Conference Lausanne, Switzerland (ICANN'97). Berlin / Heidelberg: Springer 583-588.

Soden, Wolfram von (1994): *The ancient Orient*. An introduction to the study of the ancient Near East. Michigan: Wm. B. Eerdmans Publishing Co.

Formate als Sackgassen: Handlungsempfehlungen

Bohl, Benjamin W.
bohl@edirom.de
Zentrum für Musik- und Filminformatik, Hochschule Ostwestfalen-Lippe

Berndt, Axel, Dr.
berndt@hfm-detmold.de
Zentrum für Musik- und Filminformatik, Hochschule Ostwestfalen-Lippe

Senft, Björn
bsenft@s-lab.uni-paderborn.de
Software Quality Lab, Universität Paderborn

Im Kontext des *Zentrum - Musik - Edition - Medien* beschäftigen sich die Autoren mit der Modellierung und Codierung musikalischer Phänomene. Formate zur Codierung von Musik reichen von ASCII-basierten Codierungen (Humdrum, ABC-Notation) über SGML-basierte Formate (SMDL) und XML-basierte Formate (MusicXML, MEI) bis hin zu technischen Steuerdaten (MIDI und spezifische Formate für Sequencer-Programme) oder Audiodaten (FLAC, MP3) (vgl. Selfridge-Field 1997).[1] Es sind die spezifischen Anforderungen und Ansprüche, der Fokus auf die Darstellung bestimmter (musikalischer) Phänomene, und die Ausrichtung auf einen bestimmten Nutzungskontext, die jedes Format prägen und ihm seine Berechtigung geben (vgl. Veit 2006). Jedoch stellt der Informationsaustausch zwischen den verschiedenen Nutzergruppen, mit ihren spezifischen Formatvorlieben und den jeweils relevanten -erfordernissen, ein essentielles Problem dar. Ausgehend von dieser, zwar im Beispiel musikspezifischen, im Kern jedoch allgemeingültigen Problemstellung, sollen im Folgenden Handlungsempfehlungen entwickelt werden, die zur Planung und Einschätzung digital arbeitender Projekte herangezogen werden können.

„Während MusicXML als Austauschformat inzwischen größte Verbreitung gefunden hat, versucht MEI ausdrücklich, musikeditorische Anforderungen zu erfüllen.“ (Kepper 2009: 216). Im Vergleich zu MusicXML und vielen anderen Codierungsformaten für Musik (vgl. Selfridge-Field 1997), ermöglicht das XML-basierte Codierungsformat der Music Encoding Initiative (MEI) eine umfassende metadatliche Beschreibung (vgl. Richts / Herold 2014), sowie die Codierung editorischer Sachverhalte (vgl. Roland / Kepper 2014). Aufgrund dieses Alleinstellungsmerkmals hat es sich im Bereich der digitalen Musikedition etabliert und findet sich inzwischen auch im Recommended Formats Statement der Library of Congress (vgl. Library of Congress 2015). Somit ist es nachvollziehbar, dass zunehmend mehr Editionsprojekte auf MEI zurückgreifen, nicht zuletzt, um dem *Digital Turn* und seiner Forderung nach einer nachhaltigen Bereitstellung von Forschungsdaten zur Nachnutzung nachzukommen.

Denkbare Nutzungsszenarien in diesem Kontext sind u. a. im Music Information Retrieval (MIR), in der Interpretationsforschung sowie in der weiteren Verarbeitung von Musikdaten (Musikgenerierung und -adaption), im Notendruck und in der Musikproduktion verortet. Jede Disziplin bringt ihre eigenen Anforderungen an die Modellierung der Informationen mit.

- Im MIR werden einfache Strukturen benötigt, etwa CSV-Daten, um ein schnelles Parsen für Echtzeit-Anwendungen zu gewährleisten.
- In der Interpretationsforschung spielt die Analyse von Audiodaten eine wichtige Rolle. MEI ist dafür nicht spezifisch genug, enthält keine Audiodaten und keine Möglichkeit, solche Analyseergebnisse zu repräsentieren.
- Für die Musikgenerierung und -adaption werden Tondaten und Steuerdaten vorausgesetzt. Im Falle von MEI sind erstere unvorteilhaft strukturiert, deshalb aufwendig zu prozessieren. Steuerdaten lassen sich nur unzureichend einbinden. Auch die Musikproduktion arbeitet vornehmlich mit elektronischen Steuerdaten, sowie Audiodaten.
- Für das Layout des graphischen Notenbildes, ist die logische Struktur der musikalischen Informationen zweitrangig. Aufgrund der zu geringen und unvollständigen Unterstützung durch Notationsprogramme (etwa mittels Importer) ist MEI für den Notendruck derzeit irrelevant.

