

Martin Schröttner

Generating Metadata of Cultural Heritage 3D-Assets

Master's Thesis

Graz University of Technology

Institute of Computer Graphics and Knowledge Visualization
Head: Univ.-Prof. Dipl.-Ing. Dr.techn. Dieter W. Fellner

Supervisor: Univ.-Doz. Dr.-Ing. Sven Havemann

Graz, September 2013

This document is set in Palatino, compiled with pdfL^AT_EX₂ ϵ and Biber.

The L^AT_EX template from Karl Voit is based on KOMA script and can be found online: <https://github.com/novoid/LaTeX-KOMA-template>

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, _____

Date

Signature

Eidesstattliche Erklärung¹

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am _____

Datum

Unterschrift

¹Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Abstract

Digitization is one way to facilitate access to our cultural heritage and to reduce part of the loss of information caused by deterioration and damage. Rich and significant metadata consisting of information about the digital data and the documentation of its origins (from the real object to the 3D-model) is crucial for searching and retrieval of digitized assets in a cultural heritage database. However, generating metadata is expensive as it is a very time consuming semi-manual process. Additionally, new and improved technologies for digitization lead to an exponential increase of digitized cultural heritage objects. Therefore, novel approaches for mass generation of metadata are indispensable.

This thesis presents an approach that is generic, minimizes user assistance, and is customizable for different metadata schemes and storage formats, as it is based on generic forms. The approach and its implementation as the software tool *MetadataGenerator* were developed in context of the project 3D-COFORM with the aim of researching on technologies for 3D-documentation in the cultural heritage domain. The approach scales well and was tested in the course of several campaigns in collaboration with partners of the 3D-COFORM project, such as the Archaeology Museum Schloss Eggenberg.

Zusammenfassung

Digitalisierung unseres kulturellen Erbes ist eine Möglichkeit, um einerseits den Zugang zu erleichtern und andererseits einen Teil des Informationsverlustes, der durch Verfall und Beschädigung entsteht, zu verringern. Um die digitalisierten Kulturgüter in einer Datenbank zu suchen und abfragen zu können, werden umfangreiche und aussagekräftige Metadaten benötigt, welche Informationen über digitale Daten und die Dokumentation der Entstehung (vom realen Objekt bis zum 3D-Model) beinhalten. Die Metadaten-erzeugung ist jedoch ein halbautomatischer, sehr zeitaufwendiger Prozess und kann daher zu hohen Kosten führen. Durch ständige Verbesserung und Neuentwicklungen der Technologien, die zur Digitalisierung zur Verfügung stehen, ist ein exponentieller Anstieg an digitalisierten Kulturerbe-Gütern zu beobachten. Deshalb werden neue Lösungsansätze für die Massenerzeugung von Metadaten benötigt.

In dieser Masterarbeit wird ein generischer Lösungsansatz präsentiert, der den Aufwand für den Benutzer reduziert. Die Verwendung von generischen Eingabe-Formularen ermöglicht eine einfache Anpassung an verschiedene Metadaten-Schemata, sowie an verschiedene Formate zum Speichern der generierten Metadaten. Der Lösungsansatz und die Umsetzung, als Software-Werkzeug *MetadataGenerator*, wurden als Teil des Projektes 3D-COFORM entwickelt. Das Projekt hatte das Ziel, an Technologien für 3D-Dokumentation im Bereich Kulturerbe zu forschen. Der Ansatz skaliert sehr gut und wurde im Rahmen mehrerer Kampagnen in Zusammenarbeit mit 3D-COFORM Projektpartnern, wie dem Archäologiemuseum Schloss Eggenberg, getestet.

Publications

Some parts, ideas and figures of this thesis have appeared previously in the following publications:

- [30] Xueming Pan, Thomas Schiffer, Martin Schröttner, René Berndt, Martin Hecher, Sven Havemann, and Dieter W. Fellner. “An Enhanced Distributed Repository for Working with 3D Assets in Cultural Heritage.” EN. In: *Progress in Cultural Heritage Preservation. 4th International Conference, EuroMed 2012, Limassol, Cyprus, October 29 – November 3, 2012. Proceedings*. Ed. by Marinos Ioannides, Dieter Fritsch, Johanna Leissner, Rob Davies, and Fabio Remondino. Vol. LNCS. Lemesos: Springer, Oct. 2012, pp. 349–358. ISBN: 978-3-642-34233-2. DOI: 10.1007/978-3-642-34234-9_35.
- [31] Xueming Pan, Thomas Schiffer, Martin Schröttner, Sven Havemann, Martin Hecher, René Berndt, and Dieter W. Fellner. “A Scalable Repository Infrastructure for CH Digital Object Management.” EN. In: *IEEE-EXplore digital library*. Milano: IEEE, Sept. 2012, pp. 219–226. ISBN: 978-1-4673-2564-6. DOI: 10.1109/VSMM.2012.6365928.
- [32] Xueming Pan, Martin Schröttner, Sven Havemann, Thomas Schiffer, René Berndt, Martin Hecher, and Dieter W. Fellner. “A Repository Infrastructure for Working with 3D Assets in Cultural Heritage.” In: *International Journal of Heritage in the Digital Era* (Volume 2, Number 1 / March 2013 2013), pp. 143–166. ISSN: 2047-4970 (Print). DOI: 10.1260/2047-4970.2.1.143.
- [39] Martin Schröttner, Sven Havemann, Maria Theodoridou, Martin Doerr, and Dieter W. Fellner. “A Generic Approach for Generating Cultural Heritage Metadata.” EN. In: *Progress in Cultural Heritage Preservation. 4th International Conference, EuroMed 2012, Limassol, Cyprus, October 29 – November 3, 2012. Proceedings*. Ed. by Marinos Ioannides, Dieter Fritsch, Johanna Leissner, Rob Davies, and Fabio Remondino. Vol. LNCS. Lemesos: Springer, Oct. 2012, pp. 231–240. ISBN: 978-3-642-34233-2. DOI: 10.1007/978-3-642-34234-9_23.

Acknowledgements

This master thesis would not have been possible without the help of many people. First of all, I would like to express my gratitude to my supervisor Sven Havemann. Without his guidance and persistent help, this thesis would not have been possible. Special thanks also to (Peter) Xueming Pan, René Berndt and Christian Caldera whose encouragement, suggestions and comments were invaluable. I received generous support by the entire staff of the Institute of Computer Graphics and Knowledge Visualization at Graz University of Technology as well as Fraunhofer Austria Research. Furthermore, I would like to thank my friends, and family for helping and supporting me during the entire studies. Last but not least, I am deeply grateful to Michaela for reviewing and for never stopping encouraging me.

Contents

Abstract	iv
Publications	vi
1 Introduction	1
1.1 Motivation	2
1.2 Outline	4
2 Related Work	6
2.1 Metadata Schemes	6
2.1.1 Dublin Core	7
2.1.2 CIDOC-CRM and CRMdig	7
2.2 Cultural Heritage Metadata Generation	9
2.3 3D-COFORM	12
2.3.1 Repository Infrastructure	12
2.3.2 IngestionTool	14
3 Digitization of CH 3D-Assets	16
3.1 Digitization Processing Chain	16
3.1.1 The Way to 3D-Data of Real Assets	16
3.1.2 Post Processing	20
3.1.3 Digital Master Model	23
3.2 Use Cases	24
3.2.1 Change and Restoration Monitoring	24
3.3 Benefits of Digitized Cultural Heritage Assets	26
4 Generating Cultural Heritage Metadata	27
4.1 Metadata Sources	28
4.2 Combining and Structuring	30
4.3 Storing Metadata	32
4.3.1 Metadata Storage Places	32
4.3.2 Information Linking	32
4.3.3 Metadata Storage Formats	33
4.3.4 Format Conversation	33
4.4 Iterative Metadata Refinement	36

5	A Generic Approach for Generating Cultural Heritage Metadata	37
5.1	Generic Dynamic Input Forms	38
5.2	Generic Formats	40
5.2.1	Forms Definition Format	40
5.2.2	Intermediate Storage Format	41
5.2.3	Template Definition Format	42
5.3	Directory Structure Information	44
6	MetadataGenerator	45
6.1	Introduction	46
6.2	Features	47
6.3	System Design	48
6.3.1	Graphical User Interface (GUI)	48
6.3.2	Storage	53
6.3.3	Transform	53
6.4	Avoiding Errors	53
6.5	CIDOC-CRM / CRMdig Forms	54
6.5.1	Project Event	54
6.5.2	Capture Event	56
6.5.3	Detailed Sequence Event	57
6.5.4	Acquisition Event	58
6.5.5	Process Event	61
6.6	Ingest Script Generation	64
6.7	Implementation Details	65
7	Testing and Results	68
7.1	Test Campaigns	68
7.2	MG Use Case Description	70
7.3	MG Results	70
8	Conclusion	72
8.1	Contribution and Benefit	72
8.2	Future Work	73
	Bibliography	74

List of Figures

1.1	Image of the burnt away painting from Claude Monet	1
2.1	CIDOC-CRM Example: 1945 Yalta conference	8
2.2	Example for CRMdig extending CIDOC-CRM	9
2.3	RI system overview.	13
2.4	IngestionTool	15
3.1	3D-Scanner Example: Microsoft® Kinect™	17
3.2	Overview of the ARC3D system	18
3.3	3D-Modelling Software Example: Autodesk Maya 2012	19
3.4	GML Interpretation Example	20
3.5	Meshlab Example	22
3.6	Overview of the dataflow from real object to the 3D-model	23
3.7	Example Processing Chain: Digital Master Model (DMM)	24
4.1	Overview of the Process of Generating Metadata	28
5.1	Directory Structure Information	44
6.1	Overview of the MetadataGenerator Dataflow	48
6.2	Main Window of the MetadataGenerator	49
6.3	Input Fields of the MetadataGenerator	51
6.4	Acquisition Event Chain of the MetadataGenerator	52
6.5	Project Event Form	56
6.6	Capture Event Form	58
6.7	Detailed Sequence Event Form	60
6.8	Acquisition Event Form	60
6.9	Process Event Form	64
6.10	Ingest Script Config Window	65
6.11	Three-tier Software Architecture of the MetadataGenerator	66
7.1	Image Acquisition Environment	69
7.2	Example Digitization Results.	71

List of Tables

5.1	Basic Generic Form Fields	38
5.2	Node Types of the Forms Definition Format	40
6.1	Input Fields of the Forms Definition Format	50

Listings

3.1	GML Example	20
4.1	Simple XML Example	34
4.2	Simple JSON Example	34
4.3	Simple RDF Example	35
4.4	Simple CIDOC-CRM / CRMdig encoded in RDF Example	35
5.1	Example showing generated metadata of a Project Event	39
5.2	Example showing Forms Definition Format	41
5.3	A Project Event represented in the Intermediate Format.	42
5.4	Example Template Definition	43
6.1	Input Fields Demo Code	51
6.2	Project Event Description	55
6.3	Capture Event Description	57
6.4	Detailed Sequence Event Description	59
6.5	Acquisition Event Description	61
6.6	Process Event Description	63

1 Introduction

Preserving our cultural heritage enables explaining our rich cultures as well as our political, social and educational values to the now living and yet unborn generations. Therefore, it is an important mission to ensure long-term access for humanity by preserving cultural heritage for the future. One problem of this task is that deterioration and damage of cultural heritage artifacts causes a loss of information.

In October 2012, several paintings by Picasso, Monet (see Figure 1.1), Gauguin, Matisse etc. were stolen from the Art Gallery of Rotterdam. The art thieves were caught some days after the committed crime, but no trace of the paintings was found. Months later, one criminal's mother admitted having thrown the valuables into her oven. The famous paintings were burnt away to nothing but ashes and few remains of a canvas. (Summarized and translated from newspaper article [14])



Figure 1.1: Image of one of the burnt away paintings: 'Waterloo Bridge, London' by Claude Monet (1901). (Image source [57]; License: public domain)

At archeological excavations, important information about a specific object, for example its position in the field, can be irretrievably lost. Furthermore, digging deeper to lower layers, built earlier in time, destroys the younger

layers above it. Thus, archeology destroys its own object of study by removing the layers. This makes detailed documentation of all kinds of information indispensable.

Digitization can help to prevent loss of information, but only to a certain extent. A digitized cultural heritage artifact is a digital copy consisting of specific characteristics of an object (such as colour and shape) obtained at a particular point in time. This digital replica does not exclusively include verbal information, but it can give visual impressions by adding photos, for example. However, the world is 3D, presenting a 3D-model of an object is therefore a more intuitive way. Contrary to photos, 3D-models are known to represent the surface of an object: giving a visual impression and showing the spatial dimension. Thus, 3D-models provide immersive experiences coming close to those of real objects.

1.1 Motivation

A digitized cultural heritage object, like the 3D-model of an Egyptian vase, can be made easily accessible to other people. But having access to a 3D-model remains very restricted or even useless unless additional information for documentation and retrieval purposes, the so called *Metadata*, is given. This data can also be described as “data about data” [27]. For example how old is the vase, what is it made of, who is the current keeper, date of creation or a textual description. This information is essential for retrieval and, e.g., for understanding the historical context or the importance of the object.

Digitization must have the aim of being fit for purpose, e.g., for preservation. Moreover, digitized information can be available in such a high resolution, so that reproduction is possible. This can help to establish long-term accessibility for cultural heritage. However, information is often lost during the process of digitization. Each digitization technology has its limitations at resolution and quality of the result. Furthermore, a processing step on the digital data can cause an accepted as well as an unwanted loss of information. As a consequence, a detailed documentation or description of all processing and digitization steps, the so called *Paradata* [25], is needed. Paradata are what measuring device was used, which software was applied and its used parameters for post processing, etc. Paradata helps to keep track of the loss of information. It gives the user the possibility to go one or more steps back in the processing chain, e.g. from result *Digital Master Model* (see Section 3.1.3) to the input data. This is essential for evaluating and proving the authenticity of the result.

Through digitization of cultural heritage artifacts, information like shape or appearance properties (color), can be preserved. Therefore, the loss of specific items of information can be reduced or even stopped. The problem arises that, nowadays, generating metadata is a costly process. Thus, this process is very time-consuming, which makes it even more expensive to create a considerable amount of metadata. For a museum like the Victoria & Albert (V&A) museum [50], which is currently hosting more than 2.5 million assets, a considerable number of assets has to be digitized. To solve these problems it is necessary to develop and provide suitable technologies to make the generation process of metadata and paradata practicable and inexpensive. Auto-generation would be the best solution, but only a part of information can be created automatically. Other data can be obtained, for example, by using expert knowledge.

Despite the fact that digitization has some caveats, acquiring metadata is indispensable. Digitized data is a very useful addition to the real object. This information about the real object, namely the metadata, connects also to the digital object, e.g. a 3D-model. The digital object provides a preview, so that the real object can be taken on demand, for example for research and exhibitions. Although, in many cases the 3D-model will satisfy the needs of research (e.g., for comparing the object to others or to attribute it to a period), a more detailed research, e.g. radiocarbon dating, requires the presence of the original object.

For museums like the Victoria & Albert (V&A) museum creating digital replicas is a sensitive matter. Pan et al. [32] has shown that, on the one hand, art keepers are in favour of disseminating their digital assets for many purposes, such as virtual exhibitions or promotional aims. On the other hand, it is understandable that data owners also have reservations concerning copyright and intellectual property rights (IPR). Nevertheless, museums and their staff work closely together with universities and initiatives like 3D-COFORM (see Section 2.3) to support and influence the development of modern digitization technologies with their objective of preserving cultural heritage without losses and, simultaneously, of keeping their digital assets under control.

It has been made clear that preserving our cultural heritage and thus, its digitization and collecting correspondent metadata are of major importance. Therefore, the development of suitable technologies for these purposes is a very interesting challenge. Harvesting the background knowledge about the cultural heritage objects of experts must be as easy, secure and low time consuming as possible. Thus, an approach that scales well for millions of assets is required. This sets the scene for the problem of mass-generation of metadata, which this thesis approaches.

As previously shown, metadata are important and indispensable. However, their generation signifies additional effort and cost. In this thesis, we want to present a system which comes close to the following requirements:

- **Collecting information from users:** User generated information is indispensable in many cases. Information harvesting from users should be user-friendly and intuitive. Furthermore, the effort for the user should be minimized by a redundancy free collection of information.
- **Automatable metadata generation:** If it is possible, metadata should be generated automatically for reducing user effort, user mistakes, and costs.
- **Updating Metadata:** A possibility for updating generated metadata, automatically (e.g. per batch script) or by user, should be provided.
- **Suitable for mass generation:** The approach should be able to efficiently generate a considerable amount of metadata from large projects, e.g. by reusing previously collected information.
- **Adaptable:** The approach should be adaptable to different metadata schemes and to changes in the used metadata scheme.

This leads to the generic approach for generating cultural heritage metadata described in Chapter 5.

1.2 Outline

This thesis has its focus on cultural heritage metadata generation. The following chapters provide a theoretical overview of digitization of 3D-assets and metadata generation as well as a practical solution approach.

Chapter 2 shows related work discussing important metadata schemes of the cultural heritage domain. Furthermore, the project *3D-COFORM* will be described. This initiative is funded by the European Commission and searches for solutions for preserving cultural heritage with the focus on 3D. Solutions mentioned in this thesis are all part of the *3D-COFORM* project.

Chapter 3 of this thesis will give an insight into the possibilities for obtaining digital cultural heritage 3D-assets. Another purpose of this chapter is to explain the post processing of metadata and the processing chain. Some use cases will be referred to in order to illustrate the applicability of digitized cultural heritage objects.

Different aspects for generating metadata will be shown in Chapter 4. Moreover, possible metadata sources and opportunities for combining and structuring as well as for storing are listed and explained.

A generic solution approach for generating cultural heritage metadata will be presented in Chapter 5. It uses generic dynamic input forms and generic formats for collecting information from the user.

The MetadataGenerator (see Chapter 6) serves as an implementation example of the generic solution approach. Forms are efficiently applied to collect user information and to present auto-generated information for user review.

Chapter 7 presents the testing and results of the semi-automated metadata generation by the MetadataGenerator. Several (test) campaigns processed by trainees help to improve the MetadataGenerator.

Eventually, this thesis presents its scientific contribution and findings. Furthermore, a conclusion and an outlook for future work will be given in the last chapter (see Chapter 8).

2 Related Work

Contents

2.1	Metadata Schemes	6
2.1.1	Dublin Core	7
2.1.2	CIDOC-CRM and CRMdig	7
2.2	Cultural Heritage Metadata Generation	9
2.3	3D-COFORM	12
2.3.1	Repository Infrastructure	12
2.3.2	IngestionTool	14

As long as 3D-data of cultural heritage are acquired, metadata are important. Therefore, projects like 3D-COFORM (see Section 2.3) or Europeana [11] need rich, descriptive metadata. Furthermore, a suitable metadata scheme is required which defines, how to structure the metadata in the cultural heritage domain. For this purpose, many metadata standards are available. Two different well-known standards are CIDOC-CRM (see Section 2.1.2) and Dublin Core (see Section 2.1.1). Examples for related metadata generation aspects and techniques will be discussed in Section 2.2. Some parts of this chapter were previously published at [39, 30, 31, 32].

