Der XML-Workflow

Wolfgang Schmidle

7. April 2010

Inhaltsverzeichnis

1 Die Verarbeitungsstadien

Beachte, dass Work Orders 1 bis 5 mit den DESpecs 1.1.2 und Work Orders 6 bis 9 mit den DESpecs 2.0 geschickt wurden. Unterschiede sind zum Beispiel das Format von Figures und von Tabellen.

Im Gegensatz zu den ersten Skripten dürfen die hier beschriebenen Bearbeitungsschritte die Zeilenstruktur verändern, zum Beispiel eine Zeile hinzufügen.

Um einen schema-konformen XML-Text zu erzeugen, müssen in der Regel nicht alle hier aufgeführten Schritte durchgeführt werden.

Die Aufteilung in prä-XML, wohlgeformtes XML und schema-konformes XML ist manchmal etwas künstlich, zum Beispiel `<hd>` erst zu `<hd/>` und in einem getrennten Schritt zu `<handwritten/>`, aber konzeptionell klarer. Ob diese Aufteilung in der Praxis hilfreich oder nicht hilfreich ist, muss sich noch zeigen.

Manchmal kann man dieses konzeptionelle Trennung jedenfalls kaum aufrechterhalten. Zum Beispiel sollte eine Form wie `deniq;` bereits zu `<reg orig="deniq;">denique</reg>` geworden sein, bevor die `<s>` markiert werden.

Legt die Hierarchie der inline-Elemente (z.B. `<var>` in plaintext, `<ref>` im inline model) eine Verarbeitungsreihenfolge nahe?

1.1 Änderungen im raw text

1. Dateiformat: Das Textformat ist bereits utf-8. Standardisiere BOM (falls nötig), Zeilenenden von CRLF zu LF. Dies ist die einzige Änderung, die automatisch im raw text gemacht wird. Nur echte Textkorrekturen werden später ebenfalls im raw text gemacht.

1.2 Vorbehandlung des Textes

1. Zeichenvorrat: Bestimme den Zeichenvorrat des Textes. Erstelle eine Liste ungewöhnlicher Zeichen. Beispiele:
   - „latin small letter iota“ aus dem Unicode-Block „IPA Extension“ anstelle von „dotless i“
   - ÿ statt ij in kursivem Text
   - „substitute“ (U+001A)
   - zwei spaces hintereinander

- Prüfe, ob Zeichen im Text sind, die in den Skripten intern verwendet werden: ¤, ¥. (Diese Währungszeichen als reservierte Zeichen wurden ausgewählt, weil sie in alten Texte wahrscheinlich nicht vorkommen, aber in vielen modernen Fonts verfügbar sind.)

2. Zeichenstandardisierung: ersetze ungewöhnliche Zeichen durch das Standard-Zeichen.

   Test: Alle Zeichen des Textes sind in einer Whitelist von Unicode-Blöcken enthalten.

3. escape sequences:

   - Löse alle escape sequences auf. Beachte, dass `\'` (U+0027) im Text zu `\'` (U+2019) werden kann. Beachte auch: `\-` wird hier zu einem combining macron, weil es so im transkribierten Text steht. Später wird das in den meisten Fällen zu einer Tilde korrigiert, weil das Makron normalerweise eine falsche Transkription einer Tilde ist. Genauso ist `\,e` in den meisten Fällen in Wirklichkeit ę, wird aber hier zu ę.
   - Löse `$` auf
   - Zeichen, die wir absichtlich nicht in die Specs aufgenommen haben, zum Beispiel `<^>9</^>` für ᵠ (modifier letter us, U+A770)
   - löse `{ }` auf: `{ij}` auf zu ij (später stillschweigend zu `ij`), `{ae}` zu ę
   - zum Schluss: Unicode-Normalform NRC

   Test: Im Text ist kein „\“ mehr vorhanden, kein `{ }` mehr vorhanden.

4. Interpunktion normalisieren: Zum Beispiel spaces vor „:“ weg. (Hier ist die Frage, ob wir Information verlieren, die wir gerne konservieren würden. Beispiel „EPISTOL AE“). Ziel ist wieder, dass sich die folgenden Skripte auf ein einheitliches Format verlassen können. Beispielsweise muss das reg-Skript, das unter anderem `q;` durch `que` ersetzt, nicht noch prüfen, ob es `q ;` gibt.