Seine durchaus beabsichtigten Uneindeutigkeiten und die Möglichkeiten der Anreicherung mit editionsspezifischen Informationen machen MEI zu einem für die digitale Musikedition mächtigen, für die exemplarisch beschriebenen weiteren Nutzungszenarien jedoch unpraktikablen Format. Dies birgt die Gefahr, dass trotz der Bereitstellung der in MEI codierten Editionsdaten das Ende der Nutzungskette bereits erreicht ist.

Ähnliche „Sackgasseneffekte“ lassen sich auch in anderen Nutzungskontexten und deren Formaten beobachten. Zu deren Überwindung sind mehrere grundlegende Szenarien denkbar: die Erweiterung eines bestehenden Formates, die gleichzeitige Nutzung

mehrerer Formate (ggf. gekapselt in einem Container-Format), oder die Konvertierung in andere Formate. Jeder Ansatz ist mit spezifischen Vor- und Nachteilen verbunden, die im Folgenden diskutiert werden sollen.

Lösungsansätze

Erweiterung

Eine naheliegende Lösung mag in der Erweiterung des Formats bestehen. Das beinhaltet die Modellierung, Formalisierung und Implementierung der neuen Elemente. Während dies für punktuelle Phänomene noch praktikabel sein mag, ziehen umfangreichere Erweiterungen eine immer größere Komplexität der Datenstruktur nach sich. Insbesondere dann, wenn eine bereits die spezifischen Erfordernisse der einen Anwendungsdomäne widerspiegelnde Struktur mit einer weiteren, einer ganz anderen Domäne Rechnung tragenden Struktur, überlagert wird. Dies kann im Falle vom MEI bereits jetzt beobachtet werden, wie beispielsweise durch das Nebeneinander verschiedener Zeitdarstellungen (symbolische / musikalische Zeit, Aufführungszeit) und einzelne, jedoch unvollständige Querbezüge zum MIDI-Standard.[2] Gegebenenfalls kann ein grundsätzlich neues Datenformat dabei entstehen.

Die Replikation relevanter Informationen in einen neuen und entsprechend anders strukturierten Bereich des Formats würde hingegen in einem wenig (speicher-)effizienten Format mit zahlreichen Redundanzen resultieren. Generell sind Redundanzen aufwendig zu pflegen. Hierbei auf Referenzen zurückzugreifen kann sowohl die Gefahr von Inkonsistenzen, als auch den Wartungsaufwand verringern.

Zudem muss sich ein “allen gerecht werden wollendes” Format neben den spezialisierten, etablierten Formaten durchsetzen können. Dass dies gelingt, ist höchst fraglich, da zunächst alle Verarbeitungsverfahren und Werkzeuge, die für die etablierten Formate bereits bestehen, neu implementiert oder zumindest angepasst werden müssen. Ferner sind die etablierten Formate und ihre zugehörigen Werkzeuge gerade dank ihrer Spezialisierung auch für ihren jeweiligen Anwendungskontext optimiert und unterstützen die effiziente Arbeit mit den Daten. Ein weniger spezialisiertes Format ist daher oft ineffizienter, nicht nur hinsichtlich seines Speicherbedarfs, sondern auch hinsichtlich des benötigten Rechen- bzw. Verarbeitungsaufwandes.

Manuelle parallele Datenhaltung

Wenn die Konkurrenz zu etablierten Formaten vermieden werden soll, was im Sinne der Nachhaltigkeit generell zu empfehlen ist, bietet sich die parallele Bereitstellung der Daten in mehreren Formaten an. In Abhängigkeit der zu adressierenden Anwendungsszenarien und den damit einhergehenden Anwenderprofilen wird eine Auswahl der relevanten Formate getroffen. Die Daten werden nun parallel in jedem dieser Formate gepflegt. Das kann unter Zuhilfenahme der dafür existierenden Werkzeuge geschehen, sodass kein Software-technischer Entwicklungsaufwand anfällt. Jedoch entsteht ein Mehraufwand in der Datenpflege, denn die allen Formaten gemeinen Inhalte (Redundanzen) müssen synchron gehalten werden. Jedes Format hat ferner seinen eigenen Anwendungskontext mit entsprechenden, spezialisierten (nichtredundanten) Inhalten. Automatismen, welche dem Anwender diesen Synchronisationsaufwand abnehmen, sind im Allgemeinen nicht vorhanden; die Arbeit geschieht „manuell“. Die richtige Verwendung und Pflege der Daten und Werkzeuge erfordert eine entsprechende Bearbeitungsdisziplin der Editoren. Dies stellt eine Gefahr für die Konsistenz des Datensatzes dar und birgt die Gefahr des Zerfalls des Datensatzes in einzelne unzusammenhängende, weil nicht synchrone, Datenobjekte.