2.1 Metadata Schemes

Items of metadata, which are grouped and structured according to the needs of a specific domain or type of information resource, are called *Metadata Scheme*. The meaning of the items is known as the semantics of the scheme and the item-values are the content. The metadata scheme defines the specific item names and its semantics. Furthermore, distinct rules can be specified, e.g. for content presentation or restriction to certain values. [27]

2.1.1 Dublin Core

A noteworthy forerunner of developing a metadata scheme is the open organization *Dublin Core Metadata Initiative* (DCMI) supporting “shared information in metadata design and well-tried practices across a broad range of purposes and business models” [9]. This project was started in the middle of the 1990s and its purpose was to create a public entry point and formal repository for documentation of web resources and physical resources like objects, artifacts, etc. Today every individual and organization worldwide is offered the possibility of contributing to the Dublin Core Metadata Initiative and working on the development of (online) metadata standards.

Developed and now maintained by the DCMI is the *Dublin Core* (DC) [9] metadata scheme. The DC defines a rather simple and concise set of 15 elements originally designed for describing web-based documents. These 15 elements (Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, and Rights) are also called “Dublin Core Metadata Element Set” which became ISO standard in 2006 (current version: ISO 15836:2009 [19]). Further development of the DC led to the so called *Qualified Dublin Core* which enables more extensibility and finer semantic distinctions by using qualifiers to refine an element. For example, the qualifier “created” can be used to denote the element “Date” as creation date of an object. Qualified DC is backward compatible to the original, now called *Simple Dublin Core*, version. In both versions all elements are repeatable and optional, and can be arranged in arbitrary order. [27]

The DC consists of key-value pairs, where the number of keys is limited to 15 elements and the value is not suitable to represent other data objects with metadata. These pairs are insufficient for describing arbitrary relations between data needed to build a semantic network.

2.1.2 CIDOC-CRM and CRMdig

DC is a very useful and popular metadata scheme, even in the cultural heritage domain. However, Doerr [6] and Havemann et al. [18] have shown that DC has severe limitations. The scheme particularly lacks relations, which limits the expressiveness. For example, a metadata property like the Creator of an Egyptian vase is fine for finding other vases of the same artist, but how to retrieve knowledge about objects produced by other creators of the same artist’s family? Another example showing the complexity of relations in cultural heritage is a famous CIDOC-CRM example (see Figure 2.1), the network of relations of the 1945 Yalta conference [6].

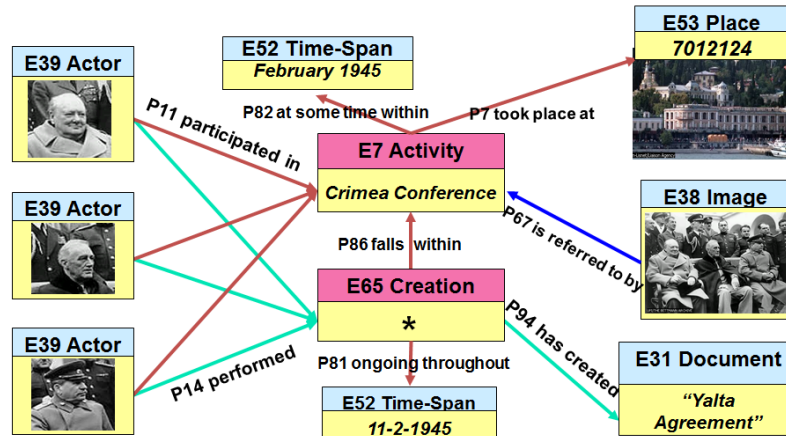


Figure 2.1: An example for CIDOC-CRM showing relations at the 1945 Yalta conference. Relations between three statesmen (Actors), the famous photograph of them (Image), the signed agreement (Document), the geographical location (Place), the negotiation period (Time-Span), the occasion (Activity), and the signing date (Time-Span). (Image source [6])

The CIDOC-CRM (*Conceptual Reference Model*) [4] is exceptional since it is an “empirical ontology” created over a period of 10+ years, and since 2006 it is even an ISO standard (ISO 21127:2006 [21]) for the description of facts and relationships in cultural heritage. This reference model for concepts makes, e.g., metadata schemes based on these concepts compatible to each other. It was published in 2000 as a working draft for exchanging and integrating cultural heritage metadata between museums by the “International Committee for Documentation of the International Council of Museums”(ICOM-CIDOC). CIDOC-CRM uses an event-centric model, meaning that all information is part of an event or related to an event. It is consisting of 90 object classes like E5 (event) or E21 (person) and 149 properties such as P5 (consists_of / forms_part_of) or P7 (took_place_at / witnessed). Note that properties are formulated, so that the sentence can be read in either direction. CIDOC-CRM enables describing all possible kinds of relations between objects in a semantic network. Therefore, it can be seen as a scheme with high comprehensiveness, as it was classified by Ullrich et al. [46] because of its semantic richness. Furthermore, many mappings of other metadata schemes to CIDOC-CRM are available, even a mapping for Dublin Core [5].

An extension of the CIDOC-CRM ontology is *CRMdig* [7], which helps to document the provenance of digitized objects by answering the questions WHO, WHERE, WHEN, WHAT and HOW. For answering these questions, *CRMdig* provides additional, derived classes, like the example in Figure 2.2: D1 (digital object), D8 (digital device) or D13 (digital information carrier) as

well as additional, derived properties, e.g. L10 (had input) or L12 (happened on device). A defined hierarchy of event classes and digital things acts as a guideline and helps to bootstrap the documentation.

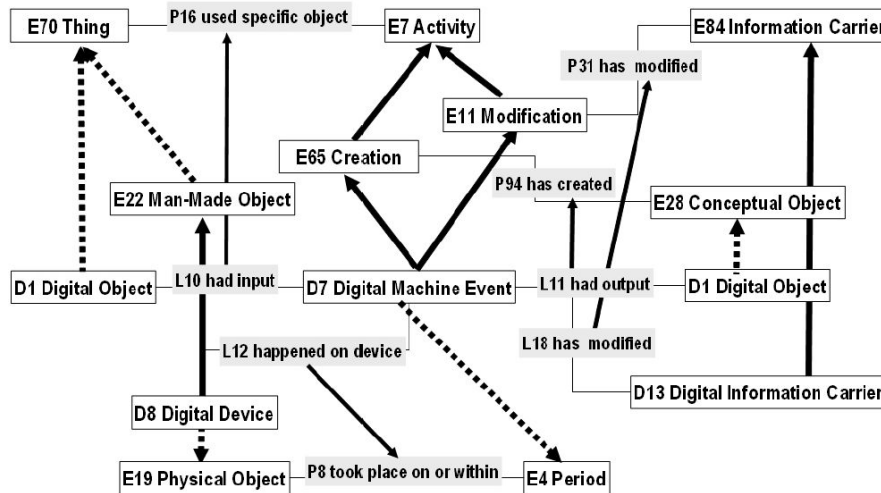


Figure 2.2: This is an example showing CRMdig extending CIDOC-CRM: The CRMdig class D7:Digital Measurement Event is a derivation of CIDOC-CRM classes E65:Creation and E11:Modification. Thus, a documentation of the provenance of human created things by a digital machine will be possible. (Image source [7])

By using both CIDOC-CRM and CRMdig, a comprehensive documentation of all related information including the digitization and processing of digitized cultural heritage objects can be achieved. However, CIDOC-CRM with the CRMdig extension acts as framework and defines how cultural heritage information can be exchanged and how semantic relations can be arranged. It does not define, however, what should (or even must) be documented. Every organization has its own priorities in the information or metadata policy. This thesis is based on an generic approach previously published in [39] (see Chapter 5), which can react to different policies in a flexible way, e.g., easy changing of the used CIDOC-CRM based metadata scheme.

2.2 Cultural Heritage Metadata Generation

When generating metadata for cultural heritage many different aspects have to be taken into consideration concerning the used technology and the area of application. In the following, some example projects and technologies for generating cultural heritage metadata will be described.

Metadata for 3D-Navigation in Cultural Heritage

The aim of the project *Creative Histories* (2004-2007) was to create an application for 3D-navigation on historical sites. This cultural heritage project presents information from different epochs on a PC or mobile device. Krenn, Sieber, and Petschar [23] show the metadata generation part of the project using the example of "The Josefsplatz Experience". For this project, metadata generation in two stages is used. First, human experts identify relevant legacy documents, which are the source for the automatic extraction of information. Second, the extracted information is processed in order to obtain suitable metadata for the application. Furthermore, initial metadata will be extracted from digital sources of the Austrian National Library. [23]

A project similar to *Creative Histories* is *Archeoguide* [53] with the aim of 3D representation and navigation on archeological sites, e.g., ancient Olympia in Greece. *Archeoguide* uses a portable computer and Head Mounted Display for the audio-visual presentation. For storing information and metadata about a site an own repository was developed, which stores the metadata, created by experts or extracted from certain digital sources, in a project specific structure. This information is used to generate predefined tours for navigation through the archeological site.

The metadata generation of the aforementioned and similar projects are very special solutions for the developed applications. However, the previously mentioned projects show that automatic extraction of information from digital sources, even for 3D-data, is feasible, whereas human interaction for the decision of relevance is indispensable. The approach presented in this thesis is more flexible, can use arbitrary information sources and reduces the effort for the user.

Crowdsourcing - User-generated Metadata

Crowdsourcing can be suitable in the cultural heritage domain for generating metadata by users or non-experts. It has challenges like finding sufficient and loyal users, as well as maintaining a reasonable level of quality [28]. However, social tagging [44, 45] or collaborative indexing [3] are very powerful for crowdsourced identification, even of cultural heritage artifacts. Tagging by non-experts improves searching and retrieval by users, due to the used vocabulary which is significantly different to museum documentation [28].

An example for an approach based on crowdsourcing is shown by Van Hooland [48]. It uses user-generated metadata for cultural heritage images. The user comments images with an arbitrary text. A search mechanism

searches for comments which include the searched text and delivers the image and the existing comments. This approach shows the benefits and problems of user-generated metadata. Drawbacks are that users make mistakes and without a given structure useless information like questions, personal experiences, or opinions will be entered. Other users, however, can help to correct errors and mistakes, thus postings of corrections of existing metadata are the most recurrent type of comments.

Unstructured user-generated metadata has advantages for searching, but no semantic network with semantic relationships can be built without a structured metadata scheme. So the great benefit of relations and relation based search will be lost. Crowdsourcing can help to reduce the costs of metadata generation. However, without experts, e.g., for helping to confirm correctness, mistakes can hardly be avoided. The approach presented in this thesis uses structured or scheme based metadata acquisition of the user, but information which does not match with the scheme can be added as notes, so that the information is not lost.

Metadata Generation for Learning Objects

Several definitions for a *Learning Object (LO)* are available. The Learning Technology Standards Committee of the IEEE [24] defines an LO as “any entity, digital or non-digital, that may be used for learning, education or training”. Therefore, cultural heritage data can be seen as Learning Objects and thus, the generation of LO metadata is similar to the generation of CH metadata.

A comparison of automatic and collaborative metadata generation for LOs discussed by Bauer, Maier, and Thalmann [2] results in the suggestion to use a hybrid solution. Some of the general and technical metadata elements for describing LOs, e.g., author, format, or size, can be automatically generated. Furthermore, collaborative tagging could be used for typically human interpreted elements, like keyword or subject. Additionally, it is possible to use tags obtained from collaborative tagging for training in an automatic approach (applied in [40]).

Another example for an approach for automatic metadata generation is presented by Saini, Ronchetti, and Sona [35]. This approach automatically assigns textual learning resources to a given taxonomy with a simple portability based classifier, which allows an association between ontological metadata and the learning resource. This solution has the restriction that only predefined relations between the classes can be detected. This technique could be included by the generic approach presented in this thesis, e.g., in a classification module.

2.3 3D-COFORM

The main goal of 3D-COFORM consortium [10] was to develop and refine existing technologies which help to establish 3D-documentation as an affordable and practical solution for long term documentation of cultural heritage objects, like museums exhibits. The project was funded with about 8.5 million Euro over 4 years (2008 - Nov. 2012) by the Seventh Framework Program of the European Commission under grant agreement no.231809 (IP project “3D-COFORM”). The 3D-COFORM project brought together 19 partners to form several teams working on 3D-capture, 3D-processing, the semantics of shape, material properties, metadata and provenance, integration with other sources (media) and finally, on search, research and dissemination to the public and professional alike.

2.3.1 Repository Infrastructure

Once the digitized cultural heritage assets and its metadata are available, an appropriate storage solution is needed. But only keeping the data is not enough, the system should include a management of the data which handles, e.g., the user permissions, metadata retrieval or upload and download.

The *Repository Infrastructure (RI)* [8, 29] developed for the 3D-COFORM project is able to accomplish the demands on a storage system for cultural heritage assets. The data is stored on dedicated servers, which can be physically located in an arbitrary place. Furthermore, all (3D) data are stored in relation to their appropriate metadata and a comprehensive semantic search and retrieval on the metadata is provided.

The Repository Infrastructure as distributed Content Management System (dCMS) of the 3D-COFORM project has been mainly developed by Xueming Pan and René Berndt, first described in Pan et al. [29] and later at [30], [31], [32]. This section is based on these papers, more details will be published in the PhD thesis of Xueming Pan.

System Design

The RI is composed of three main parts: the RI-Central, the Locations, and many Clients. An overview of the relations between these parts is given in Figure 2.3. The system has a star topology, where the RI-Central is the central point of communication and initializes all other communications between Locations and Clients. The used topology ensures that the system works as long as RI-Central is operational, even if one or more Locations go offline.

Owners can choose to re-replicate their digital assets over different Locations, and ideally, the online status of the Locations is transparent for the clients.

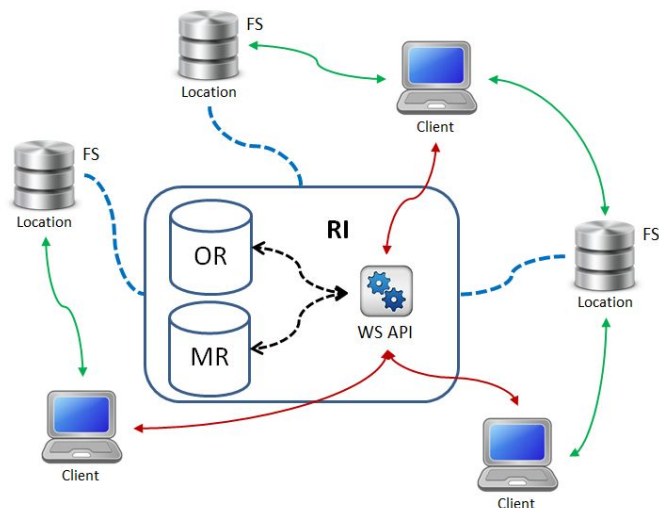


Figure 2.3: RI system overview. The box represents RI-Central, contains a relational object database (OR), and maintains a connection to the internal MR and OR servers. RI-Central communicates with clients (red arrows) and Locations (dashed blue lines). For large data transfer, RI-Central initiates direct communication between Location and clients (green arrows). Black dashed arrows denote the RI internal web service communication. (Image previously published in: [32])

The Core Component, RI-Central

RI-Central is both the public portal and the “brain” of the RI infrastructure. It is composed of (a) the ORDB, a conventional relational database storing all schematic information on files and transactions, and a backup of the CIDOC-CRM metadata; (b) the metadata repository (MR) containing the semantic network built from the CIDOC-CRM metadata; and (c) a webservice for public access to communicate both with clients and Locations.

The Usage of Locations

Locations are one of the key features of the RI system. The connected Locations are the place where binary datasets are physically stored. For performance reasons, the datasets are transmitted directly from the client computer to the Location (upload) or vice versa (download). The transmission, however,

must be initiated by RI-Central which generates a one-time URL that points to the Location and is sent to the client.

Datasets of users from different institutions can reside on the same Location without compromising visibility. Institutions have complete control over their data; it might be, though, that data from one institution are temporarily transferred to a Location of another institution, but nobody in this other institution has any permission on that dataset. To resolve this situation, special roles with special permissions are introduced, e.g., the Location Admin who has access to all datasets on his Location and is able to notify the respective owners.

A Location is composed of three main components:

- a) **Web service interface:** This compact interface accepts the requests from the central server, handles data transfer requests (for upload/download), and generates the temporary transfer URL used by the end user.
- b) **Local database:** This DB contains only three tables, DownloadRequest, UploadRequest, and ReplicaJob. The first two tables are used to store the temporary and disposable tickets generated by RI-Central with a random ID. As mentioned in section 3.1(8.), users can create replicas of their datasets. The ReplicaJob table is used to store the replica's status and other information.
- c) **Local file system:** It is used to store the binary datasets of the institution and replicas from authorized institutions. Since 3D-models can be quite large (> 2 GB !), we recommend that the attached harddisk space is sufficiently large.

2.3.2 IngestionTool

The *IngestionTool* [13, 15] shown in Figure 2.4 is designed for entering high quality metadata. It was developed by FORTH-ICS in Heraklion (Greece) as part of the 3D-COFORM project. The IngestionTool uses hard-coded forms and needs an online connection to the 3D-COFORM RI. The implementation in Java makes it platform independent.

The great advantage of the online setting is that the user gets immediate feedback on the validity of the input data. If, for instance, in the course of defining an acquisition event, the photographer is defined, and the unique ID (see Section 4.3.2) of this person is already used in the system, then the IngestionTool gives immediate feedback. All entities that are already present in the MR (persons, places, institutions, devices, software, etc.) can be used to fill out the metadata forms. This is ideal for high-quality ingestions of small amounts of data. The graphical user interface of the IngestionTool is designed

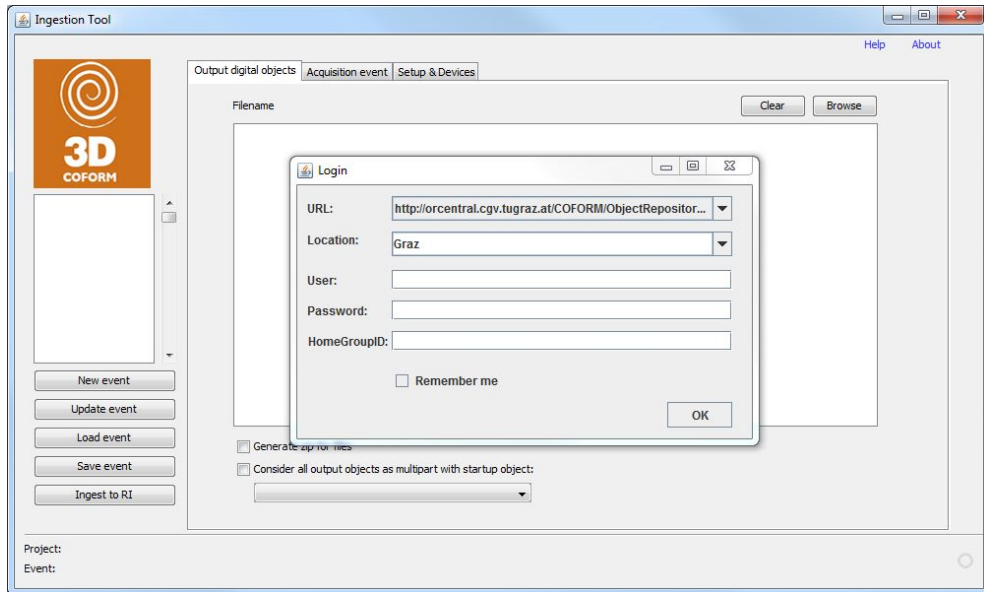


Figure 2.4: This figure shows the IngestionTool main window and its login window for the RI.

in such a way that it makes creating the CRMdig process description feasible also for non-IT professionals and for end users with a CH background.

