universität wien

DISSERTATION / DOCTORAL THESIS

Titel der Dissertation /Title of the Doctoral Thesis

„Dynamic Cross-Model Multimedia Content Composition in Personal Area Networks"

verfasst von / submitted by

Belal Abu Naim

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Doktor der technischen Wissenschaften (Dr. techn.)

Wien, 2017/ Vienna 2017

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 786 881
Dissertationsgebiet lt. Studienblatt / field of study as it appears on the student record sheet:	Informatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Klas

Acknowledgement

Foremost, I would like to express my sincere gratitude to Prof. Dipl.-Ing. Dr. Wolfgang Klas for being my supervisor. Thank you very much for your continuous support of my doctoral study and research, for your patience, motivation, enthusiasm, and immense knowledge. Thank you for the long hours we spent in talking, working on my thesis, discussing ideas and interesting research topics and issues, and all your enormous efforts. Your guidance helped me in all the times of research and writing of this thesis. It is a pleasure to work with you and I enjoyed my time being your doctoral student. I could not have imagined having a better supervisor.

I would like to thank the colleagues at the faculty of Multimedia Information Systems at the University of Vienna for their support and motivation. Special thanks goes to Ms. Manuela Schena for her support and arranging the appointments.

I would like to thank my colleagues at the work: Dr. Dieter Meinhard and MMag. Manfred Hackl for their support, motivation, stimulating discussions, and valuable feedback. Special thanks goes as well to MMag. Manfred for correcting the German translation of the Abstract. Further thanks goes to many other colleagues and friends for their support during writing this thesis.

Finally, I would like to express my sincere gratitude to my parents, Ahmad and Qamar, for urging me and giving me the opportunity to study, for teaching me the value of the knowledge, and for their passion for the education. I would like to thank my sister and brother for motivating me and helping me from the early beginning, throughout, and till the end of the path of my doctoral study. Special thanks goes to my friend, life partner, and wife Dana, thank you very much for all the motivation, support, advice, and for encouraging and giving me the time to finish writing this thesis.

Abstract

The past few decades witnessed the evolution of the Information Technology that characterized our epoch as the Information Age. This evolution impacting both software and hardware inspired us to identify four main technology areas relevant to this thesis: smart devices, Semantic Web technologies, the Social-Network platforms, and networking technologies. These four technology areas have an impact on multimedia content authoring and sharing. In this thesis, we analyze this impact by examining related aspects to this topic considering these technology areas. Then, we present our own approach that incorporates these four technology areas and covers topics not yet covered by other approaches.

With the massive proliferation of smart devices, mobile computing has taken over a significant role, replacing in some cases desktop computing and motivating the existence of new services, platforms, and technologies. The ever-expanding role of Social-Network platforms in our lives has changed the notion of multimedia content sharing. Hence, multimedia content sharing can be perceived as the exchange of user generated content both at private and public levels by means of different platforms, devices, and technologies. Consequently, new requirements for multimedia content authoring and sharing emerged. In this thesis, we identify these new requirements and present our innovative approach called Dynamic Cross-Model Composition (DCMC). The DCMC approach aims at the automatic analysis, authoring, and sharing of augmented multimedia presentations driven by a user's detected Subject of Interest (SOI) in personal area networks (PAN). The DCMC approach introduces an enhanced multimedia content authoring process. This process builds on a generic process chain for multimedia content composition and adds additional phases. The phases include: PAN formation, Analysis, SOI inference, Enrichment, Selection, Composition, Transformation, and Presentation. The enhanced authoring process is supported by a flexible and generic service-oriented architecture enabling a SOI-driven rule-based authoring and sharing of multimedia content.

Zusammenfassung

Das aktuelle Zeitalter lässt sich auf Grund der Evolution der Informationstechnologie während der letzten Jahrzehnten als Informationszeitalter charakterisieren. Aus dieser Entwicklung heraus, die Software und Hardware gleichermaßen betrifft, lassen sich vier, für diese Arbeit relevante, technologische Bereiche unterscheiden: Smartphones, Semantische Web-Technologien, Sozial-Media-Plattformen und Netzwerktechnologien. Diese vier Bereiche haben signifikante Einfluss auf die Erstellung und das Teilen von Multimedia-Inhalten. In dieser Arbeit analysieren wir diesen Einfluss, indem wir angelagerte Aspekte des Themas mit Bezug auf diese Technologiebereiche untersuchen. Im Anschluss präsentieren wir unseren eigenen Ansatz, der diese vier Technologiebereiche einbezieht und Themen umfasst, die noch nicht von anderen Ansätzen abgedeckt sind.

Mit der massiven Verbreitung von intelligenten Mobilgeräten hat das Mobile-Computing eine zentrale Rolle übernommen und ersetzt in etlichen Fällen das Desktop-Computing. Dies hat zum Aufkommen neuer Dienste, Plattformen und Technologien geführt. Die ständig wachsende Bedeutung von Sozial-Media-Plattformen im Alltag hat den Begriff des Multimedia-Content-Sharing verändert. Multimedia-Content-Sharing kann nun als Austausch von benutzergenerierten Inhalten sowohl auf privater als auch auf öffentlicher Ebene mittels unterschiedlicher Plattformen, Geräte und Technologien verstanden werden. Dadurch sind neue Anforderungen an die Bereitstellung und das Teilen von Multimedia-Inhalten entstanden. In dieser Arbeit identifizieren wir diese neuen Anforderungen und präsentieren unseren innovativen Ansatz namens Dynamic Cross-Model Composition (DCMC). Der DCMC-Ansatz zielt auf die automatische Analyse, die Bereitstellung und das Teilen von augmentierten Multimedia-Inhalten, angestoßen in einem Personal Area Networks (PAN) über die ermittelten Subjects of Interest (SOI) eines Anwenders. Der DCMC-Ansatz stellt einen erweiterten Multimedia-Content-Bereitstellungsprozess dar. Dieser Prozess baut auf einer generischen Prozesskette für die Zusammensetzung von Multimediainhalten auf und fügt weitere Phasen hinzu. Die Phasen umfassen: PAN-Bildung, Analyse, SOI-Inferenz, Anreicherung, Selektion, Komposition, Transformation und Darstellung. Der erweiterte Bereitstellungsprozess wird durch eine flexible und generische serviceorientierte Architektur unterstützt, die eine SOI-gestützte, regelbasierte Erstellung und gemeinsame Nutzung von Multimedia-Inhalten ermöglicht.

Short Table of Contents

Contents

1. Introduction

In the era of smart devices, social network platforms, the Semantic Web technologies and applications, e.g., the Web of Data, and the networking technologies, the world around us is overwhelmed with a massive amount of digital resources. However, users are concerned only about those resources that could be of interest to them. Given that, on the one hand, locating and interlinking locally available resources with the resources provided by the Web of Data remain tedious tasks for non-technical users. First, because the Web of Data offers services targeting machine processing, second, locally available resources share implicit subjects, that cannot be uncovered by users without being assisted by computers. On the other hand, due to the predefined role of the user as a content consumer, current multimedia authoring solutions mainly consider topics such as authoring process efficiency and multimedia content delivery, which affected the nature of those provided solutions [Bulterman et al. 2013]. This context challenges applications, on the one hand to be capable to analyze users' content and detect one or more Subjects of Interest (SOI), to make use of the detected SOI to interlink the analyzed content with relevant multimedia content retrieved from the Web of Data, and to make use of the interlinked content to author new fancy multimedia documents. On the other hand, to handle multimedia content sharing following a new understanding to this topic, which can be realized by supporting multimedia content sharing using channels other than the social network platforms, e.g., personal area networks, and supporting sharing between devices as well as users.

Nowadays, users possess different types of personal area network (PAN)-enabled smart devices, which allows for the creation of multimedia content such as photos, videos and audios. Thanks to networking technologies, users can easily interconnect their smart devices, e.g., in a peer-to-peer (P2P) network, to make the captured multimedia content available on their home PAN. Consequently, users tend to generate and share noteworthy amounts of multimedia content by using their smartphones, in addition to other types of smart devices. Social network platforms such as Snapchat [Snap Inc. 2016], Twitter [Twitter Inc. 2016] and WhatsApp [WhatsApp Inc. 2016], etc. have millions of subscribed active users, capturing and sharing daily a vast amount of multimedia content. Such content can be interlinked, enriched, and used to author fancy multimedia presentations. The multimedia content shared by the users can be either in form of structured multimedia documents such as photo albums or baseline media elements such as images and videos. However, both forms of such shared content encode additional hidden information about the users'

Subject of Interest (SOI). The Web of Data represents a rich source for publicly made available huge amount of information. This information can be interlinked with the content produced by the user, however, those resources remain unreachable by users making use of traditional search and browsing methods. One reason is that the Web of Data offers a massive number of resources, which makes it inconvenient to humans to browse, search, or discover those resources by means of manual Semantic Web browsers. A second reason is that the Web of Data is based on standards for querying, processing, and presenting data by machines, not humans. Nevertheless, the information available by the Web of Data can be of high value to the user, provided that the relevant parts of the information can be discovered, interlinked, and presented by a dedicated multimedia content authoring process. To enable such targeted information discovery, a multimedia authoring process should provide methods to detect one or more SOI that can be used to guide the discovery process. The SOI should result from an analysis of the multimedia content generated by the user.

To our knowledge multimedia content authoring and sharing systems that: i) provide a focus on an analysis process placing users and their SOI in the center, ii) represent a new understanding for multimedia content sharing, and iii) make intensive use of Web of Data and Semantic Web standards to enable rule-based content analysis and SOI-driven multimedia document composition do not exist. Consequently, we argue in this thesis for the need of an innovative multimedia content authoring and sharing approach, taking into consideration the issues we pointed out. Our solution is designed following well-recognized principles of a service-oriented architecture (SOA) and is implemented by the generic framework for Dynamic Cross-Model Composition (DCMC) [Abu-Naim and Klas].

1.1 Motivation and goal of this work

Living in the Information Age gives us the opportunity to witness the worldwide evolution of the Information Technology, in its both software and hardware sides. This evolution in software and hardware inspired us to identify four main technology areas that motivate our work. These are the smart devices mainly the smartphones, the Semantic Web technologies and applications such as the Web of Data, the social network platforms, and the networking technologies.

Social network platforms and new generations of smartphones cause a change in the role of the users from being primarily multimedia content consumers to becoming multimedia content producers and consumers. The vast amount of daily generated and exchanged user's multimedia content using smartphones and social network platforms introduces new requirements and urges revisiting traditional solutions focusing on the user as an active producer and consumer for the multimedia content instead of being a (passive) consumer only. In addition to that, the ever-expanding number of the users of the social network platforms indicates the willingness of the users to share content and explore new options for content sharing. In the meanwhile, the new generations of smartphones enjoy advanced software as well as hardware capabilities, which call for and enable more advanced users' interaction and applications. The Semantic Web technologies define standards and methods that can be employed to build more intelligent and flexible systems. These systems can make use of technologies such as RDF and inferencing. Using RDF as a semantic representation scheme provides systems with a flexible yet well-defined data structure. Finally, the networking technologies open the door for new ways of interconnecting the smart devices making use of different communication channels and protocols. Hence, interconnecting these smart devices can go much further beyond the limits of the peer-to-peer communication by allowing different types of smart devices with different networking capabilities to form personal area networks to exchange all types of data, i.e., from raw sensor data to sophisticated digital objects. Putting these aspects together, which can be inherited from the four main technology areas, motivates our work and urges us to define new requirements for the applications of multimedia content authoring and sharing. Consequently, applications supporting such requirements call for an appropriate architecture, including the offering of generic, modular, scalable, distributed services of an authoring and sharing environment.

In DCMC, we reflect those identified requirements mainly by:

1. Reconsidering the flow of the authoring process by providing an analysis phase to detect user's SOI. These detected SOI are used to select and enrich related multimedia elements with discovered information from the Web of Data, which can be included in the authoring process.
2. Presenting a rule-based multimedia composition system that incorporates the Semantic Web standards intended to extract knowledge from a RDF-Graph, this rule-based composition allows for alternating and configuring the behavior of the authoring process and accordingly its results by adapting or providing new set of composition rules.

This thesis contributes concepts and solutions in the research areas of multimedia content analysis, authoring, and sharing, and requirements engineering and application architecture and design. In particular the contributions can be associated with the following research topics:

1. Content analysis:
    - Support the new role of users as content producers by providing means to analyze the multimedia content generated and shared by the users in order to detect their Subject of Interest.
2. Content authoring:
    - Enhance flexibility and dynamics of the authoring workflow and media composition including further incorporation of Semantic Web technologies.
    - Provide a framework that allows for a more flexible incorporation of the Web of Data allowing for more enriched authoring process and content.
3. Content Sharing:
    - Provide a framework that allows for different types of smart devices to form personal area networks to exchange data using different networking technologies.

4. Requirements engineering and applications architecture and design:
   - Provide a description of a comprehensive set of requirements derived from the motivating scenarios and the different phases of the multimedia content authoring process as defined by the DCMC approach.
   - Provide a Software Engineering approach following the notion of the generic programming to enable multiple implementations of the interfaces (services) of the framework. Such multiple implementations add flexibility and widen the range of analysis provided by our approach, allowing our approach to support several multimedia document formats, media element types, metadata schemes, and many other aspects.
   - Provide a mobile media player capable for playing back SMIL presentations. This media player is primarily provided to facilitate testing of our approach on mobile devices. Our mobile media player extends the Multimedia Messaging Service (MMS) application provided by the mobile platforms and intended to display the content of MMS.

1.2 Application Scenarios

In this section, we present three illustrative application scenarios in the domain of multimedia content authoring and sharing in personal area networks. Each application scenario has its individual focus regarding the issues of multimedia content authoring and sharing in personal area networks. By providing these three application scenarios, we aim at covering a wide range of applications in the domain of the subject of this work. These three application scenarios are:

1. Subject of Interest authoring in personal area networks. The first application scenario deals with multimedia content authoring and sharing in personal area networks driven by one or more detected SOI.
2. Personal multimedia content sharing in home networks. The second application scenario considers collecting user generated content (UGC) produced by different smart devices and automatic generation of multimedia media presentations incorporating these UGC and smart devices.
3. Analysis-driven authoring of multimedia content. The third application scenario deals with content analysis and authoring of augmented multimedia presentation.

A scenario serves describing a user's interaction with the system, hence, it focuses on the user's requirements, which are distinct from technical or business requirements. In the following scenarios, we concentrate on the tasks that the system supports considering the users and the context of use. Hence, each of the presented scenarios is characterized as follows: it is identified by a novelistic title, it incorporates one or more fictitious characters, it presents the context of using the system, and it describes the tasks to be fulfilled.

1.2.1 The mesh of the two colleagues' Subject of Interest

Mr. Baker is an Information Technology (IT) expert. Environmental IT solutions are his main domain of expertise. He is a photography hobbyist. In addition to his passion for urban photography, he usually loves to wander in nature to explore and take photographs of plants and wild animals. He uses an online photo album where he shares his photos with friends and other online users who have the same interest.

Mr. Thomas is a work colleague of Mr. Baker; he works as a network and system administrator. He is married and has two kids. He uses the social network platforms to share with friends his thoughts, personal experiences, and diaries. He is always interested in the content that other members of his friends' networks share.

As it is vacation time, Mr. Baker has planned to spend a few days near the Lake Neusiedl. In the nearby National park he can observe the birds, the landscape, the different species that live in the area and document his trip with his own notes and photographs. Mr. Thomas is travelling together with his family to Barcelona, they will spend one week visiting the main touristic points of interest in the city including the Zoo Park and the Barcelona Aquarium. There, they can download some multimedia content about the different species hosted.

Apparently, Mr. Baker and Mr. Thomas share the same Subject of Interest (SOI), and since both are socially active and usually tend to share multimedia content and life experiences on social network platforms, they also use a content sharing application to share multimedia content in personal area networks (PAN). This content sharing application allows for sharing and automatic authoring of augmented multimedia content derived by one or more SOI. As they share the office room, their smartphones can automatically establish a PAN and exchange data. The authoring process of content sharing application infers the SOI and as a part of the process the newly composed document can be enriched with content retrieved from the Web of Data, i.e., from DBpedia [DBpedia Org. 2017]. After the process completes, both colleagues will be notified about the new multimedia content. Each colleague can open the new document, and find an interesting multimedia content about their common SOI, e.g., "Mammals" retrieved from open data sources, accompanied with images they have taken and seen before.

1.2.2 An augmented multimedia evening on the couch

Mrs. Wagner works in business development for a Japanese high technology home appliances company. She is responsible for a market segment covering Central Europe and North Africa. Due to the nature of her job, Mrs. Wagner travels quite often to attend conferences and meet existing as well as new and potential customers. Usually, she spends the free time of her business trips in sightseeing. She prefers to participate in walking tours, where she explores new places and uses her GPS and Wi-Fi-equipped camera to make geo-tagged photos and videos. She keeps a personal digital-diary about her frequent trips. In this digital-diary she includes her own photos and videos, audio narrations with her voice describing those places, digital maps, and other multimedia content she gathered before and during a trip. At home she uses her smart-TV to display the diaries as multimedia presentations.

It is Monday early in the morning, Mrs. Wagner is travelling to Vienna for a one-day trip to meet one of her customers. After the meeting, Mrs. Wagner had few hours before her flight back home. She decided to walk to the metro-station, where she can take the airport-train. On her way to the metro-station, she did some sightseeing, photographing some sights using her camera and recording some audio narratives using her smartphone about these sights. Just before she arrived to the metro-station, she took a small break and enjoyed a cup of coffee in a nearby traditional Viennese café.

In the evening, she arrived home, she would like to get assisted in creating a digital-diary for her short tour in Vienna consisting of the photos and videos taken, information about the places she visited, properly combined with the audio narratives, and augmented with additional content retrieved from, e.g., web sources. And then, displaying the created digital-diary as a multimedia presentation on her smart-TV while she sits relaxed on her couch enjoying some snacks.

To assist Mrs. Wagner, one can imagine that, e.g., the photos and videos from her camera are streamed to her smartphone. Her smartphone analyzes the media elements to detect their locations based on annotated GPS data. Next, the smartphone links those media elements to the recorded audio narratives and collects additional information in form of multimedia content from external sources, e.g., using LOD [W3C Org. 2016a] servers. Finally, the smartphone composes the photos, videos, audio narratives, and the augmented multimedia content into a form of multimedia presentation and streams it to the smart-TV to be presented.

1.2.3 The great catch of the Web of Data

Mrs. Otoma is an architect, she works for a well-known architecture and design bureau in Zürich. She is interested in architectural work as well as interior design made by famous architects. She is occasionally invited by universities to give presentations about specific topics related to architectural design. Along with her presentations, she likes to share with her audience some additional material in form of multimedia presentations about some interesting constructions, buildings, or design concepts. She knows that the Web of Data offers a huge amount of publicly made available information, but since she is not specialized in IT, she faces difficulties reaching the information which the Web of Data offers.

Recently, she has received an invitation from a University in Dublin to make a presentation about postmodern design. Prior to the presentation in Dublin and during visiting an exhibition, she noticed an impressive master piece of furniture designed by a famous architect and thought that it would be nice to prepare a multimedia presentation about that architect, his work as well as similar works, which she can later share with the audience of her coming presentation in Dublin. Using some photos, which she has taken using her smartphone for that piece of furniture, she would like to collect information from the Web of Data about the architect, his designs, and similar works and include all the collected information in an automatically generated multimedia presentation ready to be shared with her audience.

To assist Mrs. Otoma, one can imagine that, e.g., the photos on her smartphone are analyzed to detect the objects that these photos include. Next, her smartphone collects information about these detected objects from the open sources provided by the Web of Data, e.g., the DBpedia [DBpedia Org. 2017]. Finally, her smartphone authors a multimedia presentation by making use of one of the pre-configured multimedia presentation templates included in the system. To enable sharing of the multimedia presentation later on, the smartphone generates a QR-code that she can include in her presentation. Interested audience can make use of that QR-code to download a copy of the presentation over an established personal area network connection between her smartphone and their smartphones.

1.3 Summary of contributions

The research in this work contributes to various fields: multimedia content analysis, authoring, sharing, requirements engineering and application architecture and design. The approach of a Dynamic Cross-Model Composition framework (DCMC) [Abu-Naim and Klas] developed and presented in the course of this thesis is built upon the contributions to those research areas. The DCMC approach provides an enhanced multimedia content authoring process supported by a flexible and generic framework for rule-based content composition driven by the Subject of Interest of users.

The enhanced multimedia content authoring process in DCMC follows the generic process chain of multimedia composition as presented in [Scherp and Boll 2005c]. DCMC extends that process and adds additional phases, which yielded an enhanced authoring process consisting of eight phases. This enhanced authoring process is supported by and makes use of Semantic Web technologies in its different phases. It is driven by the Subject of Interest (SOI) of users, which is inferred from content shared by various users and/or devices. By converting the multimedia source documents to RDF-Graph, DCMC allows for the enrichment of content by interlinking with sources from the Web of Data. Inference rules guide the selection of additional content elements relevant to the SOI. The authoring process results in newly composed and augmented multimedia content according to individual composition rules and/or pre-specified composition templates. The new content is finally transformed from RDF-Graph into the document format needed for distribution and presentation.

Towards identifying the requirements of DCMC, we followed the notion of User Stories as defined by the Agile Software Engineering methodology to express the requirements. This kind of requirements serve two purposes, (i) it targets an audience with non-technical background, and at the same time, (ii) it provides a reasonable technical level of specifications that enabled the design of the service-oriented architecture (SOA) of the DCMC. Thus, we identified two distinctive groups of requirements: requirements for personal area network establishment and data exchange and requirements for the multimedia content authoring process. The DCMC reflected those new requirements as follows: i) Providing a system architecture that explicitly separates between the networking and the multimedia content authoring functionalities. ii) Reconsidering the flow of the authoring process by providing an analysis phase to detect user's SOI. These detected SOI are used to select and enrich related multimedia elements with discovered information from the Web of Data, which can be included in the authoring process. iii) Presenting a rule-based multimedia composition system that incorporates the Semantic Web standards intended to extract knowledge from a RDF-Graph created during the authoring process. Using RDF as the internal, canonical, semantic representation scheme provides our system with a flexible yet well-defined data structure that enables the different parts of the system to flexibly add RDF data, which contributes to and enables the different phases of the authoring process. The rule-based composition allows for alternating and configuring the behavior of authoring process and accordingly its results by adapting or providing new set of composition rules.

Having identified the requirements and the authoring process, we defined the service-oriented architecture (SOA) of the DCMC, which consists of seven logically separated components implemented by the DCMC prototype. The Networking Framework enables the interconnection of multiple devices using different PAN technologies to form a PAN. The other frameworks and components supporting the multimedia content authoring allow for the realization of a flexible, extendible, and distributed authoring process of multimedia presentations. The SOA of DCMC allows to incorporate multiple PAN-enabled devices to author and present multimedia content with a focus on the users' Subject of Interest (SOI), enriched and infused with media content retrieved from LOD servers. In addition to that, modularity and generality of our SOA also allow for offering the functionality as part of a content sharing platform.

For testing purposes and to support the playback of SMIL presentations on mobile devices, we provided a mobile media player that is capable to playback SMIL presentations. This mobile media player extends the Multimedia Messaging Service (MMS) application provided by the Android platforms intended to display the content of MMS. The MMS application is part of the Android's open source software development kit (SDK).

1.4 Overview

The remainder of this work is organized as follows: subsequent to the introduction, the state of the art analysis is presented in chapter 2. Chapter 3 presents the DCMC approach. Chapter 4 provides the proof of concept. And finally, in chapter 5 we conclude this work.

In chapter 2, we present the underlying infrastructure and concepts for mobile computing and multimedia document authoring and sharing. We present relevant concepts and techniques for mobile networks and personal area networks, mobile operating systems and platforms, the relevant W3C standards to this work, and existing approaches and applications in the domain of multimedia content authoring and sharing. We present a set of criteria and in-depth analysis of the related research approaches based on these criteria.

In chapter 3, we present the requirements and the service-oriented architecture (SOA), implemented by the DCMC prototype, guided by the phases of the authoring process and the application scenarios.

In chapter 4, we present the underlying concepts, the specifications and the implementation of the different components and frameworks of the DCMC.

Figure 1-1 depicts the relationships between the analysis presented in chapter 2, the conceptual design presented in chapter 3, and the development of DCMC presented in chapter 4.

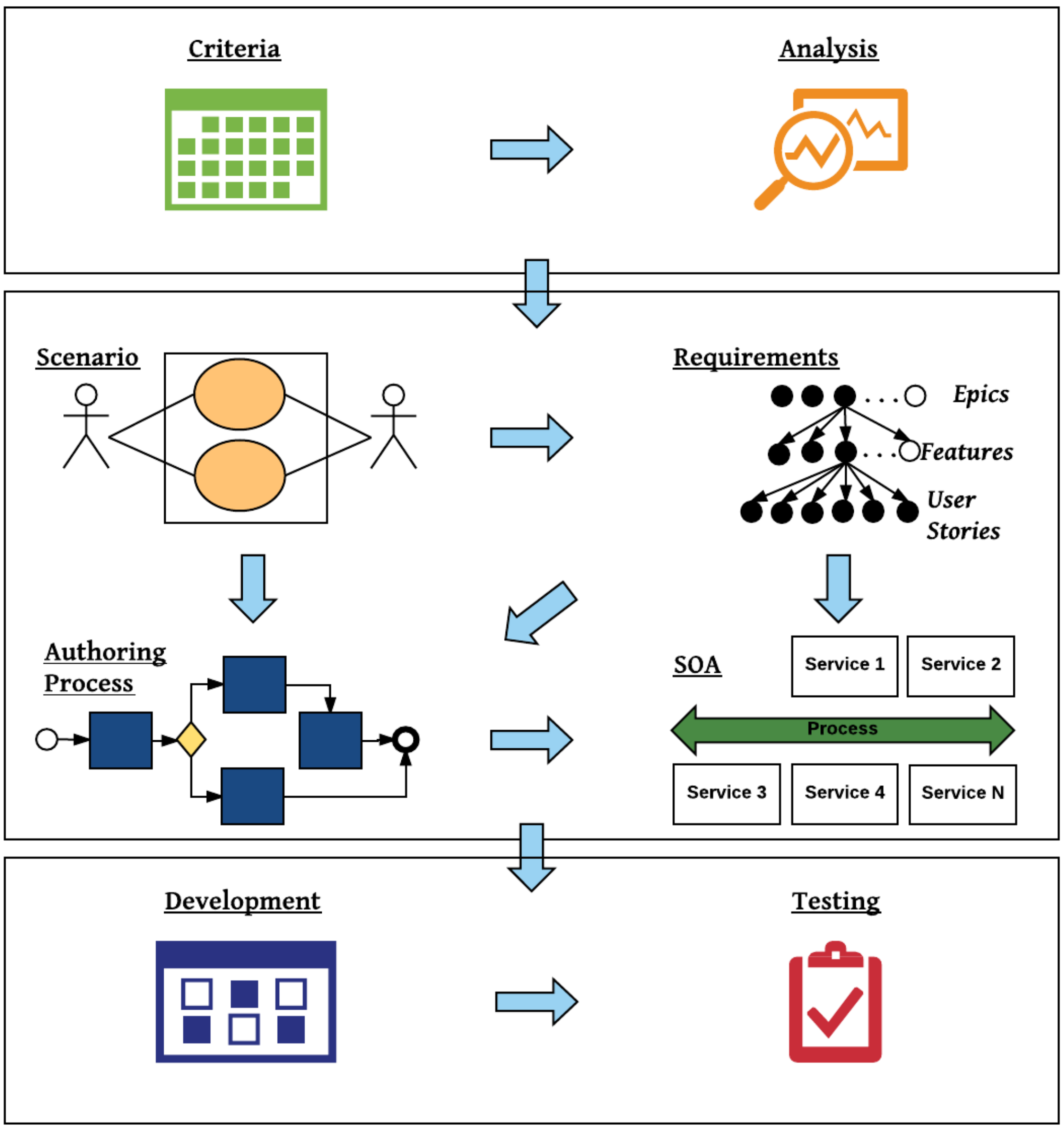


Figure 1-1: Analysis, Conceptual Design, and Application Development

The upper part of the figure depicts the relationship between the criteria and the analysis. The result of the analysis provides input to the conceptual design of the DCMC approach, which is depicted in the middle part of the figure. This middle part shows the relationships between the application scenarios, the authoring process, the requirements, and the SOA. The scenarios provide input to the requirements and the authoring process. The requirements specify the flow of the authoring process and the design of the service-oriented architecture. The authoring process influences the design of the services and the components of the system. The conceptual design provides the required input to the development and the testing of the system, which is depicted in the lower part of the figure.

2. State of the Art analysis

2.1 Introduction

Users of mobile devices heavily contribute to the massive increase of multimedia content production and consumption. The new generation of mobile devices known as "smartphones" offer their users many resources, almost similar to what desktop computers can offer. Those resources include high performance processors, large memory and hard disk capacities, high-resolution screens and cameras, GPS modules, sensors and more. Platforms and applications like social networks to some extent are responsible for massive content creation and consumption by a huge number of users. Mobile users represent a considerable number of those users. They rely on their mobile devices to maintain an always-on relation to information, personal, and social networks. Sharing daily life events and content of almost any type via social network platforms became very popular. W3C standards for the Semantic Web have contributed to the establishment of a worldwide infrastructure supporting many new applications and services processing and offering content. The Linked Open Data cloud is one famous example for such an infrastructure that may significantly enhance multimedia content production and consumption by users.

Capabilities of smartphones with respect to their display, memory, processing power, battery life, and hardware improved significantly. In the past, mobile devices were shipped with limited resources, which used to be considered as disadvantages and urged application developers to take every possible countermeasure to optimize their applications. However, with introduction of the recent generation of smartphones, those disadvantages vanished. The many fancy features of smartphones influence the behavior of the users and their interaction with their devices. Smartphones are used now in photography, video recording, web surfing, sending and receiving emails, as a personal organizer, a radio, a navigation system, a trip advisor, and in personal networking. One example of the advanced features offered by smartphones are Siri [Apple Inc. 2015b] and Cortana [Microsoft 2015a]. Siri and Cortana are applications that listen and deal with their users' requirements in more advanced and intelligent manner.

The World Wide Web (WWW) has evolved many years prior to the arrival of smartphones. WWW has gradually transitioned from being a means of sharing and browsing read only documents to a means of social networking and multimedia content sharing. A study [Ayaan Mohamud 2011] about the *Frequency of Use and Method of Access for Mobile Social Networking/Blog Audience* has provided numbers illustrating the massive increase of mobile access to social networks (see Table 2-1).

Well-defined Semantic Web standards such as URI and RDF enabled the establishment of the Linked Open Data services beginning in 2007. Wikipedia and its extracted dataset (DBpedia) is the most famous example. The DBpedia dataset describes about 38.3 million *things* in 125 languages [DBpedia Org. 2017]. The number of publicly available datasets in 2014 reached 1091 [LOD 2014] datasets compared to 295 datasets in 2011 [Anja Jentzsch, Richard Cyganiak,Chris Bizer 2011] achieving an increase of 271%.

Frequency of Use and Method of Access for Mobile Social Networking/Blog Audience
3 Month Avg. Ending September 2010 vs. September 2011
Total EU5 (DE, ES, FR, IT and UK) Mobile Subscribers Age 13+

	Total Audience (in thousands)		
	Sep-10	Sep-11	% Change
Accessed Social Networking Site or Blog Ever in Month	38,395	55,125	44%
Accessed Social Networking Site or Blog Almost Every Day	15,438	25,779	67%
Social Networking Access Method:			
Via Mobile Browser	23,855	31,307	31%
Via Application	12,057	24,208	101%

Table 2-1: The increasing number of mobile access to social networks

As of today, one can clearly state that users tend to join the social networks to share their daily live experiences in different media formats such as text, photo, or video. Smartphones have become the main means of sharing multimedia contents. However, multimedia content sharing is not limited to social network platforms. Thanks to networking technologies such as Bluetooth [Bluetooth SIG 2016] and NFC [NFC 2016], multimedia content can be shared in personal area networks (PAN), which can be formed by several but different types of mobile devices such as smartphones, smart-TV sets, and other PAN-enabled devices. Furthermore, the Linked Open Data cloud offers huge potential to ease the mash up of content from distributed information sources and, hence, allows for new forms of content production. Two questions are of particular interest in this context:

i) How do multimedia authoring and sharing approaches in general incorporate on the one hand the willingness of users to (freely) share multimedia content, and on the other hand the ever-increasing number of Linked Open Data sources?

ii) Do those approaches and applications consider multimedia authoring and sharing within personal area networks as well?

Trying to answer these questions motivates our research in this chapter, thus we need to look into the following topics: How are Semantic Web technologies, multimedia content produced by the users, and the content offered by Linked Open Data sources incorporated in the process of multimedia document authoring and sharing? We look into the various approaches and applications, the support of modern mobile operating systems for multimedia content authoring, sharing, and viewing. We provide a comprehensive, complete global picture on the state of the art in multimedia document authoring and sharing.

2.2 Background: Mobile Networks and computing

In this section we discuss different topics that influenced directly and indirectly the development of the process of multimedia content authoring and sharing in the context of mobile devices. To address this topic from the perspectives of both the IT-industry sector and the scientific community later on in this chapter. We discuss the contribution of the following parties in this challenging topic: i) The providers of the mobile network and communication protocols. ii) The producers of mobile devices and in particular the (smartphones). iii) The developers of the operating systems of mobile devices. The efforts of these parties have contributed remarkably to multimedia content authoring, sharing and viewing using the mobile devices.

2.2.1 Short history on the evolution of mobile networks

By tracing the evolution of mobile networks over the past three to four decades, we clearly recognize the influence of the increasing demand of the users on having more media content being consumed on mobile phones. With the introduction of mobile devices, the voice service was the first and only service that the analog cellular networks (1G) has offered. The change from analog to digital service has occurred in the digital cellular network (2G). New services like short message service (SMS) and data connections to the Internet have been introduced. 2G networks enabled the access to media content such as screen background images and ringtones on mobile phones. 2G networks has added another remarkable change which is the split that resulted in having on one hand the network provider, and the service provider on the other hand [Fitzek, Frank H. P., Reichert, Frank (Eds.) 2007]. This split has led to the existence of four main players in the mobile's industry, the network provider, the service provider, the device manufacturer and the customers [Fitzek, Frank H. P., Reichert, Frank (Eds.) 2007]. With the introduction of Mobile Broadband Data (3G) networks, more attention was paid to the services to satisfy the requirements of the customers. People started to use their mobile phones more and more in their daily life. Personal services and community services are the two main dominant types of services. This has led to more demand on accessing the data. The evolution of mobile networks and the emerging requirements of the customers have created an opportunity for the existence of different operating systems that target mobile phones, and for the manufacturers to produce SIM card enabled devices. Native IP networks (4G) was the result of the industries' efforts to cope with the increasing demand on bandwidth consuming applications such as media streaming application [Tanenbaum and Wetherall 2011]. These advances in the infrastructure of mobile networks have paved the road for many platforms and applications that enable their users to produce and consume a vast amount of multimedia content.