Konvertierung

Möchte man den aus der parallelen Datenhaltung resultierenden manuellen Mehraufwand vermeiden, bietet die Nutzung von Konvertern eine Erleichterung. Der Nutzer arbeitet, so lange es seiner Fragestellung genügt, in ein und demselben Format und konvertiert es erst bei Bedarf in andere Formate. In einer entsprechenden Arbeitsumgebung kann dies durch Automatismen unterstützt werden, welche bei Veränderungen an einem Objekt die Synchronisation mit den Parallelobjekten durchführen. Konverter können innerhalb bestehender Anwendungsprogramme in Form von Importern die formatübergreifende Arbeit erleichtern.

Sofern jedoch die Formate nicht äquivalent sind, wovon im Allgemeinen ausgegangen werden muss, kann die Konvertierung mit Informationsverlust verbunden sein, vor allem dann, wenn die betreffenden Informationen im Zielformat der Konvertierung grundsätzlich nicht repräsentierbar sind. So kann die Erstellung eines Datenobjektes durch Konvertierung lediglich der Startpunkt sein, an welchem die dem Ausgangs- und Zielformat gemeinsamen Inhalte übernommen werden und von wo aus die formatspezifischen Inhalte dann vom Nutzer einzupflegen sind. Sollten für das Zielformat relevante Daten im Ausgangsformat fehlen, so sind auch diese vom Nutzer zu ergänzen. Die gleiche Art von Informationsverlust ist auch bei der Rückkonvertierung zu bedenken. Für den Anwender steigt also der Pflegeaufwand für die in mehreren Formaten vorgehaltenen Daten in dem Maße,

in dem konvertierungsbedingter Informationsverlust und -ergänzung manuell ausgeglichen werden müssen. Die Konvertierung automatisiert lediglich die Pflege der redundanten Inhalte, d. h., die Schnittmenge der Datensammlung in den verschiedenen Formaten. Denkbar ist es in einigen Fällen, die nicht in der Schnittmenge enthaltenen Informationen separat zu den Datenformaten zu speichern, um sie bei der Rückkonvertierung wieder einzupflegen. Eine weitere Voraussetzung für eine (zumindest in weiten Teilen) automatisierte Rückkonvertierung stellt das Wissen über die Transformationshistorie dar.

Handlungsempfehlung

Die bisherigen Ausführungen lassen bereits erkennen: Eine bequeme Lösung gibt es nicht. Jeder der genannten Lösungsansätze findet in der Praxis bereits mehrfach Anwendung, jeweils mit den entsprechenden Vor- und Nachteilen. Diese gilt es abzuwägen, will man sich im Rahmen eines konkreten Projektes für einen Ansatz entscheiden. Dabei werden die folgenden vier Kriterien von maßgeblicher Bedeutung sein.

Nachhaltigkeit: Sollen die Daten längerfristig und über das Projekt hinaus nutzbar sein?

Wenn dies gewünscht ist, sollten die Ergebnisse in den etablierten Formaten der zur Nachnutzung angedachten Nutzergruppen gespeichert werden. Ein eigens im Projekt entwickeltes Format oder Derivat kann, wenn es sich nicht etabliert und keine dem technischen Fortschritt folgenden Aktualisierungen garantiert, keine Nachhaltigkeit sichern.

Rückfluss: Findet ein uni- oder bidirektionaler Austausch zwischen den verschiedenen, vom Projekt adressierten Nutzergruppen statt?

Ein eigens für das Projekt entwickeltes Datenformat wird den aus dem Projektkontext heraus gerichteten Austausch erschweren. Der Rückfluss wird ohne entsprechende Konverter für das eigene Format kaum praktikabel sein.

Der immer wieder auszugleichende Informationsverlust im Konverteransatz wird in einem unidirektionalen Szenarium kaum ein Problem darstellen, denn ohne den Rückkonvertierungsschritt entfällt die mehrfache Einpflege der nichtredundanten Inhalte. Die Übernahme der redundanten Inhalte wird hingegen auch beim Rückfluss erleichtert.