As mentioned before, the IngestionTool is suitable for creating few, but high-quality 3D-COFORM metadata. Therefore, it requires an online connection for providing the up-to-date RI information. The *MetadataGenerator* presented in Chapter 6 is an implementation of the generic approach introduced in Chapter 5. The *MetadataGenerator* was also developed as part of the 3D-COFORM project but, in contrast to the IngestionTool, for a different metadata creation scenario with other requirements. The *MetadataGenerator* is designed for an offline scenario and mass generation of 3D-COFORM metadata. Both tools, the IngestionTool and the *MetadataGenerator*, create CIDOC-CRM based 3D-COFORM metadata, thus the same scheme is used. However, as the *MetadataGenerator* is based on a generic approach, it is very flexible, which means that it can be easily adapted to requirement changes, e.g., changes in the used metadata scheme, which is not possible with the IngestionTool.

3 Digitization of CH 3D-Assets

Contents

3.1	Digitization Processing Chain	16
3.1.1	The Way to 3D-Data of Real Assets	16
3.1.2	Post Processing	20
3.1.3	Digital Master Model	23
3.2	Use Cases	24
3.2.1	Change and Restoration Monitoring	24
3.3	Benefits of Digitized Cultural Heritage Assets	26

Digitization of cultural heritage artifacts is an opportunity for preventing a loss of information (see Introduction Chapter 1). In the following, an overview of digitization techniques and post processing steps of digitized 3D-assets is given. The end result of a digitization processing chain, the *Digital Master Model (DMM)*, will be presented. Finally, some use cases will be discussed.

3.1 Digitization Processing Chain

The Digitization Processing Chain for 3D acquisition consists of two main steps. The first step at digitization of cultural heritage 3D-assets is to obtain a 3D-model of a real object by measurement (e.g., image, laser scan). Once a “raw” 3D-model is available, in most cases the 3D-model has to undergo one or more post processing steps, for example, to clean the 3D-model from unwanted background. The result of such a processing chain is called Digital Master Model.

3.1.1 The Way to 3D-Data of Real Assets

There are many different ways to acquire a 3D-model of a real object. In the following some examples for 3D acquisition techniques will be given. All opportunities and technologies are different in terms of the needed manpower, cost, and time as well as concerning the quality of the result.

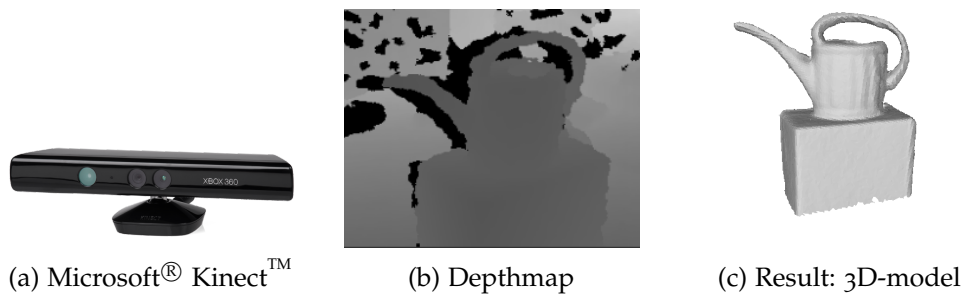


Figure 3.1: In this figure an example for a 3D-Scanner (Microsoft[®] Kinect[™]) developed for games can be seen. (a) illustrates the device. (b) presents an example range data image (depthmap) delivered by the device. (c) shows the result 3D-model created with the scanning software: ReconstructMe.

3D-Scanner

A 3D-scanner is a special device which helps to generate a 3D-model of a real object or environment. 3D-scanners are available in many different types differing in scanning technique, scan quality, resolution of the result, scan time and price. Basically, a scanner collects real-world shape data which can be used to generate a 3D-model. Figure 3.1a shows an example for a 3D-scanner developed for games, the Microsoft[®] Kinect[™]. For generation of the resulting 3D-model (Figure 3.1c) the depth information (Figure 3.1b) is used by a software like *ReconstructMe* [34] which is an implementation of the approach presented by Izadi et al. [22]. Some scanners capture additional appearance information like the color, so that a realistic 3D-model can be created. Powerful scanners, able to deliver a good result in short time, are in general more expensive.

Photogrammetry

Photogrammetry is a technique to obtain a 3D-model from photographic images. With photos from a real object, an automated 3D reconstruction can be generated. A lot of commercial and non-commercial software using images to calculate a 3D-model is available. A free to use solution for this purpose is the web-based tool ARC3D (Automatic Reconstruction Cloud) [51], developed and first described by Vergauwen and Van Gool [49] and later in Tingdahl and Van Gool [43]. This system automatically generates a textured 3D-model and range-maps (depth-maps) from the images uploaded by the user. The images of an object or scene have to be taken from around the object nearly at the same distance, but with enough overlap, which is needed for depth (range) calculation by the underlying stereo matching algorithm. A cluster

of computers is used for parallel computation of the camera calibration, the range-maps, and the result 3D-model. The result quality depends on the quality, resolution, and overlap of the images. A system overview with the different processing steps is shown in Figure 3.2.

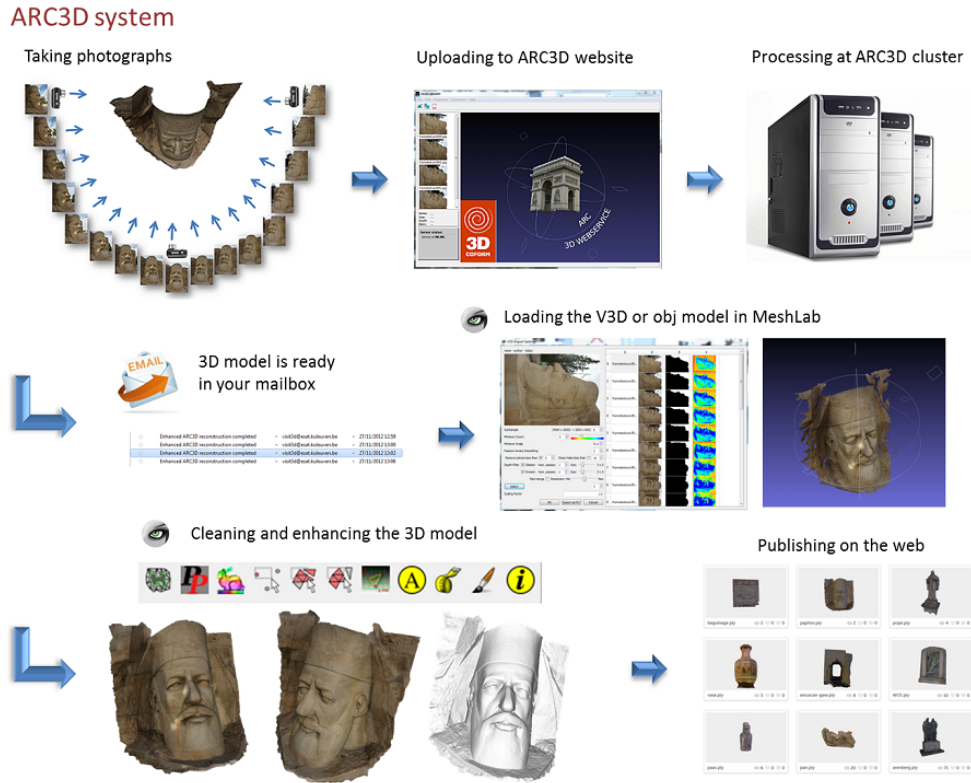


Figure 3.2: This overview of the ARC₃D system shows the entire digitization processing chain from taking photos over using ARC₃D for creating the 3D-model to its post processing and publishing. (Image source [51])

Modeling

Creating handmade 3D-models with a commercial 3D-modeling software like Maya [1] (see Figure 3.3) or an open-source software like Blender [41] is an opportunity to obtain a high quality digitized asset. Thus, the resulting model can be very detailed and complex. However, it is idealised and therefore only an abstraction. Furthermore, handmade 3D-models have as main drawback a very high time consumption, which leads to high cost.

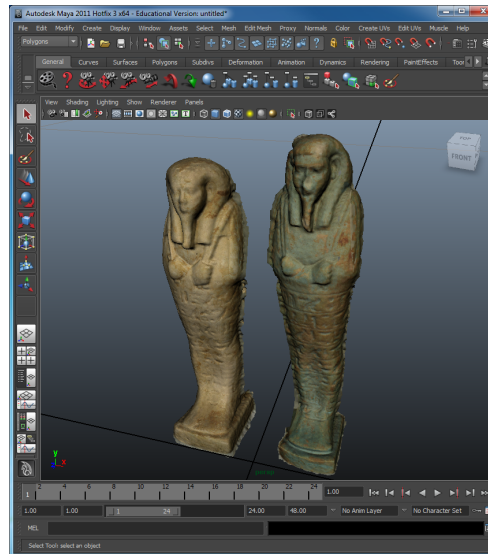


Figure 3.3: Screenshot of the a 3D-modeling software example: Autodesk Maya 2012 [1]. Beside modeling, many post processing steps, including for example 3D animation, are possible .

Generative / Procedural Modeling

In contrast to geometric modeling, where the mesh parts (e.g., triangles, vertices, etc.) are directly manipulated, in generative modeling the parameters of a function describing the 3D shape are changeable. Thus, for each different shape type a function can be defined, afterwards a new shape will be obtained by manipulating few high level parameters [37]. Therefore, a 3D-object can be represented by the parameters of a function which represents a class of objects.

The description of a function can be expressed in textual form. Thus, generative modeling becomes a kind of shape programming or textual shape description. Main benefits are that the 3D-objects can be reproduced as often as desired and that the shape can be easily changed with the parameters or even the textual description altered. However, a challenge is to find the description rule or parameter, which has to be changed and the new value, to obtain the desired result.

Generative Modeling Language (GML) was introduced by Havemann [17]. It is a implementation of the generative approach for modeling by shape description. This stack-based programming language will be interpreted at runtime. It includes many high and low level operators for describing a

shape. The 3D-model will be obtained at runtime by interpreting the GML program.

```

1 (0,0,-2) (1,1,0) 2 quad
2 /cyan setcurrentmaterial 5 poly2doubleface
3 (0,1,1) extrude
4 (0,0,1) (1,0,1) normalize 0 project_ringplane
5 (2,0,0) (0,1,-1) 2 quad
6 /yellow setcurrentmaterial 5 poly2doubleface
7 0 bridgerings

```

Listing 3.1: This listing shows a GML Example for modeling with a stack-based language. (Code source [17])

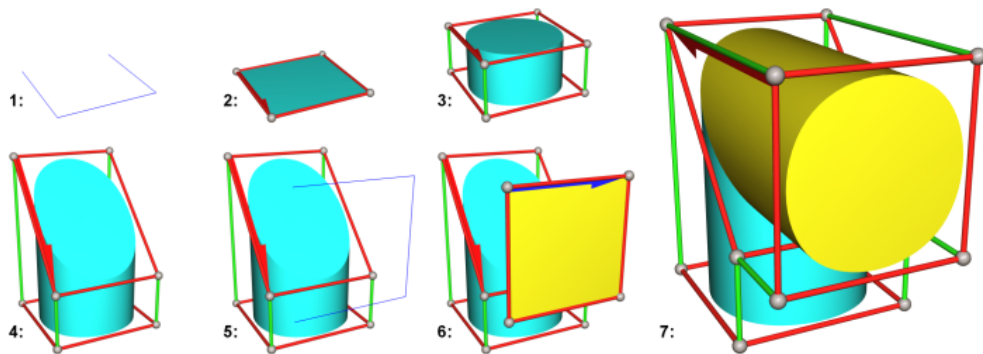


Figure 3.4: This figure shows the step by step interpretation results of the GML example in Listing 3.1 (Image source [17])

CityEngine is an example for a (commercial) software for procedural digitization or reconstruction of cities. The procedural approach and the system were developed by Parish and Müller [33]. It uses image maps, e.g., of land-water bounders, as geographical input data. The creation of an urban environment is based on a set of hierarchically organised rules which can be extended with respect to the user requirements. With the CityEngine it is possible to generate a 3D-model of the entire traffic network and buildings of a large city.

3.1.2 Post Processing

The scanning process of a 3D-model can cause unwanted artifacts in the 3D-mesh or unwanted scan parts, such as the background for example.

Therefore, a mesh has to be cleaned from such unwanted parts. These and other processing steps, done after the scanning, are part of the so called *Post Processing*.

There are certain types of processing steps to improve the visual impression. However, some of these steps would lead to inventing information, which is (without documentation) unacceptable in the cultural heritage domain. For example, missing parts of the mesh can be added to complete the result, e.g., holes in the mesh can be filled. These holes in the 3D-mesh possibly originate from the digitization process or even from the real object. Filling of such holes works mostly through the invention of (surface) information, which is generally not desired. Therefore, a detailed documentation is indispensable, which enables linking back to the measurement information for distinguishing real from invented information.

Manual 3D Processing

A 3D editing software like *Meshlab* [52] can be used for manual post processing (see Figure 3.5). For example, cleaning the mesh from unwanted artifacts can be done by selection of the unwanted mesh parts and by simply cutting them away. On demand, algorithms can be applied by the user to the entire 3D-mesh or to the user selected parts. All intermediate results must be kept and all steps have to be documented, which ensures reproducible results, error tracking and linking back to measured (real) data. This is important to guaranty the integrity of the data.

Automated 3D Processing

Post processing steps, like down-sampling or mesh smoothing, can be applied automatically on a 3D-model. Especially, global algorithms working on the entire mesh are suitable for automation. Therefore, depending on its desired result, a predefined sequencing of algorithms (automated processing chain) could be used. Automated 3D processing by a tool using the same data and parameters leads to automatically reproducible results. Therefore, automatic tracking of errors or linking between measured (real) data and virtual 3D-data is possible. Thus, it is possible to automatically ensure the integrity of the data.



Figure 3.5: This screenshot presents the open-source software Meshlab which is suitable for (manual) post processing of the 3D-mesh.

Merging of Processing Results

A processing step can use the result of more than one other processing step as input. Thus, the results of different processing steps can be merged and processed to a new result. In many cases 3D-model parts have to be merged or combined to obtain the entire 3D-model or 3D-scene. For example, a large real statue can be digitized (3D-scanned) part per part, e.g., head, body, legs, arms. The 3D-models of the parts have to be combined afterwards to obtain the complete 3D-model of the statue.

Metadata Generation

Metadata are very important, for example for answering semantic questions. Therefore, metadata generation is also an important post processing step. Some metadata can be generated automatically, but a lot of information is needed from the user. A detailed description of generating cultural heritage metadata will be given in Chapter 4.

An overview of the dataflow including the metadata generation process from the real object to the 3D-model is illustrated in Figure 3.6. At first a real

object is digitized and 3D-data are obtained. From both the real object and its 3D-data the associated metadata will be generated. Finally, the 3D-data and its metadata are inserted into a suitable storage.

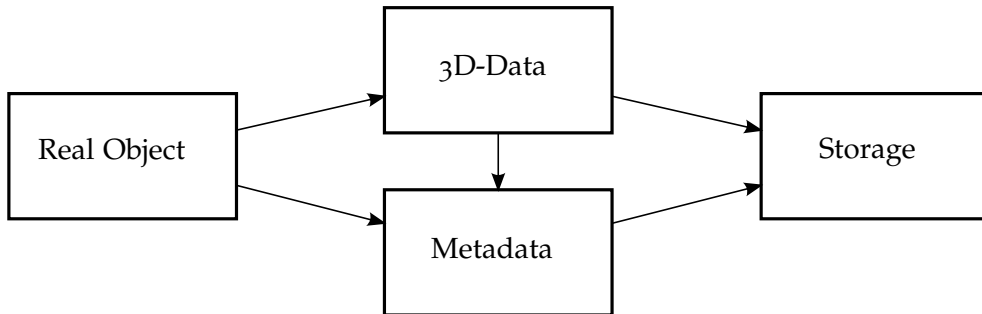


Figure 3.6: This figure shows an overview of the different steps of the dataflow (including metadata) from the real object to the 3D-model.

3.1.3 Digital Master Model

To obtain a 3D-model of a real asset, some intermediate processing steps have to be done. The input for the process chain is usually digit (raw) data, e.g., range maps from a scan or taken photos. Processing steps can be applied to obtain a modified version of the data, for example 3D-model generation. Each intermediate result can be input for other post processing steps, furthermore, multiple inputs can be combined into one output. An end result, a so called *Digital Master Model* (DMM), can also be an intermediate result for further processing steps. Since a DMM is a processed version of the measured raw data, it can not have more information or quality than available in the raw data. However, the quality of the DMM depends on the used algorithms at each processing step. Therefore, all intermediate results and digital master models should be stored including a detailed description of the processing steps, so that the data can be reproduced or improved by starting at an arbitrary step. Furthermore, detailed documentation ensures the integrity of the data and enables an error tracking. Figure 3.7 shows an example processing chain to illustrate the dataflow from various input data over intermediate results to DMMs. From a DMM, several models can be derived, for example, a simplified version of DMM is a derived model, for special purpose like the web.

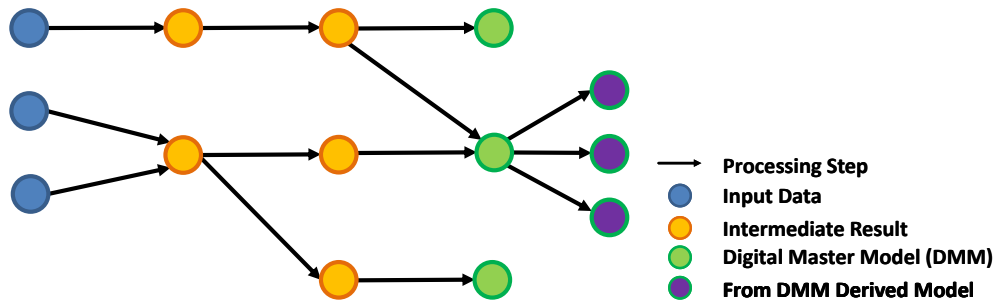


Figure 3.7: This figure presents an example of a Processing Chain; Various input over intermediate results to DMMs. From a DMM several models can be derived, e.g., simplified version of the DMM.