2.2.2 Personal Area Networks

Nowadays, users possess different types of personal area network (PAN)-enabled devices, which allows for the creation of multimedia content such as photos, videos and audios. Thanks to networking technologies, users can easily interconnect their devices, e.g., in a P2P network, to make the captured multimedia content available on their home personal area networks (PAN). A PAN is computer networks used to transfer data between different types of devices. Devices that support IEEE 802.15 [IEEE 802.15 - WPAN 2008] standards such as computers, smartphones, tablets, TVs, etc. can form a personal area network when they are placed within a distance of several meters of each other's. Different wireless technologies such as Wi-Fi [Wi-Fi 2017], Bluetooth [Bluetooth SIG 2016], or NFC [NFC 2016] can be used to establish a PAN. Based on the PAN profile [Bluetooth SIG - PAN Profile 2016], three connection scenarios are proposed (i) Network Access Point: in this scenario a Bluetooth-equipped device acts as a bridge or a router between a PAN and some other network technology e.g., GSM. (ii) Group Ad-hoc Network: in this scenario one Bluetooth-equipped device acts as a master which communicate between 1 and 7 Bluetooth-equipped devices operating as slaves. (iii) PAN User – PAN User: this PAN scenario realizes a direct peer-to-peer (P2P) communication between two Bluetooth-equipped devices. In the following, we will introduce the most applicable technologies and communication protocols for our work, namely Bluetooth (SPP and PAN protocols) and Wi-Fi.

2.2.2.1 Bluetooth

The Bluetooth Special Interest Group (SIG) [Bluetooth SIG - Core specifications 2017] has released several versions of the Bluetooth specifications, the most recent version (Bluetooth 5) was released on Dec 2016. Table 2-2 lists chronologically the adopted Bluetooth core specifications. The core specifications introduce different profiles that specifies how devices can exchange digital data using the Bluetooth technology. Those specifications include profiles for multimedia content exchange and for the formation of personal area networks (PAN). Table 2-3 encodes the Bluetooth power classes and their respective power output and ranges. Table 2-4 encodes the data throughput rates that applications can achieve by using the Bluetooth technology.

SPECIFICATION	ADOPTED ON	NOTES
CORE 5.0	06 Dec. 2016	50 Mbit/s
CORE 4.2	02 Dec. 2014	
CORE 4.1	03 Dec. 2013	
CORE VERSION 4.0	30 Jun. 2010	25 Mbit/s
CORE VERSION 3.0 + HS	02 Apr. 2010	25 Mbit/s
CORE VERSION 2.1 + EDR	26 Jul. 2007	

CORE VERSION 2.0+ EDR	04 Nov. 2005	3 Mbit/s
CORE VERSION 1.2	05 Nov. 2003	1 Mbit/s

Table 2-2: Bluetooth – history of adopted core specifications

POWER CLASS	MAX OUTPUT POWER	RANGE
CLASS 1	100 mW	~ 100 meters +
CLASS 2	2.5 mW	~ 10 meters
CLASS 3	1 mW	~ 1 meter
CLASS 4	0.5 mW	~ 0.5 meter

Table 2-3: Bluetooth – range of radio power classes in meters

VERSION	DATA RATE	MAXIMUM APPLICATION THROUGHPUT
VERSION 5.0	50 Mbit/s	
VERSION 4.0	24 Mbit/s	(Only with amplifier, and depends on the amplifier. BT itself remains 2.1 Mbit/s max)
VERSION 3.0 + HS	24 Mbit/s	(Only with amplifier, and depends on the amplifier. BT itself remains 2.1 Mbit/s max)
VERSION 2.0 + EDR	3 Mbit/s	2.1 Mbit/s
VERSION 1.2	1 Mbit/s	0.7 Mbit/s

Table 2-4: Bluetooth – the maximum data rate per Bluetooth versions

Bluetooth version 5 is released in Dec 2016 with boosted range, doubled speed, and increased broadcasting capacity by 800%. According to the Bluetooth SIG, the highlight of the new release of the Bluetooth technology is moving away from the app-paired-to-device model to a connectionless Internet of Things (IoT). [Bluetooth SIG - Core 5.0 2017].

Now that we have summarized the technical features of the different core specifications of the Bluetooth technology, we introduce in the following the relevant profiles that handle the exchange of multimedia content and the formation of PAN.

Basic Imaging Profile (BIP) defines how an imaging device can be remotely controlled, how an imaging device may print, and how an imaging device can transfer images to a storage device.

Advanced Audio Distribution Profile (A2DP) describes how stereo quality audio can be streamed from a media source to a sink.

Video Distribution Profile (VDP) defines how a Bluetooth-enabled device streams video over Bluetooth wireless technology.

Generic Audio/Video Distribution Profile (GAVDP) provides the basis for A2DP and VDP, which are the basis of the systems designed for distributing video and audio streams using Bluetooth technology.

Serial Port Profile (SPP) defines the protocols and procedures that shall be used by devices using Bluetooth for RS-232 (or similar) serial cable emulation. This profile is based on the RFCOMM protocol. It emulates a serial cable to provide a simple substitute for existing RS-232, including the familiar control signals. The scenario covered by this profile deals with legacy applications using Bluetooth as a cable replacement, through a virtual serial port abstraction. SPP is of a special interest to our work, since applications can exchange any type of digital content by streaming byte sequences over a virtual serial port. Figure 2-1 depicts the stack of the SPP.

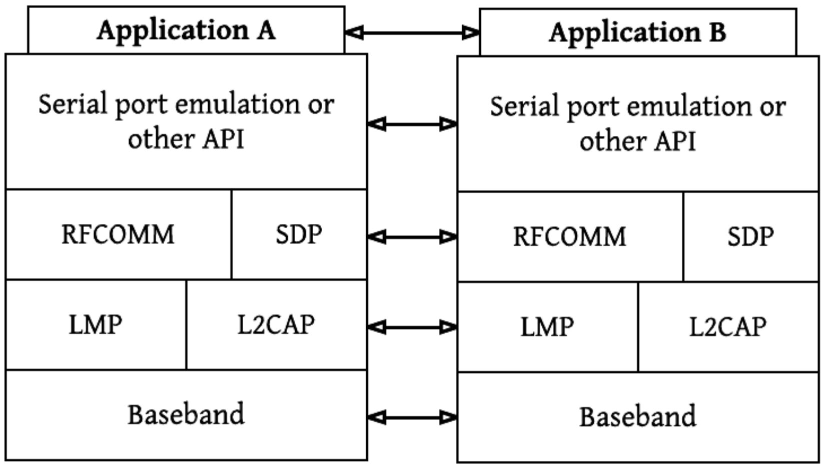


Figure 2-1: Bluetooth – the stack of Serial Port Protocol (SPP) [Bluetooth SIG - SPP Profile]

Personal Area Network profile (PAN) describes how two or more Bluetooth-enabled devices can form an ad-hoc network and how the same mechanism can be used to access a remote network through a network access point**.** PAN defines three roles:

(i) Network Access Point (NAP) and NAP Service – A Bluetooth device that supports the NAP service is a Bluetooth device that provides some of the features of an Ethernet bridge to support network services.

(ii) Group Ad-hoc Network (GN) and GN Service – A Bluetooth device that supports the GN service is able to forward Ethernet packets to each of the connected Bluetooth devices, the PAN users, as needed.

(iii) PAN User (PANU) and PANU Service - This is the Bluetooth device that uses either the NAP or the GN service.

PAN profile defines the following four Bluetooth connection scenarios:

i) *Personal*: link to one preset device.

ii) *Point to Point*: link to any one device (ad-hoc).

iii) *Point to multi-point*: link up to seven devices (Piconet).

iv) *Scatternet*: slave in one Piconet is the master of another Piconet.

The process of connecting devices via Bluetooth has three distinct steps. The flow of the process starts with *device discovery*, then *service discovery* and finally *selecting a service* to use.

Device discovery follows the inquiry protocol that has i) *Inquiry mode* to discover devices by sending inquiry packets. Short packets are sent out rapidly in a sequence of different frequencies. The inquiring device changes frequencies 3200 times a second. ii) *Scan mode* to make the device discoverable. The device changes frequencies very slowly, once every 1.28 seconds. This ensures that both *inquiry* and *scan* meet at some point on the same frequency. In *scan mode* frequency is changed regularly to avoid the interference.

The Connection protocol has i) *Page mode* which initiates the connection. To initiate a connection, the requesting device must know the 48-bit Bluetooth device address of the target device. ii) *Page scan mode* responds to incoming connection request. Table 2-5 summarizes the timing of the Inquiry and Paging modes.

OPERATION	MIN. TIME SEC.	AVERAGE TIME (SEC)	MAX. TIME (SEC)
INQUIRY	0.00125	3 - 5	10.24 - 30.72
PAGING	0.0025	1.28	2.56
TOTAL	0.00375	4.28 - 6.28	12.8 - 33.28

Table 2-5: Bluetooth - the timing of the Inquiry and Paging

Service Discovery Protocol (SDP) allows devices to discover the services provided by other devices within the PAN. Service discovery is performed after a data connection is set. Figure 2-2 depicts the State Machine as defined by Bluetooth specification. The state machine covers five states represented by Standby, Scanning, Initiating, Advertising, and Connection.

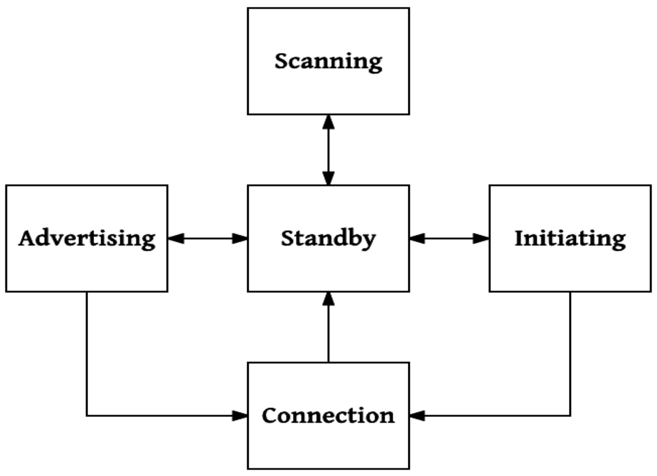


Figure 2-2: Bluetooth – the five states of the State Machine

2.2.2.2 Wi-Fi/IEEE 802.11

Wi-Fi is the trademark of the Wi-Fi alliance [Wi-Fi 2017]. Wi-Fi-enabled devices implement the standards defined by IEEE 802.11 [IEEE 802.11 WLAN 2017] for establishing wireless local area networks (WLAN). IEEE 802.11 is a set of media access control (MAC) and physical layer specifications for implementing WLAN computer communication in the 2.4, 3.6, and 5 GHz frequency bands. Table 2-6 encodes the main features of the IEEE 802 protocols.

802.11 PROTOCOL	FREQUENCY (GHZ)	DATA RATE (MBIT/S)	INDOOR RANGE (METER)
802.11-1997, B, G, N	2.4	~ 1 to 54	~ 20 to 125
A	3.6	~ 6 to 54	~ 35 to 115
A, N, AC	5	~ 6 to 86	~ 35 to 230

Table 2-6: Wi-Fi –summary of frequency, data rate, and indoor range as specified by the IEEE 802.11 protocols

2.2.3 Mobile Operating Systems and Platforms

Since the introduction of mobile phones about four decades ago, hardware evolution was accompanied by the same evolution of operating systems. Many operating systems were introduced. Some of them have survived and still maintained and in use by device manufacturers, some of them have not. However, each of those operating systems has its own approach in mobile computing and its own user experience. In this section, we introduce the most common operating systems that run on the mobile devices. This topic can be addressed from many perspectives. However, our ultimate aim is to highlight the main features of each operating system that support multimedia applications design and development.

2.2.3.1 Android

Several Linux-based platforms have been introduced for mobile devices; Maemo [Maemo OS 2007], Openmoko [Openmoko OS 2013], Qtopia [Qtopia OS 2006], Limo [LiMo Foundation 2017], and Android [Google 2017]. Many mobile manufacturers have adopted Linux-based mobile operating systems such as Samsung, Motorola, HTC, Sony and others.

Android is developed by Google and first introduced in November 2007 [Google 2017]. It is a Java-based operating system that runs on Linux kernel. Android applications are written using Java language. Applications do not run within a JavaME [Java ME 2017] virtual machine. Java-compiled classes and executables will not run as well natively in Android. Android first introduced the "Dalvik" virtual machine with just-in-time compilation (JIT) to run Dalvik "dex-code", which is usually translated from the Java bytecode. A runtime used to execute and host Android applications include the Dalvik virtual machine and the core libraries that provide Android specific functionality. Dalvik is Android's own optimized Java Virtual Machine to run the compiled Java class files in order to counter the handheld device limitations such as memory, processor speed, and power. Android Runtime (ART) in Android 5.0 and later versions replaces Dalvik as the platform default [Google 2017]. The ART runtime was introduced in Android 4.4 on an experimental basis. In October 2015 Google has released Android M - M for Marshmallow - as the eights major release of Android operating system.

Android applications can be developed using the Android SDK and ADT Plugin for Eclipse [Foundation and Inc 2017]. The development of Android framework is moving very fast and actively and the market share of Android framework is growing rapidly. Since May 2013 Google released the Android Studio [Google 2017] as the official IDE for developing application Android devices. The new IDE is based on IntelliJ IDEA [IntelliJ 2017].

With Android 5.0, 6.0 and 7.0 Google has targeted a wider range of portable devices. The new releases of the platform introduced TV and Auto profiles in addition to the existing smartphone, tablets and wearables profiles. The new features of Android platform add more support to Multimedia content acquisition and management, enabling better graphics for games, as well as audio, camera, and video processing.

2.2.3.2 IOS

iPhone [Apple Inc. 2017b] is a proprietary hardware and software platform released by Apple [Apple Inc. 2017a]. IOS inherits its architecture and technologies from Mac OS X; the Kernel is a variant of Mac kernel. It has four layers, Core OS, Core services, Media, and Cocoa touch [Apple Inc. 2015a].

Applications for iPhone are programmed with Objective-C using Xcode which is an SDK provided by Apple. Applications run natively on iPhone OS. Objective-C application framework is a set of extensions to the standard ANSI C language known as Cocoa to develop applications for Mac OS and iPhone. The two main development tools are Xcode and the Interface Builder. Applications can be developed using both Objective-C and C++. Apple compiler recognizes the code written with Objective-C by the extension *.m. A runtime system to run the compiled code is required. In 2014, Apple introduced SWIFT [Apple Inc. 2017c] as a new programming language for iOS. The code written in SWIFT can be run on Objective-C runtime system similar to code written in Objective-C and C++.

Apple has two runtime systems, modern and legacy. The most notable new feature is that instance variables in the modern runtime are "non-fragile". The runtime system acts as a kind of operating system for the Objective-C language. The application interacts with runtime at three levels; Objective-C source code, methods defined in the NSObject class of the Foundation framework, and direct calls to runtime functions.

2.2.3.3 Windows Phone and Windows 10 Mobile

As of October 2010 Windows Mobile is known as Windows Phone [Microsoft 2014b]. Windows Mobile is a compact operating system based on Windows CE targeted to resource-constrained handheld devices such as PDAs and smartphones.

The first version of Windows Mobile was based on Windows CE 3.0 and appeared in 2000 as "Pocket PC 2000". Many versions have been released afterwards. Microsoft has released Windows Phone 7 in October 2010 as the successor for Windows Mobile. Windows Phone 8.1 was released in Apr 2014. Windows 10 was released on July 2015 [Microsoft 2015b], it represents Microsoft's attempt to unify the desktop, tablet and phone operating systems into a single operating system. Windows 10 mobile shares many of the features as the Windows desktop, including the same kernel, UI elements, menus, settings, and Cortana [Microsoft 2015a]. It supports Continuum [Microsoft 2017b] on newer smartphone devices, allowing for a desktop-like experience by plugging the smartphone into an external monitor.

Microsoft Visual Studio for Windows Phone is the official tool for developing applications for Windows Phone. Application developers can develop using C# / Visual Basic.NET (.NET), C++ (CX), or HTML5/JavaScript. The .Net Compact Framework offers a collection of class libraries that provide many services and functionalities. It resides on top of the Common Language Runtime (CLR), which is in its turn placed on top of the operating System. The support for native C and C++ libraries allows some traditional Windows desktop programs to be easily ported to Windows Phone.

The .Net Compact Framework offers several libraries to enable dealing with multimedia content. The notable addition that Windows Phone 8.1 has added, is the support for the HTML5. This support is enabled by: i) Media Source Extensions (MSE) that support the HTML5. ii) Encrypted Media Extensions (EME). And iii) updates to the XAML Media Element control [Microsoft 2014b].

2.2.3.4 Java 2 Micro Edition (J2ME)

Java is the first cross platform programming language. J2ME applications can deal with many data formats such as XML, text, WML, XHTML and serialized Java objects. The applications can take the advantage of the strong wireless networking support of J2ME [Economou et al. 2008]. J2ME comes in at the third place in the family of the object-oriented programming environment created by Sun Micro Systems [Java 2017]. It is a subset of Java standard edition (J2SE), which targets resource-limited devices that cannot support the full implementation of J2SE.

J2ME follows a modular design [Kenteris et al. 2009] of configurations and profile in order to support a wide range of devices.

J2ME Configurations define the minimum features of java virtual machines and a minimum set of libraries. Currently there are two known configurations: i) the connected limited device configuration (CLDC) and ii) the connected device configuration (CDC).

J2ME profiles are implemented on the top of configurations. Those profiles are high level APIs for specific device groups. APIs provided by a profile normally consist of application life-cycle model, user interface, storage, specific network support and more, Mobile Information Device Profile (MIDP) is an example about the profiles.

J2ME applications do not have a main class as usually found in J2SE applications. J2ME provides the MIDlets instead. This implies that a J2ME application must extend the MIDlet class. The developer implements her J2ME application in the extended MIDlet class. For the execution environment, MIDlets are packaged together with resources, classes, and third party compatible libraries into a *.jar file. Developing mobile applications with J2ME has many advantages [Kenteris et al. 2009] since J2ME inherits the main assets of Java language of developing platform independent applications.

2.2.3.5 Symbian

Despite the fact that Symbian OS has been discontinued [Symbian 2016], we opted to include Symbian OS in our analysis for the sake of completeness. The roots of Symbian go back to 1980. David Potter has started to design games and office productivity software for Sinclair's personal computers under the name of Psion. In 1984, Psion Organizer was launched as the first handheld computer in the world. In 1998, Symbian started as a joint venture between Psion, Motorola, Ericsson and Nokia. In 2008, Nokia has purchased all the assets of Symbian and started down the path to make it as an open source [Symbian 2016]. In Apr 2014 Microsoft has completed the accusation of Nokia Devices [Microsoft 2014a]. However the discontinuation of the development of Symbian has been announced earlier in 2012 [Wikipedia 2014].

Symbian OS is designed for mobile devices with limited resources and expected to run months or years without being switched off. Symbian OS programming is event-based. The CPU is powered down when applications are not directly dealing with an event, this is achieved through the aid of a programming idiom called active objects. Correct use of these techniques helps to ensure longer battery life. Symbian OS has three user interface categories, S60, S80 and UIQ, the forth category S90 is merged with S60.

Using Symbian C++ provides full access to devices capabilities and speed advantage of native compiled applications. This compensates the difficulty of learning and developing with Symbian C++ in comparison to Java and Python. Python is a dynamic object-oriented open source computer programming language that was created and released by Guido van Rossum in 1989 - 1990. [Paython 2017]. Nokia provided a port to write python scripts and run them on Symbian S60 platform. Python for S60 enabled rapid application prototyping and development with scripts and provided the ability to create standalone S60 applications. Instead of using Java or Symbian C++, the Python script files can be archived in .sis package. A script file can be written using a text editor and saved as a *.py file [Nokia 2009].

Symbian OS provided a Multimedia framework to capture, edit and view media content. For example, the “ecam.lib” is dedicated to interact with the camera of the device. It has provided an object called “CCamera”, which allowed the developer of the application to control the camera. The “CCamera” used to use callback methods provided by the interface “MCameraObserver” which could be implemented and provided by the application. It worth to mention as well that the framework did not support the camera module on the emulators.

2.2.3.6 Summary

In this section, we provided a rough introduction on the main operating systems and development platforms that run on mobile devices. Table 2-7 summarizes the main features that can be connected to application development in each presented OS.

Platform	*Language*	*License*	*Prerequisites*	*Testing*
Android	Java, C, C++	Open source	Android SDK, NDK	Junit, Emulator, Device
IOS	Objective-C, C, SWIFT	Proprietary	IOS SDK	Emulator, Device
Windows Phone	C# C++ VB.net HTML5/JavaScript XAML	Proprietary	Visual Studio	Emulator, Device
Symbian	C, C++, Java ME, QT	Open Source	QT SDK	Emulator, Device
J2ME	Java ME	Standard – different implementations	J2ME SDK	Simulator, Device

Table 2-7: Summing up the programming languages, license, prerequisites, and testing options for each individual mobile platform

Table 2-8 lists the supported audio/video formats by each platform.

Platform	*Container Format (*)*	*Network Protocol*	*Video Codec*	*Audio Codec*
Android	mp4 (***)	http://audio rstp://video	H264	AAC Ic HE-AAC v2
IOS	mp4	http://	H264	AAC AAC+
Windows Phone	WMV (****) mp4	http://	H264 H263 WMV	MP3 AAC WMA
Symbian	3GP (**) mp4	rstp://	H263	AMRnb AAC

Table 2-8: Supported video/audio formats and containers by mobile platforms

(*) A container or wrapper format is a meta-file format whose specification describes how different data elements and metadata coexist in a computer file [Wikipedia 2016a].

(**) 3GP is used by many mobile phones and is based on the ISO base media file format.

(***) MP4 - standard audio and video container for the MPEG-4 multimedia portfolio, based on the ISO base media file format defined in MPEG-4 Part 12 and JPEG2000 Part 12, which in turn was based on the QuickTime file format.

(****) WMV - Windows Media Video is a video compression format for several proprietary codecs developed by Microsoft. The original video format, known as WMV, was originally designed for Internet streaming applications, as a competitor to Real Video [Wikipedia 2016c]

2.3 W3C Standards

In this section, we introduce relevant W3C standards to our work. W3C standards such as Semantic Web technologies, HTML, and SMIL are of a special meaning amongst the others to our approach. However, we are not aiming at providing full and comprehensive detailed analysis of these W3C standards. For further details, we refer the reader to the associated reference to each of these standards.

2.3.1 Semantic Web

The W3C defines the Semantic Web as the "Web of Data". Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data [W3C Org. 2016b]. In the conventional web of documents, website creators use markup languages and style-sheets to structure the information and the layout of web pages. While web browsers can display those markup authored web pages, content of those pages remain meaningless to the machines. Meanwhile, the human mind is able to interpret the meaning of content. Hence, there is a recognizable gap between what computers and humans understand [Sikos 2015]. Some resources such as images often encode machine readable metadata, however without context, the information provided by those metadata tags can be ambiguous to other software clients. To make content unambiguous and machine process-able, structured data can be added to the web sites in form of markup annotations or external metadata files. Structured data have been used in computing for decades, i.e., in SQL relational databases, where SQL queries can be executed to retrieve the information. By adding structured data to web pages, dual benefit can be achieved, that is, on the one hand, humans still can browse and read those web pages, and on the other hand machines, e.g., search engines and agents, can process those machine-readable data.

The term "Semantic Web" refers to W3C's vision of the Web of linked data. Linked data are empowered by technologies such as:

i) The Resource Description Framework (RDF). RDF will be presented in section 2.3.2 below.

ii) The Protocol and RDF Query Language (SPARQL), pronounced as "sparkle" and can be used to retrieve and manipulate information stored in RDF or in any format that can be retrieved as RDF [W3C Org. 2016c].

iii) The Web Ontology Language (OWL). OWL will be presented in section 2.3.3 below.

iv) The Simple Knowledge Organization System (SKOS) [W3C Org. 2016b]. Knowledge Organization Systems (KOS) are used for processing authority lists, classifications, thesauri, topic maps, and ontologies [Sikos 2015]. Figure 2-3 depicts the full stack to the Semantic Web.

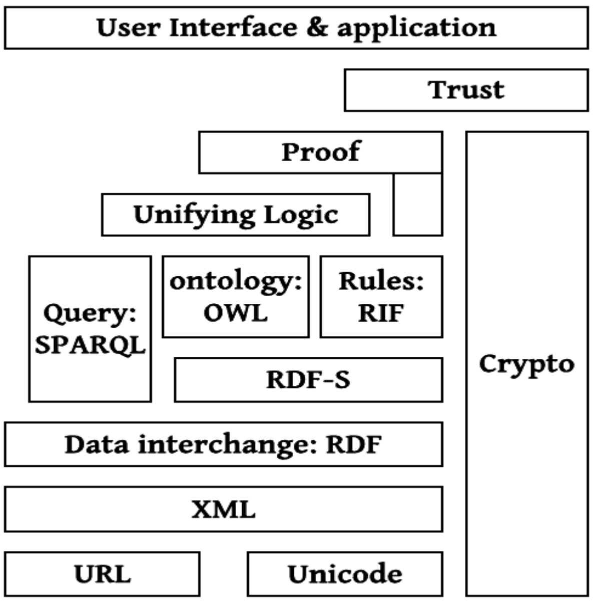


Figure 2-3: Semantic Web stack [W3C - Semantic Web 2007]

2.3.2 The Resource Description Framework (RDF)

RDF [W3C Org. 2014e] is a framework for expressing information about resources. Resources can be anything, including documents, people, physical objects, and abstract concepts. RDF provides a common framework that enables the applications to express and exchange data without loss of meaning. RDF datasets consist of collections of RDF statements (defined as triples), IRIs, literals, blank nodes and multiple graphs.

An RDF triple is constructed from three parts, these are, the Subject, the Predicate and the Object. An RDF statement expresses a relationship between two resources. The Subject and the Object represent the two resources being related; the Predicate represents the nature of their relationship. [W3C Org. 2014e].

The IRI is short for "International Resource Identifier", as the name suggests, an IRI identifies a resource. IRIs can be used in all of the three parts of a triple. For some certain cases, RDF specifications allow for defining resources without a unique global identifier, i.e., resources without IRI, which can be represented as blank nodes.

Literals are basic values that are not IRIs, hence, literals can be strings, dates and numbers. A Literal can be associated with a data-type enabling such values to be parsed and interpreted correctly. String literals can optionally be associated with a language tag.

RDF framework provides a mechanism to group RDF statements into graphs, and associate such graphs with an IRI. Hence, multiple graphs result from creating subsets of a collection of triples.

2.3.3 Ontologies (OWL)

Ontologies are used to capture knowledge about some domain of interest. An ontology describes the concepts in a domain and also the relationships between those concepts. There are different ontology languages that can be used to create an ontology, all refer to conceptualization through a data model for describing a piece of our world [Sikos 2015]. Ontology languages such as OWL [W3C Org. 2014c] support the following components [W3C Org. 2014c; Sikos 2015]:

Classes: Sets or collections of objects, classes and types in object oriented programming, concepts, or kinds of things.

Attributes: Aspects, properties, characteristics, or parameters that feature objects and classes.

Individuals: Instances or objects.

Relations: The logical relation between classes, between individuals, between collections or between different components such as classes and individual etc.

Function terms: Complex structures formed from certain relations that can be used in place of an individual term in a statement.

Restrictions: Formally defined descriptions or ranges of valid values.

Rules: If-then statements serving the logical inferences.

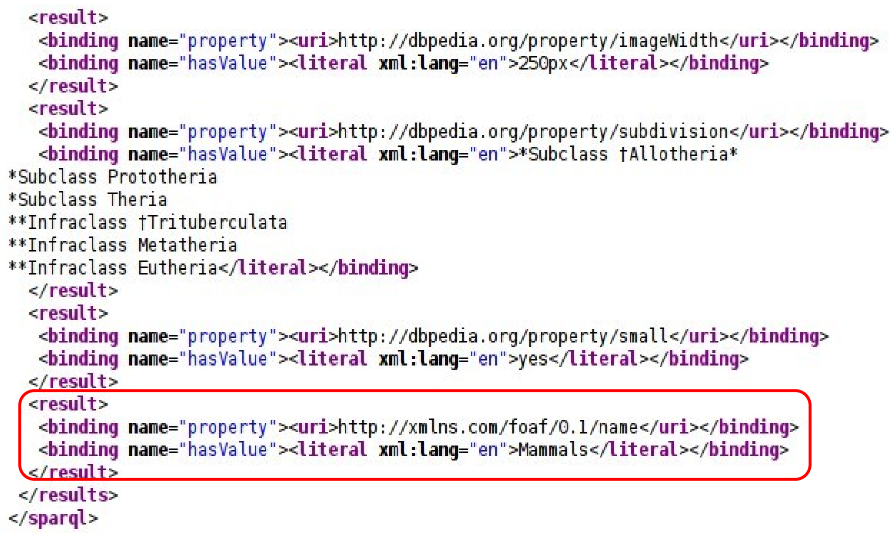
Axioms: Assertions in a logical form that, together with rules, comprise the overall theory that the ontology describes in its domain of application. Axioms are used to impose constraints on the values of classes or instances, so axioms are generally expressed using logic-based languages and can be used for verifying the consistency of the ontology.

Events: Attribute or relationship changes.

2.3.4 Linked Open Data (LOD)

In 2006 Tim Berners-Lee has introduced four rules [Bizer et al. 2009] for Linked Open Data: i) Use URIs as names for things. ii) Use HTTP URIs so that people can look up those names. iii) When someone looks up an URI, provide useful information using the standards of RDF and SPARQL. iv) Include links to other URIs, so that they can discover more things.

The result of applying these rules is a cloud of Linked Open Data that contains 1014 datasets as of 2014 compared to 295 datasets in 2011 offering more than 31 billion statements (triples) connected by more than 501 million links [LOD 2014]. The technical requirements of publishing Linked Open Data ensure that: i) the data are machine-readable, ii) the meaning of the data is explicitly defined, and iii) the data is linked to other external data sets and can in turn be referenced to from external data sets [Bizer et al. 2009]. Linked Open Data enabled new types of applications such as linked data browsers and search engines, and domain-specific linked data applications [Bizer et al. 2009]. Applications can use lookup services to discover the linked data [Hausenblas 2011]. Lookup services can be summarized as follows: i) Simple dataset-specific look up services, e.g., DBpedia. ii) General lookup services that allow looking up both terms and URIs. iii) Specified link type or entity services. Data publishers should use vocabularies of one or more of the widely deployed schemas. Those schemas are specified to make data machine readable and interchangeable. The XML snippet in Figure 2-4 depicts part of the result set of an SPARQL query about a resource sent to DBpedia APIs. As pointed out in the red box, DBpedia uses the namespace '*foaf*' to allow the inquiring client to correctly interpret a property called "*name*". In other words, the result indicates to the inquiring client that there is an attribute called "name" and it has namespace 'foaf' and a literal value "Mammals". If the inquiring client understands the 'foaf' vocabulary, the returned dataset can be correctly further processed. Table 2-9 and Table 2-10 list the top five used vocabulary schemas in 2011 [LOD 2011 2011] and 2014 [LOD 2014], besides the notable increase in the number of the datasets, we notice as well a change in the used vocabulary. Data publishers tend to use more well-established data vocabularies such as rdf and rdfs in their datasets providing for more interchangeability.



```
  <result>
   <binding name="property"><uri>http://dbpedia.org/property/imageWidth</uri></binding>
   <binding name="hasValue"><literal xml:lang="en">250px</literal></binding>
  </result>
  <result>
   <binding name="property"><uri>http://dbpedia.org/property/subdivision</uri></binding>
   <binding name="hasValue"><literal xml:lang="en">*Subclass †Allotheria*
*Subclass Prototheria
*Subclass Theria
**Infraclass †Trituberculata
**Infraclass Metatheria
**Infraclass Eutheria</literal></binding>
  </result>
  <result>
   <binding name="property"><uri>http://dbpedia.org/property/small</uri></binding>
   <binding name="hasValue"><literal xml:lang="en">yes</literal></binding>
  </result>
  <result>
   <binding name="property"><uri>http://xmlns.com/foaf/0.1/name</uri></binding>
   <binding name="hasValue"><literal xml:lang="en">Mammals</literal></binding>
  </result>
 </results>
</sparql>
```

Figure 2-4: DBpedia result set showing the usage of the 'foaf' vocabulary

VOCABULARY PREFIX	NUMBER OF DATASETS IN 2011 (TOP5)
DC*	92
FOAF	81
SKOS	58
GEO	25
XHTML	19

Table 2-9: Datasets used in the Web of Data ordered by number of datasets in 2011

VOCABULARY PREFIX	NUMBER OF DATASETS IN 2014 (TOP 5)
RDF	996
RDFS	736
FOAF	701
DCTERMS (*)	568
OWL	370

Table 2-10: Datasets used in the Web of Data ordered by number of datasets in 2014

(*) Dublin Core (DC) was introduced in 1995 -1996 before the invention of RDF in 1998. Elements of DC were declared among the early RDF properties. As RDF has matured, these properties were criticized for being underspecified and ambiguous. To overcome this problem, DCMI created fifteen properties and assigned a new namespace to them, that is the '*terms*'. Those terms go in parallel to corresponding old terms that were introduced in the *elements 1.1* namespace. DCMI encouraged data publishers to use the semantically more precise '*dcterms*' properties, as they fully comply to the emerging notions of best practice for machine-process-able metadata. [Dublin Core - Terms 2015].

2.3.5 HyperText Markup Language (HTML)

Since the first public introduction of HTML-Tags in early 90s [HTML - Public document 1992], HTML [W3C Org. 2014b] has become the standard markup language for creating web pages and web applications. In the past twenty-five years, many revisions of HTML were published, Table 2-11 summarizes those revisions:

YEAR	VERSION
1989	Tim Berners-Lee invented WWW
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	WHATWG HTML5 Living Standard

2014	W3C Recommendation: HTML5
2016	W3C Candidate Recommendation: HTML 5.1
2017	W3C Working Draft, 28 February 2017: HTML 5.2

Table 2-11: The history of the development of HTML [HTML History 2017]

HTML documents have two parts, the head part represented by the element <head>. The head part includes the title of the document, the metadata tags that add additional information about the document and other elements as depicted in Figure 2-5. The second part is the body, which is represented by the <body> tag. The body part contains all layout and the media elements that the document presents, i.e., the content of the document.

```
<HEAD>
    <META CHARSET="UTF-8">
    <BASE HREF="https://www.... /">
    <TITLE>An application with a long head</TITLE>
    <LINK REL="STYLESHEET" HREF="default.css">
    <LINK REL="STYLESHEET ALTERNATE" HREF="big.css" TITLE="Big
      Text">
    <SCRIPT SRC="support.js"></SCRIPT>
    <META NAME="APPLICATION-NAME" CONTENT="Long headed
       application">
</HEAD>
```

Figure 2-5: The <head> tag and its sub elements as defined by the HTML specifications

HTML and specifically version 5 introduces new tags such as <audio>, <video>, <svg>, etc. to enable the inclusion of new media elements such as audio, video, and animation in addition to existing elements such as images, text, and hyperlinks. HTML5 adds a corresponding Document Object Model (DOM) API for those new added elements, i.e., in case of video objects, the new video API can be used to detect support for different video formats, play a video, pause, mute audio, track how much of the video has been downloaded, and other required features to build a rich user experience around the <video> tag.