Außerdem: Skriptfehler der Chinesen: zum Beispiel der space in `'q ue` (weg aus diesem Verarbeitungsschritt, oder sogar ganz weg?)

1.3 wohlgeformtes XML erstellen

1. Prüfe ein paar Fälle, die nicht vorkommen sollten und auf Fehler bei der Transkription hindeuten:

   - `<h>`, `<mgl>`, `<mgr>`, `<fig>` jeweils nicht in allein in einer Zeile, bei `<pb>` nur noch `<rh>` erlaubt
   - nicht-existente Elemente, wie z.B. in `<scG</sc>`, oder auch `<sup>9</sup>` statt `<^>9</^>` (aber siehe unten)

- verschachtelte (vermutlich ein zuviel), und entsprechend für <h> etc.
- ohne vorhergehendes , und entsprechend für <h> etc.
- zusammengehörende Tags wie und liegen sehr weit auseinander
- zusammengehörende Tags wie <rh> und </rh> sind nicht in der gleichen Zeile

Der Sinn dieser Prüfung ist auch, dass sich die weiteren Skripte auf die Einhaltung dieser formalen Dinge verlassen können.

2. Ergänze <?xml version="1.0" encoding="UTF-8"?>, und füge root element <echo> sowie <metadata> und <text type="free"> ein. Dabei werden die Metadaten erstmal mit xxx gefüllt. (Eigentlich braucht man die Metadaten für wohlgeformtes XML noch nicht, aber es in zwei Schritte zu teilen hat sich als recht künstlich herausgestellt, weil beide Skripte einfach Textstücke in den Text hineinkopieren.)
3. Nötige Änderungen, damit der Text wohlgeformtes XML wird:
   - reservierte Zeichen in XML: & wird zu &. Formuliere das Skript so, dass es mehrere Male aufgerufen werden kann, es darf also aus & nicht & werden.
   - Attribute: <... it> wird zu <... style="it">, genauso für fr
   - ändere die Element-Namen der DESpecs in ihre Gegenstücke im ECHO Schema. Konzeptionell gibt es mehrere Teile: 1. ergänze „/“ in den ungeschlossenen Elementen wie <pb> und <hd>, 2. korrigiere verbotene Element-Namen wie <^>, 3. benenne die Elemente so, wie sie im Schema heißen. Insgesamt:
     - <pb> zu <pb/>
     - <^> zu <super>
     - <_> zu <sub>
     - <001> zu <de: unknown code="001"/>
     - <?> zu <unsure/>
     - <hd/> zu <handwritten/>
     - <h> zu <head>
     - <ind> zu
     - <tb> zu
     - <fig>, <cap>, <desc>, <var>
     - etc.

Die Aufteilung der konzeptionellen Schritte in verschiedene Skripte hat sich als künstlich herausgestellt. Es werden hier also schon Namensänderungen wie <hd/> zu <handwritten/> gemacht, die für das Ziel „wohlgeformtes XML“ nicht nötig wären. Ausnahmen: 1. <tb>, das hier mit nur vorläufig schema-konform gemacht wird. 2. <sub> und <super>, weil es später ein Skript gibt, das alle tags aufgreift, die zu <emph> werden.

4. wenn wohlgeformt: Dateiendung in xml ändern, Weitere Verarbeitungsschritte teilweise in Oxygen statt in TextWrangler, neu formatieren (whitespace muss erhalten bleiben)

1.4 Schema-konformes XML erstellen

1. <pb> schema-konform machen:
   - ergänze n. Dieses Attribut wird gebraucht, um im nächsten Schritt die Floats aus dem Absatz herauszuzuziehen. Wichtig: Bevor Figures ausgeschnitten werden, sollte geprüft werden, dass die Nummern/links in <pb> tatsächlich auf die richtigen Seiten verweisen! Denn das Skript wird wahrscheinlich die links der Floats neu durchnumerieren, wenn sich die Seitenzahlen ändern. (Oder ein getrennte Korrektur-Skript?)