3.2 Use Cases

Digitization of cultural heritage opens many opportunities. Once an asset is digitized, for example, most research work can be done with the 3D-model instead of the delicate real object. In the following, some use cases will be briefly explained.

Digital Restoration

Museums want to provide their visitors an impression of the original object. As many artifacts are (partly) damaged, digitization is a solution to complete them virtually. A further problem museums face is that sometimes artifacts break into multiple pieces before or after their excavation, which can lead to the exhibition of their single parts at several museums. Digitization opens the possibility of creating a virtual representation of the original object without dislocating any of the exhibited objects.

3.2.1 Change and Restoration Monitoring

Digitization serves the documentation of deterioration or alteration of an artifact. This can be very helpful if the state of the artifact at an earlier point in time or at its creation should be shown. Furthermore, digitization documents the state of the real object before and after restoration. This is of major importance, as every restoration means a significant change to the artifact and consequently leads to an adulteration of information. Only monitoring provides a complete documentation of the artifact's life cycle and helps to avoid or even reverse mistakes occurring at restoration.

Measurement

If the global scale is known, all measurements can be done on the 3D-model. The entire object including all details can be measured. The accuracy depends on the mesh quality respectively to the accuracy of the scan.

Virtual Exhibition

Most museums have much more exhibition objects than can be shown at once. The exhibition space is limited, so the most objects are kept in the museum store and shown only on special occasions like an exhibition focussing on a specific time-period or place in the history. A virtual exhibition presenting digitized objects inside a 3D-scene on a computer has, theoretically, no spatial limitation. Furthermore, not only objects from one museum can be shown, objects from different museums can be combined, for example, topically into one virtual exhibition. The museums exhibits can be presented in an interactive 3D-animation as shown by Zmugg et al. [59]. Virtual exhibitions can be made reachable from all over the world with the internet.

Virtual Planing of Real Exhibitions

Virtual objects facilitate immensely the planning of a museum's exhibitions. The sequence of the objects, the light or visitor paths can be planned accurately and virtually pre-shown or simulated to avoid cost.

Image from Uncommon Perspective

To take an image from an uncommon perspective like from the bottom or from inside out is normally not possible without the risk of partial damage. The virtual camera, however, which takes an image of the 3D-model, can be placed in an arbitrary position. As a result, an easy and non-destructive possibility for taking fascinating or detail images from uncommon perspectives is given.

3.3 Benefits of Digitized Cultural Heritage Assets

Digitization of cultural heritage assets is beneficial for many reasons. One advantage is that digital data can be easily copied (for backup) and transferred via the internet. Thus, an integration into modern information technologies is possible and unproblematic. Therefore, many people all over the world can be granted access to our cultural heritage, for example through a virtual exhibition. The virtual representations could have a kind of promotional effect on possible visitors, so that they are encouraged to admire the real object. Furthermore, artifacts previewing for research and exhibition planning can be made.

3D-models of real objects are virtual replicas still differing in quality and accuracy from the original. Sometimes, a 3D-model is sufficient for research purposes, for example to compare artists, styles or periods. In this way travelling of the researcher (or the artifact) can be avoided. The 3D-model should be as accurate as possible to avoid errors or not to miss important details. Therefore, the quality of the scanning and post processing is crucial for the result model, although the aim or use case determines the needed result quality.

A physical replica, which is (at first sight) undistinguishable from the original, can be made on the basis of a high quality 3D-model. Such replica might be useful if an artifact becomes lost or destroyed.

4 Generating Cultural Heritage Metadata

Contents

4.1	Metadata Sources	28
4.2	Combining and Structuring	30
4.3	Storing Metadata	32
4.3.1	Metadata Storage Places	32
4.3.2	Information Linking	32
4.3.3	Metadata Storage Formats	33
4.3.4	Format Conversation	33
4.4	Iterative Metadata Refinement	36

Generating metadata is a time and cost intensive task. This chapter explains how metadata can be generated. The process can be split into three main parts (see Figure 4.1):

- Collecting input from different sources (see Section 4.1)
- The rule or scheme based generation of new metadata by combining and structuring the source information (see Section 4.2)
- Applying a storage solution which ensures an easy reuse and format conversion of the generated metadata (see Section 4.3)

As metadata generation can be an iterative process, the output may be the input at further iteration steps, e.g. at reuse of generated metadata. Section 4.4 gives an overview about iterative metadata generation. An approach for generating cultural heritage metadata based on the aforementioned three main steps will be presented in Chapter 5.

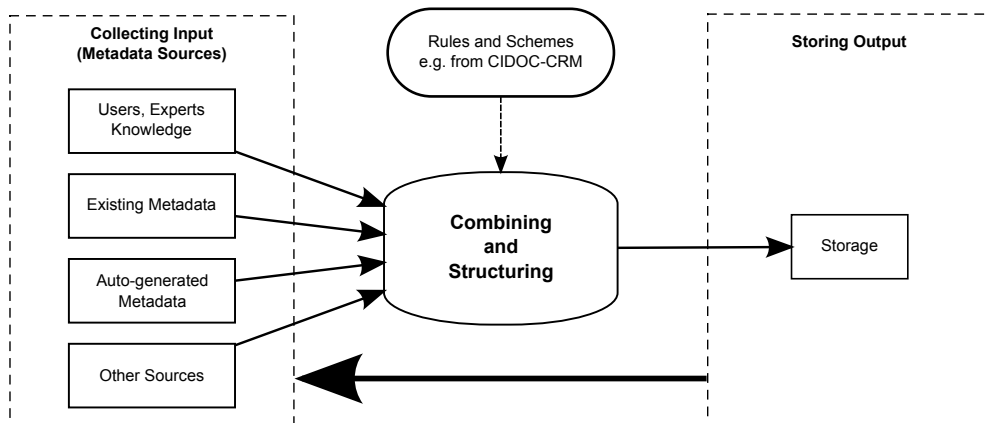


Figure 4.1: Overview of the process of generating metadata. Input from different metadata sources are combined and structured considering rules to generate new metadata output which can be stored in different ways. The output can be used as input for a further generation process.

4.1 Metadata Sources

Users and Experts

User knowledge is an indispensable information source. However, not every user is capable of giving all requested information in any situation. In some cases only an expert has enough background knowledge or experience for providing data or special information. The obvious way to acquire information is a form which should be as simple and efficient as possible. One major caveat of users or experts as sources of information is the fact that human working time is expensive. Metadata mass generation, in particular, may not only cause high cost, but, as it usually includes a lot of recurring work and information, it can also lead to many mistakes. Thus, other metadata sources should be considered for use first to obtain a low-cost metadata generation solution.

Existing Metadata

Many digital data formats include metadata which was previously generated, e.g., by the software or hardware which created the digital data. Thus, a lot of digital storage formats include a specification for file domain dependent metadata like EXIF [42] for image files. Further previously generated metadata may be available, for example background metadata or the output of a

previous generation cycle. This existing metadata can be reused as metadata source.

Extracted Metadata from Digital Data

In some cases, if no metadata is explicitly available, a generation from the content itself might be possible. For example the amount of vertices, faces, edges of a 3D-mesh or the number of image pixels can be calculated by analyzing data. Further metadata can be obtained from data by applying special algorithms. 3D-models, for example, usually do not explicitly provide measurement information and often measuring on the real object turns out to be rather complicated or even impracticable for several reasons; especially when the object is fragile. Calculating measurement information, like the size or volume of a vase, can be done on the 3D-mesh without impact on the real object.

Another example for extracting metadata is the file format identification. The extension of a file is normally used to denote the mimetype. However, the extension is ambiguous, as different mimetypes use the same extension or an extension is missing. Additionally, an extension can be easily changed. So an automated mimetype detection, which does not only rely on the extension, should be carried out. Several approaches for content based mimetype or sometimes called file type detection are available. As a lot of file formats use a header, in most cases the first 256 bytes of file content are sufficient for detecting the mimetype. For automated mimetype detection a suitable solution can be based on both techniques (extension and content based mimetype detection). The result decision can be made by a voting mechanism.

Extracting metadata from digital data has a great potential for automation. However, an individual solution could be needed for every single digital data domain (3D, image, etc.), file format, required information.

Other Metadata Sources

Each available information source is potentially a metadata source. For example metadata information could be found in the internet, e.g., in a free online encyclopaedia like Wikipedia [58] or in an online (metadata) database. Thereby, it has to be known which internet resource is applicable and how to access the data.

Furthermore, the structure of the directories in the file system can be used as a metadata information source. This structure information can be extracted

and applied to the scheme dependent metadata structure, e.g. for a scheme based on the CIDOC-CRM / CRMdig event chain (see Section 5.3).

Further sources for metadata are easily conceivable, but one must know how to use them appropriately. Basically, any additional information can be useful.

4.2 Combining and Structuring

The metadata of different sources (see Section 4.1) has to be combined and structured depending on the requirements of the domain. For cultural heritage metadata, standards are available which describe possibilities to structure the metadata in this domain. This thesis has a focus on the CIDOC-CRM and its extension CRMdig (see Section 2.1.2). This standardized ontology is event centric which means that all data are part of an event or related to an event. The events form a dependency tree or an event chain.

CIDOC-CRM enables the establishment of a semantic network which is refinable by adding supplementary items of metadata. This semantic network helps to collect all necessary facts and makes it possible to search for and retrieve (unknown) relations and data.

As the structure is given by the used scheme, it can be used previously to build a proper setting for collecting information in a structured way. This can be realized with a form where the fields are structured with respect to the used scheme. The fields can be pre-filled with the automatically usable information of all metadata sources and afterwards manually completed.

By combining the collected data, a distinction on the degree of contribution of a single user can be made. From 100 percent like in a manual process to zero percent in a fully automated process.

Manual

With the help of forms which are structured corresponding to the needed information, a user is able to enter the required information manually. But in many domains, with special regard to cultural heritage, an expert user with appropriate background knowledge is necessary.

Records of the user actions document the process or all processing steps applied on the data. This paradata is important for reproducing and verifying derived results and can be auto-generated by the software, e.g. during user interaction in background.

Crowd-based

Crowd-based metadata generation, where a lot of people work together, is another very powerful tool to reduce the cost of user generated information. This can be done explicitly like in Wikipedia [58], a free online dictionary where the content is crowd-based created and reviewed. Crowd-based information can also be implicitly generated, which means that information is created incidentally by participating people. For example, a captcha is a non-machine readable text which is commonly used to protect online services from non-human access. But the captcha service reCAPTCHA [16] uses capturing results from the user for digitizing non-machine readable texts, e.g., from books or old newspapers.

Automated

The aim is to generate metadata as automatically as possible. A lot of different approaches can help to achieve it. First of all, existing metadata which are already generated can be reused. Further, in a lot of storage formats some metadata are included like the EXIF information for images. This information can be automatically extracted and reused. However, a format transformation is sometimes necessary. If the format transformation is defined, information from metadata sources like existing metadata can be automatically converted with respect to the metadata scheme.

Semi-Automated

Some information is not automatically includable or can not be automatically generated. So the user or an expert is asked for it. The same goes for missing semantic links between several items of information. In many cases, only suggestions for metadata or semantic links can be automatically generated. A user has to decide if he wants to accept or discard the suggestion. In other cases, the user is asked to select between a range of given options.

Automated generation of metadata is the only way to reduce the human interaction to a minimum. However, sometimes a user is needed for reviewing and avoiding or at least reducing errors.

4.3 Storing Metadata

For storing metadata, different solutions are possible. Which solution will be preferred and how the metadata is processed is dependent on the domain. The three main storage solutions can be distinguished by the storage place. Further, linking of stored information is important for reuse and retrieval. This can be achieved by using an appropriate metadata storage format.

4.3.1 Metadata Storage Places

Local Metadata files can be locally stored, i.e., next to the related binary file. Working with only one of them is possible while the logical connection is preserved. Sometimes only the metadata are needed, e.g. for searching or building a semantic network. Then the usually much larger binary files can remain untouched.

Included If data and metadata are stored together inside in a file, a direct connection is realized. Having to handle only one file facilitates copying and transferring. In case the user wants to work with only one part, data or metadata, both have to be consulted. Many storage formats include metadata in the file like JPEG the EXIF information. The *Extensible Metadata Platform* (XMP) is an ISO standard (ISO 16684-1) [20] for standardized and custom metadata, which also provides guidelines for embedding XMP metadata into binary files.

Database Managing all metadata inside a database, which includes all semantic connections, opens the opportunity to search for specific information or a binary file and to find new or unknown relationships.

4.3.2 Information Linking

URI The *Uniform Resource Identifier* (URI) is a string of characters normally used to identify a web resource. Sub-classes of URIs are the *Uniform Resource Locator* (URL) for providing a method for finding something like the address of a person and the *Uniform Resource Name* (URN) for defining an item's identity like the name of a person. [55]

By using an unique URI, it is possible to create human readable information links without ambiguities. Usually, the human readable part of a unique URI has to be predefined, thus it is not auto-created.

UUID The *Universally Unique Identifier* (UUID) is a 16-Byte (128-Bit) number which can be used effectively as a unique label. The uniqueness can not be completely guaranteed, but due to the fact of possible 2^{128} , or about $3.4 \cdot 10^{38}$ different ids, it is practically unique. A UUID is usually represented by 32 hexadecimal digits, displayed in five groups separated by hyphens. This can be preceded by (non standard) “uuid:” prefix like the scheme name of a URI (used in the 3D-COFORM project). A sample UUID can look like:

```
uuid:7f671696-c560-40f1-a7d0-1a03d3ea8ba7
```

UUIDs have a great advantage, they can be independently auto-generated, which means two or more computers can create them without communication. As a result, lists of UUIDs can be merged at any time and no UUID will exist twice in the merged list. This is important for distributed metadata generation where, ideally, the unique labeling of information items can be done independently. [56]

Metadata items can be locally grouped. The connections between groups or specific items are very important. Only these connections or so called semantic links enable a reuse of information and statements about coherences. But to preserve referential integrity, a link must be unique. As suitable solutions for practically unique linking or referencing a URI can be used. In many cases a distinct identification with a UUID will be sufficient.

4.3.3 Metadata Storage Formats

Many different standard storage formats for metadata are available. For storing cultural heritage metadata, standard schemes based on CIDOC-CRM / CRMdig or Dublin Core may be encoded into standard storage formats. For example formats like XML (see Listing 4.1), JSON (see Listing 4.2) as well as HTML or even plain text are in use.

Widely used is the *Resource Description Framework* (RDF) [54] which is a W3C standard and can be expressed in XML (see RDF example: Listing 4.3). RDF data are true statements about resources, represented as simple subject-predicate-object sentences, so called triplets. A triplet contains two entities (subject and object), and one property (the predicate) describing the relation of the entities. Metadata standards like CIDOC-CRM / CRMdig (see Section 2.1.2) can be encoded in RDF (see example in Listing 4.4).

4.3.4 Format Conversation

Every system might use an own storage format, which can be optimized for the individual system requirements. However, standard formats are often

```
1 <person id="uuid:f19232f2-e2ad-4b3f-881d-69b3b1f5665f">
2   <name>
3     <first_name>Max</first_name>
4     <last_name>Power</last_name>
5   </name>
6 </person>
```

Listing 4.1: A simple XML example for storing the first and last name as well as a unique ID (UUID) of a person.

```
1 {
2   "person": {
3     "id": "uuid:f19232f2-e2ad-4b3f-881d-69b3b1f5665f",
4     "name": {
5       "first_name": "Max",
6       "last_name": "Power"
7     }
8   }
9 }
```

Listing 4.2: A simple JSON example for storing the first and last name as well as a unique ID (UUID) of a person.

used to ensure compatibility or cooperation with other systems. Since a lot of different standard and individual non-standard storage formats are in use, a format conversion is very important.

Hard-coded

One opportunity to obtain a format transformation is a hard-coded implementation. Once all transformation rules are available, a specific input format can be converted into a specific output form. But the result conversion program is unalterable and if changes are needed, only a software developer can change the code and recreate the software. This solution might be fast in process, but it is very specific and very inflexible.

Template-based

Compared to a hard-coded implementation, a template-based format transformation performs much better with regard to flexibility. Thus, any arbitrary input format can be easily converted into an arbitrary output format, e.g. a non-standard intermediate format into a standard format like RDF.

```

1 <rdf:RDF
2   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3   xmlns:foaf="http://xmlns.com/foaf/0.1/"
4   <rdf:Description
5     rdf:about="uuid:f19232f2-e2ad-4b3f-881d-69b3b1f5665f">
6     <foaf:Person>
7       <foaf:firstName>Max</foaf:firstName>
8       <foaf:lastName>Power</foaf:lastName>
9     </foaf:Person>
10  </rdf:Description>
11 </rdf:RDF>

```

Listing 4.3: A simple RDF example for storing the first and last name as well as a unique ID (UUID) of a person.

```

1 <rdf:RDF
2   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
4   xmlns:crm="http://www.ics.forth.gr/isl/rdfs/3D-COFORM_CIDOC-
5     CRM.rdfs#"
6   xmlns:crmdig="http://www.ics.forth.gr/isl/rdfs/3D-
7     COFORM_CRMdig.rdfs#"
8   <crm:E21.Person
9     rdf:about="uuid:f19232f2-e2ad-4b3f-881d-69b3b1f5665f">
10    <crm:P131F.is_identified_by>
11      <crmdig:D21.Person_Name>
12        <crmdig:L51F.has_first_name>
13          Max
14        </crmdig:L51F.has_first_name>
15        <crmdig:L52F.has_last_name>
16          Power
17        </crmdig:L52F.has_last_name>
18      </crmdig:D21.Person_Name>
19    </crm:P131F.is_identified_by>
20  </crm:E21.Person>
21 </rdf:RDF>

```

Listing 4.4: A simple CIDOC-CRM / CRMdig encoded in RDF example for storing the first and last name as well as a unique ID (UUID) of a person.

Basically, template-based implementations follow replacement rules which are applied to the data in the input format together with templates, to obtain a conversion into the output format. The replacement rules can be defined together with the templates and stored in suitable file formats like XML. A program developed to interpret the rules converts the formats.

4.4 Iterative Metadata Refinement

Metadata generation is a cyclic process leading to reuse of previously generated metadata. Furthermore, once some data are generated, further metadata can be included based on the previous information. For example, when the mimetype metadata of a file is created, the appropriate metadata of the digital object type can be linked at the next iteration step. This can be done automatically based on the information of existing metadata relations.

For generating cultural heritage metadata, different iterative procedures can be useful. The straight forward approach generates complete metadata for one object after the other. Common information, e.g., about project participants or used camera, will be generated at first appearance and subsequently reused.