Die parallele Datenhaltung wird für den bidirektionalen Austausch am praktikabelsten sein, weil sie konzeptionell vorsieht, alle Inhalte in den bevorzugten Formaten der adressierten Nutzergruppen vorzuhalten und Änderungen in allen Repräsentationen zu synchronisieren.

Synchronisationsaufwand: Wie hoch darf der Aufwand zur Datensatzpflege sein?

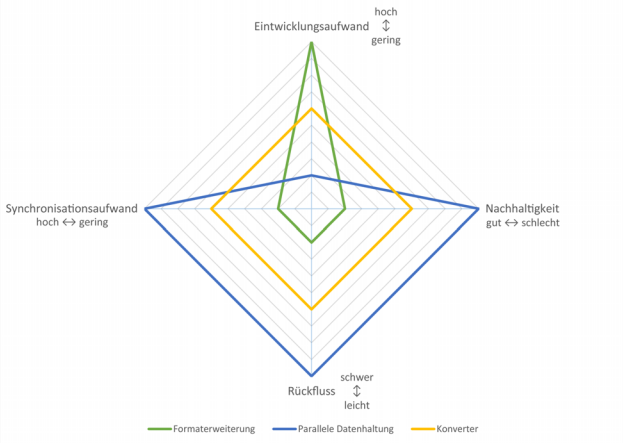
Der manuelle Aufwand zur Datensatzpflege ist bei der Vorhaltung der Daten in mehreren Formaten ohne Automatisierungen höher als bei den anderen vorgeschlagenen Lösungen. Der Rückgriff auf ein im Projekt praktisch ausschließlich verwendetes eigenes Format (eigene Formatanpassung), minimiert den Synchronisationsaufwand. Konverter stellen einen Mittelweg dar, denn die redundanten Informationen können (semi-)automatisch synchronisiert werden, lediglich die formatspezifischen Inhalte erfordern manuellen Pflegeaufwand.

Entwicklungsaufwand: Wieviel Entwicklungskapazitäten sind im Projekt vorgesehen?

Die Entwicklung eines eigenen Datenformats oder von Derivaten existierender Formate zieht auch die Entwicklung von Verarbeitungsverfahren und Werkzeuge nach sich, setzt also einen insgesamt hohen Bedarf an (Software-)Entwicklungskapazitäten voraus.

Auch Konverter verlangen Entwicklungskapazitäten, wenn auch (abhängig von der Menge der unterstützten Datenformate) in bescheidenerem Umfang, zumal für viele etablierte Formate bereits ausgereifte Konverter existieren.

Für die manuelle Pflege der Daten, parallel in mehreren Formaten, sind keine Entwicklungsarbeiten notwendig; hierfür genügen die bestehenden Editoren für die jeweiligen Formate.



Die vorstehende Abbildung veranschaulicht die Gewichtungen der vorgestellten Lösungsansätze in einem Starplot. Dies soll eine Orientierungshilfe zur Projektplanung sein und als Grundlage von Handlungsempfehlungen dienen. Selbstverständlich gibt es weiterführende, die obigen Ansätze kombinierende Möglichkeiten, die etwa im Rahmen von Datenbanksystemen oder integrierten Arbeitsumgebungen bestimmte Aufgaben vereinfachen können. Sofern diese Systeme nicht bereits bestehen, ist dies mit einem gesteigerten Programmieraufwand verbunden, der von inhaltsorientierten Projekten kaum zu leisten ist. Daraus motiviert sich der Bedarf für die Bereitstellung von projektübergreifenden und langfristigen Forschungsinfrastrukturen, eine Thematik, der sich das *Zentrum - Musik - Edition - Medien* mit der Erforschung nachhaltiger Entwicklungskonzepte im Bereich der digitalen Musikedition widmet. Für die

langfristige Bereitstellung einer solchen Infrastruktur bietet die aktuelle Förderpolitik in Deutschland jedoch nur selten die nötigen Grundlagen.

Notes

1. Bei Selfridge-Field (1997) ist eine umfassende Auflistung und Beschreibung unterschiedlicher Codierungsformate für Musik zu finden. Das Buch kann gewissermaßen als Standardwerk in diesem Bereich angesehen werden.
2. Dieser Ansatz eines immer größer werdenden Systems kann ebenfalls bei etablierten Softwaresystemen festgestellt werden. Solche komplexen Systeme werden in der Folge immer schwieriger zu warten und können ihren eigentlichen Zweck immer schlechter erfüllen. Daher gibt es bei diesen Systemen den Trend zur Modularisierung und Spezialisierung (vgl. Krahn / Rumpe 2005).