     (gibt es weitere notwendige Attribute?)
   - verwandle <rh> in ein Attribut rhead in <pb>, ignoriere dabei alle Formatierungen wie kursiv, gesperrt, etc.
2. Floats aus <s> herausziehen (vor "<s> bestimmen"!):
   <anchor>, nach dem Absatz. Vorsicht bei anchored marginal notes.
   Prüfe bei anchors im Text, ob es eine zugehörige note gibt. Akzeptiere kleine Abweichungen der Symbole voneinander, zum Beispiel 3) im Text und 3 in der Fußnote.
3. <lb/> einfügen (der whitespace muss noch korrekt vorhanden sein)
4. <s> bestimmen: Beachte Fälle wie
   - et.a.b.hoc est
   - .a.b:c.d:e.f.
   - .<lb/>a.b.
   - Wort-Abkürzungen (hier wäre es einerseits hilfreich, wenn Wortabkürzungen bereits in <reg> wären; andererseits wird der Punkt am Ende von <reg> zum Beispiel in ex . 7. quinti <reg>Eucl. </reg> oft noch als Satzendepunkt gebraucht)
   - Fälle wie &c.

An dieser Stelle gibt es eventuell einen Bruch im rein automatischen Workflow. Oft ist eine manuelle Nachkorrektur nötig. Dieses Skript muss noch feiner an ein gegebenes Buch bzw. seine Zeit anpassbar sein. Und es sollte interaktiv sein. Interagiert mit `<var>`.

5. Ersetze Fomatierungs-Elemente durch `<emph style="...">`, insbesondere auch `_ _` für `style="it"`. Denke an `<sub>` und `<super>`. Verschiebe style-Informationen so wie wie möglich nach oben im xml, zum Beispiel
`<p><emph style="it">text</emph>. </p>` wird zu
`<p style="it">text. </p>`. Anderes Beispiel:
`<mgl>_eine kur-_<lb/>_ze Notiz. _</mgl>`

6. Tabellen

7. wenn schema-konform: neu formatieren. Problem bei der automatischen Formatiierung in Oxygen: `<pb>`, `<anchor>` und die Textzeilen danach werden nicht korrekt eingerückt. Es müssen keine Zeilen umgebrochen werden, sondern nur die Anzahl der LEerzeichen am Anfang normalisiert werden. Einfaches Skript schreiben!

1.5 auf das Anzeigesystem vorbereiten

1. per Hand: korrekte Daten in `<metadata>`

2. `<div>`-Struktur für das Inhaltsverzeichnis erstellen: Erstmal `<div>` von einer `<head>`-Gruppe bis zum nächsten. Automatisch erstellte `<div>` sind alle auf demselben level. `n` und `level` werden erstmal mit `n="0"` und `level="0"` gefüllt. Korrigiere anschließend (automatisch?) bei den `<head>`, die eigentlich Footer sind.

3. (per Hand: Füge `<div>` für Volume, Front, Body, Back ein. Ersetze gegebenenfalls den `type` von `<text>`. Außerdem manuelle Nachkorrektur der `<div>`-Struktur, insbesondere die Hierarchie der `<div>` und korrekte Typen. Kann auch später nachgeholt werden, z.B. im scholarly workflow.)

4. `n` und `level` in `<div>` durchnumerieren. Getrenntes Skript, weil es beliebig häufig verwendbar ist. In der Praxis verwende ich es erst direkt vor dem Einchecken des Textes, weil es das DTD-Fragment auflöst.

5. per Hand: links zu den Seitenbildern stichprobenartig überprüfen.

6. DTD-Fragment auflösen (Nebeneffekt des Skripts für `n`/`level` in `<div>` bzw. jedes beliebigen xsl-Skriptes)

7. Abbildungen ausschneiden lassen, Zordnung von Abbildungen und `<figure>` im Text überprüfen lassen.

8. noch einmal neu formatieren

Es fehlen noch <reg>, <var>, <num>, <unknown>, die zu einem großen Teil ebenfalls automatisch eingefügt (bzw. im Fall von <unknown> entfernt) werden können, allerdings mit diversen buchspezifischen Parameter-Einstellungen. Für <s> gilt das genauso, aber <s> ist so wichtig, dass es trotzdem schon vorher eingefügt wurde.

1.6 Automatisierbare Markierungen im inline model

1. <reg>: Problem der Wort-Abkürzungen mit Kasus. Verwende dort <ref>, falls möglich.