Each procedure depends on the needed time of an iteration and on the sequence of the iteration cycles. An iteration which includes manual metadata generation, for example, is very slow, whereas auto-generation is really fast. Thus, what to do and when is dependent on the specific task. For example, during digitization of an archaeological site only auto-generated metadata is useful while working in the field; no manual metadata input is easily feasible. Therefore, an appropriate procedure could be composed of three main cycles:

1. Generating background metadata in the office
2. Automated and semi-automated creation of important data in the field
3. Completing missing information afterwards in the office

First, especially information that is needed in the field has to be generated as background metadata. Furthermore, a distinction between immediately requested data (needed in the field) and metadata that can be generated at a later point in time has to be made. Specific data has to be generated immediately, when information is at risk of being lost or destroyed. Retrospectively, information about a specific artifact, its exact position in the archeological site or the participating archeologists might be hard to remember.

5 A Generic Approach for Generating Cultural Heritage Metadata

Contents

5.1	Generic Dynamic Input Forms	38
5.2	Generic Formats	40
5.2.1	Forms Definition Format	40
5.2.2	Intermediate Storage Format	41
5.2.3	Template Definition Format	42
5.3	Directory Structure Information	44

In this chapter a generic approach for generating cultural heritage metadata based on the previous chapters of this thesis will be presented. The approach has appeared previously in the paper “A Generic Approach for Generating Cultural Heritage Metadata” [39].

The proposed approach is a three-step process for generating metadata for each file in a directory structure. First, the user is asked to enter only the essential metadata with the help of dynamically created input forms (see Section 5.1). Common information is collected only once and is referenced for reuse by unique link keys. Second, the essential data is stored as key-value pairs at a redundancy-free intermediate format for reducing storage consumption and fast reuse. Finally, the intermediate format is translated into a standardized output format like CIDOC-CRM / CRMdig encoded in RDF by using (exchangeable) templates.

Using generic formats as flexible solution for defining such formats enables a fast and easy reaction on changes of the requirement (see Section 5.2). Listing 5.1 shows an example of output generated by this system. Only the essential data require the user to fill in an input form manually. Since CIDOC-CRM / CRMdig is event-centric, the user has to define a hierarchical chain of events. Interestingly, this event chain corresponds with the directory structure. To speed up user input, drop-down boxes offer as few choices as possible, i.e., only those collected from parent directories (see Section 5.3).

Field	Description
Text	normal text
Data Selection	select existing data
Date / Time	proper date / time
Note	notes and other additional information
Group	grouping information
Switch	select mutually exclusive information

Table 5.1: Basic Generic Form Fields are necessary to construct a cultural heritage form.

The novel contribution of this approach is (a) the separation of intermediate data from RDF-generation templates and (b) that the directory structure can be used as a scope hierarchy to speed up the metadata generation process. The benefit is that generating rich metadata/paradata (CIDOC-CRM / CRMdig encoded in RDF) becomes less tedious, painful and costly for all institutions carrying out CH digitization.

5.1 Generic Dynamic Input Forms

The first step of metadata generation always requires user input. The common way to enter information is a form with a list of fields, each field being a key-value pair. The value can be a Boolean (check box), an integer or float number, or text (input field). For cultural heritage, correct semantic connections between the data are important. To avoid typing errors in semantic links, drop-down boxes (with a limited range of options) are preferable over text input fields. Furthermore, the same key can often have more than one value (museum object with multiple materials), i.e., more fields must be added dynamically to the form. Usually, input forms are defined by the developer statically in software. If changes are needed, the form code has to be edited and recompiled. This is clearly inappropriate for the purposes of this approach; instead, the form should be described in a generic format and generated at runtime.

Form Fields. Table 5.1 shows the types of form fields that are used. Standard fields are a date/time field and a text field for normal text, e.g., a name, a title or a label. Important are fields for selecting from existing alternatives to reliably reuse and link to previously entered data. Longer text for notes or annotations is entered into note fields. For grouping logically connected fields a group field is useful; and a switch field is needed for mutually exclusive fields where one or the other field should be filled, but not both.

```

1 <rdf:RDF xml:lang="en"
2   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
4   xmlns:crm="http://www.ics.forth.gr/isl/rdfs/3D-COFORM_CIDOC-
   CRM.rdfs#"
5   xmlns:crmdig="http://www.ics.forth.gr/isl/rdfs/3DCOFORM_CRMdig.
   rdfs#">
6   <crm:E7.Activity rdf:about=
7     "uuid:3a58de86-73ae-4bc2-a67d-97b7fc12a208">
8     <crmdig:L4F.has_preferred_label>
9       2012 Project Example
10    </crmdig:L4F.has_preferred_label>
11    <crm:P2F.has_type>
12      <crm:E55.Type rdf:about="http://www.3d-coform.eu/
   EventType/project" />
13    </crm:P2F.has_type>
14    <crmdig:L29F.has_responsible_organization>
15      <crm:E40.Legal_Body rdf:about=
16        "uuid:0d12058b-974e-428d-95d5-6e4413136b8b"/>
17    </crmdig:L29F.has_responsible_organization>
18    <crmdig:L31F.has_starting_date-time
19      rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">
20      2012-05-01T08:20:00Z
21    </crmdig:L31F.has_starting_date-time>
22    <crmdig:L32F.has_ending_date-time
23      rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">
24      2012-05-31T18:12:00Z
25    </crmdig:L32F.has_ending_date-time>
26    <crm:P3F.has_note>some notes</crm:P3F.has_note>
27  </crm:E7.Activity>
28 </rdf:RDF>

```

Listing 5.1: Example: The metadata of a Project Event generated by the generic solution approach; CIDOC-CRM / CRMdig encoded in RDF. Data acquired from the user are highlighted; UUIDs, typically from objects, are selected in drop-down boxes (stable link).

Field	Description
TextField	normal text, e.g., name, title, label
MultiTextField	normal text; enter multiple times
ComboField	select existing data (drop-down menu)
MultiComboField	select existing data multiple times
DateTimeField	date and time
NoteField	notes and other additional information
Tab	grouping fields on tabs
Switch	select mutually exclusive fields
SubForm	embed another form
MultiSubForm	embed another form multiple times
SelectFileField	file selection

Table 5.2: This table presents the Node types of the Forms Definition Format.

5.2 Generic Formats

Customizable generic XML-based formats will be used for the input forms (see Forms Definition Format), for the storage format (see Storage Format Definition), and for the translation templates (see Template Definition Format). Some form format information is also used for the intermediate storage format, e.g. defined names.

5.2.1 Forms Definition Format

Forms Definition Format is a simple and flexible format for dynamic input forms. Each XML node has a unique Name attribute for identification. It is also used as storage node name in the intermediate format. The form node has a Title attribute for the form title (displayed label) and a Group attribute for the storage group. It is used for grouping storage instances inside the intermediate format. The SelectKey attribute of the form node defines the form fields used to select this form in a list of form instances. The KeyText attribute contains the field text shown to the user. Combofield and MultiCombofield nodes have a FormName attribute, specifying the name of the linked form. The different form fields needed to construct a proper form and handle different input requirements are shown in Table 5.2. Listing 5.2 shows a code example defining a suitable Project Event form.

```
1 <Form Name="ProjectEvent"  
2     Title="Project Event"  
3     Group="ProjectEvents"  
4     SelectKey="EventTitle">  
5 <TextField Name="EventTitle"  
6     KeyText="Event Title" />  
7 <ComboBox Name="ResponsibleOrganization"  
8     KeyText="Responsible Organization"  
9     FormName="LegalBody"/>  
10 <DateTimeField Name="StartDateTime"  
11     KeyText="Starting Date/Time" />  
12 <DateTimeField Name="EndDateTime"  
13     KeyText="Ending Date/Time*" />  
14 <NoteField Name="Note"  
15     KeyText="Note*" />  
16 </Form>
```

Listing 5.2: Example of Forms Definition Format code which defines a Project Event form.

5.2.2 Intermediate Storage Format

The *Intermediate Metadata Storage Format* format is focused on low redundancy and human readability (see Listing 5.3). It is an easy structured XML-based storage format for the essential data. Each form is represented as an XML metadata node and each form field has its corresponding XML sub node. Form nodes of the same form are grouped into one XML form group node, for faster human search. The XML node names are defined in the forms definition format. A form field needs a unique name only in the scope of the form. Each form instance has its own unique ID, a sustainable UUID. Semantic links between forms are established using this ID. If only a part of a form is required for a semantic link, the form can be split into sub-forms, and each sub-form gets a unique ID for semantic linking.

The intermediate format is completely independent from the metadata result format, using the conversion templates explained in the Section 4.3.4. Therefore, it can be easily transformed into an arbitrary output format, e.g., into standard formats like RDF or JSON. The intermediate format is platform independent and very flexible. The metadata file is typically stored in a `./metadata` sub-folder of the folder where the data file resides.

```
1 <ProjectEvent ID="uuid:3a58de86-73ae-4bc2-a67d-97b7fc12a208">
2   <EventTitle>2012 Project Example</EventTitle>
3     <ResponsibleOrganization FormName="LegalBody">
4       uuid:0d12058b-974e-428d-95d5-6e4413136b8b
5     </ResponsibleOrganization>
6   <StartDateTime>2012-05-01T08:20:00Z</StartDateTime>
7   <EndDateTime>2012-05-31T18:12:00Z</EndDateTime>
8   <Note>some notes</Note>
9 </ProjectEvent>
```

Listing 5.3: A Project Event represented in the Intermediate Format.

5.2.3 Template Definition Format

This format describes how to translate each field of each form from the intermediate storage format to a proper standard format like CIDOC-CRM / CRMdig encoded in RDF. A conversion template contains an output code skeleton that is recursively expanded using the information from the intermediate format to generate the output format. Listing 5.4 shows an example, the translation of a Project Event data to CIDOC-CRM / CRMdig encoded in RDF. For storage efficiency each form, such as ProjectEvent, has a short and a long RDF version. The short version (`<RDFShort>`) contains only base information and the long version (`<RDF>`) contains all fields. In a field node like EventTitle the RDF code for this field is located. Wildcards starting and ending with “`__`” will be replaced. A wildcard like `__Text__` or `__ID__` will simply be replaced by text or an ID, etc. The source of information to insert is always the stored intermediate format data. The special wildcard `__RDF__` is a placeholder for further RDF code that is recursively generated from other templates, or from a template sub-form like ResponsibleOrganization.

```

1 <ProjectEvent>
2   <RDF>
3     <crm:E7.Activity rdf:about="__ID__">__RDF__</crm:E7.
4       Activity>
5   </RDF>
6   <RDFShort>
7     <crm:E7.Activity rdf:about="__ID__" />
8   </RDFShort>
9   <EventTitle>
10    <crmdig:L4F.has_preferred_label>__Text__</crmdig:L4F.
11      has_preferred_label>
12    <crm:P2F.has_type>
13      <crm:E55.Type rdf:about="http://www.3d-coform.eu/EventType
14        /project" />
15    </crm:P2F.has_type>
16  </EventTitle>
17  <ResponsibleOrganization>
18    <crmdig:L29F.has_responsible_organization>
19      __RDF__
20    </crmdig:L29F.has_responsible_organization>
21  </ResponsibleOrganization>
22  <StartDateTime>
23    <crmdig:L31F.has_starting_date-time
24      rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">
25      __Text__
26    </crmdig:L31F.has_starting_date-time>
27  </StartDateTime>
28  <EndDateTime>
29    <crmdig:L32F.has_ending_date-time
30      rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">
31      __Text__
32    </crmdig:L32F.has_ending_date-time>
33  </EndDateTime>
34  <Note>
35    <crm:P3F.has_note>__Text__</crm:P3F.has_note>
36  </Note>
37 </ProjectEvent>

```

Listing 5.4: Example Template Definition for the translation of a Project Event from intermediate format to CIDOC-CRM / CRMdig encoded in RDF.

5.3 Directory Structure Information

The directory structure of a typical acquisition campaign naturally corresponds with the event chain of CIDOC-CRM / CRMdig (see Figure 5.1). Typical folder levels are:

- Project root directory – contains all project relevant data
- Object directory – according to the plan, one object is captured after another
- Sequence directory – one folder for each series of object measurements

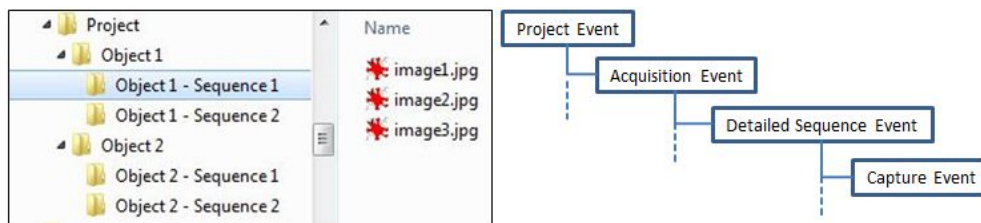


Figure 5.1: The directory structure (left) is similar to the CIDOC-CRM / CRMdig event-chain (right). Therefore the information can be used for generating metadata. (Previously published in [39])

CRMdig defines appropriate event types, namely the project creation event, the (object) acquisition event, the detailed sequence event, and finally there is one capture event for every single photograph (i.e., jpeg file). The metadata of an event is stored on the respective directory level, next to the dataset files in the file system. All relevant metadata can be found on the path to the root, since in CRMdig events typically reference to higher order events: capture to sequence events, acquisition to project events, etc. Since the metadata XML files are stored in the same hierarchy, referenced metadata can be found easily and fast.

6 MetadataGenerator

Contents

6.1	Introduction	46
6.2	Features	47
6.3	System Design	48
6.3.1	Graphical User Interface (GUI)	48
6.3.2	Storage	53
6.3.3	Transform	53
6.4	Avoiding Errors	53
6.5	CIDOC-CRM / CRMdig Forms	54
6.5.1	Project Event	54
6.5.2	Capture Event	56
6.5.3	Detailed Sequence Event	57
6.5.4	Acquisition Event	58
6.5.5	Process Event	61
6.6	Ingest Script Generation	64
6.7	Implementation Details	65

The *MetadataGenerator* (MG) is an implementation of the generic solution approach presented in Chapter 5. It is one of the tools developed in the context of the 3D-COFORM project. The MG as implementation example and the idea behind were first described by Schröttner et al. [39] and later as a part of Pan et al. [30], [31] and [32]. This Chapter is based on all these papers as well as on the MG user manual [38] and describes the development and implementation in more detail. The purpose of the developed tool is to generate semi-automated cultural heritage metadata which can be stored in CIDOC-CRM / CRMdig conform RDF-files. MG is implemented in C# using MS Visual Studio 2010, the graphical user interface (GUI) is based on Windows Presentation Foundation (WPF) [26] framework.

6.1 Introduction

Every digital object must be provided with additional meta-information to establish rich semantic connections that enable retrieval. An interesting aspect of using CRMdig is that a full documentation of the processing chain, the so-called paradata, has the potential to make the processing steps reproducible. This can be taken to the point that a digital object can actually be recomputed from the original measurements; and by replacing some of the processing steps with improved methods and algorithms, a better version of the resulting digital object can, in principle, be generated automatically at any time.

Rich metadata, process descriptions, as well as historical facts encoded in CIDOC-CRM/CRMdig, can be extremely helpful for many purposes. The only caveat is that the production of metadata is not for free; it can in fact require substantial efforts in terms of cost and time. It is not much fun to describe the fact that, e.g., one image is a cropped, resized version of another image, and to enter again the respective process parameters.

To streamline the production of process metadata for larger numbers of datasets, the MG was developed. It is designed for generating metadata and paradata re-using as much information (which was already entered before) as possible. The idea is to ask the user only for logical connections and other bits of information that are required but missing. When the user initiates the process of collecting information, the first (automatic) step is to harvest all useable information. This process exploits the folder hierarchy of the data in the file system, which corresponds very often to the structure of the processing chain (descending from super- to sub-event). The second step is to present all collected information to the user using auto-generated forms based on CIDOC-CRM and CRMdig. The user is asked to fill in the missing information and to confirm the data that were collected. All information is encoded in an XML based intermediate format and stored next to the digital object files. The benefit of using an intermediate format (instead of using the verbose RDF encoding directly) is that it is (a) easier to check, edit and change the information, and (b) easier to convert it again to RDF when RDF policies change. RDF is the standard output format of the MG for CIDOC-CRM and CRMdig since that is the format that is expected by the RI. It is automatically processed to assemble the semantic network in the MR component of the RI.

MG also offers the option to generate an upload script to transfer all datasets and metadata files of a project and to ingest them to the RI. Projects usually contain several hundreds of datasets, particularly when using 3D reconstruction from photographs. Note that the ingestion of metadata (first) and binary data (afterwards) can be decoupled, which is especially important for field

campaigns with bad internet connection. The metadata should be transferred as soon as possible to the OR-Central, e.g. for peer review, and also to the MR to enable queries. Using the upload script generated by MG, the heavy binary data can be uploaded independently at a later point in time, e.g. after returning from the field campaign.

MG stores the basic metadata as well as the main project data in the local file system and requires no online connection to the RI. Therefore, it is primarily designed for an offline scenario. Besides for field campaigns, this can also be useful for situations where the metadata are temporarily inconsistent, e.g., when identities must be replaced, joined, or differentiated over a multitude of files.

6.2 Features

Essentially, the MG is a tool for mass generation of cultural heritage metadata. In the following, the main features of the MG will be listed and described:

- MG stores the metadata in the local file system, next to the cultural heritage digital object file.
- It is possible to edit and update the (already entered) metadata.
- Preparing the metadata files for a project is an offline process which can even be done during an excavation campaign in a remote location. This is recommended, as early metadata capture reduces errors.
- Once all files and their metadata are prepared, MG can generate ingest scripts to upload all datasets with their metadata into the 3D-COFORM Repository Infrastructure (RI).
- Generating an update script for already ingested metadata is also possible.
- For script generation some additional information is used, like the directory hierarchy of digital object files for the RI group hierarchy.
- MG is mainly an offline tool, no online connection to the RI is needed for entering metadata. Only the upload at the end of the metadata entering process needs an online connection to the RI.

6.3 System Design

The MG is composed of three main modules (see Figure 6.1). (a) The GUI, which basically consists of the main window and forms for collecting metadata from the user. (b) The Storage of the generated metadata by using an intermediate format. This module manages the entered data from the GUI and provides a reuse. (c) The Transform module which converts the intermediate format into an (arbitrary) output format like CIDOC-CRM / CRMdig encoded in RDF. Moreover, the Forms Definition describes forms by using proper fields and labels. Additionally, it implicitly defines how to store the acquired information into an Intermediate Storage Format. The Template Definition describes rules for transforming an intermediate result into another (standard) output format.