HTML does not provide constructs to specify temporal synchronization between the elements [S. Boll et al. 1999]. However approaches such as [Laiola Guimarães et al. 2014] suggests a set of document extensions that can be specified by means of Cascading Style Sheets (CSS) to allow timing and synchronization of HTML elements.

2.3.6 Synchronized Multimedia Integration Language (SMIL)

SMIL [W3C Org. 2014f] is the W3C recommendation for describing multimedia presentations. SMIL 1.0 became a W3C recommendation in June 1999. The versions 2.0 and 2.1 have followed few years later, the latest version 3.0 was published in December 2008. Despite the fact that there is no distinct support by the industrial sector for SMIL, SMIL language represents an intuitive and a well-structured markup language for describing the basic aspects of multimedia presentations as well as for composing sophisticated and more advanced presentations. SMIL language introduces multiple profiles, each profile contains several SMIL modules to satisfy the purpose of that profile, e.g., the SMIL Unified Mobile Profile is a collection of SMIL modules that provide support for the SMIL Language within the context of mobile devices. The Unified Mobile Profile expects such devices to have high-resolution displays and sufficient memory and processor capacity to render SMIL documents [W3C Org. 2014f]. This profile encompasses the required modules to meet the needs of a wide range of interactive multimedia presentations. The Open Mobile Alliance (OMA) [OMA 2017] implemented SMIL language for representing multimedia presentations on handheld and mobile devices [OMA - MMS].

SMIL language introduces a set of modules that define the semantics and syntax for certain areas of functionality, e.g., layout module, timing and synchronization module, animation module and many other modules.

A multimedia presentation that is described by SMIL language consists of two main parts: i) The <head> element, which contains the <meta> tags and the <layout> tags. ii) The <body> element, which contains the timing information, and is generally composed of combinations of three main tags: sequential <seq>, parallel "<par> and exclusive <excl>.

2.4 Approaches and Applications of Document Authoring and Sharing

In this section, we discuss the approaches and applications in the domain of multimedia document authoring and sharing in order to provide a comparison of these approaches in the subsequent sections. We observe two types of approaches and applications: The first type deals with pre-composed multimedia documents. In these approaches the creation of the multimedia document is not of special interest. The problems addressed by these approaches are related to retrieving and adapting the pre-composed multimedia document, very often also targeting personalizing content by considering the use of user profiles. The second type of approaches and applications mainly addresses techniques for authoring and sharing multimedia documents. Current systems and prototypes are mainly designed following a *client-server* architecture, which eases fulfillment of all the requirements of multimedia document authoring and sharing. These systems are based on the integration of different frameworks and building blocks developed by a research community working on different areas of multimedia composition and sharing process.

2.4.1 Approaches dealing with pre-composed multimedia documents

We divided this type of approaches into two main subtypes, these are, i) the retrieval and ii) the adaptation of pre-composed multimedia documents.

2.4.1.1 Multimedia content retrieval of pre-composed multimedia documents

When looking at solutions for multimedia content retrieval in the context of mobile devices we observe that location-aware information retrieval has gained an increasing attention in the past few years. Considering the location of the user when deciding what could be important, relevant, and at the same time interesting to be presented to the user has become an important and emerging topic in multimedia retrieval systems. Many approaches and applications use the location of the user as an input to deliver services, in particular in the context of mobile devices, as one can see from the variety of apps of different categories such as tourist and city guides, city street maps, trip advisors, trip planers, hotel finders, car rentals, etc., offered in famous app-stores. Because these location-based applications deal with pre-composed multimedia documents, they focus on retrieval and delivery of the content, but do not support the authoring process of the content. Multimedia documents processed by such location aware systems usually include text and images and provide useful context-aware information to the user. Obviously, multimedia documents that are delivered by such location-aware applications can be considered as rich input for multimedia document authoring applications that perform content analysis, processing, and knowledge extraction tasks. Subsequently we provide a more detailed description of some of these approaches. In the following, we briefly summarize selected approaches dealing with multimedia content retrieval of pre-composed multimedia documents.

The MobiDENK [Krösche et al. 2004] project aims to provide a location-based multimedia content for mobile users about cultural sites in Lower Saxony. The project follows earlier approaches in providing location-based multimedia content, namely GUIDE [Cheverst et al.], deep map [Malaka and Zipf 2000] and LoL [Pospischil et al. 2002]. The main task of the application is to inform the user about the historic sites nearby her actual location by offering her multimedia content and a user friendly application on a portable computer such as PDA. MobiDENK offers the users interesting and exciting information that is easy to understand. When the user approaches a historical site, an icon is displayed on the map. Clicking on that icon a multimedia presentation starts and the content is provided as a HTML web page. The system consists of four main modules: a geographic information system module, a point of interest module, a location module, and a simple multimedia presentation module. MobiDENK does not address the process of composing the multimedia document. It only supports retrieval of multimedia content.

[Hosokawa 2008] presents a location-aware information browser implemented on BREW-based mobile phones. This approach argues that mobile users such as drivers and pedestrians decide their future activities using location-dependent information based on their points of interest. It proposes a method to implement a location-aware information browser on mobile phones to improve the process of information acquisition and presentation to help the users in making decisions based on additional information. The system is designed to look for nearby points of interests the user is not aware of. It aims to transform given map information into a user- and location-specific map with enriched information elements on the points of interest. The information browser is composed of two subsystems: 1) a map data reader and 2) an active map transformation sub-system. The browser relies on five types of data: a) map data, b) user's current location, c) point of interest, d) user's request, and e) space filter. In order to generate the so-called display-sized map the browser performs the following main steps: a) It automatically generates the space filter. b) It filters unnecessary map elements. c) It clips map elements from the minimum bounding rectangles. d) It applies the space filter to the map. e) It renders transformed map elements on the screen.

[Ahlers and Boll 2009] presents a broader understanding of user's spatial context in mobile retrieval systems. The approach aims to provide information that is more precise to the user by exploiting special spatial filters and relevance ranking methods. The approach proposes to use spatial proximity search like search at departure, search on route, search at destination of a user's path, search along a route, search along direction, or a combination of spatial proximity and temporal proximity filters. Based on these filters and the available data such as route, speed, range, and time, a combined spatial footprint for queries can be built. These techniques allow for filtering the query result items or point of interests based on the user's heading direction instead of displaying just all the items in the surrounding area. Furthermore, combining textual and spatial ranking allows for the higher ranking of result items that are farther away but still highly relevant compared to those result items that are close by but not as relevant to the user.

[D'Souza et al. 2005] In the Bluetooth Village Guide Book (VGB), the user browses the multimedia content related to an object placed in the Kelvin urban village. The user downloads an application that grants the access to a service point using Bluetooth technology. The user can then navigate through menu system and select the contents she would like to view.

2.4.1.2 Adaptation of pre-composed multimedia documents

Adaptation of multimedia content contributes to multimedia content delivery, reuse, and portability. In multimedia document adaptation, we observed two main techniques: Adaptation by manipulating the quality of the individual media content within the multimedia document. This technique includes scaling, quality reduction, modality conversion, and trans-coding (e.g., [Jannach et al. 2006]). The second adaptation technique is by replacing the multimedia content with semantically equivalent content (e.g., [Bertolotti et al. 2006]). In the following, we illustrate these two major approaches to adaptation by selected approaches.

In the approach presented in [Jannach et al. 2006] adaptation of the original media content (e.g., video) is suggested in order to conform to the available bandwidth of the network and the size of the screen. The implemented adaptation techniques are: 1) Adaptation by scaling. 2) Adaptation by quality reduction. 3) Adaptation by modality replacement. 4) Adaptation by trans-coding. The system considers adaptation problems that are related to *Session Mobility*: It is argued that a user can start watching a film using her mobile device and continue watching it using her PC. A semantic framework is used to determine the techniques, which will be used for the adaptation process amongst the available adaptation systems. The adaptation process is achieved through two main steps: 1) Planning by examining the features of the requested multimedia content and the specifications of the target device. 2) Configuring a suitable sequence of adaptation services that is built based on the planning results by a knowledge-based approach. This approach integrates the MPEG-21 multimedia framework and Semantic Web technology by combining concepts from OWL-S and MPEG-21.

The approach presented in [Cesar et al. 2008] combines structured multimedia document adaptation and mechanisms of session transfer. The approach provides a continuous dynamic adaptation of multimedia content in a given user's context at any given moment. This approach argues that the multimedia content does not necessarily have to be rendered on a single device but it can be transferred and presented across many devices. Hence, it targets the presentation to the "sphere" of a user, i.e., devices surrounding the user. To achieve the requirements, the system should be aware of the description of the actual device, dynamically generate the multimedia presentation, and provide a session continuity mechanism.

In the approach described in [Laborie et al. 2007] the adaptation of the multimedia document takes place prior to delivery. The approach discusses local and global adaptation. Local adaptation implies the individual adaptation of each multimedia component separately. Global adaptation implies the adaptation of the whole document. The approach presented focuses on global adaptation. Adaptation is applied on a composed multimedia document and it considers the user's profile, for example, if the profile of the user may prohibit some type of contents, such content will be removed during adaptation.

In the work presented in [Bertolotti et al. 2006] a framework for the characterization of document adaptation in the presence of both physical and user oriented context requirements is proposed. The system represents the document presentation by means of an automaton and is aware of all the active multimedia components at any time. Based on semantic equivalences between multimedia content fragments specified by the document authors and stored as metadata in a database, the system chooses equivalent components as alternatives for components that cannot be delivered for presentation. In the absence of equivalence, undeliverable media components are replaced with media components that minimize the loss of information/quality in the presentation.

2.4.2 Approaches focusing on authoring and sharing of multimedia documents

The system presented in [Tummala and Jones 2005] provides the users with an easy to use web-based interface to add their own contents and related location information. Contents can be of any type of media, i.e., text, pictures, audio, or video. The system targets both groups of users, content providers and content consumers, and it does not require advanced technical skills for adding content in a time efficient manner. For this group of content consumers, the content can be accessed with any web browser-enabled device. The design goals are as follows: 1) Easy and device platform-independent access. 2) Balance between ease of content management for content providers and quality of content for users. 3) Comprehensive range of location-specific content. 4) An intuitive, well-defined structure of content, based around the concepts of places, organizations, and events on a timeline. 5) Simple method of associating a place with its geographic coordinates. 6) Incorporation of legacy contents. 7) Content provider management rights. 8) Universal support for any Locale scaling for any geographical region. 9) Valuable interface for user output.

CoCoMA [Eidenberger et al. 2007] is an integrated platform developed in the framework of the DELOS II European networks of Excellence on Digital Libraries. The approach combines frameworks for retrieval, adaptation, authoring, and presentation. The participants of the project define two phases in the life cycle of multimedia content, *Presentation Creation* and *Presentation Consumption*. The integrated platform includes components of previous projects from contributing authors. The main components of the integration project that cover *Presentation Consumption* are 1) MM4U, which is a framework to help developers in developing personalized multimedia applications that follow a general multimedia personalization process. 2) KoMMA that is a multimedia content adaptation server that supports basic and advanced adaptation operations by integrating tools and algorithms and that performs an adequate sequence of adaptation operations on the media content. The main components that cover *Presentation Creation* are 1) VizIR, which is an open development framework that allows for the use of multimedia retrieval techniques in the course of content creation. 2) DS-MIRF, which is a framework that facilitates the development of knowledge-based multimedia applications utilizing and extending the MPEG7 and MPEG-21 standards. 3) SyMPA, which is the presentation specification and generation component of the CoCoMA architecture.

In the multi-channel multimedia presentation generation approach presented in [Scherp and Boll 2005c] the authors propose a multimedia document structure consisting of: 1) a temporal model based on a local and global timeline, 2) a spatial model offering either an absolute positioning or a relative positioning of content elements, and 3) an interaction model for enabling some navigational functions over the multimedia document. The main idea of this approach is to separate between the media elements and logical structure of the underlying multimedia document on the one hand, and the presentation of the document on the other hand. This approach aims to define an abstract multimedia composition model for documents that can be transferred to different multimedia presentation models for final presentation. It employs some mapping algorithms that transform the abstract model to a representation model based on the abilities of the target device.

The framework on context-aware personalization for mobile multimedia services presented in [Weiß et al. 2008] supports the filtering of contents based on the preferences of a user and context information. The framework introduces the notion of static and dynamic context, and it proposes two different approaches for filtering contents depending on the kind of context.

xSMART [Scherp and Boll 2005a] is a context-driven authoring wizard that supports the author throughout the steps of a multimedia content composition process. It offers the very basic set of multimedia composition functionality. This includes the selection of multimedia content and the creation of a simple layout in a page-oriented fashion.

This work constitutes an approach toward a semi-automatic multimedia composition tool.

[Kenteris et al. 2009] presents an approach to generate a tourist guide by means of a personalized J2ME-application customized for a certain user according to her mobile device specifications. The tourist guide is an offline application available as a download from a server. Any user creates a profile capturing his/her preferences. That profile will be used later on to compile a customized the J2ME-application. The multimedia content for the tourist guide is encoded as XML document and is packaged with the J2ME-application including a customized program to present the multimedia content on the device of the user. The user is offered a main menu that enables her to browse the multimedia documents that were originally selected on the website serving as the authoring environment prior to compiling and then downloading the customized application. This work constitutes an approach of manual selection, a kind of pre-specified template-based composition, and manual viewing of contents. Content composition covers spatially arranging text and images. User interaction is supported by means of navigation through the selected multimedia documents.

[Economou et al. 2008] represents early work trying to understand the technical needs for authoring tools and development platforms in the domain of cultural applications for mobile devices. The authors introduced three types of cultural applications for mobile devices based on a review of the state of the art technologies for developing mobile cultural and tourist applications. The authors concluded a set of requirements for developing cultural applications on PDAs and mobile devices. They made case studies on three different applications: 1) An authoring tool for the development of a cultural multimedia application on PDAs. 2) A newsreader mobile phone application. 3) An electronic tourist guide on mobile phones. Although the technical platforms significantly changed in the meantime, the overall requirements concluded are still valid.

[Jokela et al. 2008] introduces a Mobile Multimedia Presentation Editor that enables the user to compose multimedia content on the mobile device by systematically following a user centered design approach. The authors have conducted a study and concluded four principles for designing and developing the mobile multimedia presentation editor. The four principles are *flexibility*, *awareness of task context*, *expressiveness*, and *personalization*. The designed editor interface has four views. 1) The presentation view allows the user to browse and compose new presentations. 2) The player view allows the user to view the selected presentation. 3) The edit presentation view provides the user with tools to create and edit SMIL presentations. 4) The preview presentation view allows the user to preview the presentation at any point during the composition.

[Xiao et al. 2010] presents an application called iPhotobook that allows for the creation of photobooks on mobile devices. The application integrates algorithms for image selection, cropping, pagination, page layout, and background scaling. The users can arrange their photo album through a graphical user interface (GUI) that supports touch and motion gestures. This work can be considered as semi-automatic approach, because of the combination of algorithms for handling images and manual graphical arrangement of the album by the user.

A framework for ontology-driven multimedia analysis and composition [Dasiopoulou et al. 2007] introduces two complementary approaches. 1) A semantic framework for multimedia retrieval, which automates the generation of annotations of media elements from high level semantics, i.e., annotations, qualitative attributes, i.e., color homogeneity, and low-level features, i.e., color models. The framework uses RDFS schema for knowledge representation and employs F-Logic rules to describe how tools for multimedia content analysis should be applied. 2) A semantic framework for multimedia composition, which is based on reasoning over domain knowledge. This results in an adaptive and automated multimedia generation process. This approach aims at enabling systems that can automatically annotate multimedia resources and compose multimedia presentations.

In the approach presented in [Scherp and Boll 2005c] the authors tackle the problem of providing the multimedia content to a wide range of devices. To achieve this goal, they presented an approach of multi-channel generation of multimedia presentations. They analyzed the different multimedia presentation formats and developed an abstract document model that embeds the central characteristics of multimedia presentation formats. The abstract model can be translated to a concrete multimedia presentation format that is accepted by the target device.

mProducer [Teng et al. 2004] is a system for generating personal experience content on mobile devices which has four components: A Storage Constrained Uploading (SCU) algorithm, a sensor to detect and remove the blurry frames, a map-based content management interface, and a key frame-based editing tool. The process of authoring comprises two phases: During the capturing phase data are captured and saved either on the device or on the server. This phase includes preparing and finding key frames and filtering blurry frames. During the editing phase the user selects content to be edited by choosing a point on the map. Then a list of content elements which are organized based on capturing location is displayed

[Barrenho et al. 2006] presents the authoring tool InAuthoring, that provides a graphical user interface for creating spatial stories and gaming activities. It is part of the InStory project, which developed a client-server system enabling the users to interact with a server by providing their GPS location while exploring a site of e.g., cultural, historical, or natural interest. The user receives multimedia content represented in a structured XML document. InAuthor provides a graphical user interface that allows a visual approach to activity creation and respective geo-referencing. The authoring process is performed by dragging and dropping the media content such as text and images into the workspace and arranging the position of the media items. Afterwards the content can be connected to related physical geo-locations.

In [Scherp 2008] the author introduces a canonical processes mapped to a general creation chain for authoring personalized semantically rich multimedia presentations. The general creation chain is derived from intensive analysis of the approaches and systems for authoring, personalizing, and semantically enriching multimedia presentations conducted by the author.

The general creation chain has four phases: selection of media element, assembling content, transforming resulting presentation into a multimedia document format, and presenting the resulted document on the target device. These four phases support eight canonical processes for media production. The first phase includes the creation of media assets by means of querying a storage system. The second phase comprises all the process steps for the composition of the content, including annotation of media elements with metadata, represented in an abstract representation model. The third phase covers the packaging and transformation of the abstract representation into a concrete format needed for publishing the content. The last phase covers delivering and rendering of the content to the end user. The defined processes were implemented in the SemanticMM4U framework.

[Rabbath et al. 2010] is an approach to generate photobooks from the shared photos on the social network platforms. The authors propose an approach to automatically detect media elements, i.e., the photos that match a query such as "where", "when", "what", and "who" in the social network of a user, and then intelligently arrange and compose the detected photos into a printable photobook.

POLI [Fan et al. 2015] represents an interactive multimedia authoring and retrieval system using mobile devices. The authoring process is triggered by locating hotspots on physical objects, e.g., a painted picture in a museum. Once an authoring region has been spotted, the system administrator can author digital content by using a mobile authoring application. The end user uses a mobile client application to retrieve the authored content form a web server by using the same hotspot locating mechanism.

[Yin et al. 2013] presents a system to automatically build snippets in text-overlaid image styles for browsing social media on mobile devices. The proposed system extracts text posts from a social media platform, e.g., twitter and a dominant image. Both the text posts and the image are extracted by means of a modified TextRank algorithm. The extracted elements are used to compose social snippets following aesthetic rules and visual perception principles.

SIMPEL [Murthy et al. 2006] presents an authoring environment to compose SMIL multimedia documents. This authoring software addresses issues such as selecting or working with information elements at sub-document level while retaining the original context, and describing the integration or packaging of such elements, and making use of minimal storage during the authoring activity.

[Hong and Kim 2012] presents a multimedia authoring and virtual collaboration system that produces multimedia presentations for e-learning content. The authoring process makes use of SMIL to enable the author (the teacher) to compose multimedia presentations (lectures) combining video with power point slides.

[Azevedo et al. 2013] presents a semi-automatic multimedia authoring application that enables inexperienced users to author multimedia presentations by following a template-based approach.

RhythmPix [Loui et al. 2005] represents an approach that enables users to manually enrich image albums with music and style to create easy-to-share presentations. It encompasses three modules that enable users to collect, combine, and compose images, music, and other related content. Created presentations can be viewed with devices such as VCD, SVCD, and DVD players, as well as on PCs.

2.5 Criteria for detailed analysis of related approaches

In this section, we develop a set of criteria that helps us to come up with an in-depth analysis of the related approaches presented in section 2.4 above. We decided to follow a process-oriented view on the model for authoring and sharing. The general process of creating and personalizing multimedia documents as presented in [Scherp and Boll 2005b] can serve as an appropriate reference model, because it is very generic and intuitive. This reference process consists of four phases: Phase 1 is about the selection of the media assets such as images, text, video and audio for the composition. Selecting the media content depends on input parameters such as the profile of the user and the subject of the document. Phase 2 is about the assembly of the selected media assets in time, space, and by means of user interaction elements and its representation in a coherent, structured, abstract multimedia document. Phase 3 is about transforming the abstract multimedia document to a concrete multimedia format. Phase 4 is about presenting the transformed multimedia document which includes delivery, rendering, and viewing the multimedia document. Figure 2-6 depicts the four phases of the generic multimedia document creation and personalization process. For more details about the general process of creating and personalizing multimedia documents we refer the reader to [Scherp 2008].

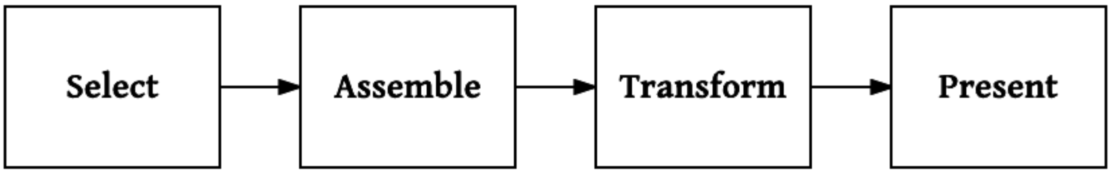


Figure 2-6: Generic multimedia content authoring process as presented in [Scherp and Boll 2004]

Based on the two types of approaches identified in section 2.4 above, these are **(i) approaches dealing with pre-composed multimedia documents, and (ii) approaches focusing on authoring and sharing of multimedia documents,** we define criteria enabling us to perform a structured in-depth analysis and comparison between the approaches and applications previously discussed.

Table 2-12 provides a full summary of the set of criteria. The first column encodes the number of the criterion. The second column names the criterion and its indicator: Each criterion is either mapped to a single phase of the four phases of the reference process or to the whole process. The letter (C) indicates that the criterion is related to the whole process of *Composition*, (S) indicates that the criterion is related to the first phase *Selection*, (A) indicates that the criterion is related to the second phase *Assembly*, (T) indicates that the criterion is related to the third phase *Transformation,* and (P) indicates that the criterion is related to the last phase *Presentation*. The third column lists the range of categories used for the evaluation of each criterion. Next to the individual values we denote the abbreviation of the value as used in the analysis results as shown in Table 2-13 and Table 2-14.

NO.	CRITERION (INDICATOR)	RANGE OF CATEGORIES USED FOR EVALUATING A CRITERION
1	Where (C):	Server (S) Client (C) Distributed (S-C) (S-S) (C-C)
2	How (C):	Manual (M) Automatic (A) Semi-automatic (S)
3	When (C):	Pre-composed (P) On-the-fly (O)
4	Context type (C):	Static (S) Dynamic Simple (DS) Dynamic Analyzed (D)

5	Context usage (C):	Composition(C)
		Adaptation(A)
		Retrieval(R)

6	Adaptation (C):	Select and replace (S) Transform(T)
7	Domain (C):	Domain-specific(D) Generic (G)
8	System type (C):	Open system(O) Closed system(C)
9	Content types (C):	Text(T), Image(I), Audio(A), Video(V), Links(L)
10	Selection (S):	Automatic: - Mathematical/statistical (M) - Semantic (S) Manual: - Search (R) - Manual selection (L)
11	Sources (S):	Predefined sources (P) On-the-fly detection of resources (O) LOD (L)
12	Composition (A):	Template-driven (T) Free-style/human-guided (F)
13	Annotation (A):	XML, MPEG-7, MPEG-21, Textual, RDF
14	Multimedia document requirements (A):	Temporal (T) Spatial (S) Interaction (I)
15	Document model (T):	Open (O) Standardized (S)

NO.	CRITERION (INDICATOR)	RANGE OF CATEGORIES USED FOR EVALUATING A CRITERION
		Proprietary (P)
16	Frameworks & Platforms (C)	Platforms dependent (D) Platforms independent (I)
17	Networking technology (P):	WLAN-based (W) PAN-based (B)
18	Delivery (P):	Streaming (S) Download (D)
19	Presentation (P):	Web browser (W) Proprietary app (A): Flash(F) Real Player(RP) MMS Player (MP) Generic player (GP)

Table 2-12: Criteria for comparing and analyzing the approaches and applications

In the following, for each criterion we introduce the name of the criterion used for referencing the criterion in subsequent sections and a more detailed description:

Where – the components of the computing architecture where the composition of multimedia content is performed.

This criterion is evaluated by means of three different architectural concepts:

Server (S): the composition of the multimedia document takes place at the server side. (ii) Client (C): the composition of the multimedia document takes place at the client side, in particular mobile devices. (iii) Distributed: the composition is distributed amongst several devices and, hence, may take place either on multiple servers (S-S), or multiple clients (C-C), or on servers and clients (S-C).

How – the degree of user involvement in the composition process.

This criterion is evaluated by the degree of automation of the composition process:

Manual (M): Manual composition implies that the process steps of selection, assembling, and transforming the multimedia content are performed manually by the user. (ii) Semi-automatic (S): semi-automatic composition implies that some of the process steps are performed automatically, some others are performed manually. Examples of automatically performed steps are classifying the media content based on certain criteria, annotating the content, and providing a template or wizard for completing the process of the composition. Examples for manually performed steps are the final selection and arrangement of media elements according to temporal, spatial, and interaction relationships, and finally choosing the final encoding format. (iii) Automatic (A): Automatic multimedia document implies that the application performs the consecutive steps of the general multimedia composition process automatically and without interaction with the user.

When – the time the composition process steps take place.

This criterion is evaluated with respect to the following categorization:

Pre-composed (P): this implies that the composition is performed at a certain point of time prior to the consumption of the document. The pre-composed documents can be saved in repositories and retrieved upon the request of the user. However, adaptation techniques may still be applied before the pre-composed multimedia document is delivered to a target device. (ii) On-the-fly (O): all the composition steps are performed at the time the user requests the multimedia document for presentation. This allows for the incorporation of users-specified parameters, user context, and profile of target device.

Context type – the kind of contextual information considered during content composition.

This criterion is evaluated with respect to the following categorization: (i) Static (S): static contextual information covers pre-specified information about the user (e.g., user profile) and about devices, e.g., screen size, support for colors, and supported media types etc. These information elements do not change during the composition process. (ii) Dynamic Simple (DS): Dynamic contextual information covers information elements changing over time, e.g., GPS based positioning information on the user. (iii) Dynamic Analyzed (D): advanced contextual information is a result of a further analysis of the information covered by the previous context types. For example, given the dynamic contextual information of GPS coordinates, further analysis may conclude the city, the historical site, or a district in the city to be considered during the content composition.

Context usage - the purpose contextual information is used for.

Contextual information can be used for different purposes, we distinguish between three types of usages; hence, this criterion is evaluated with respect to the following categorization:

Composition (C): contextual information is used in any, one or more phases of the phases of the authoring process. (ii) Adaptation (A): contextual information can be used as input for the adaptation of a pre-composed multimedia document. (iii) Retrieval (R): contextual information can be used to construct a query for retrieving pre-composed multimedia documents.

Adaptation – the manipulation of the multimedia document or the mono-media elements.

This criterion is evaluated based on the different adaptation techniques we presented in section 2.4.1.2 above; thus, it provides the following categories: Select and Replace (S): some devices do not provide support for playing back certain media formats, e.g., video codecs. Those not supported media elements can be replaced by selecting another semantically equivalent alternatives. (ii) Transform (T): to transform the multimedia document from one document format to another (e.g., SMIL to HTML5), or to transform the encoding of one media element from one format to another (e.g., JPEG to GIF).

Domain – the domain of knowledge that the application is optimized for.

This criterion is evaluated with respect to the following categorization: (i) Domain-specific applications (D): applications that support multimedia composition for a specific domain of knowledge, i.e., tourism, medical etc. (ii) Generic (G): application and frameworks that support generic authoring of multimedia documents, i.e., those frameworks do not require that the authored multimedia document should belong to a specific domain of knowledge.

System type – Closed or Opened systems following the closed and open world assumption.

In closed-world assumption, a statement that is true is also known to be true, conversely, a statement that is not known to be true, is false. In open-world assumption, having a statement that is not known to be true, does not imply it is false. We apply same concepts on this criterion, hence it can be evaluated with respect to the following categorization:

Closed Systems (C): Closed Systems can only see those media content that are available in the surrounding environment. (ii) Open Systems (O): Open Systems do not regard the elements that they do not reach as if they do not exist.

Content types – types of mono-media elements that the system supports.

This criterion is evaluated by detecting the types of the mono-media elements that the compositing application includes/supports in the authoring process. In our analysis, we consider the following types: (i) text, (ii) image, (iii) audio, (iv) video and (v) links.

Selection – how media elements are retrieved and selected.

This criterion is evaluated by means of the following two techniques for media selection:

Automatic selection (A): the authoring environment selects the media elements automatically by either applying Semantic Web technologies (S) or by applying mathematical and statistical models (M). (ii) Manual Selection (M): the authoring environment enables the author of the multimedia document to select the media elements either manually (L), or by providing search functionality (R).

Sources – sources of media elements used in the composition.

This criterion is evaluated by considering the following sources of media elements: (i) pre-defined sources (P): the application provides certain media elements that the author can use to author a new document. (ii) on the fly detected sources (O): authoring application detects the media elements on the fly. (iii) Linked Open Data (LOD) sources: the application supports retrieving media content from the web, e.g., LOD APIs.

Composition – template-driven composition or free human-guided style.

This criterion is evaluated by means of two composition techniques: (i) template-driven composition (T): the authoring application provides a set of templates to the author; the author can choose one of those templates. Optionally the authoring environment can guide the author through the authoring steps by means of a composition wizard. (ii) Free human-guided style (F): the authoring environment grants the author full control on the authoring process.

Annotation – metadata schemes used to enrich the composed multimedia document with additional useful metadata.

This criterion is evaluated in respect with the following annotation schemes: (i) Standard annotation (S), in this method, standard metadata schemas are used. This implies that annotating multimedia documents makes use of the standard metadata schemas, i.e., EXIF, MPEG-7 and MPEG-21. (ii) Custom annotation (C), custom annotation implies that the authoring application uses either structured annotation tags (e.g., meta tags) or unstructured textual description about the content, i.e., text to provide additional information about the multimedia media content.

Multimedia document requirements – fulfillment of basic multimedia document requirements.

This criterion is evaluated by evaluating whether the authored content fulfill the basic requirements of multimedia documents or not. Basic requirements include (i) Temporal (T): this requirement is evaluated by detecting the ability of the authoring application to use either local or global timeline to create temporal relationships between the media elements. (ii) Spatial (S): this requirement is evaluated by detecting the ability of the authoring application to use either point based or relative system to create spatial relationships between the media elements. (iii) Interaction (I): this requirement evaluates the ability of the authoring application to provide interactional links. Interactional links in multimedia documents enable the user to interact with the document to control the advances of viewing the document and other dynamic elements such as video and audio. We will provide more details about those basic requirements in chapter 3.1.3. However, for detailed information about this topic, we refer the reader to [S. Boll et al. 1999].

Document model – the support of standard/known multimedia document models.

This criterion is evaluated with respect to three categories of multimedia document models: (i) Standard open source models (S): the authoring application makes use of one of the standard and open source multimedia document models, i.e., SMIL or HTML5 etc.) (ii) Proprietary (P): the authoring application makes use of one of the proprietary multimedia document models, i.e., Flash, PowerPoint. (iii) Open (O): the authoring application makes use of own document model, that can be represented and encoded by means of markup languages such as XML.

Frameworks & Platforms – frameworks and platforms used to develop and run the application.

This criterion is evaluated by means of the following categorization: (i) Platforms dependent approaches (D): approaches that target a specific mobile framework, e.g., Android, IOS etc. (ii) Platforms independent applications (I): approaches that can be implemented on any mobile framework.

Networking technology – the networking technology that the application uses to communicate data.

This criterion is evaluated by means of two different computer networking technologies: (i) Wireless Local Area Networks (WLAN) (W): WLAN uses the internet protocol (TCP/IP) to interconnect devices and exchange the data. (ii) Personal Area Networks (PAN) (P): PAN use both wired technologies such as USB and WrieFire and wireless technologies such as Bluetooth, IrDA, ZigBee etc. to interconnect devices and exchange the data.

Delivery – how multimedia document are delivered to the recipient device.

This criterion is evaluated by means of two main media content delivery techniques: downloading (D): multimedia content consuming device should download the complete content to be able to display it. (ii) streaming (S): streaming the document implies that the end consumer can display the document while downloading it.

Presentation – the media player that is used to display the multimedia content.

This criterion is analyzed with respect to the media player that is used to display the multimedia content. Operating systems determine which media player to use to display the media content based on the file extension, i.e., based on the format on the media content. In the context of mobile devices, we consider in our analysis the following applications and media players (i) web browsers (W): web browsers support HTML/HTML5 natively. Other document formats such Flash can be viewed by installing additional plugins. (ii) proprietary players (A): proprietary player usually supports viewing some proprietary document format, e.g., Adobe Flash player. (iii) MMS players (MP): MMS players are embedded in the pre-installed messaging application on mobile devices. Since the MMS protocol implements SMIL (see section 2.3.6 above), they support a simplified version of SMIL document format, hence, they are capable to display mobile multimedia messages (MMS). (iv) Generic Players (GP): generic players support several document formats, e.g., Real player, media player from Microsoft etc.

2.6 Analysis of related work

In this section, we provide an in-depth analysis of the related approaches based on the criteria that we defined in section 2.5. The undertaken analysis is a two levels analysis; meaning that one level analysis is a macro level analysis, and the two-level analysis is a micro level analysis. The macro level analysis provides a large-scale analysis by examining criteria elements that relate to the process of multimedia document authoring and sharing indicated by the letter (C), i.e., criteria 1 to 9. Table 2-13 lists the approaches included in the macro analysis and the results of applying each criterion from 1 to 9 on those approaches. The micro level analysis provides an in-depth analysis by examining criteria elements that relate to each phase of the phases of the generic process of multimedia content composition indicated by the letters (S), (A), (T), and (P), i.e., criteria 10 to 19. Table 2-14 lists the approaches included in the micro analysis level. In both tables the columns represent the criteria, the rows represent the examined approaches. For the analysis, we followed a vertical analysis approach. Vertical analysis enabled us to (i) provide a cross-application criterion-based analysis, and (ii) define patterns in the approaches and applications.

