
# Data Collection and Preparation for Currency Adaptation

Applied Data Science in an Industrial Banknote Image Recognition Framework

Degree course: [BSc Informatik]
Authors: [Sophie Haug]
Constituent: [CI Tech Sensors AG]
Experts: [Prof. Dr. Macha Kurpicz-Briki]
Date: 17.04.2021

# Versions

| Version | Date | Status | Remarks |
|---|---|---|---|
| 0.1 | 01.08.2013 | Draft | Lorem ipsum dolor sit amet |
| 0.2 | 21.08.2013 | Draft | Phasellus scelerisque |
| 0.3 | 02.09.2013 | Draft | Donec eget aliquam urna. Lorem ipsum dolor sit amet |
| 1.0 | 26.01.2014 | Final | Lorem ipsum dolor sit ametPhasellus scelerisque, leo sed iaculis ornare |
| 1.1 | 31.01.2014 | Correction | Layout changed |
| 1.2 | 07.02.2014 | Addition | Chapter 1.1 extended |

# Management Summary

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus scelerisque, leo sed iaculis ornare, mi leo semper urna, ac elementum libero est at risus. Donec eget aliquam urna. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc fermentum nunc sollicitudin leo porttitor volutpat. Duis ac enim lectus, quis malesuada lectus. Aenean vestibulum suscipit justo, in suscipit augue venenatis a. Donec interdum nibh ligula. Aliquam vitae dui a odio cursus interdum quis vitae mi. Phasellus ornare tortor fringilla velit accumsan quis tincidunt magna eleifend. Praesent nisl nibh, cursus in mattis ac, ultrices ac nulla. Nulla ante urna, aliquet eu tempus ut, feugiat id nisl. Nunc sit amet mauris vitae turpis scelerisque mattis et sed metus. Aliquam interdum congue odio, sed semper elit ullamcorper vitae. Morbi orci elit, feugiat vel hendrerit nec, sollicitudin non massa. Quisque lacus metus, vulputate id ullamcorper id, consequat eget orci .

# Contents

**Management Summary** **i**

**1. Introduction** **1**
1.1. Introduction . . . . . 1

**2. Project Goals** **3**
2.1. Initial Position . . . . . 3
2.2. Project Goals . . . . . 4

**3. Project Methods** **5**
3.1. Introduction of Note Sets as a new note data container . . . . . 5
3.2. NoteSet Generation . . . . . 5
3.3. Feasibility of NoteSet replacing Notelist in the entire Toolchain . . . . . 6

**4. Note Set Data Container Specification** **7**
4.1. Note Set Annotation Specification . . . . . 7

**Glossay** **11**

**Bibliography** **13**

**List of figures** **15**

**List fo tables** **17**

**APPENDICES** **19**

**A. Arbitrary Appendix** **19**

**Index** **19**

**B. Additional Appendix** **21**
B.1. Test 1 . . . . . 21

**C. Content of CD-ROM** **23**

# 1. Introduction

## 1.1. Introduction

### 1.1.1. Purpose of this document

This document serves as a documentation for the project "Data Collection and Preparation for Currency Adaptation"which is carried out in the scope of the BFH module "Project 2" BTI7302.

The client of the project is the company CI Tech Sensors AG (subsequently referred to as CI Tech) and the project is supervised on the part of CI Tech by Lukas Burger, head of application software.

The project is supervised on the part of BFH by Prof. Mascha Kurpicz-Briki.

### 1.1.2. Vision

In the scope of this project a process for the data collection and preparation for the currency adaptation workflow should be developed. Namely, the currency data needed to train and develop algorithms in CI Tech's currency recognition and classification framework should be provided as centralized labelled data sets. These data sets can be generated automatically from the centralized Data Pool, they are version controlled and a uniform process for their updating and management will be implemented. With this change, it will be possible to directly track the influence a new banknote recording has on an existing currency recognition software. Also it will be possible to use dedicated sub-data sets for training specific algorithms, for example data sets consisting of banknotes with a dogear.