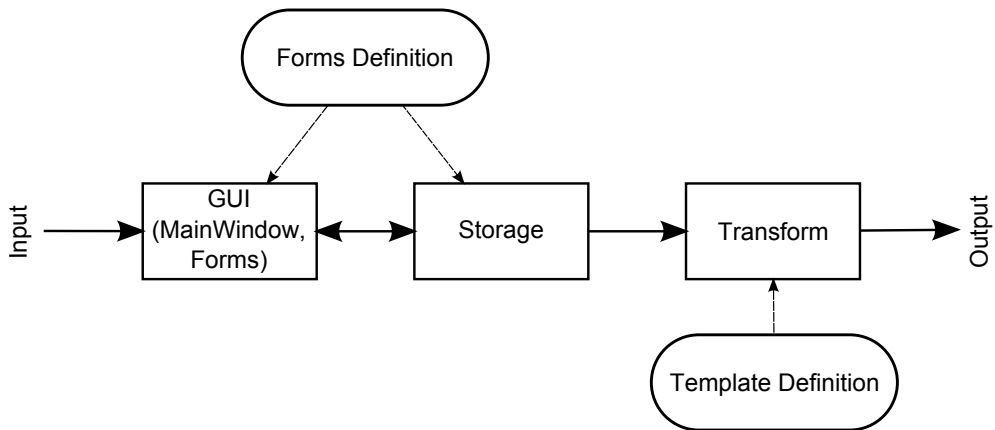


Figure 6.1: An overview showing the MetadataGenerator dataflow and main modules. It consists of 3 modules: GUI managing user input, Storage which uses Intermediate Storage Format, and Transform for converting into a standard output format.

6.3.1 Graphical User Interface (GUI)

The GUI of the MG basically consists of the main window and several dynamically generated input forms. The main window (see Figure 6.2) provides two primary opportunities: (1) selecting binary files as input or output of CIDOC-CRM / CRMdig events and (2) picking a new form for entering metadata by the menu. The MG manual (also available per Help menu) provides a detailed description of further usage.

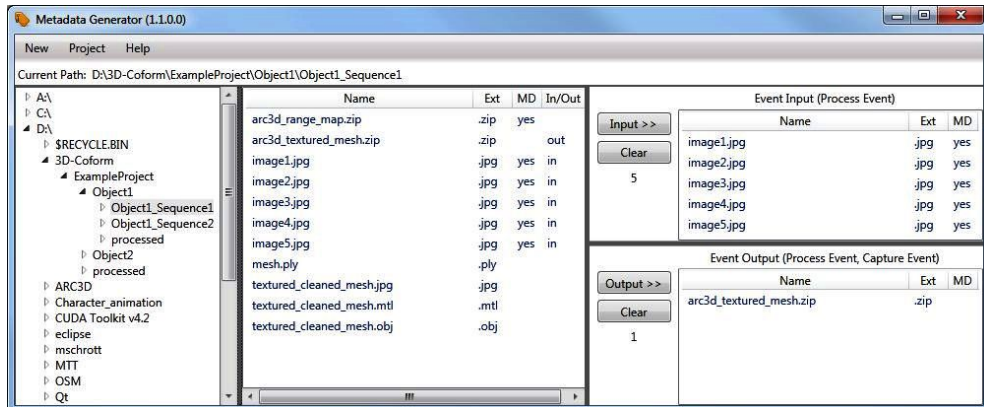


Figure 6.2: The MetadataGenerator main window is composed of a directory view (left) for selecting the current working directory, a file view (middle) for selecting the files as input or output, and the current selection (right) used for the next event.

Forms The MG uses a Form Definition Format file (see Section 5.2.1) to dynamically generate a form at run-time. The collected form data acts as logical group of metadata which has one unique ID for retrieval, reuse and linking. A form is defined as a sequence of fields. Table 6.1 lists the different types of input fields which are available for collecting information. An example form code which includes all kinds of input field is shown in Listing 6.1 which generates the form example shown in Figure 6.3.

Some fields just help to enter key-value pairs, e.g. the TextField, but other fields provide additional opportunities, such as the ComboField which enables the selection of existing logical data groups. This selection is realized by a drop down menu. Figure 6.4 shows an example acquisition event chain where each event is a form representing a logical metadata group which is connected by selection of the previous event with a ComboField. If the data is missing and can not be selected, 'New' generates a new form for entering the missing data. The 'Edit' button opens the linked data form for reviewing or editing. Other fields are designed for assigning values to the same key multiple times. The MultiTextField, for example, provides the opportunity to assign different values to one key (e.g., used for alternative event labels). Adding a new field can be done at run-time by the + button and the - button removes a field (one field always remains).

A generic form which will be dynamically generated needs opportunities to control the flow of input fields. Furthermore, a logical grouping of input fields improves the readability and reusability. This can be done by using SubForm, a flow control field which includes another form. The fields of the included form will not differ visually from fields of the original form, but

Field	Usage / Code Example / Result Example
TextField	normal text, e.g., name, title, label
	<code><TextField Name="Label"KeyText="Label"/></code>
	in Figure 6.3 the Label field
MultiTextField	normal text; enter multiple times
	<code><MultiTextField Name="OtherLabel" KeyText="Other Label*"/></code>
	in Figure 6.3 the Other Label field
ComboField	select existing data (drop-down menu)
	<code><ComboField Name="ProjectEvent" KeyText="Project Event" FormName="ProjectEvent"/></code>
	in Figure 6.3 the Project Event field
MultiComboField	select existing data multiple times
	<code><MultiComboField Name="Type" KeyText="Type" FormName="AcquisitionType"/></code>
	in Figure 6.3 the Type field
DateTimeField	date and time
	<code><DateTimeField Name="DateTime" KeyText="Date/Time*"/></code>
	in Figure 6.3 the Date/Time field
SelectFileField	select a binary file
	<code><SelectFileField Name="FileName" KeyText="File Name" /></code>
	in Figure 6.3 the File Name field
NoteField	notes and other additional information
	<code><NoteField Name="Note" KeyText="Note*"/></code>
	in Figure 6.3 the Note field

Table 6.1: The input fields of the Forms Definition Format with a short usage description and references to the code and result examples.

```

1 <Form Name="InputFieldsDemo"
2   Title="Input Fields Demo"
3   Group="Demos"
4   SelectKey="Label"
5   Storage="Local">
6   <TextField Name="Label" KeyText="Label"/>
7   <MultiTextField Name="OtherLabel" KeyText="Other Label*"/>
8   <ComboField Name="ProjectEvent"
9     KeyText="Project Event"
10    FormName="ProjectEvent"/>
11  <MultiComboField Name="Type"
12    KeyText="Type"
13    FormName="AcquisitionType"/>
14  <DateTimeField Name="DateTime" KeyText="Date/Time*"/>
15  <SelectFileField Name="FileName" KeyText="File Name" />
16  <NoteField Name="Note" KeyText="Note*"/>
17 </Form>

```

Listing 6.1: Input fields demo code showing definitions of the MetadataGenerator input fields types: TextField, MultiTextField, ComboField, MultiComboField, DateTimeField, SelectFileField, NoteField.

The screenshot shows a window titled "New Input Fields Demo" with a yellow background. It contains several input fields and buttons:

- UUID:** A text field containing "uuid:cc6b612e-6ebe-4dc0-b574-244c597376ad" with "New" and "Edit" buttons.
- Label:** A text field containing "my label".
- Other Label*:** A text field containing "my other label" with "+" and "-" buttons.
- Project Event:** A dropdown menu showing "2013 Project Example Graz (uu)" with "New" and "Edit" buttons.
- Type:** A dropdown menu showing "data acquisition (uuid:5df380b)" with "+" and "-" buttons.
- Date/Time*:** A date/time picker showing "07.06.2013" and "15" for the day, with time set to "10:00:00".
- Note*:** A text area containing "some notes".

At the bottom right, there are "Save" and "Cancel" buttons.

Figure 6.3: Form demo example showing input fields of the MetadataGenerator: Label, a TextField; Other Label, a MultiTextField; Project Event, a ComboField; Type, a MultiComboField; Date/Time, a DateTimeField; Note, a NoteField. The * symbol after a key text denotes an optional field.

the SubForm is a logical group which gets its own unique ID for reuse. If a SubForm is needed multiple times, the MultiSubForm field can be used to create a visually separated part for the SubForm fields, so that it can be duplicated by + or - buttons. Another option of visual grouping is the Tab field which helps to show input fields on different tabs inside the same form. For collecting data from either input fields or other fields, the Switch field can be applied. This field restricts the user opportunities to certain logical groups of input fields. Users select the desired group of items with the help of radio buttons. The collected information of tabs is usually stored together as output of one form, whereas Switch field simply stores the information of the currently selected group of items.

The figure displays four overlapping windows from the MetadataGenerator application, illustrating an acquisition event chain. Each window represents a logical metadata group with its own unique ID (UUID). The windows are:

- New Project Event:** Contains fields for UUID (uuid:8fb01174-2ae7-4fac-b2c5-b0f26fd465c7), Event Title (2011 Project Example Graz), and Responsible Organization (Graz University of Technology, Gi).
- New Acquisition Event:** Contains fields for UUID (uuid:f1227678-737d-44e1-85cf-5b5951550695), Event Title (2011 Acquisition Object1 Graz), Type (photography), and a 'Project Event' dropdown menu showing '2011 Project Example Graz'.
- New Capture Event:** Contains fields for UUID (uuid:a50e623c-e5ca-449b-b7c7-a24affeda821), Event Title (2011 Capture image1.jpg Graz), and a 'Detailed Sequence Event' dropdown menu showing '2011 Detailed Sequence Object1'.
- New Detailed Sequence Event:** Contains fields for UUID (uuid:7a48bc83-0122-41a7-8fcc-55e80a31a744), Event Title (2011 Detailed Sequence Object1 Grz), Type (photography), and an 'Acquisition Event' dropdown menu showing '2011 Acquisition Object1 Graz'. It also includes fields for Person* (Martin Schröttner), Device (Nikon D50 Camera), Additional Device*, Starting Date/Time*, and Ending Date/Time*.

Arrows indicate the flow of data from the 'Project Event' dropdown in the 'New Acquisition Event' window to the 'Acquisition Event' dropdown in the 'New Detailed Sequence Event' window, showing how each event is connected to the previous one.

Figure 6.4: An example acquisition event chain expressed by MetadataGenerator forms. Each event in its form represents a logical metadata group which is connected by selection of the previous event with a ComboField. (Image previously published in [32])

6.3.2 Storage

When all requested information has been filled in and the user decides to store it, the MG applies a storage solution based on the Intermediate Storage Format presented in Section 5.2.2. This format is suitable for the MG, as the stored data can easily be reused and transformed into any other standardized format. Reusage and referencing are realized by linking with unique IDs of stored logical data groups. Another advantage of this solution is that the storing of form-data is indirectly defined by the Forms Definition Format. Furthermore, stored data can be easily restored and filled into the appropriate new form, e.g. for review or update. Other characteristics of this format are its human readability and low storage consumption.

The storage is realized as XML-file in the file system. The metadata of each binary file encoded in Intermediate Format are stored in one XML-file per directory. For reuse or retrieval, only XML-files of the current scope are used to build a search database on demand. The current scope includes metadata files from the current directory to the root directory (or project main directory) and the metadata files of all current child directories. Thus, the scope corresponds to the CIDOC-CRM / CRMdig event chain (see Section 5.3).

6.3.3 Transform

Initiated by the user, a project can be transformed into a standard storage format like CIDOC-CRM / CRMdig encoded in RDF. All metadata of a project will be transformed by using the templates defined by the Template Definition Format presented in Section 5.2.3. As a result, a metadata file for each binary file is generated. For each directory common metadata is collected in one background metadata file, which reduces storage consumption and avoids ambiguity. After transforming, the binary files and its metadata as well as the background metadata are ready for upload to the RI.

6.4 Avoiding Errors

A big challenge at metadata generation is to avoid errors caused by users as well as during auto-generation of data. As it is difficult to avoid errors completely, it must be still possible to correct them. Thus, an opportunity to update or change metadata must be given. The RI provides the opportunity to update; the new metadata version replaces the former, but all older versions will be archived to keep referential integrity.

Collecting information from users can easily cause mistakes. Therefore, it is crucial to keep the structure of forms being used to ask the user for detailed information as simple and logical as possible. Thus, user errors can be reduced by splitting the requested information into logical groups, which are presented as own parts inside a form. Furthermore, for reasons of clarity and comprehensibility it can be useful to split a form into two or more separated forms. Another way to prevent user errors may be pre-filling of forms with auto-generated information or suggestions. The pre-filling has the additional benefit to help to speed up the user input.

Another field in which errors can occur is the auto-generation of metadata. The error depends on the used technique for auto-generating the information. For example, defective results could be obtained at automatic mimetype detection using only the file extension, as different mimetypes use the same extension. Combining different techniques could increase the reliability of the result. However, an error can still occur but with lower probability. A residual error could be avoided by user review.

6.5 CIDOC-CRM / CRMdig Forms

In this section the main forms of the CIDOC-CRM / CRMdig event chain will be described in more detail. For each event the forms definition code, a screenshot of the result and a user-guide based on the MG manual [38] will be illustrated.

6.5.1 Project Event

The starting event of the event chain is the Project Event. Every other event is directly or indirectly (through other events) linked to a Project Event. So the first step is to create a new Project Event and store the metadata in the root directory of the project. All associated project files have to be placed in a subdirectory of the root directory in order to ensure a useful metadata hierarchy for search and retrieval. The Project Event defined in Forms Definition Format is shown in Listing 6.2 with the result window presented in Figure 6.5.

In the following, a user guide for the Project Event is provided:

1. Select the project root directory in the MG
2. Get a new "Project Event" form via the menu: New → Event → Project Event
3. Fill (enter or select) the following form fields (Figure 6):
 - Event Title: The event title, according to the rules
 - Responsible Organization: The responsible organization for this project
 - Starting Date/Time: The project starting date and time
 - Ending Date/Time: The project ending date and time (optional)
4. Check if the project root directory (displayed on the window bottom) is correct. All project files should be in the subdirectories of the root directory. If necessary, go one folder up (by pressing the "One Folder Up"-Button)
5. Save the input

```
1 <Form Name="ProjectEvent"
2   Title="Project Event"
3   Group="ProjectEvents"
4   SelectKey="EventTitle"
5   Storage="User">
6 <TextField Name="EventTitle"
7   KeyText="Event Title" />
8 <ComboField Name="ResponsibleOrganization"
9   KeyText="Responsible Organization"
10  FormName="LegalBody"/>
11 <DateTimeField Name="StartDateTime"
12   KeyText="Starting Date/Time" />
13 <DateTimeField Name="EndDateTime"
14   KeyText="Ending Date/Time*" />
15 <NoteField Name="Note" KeyText="Note*" />
16 </Form>
```

Listing 6.2: The Project Event form described in Forms Definition Format.

Figure 6.5: This figure presents the Project Event Form generated at runtime by using the Forms Definition.

6.5.2 Capture Event

If a file is captured from a real object like a jpg-image, the metadata is represented as a Capture Event. The Capture Event defined in Forms Definition Format is shown in Listing 6.3 with the result window presented in Figure 6.6.

In the following, a user guide for the Capture Event is provided:

1. Select all images of a sequence and add them to the output by the "Output >>" button
2. Get a new "Capture Event"-form via the menu: New → Event → Capture Event
3. Fill (enter or select) the following form fields (some fields are already prefilled):
 - Event Title: The event title, according to the rules
 - Detailed Sequence Event: The (super) detailed sequence event
 - Digital Object / Multipart Digital Object: Choose if the captured object is a normal digital object (one file) or a multipart digital object (several files forming one logical object)

- For a digital object:
 - File Name: The file name of the binary (normally preselected)
 - Object Type: The object type of the binary (at least one)
 - Mime/File Type: The mime type of the binary (normally preselected)
 - For a multipart digital object:
 - Label/Name: The label/name of the object
 - Object Type: The object type (at least multipart digital object)
 - Part Digital Objects: The single parts of the multipart object
 - Object Notes: Some object notes (optional)
4. Save the input
 5. For the next Capture Event the entered input of the last Capture Event will be used as template.

```

1 <Form Name="CaptureEvent"
2   Title="Capture Event"
3   Group="CaptureEvents"
4   SelectKey="EventTitle"
5   OutputNeeded="True">
6   <TextField Name="EventTitle"
7     KeyText="Event Title" />
8   <ComboField Name="DetailedSequenceEvent"
9     KeyText="Detailed Sequence Event"
10    FormName="DetailedSequenceEvent"/>
11   <SubForm Name="DigitalObject"
12     FormName="DigitalObject" />
13 </Form>

```

Listing 6.3: The Capture Event form described in Forms Definition Format.

6.5.3 Detailed Sequence Event

All Capture Events from the same object forming a unit for processing are collected in a Detailed Sequence Event order to store the common metadata information. The Detailed Sequence Event defined in Forms Definition Format is shown in Listing 6.4 with the result window presented in Figure 6.7.

Figure 6.6: This figure shows the Capture Event form generated at runtime by using the Forms Definition.

In the following, a user guide for the Detailed Sequence Event is provided:

1. Fill (enter or select) the following form fields:
 - Event Title: The event title, according to the rules
 - Type: The event type, e.g. photography event
 - Acquisition Event: The (super) acquisition event
 - Person: The person who has created the binary files, e.g. images (optional)
 - Device: The used device, e.g. the photo camera
 - Additional Device: Any particular, additionally used device, such as a camera lens (optional)
 - Starting Date/Time: The sequence starting date and time (optional)
 - Ending Date/Time: The sequence ending date and time (optional)
2. Save the input

6.5.4 Acquisition Event

All Detailed Sequence Events of the same (acquired) object are collected in an Acquisition Event, which stores the common information of all Detailed Sequence Events. The Acquisition Event defined in Forms Definition Format is shown in Listing 6.5 with the result window presented in Figure 6.8.

```

1 <Form Name="DetailedSequenceEvent"
2   Title="Detailed Sequence Event"
3   Group="DetailedSequenceEvents"
4   SelectKey="EventTitle">
5   <TextField Name="EventTitle"
6     KeyText="Event Title" />
7   <MultiComboField Name="Type"
8     KeyText="Type"
9     FormName="AcquisitionType"/>
10  <ComboField Name="AcquisitionEvent"
11    KeyText="Acquisition Event"
12    FormName="AcquisitionEvent"/>
13  <MultiComboField Name="Person"
14    KeyText="Person*"
15    FormName="Person"/>
16  <ComboField Name="Device"
17    KeyText="Device"
18    FormName="Device"/>
19  <MultiComboField Name="AdditionalDevice"
20    KeyText="Additional Device*"
21    FormName="Device"/>
22  <DateTimeField Name="StartDateTime"
23    KeyText="Starting Date/Time*" />
24  <DateTimeField Name="EndDateTime"
25    KeyText="Ending Date/Time*" />
26 </Form>

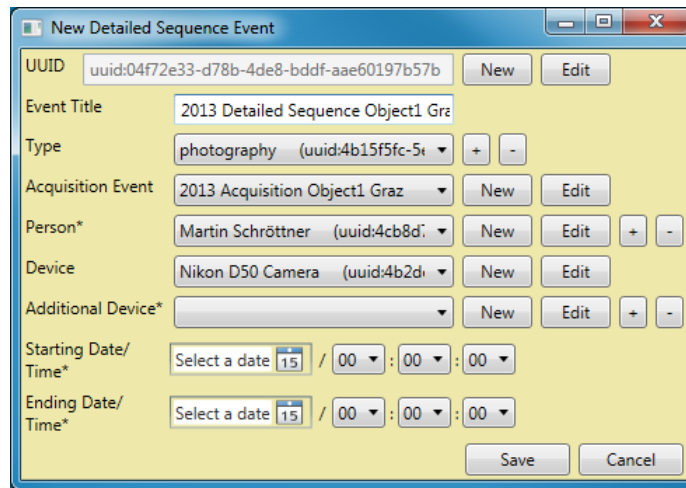
```

Listing 6.4: The Detailed Sequence Event form described in Forms Definition Format.