Level 1 (Macro analysis)

1. **Where (C):** The *"Where"* criterion influences the overall approach of the multimedia document authoring. Approaches that target multimedia content authoring in the *Server* (S) environment shown less concern about hardware capabilities and computation power of the server. Hence, those approaches handle several types of media elements such as image, text, video and audio as in [Scherp and Boll 2005c] and [Barrenho et al. 2006]. Server approaches support the adaptation as in [Dasiopoulou et al. 2007], and more complicated user interaction and user interfaces. Approaches that target the *Mobile Device* (C) consider to the limited resource of mobile devices. New generation of mobile devices offer more memory and processing power, however the small touch screens remain as drawback towards advanced and interactive user interfaces, hence applications consider simple yet intuitive user interface design as seen in [Jokela et al. 2008] and [Xiao et al. 2010]. *Distributed* composition in its different variations (S-S), (S-C), and (C-C) is not observed among the approaches we presented in section 2.4. In the distributed composition approach, the tasks of the authoring process can be distributed over multiple interconnected devices. Each device executes its tasks and return the results to the device that assigned these tasks.
2. **How (C):** The *"How"* criterion defines three authoring styles, *manual* (M), *automatic* (A), and *semi-automatic* (S). We have not observed any influence of the previous criterion *"Where"* on this criterion. That is, both *Server* (S) and *Mobile Device* (C) approaches support all authoring styles. Approaches that support *semi-automatic* authoring process, e.g., as in [Xiao et al. 2010] and [Scherp 2008] argue that it is not feasible to have either a *manual* or an *automatic* authoring process. Those approaches argue that a *manual* authoring process is resources and time consuming process, meanwhile, an *automatic* process can be very complicated and

high quality results cannot be guaranteed. Hence, a *semi-automatic* process bridges the gap between the other two composition styles. In the initial steps, *semi-automatic* authoring process performs initial composition tasks on behalf of the author, such as the selection of the related media content. In following steps, the author decides which media elements to include, and the final structure and layout of the multimedia document.

3. **When (C):** The *"When"* criterion defines the point of time the authoring of the multimedia content takes place. Early approaches and domain specific applications tended to use the *pre-composed* document approach (P), e.g., as in [Krösche et al. 2004], [Teng et al. 2004], [D'Souza et al. 2005]. Those approaches use different types of context, i.e., *Static* (S), *Dynamic Simple* (DS), and *Dynamic Analyzed* (D) to adapt and personalize the pre-composed multimedia content. *On the fly* (O) composition approaches use both context types (S) and (DS) to produce multimedia documents tailored to those context inputs about the user.

4. **Context type (C):** Approaches make use of the *Static* (S) and *Dynamic Simple* context (DS) either to select the most relevant content to the user, e.g., as in [Krösche et al. 2004] and [Kenteris et al. 2009], or to make the proper adaptations to the multimedia content as in [Scherp and Boll 2005a]. In *Static* context (S), the user provides profile's information, additional context information such as the screen's size and the media types supported by the presentation device can be automatically detected. This type of context data remains static and does not change over the time, obviously, unless the user changes her device, however, this implies that the usage of this type of context can be limited to certain tasks such as content selection and adaptation as seen, e.g., in [Kenteris et al. 2009] and [Hosokawa 2008]. In *Dynamic Simple* (DS) context, certain information such as GPS location changes more often. Approaches use (DS) context information in multimedia content retrieval, however unlike (S) context information, this type of context widen the range of the selection. In *Dynamic Analyzed* (D) context data, light weight analysis processes can be executed to deducted context data, e.g., by analyzing sensor data. In the presented approaches, we observed less usage of (D) context data.

5. **Context usage (C):** Generally, context information can be utilized throughout the phases of the multimedia authoring process as seen in [Scherp and Boll 2005a]. Approaches define what type of context information and how to use this additional context information in the authoring process. This implies that depending on the authoring phase, the authoring process specifies which information to use and how, e.g., in the selection phase, the authoring process selects the media elements that are technically and/or semantically relevant to user's profile.

6. **Adaptation:** Adaptation is a basic requirement of multimedia content. As we presented in section 2.4.1.2, Adaptation can be either semantically by means of substitution or technically by means of transformation. Approaches apply *Semantic Adaptation* (S) by substituting the individual media content within a

given multimedia document with same semantically equivalent media content as in [Scherp and Boll 2005c]. Approaches apply *Transformation* (T) adaptation by changing the physical features of the media elements, i.e., by changing the size of an image or changing the format of a video, e.g., as seen in [Xiao et al. 2010] and [Dasiopoulou et al. 2007].

7. **Domain (C):** For this criterion we analyze two types of multimedia document composition approaches. i) *Domain* specific approaches (D): approaches that assume that the author of the multimedia is a specialist in a certain domain of knowledge, e.g., tourism. The authoring process guides the author throughout the authoring process, e.g., by means of authoring wizard. This allows the author to concentrate on selecting the content of authored multimedia document. Hence, these approaches do not require highly qualified technical users. ii) *Generic* approaches (G): approaches that provide a multimedia content authoring framework and tools. The authoring environment in these approaches offers the user full control over all the phases of the authoring process. *Generic* composition approaches and tools usually require highly qualified users.
8. **System type:** For this criterion we look at the sources of the media content that the authoring approach considers. Based on the defined categories we concluded that the approaches we presented in section 2.4 can be considered as closed system.
9. **Content types (C):** The Content type criterion can be linked to other criteria such as *Where*, *Document Model*, *Adaptation*, and *Domain*. The variety of media content types indicates how much advanced the authoring process is. However, the analyzed approaches vary in their support for different types of media content. Client (C) supports specific types of media elements such as video as seen in [Teng et al. 2004] or image as seen in [Xiao et al. 2010]. Server (S) tends to support a wider range on multimedia content types, i.e., text, image, audio, and video as seen in, e.g., [Scherp and Boll 2005c; Barrenho et al. 2006; Dasiopoulou et al. 2007]. The *Document Model* used by an approach influences the supported media types by that approach. Approaches that make use of advanced *Document Models* such as SMIL support a wider range of media types. Select and Replace (S) *Adaptation* is used by approaches that support continuous media content such as video and audio, e.g., [Scherp and Boll 2005c]. Approaches that support discrete media content such as image used Transform (T) *Adaptation*, as seen in [Dasiopoulou et al. 2007; Xiao et al. 2010]. Domain-specific (D), e.g., approaches targeting Tourism and Location Based Services tended to use specific types of media content such as image and text as seen in [Krösche et al. 2004; D'Souza et al. 2005].

APPROACH OR APPLICATION / CRITERIA	WHERE	HOW	WHEN	CONTEXT TYPE	CONTEXT USAGE	ADAPTATION	DOMAIN	SYSTEM TYPE	CONTENT TYPE
[KRÖSCHE ET AL. 2004]	S	M	P	DS (GPS)	R	-	D (TUR)	C	T I
[SCHERP AND BOLL 2005A]	S	S	O	DS	C	T	D	C	T I A V
[KENTERIS ET AL. 2009]	S	M	P	S	R	-	D	C	T I
[HOSOKAWA 2008]	S	-	-	DS (GPS)	R	-	D (POI)	C	-
[ECONOMOU ET AL. 2008]	S	M	P	-	-	-	D (MUS)	C	I V
[JOKELA ET AL. 2008]	C	M	O	-	-	-	G	C	T I A
[XIAO ET AL. 2010]	C	S	O	S	A	T	D (PHOTO)	C	I
[DASIOPOULOU ET AL. 2007]	S	S	O	S	A	T	D	C	T I A V
[SCHERP AND BOLL 2005C]	S	S	O	S	A R	T S	D	C	T I A V
[TENG ET AL. 2004]	C	M	P	DS (GPS)	C R	-	G	C	V
[D'SOUZA ET AL. 2005]	S	M	P	DS	R	-	D (MUS)	C	T I A
[BARRENHO ET AL. 2006]	S	M	P	DS (GPS)	R	-	D (POI)	C	T I A V
[SCHERP 2008]	S	S	O	S-DS	R	-	D	C	T I A V

Table 2-13: Comparison of approaches based on macro level analysis

Level 2 (micro analysis)

10. **Selection (S):** In selection phase, media content is selected following different selection techniques as defined in this criterion. The selection technique depends on the degree of user involvement in the composition process, as discussed in *"how"* criterion. *Semi-automatic* approaches as in [Xiao et al. 2010] make use of multiple selection techniques, i.e., in the first selection step, media elements are selected on-the-fly and grouped based on their technical features, e.g., color histograms. In the next selection step, the author selects certain content manually. Generally, *Semi-automatic* approaches provide algorithms to automatically group and sort the media elements to assist the author to locate and chose the media elements for the inclusion in the multimedia content. Systems that follow the *manual* approach as in, e.g., [Krösche et al. 2004] and [Jokela et al. 2008], offer the user the access to media repositories. *Manual* selection can be accomplished either by browsing the file system or by using keywords search to locate the media elements.
11. **Sources (S):** Media elements such as audio files, video files, and images can be stored in multimedia repositories. Approaches such as [Kenteris et al. 2009] and [Jokela et al. 2008] provide means to the author to access these *pre-defined* (P) sources. The author of the multimedia document can select specific media elements form those sources by means of the selection techniques as presented in the criterion "*Selection*". Approaches such as [Rabbath et al. 2010] target media elements stored in social area networks. Hence, these approaches support *On-the-fly* (O) sources. The "Web of Data" offers a massive amount of multimedia resources provided by Linked Open Data (LOD) APIs. Such rich sources need to be considered by the multimedia authoring tools and systems.
12. **Composition (A):** *Domain* (D) specific applications such as [Scherp and Boll 2005a] and [Xiao et al. 2010] presented in the macro analysis level tended to support *Template-driven* (T) composition. In *Template-driven* composition, applications provide templates for the multimedia document. The author can select one of the provided layout templates to be used as a layout for the multimedia document. This technique allows the author to concentrate on the content of the document rather than the design. *Free human-guided style* (F) is supported by approaches such as [Kenteris et al. 2009] and [Dasiopoulou et al. 2007]. Those approaches offer *manual* or *semi-automatic* authoring process. Manual authoring of multimedia document require highly skilled users, however approaches such as [Jokela et al. 2008] provide an intuitive graphical user interface (GUI) to compensate the complexity of the manual composition.
13. **Annotation (A):** Multimedia documents and their included mono-media elements can be enriched with additional information. This additional information enables the authoring process to perform several tasks such as selection, adaptation and other tasks. In the approaches we presented in our analysis, we have not detected explicit support for annotation or annotation process. In [Scherp 2008], as part of the *assembly* (A) phase, the authoring process annotates the individual media

elements by means of RDF statements to enable future reuse of the resources. As presented in section 2.3.1 above, adding annotations by means of RDF scheme allows other machines to find and interpret the semantics of these media elements.

14. **Multimedia document requirements (A):** This criterion is part of the *assembly* (A) phase. To fulfill the basic requirements of multimedia documents, the authoring process must support *temporal* (T), *spatial* (S), and *interactional* (I) features of multimedia documents. Approaches that make use of standard multimedia document models such as SMIL or an abstract document model such as ZYX [S. Boll and W. Klas 2001] as seen in [Scherp and Boll 2005a] and [Scherp 2008], offer the user an authoring environment that is capable to fulfill those basic requirements. Since HTML does not support the temporal model as presented in section 2.3.5, approaches that use HTML such as [Krösche et al. 2004] support only the spatial and interactional models.

15. **Document model (T):** Approaches such as [Scherp and Boll 2005a] and [Teng et al. 2004] make use of the *Standardized* (S) multimedia document models such as SMIL, or *Proprietary* (P) models such as Flash or PowerPoint. Those applications deliver multimedia documents that can be viewed by the available media players. This implies that the multimedia documents that these approaches produce are interchangeable, meaning that, users can share and view those multimedia documents without being forced to install extra software other than the standard software to playback those documents. Approaches such as [Kenteris et al. 2009] and [Dasiopoulou et al. 2007] use *Open* (O) models. Those approaches provide their own media player that is capable to view the multimedia documents.

16. **Frameworks & Platforms :** This criterion examines the execution environment that each approach is optimized for. Approaches such as [Xiao et al. 2010], [Jokela et al. 2008] and [Teng et al. 2004] are optimized to run on mobile devices, this means those approaches are developed for a specific platform and device. [Kenteris et al. 2009] presents an approach to generate a tourist guide by means of a personalized J2ME-application customized for a certain user according to her mobile device specifications. The client side of this approach requires a device that runs J2ME virtual machine. The approaches that are designed to run on the S*erver* (S), do specify explicitly a platform.

17. **Networking technology (P):** This criterion is part of the *presentation* (P) phase. In this criterion we examine the networking technology that is used to transfer the multimedia document form the authoring device to the presentation device. The examined approaches make use of mobile networks such as [Xiao et al. 2010]. Approaches that support *Server* (S) side composition as in, e.g., [Dasiopoulou et al. 2007] and [Scherp and Boll 2005c], rely on WLAN and mobile networks to deliver the multimedia content.

18. **Delivery (P):** This criterion is part of the *Presentation* (P) phase. Multimedia documents delivery varies between the examined approaches. Since standard document models such as SMIL and HTML add native support to media elements streaming, approaches such as, e.g., [Scherp and Boll 2005a] and [Teng et al. 2004] support *Streaming* (S) implicitly, even if their authoring process supports explicitly the *Downloading* (D) method. Approaches that support offline playback of the authored multimedia documents as seen in, e.g., [Kenteris et al. 2009] and [Dasiopoulou et al. 2007] rely on the *Downloading* (D) method.
19. **Presentation (P):** The document model that is used by the authoring applications to represent the multimedia document decides the playback medium. Approaches that use *Standardized* (S) multimedia document models such as SMIL, or *Proprietary* (P) models such as Flash or PowerPoint are not obliged to provide their own media players. However, applications such as [Kenteris et al. 2009] provided their own media player that is capable to playback the multimedia documents produced by this approach.

APPROACH OR APPLICATION / CRITERIA	SELECTION (S)	SOURCES (S)	COMPOSITION (A)	ANNOTATION (A)	MULTIMEDIA DOCUMENT	DOCUMENT MODEL (T)	PLATFORM AND FRAMEWORKS	NETWORKING TECHNOLOGY (P)	DELIVERY (P)	PRESENTATION (P)
[KRÖSCHE ET AL. 2004]	-	P	-	-	S	S	D	W	S	W
[SCHERP AND BOLL 2005A]	R	P	T	-	S T L	S	I	W	D	GP
[KENTERIS ET AL. 2009]	R	P	F	-	S	O	D	W	D	A
[JOKELA ET AL. 2008]	L	P	F	-	S T	S	D	GPRS MMS	-	A
[XIAO ET AL. 2010]	M L T	O	T	-	S	O	D	G3	-	A
[DASIOPOULOU ET AL. 2007]	S	P	F	-	S T I	O	I	W	-	RP
[SCHERP AND BOLL 2005C]	L	P	F	-	S T I	S	I	-	D	GP
[TENG ET AL. 2004]	S L	P	F	-	T	S	I	W	-	GP
[SCHERP 2008]	R	P	F	RDF	S T I	S	I	-	D	GP

Table 2-14: Comparison of approaches based on micro level analysis

2.7 Conclusion

In this chapter we presented the underlying infrastructure and concepts for mobile computing and multimedia document authoring and sharing. We have presented mobile networks and personal area networks, mobile operating systems and platforms, the relevant W3C standards to our approach; and selected examples of the existing approaches and applications in the domain of multimedia document authoring and sharing. We developed a set of criteria that helped us to come up with an in-depth analysis of the related approaches presented in section 2.4. Our analysis involved two levels of analysis, which enabled us to (i) provide a cross-application criterion-based analysis, and (ii) define patterns used by approaches and applications. Based on this analysis, we concluded that:

1. There is a clear need for smart authoring and sharing of multimedia content in personal area networks, this motivates us to introduce a new innovative approach to smart authoring and sharing of multimedia content in personal area networks based on user's subjects of interest. Personal area networks seem to be rich environments for authoring and sharing multimedia content. Hence, this work is motivated by four aspects:
   - The evolution of smartphones into very powerful devices for handling multimedia content.
   - The rapid growth of social network communities and as a consequence of thereof the willingness to share content.
   - The well-established technologies of Semantic Web providing powerful tools and techniques, and the increasing number of Linked Open Data access points providing a rich source for enriching the authoring process with additional content.
   - The networking technologies enabling mobile devices to interconnect and exchange data.
2. Our approach needs to fulfill the following requirements:
   - Enable the users to share multimedia content in personal area network environment.
   - Analyze the content provided by various users to explore what can be of interest to a user for the purpose of composing interesting pieces of content.
   - Detect and interrelate information provided by the member devices of personal area networks.
   - Make use of the "Web of Data" to augment the detected multimedia content.
   - Incorporate the Semantic Web technologies in the authoring and sharing process.
3. Unlike other presented approaches and applications, this approach for authoring process is driven by the user's Subject of Interest, which can be inferred from the analysis of some initial content that serves as the starting point for the authoring process.

3. The DCMC Approach

Having presented background information, the related work and technologies, and the state of the art of the research on multimedia documents authoring and sharing, we present in this chapter our own approach called Dynamic Cross-Model Composition of multimedia documents in personal area networks (DCMC [Abu-Naim and Klas]). The DCMC approach is motivated by the following four aspects: (i) the evolution of smart devices' hardware and operating systems presented in chapter 2.2.3. (ii) User's tendency to join social network platforms to share multimedia content about their daily life events. This tendency is supported by the related statistics as shown in Table 2-1. (iii) Well-established methods and technologies of Semantic Web presented in chapter 2.4.1. And (iv) the increasing establishment of Linked Open Data (LOD) APIs as depicted in Table 2-9 and Table 2-10.

In section 3.1 we discuss the disambiguation of the term "Multimedia". In section 3.2 we present the derived requirements for multimedia content authoring and sharing in personal area networks. From the requirements presented in section 3.2 we design and present in section 3.3 the enhanced authoring process incorporated in DCMC. In section 3.4 we define and introduce the service-oriented architecture of DCMC.

3.1 Multimedia 101

3.1.1 "Multimedia" Disambiguation

Nowadays, in the context of multimedia computing the term "Multimedia" is used to describe a specific type of digital content. Multimedia computing is defined as the manipulation and presentation of digital contents in a computer system [Little 2003]. "*Multimedia computing is an application oriented technology that caters to the multisenary nature of humans and is based on the evolving ability of computers to store, transmit, and convey diverse types of information*" [Little 2003]. In computer applications particularly, the term "Multimedia document" is used to identify a digital material container that includes different types of digital content such as video, audio, image and text. The term "Multimedia" consists of two parts, the adjective "multiple" and the noun "media" which is the plural of "medium". In the dictionary, "multiple" as an adjective has a straight forward meaning, that is "having or involving several parts, elements, or members". Meanwhile, the noun "medium" has several meanings, those relevant definitions in this context are: (i) "A particular form of storage material for computer files, such as magnetic tape or discs." (ii) "A means by which something is communicated or expressed". (iii) "An agency or means of doing something". (iv) "The material or form used by an artist, composer, or writer". To reach the most applicable definition of Multimedia in the context of our work, we examine the four definitions of "Medium". The definition (i) of "Medium" in the word "Multimedia" tells us that "Multimedia" means multiple storage media that can be used to store analog or digital data (e.g., magnetic tape, CD, DVD, or flash memory). By examining the definitions (ii) and (iii), "Multimedia" means the media channels that can be used to communicate analog or digital data to recipients (e.g., TV, Radio, Magazines, or social media). The definition (iv) leads us to a more acceptable definition of the term "Multimedia" in the context of multimedia computing, that is the material or form (e.g., Text, Image or Video) used by an artist or composer (e.g., the content author). Multimedia materials or content are involved in many fields of application computing such as Advertising, Education, Tourist Information and many others [Little 2003]. However, as a conclusion about the term "Multimedia", "Multimedia" is used in several fields with a distinctive meaning of the term in each field.

3.1.2 Multimedia documents, models, and presentation formats

In this section we define multimedia documents, models, and presentation formats. A presentation ready multimedia document should realize the separation between two logically distinguished parts, the document model and the presentation format. A document model defines the elements and the characteristics of a multimedia document. It defines as well modeling primitives that cover the aspects of a multimedia presentation. The presentation format defines how a multimedia document can be exchanged and rendered for presentation or playback.

Multimedia document models are data schemes that can be used to construct the relations between some of the multimedia document constituents. Multimedia document constituents [Klas and Sheth 1998] include: (i) The media element that can be either discrete (e.g., Text, Image) or continuous (e.g., Video, Audio) [Heller et al. 2001] [Klas and Sheth 1998]. (ii) The media descriptions that can be either characterizing or descriptive. Characterizing describes technical features of media elements (e.g., media format, size, duration). Descriptive is content-based or semantic descriptions (e.g., author name, creation date). And (iii) the composition which encapsulates the temporal and spatial design, the relations between media components, and the styles and layout for single components. Well-established existing document models are provided either by the software industry as proprietary format (e.g., Flash) and the World Wide Web Consortium (W3C) (e.g., SMIL 3.0 [W3C Org. 2014f], HTML5 [W3C Org. 2014b], SVG [W3C Org. 2011]) as standards, or by the scientific community (e.g., ZYX [S. Boll and W. Klas 2001], Madeus [Jourdan et al.]) as scientific contributions.

Multimedia presentation formats determine the representation of a multimedia document for exchange and rendering. Presentation formats can be either standardized such as the World Wide Web Consortium (W3C) standards (e.g., SMIL [W3C Org. 2014f] and SVG [W3C Org. 2011]), or proprietary format (e.g., Flash).

However, in both cases, it is difficult to separate between the document model and the presentation format, e.g., Flash format defines implicitly the document model in Flash documents.

As a conclusion for our approach, a document model should consider multimedia document authoring capabilities and characteristics. A multimedia document is considered as an instance of a multimedia document model. A presentation format should provide a serialization of such a document for the purpose of exchange and playback.

3.1.3 Basic requirements to multimedia documents

As presented in the previous section, a multimedia document consists of two parts, the document model and the presentation format. This means, besides the one requirement of the separation between the model of the document and its presentation format as discussed in section 3.1.2, a multimedia document has to realize at least three other basic requirements. These basic requirements are the temporal model, spatial model, and interaction possibilities of a multimedia document [S. Boll et al. 1999]. Following we present a brief description of these three features:

Temporal model: A temporal model describes the temporal order between media elements of a multimedia document [S. Boll et al. 1999]. By means of temporal models, authors of multimedia documents can determine when a specific media element is presented within the lifecycle of a multimedia presentation. Temporal models describe temporal dependencies between the media elements of a multimedia document [S. Boll and W. Klas 2001]. For example, media elements contained in a multimedia document can be presented *simultaneously*, *overlap*, *precede* or *after* another media element. Available temporal models are (i) Point-based temporal models: point-based temporal models depend on the time space. Each event or element in the model is associated with a time point. The "timeline" is an example of this approach. (ii) Interval-based temporal models: interval-based temporal models depend on Allen Expressions [Allen 1983]. Each media element is considered as time intervals ordered according to some relation. Allen relation expressions encode relations such as before, meets, overlap, during and other relations. (iii) Interval expressions-based temporal model: interval expressions-based temporal model depends on the Interval Expressions presented in [Keramane and Duda 1997]. This model claims to overcome the drawbacks of the two temporal models presented in (i) and (ii). This temporal model approach can deal with the unknown interval duration. Interval expressions encode relations such as seq(a,b), follow(a,b), par-begin(a,b) and other relations. (iv) Event-based temporal models: In an event-based model of time, events determine the temporal course of the presentation. An event is connected to actions and when an event occurs, e.g., a video reaches a certain point in time, the corresponding actions, typically start and stop the presentation of other media elements, is carried out [S. Boll et al.]. (v) Script-based realization of temporal relations: in script-based temporal models the temporal relation between media elements is realized by providing script programs written in scripting language to comprise temporal synchronization operations. [S. Boll and W. Klas 2001] [Keramane and Duda 1997].

Spatial layout: A spatial model describes the positioning of the media elements of a multimedia document in relation to each other. With the aspects of the spatial model, the author can arrange the elements of the multimedia documents within the X,Y and Z axes of the presentation. Available spatial models are (ii) absolute positioning: in absolute positioning there are two possibilities to place the media elements in the spatial space of the multimedia document: (a) Media elements are placed at an absolute position with respect to the origin of the coordinate system. (b) Media elements are placed at a position relative to another media element. (ii) Directional relations: in directional relations the arrangement of the media elements in the spatial space is described in relation like north, north-west. To add more granularity relations like strong-north and week-north were introduced. (iii) Topological relations: in topological relations; between any two continuous region objects, the following eight topological relations can be distinguished: disjoint, meet, overlap, covers, covered-by, contains, inside, and equal. [S. Boll and W. Klas 2001] [S. Boll et al.].

Interaction possibilities: Interaction with the multimedia documents describes how the user interacts with multimedia presentations in order to choose between different presentation paths and perform other actions. Interaction models enable the user for example to select or repeat parts of presentations, speed up a movie presentation, or change the visual appearance. Available types of interaction are (i) navigational interaction: in navigational interaction the user is granted the control over the flow of the presentation. It allows the user to select one of the possible presentation paths. (ii) Scaling interaction and (iii) movie interaction: in scaling interaction and movie interaction the user is able to manipulate the visible and audible layout of a presentation. For example, the user can change presentation's volume or spatial dimension. Navigational and design interactions should be specified within multimedia documents, whereas movie interactions are expected to be offered by the presentation engine. [S. Boll and W. Klas 2001] [S. Boll et al. 1999].

3.2 The Epics of DCMC

The overall goal of DCMC approach is to achieve an automatic authoring and sharing of multimedia documents in personal area networks by defining an enhanced multimedia content authoring process. In this chapter we use the term "authoring process" to refer to the authoring process of DCMC. The authoring process in DCMC is capable to: (i) Automatically establish a personal area network to enable data exchange between the connected devices. (ii) Automatically analyze and exchange the multimedia content shared by the users. (iii) Compose multimedia documents about the inferred Subject of Interest (SOI). (iv) Retrieve and use additional data from LOD sources. (v) Achieve a cross-multimedia document model authoring. In this section we present our requirements to the authoring process in PAN. We follow the notions of the Agile Requirements Engineering methodologies to define the functional requirements of the system.

In Agile Requirements Engineering methodologies; functional requirements are reflected as User Stories. User Stories are defined in different layers of abstraction: Epics, Features and Stories [Gunyho and Gutiérrez Plaza 2011]. In this work, we use the acronym EFUS to represent Epics, Features and User Stories. EFUS are different in definition, scope and size. An Epic represents the highest-level expression of a customer need [Leffingwell 2011]. Thus, it can be understood as an abstract system feature that encompasses several Features (e.g., booking of hotel rooms). A feature represents a short, descriptive, value delivery and benefit oriented statement [Leffingwell 2011], e.g., registered users can book rooms, Features can be further decomposed into several User Stories. A User Story describes functionality that will be valuable to either a user or purchaser of a system or software [Cohn 2004]. It describes a concrete system function and is typically written in a human readable language. It captures the "who", "what" and "why" of a requirement in a simple, concise way [Leffingwell 2011] [Cohn 2004] [Wikipedia 2016b] . User Story often takes a standard user-voice form [Leffingwell 2011] or active-voice [Cohn 2004] of the following:

```
As a {user role}, I can {activity} so that {business value}
```

For example, As a *registered user* I want to *search* un-booked rooms so that I can *book* a room by myself. As a summary, an Epic encompasses several Features, a Feature can be decomposed into several User Stories. Figure 3-1 depicts the hierarchical relationship between Epics, Features, and User Stories.

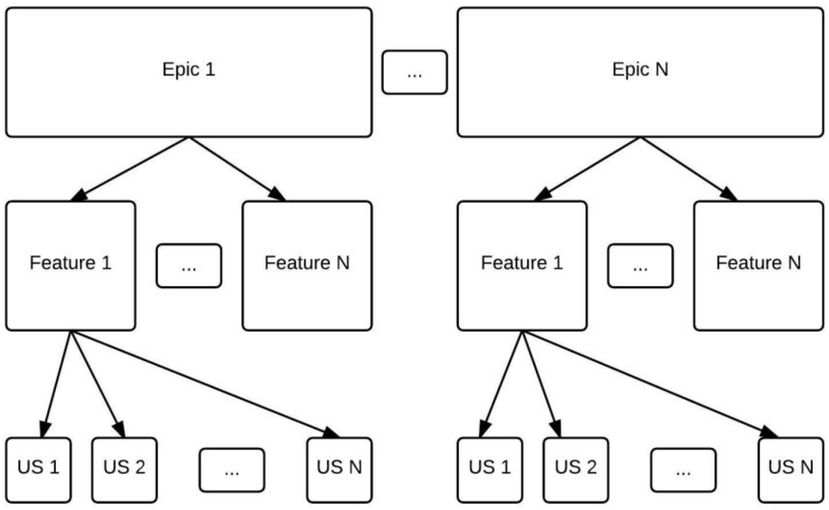


Figure 3-1: The hierarchical relationship between Epics, Features, and User Stories

User stories provide a clear description about the required function and are usually written by the customer [Leffingwell 2011]. However, User Stories lack technical details needed by the software developers to correctly interpret and implement the required function, i.e., in the User Story given in the previous example, the developer can implement the search function in at least two different ways. The developer can let the user search for un-booked rooms by either (i) providing two dates (e.g., check in and checkout dates), or (ii) by providing a date and the length of her stay (e.g., arrival date and number of days). To address this lack of information, acceptance criteria [Leffingwell 2011] or acceptance tests [Cohn 2004] can be attached to a User Story to provide more details about the required function, e.g., the user must provide check-in and check-out dates to search for un-booked rooms, date format must be as follows: yyyy-MM-dd. Such criteria give the developer clear technical requirements to implement the function correctly and as expected by the customer. Table 3-1 lists the examples of Epics, Features, User Stories, and acceptance criteria.

Epics	*Features*	*User Stories*	*Acceptance criteria*
Booking of hotel rooms	Registered users can book hotel rooms	As a registered user, I want to search for the un-booked rooms so that I can book a room by myself	The user shall provide check-in and check-out dates to search for un-booked rooms. Date format shall be as follows: yyyy-MM-dd.

Table 3-1: Examples of Epic, Feature, User Story, and Acceptance Criteria

Meanwhile, in conventional software development methodologies, e.g., waterfall methodology, system features and services are defined by means of requirement statements. Software system requirements are often classified as functional requirements, non-functional requirements, or domain requirements [Sommerville 2001]. Functional requirements define typically the functions of a system and its components [Sommerville 2001]. Functional requirements are usually supported by non-functional requirements. Non-functional requirements specify criteria that can be used to judge the operation of a system, rather than specific behaviors, i.e., non-functional requirements impose constraints on the design or implementation (such as performance requirements, security, or reliability) [Sommerville 2001]. Domain requirements are derived from and reflect the fundamentals of the application domain. Domain requirements vary from being new functional requirements to the software, to being constraints on existing functional requirements [Sommerville 2001].

In the course of defining the requirements to DCMC, we define the functional requirements by means of the presented Agile Requirements Engineering methodologies, i.e., we make use of EFUS. This enables us to adopt the point of view of the user and speak her language. This yields a set of requirements that target audience with non-technical background. In the meantime, we achieve a reasonable technical level of specification that enable us to define and design the authoring process and its sub-processes and the components of the system. We define non-functional requirements following the conventional methodologies. As discussed earlier in this section, many non-functional requirements can be considered as constraints on the behavior of the system [Sommerville 2001; Cohn 2004; Leffingwell 2011]. In contrast, User Stories represent functionality that will be valued by users [Cohn 2004]. Considering these characteristics of User Stories and non-functional requirements, we opted to follow conventional methodologies to define non-functional requirements; as non-functional requirements are not directed to the end users of the system. However, in order not to leave incomplete discussion about non-functional requirements in agile requirements engineering methodologies, we add that non-functional requirements are handled in agile requirements engineering methodologies as constraints on the system as seen in [Cohn 2004; Leffingwell 2011]. However, these approaches recommend to use the user-voice form or the active-voice to define non-functional requirements only when it adds value. Thus, it makes sense not to use user-voice form or the active-voice when they do not add any value. [Leffingwell 2011].

Since our approach targets multimedia content authoring and sharing in PAN, PAN establishment and management is part of the authoring and sharing process. However, networking and personal area networks relate to a different discipline of science rather than the Multimedia Computing. Based on that, we concluded to define two logically separated groups of EFUS to separate between the networking and the multimedia computing. In addition to that, by adopting a grouping approach for the functions of the system we aim at applying the software design concepts of the separation of concerns (SoC) [Parnas 1972]. SoC concept refers to the ability to identify, encapsulate, and manipulate only those parts of software that are relevant to a particular concept, goal, or purpose [Tarr et al. 2001].

3.2.1 Epics, Features, and User Stories (EFUS)

The Epics of DCMC are derived from the different phases involved in the enhanced multimedia content authoring process. The authoring process targets multimedia document authoring and sharing in personal area networks (PAN). PAN represents the computing environment for the authoring process. From the characteristics of PAN, the application scenarios we presented in chapter 1.2, and the overall goals of DCMC, we have concluded two logically separated groups of EFUS for the authoring process in PAN. Group 1: EFUS for PAN establishment and data exchange. Group 2: EFUS for the authoring process.

3.2.1.1 EFUS for personal area network

Personal area network establishment precedes the authoring process of multimedia documents. It provides the data exchange layer to DCMC. Hence, it should facilitate the connection of the devices with the PAN and data exchange between those devices. Based on that, we concluded the following one Epic for the PAN in DCMC:

```
EPIC - E1: PAN management
```

Short description: PAN-enabled devices can form and join a PAN and exchange data to support and facilitate the multimedia content authoring process.

The personal area network profile proposes three PAN scenarios [Bluetooth SIG - PAN Profile 2016] as presented in chapter 2.2.2.1: (i) Network Access Point: in this PAN scenario a Bluetooth equipped device acts as a bridge or a router between a PAN and some other network technology, e.g., GSM. (ii) Group Ad-hoc Network: in this PAN scenario one Bluetooth equipped device acts as a master which communicate between

1 and 7 Bluetooth equipped devices operating as slaves. (iii) PAN User: this PAN scenario realizes a direct peer to peer communication between two Bluetooth-equipped devices. Based on those three PAN scenarios, the scenario presented in (ii) appears to be the most appealing scenario, hence, we concluded that the DCMC approach must provide support for the Group Ad-hoc Network scenario. By considering the selected PAN scenario and by having a deeper look at the defined Epic we detected two aspects that we mapped into Features. Following we present and discuss those two aspects and the resulted Features. These two aspects are concerned with the connection and with the data exchange between devices.

Features:

The first aspect is concerned with PAN formation and management. This yielded the first feature of the system:

E1F1: A PAN-equipped device can establish a connection with other devices

To provide media content to the authoring process. DCMC approach must support the establishment of a PAN to enable PAN-enabled devices to interconnect and exchange data. PAN can be formed using several wireless technologies as presented in chapter 2.2.2. DCMC approach must support the communication between the interconnected devices over a virtual serial port. A serial port is a communication interface that connects two devices, e.g., RS-232 [ARC Electronics 2016]. Serial ports transfer one bit at a time [techterms 2016]. A virtual serial port is a software created serial port that emulates the physical serial port [Bluetooth SIG - SPP Profile 2016]. By using serial port communication, DCMC approach can exchange any type of digital contents.