# 2. Project Goals

## 2.1. Initial Position

### 2.1.1. Background

The 80/20 rule of Data Science states that most data scientist spend about 20% of their time on actual data analysis and 80% of their time finding, cleaning and preparing data. The process of readying data for the training, testing and implementation of an algorithm thus is not only a highly important step in the Data Science Lifecycle, it's maybe the costliest.

CI Tech Sensors has over 30 years experience in the field of Banknote Image recognition. One of its core competences is the development of currency adaptation software. Namely the so-called Currency Data File is a software, which is loaded onto the banknote reader, and enables it to identify and classify banknotes for specific currencies, as well as determine their fitness and authenticity. This software is continuously updated and developed by a dedicated team of Currency Adaptation Specialists at CI Tech Sensors, whose work relies on the availability of a huge data pool of Banknote Raw Data.

Over the last three years, there has been ground-breaking shift in raw data management: From folder-based data storage on both local and central file drives - which heavily relied on filenames for identification and was thus prone to ambiguity and duplication of data - towards a centralized data pool, where each file was given a unique ID. At the same time a cloud based Microservice architecture (MOVEm Data Management Toolchain) for the management and maintenance of this data pool was developed. This development comes with a number of exciting possibilities, some of which to explore is the goal of this project. The focus will be on the collection and preparation of tailored banknote sets.

In earlier days, the collection and preparation of banknote data had to be manually done by the Currency Adaptation Specialist. They would have to navigate to the relevant folder, select the relevant files and sort them by denomination, emission, orientation and other criteria, an information taken from a file's annotation. Not only is this process cumbersome and error-prone in itself, it also solely relies on the file annotation, which is made by a human annotator before the actual banknote recording, and thus can contain errors as well.

Today it's already possible to group banknote raw data into sets, using a text-based file format called notelist file . Adaptations specialists will manage and create their own notelist files locally according to their specific needs. Notelists contain references to individual file IDs and Note IDs as well as additional annotational information and the file path where files can be found. However this format comes with some problems of its own.

### 2.1.2. Overview of the existing CI Tech adaptation toolchain

In this section a short overview is given over some of the tool in the existing CI Tech adaptation toolchain. This list is by far not exhaustive and only names the tools which are mot relevant to the project at hand

- *MCM* (MOVE Currency Data Modeller) is a desktop application and the main tool of currency adaptation specialists in their daily work. It is used to develop, test and adapt so called *Currency Data Files*, the software which is loaded onto the banknote reader and enables it to recognize and classify banknotes. MCM is a software monolith that has grown and extended its functionality over decades, but it doesn't lend itself very easily to modularization. MCM is also the interface to the entire algorithmic framework. Recordings of banknotes can be loaded from the filesystem as .NIF files (binary files) or .nl (XML Format) files. The latter also allows loading notes from the central Data Pool via MFX instead of the local file system.

- *MFX* (MOVE File Index) is a web based microservice that allows the lookup of files from the central Data Pool. Namely, it allows to retrieve the storage path in the pool for a given File ID. Using this service, MCM can load files by File ID instead of a file path.
- *MDS* (MOVE Data Service) is a web based microservice that allows the lookup of note and image data. It allows to query the most relevant note data found in the (binary) .NIF files as well as images via REST api. A future vision which is in ongoing development is that entire sets of notes could be loaded into MCM via this REST interface instead of parsing large amounts of binary data.

### 2.1.3. Stakeholders

- The Head of Application Software will supervise the project and provide technical advice
- The Head of Currency Adaptation will provide additional support from the perspective of Currency Adaptation specialists.
- The Currency Adaptation team will be the primary user group

## 2.2. Project Goals

### 2.2.1. Accurate, consistent and complete reference data sets

The goal is to have accurate and consistent data sets, which are generated and persisted in a way that makes use of the Microservice based MOVEm Data Management Toolchain. Namely these test sets should fulfil the following criteria:

- Accuracy: Data in test sets has been checked for correct labeling and or given a label in case it was missing.
- Consistency: Different adaptations for the same currency should use the same data sets.
- Completeness: In the generation of reference note data sets, all data in the pool should be considered.