   Test: Kein Zeichen, das normalisiert werden soll, darf hinterher noch im Text (außerhalb von orig="...") sein, zum Beispiel kein Zeichen mit Tilde (mit Ausnahmen in manchen Sprachen). Für Latein/Benedetti:

   - Zeichen mit Tilde (ã ẽ ĩ õ ũ ñ) [beachte: e mit Tilde U+1EBD ist hier mit combining tilde geschrieben, denn bei e mit Tilde ist ein Font-Fehler in DejaVu 2.29]
   - combining tilde (insbesondere p̃ t̃ q̃ r̃)
   - combining acute (insbesondere q́)
   - medievalist characters: ꝑ ꝓ ꝙ ꝗ ꝕ ꝓ ꝛ
   - weitere: ę Í
   - Apostroph: insbesondere wird ’ manchmal für ' gehalten (in den Abschnitten in Benedetti mit xml: lang="it" bzw. xml: lang="ita" ist ' dagegen erlaubt)

2. <num> und <var> einfügen, um die entsprechenden Textstücke vor der morphologischen Analyse zu verbergen. (Beachte: <var> interagiert mit <s>.) Entferne <emph> in Variablen.

3. <foreign> zumindest für griechischen Text, und xml: lang.

4. <unknown> soweit wie möglich durch das korrekte Zeichen ersetzen. Insbesondere „dotless i“. Gleiche es dafür mit einer Liste ab. Prüfe vorher, ob die Liste für das entsprechende Buch korrekt ist. (Die Liste sollte korrekt sein für alle bisher getippten europäischen Bücher.) Problem, dass wir dadurch Information verlieren und Fehler der Chinesen nicht mehr gut nachvollziehen können.

Alle inline-Skripte sollen möglichst wenige Annahmne darüber machen, welche anderen inline-Skripte bereits angewendet wurden. Jedes Skript soll auch mehrere Mal anwendbar sein. (Allerdings wäre es einfacher, wenn zum Beispiel <reg> vor <var> markiert wird.)

Testparcours für jedes Skript, insbesondere für die inline-Skripte. Jeweils Zeile vorher und Zeile nachher. Nach Anwenden des Skriptes sollten beide Zeilen idenisch sein. Beispiel `var-testparcours.txt`. Sammle dort auch die Fälle, die noch nicht implementiert sind.

allgemeines Test-Skript: z.B. gibt es nach Anwenden des Skript zwei Spaces hintereinander? Das muss kein Fehler des Skriptes sein, aber es deutet auf ein Problem hin.

Gesamt-Test: Keine Punkte mehr im Text, die nicht

- Satzende-Punkte sind (`<s>Bla bla bla. </s>`)
- in einem Tag verschwinden (`<ref>ex .7. quinti Eucl.</ref>`)
- zu einer Zahl gehören (`.11.`)

1.7 Ebenfalls automatisierbar?

1. Formeln
2. Abgleich mit Donatus:
    - Einfügen fehlender Bindestriche
    - Korrektur von fehlenden/überflüssigen Spaces
    - korrigiere Standardfehler wie fumptis, fint, bumanitate in kursiv
3. Figures nachbearbeiten; beachte DESpecs 1.1.2 versus 2.0
4. Tables nachbearbeiten; beachte DESpecs 1.1.2 versus 2.0

1.8 auf den scholarly workflow vorbereiten

1. IDs einfügen (es könnte ein Modul geben, in dem das `id`-Attribut gefordet wird, und das mit der Zwiebelstruktur in diesem Stadium in Aktion tritt. Dann müssen wir nicht in den usage guide schreiben: Es ist zwar formal optional, aber es sollte verwendet werden.)

1.9 Spezial-Themen

1. GIS: `<person>`, `<place>`, `<time>`, `<event>`

1.10 scholarly workflow

1. ersetze `<wrong/>` durch `<sic/>`, löse `<unsure/>` auf
2. weitere `<reg>`, Korrekturen von bestehenden `<reg>`
3. `<ref>`
4. `<foreign>`
5. entferne library stamps
6. „old-style numerals typed as letters“, zum Beispiel `ex .II.` statt `ex .11.`, aber auch andersherum: `10. BENEDETTI` statt `IO. BENEDETTI`
7. Wörter mit einzelne griechischen oder einzelnen lateinischen Buchstaben (automatisierbar?)
8. Wörter mit einzelnen Großbuchstaben mitten im Wort (`ClaZomenius`). Häufig ist die Ursache ein fehlendes Space vor dem Großbuchstaben.