The second aspect is concerned with data exchange between the interconnected devices. This yielded the second feature of the system:

E1F2: Interconnected devices can exchange data

During the lifecycle of a PAN in DCMC, data exchange takes place in three phases. These three phases can be labeled as (i) *before*, (ii) *during*, and (iii) *after* the authoring process of multimedia documents. The *before* phase is the phase that precedes the authoring process. Interconnected devices exchange data that the authoring process requires to infer the Subject of Interest. Exchanged data in this phase consist of data that belong to the level of metadata, i.e., no media resources are yet exchanged. The *during* phase is the phase that accompany the authoring process. Interconnected devices exchange particular media recourses that the authoring process requires. The *after* phase is the phase that follows the completion of the authoring process.

Interconnected devices exchange the result of the authoring process, that is the composed multimedia document.

Now that we have defined and discussed the Features for PAN in DCMC, next we proceed in decomposing those generated Features to define the User Stories.

User Stories - Feature 1 (E1F1):

```
E1F1: A PAN-equipped device can establish a connection with
      other devices.
```

To enable PAN-equipped devices to form a PAN and communicate efficiently, DCMC approach must provide support for a standard PAN formation process. A standard PAN formation process in its very basic form should at least include the following phases: (i) Scanning phase: in scanning phase PAN-equipped devices scan the surrounding environment to discover other PAN-equipped devices. (ii) Listening phase: in this phase a PAN-equipped device allocates a virtual communication port to listen to incoming connections. (iii) Connection phase: this phase follows the scan phase. Assuming that a PAN-equipped device - called the scanning device - has successfully discovered another device - called the listening device - in the surrounding environment, the scanning device initiates a connection to the listening device. Once the listening device accepts the incoming connection, a communication channel is established and both devices can transfer data. Connection phase must specify how to open, maintain, use, and close the connection. Based on the presented process, we decomposed E1F1 into six User Stories. Following, we present and discuss these User Stories:

```
US1:  As a user I want my PAN-equipped device to scan for
      other PAN-equipped devices so that they can connect to
      each other
```

US1 conforms to the first phase of the PAN formation process we presented earlier in this section. A PAN-equipped device must be able to scan the surrounding sphere to discover other PAN-equipped devices. Once the scanning device has discovered other devices, the scanning device initiates connections to the discovered devices. These discovered devices either accept or reject the incoming connections based on their role in the PAN. Only devices with master role are allowed to accept incoming connections. This implies that devices with slave role must reject incoming connections.

US2: As a user I want my PAN-equipped device to set a listener for incoming connections so that other PAN-equipped devices can connect to it

As concluded earlier in this section, the architectural model of PAN in DCMC approach must support the Group Ad-hoc Network scenario, that is the master-slave model. DCMC approach must define rules for PAN establishment and management to support this kind of architectural models. This implies that these PAN rules must define (i) a mechanism to designate the master role to a device among other devices. (ii) a mechanism for the master device to set a listener to listen to incoming connections. (iii) a mechanism to establish and maintain connection channels between the master and the slave devices. Master role requires that a master device (i) must provide a way for other devices to connect to it, and (ii) manages and maintain the communication with the slave devices. This implies that once a device is granted the master role, it must set a listener to listen to the incoming connections. The master device must keep the listener available for other devices so they can initiate a connection.

US3: As a user I want my PAN-equipped device to initiate or accept the connections so that it can form or join a PAN

To enable a PAN-equipped device to move from the *Scan* phase to the *Connection* phase as defined by the PAN formation process. On the one hand the scanning device must initiate a connection. On the other hand, the listening device must accept the incoming connection. Once both devices have established a connection, this implies that a PAN has been established or joined. Both devices must set a socket for that connection and use it to transfer data.

US4: As a user I want my PAN-equipped device to maintain the connections so that my devices remain connected with the PAN

As discussed in US2 the master role requires that the master device must provide mechanisms to manage and maintain the communication with the slave devices. Connection management and maintenance mechanisms must define: (i) States for the connection, e.g., is connecting, connected, or disconnected. The states of the connection influence the behavior of the master device. (ii) Keep-alive signals to monitor the connection, e.g., heartbeat messages every one second. To lose the keep-alive signals indicates that the connection between devices has been terminated. And (iii) a mechanism to re-establish a lost connection.

US5: As a user I want my PAN-equipped device to use the established connections so that data can be exchanged with other devices

In the connection phase the connected devices must use the connection channels to transfer the data. The data is transferred via a network socket that the connected devices have set after establishing the connection. DCMC approach must provide protocols for the exchange of data via a connection channel. The exchanged data include the multimedia resources and the system messages. We will discuss the data exchange in more details in the next section along with the discussion about feature E1F2.

US6: As a user I want my PAN-equipped device to close a connection so that my device can leave the PAN

PAN formation in DCMC is spontaneous and PAN is characterized by being short-term living networks. This implies that the master device must define how to behave once a connected device closes the connection, i.e., leaves the PAN. DCMC approach must specify a clean process for closing the PAN connection. This connection closing process must ensure that the connection socket is closed, the system resources are released, and the system does not attempt to re-establish the closed connection.

User Stories - Feature 2 (E1F2):

E1F2: Interconnected devices can exchange data

The authoring process requires three types of data that the interconnected devices shall exchange. These are device information, the media resources and their associated meta data, and the system messages that can be exchanged via a request/response model. Further we have decomposed this feature into three User Stories. Following, we present and discuss these User Stories.

US7: As a user I want my PAN-equipped device to access device information so that it can be shared with other devices

DCMC needs to identify the capabilities of the interconnected devices to use device's information in the authoring process. PAN-equipped devices can be collated to several types and classes of devices (e.g., smartphones, laptops, TVs, wearables, audio devices, etc.). Only by providing the correct information about interconnected devices, the authoring process can decide on a suitable role for the device and assign suitable tasks to it. Device profile provides information about the type or the class of the device, screen size of the device if the device is a display device, and about availability of internet connection. The type of device allows the authoring process to decide the most appropriate role that can be assigned to the device during the lifecycle of the authoring process. In addition to that, it allows as well the authoring process to assign the suitable authoring task or tasks to an interconnected device. Screen size influence the dimensions of the final presentation. Availability of internet connection allows the authoring process to assign to an interconnected device tasks that involves retrieving additional media contents and collecting data from the web.

US8: As a user I want my PAN-equipped device to exchange data with other devices so that my PAN-equipped device can access media content provided by other interconnected devices

Exchange of media content and their associated metadata that reside on the devices forming the PAN characterize the DCMC approach. The authoring process in its initial phases needs to access the metadata associated with the media contents. Metadata stem either from the annotations associated with the media elements or can be generated by the authoring process. Metadata allow the authoring process to determine whether to include a certain media element in the final multimedia document or not. Selected media elements can be fetched later in an advanced phase of the phases of the authoring process. Since the selected media elements can possibly be scattered on multiple PAN interconnected devices, DCMC needs to have access on these resources to complete the authoring process. This can be achieved only when DCMC enables the interconnected devices to exchange data.

US9: As a user I want the interconnected devices to exchange messages so that they can guide the authoring process

The DCMC approach must provide a messaging protocol to enable efficient communication between the interconnected devices. This messaging protocol shall define different types of messages to enable an optimized and coordinated authoring process. The messaging protocol shall define the dialect of communication between the interconnected devices. This dialect of communication allows the interconnected devices to interpret the contents of the message and behave accordingly. For example, a message can be: (i) A task that the receiving device must accomplish. (ii) A request to provide some specific media elements. Or (iii) A response form the device to an already received message. The exchanged messages shall influence the behavior and guide the authoring process throughout its lifecycle.

Table 3-2 lists the EFUS for the PAN.

Epics	*Features*	*User Stories*
EPIC 1: PAN management	E1F1: A PAN-equipped device can establish a connection with other devices	US1: As a user I want my PAN-equipped device to scan for other PAN-equipped devices so that they can connect to each other
		US2: As a user I want my PAN-equipped device to set a listener for incoming connections so that other PAN-equipped devices can connect to it
		US3: As a user I want my PAN-equipped device to initiate or accept the connections so that it can form or join a PAN
		US4: As a user I want my PAN-equipped device to maintain the connections so that my devices remain connected with the PAN
		US5: As a user I want my PAN-equipped device to use the established connections so that data can be exchanged with other devices
		US6: As a user I want my PAN-equipped device to close a connection so that my device can leave the PAN
	E1F2: Interconnected devices can exchange data	US7: As a User I want my PAN-equipped device to access device information so that it can be shared with other devices
		US8: As a user I want my PAN-equipped device to exchange data with other devices so that my PAN-equipped devices can access media content provided by other interconnected devices
		US9: As a user I w the interconnected devices to exchange messages so that they can guide the authoring process

Table 3-2: EFUS – personal area network

3.2.1.2 EFUS for multimedia authoring process

In the previous section we introduced and discussed the EFUS for forming PAN and data exchange between the interconnected devices. In this section we introduce and examine the EFUS for multimedia content authoring process. To define the requirements of the authoring process, we follow the same approach as used to define the EFUS for PAN. In chapter 1.1 we have presented and discussed the four constituents of this work: (i) the evolution of smart devices hardware and operating systems. (ii) Users' tendency to join social networks and to share multimedia content and daily life events. (iii) The well-established methods and technologies of the Semantic Web and its applications, e.g., the Linked Open Data (LOD) APIs. And (iv) the networking technologies. Motivated by these four aspects we defined the ultimate goal of DCMC, that is an automatic analysis of multimedia contents, and authoring and sharing of multimedia content driven by the detected Subject of Interest in personal area networks. The ultimate goal of DCMC supported by the four motivating aspects and represented by the automatic authoring process helped us to define the application scenarios presented in chapter 1.2. The deep analysis of the application scenarios induced three main aspects that guide the definition of EFUS in group 2: (i) what is the user's Subject of Interest (SOI); (ii) how coherent can multimedia documents be authored automatically in PAN; (iii) how to share and present the results of the authoring process. The derived three Epics are: (E2) Subject of Interest inference, (E3) Multimedia document authoring, and (E4) Multimedia document presentation.

In the following section, we discuss the Epics and define the Features and User Stories for the authoring process.

EPIC - E2: Subject of Interest inference

Short description: The authoring process analyzes the shared multimedia contents, represents the results by means of Semantic Web standards, and shares the results to enable for the inference of the Subject of Interest.

The Semantic Web standards define technologies on how knowledge can be represented, e.g., RDF [W3C Org. 2014d] and detected by means of querying the data, e.g., SPARQL [W3C Org. 2016c] or reasoning over data through inferencing rules. This implies that DCMC approach can make use of the available W3C standards of Semantic Web [W3C Org. 2016b] to enable the inference of the Subject of Interest. By having a deeper look at *Epic 2* we detected two aspects that we mapped into Features. The first aspect is concerned with the ability of the authoring process to access and share the multimedia content. The second aspect is concerned with the ability of the authoring process to perform the required analysis on the multimedia content. Only by supporting these two aspects DCMC approach can achieve the inference of the Subject of Interest.

Features:

The first aspect is concerned with accessing and sharing multimedia content. This yielded the following feature:

E2F1: Interconnected devices share information about multimedia content

To provide multimedia content to the authoring process. DCMC approach must have a permission to access the multimedia resources that the user is willing to share. Once DCMC approach has located these multimedia resources, it can analyze these multimedia resources and use the result of the analysis in the authoring process.

The second aspect is concerned with the analysis of multimedia content. This yielded the following feature:

E2F2: Analyze the multimedia content

The results of the analysis process provide the required input to the authoring process to enable the inferencing of the Subject of Interest. The analysis process takes place in two separate phases. These two phases can be labeled as (i) *local* and (ii) *global* analysis of multimedia resources. The *local* phase is the phase that precedes the authoring process. DCMC runs the analysis process on the local multimedia resources. Local multimedia resources are the resources that the user is willing to share and exist on her mobile device. The result of the analysis must be persisted and used in the *global* phase. The *global* phase is the phase that accompany the authoring process. Interconnected devices exchange the results of the local analysis. DCMC combines the results of the local analysis provided by the interconnected devices. By combining the local results DCMC can run global analysis on the combined results. Global analysis does not include the multimedia resources but rather their metadata and captured features. This global analysis yields in what we define as the Subject of Interest (see section 3.3.1).

Now that we have defined and discussed the Features for E2, next we proceed in decomposing those generated Features to define the User Stories.

User Stories - Feature 1 (E2F1):

E2F1: Interconnected Devices share information about multimedia content

DCMC approach must have access to the multimedia resources. The user can determine the resources she wants to share. DCMC approach must specify how to locate and access these multimedia resources. Based on that, we detected two aspects for this feature. These two aspects are concerned with (i) proposing multimedia resources to be shared. And (ii) locating and accessing those multimedia resources. These two aspects yielded the following two User Stories:

US10: As a user I want to determine the multimedia resources that I want to share so that I have privacy control over the shared resources

DCMC approach must allow the user to determine which resources the user wants to share. This implies that DCMC approach must provide means to the user to locate, choose, or flag these multimedia resources as shared resources. Multimedia resources can be flagged as shared resources by either adding them to a shared folder or by changing their access properties, e.g., set the system property "is shared" to true. DCMC approach is then able to recognize the shared multimedia resources and use them as input data in the analysis process.

US11: As a user I want the authoring process to have access on the shared media resources on my device so that content of these media resources can be analyzed and shared

To enable DCMC approach to analyze the multimedia resources, the analysis process must be able to locate and access the multimedia resources serving as potential sources for the composition. The analysis process must access only the shared resources. Shared resources can be recognized when they either exist in the shared folder or when they are flagged as shared.

User Stories – Feature 2 (E2F2):

E2F2: Analyze the multimedia content

In the course of defining this system feature, we have introduced two scopes for the analysis of multimedia resources, a *local* scope and a *global* one. The *local* multimedia resources analysis focuses on analyzing the individual media elements. The analysis process must support two types of analysis. High-level semantic analysis and low-level feature analysis. High-level semantic analysis targets the information about a media element represented by means of metadata, the context of the media element, and spatial, temporal, and interactional features of that media element. Low-level feature analysis aims at exploring hidden information about a media element that adds value to the authoring process. The *global* analysis focuses on analyzing the results of the local analysis. This implies that the results of the *local* analysis provided by the interconnected devices represent the input for the *global* analysis process. *Global* analysis aims at exploring one or more common Subjects of Interest. The Subject of Interest can be inferred by means of Semantic Web technologies. Based on this discussion about this system feature, we decomposed it to the following User Stories:

US12: As a user I want the authoring process to perform high-level semantic analysis on my contents so that only relevant media elements can be included in the multimedia document

The analysis process in DCMC approach must consider several aspects to achieve a complete analysis process that supports the concepts of the multimedia documents authoring in DCMC. The analysis process must capture the semantics related to the source document, and the individual media elements included in these documents. These semantics include the metadata that annotate the source document and its individual media elements, and the composition semantics of the individual media elements. The composition semantics are the attributes that define the temporal, spatial, and interactional characteristics of the individual media elements.

US13: As a user I want the authoring process to perform low-level feature analysis on my content so that a variety of implicit features of the media elements are detected

To detect the hidden features of the individual media elements, the analysis process must perform a low-level feature analysis to capture additional information about the individual media elements, e.g., color histograms. The results of the low-level feature analysis must be stored with and linked to the analyzed media element.

US14: As a user I want the authoring process to represent the result of the analysis by means of Semantic Web technologies so that the results can be easily processed

DCMC approach aims at using the well-defined standards of W3C Semantic Web. This implies that DCMC approach must encode the results of the analysis by means of Semantic Web standards, e.g., RDF [W3C Org. 2014e] to be able to process these semantics. Using those standards allows DCMC to reason over data by means of logic-based dialects to infer the shared Subject of Interest.

US15: As a user I want the authoring process to share the result of the analysis so that SOI can be detected

The results of the *local* analysis performed by an individual device must be made available for the analysis process at the global level, i.e., the *global* analysis process performed by the interconnected devices to define one or more Subjects of Interest. The pieces of *local* analysis results must be combined into one data repository. The *global* analysis process applies a set of rules to reason over the data. The reasoning process aims at exploring new relationships, i.e., the Subject of Interest.

Now that we have covered E2, we present E3, its Features, and User Stories. E3 handles the authoring process.

EPIC -E3: Multimedia document authoring

Short description: The authoring process selects media resources, retrieves relevant media elements from web resources, and arranges these resources in a logical order into a multimedia document.

The authoring process of multimedia documents consists of selecting the media elements and organizing them in spatial, temporal, and interactional relationships. The selection phase in the authoring process in DCMC approach defines two steps: (i) selecting media elements from the media resources provided by the interconnected devices. (ii) retrieving new media elements from external sources, e.g., LOD servers. Organizing the selected media elements in spatial, temporal, and interactional relationships in DCMC follows two approaches, that are (i) template-based approach and (ii) free-style composition approach. Template-based approach implies that the authoring process selects one of the multimedia document templates that best suites the selected media elements. Free-style approach implies the authoring process organizes the selected media elements in the multimedia document considering their original spatial, temporal, and interactional features detected from the source multimedia document. Based on the discussion about E3 we detected two aspects that we mapped into Features. The first aspect is concerned with: (i) Selecting the relevant media elements among the available media resources. And (ii) retrieving additional relevant media resources making use of LOD APIs. The second aspect is concerned with the ability of the authoring process to arrange the selected media elements into coherent multimedia document.

Features:

The first aspect is concerned with selecting the relevant media elements among the available media resources. This yielded the following Features:

E3F1: Use of available resources to select media elements

The selection process provides media elements to the authoring process. The selection process must select only those media elements relevant to the Subject of Interest. This means that the selection process must provide various selection techniques to enable selecting the most relevant media elements. The available media resources are the media resources provided by the interconnected devices and the media elements provided by LOD servers. Retrieving additional relevant media elements to the Subject of Interest from external resources, e.g., LOD servers characterize the DCMC approach. The Semantic Web technologies present standards for the Linked Data [W3C Org. 2016a]. Linked Data standards define how the relationships between various data can be created, e.g., RDF [W3C Org. 2014e], and how the hidden information in those relationships between the pieces of data can be uncovered, e.g., SPARQL [W3C Org. 2016c]. By adding support to make use of Linked Data standards, DCMC approach can enrich the authoring process with additional data. Once the selection process has selected the relevant media elements, the selected media elements can be further used in the authoring process.

The second aspect is concerned with the ability of the authoring process to arrange the selected media elements into a coherent multimedia document. This yielded the following feature:

E3F2: Organize the selected media elements into a coherent multimedia document

In order to make use of the selected media elements these selected media elements must be organized into a coherent multimedia document. As presented in section 3.1.2, a multimedia document consists of two parts, the document model and the presentation format. This feature in DCMC targets the model of the document. DCMC makes use of ZYX model [S. Boll and W. Klas 2001] to provide an abstract document model for the authoring process. By using an abstract document model, the authoring process preserves the presentation format neutrality of DCMC.

Now that we have defined and discussed the Features for E3, next we proceed in decomposing those generated Features to define the User Stories.

User Stories - Feature 1 (E3F1):

E3F1: Use of available resources to select media elements

DCMC provides two resources for selecting media elements. Media elements provided by the interconnected devices, and media elements provided by LOD servers. The authoring process must collect the media elements provided by the interconnected devices in one media repository. In DCMC this media repository is called the media pool. This media pool does not contain the real media elements, but rather it contains references to the media elements and all the extracted information about those elements. Additional media elements that are considered to be relevant to the Subject of Interest can be retrieved form LOD servers. This implies that DCMC must provide support for the standards of Linked Data [W3C Org. 2016a] provided by the Semantic Web standards [W3C Org. 2016b]. Retrieved media elements from LOD servers must be properly added to the media pool. Based on this analysis we decomposed E3F1 into the following three User Stories:

```
US16: As a user I want the authoring process to collect the
      media elements provided by the interconnected devices in
      a single media pool so that the authoring process is
      enabled to apply an efficient selection process
```

As presented earlier in this chapter, Group Ad-hoc PAN encompasses 2 to 8 devices. This means that the media elements which are available to the authoring process are scattered over the interconnected devices. To enable an efficient selection process DCMC approach must provide means to deal centrally with the available media elements. This implies that DCMC approach must provide a central repository to collect these media elements and their metadata. The central repository perceives the required information by the authoring process about the available media elements.

```
US17: As a user I want the authoring process to select media
      elements using rule-based methods and Semantic Web
      technologies so that only media resources relevant to a
      given SOI are selected
```

The information about the media elements must be encoded and presented by means of Semantic Web technologies, e.g., in RDF datasets. Using RDF to represent the information about media elements provided by the interconnected devices enables to apply the reasoning techniques provided by the Semantic Web technologies to discover the hidden relationships between those media elements. The media elements vary in their relevance to the Subject of Interest. This implies that the selection process must choose only those resources that fall within a certain threshold. This threshold is determined by the relevance of the media element to a Subject of Interest.

```
US18: As a user I want the authoring process to access the
      Linked Open Data APIs so that the authored multimedia
      document is enriched with additional data
```

DCMC approach aims at making use of the resources provided by the web. Linked Open Data servers provide information about a massive variety of resources. The added value of these materials can be realized only when they are discovered, consumed and linked to other resources. This implies that the DCMC approach must provide means to communicate with LOD servers to discover additional information or media elements about the Subject of Interest. This enables the authoring process to enrich the authored multimedia document with additional resources. In the case where the Subject of Interest takes the form of terms, e.g., "coffee", "Jazz", etc. there is a high probability that those terms lead to ambiguous Subject of Interest. This implies that the authoring process must define rules to disambiguate these terms.

User Stories - Feature 2 (E3F2):

E3F2: Organize the selected media elements into a coherent multimedia document

DCMC approach aims at composing a multimedia document about the user's Subject of Interest. Multimedia document composition can be described as organizing the selected media elements into spatial, temporal, and interactional relationships. The selection process provides the media elements linked to the Subject of Interest. Those selected media elements must be presented in a meaningful way to the user. This implies that DCMC approach must provide means to organize the selected media elements in a logically correct order. Based on that, on the one hand the authoring process must fulfill the basic requirements of multimedia documents presented in section 3.1.3. On the other hand, the authoring process must preserve the meanings of and the relationships between the selected media elements. The authoring process makes use of the detected composition features of the selected elements. These features encode the attributes of the temporal, spatial, and interactional relationships of the media elements in their original multimedia document. However, DCMC approach must provide a fallback solution in case those provided features were insufficient to conclude the structure of the multimedia document under composition. Based on this analysis of the second feature we decomposed it into two User Stories as follows:

US19: As a user I want the authoring process to arrange the media elements in a coherent multimedia document so that well-recognized, basic requirements given by the target document model are fulfilled

Multimedia documents are characterized by their ability to present the temporal, spatial, and interactional relationships between the individual media elements. Media elements can be of any type of digital contents, e.g., Test, Image, Video, etc. DCMC approach must preserve these basic characteristics of multimedia documents. This implies that the authored multimedia documents must fulfill the basic requirements of the multimedia documents. We refer the reader to section 3.1.3 for additional discussion about these basic requirements.

US20: As a user I want the authoring process to allow template-driven and free-style-driven authoring so that the system produces meaningful multimedia presentations

The Analysis process as discussed in E2 captures the high-level semantics as well as the low-level features of the individual media elements. The high-level semantic features of the media element provide information to the authoring process about the composition features of the media element. These composition features encode information about the media element such as its dimensions, its spatial position in relation to other media elements or in the source multimedia document, etc. DCMC approach aims at an automatic authoring process of multimedia documents. This implies that the authoring process must make use of these composition features to organize the media elements in a logically correct position. In the cases where the detected composition features lack the required information by the authoring process to determine the position of the media element, DCMC approach must provide a fallback solution. This fallback solution is represented by enabling the authoring process to use either a free-style composition or a template-based composition. The free-style composition of multimedia documents implies that the authoring process constructs a new multimedia document. The authoring process detects the positions of the selected media elements form the captured composition features. Based on these detected composition features the authoring process determines the positions of the selected media elements in the new multimedia document. Template-based composition implies that the authoring process selects and applies a pre-composed multimedia document template on the selected media elements. The templates dictate the positions of the selected media elements in the composed multimedia document.

The last Epic (E4) handles the presentation of the multimedia document, in the following we present E4, its Features, and User Stories.

EPIC - E4: Multimedia document presentation

Short description: DCMC shares the results of the authoring process with the interconnected devices to be presented to the user.

Presenting the authored multimedia document is the final phase of the lifecycle of the authoring process. Only by sharing and presenting the outcome of the authoring process, DCMC can demonstrate the added value of all the work performed in the preceding phases. The presentation phase encompasses multiple steps. As discussed earlier in this chapter, DCMC separates between the model of the document and the presentation format. In the sharing step the composed multimedia content represented by an abstract document model, e.g., ZYX model [S. Boll and W. Klas 2001], must be communicated to the interconnected devices. The sharing step is followed by transformation step. Based on the capabilities of each interconnected device, each device in the transformation step converts the received multimedia document model to a presentation format that it supports. The transformation step results in a presentation ready multimedia document. In the last step, the presentation step, the user is notified about the new authored multimedia document and can view it. Based on the presented steps, we detected two aspects that we mapped into Features.

Features:

The first aspect is concerned with sharing the results of the authoring process. This yielded the following feature:

E4F1: Share the authored multimedia document

Sharing the results of the authoring process paves the way for presenting the new multimedia document to the user. By sharing the results of the authoring process, the interconnected devices can proceed in preparing the authored multimedia content to be presented. This preparation step encompasses the collection of the individual media elements that the authoring process has chosen to be included in the multimedia document. Once the authoring process has collected these media elements, the transformation step can be triggered.

The second aspect is concerned with the ability of the authoring process to transform the model of the composed multimedia document to the correct presentation format, and to notify the user about the new multimedia document. This yielded the following feature:

E4F2: Transform and present the multimedia document

Reaching the presentation phase implies that the authoring process must perform one last step before the authored multimedia document can be rendered and presented, and the user is notified about the new content. In this last step, the authored multimedia document is transformed to a concrete presentation format. The individual media elements must be transformed and adapted to conform with the capabilities of the presentation device. As discussed earlier in this section, the authoring process represents the authored multimedia document by means of an abstract document model. Thus, DCMC approach must support the transformation of the abstract multimedia document model to a concrete presentation format.

Now that we have defined and discussed the Features for E4, next we proceed in decomposing those generated Features to define the User Stories.

User Stories - Feature 1 (E4F1):

```
E4F1: Share the authored multimedia document
```

DCMC approach must define how the results of the authoring process can be shared. When presenting the communication protocols between the interconnected devices (see section 3.2.1.1), we discussed the types of the data messages that the interconnected devices can exchange. When sharing the results of authoring process, the interconnected devices must recognize that the received data represent the result of the authoring process. Consequently, since the individual media elements are distributed over multiple devices, the interconnected devices must be able to collect those media elements to be rendered with the final multimedia presentation. Based on that, we concluded the following User Stories:

```
US21: As a user I want the authoring process to share the
      authored multimedia document so that I can play it back
```

DCMC approach must specify how the results of the authoring process can be shared. This implies that the messaging system we defined earlier in this chapter must provide mechanisms on how to communicate the results of the authoring process and the selected media elements between the interconnected devices.

```
US22: As a user I want the authoring process to collect the
      individual media elements so that these media elements
      can be rendered in the final multimedia presentation
```

As discussed earlier in this chapter (see section 3.2.1.2) DCMC approach collects references to the media elements in the media pool. The authoring process uses these references in the composition phase. This implies that at some point the authoring process must collect these media elements to enable rendering them during the presentation. DCMC approach must define mechanisms within the messaging protocol on how to collect these media elements from the interconnected devices.

User Stories - Feature 2 (E4F2):

E4F2: Transform and Present the multimedia document

In order to be able to present the authored multimedia document to the user, the authored document represented by an abstract document model must be transformed to a concrete presentation format, e.g., SMIL [W3C Org. 2014f] or HTML5 [W3C Org. 2014b]. This implies that DCMC approach must provide means to transform (i) the individual media elements, and (ii) the multimedia model to a presentation format that the playback device supports. The presentation of the new multimedia document requires user interaction. That is, the user can at least choose when to view the new multimedia document. This implies the DCMC approach must specify how to notify the user about the new multimedia document being available for playback. Based on that, we decomposed this feature into the following User Stories:

US23: As a user I want the authoring process to adapt the media elements so that they conform to the capabilities of my device

The individual media elements of different types such as images, videos, or audios can have different formats, e.g., an image can be formatted as JPG or PNG, etc. These media elements can have as well different quality features such as size and resolution. DCMC approach must provide mechanisms to adapt those individual media elements' features to conform with capabilities of the presentation device.

US24: As a user I want to transform the multimedia document to a device-specific presentation so that it can be played back on the device

Group Ad-hoc PAN can be formed by different types of mobile devices. Mobile devices have different presentation capabilities, i.e., different audio and video decoders, different screen sizes and resolutions, and different output hardware (e.g., a screen or a speaker). DCMC approach must consider such diversity in devices' capabilities and grant the presentation device the freedom to playback to multimedia document according to the capabilities of the presentation device.

US25: As a user I want the authoring process to notify me about an available multimedia presentation so that I am aware of it

DCMC approach aims at automatic multimedia document authoring and sharing. The automatic authoring process does not require any user interaction. However, the presentation of the authored multimedia document does. It implies that DCMC approach must specify a notification system. Once the authoring process has completed the authoring of a new multimedia document, the user must be notified. The user can playback the document at any chosen time.

Table 3-3 summarizes the EFUS for the multimedia authoring process:

Epics	Features	User Stories
E2: Subject of Interest inference	E2F1: Interconnected devices share information about multimedia content	US10: As a user I want to determine the multimedia resources that I want to share so that I have privacy control over the shared resources
		US11: As a user I want the authoring process to have access on the shared media resources on my device so that content of these media resources can be analyzed and shared
	E2F2: Analyze the multimedia content	US12: As a user I want the authoring process to perform high-level semantic analysis on my contents so that only relevant media elements can be included in the multimedia document
		US13: As a user I want the authoring process to perform low-level feature analysis on my content so that a variety of implicit features of the media elements are detected
		US14: As a user I want the authoring process to represent the result of the analysis by means of Semantic Web technologies so that the results can be easily processed
		US15: As a user I want the authoring process to share the result of the analysis so that SOI can be detected
E3: Multimedia document authoring	E3F1: Use of available resources to select media elements	US16: As a user I want the authoring process to collect the media elements provided by the interconnected devices in a single media pool so that the authoring process is enabled to maintain an efficient selection process
		US17: As a user I want the authoring process to select media elements using rule-based methods and Semantic Web technologies so that only media resources relevant to a given SOI are selected
		US18: As a user I want the authoring process to access the Linked Open Data APIs so that the

		authored multimedia document is enriched with additional data
	E3F2: Organize the selected media elements into a coherent multimedia document	US19: As a user I want the authoring process to arrange the media elements in a coherent multimedia document so that well-recognized, basic requirements given by the target document model are fulfilled
		US20: as a user I want the authoring process to allow template-driven and free-style-driven authoring so that the system produces meaningful multimedia presentations
E4: Multimedia document presentation	E4F1: Share the authored multimedia document	US21: As a user I want the authoring process to share the authored multimedia document so that I can play it back
		US22: As a user I want the authoring process to collect the individual media elements so that these media elements can be rendered in the final multimedia presentation
	E4F2: Transform and Present the multimedia document	US23: As a user I want the authoring process to adapt the media elements so that they conform to the capabilities of my device
		US24: As a user I want to transform the multimedia document to a device-specific presentation so that it can be played back on the device
		US25: As a user I want the authoring process to notify me about an available multimedia presentation so that I am aware of it

Table 3-3: EFUS - Multimedia Authoring Process

3.2.2 Non-functional Requirements

In the previous section we have defined the functional requirements of DCMC by means of agile Software Engineering methodologies, i.e., we used Epics, Features and User Stories (EFUS). In the course of introducing the EFUS, we have discussed about the feasibility of using EFUS to define the non-functional requirements of the system. As pointed out, it is recommended to use EFUS only when they add value to the requirements [Cohn 2004; Leffingwell 2011]. By examining the characteristics of the non-functional requirements in DCMC, we decided to follow traditional methodologies to define the non-functional requirements for the DCMC. In traditional software design and development methodologies, e.g., the waterfall model, requirements of the system are defined in the analysis phase [Hoffer et al. 2002] or in the requirements definitions phase [Sommerville 2001]. To determine the requirements, several methodologies can be used. This includes: (i) interviewing and listening, (ii) questionnaires, (iii) prototyping and other methods [Hoffer et al. 2002]. Traditional methodologies use the requirement statements to define the requirements of the system. Requirement statements consist of two parts: (i) subject such as a stakeholder or an actor, and (ii) a predicate that specifies a condition or an action [Zuby Ezeasor, Martin Gorm Pedersen 2013]. Besides well-established Software Engineering requirements, e.g., the FURPS model [Grady 1992; Leffingwell 2011], we derived from the characteristics of the DCMC approach and the given scenarios additional four aspects that our approach must consider. These are concerned with performance requirements, using the Semantic Web technologies, supporting the standard presentation formats, and the multimedia document model independent presentations.

3.2.2.1 Software Engineering requirements (NFR1)

Software Engineering requirements address variety of emerging software needs. DCMC approach must consider these fundamental software needs. The acronym FURPS [Grady 1992; Leffingwell 2011] represents part of these needs. FURPS stands for functionality, usability, reliability, performance, and supportability. The list can be extended with additional aspects such as portability, reusability, security [Cohn 2004] and scalability. A computing platform with multiple services enables a form of modularity called service-oriented architecture (SOA) [Shaw 2011]. By adopting a service-oriented architecture, DCMC approach aims at incorporating these emerging software requirements. A service-oriented architecture enables to decompose the application into *N* groups of services. Each service can be defined at a specific level of abstraction with well-defined interfaces. Hence, a service-oriented architecture enables flexibility and better performance in the applications where performance is critical. Modularity means grouping the functionality of the application into independent and interchangeable modules. A modular software architecture: (i) enables to reuse the design knowledge and make the system understandable by others, (ii) allows for segmentation of work, i.e., assignments of tasks and responsibilities, and (iii) allows for the substitution of new modules can substitute for older ones without compromising the harmony of the system [Shaw 2011].

3.2.2.2 Performance requirements (NFR2)

In DCMC, multimedia document authoring is accomplished on-the-fly in a PAN environment. In addition to the short lifecycle of PAN, a PAN environment encompasses a short-ranged low-powered wireless network to transmit data between interconnected devices with potentially limited resources. This implies that the authoring process and all its sub-processes including the exchange of data between the interconnected devices must be designed in terms of efficient performance. Thus, DCMC must consider the bandwidth of the network and the characteristics of the interconnected devices.