### 2.2.2. Mechanism for subclassifying Data Sets according to context-dependent criteria

It should be possible to generate data sets according to user specific criteria like: "All EUR notes that were classified as fake", "All USD notes for which tape was detected". A process should be installed which allows specialist users to generate such datasets from queries. These datasets will have to fulfil the same criteria as the reference data sets.

### 2.2.3. Automated process for generating and maintaining Reference Note Data Sets

A process has to be defined for generating Reference Data Sets automatically as well as updating and expanding them when new data is added to the Data Pool. A process has to be defined for generating Reference Data Sets automatically as well as updating them when new data is added to the Data Pool. Part of this process is also the management of Reference Data Sets in a Version Control System.

### 2.2.4. Technical Analysis of the existing Toolchain

To achieve the previously listed goals, the introduction of a new data container will be necessary to make the existing toolchain compatible with this new paradigm. While this shift can only happen over a longer period, it would be very undesirable to have a new data container or format which will only be deployable for a part of the toolchain. The goal is therefore to have an assessment on the feasibility of implementing the new data container into the existing toolchain.

# 3. Project Methods

## 3.1. Introduction of Note Sets as a new note data container

We will introduce Note Sets as a new data container for referencing and persisting sets of note data. While these Note Sets will be exported to files, they do not constitute a file format in the narrower sense, but rather a new paradigm as how to think of collection of notes: No longer as notes that were recorded in the same transaction and thus in the same binary file (in earlier times this was the only way one could think of collections of notes), but as a collection of IDs.

The note set data container will have the following advantages:

- Compactness: Note Sets will be minimalist in the sense that notes are references by ID only. There is no additional information like annotation stored in the note set format.
- Independency of file locations: With the introduction of MFX the actual location of a file in the pool becomes redundant. Therefore, the note set does not store any file paths.
- Equivalence with respect to annotation: Unlike the Notelist format, Note Sets will no longer have an individual annotation per note. Rather, they have at most one annotation per note set, which is to be persisted in the note set's filename. Thus, note sets are considered equivalent with respect to annotation

Meanwhile, the new data container must at least satisfy the following requirements:

- Convertability to and from the existing Notelist format. The Notelist format has been established as the standard way of storing data sets in Currency Adaptation. Even if one were to pursue the goal of replacing the Notelist format by the Noteset format, convertability would be needed at least for a period of transition. The reasons for that are that a) Noteset import is supported by MCM however in a yet limited way. b) the Python MCM wrapper library mcmp, which is used to load notes and run algorithms from outside MCM, does currently not support the loading of Notesets.
- Differentiability. Since they are so minimalist, Note Sets are more ideal than previous formats for quick comparison. We need a way to compare and find the difference between Note Sets reliably and efficiently.

The Note Set data container will be introduced in a small python library providing the abovementioned functionality. This library is then deployed as a python package and can be used in the python toolchain, including MDS and MFX, as well as in Jupyter notebooks. In a later step it remains to analyze the feasibility of introducing this data container in the .NET toolchain as well.

## 3.2. NoteSet Generation

Using different methods and for a set of exemplary currencies a first version of NoteSets are generated from the Data Pool, In the course of this existing client software of MDS and MFX are extended to allow for more complex queries.

In a first phase Jupyter Notebooks will be used to generate the NoteSets. Different methods and their results, i.e. the resulting NoteSets, will be compared and presented to the Currency Adaptation Team. Once the NoteSet generation process has been refined on a technical level, it remains to decide how to include it in the adaptation specialist's workflow.