Bibliographie

Kepper, Johannes (2009): "XML-basierte Codierung musikwissenschaftlicher Daten – Zu den Voraussetzungen einer digitalen Musikedition", in: *it – Information Technology* 51: 216–221.

Library of Congress (2015): *Preservation: Recommended Formats Statement* https://www.loc.gov/preservation/resources/rfs/textmus.html [letzter Zugriff 08. Januar 2016].

Krahn, Holger / Rumpe, Bernhard (2005): *Evolution von Softwarearchitekturen*. Informatik-Bericht 2005-04. Braunschweig: TU Braunschweig http://www.se-rwth.de/~rumpe/publications20042008/Evolution-von-Software-Architekturen.pdf [letzter Zugriff 08. Januar 2016].

Richts, Kristina / Herold, Kristin (2014): *Daten- und Metadatenformate in den Fachdisziplinen: Musikwissenschaft* https://wiki.de.dariah.eu/display/publicde/3.3+Musikwissenschaft [letzter Zugriff: 04. Januar 2016].

Roland, Perry / Kepper, Johannes (eds.) (2014): *Music Encoding Initiative Guidelines*. Release 2013. Revision 2.1.1. Charlottesville / Detmold: Music Encoding Initiative Council http://github.com/music-encoding/music-encoding/releases/download/MEI2013_v2.1.1/MEI_Guidelines_2013_v2.1.1.pdf [letzter Zugriff: 04. Januar 2016].

Selfridge-Field, Eleanor (1997): *Beyond MIDI*. The handbook of musical codes. Cambridge: MIT Press Ltd.

Veit, Joachim (2006): "Musikwissenschaft und Computerphilologie – eine schwierige Liaison?", in: *Jahrbuch für Computerphilologie* 7: 67–92 http://computerphilologie.uni-muenchen.de/jg05/veit.html [letzter Zugriff 30. September 2015].

Algorithmische Visualisierungen: Ausdruck von Routinen und Denkstilen in den Digital Humanities

Bubenhofer, Noah
bubenhofer@cl.uzh.ch
Universität Zürich, Schweiz

Zu den wichtigsten Arbeitsinstrumenten der Digital Humanities gehören die algorithmische Verarbeitung von Daten, die statistische Modellierungen von Zusammenhängen in Daten und visuelle Analysemethoden, um Daten verstehen zu können. Diese Instrumente folgen bestimmten Routinen wissenschaftlicher Praxis: Sie verwenden erprobte Algorithmen und halten sich an Standards der Datenmodellierung. Gleichzeitig konstituieren sie aber auch die wissenschaftliche Praxis mit – sie sind, um mit Ludwick Fleck zu sprechen, Mittel zur Profilierung eines Denkstils innerhalb eines wissenschaftlichen Denkkollektivs. Dazu gehören nicht nur die erwähnten Arbeitsinstrumente, sondern auch sprachliche Mittel: Fachbegriffe, Metaphern, Stile.

Um die Verknüpfung von Arbeitsinstrumenten und wissenschaftlichen Routinen genauer zu verstehen, müssen die Praktiken und Kulturen, in denen diese Instrumente erstellt werden, reflektiert werden. Welchen Einfluss hat die Wahl einer bestimmten Programmiersprache zur Implementierung eines Algorithmus auf die Digital-Humanities-Praxis? Beispielsweise die Verwendung des „postmodernen“ Perl (Wall 1999) statt Python? Welchen Einfluss hat die Programmiersprache auf die Art der algorithmisch erstellen Visualisierung? Beispielsweise die Verwendung von D3.js, P5.js, R oder der Software Excel (Bubenhofer 2015)? Welchen wissenschaftlichen Paradigmen entspringen populäre statistische Modellierungen? Beispielsweise Topic Modelling oder Support Vector Machines? Und mit welchen sprachlichen Mitteln werden die angewandten Instrumente in den wissenschaftlichen Diskurs eingebracht und legitimiert?

Wissenschaftsgeschichtliche Ansätze von Ludwik Fleck (Fleck 1983, 2011) oder auch Thomas S. Kuhn (Kuhn 1996) lassen sich fruchtbar verknüpfen mit Überlegungen der Software Studies (Fuller 2003; Mackenzie 2006; Manovich 2013; Cox / McLean 2012), die den kulturellen Kontext und die soziale Praxis als Einflussfaktoren von Software-Erstellung und -Nutzung betonen. Ebenso existiert eine Diskussion um die Rolle von Algorithmen, statistischen Modellierungen oder