3.2.2.3 Make us of Semantic Web technologies (NFR3)

This non-functional requirement is derived from the overall concept of the DCMC and the aspects motivating this work. In the era of cloud computing and Service-Oriented systems, applications must be designed to cope with the emerging concepts of the Internet of Things (IoT). For DCMC this means that the authoring process must make use and incorporate the Semantic Web technologies in its phases and where it is applicable. On the one hand, this constraint on the authoring process urges the use of technologies such as RDF, linked data, reasoning rules and other W3C Semantic Web standards. On the other hand, DCMC can be easily integrated with other systems that adopt the same W3C technologies and standards. This implies as well, that DCMC supports scalability, availability, and interoperability.

3.2.2.4 Support to standard presentation formats (NFR4)

To respect the idiomatic metaphor "don't reinvent the wheel", DCMC approach must avoid defining a new multimedia presentation format. For the following reasons, the authoring process must make use of available presentation formats: (i) DCMC approach can use the presentation players available on the market, this implies that the authoring process can serve the authored multimedia document to a wide range of devices. (ii) Defining a new presentation format contradicts with requirement of the ability of DCMC approach to integrate with other similar systems. (iii) There are no emerging needs that urge creating a new presentation format. (iv) Developing a new presentation format implies that we must undertake the burden of the development of a new player, which in any case will not be feasible. Based on that, DCMC approach must use W3C standard formats such as SMIL, HTML5 and SVG, as well as the legacy industry-standard Flash.

3.2.2.5 Presentation independent multimedia document model (NFR5)

This is a centric non-functional requirement in DCMC approach. It imposes a constraint on the authoring process and its related functional requirements, which we already defined and discussed in the previous section. DCMC must ensure a presentation independent authoring process. This implies that authored multimedia document can be transformed to the presentation format that the target device supports. This requires that DCMC approach must make use of an abstract multimedia document model. In this work, we have concluded to use ZYX model [S. Boll and W. Klas 2001] to provide the internal abstract model for the authoring process.

3.2.3 Summary

In this section, we identified the requirements of the DCMC approach. We explained the methodologies that we followed to come up with the functional and non-functional requirements. Those requirements are generated from the identified gap between the related work analyzed in chapter 2.4 and the concepts and the ultimate goals of DCMC approach. We concluded that there is no existing solution provided by research approaches nor in the market that serves all the requirements that we have identified. In particular, there is no all-embracing research approach like DCMC approach that integrates all the concepts we discussed in the previous sections in one system. Based on those requirements, we present in the following section the enhanced authoring process of DCMC approach.

3.3 Authoring process

The DCMC aims at the composition of multimedia content based on knowledge explored and extracted from content individual users have and are willing to share and which is encoded in different kinds of multimedia models. As discussed in previous sections, our approach is motivated and influenced by four aspects. These four aspects are: (i) the evolution of smart device's hardware and operating systems, (ii) user's tendency to join social networks to share multimedia content about daily life events, (iii) well-established methods and technologies of Semantic Web including the increasing establishment of Linked Open Data (LOD) APIs, and (iv) the networking technologies. The ultimate goal of DCMC and the presented motivation provided input to identify Epics, Features, and User Stories (EFUS) for DCMC. These identified EFUS provide the required material that we need to define an enhanced authoring process and its sub-processes. Hence, the enhanced authoring process must reflect the presented concepts and the identified requirements. In this section, we discuss and define the concept of the Subject of Interest, we present briefly the general multimedia content authoring process, and we define the enhanced authoring process of DCMC and present its phases and sub-processes.

3.3.1 The Subject of Interest (SOI)

We have used the term Subject of Interest (SOI) in many places and occasions throughout this work. In this section, we discuss the thoughts and the inspiring concepts behind the SOI term.

In the domain of Location-Based Services (LBS), the term "Point of Interest" (POI) is used to refer to any object that the user is interested in and has a static geographical location. A geographical location is usually determined by its topographical coordinates that are typically expressed by means of longitude and latitude. Examples of POIs are museums, statues, bus stops, etc. A location can be of interest to the user depending on the next activity the user is willing to undertake in a certain point of time, i.e., if the user is willing to go for a coffee break, café houses nearby are the POIs of that user at that particular moment. Afterwards, if the same user intends to have another activity after the coffee break, her set of POIs will be changed accordingly. That is, the new set of POIs is where the user can perform her next activity. From this short analysis of POIs, we can say that the POI concept covers one aspect of the aspects that can be linked to the interests a user has. This aspect is mainly linked with a geographical location, i.e., the location where the user can perform her future activities, i.e., her interests.

The Resource Description Framework (RDF) [W3C Org. 2014e] uses statements to interrelate things. RDF statements consist of three parts, the Subject, the Predicate and the Object. The Subject and the Object represent the two resources being linked; the Predicate represents the nature of their relationship. The relationship is phrased in a directional way (from Subject to Object) and is called in RDF a Property [W3C Org. 2014d], e.g., <Bob> <is interested in> <a Cup of Coffee>. In this statement "Bob" is the Subject, "is interested in" is the Predicate, and "a Cup of Coffee" is the Object. In the RDF framework the Subject of a statement can be the Object in another statement, and vice versa, the Object in one statement, can be the Subject in another statement. However, since the relation in RDF is uni-directional, i.e., from Subject to Object, the Subject is what we aim at inferring information about. Figure 3-2 visualizes the basic RDF statement as presented in the previous example. The RDF statement depicted in Figure 3-2 suggests that we can infer information about the Subject "Bob" however no additional information about the Object "Cup of Coffee" can be inferred. By adding a second statement, e.g., <a Cup of Coffee> <is prepared from> <Coffea Arabic>, we can infer additional information about the Subject "a cup of coffee". Obviously, the difference between the two statements is that, the Object of the first statement became the Subject in the second statement. Hence, we are able to infer additional data about "a cup of coffee".

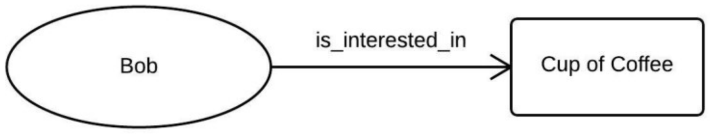


Figure 3-2: Basic representation of RDF triple

Based on the above discussion, we conclude that the Subject in RDF statements represents a central concept. This Subject can represent both tangible and intangible things. However, the POI represents a concept that can be linked to tangible things only, i.e., to geographical places. Nevertheless, these tangible things represent part of the interests of the user that DCMC aims to find by analyzing the content, which the user provides. When combining the two concepts, which the POIs and RDF triples represent, the result is characterized by attaching the Subject of the RDF statement to the Interests of the user, resulting in what we define as the Subject of Interest (SOI).

The SOI provides a central concept to DCMC, it consists of the following main features: (i) It inherits the features and attributes of the POIs and the Subject of the RDF statement. (ii) It represents the things that matter to the user. (iii) It covers a wider range of tangible and intangible things, i.e., it can be a geographical location, an abstract concept or an idea. (iv) It is not only linked to the future activities of the user, the SOI can be linked to the past, the present, and the future.

3.3.2 General Multimedia Document Authoring Process

The general process of creating and personalizing multimedia documents presented in chapter 2.5 consists of four phases: Phase 1 is about the selection of the media assets such as images, text, video, and audio for the composition. Phase 2 is about the assembly of the selected media assets in time, space, and by means of user interaction elements and its representation in a coherent, structured, abstract multimedia document. Phase 3 is about transforming the abstract multimedia document to a concrete multimedia format. Phase 4 is about presenting the transformed multimedia document which includes delivery, rendering, and viewing of the multimedia document. This general process of creating and personalizing multimedia documents can serve DCMC as an appropriate reference model, because it is very generic and intuitive, thus, the enhanced authoring process of DCMC builds on top of this reference process.

3.3.3 Enhanced Authoring process of DCMC

The enhanced multimedia document authoring process depicted in Figure 3-3 is derived from the requirements, which we have identified in section 3.2. The core of multimedia authoring process in DCMC follows the reference process introduced in section 3.3.2. DCMC extends that reference process and adds additional phases. These phases are guided by the identified requirements, which yielded an authoring process consisting of eight phases:

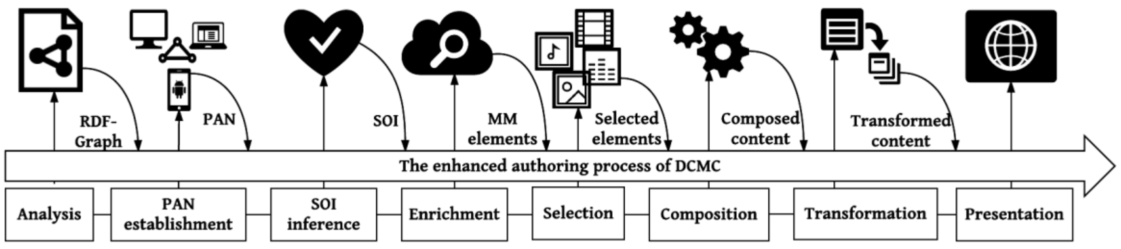


Figure 3-3: The enhanced authoring process of DCMC approach

Phase 1 - *Analysis*:

Short description: the multimedia documents that the users share are analyzed individually and the results are encoded in RDF.

In the course of defining the term SOI, we stated that the Resource Description Framework (RDF) [W3C Org. 2014e] uses statements to interrelate things. The main task of the analysis process is inspired by this basic function of RDF. This implies that the main task of the analysis process must achieve: (i) interrelating the document with its metadata, composition features, and the included media elements. (ii) interrelating all the multimedia documents to the user or the user's device. The final result will be the dataset that provides the substance to the authoring process.

The analysis process starts by locating all the multimedia documents that the user shares. For each document two types of analysis can be performed, global and local analysis. The global analysis captures the semantics of the multimedia document by means of interpreting: (i) the metadata of the document, (ii) the composition features and the relationships between the individual media elements included in the multimedia document. The local analysis focuses on the individual media elements. The main function of the local analysis is to detect the high-level features represented by the metadata as well as the low-level features represented by physical attributes of the media elements.

The analysis process performs the global and local analysis tasks on each multimedia document that the user shares. The final results of the analysis process form the main dataset that the authoring process require. Figure 3-4 depicts that analysis process.

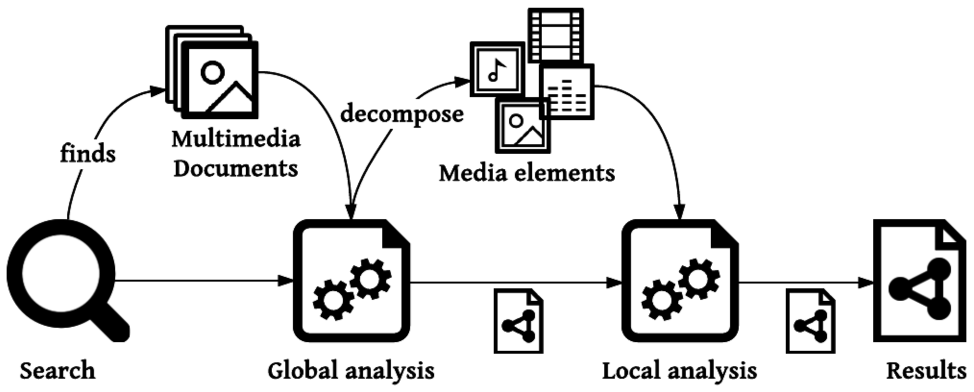


Figure 3-4: The analysis process

We make use of RDF to encode the datasets, since RDF enables us to make use of the Semantic Web technologies intended to extract knowledge from RDF-Graphs.

Phase 2 - *PAN establishment*:

Short description: a personal area network is established by interconnecting two or more devices.

The PAN establishment phase aims at forming the networking environment to enable the interconnected devices to exchange data. Guided by the defined requirements of PAN and the DCMC, the PAN forming process targets the Group Ad-hoc network. Group Ad-hoc networks adopt the master-slave architecture and its related specification. It specifies that 1 to 7 slave-devices can be connected to a master device. PAN formation process starts once a PAN-equipped device searches for another PAN-equipped device. Two possible scenarios for the result of the search can be anticipated: (i) The searching device does not find any other devices. In this case, the searching device shifts to listening mode and waits for incoming connections. This implies as well that the listening device is potentially the master of the PAN. (ii) The searching device discovers one or more PAN-equipped devices. In this case, the searching device initiates one or more connections. Only a device in the listening mode can accept incoming connections. Once the connection is established, the PAN is formed.

Phase 3 - *SOI inference*:

Short description: one or more SOI are inferred from the multimedia content represented by the datasets.

Once the PAN has been formed, slave devices share with the master device the datasets that resulted from the analysis process. The master device collects these datasets and merges them into one master dataset. The master dataset is the input data source to the SOI-inferencing process. This SOI-inferencing process makes use of the inferencing rules that the DCMC provides for processing the master dataset. The result of the reasoning process is one or more SOI. Figure 3-5 depicts the SOI inferencing process. As mentioned in **Phase 1** above, we make use of RDF of encode these datasets.

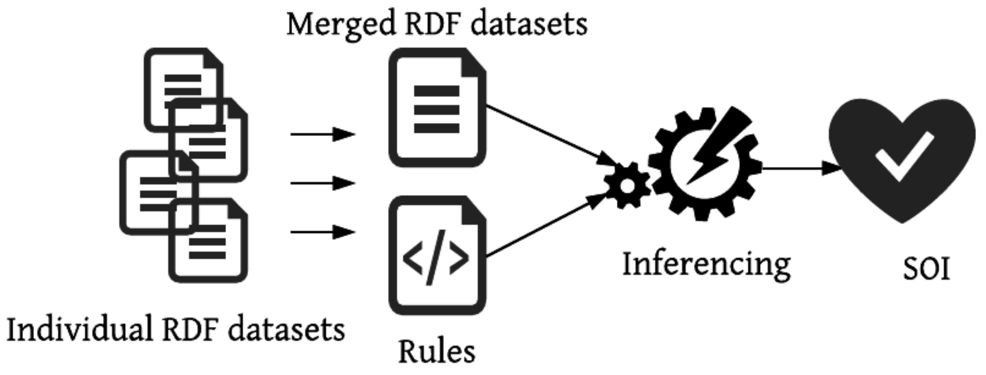


Figure 3-5: SOI-inferencing process

Phase 4- *Enrichment*:

Short description: additional information about the SOI are retrieved from LOD sources.

Having identified one or more SOI in the *SOI inference* phase, the authoring process communicates with LOD servers to retrieve additional resources about the discovered SOI. The specifications of the Linked Data [W3C Org. 2016a] define how to represent these resources by means of RDF triples. This implies that the results of the enrichment process represented by the retrieved resources can be merged into the master dataset and linked to the SOI. Figure 3-6 illustrates the enrichment process. The retrieved resources, which are represented in RDF triples, provide additional information about the SOI. The additional information can be of any type of the known media types, i.e., image, text, video, etc.

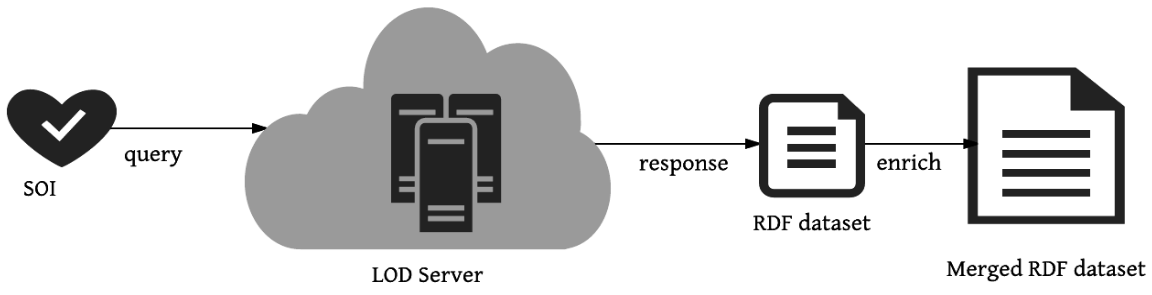


Figure 3-6: The enrichment process

Phase 5 - *Selection*:

Short description: media elements related to the SOI are selected for the inclusion in the new multimedia document to be composed.

The SOI is detected from the information that the multiple devices have provided. This implies that: (i) the related media resources that are linked to the SOI are scattered over several devices, (ii) the strength of the relation between the SOI and the linked media elements can vary. The selection process starts by collecting the individual media elements from various devices in the media pool. The media pool encompasses links to the source elements, the detected information about these elements, and resources that the enrichment process has collected. Once these resources become available to the selection process, the selection process chooses from these resources by applying the selection rules that DCMC offers for this phase. Figure 3-7 depicts the different steps of the selection process.

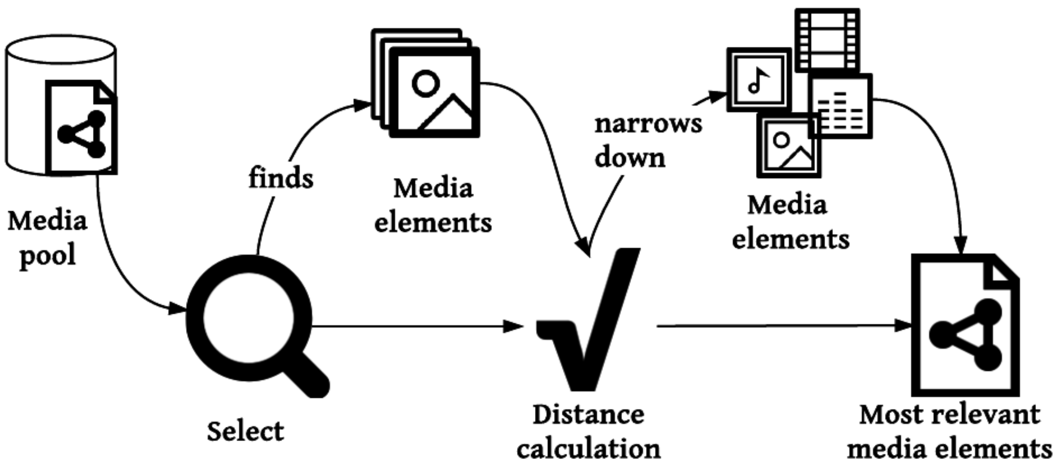


Figure 3-7: The selection process

Phase 6 - *Composition*:

Short description: organizing the selected media elements into a coherent multimedia document by determining the spatial and temporal relationships.

The selection process provides the media elements that the composition process can make use of. These media elements stem from two different resources: (i) Elements from PAN resources; these are media elements that the analysis process extracted from the shared multimedia documents. (ii) Elements from LOD resources; these are media elements that the enriching process has retrieved from the LOD servers. Since PAN elements are extracted from the shared multimedia documents, additional data about their composition features are available to the composition process, i.e., the attributes of their temporal and spatial spaces. This information supports the authoring process in arranging these elements into a new coherent multimedia document.

Since elements retrieved from LOD sources lack composition features, i.e., the spatial and temporal attributes, this implies that the composition process must make use of the rules that DCMC provides. These rules guide the composition process to infer composition features for these elements.

Having provided this input information about the composition process, we can have a closer look at its steps. Similar to the selection process, we apply a set of composition rules on the selected media elements. The first step is to determine the level of complexity of the new multimedia document. The level of complexity of the document ranges between basic multimedia documents and highly sophisticated fancy documents. In this step, the authoring process chooses between the template-based composition and the free-style composition. Once the composition process has defined the level of the complexity, a set of composition rules can be applied on the selected media elements to infer their temporal and spatial attributes in the new document. Finally, the results of the composition process are encoded in RDF.

Phase 7 - *Transformation*:

Short description: transform the composed multimedia document that is described in terms of RDF triples to the internal abstract multimedia document model.

Once the composition process has provided the composed multimedia document that is represented in RDF triples, the transformation process can convert those RDF triples to their equivalent representations in the multimedia model. To illustrate the process, RDF datasets can be visualized as a tree. Figure 3-8 depicts an RDF dataset and its visualization as a tree, and the layout that this RDF dataset represents.

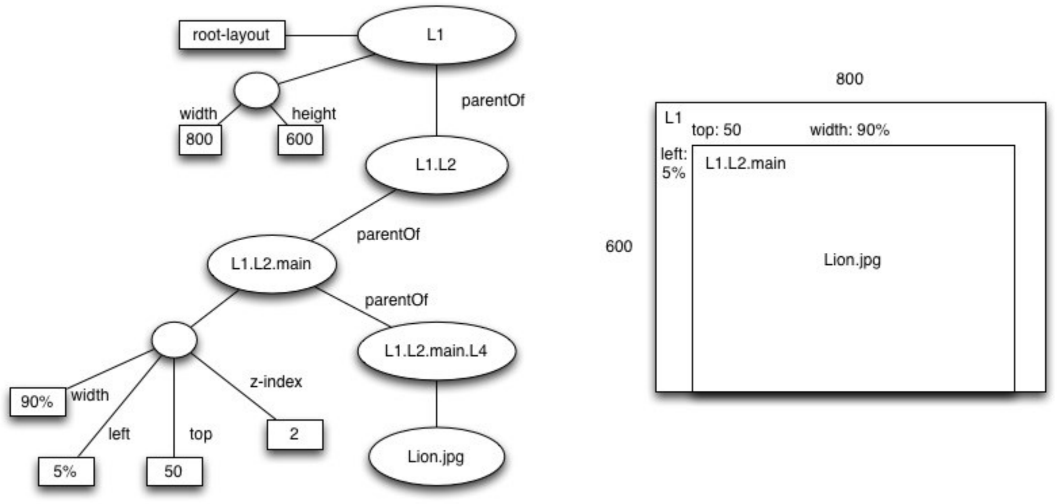


Figure 3-8: A RDF dataset and its visual representation as a tree

The node of the tree is the Subject that all the sub-nodes are linked to. This implies that the root node in the RDF tree represents the composed multimedia document and all the sub-nodes serve as a description to the multimedia document and its media elements. The transformation process iterates over all the nodes and their linking predicates to translate them to their equivalent attributes in the multimedia model. The Predicates are translated to attributes. The Objects represent the values of these attributes, covering both primitive and complex values. These attributes and their values belong to a Subject, e.g., to a media element.

We shall point out that there is a difference in the definition and the aim of the transformation phase between the reference multimedia process and DCMC. In the reference process the transformation process aims at transforming the composed multimedia model directly to the final presentation format. However, in DCMC, taking into consideration compatibility and efficiency factors, the transformation phase transforms the RDF dataset that represents the composed multimedia document into an abstract multimedia document model. This intermediary step serves the purposes of: (i) Allowing the target presentation device to determine the final presentation format, which obviously is supported by that device. (ii) Since the authoring process runs on a distributed authoring environment involving potentially different types of smart devices, it is more efficient to adopt a de-centralized transformation process instead of a centralized transformation process.

Phase 8 - *Presentation*:

Short description: the authored multimedia document is transferred, converted and presented on the devices.

The *Transformation* phase produces the multimedia document model that can be converted to a concrete presentation model. This conversion to a concrete presentation format is the first step in the *Presentation* phase. The presenting device selects the concrete presentation format that it supports. Once the conversion is completed and the presentation is created, the authoring process notifies the user about the availability of the new multimedia document.

3.3.4 Summary

In this section, we drew the complete picture of the authoring and sharing process for multimedia documents in PAN. The authoring process in DCMC is guided by means of the semantics extracted from the original sources. The core of the multimedia authoring process in DCMC follows the reference process for multimedia composition presented in section 3.3.2. The authoring process in DCMC extends that reference process with additional phases to support the composition of multimedia documents about the user's SOI in PAN. The authoring process incorporates eight phases that are derived from and guided by the requirements that we have identified in section 3.2. The driving key element of the authoring process is what we define as the Subject of Interest (SOI), and which we infer from the content provided by the user and use to initiate the process of multimedia content authoring and sharing.

3.4 The Architecture of DCMC

In sections 3.2, and 3.3 we defined the requirements and the phases of the authoring process. In this section we present the service-oriented architecture (SOA) of DCMC.

In the course of defining the requirements we have identified three logically separated groups of requirements: (i) requirements for PAN management, (ii) requirements for multimedia document authoring and sharing, and (iii) non-functional requirements that add constraints on the system. These requirements guided the definition of the authoring process that consists of eight phases, each phase represents a logically encapsulated set of services, tasks and functions. In order to reflect the requirements and the phases of our authoring process by means of a software system, we identified seven logically separated components that construct the DCMC application. Table 3-4 depicts the logical relationships between the requirements, the phases of the authoring process, and the seven components of the system.

Requirements group	***Authoring Process***	***Components***
Networking	PAN establishment	Networking Framework
Multimedia Authoring	Analysis	Analyzer
	SOI inference	
	Enrichment	LOD API framework
	Selection	Selector
	Composition	Composer
	Transformation	Framework for multimedia document and RDF
	Presentation	
Non-functional	Constraints	

Table 3-4: The logical mapping of requirements, process and SOA

There are varying definitions of components [Murray et al. 2004]. A component can be defined as an irreducible part of aggregation of parts that make up a system, also called a subsystem [Hoffer et al. 2002]. A component is a software unit or data unit that has a defined interface for which it provides an instantiation [Schmidt and Buschmann 2003; Murray et al. 2004]. Components can be perceived as service providers. Services can be defined as loosely-coupled independent computation units [Shaw 2011]. This implies that components characterized by: (i) being independent executable entities, and (ii) publishing their interfaces and all interactions are through those interfaces [Sommerville 2001].

A framework is a generic structure that can be extended to create a more specific sub-system or application [Sommerville 2001]. Frameworks are sub-system designs made up of a collection of abstract and concrete classes and the interface between them [Wirfs-Brock and Johnson 1990].

Based on these definitions of components and frameworks, we deduct the following features that can be linked to components and frameworks: (i) A collection of sub-systems construct a framework. (ii) These sub-systems define interfaces to enable the interaction with another sub-system. (iii) Components are sub-systems that provide specific services. A Computing platform with multiple services enables a form of modularity called (SOA) [Shaw 2011]. (iv) Components can be specified abstractly, or can provide an implementation for other abstract components. (v) Frameworks are not standalone systems; systems are constructed by coherently integrating a number of frameworks in a SOA.

This analysis of frameworks and components provides us the basis to decompose the DCMC system into frameworks and components, and allows us to define those frameworks and components at an abstract level. This means that we can define generic services, their interfaces, functions, and interactions. Such definitions enable us: (i) to describe the components of the system and the way they interact with each other in terms of abstract services without having to get into the details of the implementation, and (ii) to present the system and its parts, processes, and services at a manageable theoretical level.

3.4.1 Networking Framework (NtF)

The Networking Framework (*NtF*) as its name suggests provides the required networking services. Those services are derived from the User Stories that we have defined in section 3.2.1.1 and the process for forming the PAN we described in 3.3.3. The framework serves two different levels of use: (i) application level and (ii) component level. At the application level, the DCMC application uses the services that the Networking Framework provides to form a PAN, search for devices, initiate connections, accept connections, listen to incoming connections, and exchange data. At the components level, the components that need to access web resources make use of *NtF* services to establish a http connection and perform request/response interactions. Based on that, we defined two main services that the Networking Framework provides. These services are represented by the two interfaces *PANService* and *HTTPService*. Both services ensure to provide means to the components of the system to exchange data without any constraints on the types of the data being exchanged. This implies that those services provide data transport layer to send and receive data without requiring the data to be of a specific data type.

The interface *PANService* provides the functions listed in Table 3-5, these functions reflect the User Stories and support the PAN formation process. The design of the *PANService* interface considers the neutrality of PAN technologies. In chapter 2.2.2 we presented the available technologies that can be used to form a PAN. The *NtF* specifies APIs that can be implemented for different PAN technologies to support networking services by using technologies such as Bluetooth, Wi-Fi, and others.

Function name	***Description***
Connect	Initiates a connection to a listening device
Disconnect	Disconnects the connection
Send	Sends data to an interconnected device
Receive	Receives data from an interconnected device
Scan	Scans the surrounding environment to discover other PAN-equipped devices
Listen	Listens to connections that other PAN-equipped devices initiate

Table 3-5: PANService – list of functions

The interface *HTTPService* provides functions inherited from the HTTP protocol, those functions allow the applications to send and receive data using HTTP. Table 3-6 lists the functions provided by the *HTTPService.*

Function name	***Description***
Get	Represents a GET method as specified by the HTTP protocol
Post	Represents a POST method as specified by the HTTP protocol
Put	Represents a PUT method as specified by the HTTP protocol
Delete	Represents a DELETE method as specified by the HTTP protocol

Table 3-6: The HTTPService - list of functions

A system that hosts applications dedicates for each application a system thread called the *main-thread* to execute its tasks on this *main-thread*. Applications run various tasks on the *main-thread,* unless an application dedicates a separate thread for a particular task. Running tasks on the *main-thread* is controlled by the flow of the application. This implies that tasks run synchronously, i.e., a task can run only once the preceding task has completed its job. This means, however, that heavy weight tasks block the flow of the application or the *main-thread* till they are completed. This synchronous execution of tasks can be overridden by adopting an asynchronous task execution. An asynchronous execution of tasks can be achieved by means of dedicating separate threads to these tasks. This implies that a callback mechanism must be provided to these asynchronous tasks to allow them to communicate with the application running on the *main-thread* to notify about their progress. Callback functions can be defined as methods that are called in response to events that are recognized by the framework [Sommerville 2001]. Callback patterns can be implemented by means of providing callback methods and listeners that listen to those callback methods. Listeners that are bound to the *main-thread* executes those callback functions once they are called by a process that runs on its own thread. Figure 3-9 depicts the callback pattern.

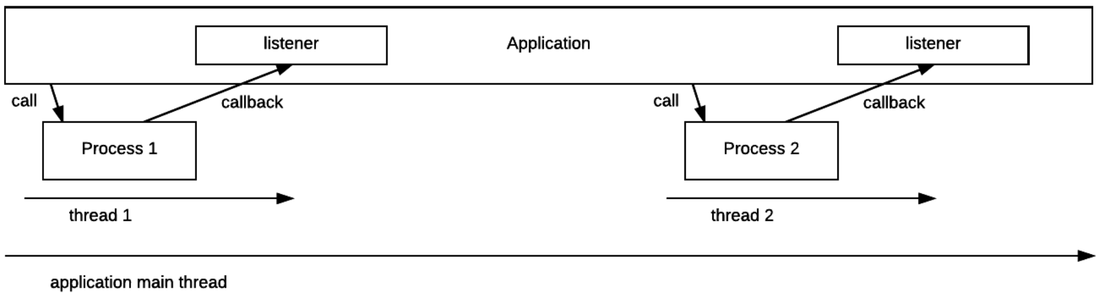


Figure 3-9: The callback pattern

The *PANService* interface provides the following callback functions.

Function name	***Description***
onConnected	Notifies about a successful connection with another device
onListen	Notifies about switching the device to listen mode
onConnectionClosed	Notifies about a closed connection
onConnectionLost	Notifies about a lost connection
onSend	Notifies about sending data
onReceive	Notifies about receiving data

Table 3-7: List of callback functions provided by the PANService

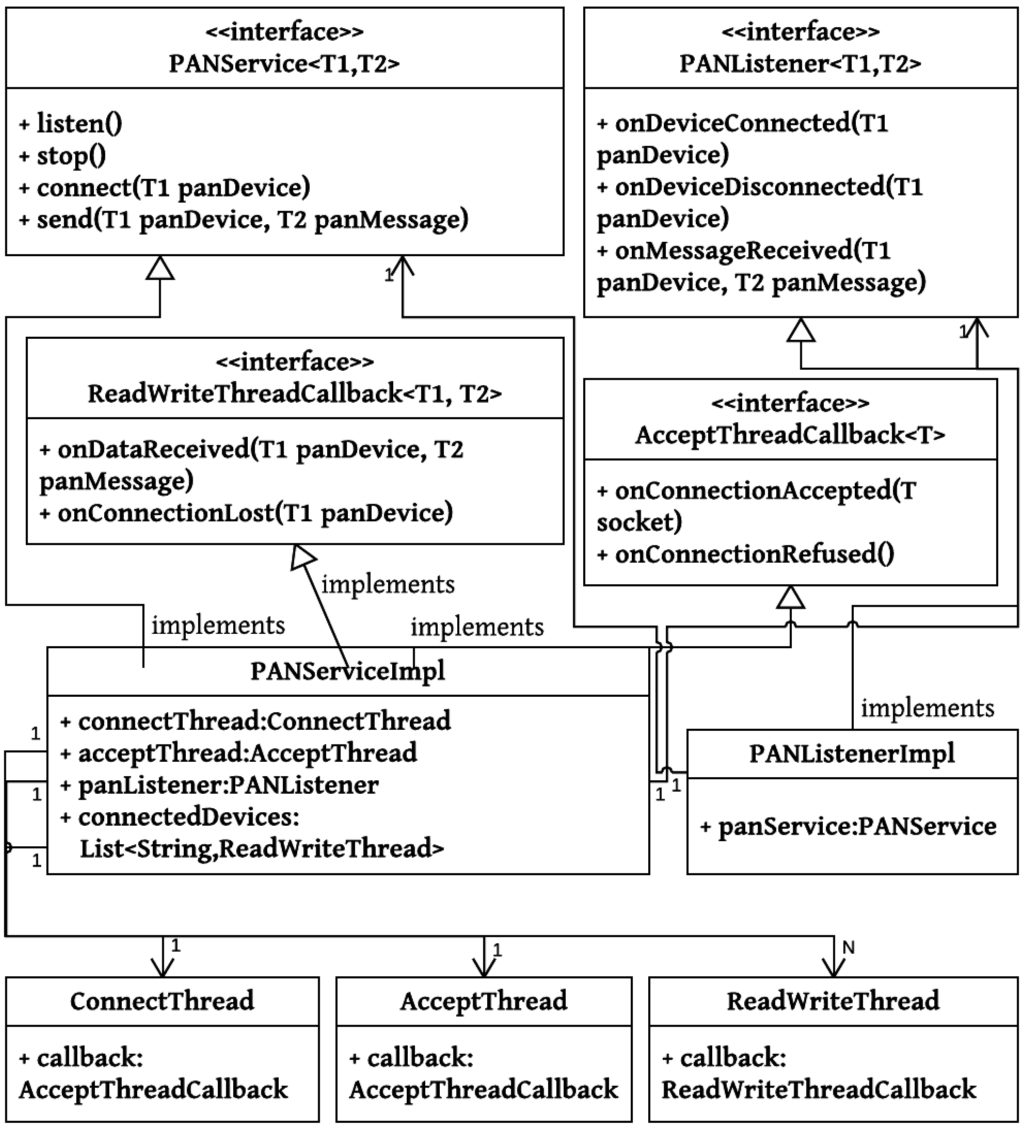
The *HTTPService* interface provides the following callback functions:

Function name	***Description***
onHttpSucess	Notifies about a successful HTTP request/response
onHttpError	Notifies about a failure on a HTTP request/response

Table 3-8: List of callback functions provided by the HTTPService

PANServiceImpl class implements *PANService* interface defining four functions: (i) The *listen*() function, called by the listening device. It starts the *AcceptThread* that calls the callback functions that the interface *AcceptThreadCallback* provides. (ii) The *connect(T1 panDevice)* function, called by the connecting device. It starts the *ConnectThread* that calls the callback functions provided by *AcceptThreadCallback*. (iii) The *send(T1 panDevice, T2 panMessage)* function, which starts the *ReadWriteThread* that uses the open connection socket to transfer data. The *ReadWriteThread* uses the callback functions that the *ReadWriteThreadCallback* interface defines. (iv) The *stop()* function can be called by interconnected devices in order to stop the *PANService* and its active threads. The *PANListenerImpl* class implements the *PANListener* interface that provides callback functions to communicate the PAN events to the application. These events are represented by: (i) *onDeviceConnected(T1 panDevice)*, which informs the application about a successful connection with a PAN-enabled device. (ii) *onDeviceDisconnected(T1 panDevice)*, which informs the application about the disconnection of a device. (iii) *onDataReceived(T1 panDevice, T2 panMessage),* which communicates the received panMessage of type <T2> from panDevice of type <T1> to the application.