The NoteSet generation workflow in this development phase can be described as follows:

1. Crawl MFX for a chosen currency to obtain all file paths of .NIF files for that currency

2. With a list of all FileIDs and corresponding filepaths, use a labeling method to label all the notes in alle the files
3. Based on the obtained labeling, split the set of all Notes into labelled NoteSets (within a NoteSet, all elements have the same label)

It remains thus to choose a labeling method. The following methods will be used and compared

1. For each note in the set of all notes, query MDS to obtain the recognition and classification done by the reader for this note. We use the banknote reader (and the CDF which was loaded at the time of recording) as the labeller.
2. Load all the notes from the set of all notes into MCM and use algorithm(s) from the algorithmic framework of MCM to determine properties of each note based on which the notes can be labelled. We use MCM as the labeller. To do this in an automated way, a Python wrapper for MCM has to be used, which unfortunately only allows restricted use of MCM.
3. Use an algorithm external to MCM as a kind of plug-in to process and label the notes. This is the most experimental method, but maybe also the most interesting one. It has the advantage that we are free of MCMs monolithic architecture in our labeling process, but also the challenge that the MCM algorithmic framework is highly developed and refined and can hardly be bested by an external classifier.

The following figure illustrates the three methods

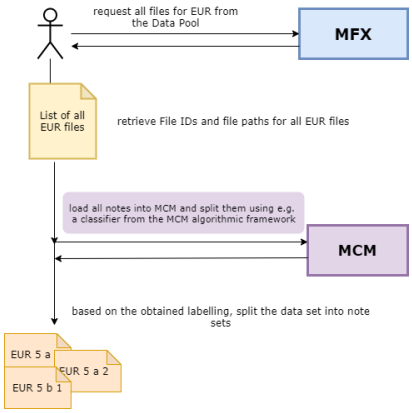


Figure 3.1.: A really Awesome Image

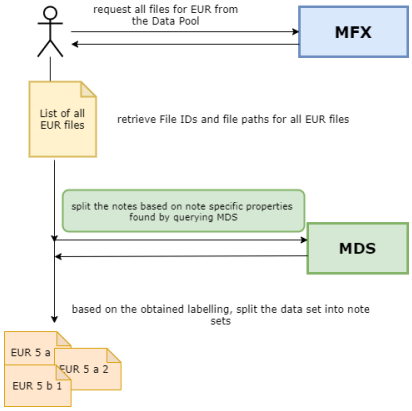


Figure 3.2.: A really Awesome Image

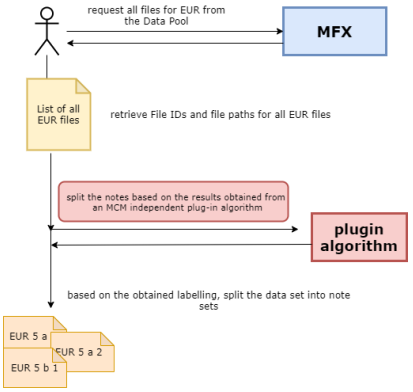


Figure 3.3.: A really Awesome Image

## 3.3. Feasibility of NoteSet replacing Notelist in the entire Toolchain

In a next step an analysis is done of what it would mean to replace the standard Note Data Container Notelist by the newly introduced NoteSet. Especially, how complex costly it would be to refactor the main adaptation tool MCM to work with NoteSets. This is important because MCM will always set the standard of the currency adaptation workflow and if NoteSets are to be the standard data container, MCM needs to be able to handle them in an efficient and user-friendly way.

# 4. Note Set Data Container Specification

## 4.1. Note Set Annotation Specification

### 4.1.1. Motivation

As already mentioned, we introduce Note Sets as a container for notes which are equivalent with respect to a yet to be defined annotation, the labeling of the set. Since the Note Set itself is basically just a set of IDs or keys, all substantial information will be in the annotation. The first challenge is to select the properties which should be part of the annotation. The second is the question of how to persist the annotation within the Note Set. Since we do not want to introduce a new Note container format, it's not possible to add the annotation as a sort of header in an XML or JSON format. This would mean that a new format has been introduced and some kind of schema defined for it.

The only option that remains is thus to store the set's annotation within the filename itself when exporting a Note Set to disk. For this a special file format string has to be introduced (see below).

