Data Tools for MIMO

Contents

[1 Introduction 2](#_Toc265068369)

[2 Overview of MIMO data tool 2](#_Toc265068370)

[3 Individual Tools 4](#_Toc265068371)

[3.1 Development Environment 4](#_Toc265068372)

[3.2 M+Exporter 4](#_Toc265068373)

[3.3 Fix MPX 4](#_Toc265068374)

[3.4 MPX-RIF (Resource Information Faker) 4](#_Toc265068375)

[3.5 MPX Decider 4](#_Toc265068376)

[3.6 MIMO Resource Uploader 4](#_Toc265068377)

[3.6.1 TODO 4](#_Toc265068378)

[3.7 MPX2LIDO Mapping 4](#_Toc265068379)

[3.8 OAI Server 4](#_Toc265068380)

Maurice Mengel, Version 1 (2010-06-23)

# 1 Introduction

This document is meant for AR and might be also useful for writing contracts (Werkverträge) if necessary.

# 2 Overview of MIMO data tool

1 Development Environment

State: needs to be setup and explained to AR

cygwin

cygwin packet mangement, git, github, perl

Metadata Data storage on S3

e-mail

2 M+Exporter (PERL, XSLT)

Input: RTF as produced by MuseumPlus (M+)

Output: MPX-lvl2

Purpose: keep everything in the logic of M+, just transformation

Install type: as remote service. Send by RTF by mail and receive output by mail

State: works slowly, but reliable

3 Fix MPX (XSLT)

Input: MPX-lvl2

Output: MPX-lvl2

Purpose: correct semantic errors

State: Ready, needs adaptions to instrument data and possibly debugging (involves input from AR)

4 MPX-RIF (Resource Information Faker)

Input: MPX-lvl2

Output: MPX-lvl2

Purpose: fake mume information based on filenames and directories

Remark: We should only use this for audio and video, not for images, but we may use it for images if no other

Install type: remote or local

State: proof-of-concept available, needs a lot of work

5 MPX Decider (perl)

Input: MPX-lvl2

Output: MPX-lvl2

Purpose: mark data which is public or private based on rules, possibly also filter data accordingly

Install type: remote or local

State: not begun, concept see: mauricemengel.de

6 MIMO Resource Uploader (Perl)

Input: MPX-lvl2

Output: none

Result: will create jpgs for MIMO (if necessary) and upload for files

Install type: needs to be executed at SMB

State: not yet begun

7 MPX2LIDO Mapping for Musical Instruments

Input: MPX-lvl2

Output: LIDO (actually, we have to export into OAI-LIDO)

Technology: this is actually part of OAI Server, but excluded since important and time-consuming

8 OAI Server

Input: MPX-lvl2

Output: various metadata formats including LIDO

Install type: on MIMO web server

State: not yet begun

# 3 Individual Tools

## 3.1 Development Environment

## 3.2 M+Exporter

## 3.3 Fix MPX

## 3.4 MPX-RIF (Resource Information Faker)

## 3.5 MPX Decider

## 3.6 MIMO Resource Uploader

### 3.6.1 TODO

Purpose: Basically upload jpg to MIMO, create jpgs from tifs if necessary. Currently unclear if we should have a local copy.

Algorithm

Write a perl-script

1. walk thru MPX and identify resources (images, sound, video) which are meant for MIMO (freigabe="public" or freigabe="web" etc.)
2. first create jpgs from tifs if not already a jpg
3. should we store a local copy?
4. possibly: quality control check if image has the correct format (not bigger than 800 px longest size)
5. new name has mulId as identifier
6. upload to MIMO platform
7. logging: report when somethings missing, report when successful? report if already exists?
8. how to do updates work? Check if already exists before upload

## 3.7 MPX2LIDO Mapping

## 3.8 OAI Server