden nur optional erfasst

- kann nur approximativ in ein Notationsformat übersetzt werden (Notendarstellung, Vortragsbezeichnungen etc.)

- Keine Layoutunterstützung: damit ist MIDI für "Rendering" (z.B. Druckausgabe) nicht geeignet

- Keine normierte Unterstützung der Akkordspeicherung (Akkorde können als Textannotationen gespeichert werden).

MIDI scheidet damit als Grundlage für ein elektronisches Notationsformat aus - was keine Schande ist: dafür war es nie gedacht. Ein zukünftig normiertes Notationsformat muss jedoch darauf achten, dass die darin erfassten Musikstücke in MIDI umgesetzt werden können und dass MIDI - soweit möglich - im Input verlustfrei akzeptiert wird. Hier schlägt die Stunde der Konverterbauer.

7. Fazit

Notationsformate für die digitale Speicherung von Musikwerken wurden in der Vergangenheit häufig proprietär entwickelt. Das Problem des Datenaustauschs zwischen verschiedenen Softwareanwendungen ist damit gegeben. Häufig wurde in der Vergangenheit nur das MIDI-Format unterstützt, das aber – wie gezeigt – sehr viele Nachteile und Probleme mit sich bringt. Hier kann das relativ neue Format XML als Austauschformat Abhilfe schaffen. XML kann viele Schwachpunkte von MIDI wirkungsvoll umgehen. Das letztgenannte Format kann mit Hilfe der XML-Sprachen XSL, XPATH und XQUERY aus einem in XML-Format gespeicherten Musikwerk MIDI erzeugen, aus gescannten PDF-Vorlagen XML-Daten erzeugen, die in die proprietären Softwaresysteme zur Notenerfassung überführt werden können. Weitere Anwendungen sind der Datenaustausch von XML nach LILYPOND, von XML nach ABC etc. Durch das Vorliegen eines Musikwerkes in XML-Format können umfassende automatisierte Analysen vorgenommen werden, die vorher nur mit großer Mühe erstellt werden konnten

8. Literatur/Internetquellen

Castan, Gerd: Notationsformate. Download unter www.music-notation.info

Kepper, Johannes: Codierungsformen von Musik, Referat beim Kolloquium des Ausschusses für musikwissenschaftliche Editionen der Union der deutschen Akademien der Wissenschaften, Akademie der Wissenschaften Mainz, 16.-18. November 2006. Download unter**:** www.adwmainz.de/fileadmin/adwmainz/MuKo_Veranstaltungen/S2-Digitale_Medien/kepper.pdf

Lehner, Wolfgang/Schöning, Harald: XQUERY - Grundlagen und fortgeschrittene Methoden. Dpunkt-Verlag 2004

Neuschwander, Hans Werner: MIDI - (Musik- und MIDI-Grundlagen), Fachhochschule Kaiserslautern, Fachbereich Elektrotechnik/Informationstechnik, Manuskript 2003

www.recordare.com

www.capella.de

www.finale.com