### 4.1.2. Criteria for selecting annotational elements

The selection of the elements which should be part of the annotation was overall not that easy, on the one hand because there are hundreds of properties in a NIF file and we want to be as specific as possible, on the other hand, a Note Set's annotation should be as simple as possible, namely because we want to persist the annotation in a Note Set's filename, which ideally should be a readable and not too long string. Finally after having consulted with the adaptation specialist team, the selection was done by the following criteria: The mandatory properties are properties which are needed to identify notes in MCM, the same criteria needed when mapping a loaded note to a reference note in a CDF. The optional properties (with the exception of comment) are criteria for which it is very likely that a user wants to split or filter a set of notes by when analyzing or training algorithms.

### 4.1.3. Mandatory Elements of the Note Set Annotation

The following attributes form the mandatory part of the Note Set's annotation. These are all derived from properties set in the NOTEMSG2 chunk of the NIF file.

**Currency** Corresponds to the ISO string matching the the numeric ISOCODE value in the NOTEMSG2 Chunk.

**IsoGroup**

**Emission** Emission of the notes in the Note Set, corresponds to the value of the Chunk EMISSION in NOTEMSG2

**Orientation** Orientation of the notes in the Note Set, corresponds to to ORIENTATION in NOTEMSG2

**Type** NoteType (indicating the reference note this note is matched to by the CDF which was used in recording). For currencies like, GBP with different printers (for example HKD[1] or GBP[2]), this type is necessary to identify a note. For other currencies with only one printer, like CHF, denomination and emission is usually enough to identify a note.

[1] Hong Kong Dollar
[2] Great Britain Pound

### 4.1.4. Optional Elements of the Note Set Annotation

The following attributes are optional elements of a Note Set Annotation. They are not identifying properties but qualify a note (set).

**Class** Corresponds to the value CATEGORY in NOTEMSG2 Chunk and is an attribute for authenticity and fitness of a banknote, for example, Category 4a notes are authentic and fit notes, Category 2 are counterfeits. It's a likely use case that a user would want to split notes across a currency by category to train and analyze fitness and security algorithms.

**Security and Fitness flags Flags** Security and Fitness Flags are set by certain algorithms and indicate that a note might be classified as unfit or counterfeit. For example, the fitness flag STAIN indicates that the note has some kind of smudging, the security flag ROBBERY_INK indicates that the presence of robbery ink was detected.

**Reject Reason** This is an enum value found corresponding to the value REJECT_REASON in NOTEMSG2 chunk. It indicates the reason why a note was rejected or that it was not rejected.

**Comment** This is an optional comment added by the user and the only value which does not come from the NIF file

### 4.1.5. Specifying the file format string

As already stated, the Note set's annotation is to be converted into a filename when exporting a Note Set to ASCII. The file format string to be constructed has to be readable, complete and indicate the optionality of properties, i.e. distinguish the optional properties from the mandatory ones. Finally, the goal is to codify the format into a regular expression and implement an annotation parser in the python noteset library *pynoteset*.

#### Preliminary Decisions

- Generic placeholder for non-specified mandatory properties: Since we allow Note Sets to be merged, we have to anticipate the scenario where e.g. a Note Set contains notes of various denominations. Therefore we need a placeholder for all mandatory properties indicating the state of 'Undefined' or mixed. This placeholder is 'XXX' for currencies and isogroups and 'X' otherwise, a practice which is already in use in the recording of note data and thus one that users are already familiar with.
- Allowing of both upper- and lowercase spelling: While it is common to use uppercase notation for currency ISO codes, we will allow both upper- and lowercase notation to make the already fairly complicated string format less restrictive.

### 4.1.6. Security and Fitness Flags

Security and Fitness Flags are set by certain algorithms and indicate that a note might be classified as unfit or counterfeit. For example, the fitness flag STAIN indicates that the note has some kind of smudging, the security flag ROBBERY_INK indicates that the presence of robbery ink was detected.

Since it is a probable use case that one might want to generate set of notes where a specific flag is set (to train the algorithm which sets said flag), it's important that flags be persisted in a note set's annotation as well.