In the following, a user guide for the Acquisition Event is provided:

1. Fill (enter or select) the following form fields (Figure 6.8):
 - Event Title: The event title, according to the rules
 - Type: The event type, e.g. photography event
 - Project Event: The (super) project event
 - Acquisition Location: The acquisition location, e.g. where the photos were made
 - Acquired Object: The acquired (real) object
 - Starting Date/Time: The acquisition starting date and time (optional)
 - Ending Date/Time: The acquisition ending date and time (optional)
2. Check if the save directory (displayed on the window bottom) is correct, only events stored in the same directory or in a subdirectory are able to provide a selection of this acquisition event as super event
3. Save the input

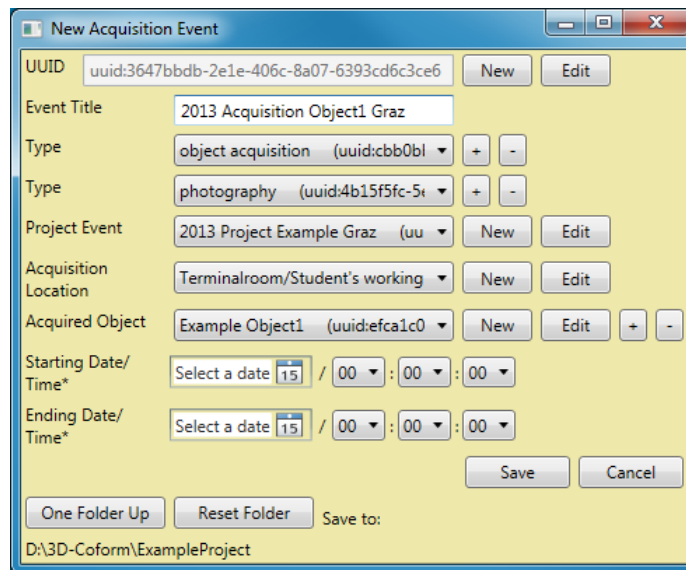
6 MetadataGenerator



The screenshot shows a window titled "New Detailed Sequence Event". It contains the following fields and controls:

- UUID: with "New" and "Edit" buttons.
- Event Title:
- Type: with "+" and "-" buttons.
- Acquisition Event: with "New" and "Edit" buttons.
- Person*: with "New", "Edit", "+", and "-" buttons.
- Device: with "New" and "Edit" buttons.
- Additional Device*: with "New", "Edit", "+", and "-" buttons.
- Starting Date/Time*:
- Ending Date/Time*:
- Buttons: "Save" and "Cancel".

Figure 6.7: This figure presents the Detailed Sequence Event form generated at runtime by using the Forms Definition.



The screenshot shows a window titled "New Acquisition Event". It contains the following fields and controls:

- UUID: with "New" and "Edit" buttons.
- Event Title:
- Type: with "+" and "-" buttons.
- Type: with "+" and "-" buttons.
- Project Event: with "New" and "Edit" buttons.
- Acquisition Location: with "New" and "Edit" buttons.
- Acquired Object: with "New", "Edit", "+", and "-" buttons.
- Starting Date/Time*:
- Ending Date/Time*:
- Buttons: "Save" and "Cancel".
- Folder navigation: "One Folder Up", "Reset Folder", and "Save to:".
- Path:

Figure 6.8: This figure shows the Acquisition Event form generated at runtime by using the Forms Definition.

```

1 <Form Name="AcquisitionEvent"
2   Title="Acquisition Event"
3   Group="AcquisitionEvents"
4   SelectKey="EventTitle"
5   Storage="User">
6   <TextField Name="EventTitle"
7     KeyText="Event Title" />
8   <MultiComboField Name="Type"
9     KeyText="Type"
10    FormName="AcquisitionType"/>
11  <ComboField Name="ProjectEvent"
12    KeyText="Project Event"
13    FormName="ProjectEvent"/>
14  <ComboField Name="AcquisitionLocation"
15    KeyText="Acquisition Location"
16    FormName="Location"/>
17  <MultiComboField Name="AcquiredObject"
18    KeyText="Acquired Object"
19    FormName="PhysicalObject"/>
20  <DateTimeField Name="StartDateTime"
21    KeyText="Starting Date/Time*" />
22  <DateTimeField Name="EndDateTime"
23    KeyText="Ending Date/Time*" />
24 </Form>

```

Listing 6.5: The Acquisition Event form described in Forms Definition Format.

6.5.5 Process Event

When a file is processed or modified by software, it is represented as a Process Event in the metadata.

The Process Event defined in Forms Definition Format is shown in Listing 6.6 with the result window presented in Figure 6.9. In the following, a user guide for the Process Event is provided:

1. Select the captured images (provided that the metadata for these images already exists) and add those to the input by the "Input >>" button
2. Select the range map zip file and the textured mesh zip file and add them to the output by the "Output >>" button
3. Get a new "Process Event" form via the menu: New → Event → Process Event

4. Fill (enter or select) the following form fields (some fields are already prefilled):
 - On the first tab: Metadata (Figure 10)
 - Event Title: The event title, according to the rules
 - Type: An event type, what was done at this processing
 - Project Event: The (super) project event
 - Person: The person who has done the processing (optional)
 - Responsible Organization: The responsible organization for the processing
 - Software: The used processing software
 - Software Parameters: The used software parameters (optional)
 - Parameter File: A parameter file (optional)
 - Location: The location where the processing was carried out
 - Starting Date/Time: The start date and time of the processing
 - Ending Date/Time: The end date and time of the processing (optional)
 - Process Notes: Some notes / additional info (optional)
 - On the second tab: Input (Figure 11)
 - Digital Input Object: The input object(s) for the processing
 - On the last tab: Output (Figure 12)
 - Digital Object / Multipart Digital Object: Choose if the captured object is a normal digital object (one file) or a multipart digital object (several files which form one logical object)
 - For a digital object:
 - * File Name: The file name of the binary (normally preselected)
 - * Object Type: The object type of the binary (at least one)
 - * Mime/File Type: The mime type of the binary (normally preselected)
 - For a multipart digital object:
 - * Label/Name: The label/name of the object
 - * Object Type: The object type (at least multipart digital object)
 - * Part Digital Objects: The single parts of the multipart object
 - Object Notes: Some object notes (optional)
5. Save the input

```

1 <Form Name="ProcessEvent"
2   Title="Process Event"
3   Group="ProcessEvents"
4   SelectKey="EventTitle"
5   InputNeeded="True"
6   OutputNeeded="True">
7   <Tab Group="ProcessEventTabs" Header="Metadata">
8     <TextField Name="EventTitle" KeyText="Event Title"/>
9     <MultiComboField Name="Type" KeyText="Type"
10      FormName="ProcessType"/>
11     <ComboField Name="ProjectEvent" KeyText="Project Event"
12      FormName="ProjectEvent"/>
13     <MultiComboField Name="Person" KeyText="Person*"
14      FormName="Person"/>
15     <ComboField Name="ResponsibleOrganization"
16      KeyText="Responsible Organization"
17      FormName="LegalBody"/>
18     <MultiSubForm Name="SoftwareWithParameter"
19      Header="Software"
20      FormName="SoftwareWithParameter"/>
21     <ComboField Name="Location" KeyText="Location"
22      FormName="Location"/>
23     <DateTimeField Name="StartDateTime"
24      KeyText="Starting Date/Time"/>
25     <DateTimeField Name="EndDateTime"
26      KeyText="Ending Date/Time*"/>
27     <NoteField Name="ProcessNote" KeyText="Process Note*"/>
28   </Tab>
29   <Tab Group="ProcessEventTabs" Header="Input">
30     <MultiComboField Name="InputDigitalObject"
31      KeyText="Digital Input Object"
32      FormName="DigitalObject"
33      StorePath="true"/>
34   </Tab>
35   <Tab Group="ProcessEventTabs" Header="Output">
36     <MultiSubForm Name="OutputDigitalObject"
37      Header="Digital Output Object"
38      FormName="DigitalObject" />
39   </Tab>
40 </Form>

```

Listing 6.6: The Process Event form described in Forms Definition Format.

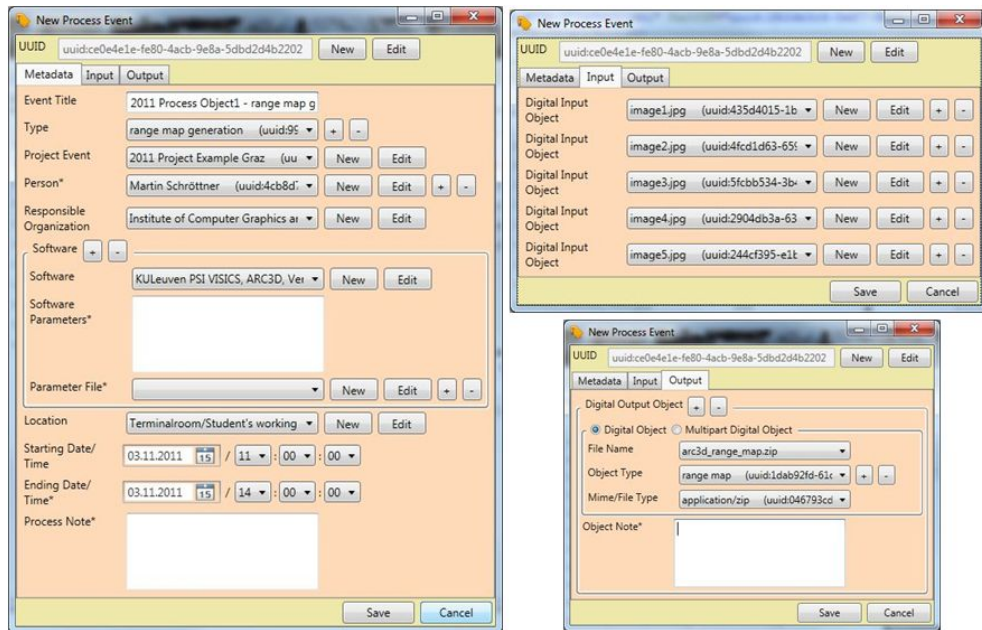


Figure 6.9: This figure shows the Process Event form generated at runtime by using the Forms Definition. In this example the captured images are processed by Arc3d to get range maps.

6.6 Ingest Script Generation

An additional service of the MG is the opportunity to generate ingest scripts for the *riclient* [36], a command line tool for clients to communicate with the RI. For each project in the scope of the current selected directory, a script can be generated. This script is used by the *riclient* tool to upload the whole project with all binary data, metadata, etc. to the RI.

After RDF files have been created, it is possible to generate an ingest script via menu: Project > the name of the project > Create Ingest Script. The ingest script is a windows batch script and will be stored in the ingest directory of the MG located inside the user's "Documents" in the "MetadataGenerator" directory. The Ingest Script Config requests the user to insert specific information which will be stored only inside the script file (Figure 6.10).

In the following, a user guide for the Ingest Script Config is provided:

- Script File Name: The filename of the script (without extension), a suggestion will be given
- Ingest To: The URL of the RI-server, where the project has to be ingested. It is possible to select the RI-Default for final versions, RI-Debug for test, or last used URL.
- Location Name: The name of the RI location where the binaries should be stored
- Username: The username of the ingest user with the required permissions
- Password: The user password (will be stored in plaintext in the script file!)
- Ingest Group UUID: The UUID of the RI group within which all data will be stored, must be an already existing group, e.g. the home group.
- Home Usergroup Name: The name of the user group to get full permission for the data (optional)
- Coform Usergroup Name: The name of the user group to get only download permission for the data (optional)

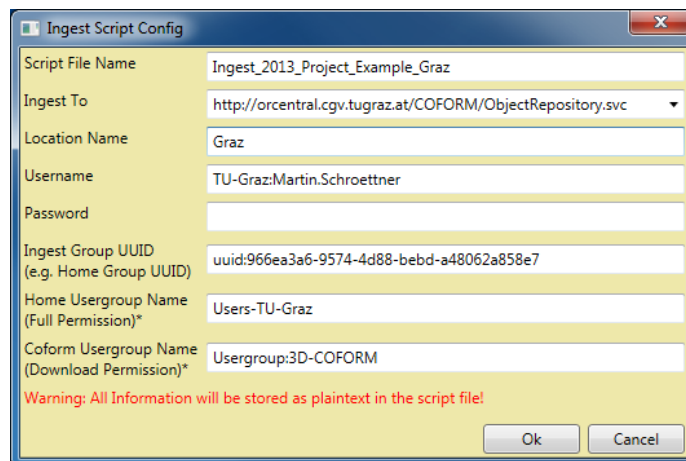


Figure 6.10: This figure shows the Ingest Script Config Window showing opportunities for adjusting to script generation.

6.7 Implementation Details

For the implementation of the MG, many design decisions for specific solutions had to be made. In the following, constraints and justifications of some main decisions and solutions are explained.

Development Environment and Used Framework

The 3D-COFORM RI uses Microsoft Windows 2008 R2, MS SQL Server 2008 R2 for the OR-Central Server and the Services are implemented with C# by using ASP.NET 4.0 framework. Therefore, it was suitable to implement the MG also with C# by using .NET framework 4.0 and additionally, the MS WPF framework was applied for the GUI. For this, it was obvious to use MS Visual Studio 2010 as a development environment.

Software Architecture

The software architecture of the MG is based on the three-tier architecture (see Figure 6.11). This architecture consists of the presentation tier, the processing or control tier, and the storage tier. The presentation tier of the MG is the GUI including the forms, which are used to get data from the user. The FormsManager which acts as form building and data controller, is part of the Control / Processing tier. A further part of this tier is the Transform. Initiated by the user, it converts stored metadata from the Intermediate Storage Format into a standard output format. Finally, the Storage tier manages the storing of the metadata and provides a reuse in forms controlled by the FormsManager.

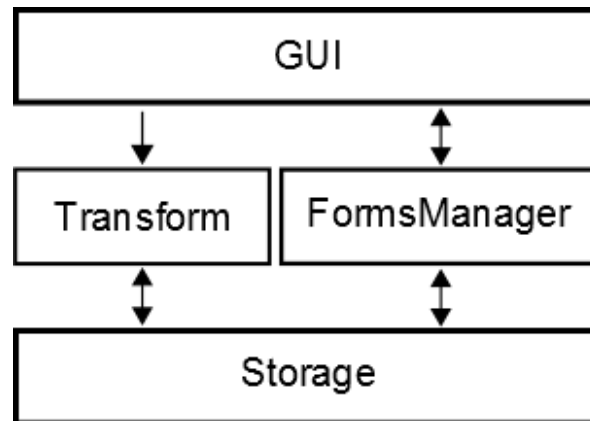


Figure 6.11: This figure shows the three-tier software architecture of the MetadataGenerator. (a) The GUI including the forms is the presentation tier. (b) The Transform and the FormsManager represent the Control / Processing tier. (c) The Storage tier managing storing of the metadata.

Dynamic Generated Forms

The used frameworks provide many predefined support classes, e.g., for XML-file or Windows GUI building. Thus, the form field development with WPF was a straight forward solution. Each form field including its behaviour at user interaction can be easily defined. The challenge was the dynamic arrangement of fields to obtain forms by using the XML-based Forms Definition Format. It is to consider that some fields (e.g., the MultiTextField) can be duplicated or removed by the user at runtime. A solution was found by using a unique ID (UUID) for each item: form, form field, and group of form fields. The ID will be registered with the appropriate field name (must be unique per form) by a suitable dictionary. While saving the data of a form, the field name sequence defined in Forms Definition Format is used for reading the data of the dictionary and for storing it into an Intermediate Storage Format.

7 Testing and Results

In the context of the 3D-COFORM project, different test campaigns, e.g. with museums hosting many exhibits, were realized. This chapter shows some example (test) campaigns and statistics about their results. The data was generated by in total 16 trainees over a period of 4 weeks per person.

7.1 Test Campaigns

Many (test) campaigns were successfully realized. Basically, each (test) campaign consists of 5 main steps:

1. Take photos from exhibits
2. Generate 3D-model from photos
3. Post Processing of the 3D-data
4. Generate metadata and paradata
5. Ingest all data into 3D-COFORM RI

In the first step, photos from each exhibit are taken in a suitable illuminated surrounding with the help of a turntable. With these pictures from all sides of the object, a 3D-model is generated with the web-based photogrammetry tool Arc3d (see Section 3.1.1). Once the 3D-model of the exhibit is available, it can be post processed, e.g., mesh cleaning, sub sampling and/or aligning to the axis. Furthermore, for all the previous steps and data results, the associated metadata and paradata are generated with the MG. Finally, all data (images, 3D-data, metadata, etc.) are ingested to the 3D-COFORM Repository Infrastructure (RI) with an MG generated ingest script generated by the MG and the riclient tool.

Archaeology Museum Schloss Eggenberg

A cooperation between the Archaeology Museum Schloss Eggenberg and the Institute of Computer Graphics and Knowledge Visualization (CGV) at Graz University of Technology was established with the aim to test technologies for 3D digitization of museum exhibits. This campaign with the Archaeology

Museum Schloss Eggenberg which is a museum department of Universalmuseum Joanneum [47] was of major importance to the development of the MG by providing the main part of the test data for the implementation. The aim of this campaign was the testing of the processing chain for digitization of several museum exhibits, including the storing in the RI and the metadata generation. Therefore, a suitable image acquisition environment with diffuse illumination for taking turntable photos was built at the museum (see Figure 7.1). Afterwards, all post processing steps were applied at CGV.



Figure 7.1: Archaeology Museum Schloss Eggenberg: The image acquisition environment with diffuse illumination for taking photos. (Image source [12])

Gipsmuseum

This campaign was a cooperation between *Gipsmuseum* of the Institute of Archeology of Karl-Franzens University (KFU) and CGV. Primarily, the Gipsmuseum is exhibiting 1:1 plaster copies of ancient Greek and Roman statues. The steps of digitization of 24 selected statues are documented in [12]. As this campaign was one of the first test campaigns to test image acquisition and generation of 3D-models, the final metadata was generated after finishing the implementation of the RI and the MG.

7.2 MG Use Case Description

The MG's main task as part of each test campaign is the semi-automated generation of metadata and paradata. Principally, metadata is required for:

- Exhibits
- Photos
- 3D-models
- Processing steps

The metadata about the exhibits are data from the real object, e.g., material, current keeper, inventory number. Metadata from photos include the used camera type, the photographer's name, etc. Important metadata from 3D-models are the used creation software and parameters. Finally, the documentation of all processing steps from taking the photos to the processing of the 3D-model mesh have to be included.

The generated metadata has to conform to the 3D-COFORM metadata scheme and the results are encoded in RDF. As additional service, the MG should generate an ingest script for the recipient tool, which uploads all data and metadata to the RI.