Figure 3-10 depicts the services and the components of *NtF*:



<<interface>>
PANService<T1,T2>
+ listen()
+ stop()
+ connect(T1 panDevice)
+ send(T1 panDevice, T2 panMessage)
<<interface>>
PANListener<T1,T2>
+ onDeviceConnected(T1 panDevice)
+ onDeviceDisconnected(T1 panDevice)
+ onMessageReceived(T1 panDevice, T2 panMessage)
<<interface>>
ReadWriteThreadCallback<T1, T2>
+ onDataReceived(T1 panDevice, T2 panMessage)
+ onConnectionLost(T1 panDevice)
<<interface>>
AcceptThreadCallback<T>
+ onConnectionAccepted(T socket)
+ onConnectionRefused()
implements
implements
implements
implements
PANServiceImpl
+ connectThread:ConnectThread
+ acceptThread:AcceptThread
+ panListener:PANListener
+ connectedDevices: List<String,ReadWriteThread>
PANListenerImpl
+ panService:PANService
1
1
1
1
1
1
1
1
1
N
ConnectThread
+ callback: AcceptThreadCallback
AcceptThread
+ callback: AcceptThreadCallback
ReadWriteThread
+ callback: ReadWriteThreadCallback

Figure 3-10: The Architecture of NtF presented as a class diagram

3.4.2 Framework for Multimedia Documents and RDF (FMMDR)

Since various document formats like SMIL, SVG, or HTML5 need to be considered when handling multimedia content, *FMMDR* provides services for transforming different multimedia document models such as SMIL, SVG, and HTML5 from/to RDF encoded content, whereas RDF serves as the internal, canonical, semantic representation scheme.

The non-functional requirements impose constraints on the design of this transformation process, i.e., the abstraction from concrete document models like SMIL, SVG, or HTML5, and decoupling the internal processing logic from various individual document formats. Therefore, *FMMDR* internally makes use of an intermediate representation, an abstract multimedia document model such as ZYX [S. Boll and W.

Klas 2001] (Since ZYX offers all the abstract modelling primitives, there is no need to invent a separate model). The transformation process encompasses two steps: (i) A standard multimedia document model, e.g., SMIL, is converted from/to the abstract document model, i.e., ZYX. (ii) The abstract document model is transformed from/to RDF model, i.e., entities and their properties in the abstract document model are transformed to RDF-triples.

The left side of Figure 3-11 represents services and components of *FMMDR*. The *MultimediaDocumentService* is responsible for the first step of the transformation process. The *XMLParser* de-serializes the source document into programming language objects. The *Mapper* maps such objects to entities in the ZYX model.

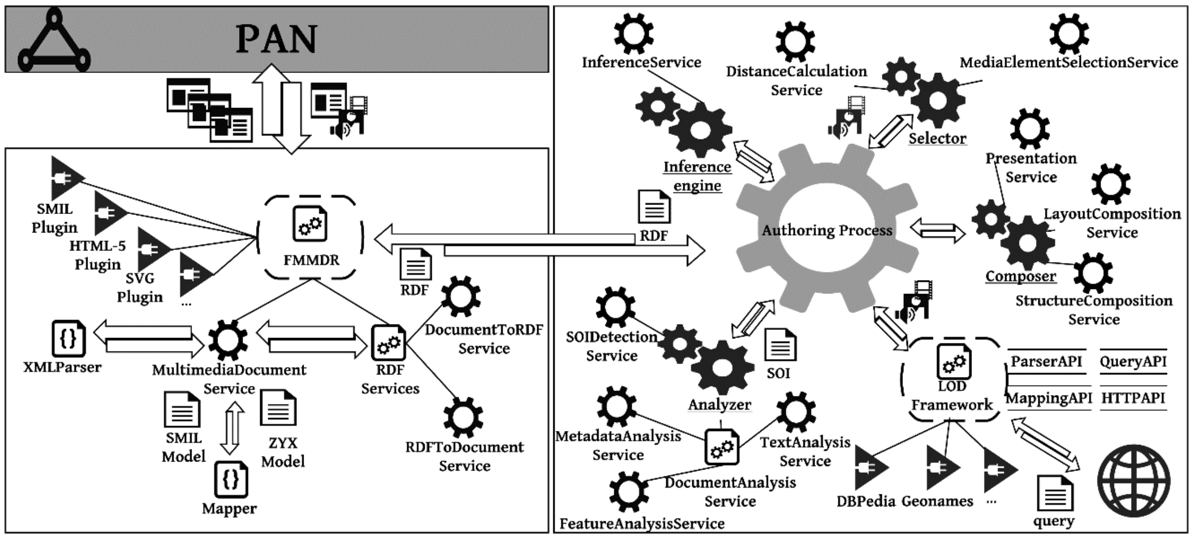


Figure 3-11: Components for the authoring and sharing process for Multimedia Document in PAN

The RDF services provide two services that support the second step of the transformation process, the *DocumentToRDFService* and the *RDFToDocumentService*. By this generic design, *FMMDR* allows its services to handle various types of multimedia document formats and models, indicated by various plugins depicted next to the *FMMDR* component in the figure.

FMMDR framework provides services for mapping different models of multimedia documents such as SMIL, SVG, or HTML5 to RDF, and mapping RDF-encoded documents into different multimedia document models. These mapping services realize a central aspect in DCMC approach. In this regard, DCMC approach inherits the main concepts represented by the object-oriented programming (OOP). OOP can be defined as a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an *instance* of some *class*, and whose *classes* are all members of a *hierarchy of classes* united via inheritance relationships [Booch and Booch 2007]. Based on that, we induce the following aspects: (i) an *object* represents an instance of a *class*, e.g., File, Image, Car, etc. (ii) A *class* models tangible or intangible real-world things by means of specifying *properties* that characterize these things, e.g., the car has a color, these *properties* have values, e.g., the color is green. (iii) *properties* can be either primitive data types such as String, Numeric, Boolean, etc. or complex data types such as other classes, e.g., Color, Address, etc. Figure 3-12 depicts those aspects that OOP represents by means of a UML class diagram.

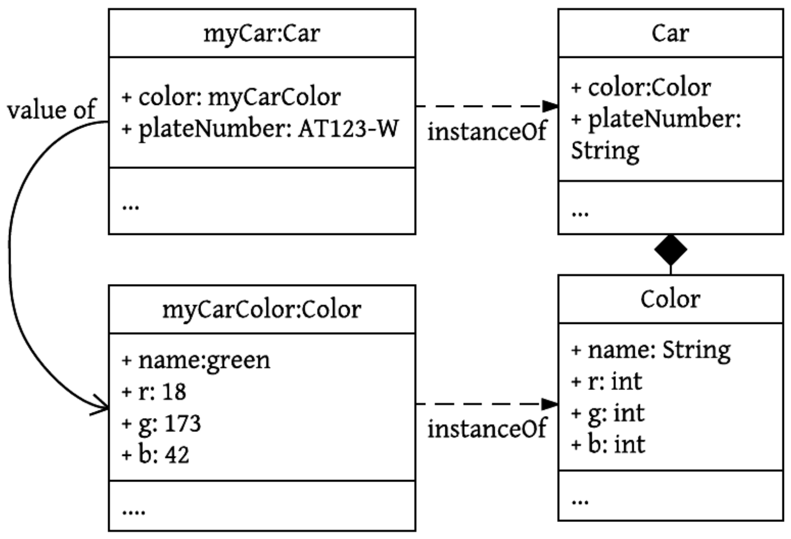


Figure 3-12: Classes, properties and instances in OOP

By applying these concepts presented by the OOP on multimedia documents, we conclude the following guidelines: (i) a multimedia document is an *object* that represents an *instance* of a multimedia document *class*. (ii) The multimedia document *class* has a set of hierarchical primitive and complex *properties*.

In the previous section we presented the three parts of RDF statements, these are Subjects, Predicates and Objects. The structure of a *class* can be described by means of RDF triples as depicted in Table 3-9. To map an *instance* of a *class* into RDF statements, we make use of the following logic: (i) an *instance* of the *class* can be mapped as an RDF-Subject, (ii) the *properties* of the *instance* can be mapped as RDF-Predicates, and (iii) the *values* of those *properties* can be mapped as RDF-Objects. These RDF-Objects can be either in form of Literals or Resource values, i.e., simple or complex data types. *FMMDR* adopts this logic to achieve the mapping between multimedia documents and RDF triples. These mapped multimedia documents can be reconstructed to their *instance* of a *class* representation by simply reversing the presented mapping logic.

Instance	***RDF part***
Instance: *myCar:Car*	Subject
Attribute: *plateNumber*	Predicate
Value: *AT123-W*	Object

myCar hasPlateNumber: AT123-W

Table 3-9: Mapping between instances of classes and RDF Statements

Having introduced the mapping logic that can be applied by the DCMC, we specify the services that *FMMDR* provides. The main target of *FMMDR* is to provide the mapping service as we mentioned earlier in this section.

The interface *DocumentToRDFService* aims at providing the mapping service from a model of a multimedia document to RDF. In the previous sections, we introduced different models of multimedia documents such as SMIL, SVG, or HTML5. In order to handle these different models, the *DocumentToRDFService* service is defined at an abstract level, meaning that, at the abstract level, we define a service that converts a model of a multimedia document to RDF, but we do not specify any concrete multimedia document model such as SMIL or HTML5 that the service handles. The input of the service is a documents of type multimedia document. The output is the representation of this document encoded as RDF triples. To support a certain type of multimedia document models, for example SMIL, an implementation of the abstract service of the framework for the model of SMIL should be provided. By providing this abstract level, the *FMMDR* can add support to a wide range of multimedia document models provided that the service is implemented for these specific models.

The interface *RDFToDocumentService* aims at mapping the RDF-encoded documents back into a multimedia document model. The design of *RDFToDocumentService* interface follows the same concepts of abstract services that we have presented for *DocumentToRDFService* service. This implies that this service can map an RDF to any multimedia document model provided that the service is implemented for that specific model.

Multimedia documents such as SMIL and HTML are structured by means of markup languages, e.g., XML. In order to handle the XML representation of a multimedia document, *FMMDR* transforms the XML representation of a multimedia document from/to its object model representation. This transformation process is known as serialization and de-serialization of objects. The serialization is to transform the objects of a programming language to a data-stream, this data-stream can be consumed in many ways, one of these ways is to save this data-stream to a text file. De-serialization is to transform a data-stream to programming language objects. The transformation process is used by *FMMDR* to transform the multimedia documents represented by means of XML syntax to objects, and transform these objects back to XML documents. *FMMDR* provides a generic *XMLParser* class that handles the processes of the serialization and de-serialization. The *XMLParser* class aims at providing a generic parsing service that handles the transformation process. The *FMMDR* provides a *Mapper* class, this *Mapper* class aims at providing a generic mapping service that handles the mapping of the transformed objects from/to entities in the abstract model that serves as the internal, canonical multimedia model, e.g., ZYX model.

Now that we have defined the main services and components of *FMMDR*, Table 3-10 summarizes those services and components and the functions they provide:

Service/Component	***Function name***	***Description***
DocumentToRDFService	toRdf	Converts the document to RDF
RDFToDocumentService	toDocument	Converts the RDF to document
XMLParser	fromXml	De-serializes a stream represented in XML to a software object
	toXml	serializes the object to a stream the represents the XML file
Mapper	fromMultimediaDocumentModel	
	toMultimediaDocumentModel	

Table 3-10: Services, Components, and Functions of (FMMDR)

3.4.3 Analyzer

The *Analyzer* depicted in the right side of Figure 3-11 supports the analysis process. The analysis process aims on the one hand at capturing the required information about the available media elements needed by the authoring process, and on the other hand at detecting the SOI. The *Analyzer* comprises two services: (i) The *DocumentAnalysisService* performs high-level and low-level feature analysis. It makes use of additional three sub-services: *MetadataAnalysisService*, *TextAnalysisService*, and *FeatureAnalysisService*. (ii) The *SOIDetectionService* uses the analysis results that the *DocumentAnalysisService* and its sub-services provide to detect one or more SOI.

The *DocumentAnalysisService*, in the *Analysis* phase the *Analyzer* performs both high-level and low-level features analysis.

The high-level analysis targets the metadata level that provides information about the multimedia document and each element of the media elements that the multimedia document includes. Multimedia document models such as SMIL, HTML5, and SVG, provide additional data about the multimedia document in metadata tags. The metadata tags are XML tags that are placed within the document and contain additional information. As an example, a metadata tag in SMIL model has the tag <meta> and attributes such as "xml:id", "name", and "content". A SMIL document can contain several "meta" tags. The *Analyzer* detects these meta tags and encodes them into RDF statements in a similar mapping logic that we used to encode the *instances*, the *properties* and the *values* of the *objects* that represent the multimedia documents. Multimedia document models such as SMIL, HTML5, and SVG support the representation of the metadata in RDF syntax. If RDF metadata parts are present, the *Analyzer* detects those RDF parts and includes them in the analysis.

The interface *DocumentAnalysisService* provides the *analyzeDocumentMetadata* function that runs the metadata analysis task. Individual media elements such as images, videos and audios can be annotated with additional data, i.e., metadata. To capture such metadata about an individual media element, many metadata standards have emerged to serve this purpose, e.g., EXIF [EXIF.org 2016], DCMI [Dublin Core Metadata Initiative (DCMI) 2016] and XMP [XMP 2016].

In order to deal with those different metadata formats the *Analyzer* dedicates an abstract metadata analysis service, called the *MetadataAnalysisService.* The *MetadataAnalysisService* aims at converting the metadata about an individual media element to RDF triples. By implementing the *MetadataAnalysisService* for a specific metadata scheme, e.g., EXIF, the *Analyzer* can process the metadata, which are encoded by means of EXIF scheme. *MetadataAnalysisService* can detect this additional information about a specific media element, which are encoded by means of one of the standard metadata schemes. The format of media elements such as images, videos and audios supports standard metadata schemes. However, it is common that media elements of the mentioned types do not include enough metadata, mainly because the producers of those media elements did not provide those additional metadata.

The *Analyzer* provides *FeatureAnalysisService*, which detects additional information about the media elements by performing a low-level features analysis, e.g., detecting the histogram of an image. Multimedia document models such as SMIL and HTML allow the inclusion of media elements directly in the body of document, i.e., not as a referenced external resource, e.g., the text elements. Text elements are included in the body of the multimedia documents and encoded as ASCII code. For those directly included media elements the *Analyzer* provides the *TextAnalysisService*. The *TextAnalysisService* indexes the text and detects keywords that can represent the subject this text is about.

The *SOIDetectionService* uses the results of the *DocumentAnalysisService* and its sub-services to detect one or more SOI. The *DocumentAnalysisService* encodes the results of the analysis process by means of RDF statements. As presented in the previous section, the W3C Semantic Web technologies define how reasoning on RDF datasets can be performed. The *SOIDetectionService* detects a SOI by applying reasoning techniques available by means of W3C Semantic Web standards intended to extract knowledge from RDF datasets. Overall, the analysis process provides a set of media elements that can be used by the authoring process. Those media elements stem from different multimedia documents and are related to different Subjects of Interest (SOI).
Having identified the services of the *Analyzer*, we list in Table 3-11 those services and the main functions they provide.

Service name	*Function name*	*Description*
DocumentAnalysisService	analyzeDocumentMetadata	Analyzes the multimedia document and its metadata tags
MetadataAnalysisService	analyzeMetadata	Analyzes the metadata that relate to media elements
TextAnalysisService	analyzeText	Analyzes text elements
FeatureAnalysisService	analyzeFeatures	Performs a low-level features analysis
SOIDetectionService	detectSOI	Detects the SOI

Table 3-11: Services and functions of the Analyzer

3.4.4 Selector

The *Selector* component aims at selecting those media elements that mostly relate to a specific SOI. The analysis process provides a set of media elements that can be used by the authoring process. Those media elements stem from different multimedia documents and relate to one or more SOI. The S*elector* provides two services to select those media elements. The selected media elements will be used in the composition process of the new multimedia document. These two services are the *MediaElementSelectionService* and the *DistanceCalculationService.* The *MediaElementSelectionService* provides the function *selectMediaElements*, which can be called to select the media elements related to a given SOI. The *DistanceCalculationService* calculates the distance between the SOI and the collected media elements, which later on allows for narrowing down the number of the selected media elements. Only those media elements that have a distance to the SOI within a defined threshold will be selected and made available for the further authoring process. Table 3-12 lists the services that the *Selector* provides.

Service name	*Function name*	*Description*
MediaElementSelectionService	selectMediaElements	Selects the media elements related to a given SOI
DistanceCalculationService	calculateDistance	Calculates the distance between the SOI and a certain media element.

Table 3-12: Selector component services

3.4.5 Composer

The *Composer* aims at creating the multimedia document by defining the temporal and spatial relationships for the selected media elements. Considering the functional and non-functional requirements the *Composer* is comprised of the following services: *(i)* The *PresentationService* creates the RDF representation for the new multimedia document and adds the main elements such as the *keywords*, *metadata*, and the media *elements*. (ii) The *StructureCompositionService* adds the temporal relationships between the elements such as *parallel* and *sequential*. (iii) The *LayoutCompositionService* adds the spatial relationships between the media elements to the RDF representation. The composer will be presented in details in chapter 4.8.
Having identified the services of the *Composer*, we list in Table 3-13 these services and the main functions they provide.

Service name	*Function name*	*Description*
PresentationService	createMultimediaPresentation	Creates the root of the multimedia presentation and adds the basic parts
StructureCompositionService	addStructure	Creates the temporal relationships between the elements
LayoutCompositionService	addLayout	Creates the spatial relationships between the media elements

Table 3-13: Services dedicated for multimedia document composition

3.4.6 LOD API framework

The *LOD API framework* allows for communicating with LOD sources, e.g., DBpedia, GeoNames, or Open Government Data (OGD). The framework provides four modules to communicate with LOD APIs: (i) The Query API for constructing and executing a query, whereas the query could be either a SPARQL query or a GET/POST call to a RESTful web service. (ii) The Parser API, which allows to parse a retrieved LOD source document into objects. (iii) The Mapping API, which allows to map the parsed objects to the internal data model of the system. (iv) The HTTP API, which allows for establishing connections over HTTP in order to send and receive data. Figure 3-13 depicts the high-level architecture of the *LOD API framework*.

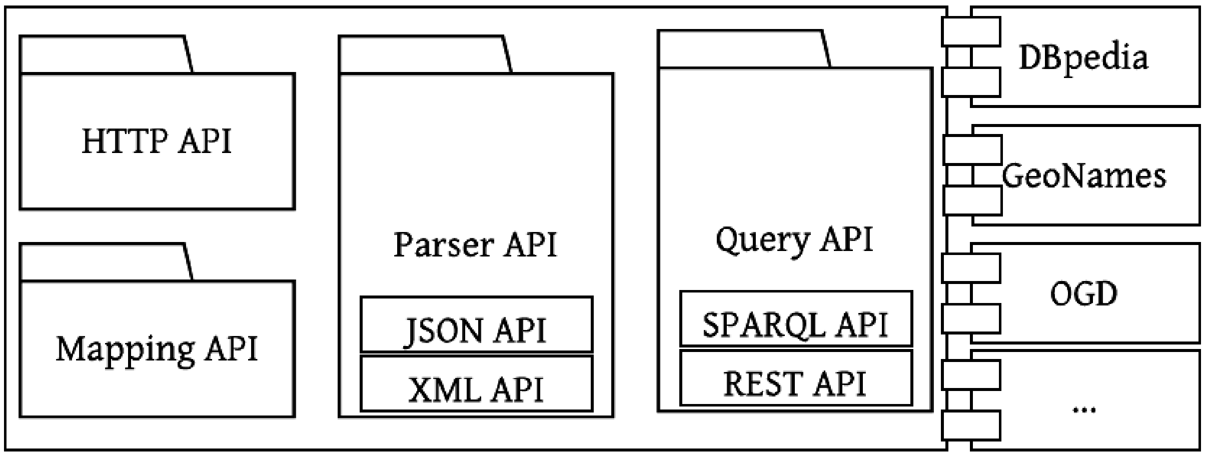


Figure 3-13: LOD API framework architecture

3.4.7 Inference Engine

The *Inference Engine* is a generic reasoning component which is invoked by other components of the system during the lifecycle of the authoring process. It provides the *ReasonnigService* that offers reasoning over a specific RDF dataset according to a given set of rules. This service can be implemented for different Inference Engines, e.g., Apache Jena reasoner [Apache Org. 2017b].

3.4.8 Summary

In this section we presented the components of the DCMC approach. It consists of seven logically separated components: 1) Networking Framework (*NtF*), which includes all the services required for establishing PAN and HTTP connections, and transferring data between devices. 2) Framework for Multimedia Documents and RDF (*FMMDR*), which is a framework for mapping different models of multimedia documents such as SMIL, SVG, or HTML5 to RDF, and mapping RDF-encoded documents back into a multimedia document model. To support a certain type of multimedia document model, for example SMIL documents, an implementation of the APIs of the framework for that document model must be provided. 3) The *Analyzer* which enables the system to work with RDF documents and to apply the rules to infer the SOI and other data required during the authoring process. 4) The *Selector* which selects the media content that mostly relates to the SOI. 5) The *Composer* which merges and composes the selected media content, considering the temporal and special relationships. 6) The *LOD API Framework*, a framework for communicating with LOD sources. Similar to *FMMDR*, in order to support communication with a certain LOD API, for example DBpedia, an implementation for this particular LOD API need to be provided. The framework provides four modules to communicate over LOD APIs, a) the Query API for constructing the query, whereas the query could be either a SPARQL query or a GET/POST call to a RESTful web service, b) the Parser API, which allows to parse a LOD source document into objects, c) the Mapping API, which allows to map the parsed objects to the internal data model of the DCMC, d) the HTTP API providing the facility to establish connections over HTTP in order to send and receive data. 7) Inference Engine, which is invoked by other components of the system during the lifecycle of the authoring process.

3.5 The envisioned Distributed Presentation Player (DisPly)

DCMC aims at an automatic authoring and sharing of multimedia documents in personal area networks. Nowadays, users possess different types of PAN-enabled devices, which allows for the creation of such multimedia documents and pave the way to introduce a new generation of presentation players targeting PAN. Thanks to networking technologies, users can easily interconnect their devices, e.g., in a P2P network, to make the multimedia content available on their home PAN. A typical use case is streaming photos or videos from Wi-Fi-enabled camera to a display device such as a smartphone or a smart-TV. In current use case scenarios, the PAN infrastructure is used to stream multimedia content in a P2P connection as mentioned above.

However, in analogy to the authoring process that uses different input channels to compose multimedia documents, a presentation player can make use of the interconnected devices to achieve a distributed playback of multimedia document. Since a PAN can encompass different types of devices, such as smartphones, smart-TVs, projectors, speakers, and even printers, there is the potential to introduce a new generation of distributed presentation players. Those players are capable to distribute and synchronize the rendering and presentation of one multimedia document using simultaneously different output channels available in a PAN environment.

3.6 Summary

In this chapter, we presented the requirements and the service-oriented architecture (SOA), implemented by the DCMC prototype, guided by the phases of the enhanced authoring process and the application scenarios. Unlike other approaches, our approach incorporates multiple PAN-enabled devices to author and present multimedia content with a focus on the users' Subject of Interest (SOI), enriched and infused with media content retrieved from LOD servers.

We used the notion of User Stories to express the requirements. This kind of requirements serves two purposes, (i) it targets an audience with non-technical background, and at the same time, (ii) it provides a reasonable technical level of specifications that enabled the design of the SOA.

The enhanced multimedia document authoring process targets content authoring in a distributed computing environment. It incorporates Semantic Web technologies in its eight phases to enable a smart and flexible authoring. The *Analysis* phase aims at analyzing the content of the user and storing the result in datasets ready to be exchanged and further processed. The *PAN Establishment* phase handles the connection and data exchange between the PAN-enabled devices. The *SOI Inference* phase aims at detecting one or more SOI from the datasets, which the interconnected devices share. The *Enrichment* phase aims at retrieving additional related information about the detected SOI to augment the content under composition with resources available on the Web of Data. The *Selection* phase aims at selecting a subset of the available media elements to be included in the composition. This subset of media elements represents those media elements that strongly relate to a given SOI. The *Composition* phase aims at arranging the selected media elements into a coherent multimedia document. The *Transformation* phase handles the transformation of the newly composed multimedia document to internal abstract multimedia document model. The *Presentation* phase handles the transfer of the composed multimedia document to the presentation device and transforming the document to its final presentation format.

The SOA developed in the course of DCMC follows well-defined Software Engineering principles. The overall architecture of the DCMC consists of seven logically separated components. These seven components provide different services serving the enhanced authoring process. Hence, modularity and generality of our SOA also allow for supporting other authoring processes as well as offering the functionality as part of a content sharing platform.

The Networking Framework (*NtF*) enables interconnecting multiple devices using various networking technologies to form a PAN. It provides a communication protocol allowing the interconnected devices to exchange and interpret different types of data and messages. The *FMMDR* provides services and framework modules to support the transformation of various multimedia document formats from/to the internal abstract multimedia document model. It allows for further transformation from the internal abstract model from/to RDF. The *Analyzer* provides rule-based analysis services. These services support both local and global analysis, as well as the detection of the SOI. The *Selector* provides a rule-based selection service aiming at selecting media elements from a pool of elements. Those selected elements are included in the authoring process. The *Composer* provides rule-based services serving the composition of the multimedia content. The *LOD API framework* provides generic APIs and services allowing for communicating with various publicly available LOD APIs. The *Inference Engine* provides a generic reasoning service which can be invoked by other components of the system.

4. Proof of Concept (Development of DCMC)

4.1 Introduction

After having presented the requirements, the enhanced authoring process, and the architecture at an abstract level in the previous chapter, in this chapter we present the concrete implementation of each of the defined components of the DCMC. We follow a systematic approach in presenting those components, i.e., each section describing a component consists of the following: (i) the underlying concepts, (ii) the design and specification of the component, (iii) the implementation of the component and its interfaces and classes. We introduce the SMIL-Player we used in our experiments and present an illustrating experiment in detail.

The implementation of DCMC is conducted in the programming language Java and by using the Android framework and Android studio. For additional details about application development using Android framework we refer the reader to chapter 2.2.3.1.

4.2 Basic overview of the DCMC

To enable flexible efficient and rule-based SOI detection and later on multimedia content authoring, our approach uses internally ZYX model [Boll and Klas 1999] and RDF [W3C Org. 2014d]. ZYX is an abstract multimedia document model, which offers all the abstract modelling primitives required by our approach. The ZYX model distinguishes between the *structure* of a document and its *layout*, whereas *structure* covers the temporal relationships between the media elements and *layout* covers the spatial relationships between those multimedia elements. Hence, ZYX is used in our approach to create and manipulate multimedia documents. RDF is used as internal, canonical, semantic representation scheme, which allows for using further Semantic Web standards intended to extract knowledge from RDF datasets.

Multimedia content can be represented by means of different formats like SMIL, SVG, or HTML5. This requires that the DCMC should provide generic services for transforming different multimedia document models such as SMIL, SVG, or HTML5 from/to the internal representation by the ZYX model, and then ZYX model from/to RDF encoded content. The DCMC provides a Document Object Model (DOM) implemented by Java classes for multimedia document models such SMIL and ZYX.

We will refer to these implementations as SMIL-DOM, ZYX-DOM, or multimedia-DOM to refer to a multimedia document model in general. Hence, the abbreviation "DOM" as used in the context of this thesis differs from the DOM standard [W3C Org. 2009] defined by the W3C.

Figure 4-1 depicts just the first level of the internal data structure we used in DCMC to represent a multimedia document by means of RDF, which presents the main RDF resources *mediaElements*, *meta*, *keywords*, *structure*, and *layout*.

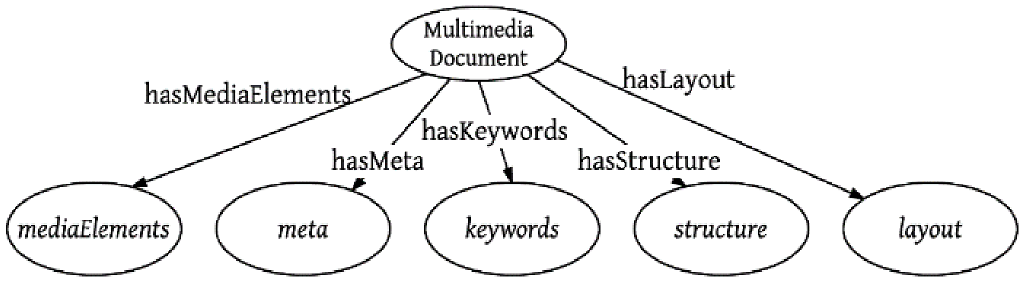


Figure 4-1: The first level of the internal data structure (RDF-Graph)

The resource *mediaElements* interlinks the individual media elements such as texts, images, videos, or audios and information on such media elements. The resource *meta* interlinks the metadata about the multimedia content. The resource *keywords* interlinks the results of the analysis process. The resources *structure* and *layout* interlink the temporal and spatial relationships of the media elements following the notions of ZYX model. Creation, adding, and maintaining of the interlinking of data to these resources *structure* and *layout* are the tasks of the *Composer* component. Additional resources and literals will be cumulatively added and interlinked to the five resources mentioned above by different components of the system.

In the subsequent sections, we use the term *RDF-Graph* to refer to the graph and its interlinked resources; this graph represents our internal data structure. The terms resource, literal, subject, object, and predicate are used as defined by the specification of RDF [W3C Org. 2014d].

The design of the services specified by the SOA of DCMC follows the notion of the generic programming [Stepanov and Musser 2017]. Briefly, generic programming enables types (classes and interfaces) to be parameters when defining classes, interfaces and methods [Oracle 2016a]. By using generic programming, DCMC maintains its implementation flexibility and enable multiple implementations of the various services. Such multiple implementations widen the range of technologies and concepts covered by our approach, e.g., the *MetadataAnalysisService* that the *Selector* component provides can be implemented to add support to several metadata schemes (e.g., EXIF, MPEG-7 and DCMI) and several types of media elements such as audio and video. The same applies to other services.

4.3 Development of the Networking Framework

4.3.1 Underlying concepts

In the course of defining the architecture of the Networking Framework (*NtF*), we have defined two main services that the framework provides: *PANService* and *HTTPService*.

The *PANService* interface specifies generic services that enable DCMC to establish and manage the connections between various types of PAN-equipped devices. It provides abstract methods such as *listen*(), *connect*(), *send*() and *stop*(). The *listen*() method provides a listener to the incoming connections. To establish a connection between two PAN-equipped devices, the *connect*() method takes an object representing a PAN-equipped device as input and initiates a connection. Once the connection has been accepted, the connection process returns the connection socket, both connected devices can use this connection socket to transfer data. The *send*() method makes use of the established connection to send and receive the data. The *send*() method sends the data over a serial port socket. Hence, data are transferred in byte arrays. The *stop*() method closes the connection and releases all the blocked resources. However, at this abstract level, the service does not yet specify any concrete technology that supports PAN, e.g., Bluetooth [Bluetooth SIG 2016], or any of the IEEE 802 family of standards [IEEE Standards Committee 2016]. This implies that the interfaces of the *PANService* must be designed following the notion of generic programming to allow for implementing those interfaces for different PAN technologies.

The *HTTPService* interface is a generic service, which provides methods such as *get*(), *post*(), *put*(), and *delete*(). Those methods are derived from the specifications of the HTTP protocol. The main idea behind the *HTTPService* interface is to provide a content independent communication service to the components of DCMC. It allows the user of the service to communicate with a server using the HTTP protocol. The *HTTPService* interface offers the mentioned HTTP methods, but do not specify the type of the exchanged data and how the request and the response look like. Hence, the user of the service is responsible for constructing the request and providing a handler to interpret the response.

As mentioned above, the abstraction in both services is realized by means of defining generic interfaces.

4.3.2 Design and specifications of the framework

4.3.2.1 PANService

The *PANService* interface depicted in Figure 4-2 provides four main services to DCMC. These four services are represented by the following functions:

```
public interface PANService<T1, T2> {
   public void listen();
   public void connect(T1 panDevice);
   public void send(T1 panDevice, T2 panMessage);
   public void stop();
}
```

Figure 4-2: The PANService interface

listen(): This function enables a PAN-equipped device to enter the listening mode. A device searches for other PAN-equipped devices in the surrounding area. When the search is completed and no other PAN-equipped devices were detected, the searching device calls *listen*(). Since the *listen*() function is blocking, i.e., it returns only after it receives an incoming connection or when it terminates for some reason, it needs to be called on separate thread. Thus, the *listen*() function starts the *AcceptThread* that in return calls one of the callback functions that the *AcceptThreadCallback<T>* provides. Figure 4-3 depicts the *AcceptThreadCallback<T>*. The *onConnectionAccepted(T socket)* function is called once a connection has been accepted, and the function *onConnectionRefused()* otherwise.

```
public interface AcceptThreadCallback<T> {
   public void onConnectionAccepted(T socket);
   public void onConnectionRefused();
}
```

Figure 4-3: The AcceptThreadCallback interface

connect(T1 panDevice): This function enables a PAN-equipped device to initiate a connection to another PAN-equipped device. Since it is a *generic* function, it takes an *instance* of <T1> that represents the PAN-equipped device as input. The input device can be of any type, e.g., a Bluetooth device. The type of the device is defined based on the concrete implementation of the interface *PANService*. The *connect(...)* function is blocking, hence it needs to run on a separate thread in a similar behavior as the *listen()* function. The *connect(…)* function starts the *ConnectThread* that calls the callback functions that the interface *AcceptThreadCallback<T>* provides. Once the connection has been accepted, the *ConnectThread* finishes its work and hands over the connection socket to the *PANService*. The *PANService* uses this connection socket to send and receive data. The *AcceptThreadCallback* interface provides the *onConnectionAccepted(T socket)* function that hands over a connection socket of the type <T>. The type of this connection socket is defined based on the concrete implementation of the *PANService*.

send(T1 panDevice, T2 panMessage): This function enables a PAN-equipped device to send data to another connected device. It takes an *instance* of <T1> that represents the PAN-equipped device and an *instance* of <T2> that represents a data message as input. Once the *PANService* has received a connection socket, it starts the *ReadWriteThread* that uses that socket to transfer data. The *PANService* dedicates and manages an instance of the *ReadWriteThread* for each connection socket. *PANService* provides the *send(…)* function to the application to be used to send messages. The *ReadWriteThread* uses the callback functions that the *ReadWriteThreadCallback* interface specifies to communicate with the *PANService*. The *ReadWriteThreadCallback* provides the *onDataReceived(T1 panDevice, T2 panMessage)* function that hands over an *instance* of <T1> that represents the sending device and an *instance* of <T2> that represents the received message. It also provides the *onConnectionLost(T1 panDevice)* function that is called when the connection is lost handing over an *instance* of <T1> that represents the disconnected device. The concrete types of <T1> and <T2> are provided by the implementation of the interfaces *PANService* and *ReadWriteThreadCallback*. Figure 4-4 depicts the *ReadWriteThreadCallback* interface.

```
public interface ReadWriteThreadCallback<T1, T2> {
   public void onDataReceived(T1 panDevice, T2 panMessage);
   public void onConnectionLost(T1 panDevice);
}
```

Figure 4-4: The ReadWriteThreadCallback interface

stop(): This function enables the application to stop the *PANService.* It guarantees that all alive threads are stopped/closed and all blocked resources by the DCMC application are released.