The problem is that there could be multiple flags set for the notes in one set, say all notes with flags STAIN and INK. In practice, this use case is not too likely, but we want to allow this scenario to remain as generic as possible. We don't, however, want the annotation string to become too long and hard to read. Apart from impeded readability, another problem is that we now have in addition to the comment property a second optional property which has a multplicity of 1...n. This makes correct parsing of an annotation string a bit more difficult.

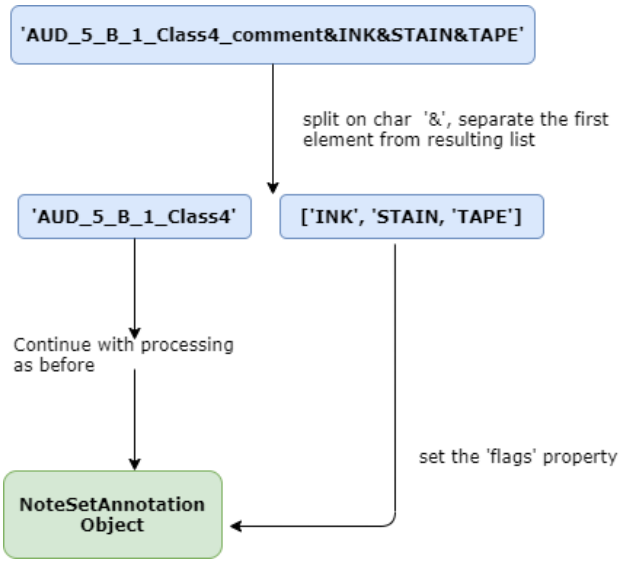


Figure 4.1.: Persisting flags in Note Set filename

# Bibliography

[1] H. Kopka, *LaTeX, Band 1: Eine Einführung*. Pearson Studium, 2002, vol. 3.

[2] F. Marti, "Schreiben über Technik – Redaktion und Gestaltung von technischen Berichten und anderen technikbezogenen Texten," Berner Fachhochschule, 2006.

# List of Figures

3.1. A really Awesome Image . . . . . 6
3.2. A really Awesome Image . . . . . 6
3.3. A really Awesome Image . . . . . 6

4.1. Persisting flags in Note Set filename . . . . . 9

# List of Tables

# APPENDICES

## A. Arbitrary Appendix

The European languages are members of the same family. Their separate existence is a myth. For science, music, sport, etc, Europe uses the same vocabulary. The languages only differ in their grammar, their pronunciation and their most common words. Everyone realizes why a new common language would be desirable: one could refuse to pay expensive translators. To achieve this, it would be necessary to have uniform grammar, pronunciation and more common words. If several languages coalesce, the grammar of the resulting language is more simple and regular than that of the individual languages. The new common language will be more simple and regular than the existing European languages. It will be as simple as Occidental; in fact, it will be Occidental.

# B. Additional Appendix

## B.1. Test 1

To an English person, it will seem like simplified English, as a skeptical Cambridge friend of mine told me what Occidental is. The European languages are members of the same family. Their separate existence is a myth. For science, music, sport, etc, Europe uses the same vocabulary. The languages only differ in their grammar, their pronunciation and their most common words. Everyone realizes why a new common language would be desirable: one could refuse to pay expensive translators. To achieve this, it would be necessary to have uniform grammar, pronunciation and more common words. If several languages coalesce, the grammar of the resulting language is more simple and regular than that of the individual languages. The new common language will be more simple and regular than the existing European languages.

### B.1.1. Environment

It will be as simple as Occidental; in fact, it will be Occidental. To an English person, it will seem like simplified English, as a skeptical Cambridge friend of mine told me what Occidental is. The European languages are members of the same family. Their separate existence is a myth. For science, music, sport, etc, Europe uses the same vocabulary. The languages only differ in their grammar, their pronunciation and their most common words. Everyone realizes why a new common language would be desirable: one could refuse to pay expensive translators. To achieve this, it would be necessary to have uniform grammar, pronunciation and more common words.

# C. Content of CD-ROM

Content of the enclosed CD-ROM, directory tree, etc.