7.3 MG Results

For the MG (test) campaigns in the context of the 3D-COFORM project, about 50,000 RDF files for about 2,100 digital assets were generated. Some example 3D-models, results of digitization are shown in Figure 7.2. We have improved the MG based on the experiences and feedback of 16 trainees who were using the MG successfully over 4 weeks. Each test user took part in a one-week training session to become acquainted with the tool and to gain experience. The more experience a user has, the faster the process of metadata generation is.

With the current MG version, the generation of 3D-COFORM metadata based in CIDOC-CRM and CRMdig takes less than 30 minutes for the acquisition of a real object consisting of several sequences (1 to 20 sequences) of about 20 to 60 images each (400 photos in total). Whereas, metadata generation for the first image which includes the entering of the data needed by the event chain (capture event, sequence event, acquisition event) takes about 60 percent of the required time. Further images can be easily added to a sequence and so the already entered metadata will be automatically reused.



Figure 7.2: Example digitization results, 3D-Models of exhibits from Archaeology Museum Schloss Eggenberg. First, the museums exhibits were digitized with ARC3D using several turntable photos, then the raw 3D-model had to be post processed in order to obtain the results visible in this figure.

8 Conclusion

Preserving our cultural heritage does not signify keeping the ashes, but passing on the embers. (Based on a proverb by Confucius)

This thesis explains and emphasizes the relevance of digitization for the preservation of cultural heritage and gives an overview of various techniques to obtain 3D-data from real artifacts. Furthermore, metadata (“data about data”) are shown to be important for searching and retrieval of digitized objects. A discussion of different opportunities for generating cultural heritage metadata leads to a generic solution approach (see Chapter 5). This thesis presents the details and background of this approach, which was previously published in Schröttner et al. [39].

The MetadataGenerator (MG) (see Chapter 6) is an implementation example of the generic solution approach, which is a framework for generating metadata. The tool was developed and used in the context of 3D-COFORM for mass generation of cultural heritage metadata related to 3D-assets. It generates CIDOC-CRM encoded RDF files and provides dynamically generated forms for entering the requested metadata. Furthermore, it is designed to help the user with automated pre-filling of as many form fields as possible and enables updating of previously entered data. The MG is capable of finding new automation possibilities. The program has been used and tested by several users, first and foremost by trainees of the Institute of Computer Graphics and Knowledge Visualization at Graz University of Technology. Many useful user suggestions have helped to improve MG.

8.1 Contribution and Benefit

This master’s thesis faces the challenge of metadata and paradata generation with the focus on cultural heritage 3D-assets. It presents a generic approach which helps to avoid problems at metadata mass generation. The framework reduces the amount of needed human interaction by providing generic forms for input and generic formats for form definition, intermediate storage and template definition. The forms offer opportunities such as the pre-filling

of automatically detectable information and the reusing of already existing information with the help of unique links. Additionally, the directory structure is used to speed up the input process. Finally, everything is stored in a standardized format or metadata scheme. The approach scales well and eases the metadata generation. Thus, the problems arising at digitization of considerable amounts of cultural heritage objects will be reduced and this additionally leads to a decrease in the cost.

Furthermore, the standardized output information is usable to build a semantic network which enables searching and retrieval of 3D-data and cultural heritage facts. The possibility of finding unknown semantic connections is provided. As a result, doors for cultural heritage research are opened. Everyone can benefit from an easier access as well as from the appropriate presentation of cultural heritage assets and their corresponding metadata.

8.2 Future Work

During metadata generation, the best solution includes creating information as automatically as possible in order to reduce or avoid manpower and costs. Metadata for a specific domain, such as cultural heritage, can be generated and stored using various schemes. For finding the best option, a distinction by the potential degree of automation would be useful. This can be obtained by calculating a measurement value for the automation level of a scheme. A first solution might be a value expressing the relative amount of fields which can be filled automatically. If the importance and filling rate of the fields are included, the value will become even more meaningful. Considering the aforementioned results, the potentially best domain dependent scheme could be found, applied and the degree of automation maximized to reduce costs.

Further improvements can be made concerning the MG by increasing the degree of automated pre-filling of forms. This can be achievable by developing specialized modules for specific pre-filling tasks. Moreover, a GUI for editing and creating of forms and templates would improve the work with the MG. Therefore, a kind of GUI based creation of the Forms Definition Format and Template Definition Format would be useful. Developing such and other features can advance the MG, increase its usability and, regarding the economic aspect, make the MG a very cost-efficient tool.

Bibliography

- [1] Autodesk. *Maya. Overview*. URL: <http://www.autodesk.com/products/autodesk-maya/overview> (visited on 06/14/2013).
- [2] Matthias Bauer, Ronald Maier, and Stefan Thalmann. "Metadata generation for learning objects: an experimental comparison of automatic and collaborative solutions." In: *E-Learning 2010* (2010), pp. 181–195.
- [3] David Bearman and Jennifer Trant. "Social Terminology Enhancement through Vernacular Engagement: Exploring Collaborative Annotation to Encourage Interaction with Museum Collections." In: *D-Lib Magazine* 11 (2005). URL: <http://www.dlib.org/dlib/september05/bearman/09bearman.html>.
- [4] Nick Crofts, Martin Doerr, Tony Gill, Stephen Stead, and Matthew Stiff. *Definition of the CIDOC Conceptual Reference Model*. ICOM/CIDOC CRM Special Interest Group, Oct. 2006. URL: <http://cidoc.ics.forth.gr>.
- [5] Martin Doerr. *Mapping of the Dublin Core metadata element set to the CIDOC CRM*. Tech. rep. ICS/FORTH, 2000.
- [6] Martin Doerr. *The CIDOC CRM, a Standard for the Integration of Cultural Information*. Powerpoint Presentation. Nuremberg, 2005. URL: http://cidoc.ics.forth.gr/docs/crm_for_nurnmberg.ppt (visited on 06/17/2013).
- [7] Martin Doerr and Maria Theodoridou. "CRMDig: A Generic Digital Provenance Model for Scientific Observation." In: 2011.
- [8] Martin Doerr, Katerina Tzompanaki K.ompanaki, Maria Theodoridou, Christos Georgis, A. Axaridou, and Sven Havemann. "A Repository for 3D Model Production and Interpretation in Culture and Beyond." In: *VAST*. Ed. by Alessandro Artusi, Morwena Joly, Genevieve Lucet, Denis Pitzalis, and Alejandro Ribs. Eurographics Association, 2010, pp. 97–104. ISBN: 978-3-905674-29-3.
- [9] Dublin Core Metadata Initiative (DCMI). *Dublin Core*. URL: <http://dublincore.org> (visited on 05/07/2013).
- [10] European Union. *3D-Coform*. URL: <http://www.3dcoform.eu> (visited on 01/16/2013).

Bibliography

- [11] Europeana Foundation. *Europeana*. URL: <http://www.europeana.eu/> (visited on 06/03/2013).
- [12] Dieter W. Fellner, Sven Havemann, Philipp Beckmann, and Xueming Pan. "Practical 3D reconstruction of cultural heritage artefacts from photographs-potentials and issues." In: *Virtual Archaeology Review 2.4* (2011), pp. 95–103.
- [13] Foundation for Research and Technology, Hellas (FORTH). *3D-COFORM - Ingestion Tool*. URL: <http://www.ics.forth.gr/is1/3D-COFORM/> (visited on 05/12/2013).
- [14] FrankfurterAllgemeine. *Gestohlene Kunstwerke wahrscheinlich verbrannt*. July 18, 2013. URL: <http://www.faz.net/aktuell/feuilleton/rotterdammer-kunstraub-gestohlene-kunstwerke-wahrscheinlich-verbrannt-12286609.html> (visited on 07/19/2013).
- [15] Christos Georgis and Ioannis Chrysakis. *Ingestion Tool - User Manual*. Version 1.5.0. Foundation for Research and Technology, Hellas (FORTH). URL: <http://www.ics.forth.gr/is1/3D-COFORM/Manual.pdf> (visited on 06/15/2013).
- [16] Google. *reCAPTCHA*. URL: <http://www.google.com/recaptcha> (visited on 05/12/2013).
- [17] Sven Havemann. "Generative Mesh Modeling." PhD thesis. Institute of Computer Graphics, Faculty of Computer Science, Braunschweig Technical University, Germany, 2005. URL: www.digibib.tu-bs.de/?docid=00000008.
- [18] Sven Havemann, Volker Settgast, Harald Krottmaier, and Dieter W. Fellner. "On the Integration of 3D Models into Digital Cultural Heritage Libraries." In: *Proc. VAST 2006 Intl. Symp. Eurographics*, 2006, pp. 161–169.
- [19] International Organization for Standardization (ISO). *ISO 15836:2009. Information and documentation – The Dublin Core metadata element set*. 2009. URL: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=52142 (visited on 05/02/2013).
- [20] International Organization for Standardization (ISO). *ISO 16684-1:2012. Graphic technology – Extensible metadata platform (XMP) specification – Part 1: Data model, serialization and core properties*. 2012. URL: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=57421 (visited on 07/02/2013).

- [21] International Organization for Standardization (ISO). *ISO 21127:2006. Information and documentation – A reference ontology for the interchange of cultural heritage information*. 2006. URL: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=34424 (visited on 03/15/2013).
- [22] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera." In: *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM. 2011, pp. 559–568.
- [23] Brigitte Krenn, Gregor Sieber, and Hans Petschar. "Metadata Generation for Cultural Heritage: Creative Histories-The Josefsplatz Experience." In: *Proceedings of EVA (Electronic Information, the Visual Arts and Beyond)*. Vienna, Austria, 2006, pp. 27–34.
- [24] Learning Technology Standards Committee of the IEEE. *Draft standard for learning technology - Learning Object Metadata*. Tech. rep. New York: IEEE Standards Department, July 15, 2002. URL: http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf (visited on 07/10/2013).
- [25] London Charter Initiative. *The London Charter*. Jan. 2006. URL: <http://www.londoncharter.org> (visited on 01/16/2012).
- [26] Microsoft Developer Network (MSDN) Online. *Windows Presentation Foundation (WPF)*. URL: <http://msdn.microsoft.com/en-us/library/ms754130.aspx> (visited on 07/02/2013).
- [27] NISO. *Understanding metadata*. National Information Standards Organization. 2004. ISBN: 1-880124-62-9. URL: <http://www.niso.org/standards/resources/UnderstandingMetadata.pdf>.
- [28] Johan Oomen and Lora Aroyo. "Crowdsourcing in the cultural heritage domain: opportunities and challenges." In: *Proceedings of the 5th International Conference on Communities and Technologies*. ACM. 2011, pp. 138–149.
- [29] Xueming Pan, Philipp Beckmann, Sven Havemann, Katerina Tzompanaki, Martin Doerr, and Dieter W. Fellner. "A Distributed Object Repository for Cultural Heritage." In: *VAST'10 Proceedings of the 11th International conference on Virtual Reality, Archaeology and Cultural Heritage*. Ed. by Alessandro Artusi, Morwena Joly, Genevieve Lucet, Denis Pitzalis, and Alejandro Ribes. Eurographics Association, 2010, pp. 105–114. ISBN: 978-3-905674-29-3. DOI: 10.2312/VAST/VAST10/105-114.

- [30] Xueming Pan, Thomas Schiffer, Martin Schröttner, René Berndt, Martin Hecher, Sven Havemann, and Dieter W. Fellner. "An Enhanced Distributed Repository for Working with 3D Assets in Cultural Heritage." EN. In: *Progress in Cultural Heritage Preservation. 4th International Conference, EuroMed 2012, Limassol, Cyprus, October 29 – November 3, 2012. Proceedings*. Ed. by Marinos Ioannides, Dieter Fritsch, Johanna Leissner, Rob Davies, and Fabio Remondino. Vol. LNCS. Lemesos: Springer, Oct. 2012, pp. 349–358. ISBN: 978-3-642-34233-2. DOI: 10.1007/978-3-642-34234-9_35.
- [31] Xueming Pan, Thomas Schiffer, Martin Schröttner, Sven Havemann, Martin Hecher, René Berndt, and Dieter W. Fellner. "A Scalable Repository Infrastructure for CH Digital Object Management." EN. In: *IEEE-EXplore digital library*. Milano: IEEE, Sept. 2012, pp. 219–226. ISBN: 978-1-4673-2564-6. DOI: 10.1109/VSMM.2012.6365928.
- [32] Xueming Pan, Martin Schröttner, Sven Havemann, Thomas Schiffer, René Berndt, Martin Hecher, and Dieter W. Fellner. "A Repository Infrastructure for Working with 3D Assets in Cultural Heritage." In: *International Journal of Heritage in the Digital Era* (Volume 2, Number 1 / March 2013 2013), pp. 143–166. ISSN: 2047-4970 (Print). DOI: 10.1260/2047-4970.2.1.143.
- [33] Yoav IH Parish and Pascal Müller. "Procedural Modeling of Cities." In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM. 2001, pp. 301–308.
- [34] PROFACTOR GmbH. *ReconstructMe - Homepage. Digitize Your World*. URL: <http://reconstructme.net/> (visited on 08/05/2013).
- [35] Paramjeet Singh Saini, Marco Ronchetti, and Diego Sona. "Automatic generation of metadata for learning objects." In: *Advanced Learning Technologies, 2006. Sixth International Conference on*. IEEE. 2006, pp. 275–279.
- [36] Thomas Schiffer. *riclient Documentation*. Version 2.3. 2012.
- [37] Christoph Schinko, Martin Strobl, Torsten Ullrich, and Dieter W. Fellner. "Scripting Technology for Generative Modeling." In: *International Journal on Advances in Software* 4.3-4 (2011), pp. 308–326.
- [38] Martin Schröttner. *MetadataGenerator - Manual*. Version 1.1. 2012.
- [39] Martin Schröttner, Sven Havemann, Maria Theodoridou, Martin Doerr, and Dieter W. Fellner. "A Generic Approach for Generating Cultural Heritage Metadata." EN. In: *Progress in Cultural Heritage Preservation. 4th International Conference, EuroMed 2012, Limassol, Cyprus, October 29 – November 3, 2012. Proceedings*. Ed. by Marinos Ioannides, Dieter Fritsch, Johanna Leissner, Rob Davies, and Fabio Remondino. Vol. LNCS. Leme-

- sos: Springer, Oct. 2012, pp. 231–240. ISBN: 978-3-642-34233-2. DOI: 10.1007/978-3-642-34234-9_23.
- [40] Marc Spaniol, Ralf Klamma, and Mathias Lux. “Imagesemantics: User-Generated Metadata, Content Based Retrieval & Beyond.” In: *Proceedings of I-MEDIA '07 and I-SEMANTICS '07*. 2007.
- [41] Stichting Blender Foundation. *Blender*. URL: <http://www.blender.org/> (visited on 06/14/2013).
- [42] Technical Standardization Committee on AV and IT Storage Systems and Equipment. *Exchangeable image file format for digital still cameras: Exif Version 2.2*. Tech. rep. JEITA CP-3451. Apr. 2002.
- [43] David Tingdahl and Luc Van Gool. “A Public System for Image Based 3D Model Generation.” In: *Lecture Notes in Computer Science*. Vol. 6930/2011. Computer Vision/Computer Graphics Collaboration Techniques 5th International Conference, MIRAGE 2011. SpringerLink, 2011, pp. 262–273.
- [44] Jennifer Trant and with the participants in the steve. museum project. “Exploring the potential for social tagging and folksonomy in art museums: Proof of concept.” In: *New Review of Hypermedia and Multimedia* 12.1 (2006), pp. 83–105. DOI: 10.1080/13614560600802940.
- [45] Jennifer Trant and Bruce Wyman. “Investigating social tagging and folksonomy in art museums with steve. museum.” In: *Proceedings of the WWW'06 Collaborative Web Tagging Workshop*. 2006.
- [46] T. Ullrich, V. Settgest, and R. Berndt. “Semantic Enrichment for 3D Documents Techniques and Open Problems.” In: *ELPUB 2010 - Publishing in the networked world: transforming the nature of communication*. 2010, pp. 79–88.
- [47] Universalmuseum Joanneum. *Homepage of Universalmuseum Joanneum*. May 12, 2013. URL: <http://www.museum-joanneum.at/> (visited on 01/16/2013).
- [48] Seth Van Hooland. “From spectator to annotator: possibilities offered by user-generated metadata for digital cultural heritage collections.” In: *Immaculate Catalogues: Taxonomy, Metadata and Resource Discovery in the 21st Century, Proceedings of CILIP Conference*. 2006.
- [49] Maarten Vergauwen and Luc Van Gool. “Web-based 3D Reconstruction Service.” In: *Mach. Vision Appl.* 17.6 (2006), pp. 411–426. ISSN: 0932-8092. DOI: <http://dx.doi.org/10.1007/s00138-006-0027-1>.
- [50] Victoria & Albert Museum. *Victoria & Albert Museum Homepage*. Apr. 25, 2013. URL: <http://www.vam.ac.uk> (visited on 03/16/2012).

- [51] VISICS group of KU Leuven. *ARC3D. Automatic Reconstruction Cloud*. Convert your images into 3D. URL: <http://www.arc3d.be/> (visited on 04/15/2013).
- [52] Visual Computing Lab - Istituto di Scienza e Tecnologie dell'Informazione (ISTI) - Institute of the National Research Council of Italy (CNR). *Meshlab*. URL: <http://meshlab.sourceforge.net/> (visited on 04/27/2013).
- [53] Vassilios Vlahakis, Nikolaos Ioannidis, John Karigiannis, Manolis Tsotros, Michael Gounaris, Didier Stricker, Tim Gleue, Patrick Daehne, and Luis Almeida. "Archeoguide: An augmented reality guide for archaeological sites." In: *Computer Graphics and Applications, IEEE* 22.5 (2002), pp. 52–60. DOI: 10.1109/MCG.2002.1028726.
- [54] W3C. *RDF*. URL: <http://www.w3.org/RDF/> (visited on 03/25/2013).
- [55] Wikipedia. *Uniform resource identifier (URI)*. URL: <http://en.wikipedia.org/wiki/URI> (visited on 05/12/2013).
- [56] Wikipedia. *Universally Unique Identifier (UUID)*. URL: <http://en.wikipedia.org/wiki/UUID> (visited on 05/12/2013).
- [57] Wikipedia. *'Waterloo Bridge, London' van Claude Monet (1901)*. URL: http://commons.wikimedia.org/wiki/File:Waterloo_Bridge_in_London.jpg (visited on 07/24/2013).
- [58] Wikipedia. *Wikipedia, The Free Encyclopedia*. URL: <http://en.wikipedia.org> (visited on 04/07/2013).
- [59] René Zmugg, Wolfgang Thaller, Martin Hecher, Thomas Schiffer, Sven Havemann, and Dieter W. Fellner. "Authoring Animated Interactive 3D Museum Exhibits using a Digital Repository." In: *VAST*. Ed. by David B. Arnold, Jaime Kaminski, Franco Niccolucci, and André Stork. Eurographics Association, 2012, pp. 73–80. ISBN: 978-3-905674-39-2.