1.11 zeitlich nicht einzuordnen

1. Special instructions: beachte die besonderen Markierungen aus den Special Instructions („Black-Box-Modul“, in jedem Verarbeitungsstadium)
2. korrigiere Transkriptions- und Tagging-Fehler, die die Weiterbearbeitung stören. Für den automatisierten Workflow: korrigiere den Fehler im raw text, dann verwende die bisher verwendeten Skripte noch einmal (zumindest in der Theorie)

1.12 Besonderheiten bei chinesischem Text

1. beachte die in `echo-chinese-text` definierten Attribute
2. lateinische Zeichen können durch ihre full-width-Version ersetzt sein, zum Beispiel „?“ durch „？“
3. verarbeite character variants automatisiert
4. verarbeite character variants im scholarly workflow so gut wie möglich. Beispielsweise würde 国 durch die Unicode-Zeichenfolge □口玉 angenähert werden.
5. Wort- und Satzgrenzen markieren

2 Die Skripte

2.1 Grundstruktur der Perl-Skripte

Perl-Skripte als Textfilter-Skripte in TextWrangler: Erhalten einen Text oder ein Textstück aus einem Text. Das Textstück wird durch den output des Skripts ersetzt.

```
#!/usr/bin/perl -w
use strict;
use warnings;
use utf8;
# binmode STDIN, ':utf8';
# binmode STDOUT, ':utf8';
use open qw(:std :utf8);
use integer;
use Unicode::Normalize;


# Filter_template.pl
# 2010-04-04
# Wolfgang Schmidle

# what does it do?
# input
# output


# text input

my @text;
while(<>) { push @text, $_; }


# go through the text

foreach (@text) {

}


# text output

print @text;


# TO DO:
# ...
```

Das Einlesen des Textstücks, also alles bis `close (ORIGINAL);`, ist bei allen Filter-Skripten gleich. Unterschiede gibt es nur in den Teilen, wo der Text bearbeitet und gedruckt wird.

3 Beispiel Angeli 1659

Das Beispiel ist zufällig ausgewählt.

3.1 Versuch 1

v1: Originaler raw text

v2: Zeilenenden normalisiert (per Hand in TextWrangler). BOM ignoriert. Zeichenvorrat und Zeichenstandardisierung ausgelassen. Escape equences aufgelöst mit `Filter_escape_sequences.pl`. (Außerdem als Vorgriff: `<007>` zu i.) Problem eines einzelnen `{`, das so im Text steht. Interpunktion noch nicht normalisiert.

v3: Verwende `Filter_wellformed_xml.pl`. Noch keine Prüfung der Fälle, die nicht vorkommen sollten. Dateiendung von `txt` zu `xml` geändert.

v4: Minimal-XML per Hand eingefügt, mit Identifier `ECHO:PVNDER1Y.xml`, Sprache `lat` bzw. `la`, Autor, Titel, Jahr, DTD-Fragment. Verwende dann `Filter_schema_konform.pl`. Problem: einziger `<rh>` ist in Wirklichkeit `<head>`, per Hand geändert (beachte: Abschnitt auf italienisch!). (Sammle die Änderungen per Hand und erstelle einen verbesserten Rohtext!)

v5: Floats (erstmal nur figures) herausziehen mit `Filter_anchor-figures.pl`. Skript stammt aus dem Eipo-workflow und muss noch angepasst werden. Vorläufig: Kommentiere alle figures aus.

`<lb/>` einfügen mit `Filter_lb_einfuegen`. Für `<s>` das Skript `Filter_s_einfuegen`: Vorläufig jedes Satzzeichen ist ein Satzende. (Fehler bei de:unknown per Hand korrigiert.)

_ _ für kursiven Text vorläufig so gelassen. Es sind keine `<n>` im Text.

Ergebnis ist ein schema-konformes XML. Noch nicht neu formatiert.

v6: `<reg>`, `<var>`, `<foreign>` ausgelassen. Keine Formeln im Text. Abgleich mit Donatus ausgelassen. Daten in `<metadata>` sind bereits korrekt.

`<pb>` durchnumerieren: Verwende `Filter_pb_numerieren`. 232 Seitenbilder und 232 <pb>: Hurra!

v7: `<div>`-Struktur: am Ende durchnumerieren (Dateiname mit Zusatz `_div`, dort damit auch DTD-Fragment raus)

v8: per Hand <s> und <lb/> gelöscht und verbesserte Skripte verwendet. Die Idee ist, nicht wieder bei v4 anfangen und dann alle figures auskommentieren (v5), pb durchnumerieren (v6) und die div-Struktur einfügen (v7) zu müssen. Die Skripte sollten sich nicht beißen. Return zwischen und eingefügt (und das entsprechende Skript korrigiert).

3.2 Versuch 2

verwendete Skripte hochladen!

lokales repository, sodass ich nicht immer zwei Versionen aktuell halten muss (Versionen in Arboreal und trunk, Versionen in Unix Filters und trunk).

*

v1:
Originaler raw text, mit folgenden Änderungen, die sich aus Versuch 1 ergeben haben:
1. den einen <rh> zu <h> gemacht.
2. den Abschnitt auf italienisch kann man im Rohtext noch nicht gut markieren; aber es wäre gut, möglichst wenige Stadien zu haben, wo per Hand nicht-triviale Dinge geändert werden. Bisher: hier im Rohtext, und wenn man die Metadaten einträgt.

Zeilenenden normalisiert (per Hand in TextWrangler).

v2:
1. Zeichenvorrat mit `Filter_1_1_check_character_range` geprüft.
2. Zeichenstandardisierung war nicht nötig. (Das entsprechende Skript gibt es auch noch nicht.)
3. Escape equences aufgelöst mit `Filter_1_2_escape_sequences`. (Das eine `{` vorher entfernt und hinterher wieder eingefügt.)
4. Interpunktion: Skript nicht angewendet, weil die Gefahr von unerwünschten Nebenwirkungen noch zu groß ist, aber die Fälle im Text durchgegangen. Insbesondere gibt es keine spaces vor `. , : ; !?`

v3:
1. Verwende `Filter_2_1_check_tags`. Keine Fehler im Text gefunden.
2. Verwende `Filder_2_2_add_minimal_xml`. Stellen, die per Hand eingefügt werden müssen, sind mit ”xx”gekennzeichnet.
3. Verwende `Filter_2_3_wellformed_xml`. Die Datei ist jetzt wohlgeformt. 4. Dateiendung per Hand von `txt` zu `xml` geändert.

v4:
1. Verwende `Filter_3_1_pb`.
2. Verwende `Filter_3_2_floats`. Das Skript achtet noch nicht auf `<n>`. Es sind aber keine `<n>` im Text.
3. Verwende `Filter_3_3_lb`.
4. Verwende `Filter_3_4_insert_s`. Trägt noch nicht alle Versatzstücke aus Benedetti, Eipo etc. zusammen.
5. das `<emph>`-Skript gibt es noch nicht, ist aber bei Angeli auch nicht nötig. _ _ für kursiven Text vorläufig so gelassen.
6. das Tabellen-Skript gibt es noch nicht. Keine Tabellen in Angeli.
7. neu formatieren in Oxygen, mit den Einstellungen „Editor - Format -XML - Leerzeichenelement erhalten“ für `<pb>`, `<head>`, `<s>`, `<caption>`, `<description>`, `<variables>`: Ausgelassen, warte stattdessen auf das Skript.

(Auch wieder per Hand <de: unknown code="007"/> durch i ersetzt.)

v5:
1. Metadaten per Hand eingefügt, mit Identifier ECHO: PVNDER1Y.xml, Sprache lat bzw. la, Autor, Titel, Jahr.
2. Verwende Filter_4_2_insert_div.
3. (weggelassen)
4. Verwende div-tags.xsl. 5. schon in Versuch 1 überprüft.
6. schon mit Schritt 4 erledigt
7. (noch nicht an der Zeit)

Weiterhin <reg>, <var>, <foreign> , <unknown> ausgelassen. An <reg> arbeite ich zurzeit, insbesondere für Alvarus.

4 Weitere Texte

Weitere zufällig gewählte Texte aus Work Order 2. Versuch: ganz automatisierter Workflow, soweit nicht unbedingt per Hand nachgearbeitet werden muss.

4.1 Archimedes 1565

1.1 LF statt CRLF im raw text. BOM-Stück BF („¿“) entfernt. Wo ist der Rest von EF BB BF („ï»¿“) ?

2.1 ok
2.2 nicht nötig
2.3 Problem: { } in griechischem Text:

```
bũ ſcribit. ὀδὲ οὗτος ἡδὺς ἐ<?>στὶ ν, ὥστε η{αὶ} μὴ μαθηματι ηὸςὦν, οὐδὲ
```

→ Text zurückgestellt

4.2 Barrow 1674

1.1 ok

2.1 Line 1492 contains ɔ from Unicode block IPAExtensions:

```
quet igitur e$$e AN . AM [-- IN. IM : Quod E. D. Hinɔ</p>
```

und was bedeutet diese Schreibweise? Steht das so im Text? Ach ja, [-- steht in den special instructions.

Hinɔ ist ein Fehler:

```
http://echo.mpiwg-berlin.mpg.de/ECHOdocuView/ECHOzogiLib?url=
%2Fmpiwg%2Fonline%2Fpermanent%2Flibrary%2FU19ERSE3%2Fpageimg&mode=imagepath&pn=51
```

-> per Hand korrigiert.

2.2 ok

2.3 Line 14 contains {:
`<p it>Martii 22. 166 {8/9}.</p>`

das Skript kann noch nicht mit Brüchen umgehen.

Line 27 contains ”: `<p>Οἱ φύσει λομςι ὸι εί ς πὰντα τὰ παθήματα, ώ \~ες ξπω εξπεῖ ν, \~`

das Skript kann noch nicht mit griechischem Text umgehen.

–> alle Warnungen ignoriert, mal schauen, was passiert

2.4 ausgelassen

3.1 Lines 89, 140, 176: <rh> ohne <pb>: sehr wahrscheinlich in Wirklichkeit <head>

Skript beschwert sich inkorrekterweise über Zeilen der Form
`<pb 57><rh>L<sc>ECT</sc>. VIII.</rh>`

`<pb 58><h>L<sc>ECT</sc>. VIII.</h>` sollte wohl `<rh>` sein
`<pb 111><h>L<sc>ECT.</sc> XVI.</h>`
etc.

`<pb 127><h>ERRATA.</h>`: return fehlt
<pb><h>Lectio I.</h>: return vergessen oder eigentlich `<rh>` ?

Line 7826 : incorrect tag:
`cùm ergò <n (_c_) ſit KG --] EG; erit LG --] FG; unde liquet rectam`

Line 8456 : </mgl> should be at the end of the line:
`<mgl (_c_)>3 _hujus_.</mgl></p>`

Line 10387 : Unknown tag or incorrect attribute:
`ſecet in ξ; ſit AG. AC:: AC. ξK; & per K intra _aſymptot<x>o</x>s_`

–> leicht zu korrigieren, aber sicherlich nicht automatisch.

4.3 Bianconi 1746

2.1 ok

2.3 ok

3.1 diverse Fehlermeldungen für Zeilen der Form
`<pb>(LXIX)</pb>`

per Hand in `<pb (LXIX)>` geändert: `<pb>([^<]+)</pb>` → `<pb \1>`

(aber noch nicht im raw text: allgemeines Problem der doppelten Arbeit, wenn man nicht alle Skripte neu anwenden will.)

sonst ok

(man könnte noch das Skript für römische Zahlen so variieren, dass es aus dem o-Attribut noch die Seitenzahl als arabische Zahl generiert.)

3.2 ok

3.3 ok, wohlgeformt. Datei umbenennen

4.1 ok

4.2 ok

4.3 ok

4.4 Fehler im Skript: `<s>potendo dire anch' io, : </s>` nachgetragen: `([^>])(\r</p>)` → `\1</s>\2`

(und Fehler: kein return bei `<p style="it"><s>`)

4.5 <head><red> per Hand in <head style="red»

Small-caps-Skript angewendet, und per Hand: `<head style="red sc">Appresso Simone Occhi, </h`

per Hand table schema-konform gemacht

2x `<handwritten>` per Hand entfernt (das Skript macht das noch nicht automatisch)

→ schema-konform

5.1 Metadaten per Hand, ok

5.2 ok

5.3 –

5.4 ok

5.5 nicht überprüft