PANListener: The *PANService* provides a callback mechanism to communicate with the DCMC application to inform about the events that occur on the networking level. This callback mechanism is realized by the *PANListener* interface. Figure 4-5 depicts the *PANListener* interface.

```
public interface PANListener<T1, T2> {
   public void onDeviceConnected(T1 panDevice);
   public void onDeviceDisconnected(T1 panDevice);
   public void onDataReceived(T1 panDevice, T2 panMessage);
}
```

Figure 4-5: The PANListener interface

The *PANListener* interface provides the following callback functions to communicate with the DCMC application:

onDeviceConnected(T1 panDevice): This callback function expects a PAN-equipped device of type <T1> as an input. It informs the DCMC application about a successful connection with a PAN-equipped device and hands over an *instance* of <T1> that represents that connected device to the application.

onDeviceDisconnected(T1 panDevice): This callback function expects a PAN-equipped device of type <T1> as an input. It informs the DCMC application about a disconnected device and hands over an *instance* of <T1> that represents that disconnected device to the application.

onDataReceived(T1 panDevice, T2 panMessage): This callback function expects a PAN-equipped device of type <T1> and a data message of type <T2> as an input. This callback method communicates an *instance* of <T2> that represents the message received from an *instance* of <T1> that represents the sending device.

4.3.2.2 HTTPService

The *HTTPService* interface provides five functions to DCMC components / application. These five functions are inherited from the HTTP protocol and shown in Figure 4-6:

```
public interface HTTPService {
   public void post(HttpPost httpPost);
   public void get(HttpGet httpGet);
   public void delete(HttpDelete httpDelete);
   public void put(HttpPut httpPut);
   public void abort();
}
```

Figure 4-6: The HTTPService interface

The functions *get(HttpGet httpGet), post(HttpPost httpPost), put(HttpPut httpPut),* and *delete(HttpDelete httpDelete)* perform the operations that correspond to their names, i.e., post, get ,put, and delete as specified by the HTTP protocol.

The *abort()* function can be used to cancel a running HTTP request, which is initiated by one of the four presented HTTP calls.

The *IResponseHandler* interface depicted in Figure 4-7 provides a generic interface to handle the response from the server. The specific type of the response can be defined by the class implementing the interface, i.e., if the application expects to receive a response of type "SmilDocument.class", the interface must be implemented to handle that type. The *IResponseHandler* interface provides three functions, the *handleResponse(T response)* and *handleResponse(T response, int responseType)* are called after a successful communication with the server. These functions pass the handled response received form the server to the application. The *onHttpError(String e)* function informs the application about an error in the connection; it passes the cause of the error to the application to enable the application to respond accordingly.

```
public interface IResponseHandler<T> {
   public void handleResponse(T response);
   public void handleResponse(T response, int responseType);
   public void onHttpError(String e);
}
```

Figure 4-7: The IResponseHandler interface

The *IResultParser* interface depicted in Figure 4-8 is a generic interface that can be implemented for a specific type of result. Similar to the interface *IResponseHandler*, if the application expects to receive a response of type "SmilDocument.class", the *IResultParser* interface must be implemented to parse "SmilDocument.class". Since servers transmit the response as a stream of bytes, the method *parse(InputStream in)* expects an *InputStream* as a parameter and returns an object of the type that the interface is implemented for.

```
public interface IResultParser<T> {
   public T parse(InputStream in);
}
```

Figure 4-8: The IResultParser interface

Finally, to transform the *InputStream* to the expected type of response, the *NtF* provides two generic classes, those are the *XMLParser<T>* and *JSONParser<T>*. Both classes do a similar job, the *XMLParser* parses the response that is represented as XML format. *JSONParser* parses the response that is represented as JSON format. By providing *generic* classes for parsing XML and JSON formats, the *NtF* aims at saving the efforts of writing a dedicated parser for each type of server response.

4.3.3 Implementation of the framework and its components

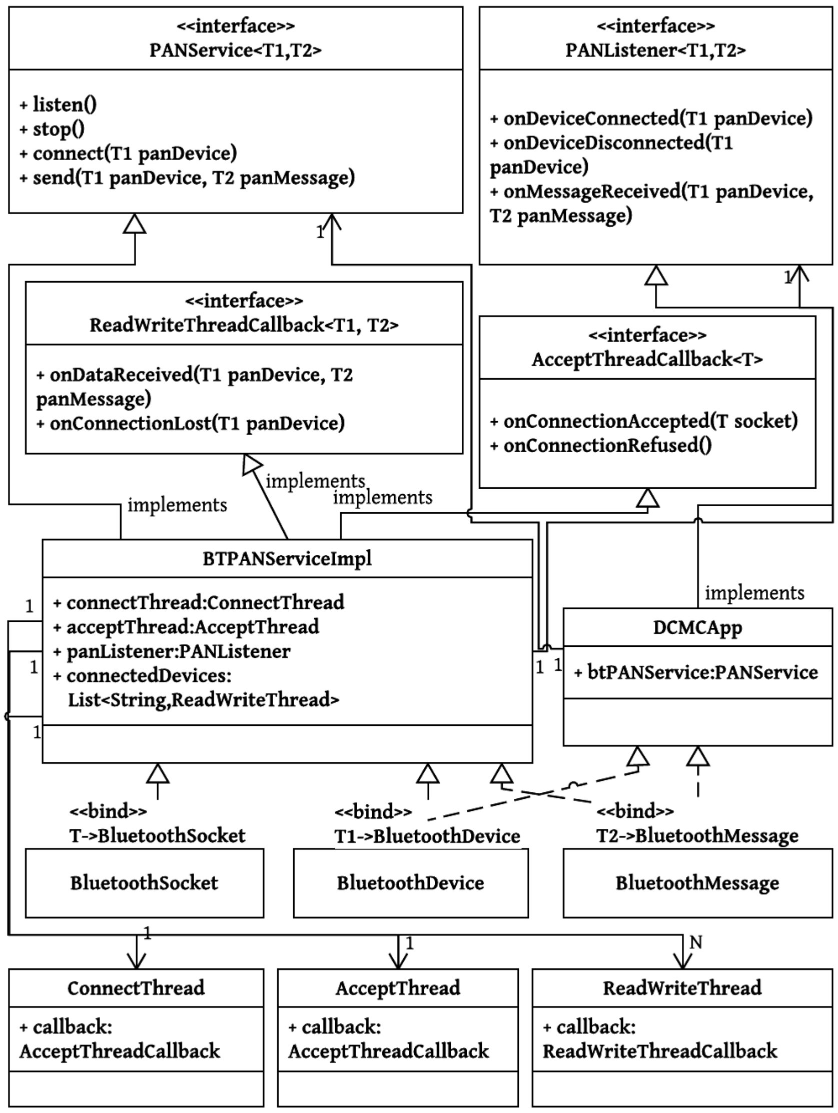
Bluetooth can be considered as the ad-hoc PAN technology, hence, we opted to provide an implementation of the *PANService* interface to support this technology. Figure 4-9 depicts the architecture of the implementation of the *PANService* to support the Bluetooth technology.

The *PANService* is responsible on the one hand for managing the connection between the devices, and on the other hand for providing a suitable infrastructure to facilitate the data exchange between the interconnected devices. Hence, the Bluetooth implementation of the *PANService* involves and provides the following classes:

BluetoothDevice and **BluetoothSocket:** These two classes are provided by the Android framework. The **BluetoothDevice** class represents a Bluetooth-enabled device. The **BluetoothSocket** represents the connection socket that has been established between two interconnected devices. This socket is used to transfer data between the two interconnected devices.

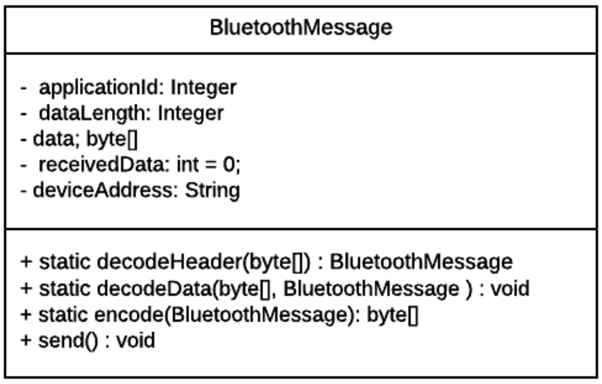
BluetoothMessage: This class represents the message format that the interconnected devices will use to exchange data. The **BluetoothMessage** consists of two parts, a *header* and a *body*. The length of the *header* of the message is 10 hexadecimal characters, whereas 4 hexadecimal characters represent the application id, and 6 hexadecimal characters represent the length of the *body*. The *body* of the message contains the payload of the message. The maximum size of the payload is given by the number of bytes that is represented by the maximum decimal value of the 6 hexadecimal characters, which is equal to 16,777,215 Bytes or ~ 16 Megabytes.

The Bluetooth message has the attributes and functions as depicted in Figure 4-10. The attribute *applicationId* of type integer informs the application about the type of the **BluetoothMessage** to enable the application to respond accordingly. The *dataLength* attribute informs the application about the expected length of the payload part of the message, to ensure that the **BluetoothMessage** is properly parsed by the receiving application. The attribute *data* represents the payload of the Bluetooth message. The attribute *receivedData* holds the number of bytes that have been received. This is an internal attribute for the **BluetoothMessage** used to check correctness and completeness of the received message. The *deviceAddress* provides the address of the device that sends the **BluetoothMessage**; this address is used to send the response to the correct Bluetooth device.



<<interface>> PANService<T1,T2>
+ listen()
+ stop()
+ connect(T1 panDevice)
+ send(T1 panDevice, T2 panMessage)
<<interface>> PANListener<T1,T2>
+ onDeviceConnected(T1 panDevice)
+ onDeviceDisconnected(T1 panDevice)
+ onMessageReceived(T1 panDevice, T2 panMessage)
<<interface>> ReadWriteThreadCallback<T1, T2>
+ onDataReceived(T1 panDevice, T2 panMessage)
+ onConnectionLost(T1 panDevice)
<<interface>> AcceptThreadCallback<T>
+ onConnectionAccepted(T socket)
+ onConnectionRefused()
implements
implements
implements
implements
BTPANServiceImpl
+ connectThread:ConnectThread
+ acceptThread:AcceptThread
+ panListener:PANListener
+ connectedDevices: List<String,ReadWriteThread>
DCMCApp
+ btPANService:PANService
<<bind>> T->BluetoothSocket
<<bind>> T1->BluetoothDevice
<<bind>> T2->BluetoothMessage
BluetoothSocket
BluetoothDevice
BluetoothMessage
ConnectThread
+ callback: AcceptThreadCallback
AcceptThread
+ callback: AcceptThreadCallback
ReadWriteThread
+ callback: ReadWriteThreadCallback
1
N

Figure 4-9: The Architecture of the Bluetooth implementation of NtF



BluetoothMessage

- applicationId: Integer
- dataLength: Integer
- data; byte[]
- receivedData: int = 0;
- deviceAddress: String

+ static decodeHeader(byte[]) : BluetoothMessage
+ static decodeData(byte[], BluetoothMessage) : void
+ static encode(BluetoothMessage): byte[]
+ send() : void

Figure 4-10: UML design of the Bluetooth message

The **BluetoothMessage** class provides the following functions:

decodeHeader(byte[]): This function is used to extract the header of the Bluetooth message from the received byte array.

decodeData(byte[]): This function is used to extract the body, i.e., the payload of the Bluetooth message from the received byte array.

encode(BluetoothMessage): This function is used to convert/encode the Bluetooth message into a byte array.

send(): This function is used to send the Bluetooth message to a device.

BTPANServiceImpl: The **BTPANServiceImpl** class implements the following interfaces:

The *PANService*<BluetoothDevice, BluetoothMessage> interface.

The *AcceptThreadCallback*<BluetoothSocket> interface.

The *ReadWriteThreadCallback*<BluetoothDevice, BluetoothMessage> interface.

BTPANServiceImpl provides implementation for the functions that the *PANService* interface specifies. The **BTPANServiceImpl** implements additionally two callback listeners. To establish a connection between PAN-equipped devices, the **BTPANServiceImpl** starts a *ConnectionThread.* If the connection is accepted, the callback function *onConnectionAccepted* is called. This function passes an instance of the **BluetoothSocket** class that represents the connection socket and an *instance* of the **BluetoothDevice** class that represents the connected device to the **BTPANServiceImpl**.

The **BTPANServiceImpl** provides an internal data structure to hold the connected Bluetooth device and the corresponding Bluetooth connection socket, both devices can use this Bluetooth connection socket to send and receive data. The *ReadWriteThreadCallback* implementation passes instances of the **BluetoothMessage** class that represent the messages to be exchanged between the interconnected devices.

BTPANListenerImpl: The **BTPANListenerImpl** class implements the *PANListener* interface. This implementation is provided by the application that uses the *NtF,* i.e., the **DCMCApp** (see Figure 4-9**)**. The functions that the *PANListener* specifies provide the DCMC application with information about the network as mentioned in preceding parts of this section.

4.4 Development of Framework for Multimedia Documents and RDF

4.4.1 Underlying concepts

Since various document formats like SMIL, SVG, or HTML5 need to be considered when handling multimedia content, the Framework for Multimedia documents and RDF (*FMMDR)* provides generic services for transforming different multimedia document models such as SMIL, SVG, or HTML5 from/to RDF encoded content. These multimedia models are represented by means of markup languages, i.e., in XML format. This implies that *FMMDR* provides a parsing utility that de-serializes multimedia models represented by XML format to their equivalent data models implemented by means of classes of object-oriented programming (OOP) languages. Since each multimedia document format has its own data structure, *FMMDR* aims at providing a generic XML parser. This generic parser makes use of concepts such as *reflection,* e.g., Java reflection API [Oracle 2016c] and *annotation,* e.g., Java Annotation API [Oracle 2016b], which OOP languages support. The *reflection API* provides means to the applications to inspect classes, fields and functions at runtime. The *annotation API* provides metadata about those classes, fields and functions. By using both APIs, *FMMDR* provides a generic *XMLParser* that inspects the classes, accesses their fields and executes functions to achieve this type of generic parsing. The *XMLParser* aims at serializing/de-serializing XML documents from/to their representation in OOP object.

As presented in chapter 3, the FFMDR uses internally an abstract multimedia document model, i.e., ZYX model, to create and manipulate multimedia documents. This implies, however, that after de-serializing a multimedia document to OOP objects, e.g., SMIL-*XML* to SMIL Document Object Model (SMIL-DOM), the created SMIL-DOM must be mapped to the internal object model of DCMC, i.e., to ZYX-DOM. *FMMDR* provides a generic *Mapper* that follows the concepts of *reflection* and *annotation* as presented above. By adopting those generic computing concepts, *FMMDR* aims to avoid implementing an *XMLParser* and object *Mapper* for each format of the available multimedia document formats. This as well minimizes the efforts of implementing and testing the components of the framework.

Multimedia content authoring process of DCMC defines RDF as its internal, canonical, semantic representation scheme. This implies that ZYX-DOM must be transformed to the internal *RDF-Graph*. *FMMDR* provides two generic interfaces to enable this transformation process: *DocumentToRDFService* and *RDFToDocumentService*.

4.4.2 Design and specifications of the framework

The *XMLParser* depicted in Figure 4-11 shows the implementation of the *XMLParser* we presented in the previous section. The *XMLParser* offers two generic functions, these are the *fromXml(File file)* function that accepts a *File* entity as an input parameter. This *File* parameter provides the multimedia document file, e.g., SomeFile.smil. The return type of this method is a generic object that is represented by the type <T>. The *toXml(T object, File file)* accepts two input parameters, the first parameter represents an entity of the type <T>, the second parameter represent the *File* entity. This method serializes the input object of type <T> into its xml representation and stores the result in the provided file object that can be saved, e.g., on an external storage medium.

```
public class XMLParser<T> {

   protected Serializer serializer;
   protected Class<T> clazz;

   public T fromXml(File file) throws Exception {
      T object = (T) serializer.read(clazz, file, false);
      return object;
   }
   public File toXml(T object, File file) throws Exception {
      serializer.write(object, file);
      return file;
   }
}
```

Figure 4-11: The XMLParser *generic* class

ZYXParser depicted in Figure 4-12 implements the *Mapper* concept we presented in the previous section. The *ZYXParser* offers two methods: (i) the *fromMMDoc(T mmDoc)* method that maps a specific multimedia-DOM of type <T> to its equivalent representation in ZYX model, it returns the *ZYXDocument* object. (ii)The *toMMDoc(ZYXDocument zyxDocument)* method that maps a *ZYXDocument* object back to a specific multimedia-DOM of type <T>. The concrete type of the multimedia document model is defined by extending the *ZYXParser* class and providing a concrete object type for <T> as depicted in Figure 4-16.

```
public class ZYXParser<T> {
   public ZYXDocument fromMMDoc(T mmDoc) throws ZYXParsingException {
      return handleFromMMDoc(mmDoc);
   }
   public T toMMDoc(ZYXDocument zyxDocument) throws ZYXParsingException
  {
      return handleToMMDoc(zyxDocument);
  }
}
```

Figure 4-12: The ZYXParser *generic* class

The *FMMDR* defines a set of ZYX annotations. These annotations can be used to annotate the classes that represent the object model of a certain multimedia document format, e.g., SMIL. Annotating these object model classes, fields and functions serves the process of mapping these items from/to ZYX-DOM. Figure 4-13 depicts the interface that implements the annotation *ZYXPresentationElement*. This annotation can be used to mark a class, an attribute, or a method as a *ZYXPresentationElement*. The *ZYXPresentationElement* inherits the features of a presentation element as specified by ZYX model. Additional information about the annotation framework will be presented in the next section 4.4.3.

```
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.TYPE, ElementType.FIELD, ElementType.METHOD})
public @interface ZYXPresentationElement {
   public ZYX type() default ZYX.TEMPORAL_ELEMENT;
   public boolean list() default true;
   public boolean root() default false;
}
```

Figure 4-13: The ZYXPresentationElement annotation interface

The *DocumentToRDFService* interface depicted in Figure 4-14 provides a generic interface that can be implemented to handle a specific multimedia-DOM and RDF model. The method *createRDFModel(T1 mmDocument)* accepts an object model that represents a specific multimedia document format as input and returns its representation in RDF model.

```
 public interface DocumentToRDFService<T1, T2> {
   public T2 createRDFModel(T1 mmDocument);
}
```

Figure 4-14: The DocumentToRDFService interface

RDFToDocumentService interface depicted in Figure 4-15 and its *createMultimediaDocument(T1 rdfModel)* function follow the same logic we presented to specify the interface *DocumentToRDFService.* The *RDFToDocumentService* aims at converting an RDF model to a specific multimedia-DOM.

```
public interface RDFToDocumentService<T1, T2> {
   public T2 createMultimediaDocument(T1 rdfModel);
}
```

Figure 4-15: The RDFToDocumentService interface

4.4.3 Implementation of the framework and its components

Having presented the underlying concepts and the specifications of *FMMDR*, we provide in this section the implemented interfaces and components.

The **SMILDocumentParser** extends the generic class *XMLParser* to enable de-serializing SMIL documents represented by means of XML format to a SMIL-DOM. Figure 4-16 presents the full implementation of the *SMILDocumentParser*. The purpose of extending the generic class *XMLParser* is to define a specific type that the parser will handle, that is in our implementation the *SMILDocument* entity type. The parsing logic itself is implemented by the generic class, i.e., by *XMLParser* class.

```
public class SMILDocumentParser extends XMLParser<SMILDocument> {
   public SMILDocumentParser() {
      super(SMILDocument.class);
   }
}
```

Figure 4-16: The SMILDocumentParser class

The SMIL-DOM in DCMC is implemented by the class *SMILDocument*. The *SMILDocument* class represents the root of a hierarchical set of classes that build the SMIL-DOM. Figure 4-17 depicts part of the *SMILDocument* class and shows the annotations, which are used to supports the generic concepts the framework provides.

```
@Root(name = "smil")
public class SMILDocument {
   @Element(name = "head", required = true)
   private Head head;
   @ZYXPresentationElement(root = true, list = false)
   @Element(name = "body", required = true)
   private Body body;
}
```

Figure 4-17: The SMILDocument class

SMILZYXParser extends the generic class *ZYXParser* to enable mapping the *SMILDocument* from/to *ZYXDocument*. Figure 4-18 depicts the full implementation of *SMILZYXParser* class. The purpose of extending the generic class *ZYXParser* is to define a specific type for the parser, that is the *SMILDocument* entity type. As already presented, the generic parser depends on the concepts of the *reflection* and *annotation,* which we discussed in the previous section. The *ZYXDocument* provides the DOM representation of ZYX model. i.e., the ZYX-DOM.

```
public class SMILZYXParser extends ZYXParser<SMILDocument> {
   public SMILZYXParser() {
      super(SMILDocument.class);
   }
}
```

Figure 4-18: The SMILZYXParser class

The implementations of the interfaces: (i) the *DocumentToRDFService* depicted in Figure 4-14, and (ii) the *RDFToDocumentService* depicted in Figure 4-15, are provided by the **ZYXToRDFServiceImpl** and **RDFToZYXServiceImpl,** respectively. Those implementations serve the conversion of ZYX-DOM to RDF and RDF back to ZYX-DOM. In chapter 3.4.2 we presented the mapping logic that we apply to transform OOP classes to RDF datasets.

The design of the *RDF-Graph* that is used internally in our approach can be described as follows:

- The multimedia document represents the root rdf:resource in the *RDF-Graph*.
- The root rdf:resource links five main rdf:resources by the predicates (*mediaElements*, *meta*, *layout*, *structure*, and *keywords*). We will refer to these resources in the rest of this chapter.
- Each of those rdf:resources will link other rdf:resources that encode a specific part of the multimedia document, e.g., the rdf:resource identified by the URI "someMultimediaDoc/layout" shown in Figure 4-19 and its relating predicate dcmc:layout, will link all the layout information provided by the layout part of the multimedia document. The same logic is used to link the relevant parts of the multimedia document to their corresponding rdf:resources.

Figure 4-19 provides an example of the first level of the internal data structure representing a multimedia document named "someMultimediaDoc" by means of RDF and showing the main five rdf:resources.

```
<rdf:Description rdf:about="someMultimediaDoc">
    <dcmc:mediaElements rdf:resource=" someMultimediaDoc/mediaElements"/>
    <dcmc:meta rdf:resource="someMultimediaDoc/meta"/>
    <dcmc:layout rdf:resource="someMultimediaDoc/layout"/>
    <dcmc:structure rdf:resource="someMultimediaDoc/structure"/>
    <dcmc:keywords rdf:resource="someMultimediaDoc/keywords"/>
</rdf:Description>
```

Figure 4-19: The first level of the internal data structure represented in RDF/XML syntax

4.5 Development of the Analyzer

4.5.1 Underlying concepts

The analysis process aims on the one hand at capturing the required information about the available media elements needed to compose a new multimedia content, and on the other hand at detecting the SOI. To achieve the required level of analysis, the analysis process makes use of two services, (i) *DocumentAnalysisService* and (ii) *SOIDetectionService*.

The *DocumentAnalysisService* represents the main entry point to the analysis process shown in Figure 4-20. It begins by running a global analysis on each multimedia document that the *RDF-Graph* contains. The results of the global analysis are added to the *RDF-Graph*. Once the global analysis is completed, the analysis process starts the local analysis looking at each detected individual media element.

The *DocumentAnalysisService* takes an RDF model as input and encodes the result of the analysis as part of the *RDF-Graph* by interlinking resulting data to the resources *meta* and *mediaElements*.

The overall document analysis is partitioned into three analysis subservices according to characteristics of media types:

(1) The *MetadataAnalysisService*, which performs high-level analysis by examining the metadata captured by means of standard schemes, such as EXIF and MPEG-7. It takes a media element as input and adds (interlinks) the discovered resources and literals to the resource *meta* in the *RDF-Graph*.

(2) The *FeatureAnalysisService* performs a low-level feature analysis aiming at detecting features such as color, edges, texture, shape, and other features form different media types. For each of those mentioned features there are different algorithms with different goals and complexity. The *FeatureAnalysisService* makes use of those algorithms to detect features and to identify objects that those features involve. It takes a media element as input and adds (interlinks) the discovered literals to the resource *keywords* in the *RDF-Graph.*

(3) The *TextAnalysisService* performs a text analysis aiming at identifying keywords that represent the subject of the multimedia documents. It makes use of text analysis algorithms, e.g., the term frequency (TF) and the inverse document frequency (IDF), or a combination of both resulting in TF-IDF weighting to identify keywords. It takes a media element of type text as input and adds (interlinks) the discovered literals to the resource *keywords* in the *RDF-Graph.*

The *SOIDetectionService* follows a rule-based approach to detect one or more subjects of interest. Hence, the *SOIDetectionService* provides a set of rules that can be applied to the *RDF-Graph*. In the implementation in section 4.5.3, we provide more details about these rules. The *SOIDetectionService* takes the *RDF-Graph* as input parameter. It makes use of the resources *keywords* and *meta* to detect one or more SOI and adds a new resource for each detected SOI. This newly added resource can be interlinked to additional resources and literals during subsequent processing steps.

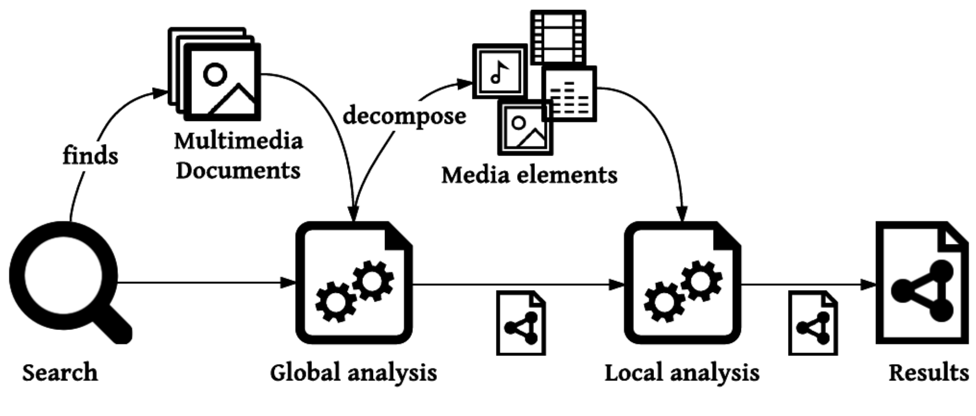


Figure 4-20: The analysis process

4.5.2 Design and specifications of the services and components

DocumentAnalysisService depicted in Figure 4-21 provides a generic interface and adopts the same concepts of generic programming we used for other components in our approach. The *DocumentAnalysisService* offers the function *analyze*. The *analyze* function takes an RDF model as input, the result of the analysis is stored in the source *RDF-Graph*. In Figure 4-1 we presented the main parts of the data structure of DCMC, which is encoded by means of RDF syntax.

The *DocumentAnalysisService* contributes to the *RDF-Graph* by linking resources to the *mediaElements* resource and the *meta* resource.

```
public interface DocumentAnalysisService<T> {
   public void analyze(T rdfModel);
}
```

Figure 4-21: The DocumentAnalysisService interface

The *MetadataAnalysisService* depicted in Figure 4-22 offers the function *addMetadata*, which accepts a media element of type <T> that can be of any type.

The *addMetadata* function contributes to the *RDF-Graph* by adding resources and literals to the resource *meta.*

```
public interface MetadataAnalysisService<T> {
   public void addMetadata(T mediaResource);
}
```

Figure 4-22: The MetadataAnalysisService interface

The *FeatureAnalysisService* depicted in Figure 4-23 offers the function *addDetectedFeatures*, which accepts a media element of type <T> that can be of any type.

The *addDetectedFeatures* function contributes to the *RDF-Graph* by adding literals to the resource *keyword.* Those added keywords can be used by the *SOIDetectionService* to detect one or more SOI.

```
public interface FeatureAnalysisService<T> {
   public void addDetectedFeatures(T mediaResource);
}
```

Figure 4-23 : The FeatureAnalysisService interface

The *TextAnalysisService* depicted in Figure 4-24 offers the two variants of the function *addKeywords*, which accepts a media element of type <T> that can be of any type.

The *addKeywords* function contributes to the *RDF-Graph* by adding literals to the resource *keyword.* Those added keywords can be used by the *SOIDetectionService* to detect one or more SOI.

```
public interface TextAnalysisService<T1, T2> {
   public void addKeywords(T1 rdfKeywordsRes, T2 text) throws IOException;
   public void addKeywords(T1 rdfKeywordsRes) throws IOException;
}
```

Figure 4-24: The TextAnalysisService interface

The *SOIDetectionService* presented in Figure 4-25 offers the function *detectSOI* that accepts an RDF model of type <T> that can be of any type. The *detectSOI* function makes use of two resources that are linked to the root resource to detect the SOI, those two resources are the resource *keyword* and the resource *meta.*

The *SOIDetectionService* contributes to the *RDF-Graph* by adding for each detected SOI an rdf:resource that can be in following phases linked to additional resources and literals.

```
public interface SOIDetectionService<T> {
   public void detectSOI(T rdfModel);
}
```

Figure 4-25: The SOIDetectionService interface

4.5.3 Implementation of the services and components

The *DocumentAnalysisService* interface is implemented by **DocumentAnalysisServiceImpl** class. It represents the main entry point to the analysis service and process. The **DocumentAnalysisServiceImpl** class provides an implementation for the main analysis function called *analyze.* Since the **DocumentAnalysisServiceImpl** class uses the Jena RDF model class as the type of the generic type <T>, the *analyze* function accepts an *instance* of Jena RDF model class as its input parameter. The analysis process presented in Figure 4-20 begins by running the global analysis on each multimedia document that the RDF model includes.

DocumentAnalysisServiceImpl class contributes to the *RDF-Graph* by linking resources to the resource *mediaElements* and the resource *meta*.

Figure 4-26 provides an example of the results of the global analysis by showing the individual media elements that a document named "Tiger.smil" contains. Those individual media elements are linked by the global analysis process as rdf:resources to the resource *mediaElements* of the source document.

```
<rdf:Description rdf:about="http://univie.ac.at/dcmc/Tiger.smil/mediaElements">
    <dcmc:mediaElement
        rdf:resource="http://univie.ac.at/dcmc/Tiger.smil/Tiger.txt"/>
    <dcmc:mediaElement rdf:resource="http://univie.ac.at/dcmc/Tiger.smil/
     Tiger_in_Ranthambhore.JPG"/>
</rdf:Description>
```

Figure 4-26: The list of media elements represented as RDF-Graph

Once the global analysis is completed, the analysis process starts the local analysis process on each detected media element. Media elements can be detected by applying the rule *ruleMediaElements* depicted in Figure 4-27.

```
[ruleMediaElements:
      dcmc:hasMediaElements(?srcDoc, ?mediaElements) AND
      dcmc:hasMediaElement(?mediaElements, ?mediaElement)
       => dcmc:hasMediaElement(?srcDoc, ?mediaElement)]
```

Figure 4-27: The rule: ruleMediaElements

The execution of the rule *ruleMediaElements* provides the local analysis process with a list of RDF triples that have in their *subject* part the source multimedia document and in the *object* part the name of the individual media element, both parts are linked with the relationship "*hasMediaElement*". The local analysis process makes use of the implementation of the following three subservices, as discussed in the previous section:

The *MetadataAnalysisService* interface is implemented by the **ImageAnalysisServiceImpl** class, which provides the function *addMetadata*. The *addMetadata* accepts an rdf:resource as its input parameter, the rdf:resource represents a media element resource of type image. Our current implementation reads the EXIF data of the images by using a class that we have implemented in Java.

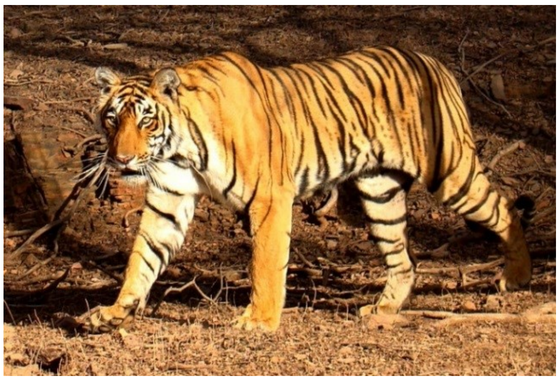
The *addMetadata* function contributes to the *RDF-Graph* by linking the extracted EXIF data to the metadata related to the source media element.

The *MetadataAnalysisService* can be implemented to add support to several metadata schemes (e.g., MPEG-7 and DCMI) and several types of media elements such as audio and video. Such additional implementations widen the range of metadata analysis provided by the DCMC.

FeatureAnalysisService interface is implemented by the **FeatureAnalysisServiceImpl** class, which provides the function *addDetectedFeatures.* As presented in the previous section, there are many algorithms that support the low-level feature analysis. For the purposes of our prototype, we used the image analysis service provided by the Clarifai project [Clarifai 2017].

The *addDetectedFeatures* function contributes to the *RDF-Graph* by linking the recognized features to the resource *keywords.* Those detected features relate to the source media element. Figure 4-28 presents the detected keywords about the shown media element resource which is called "Tiger_in_Ranthambhore.JPG".

```
<rdf:Description rdf:about="http://.../
Tiger_in_Ranthambhore.JPG/keywords">
    <dcmc:keyword>wildlife
    </dcmc:keyword>
    <dcmc:keyword>mammal</dcmc:keyword>
    <dcmc:keyword>tiger</dcmc:keyword>
    <dcmc:keyword>safari
    </dcmc:keyword>
    <dcmc:keyword>predator
    </dcmc:keyword>
    <dcmc:keyword>carnivore
    </dcmc:keyword>
    <dcmc:keyword>wild</dcmc:keyword>
</rdf:Description>
```



[Wikimedia 2017b]

Figure 4-28: List of keywords as RDF-Graph

TextAnalysisService interface is implemented by **TextAnalysisServiceImpl** class. Text can be added to multimedia documents either by including the text element directly in the body of the document or by using a reference such as the "*src*" attribute to refer to an external text file. The *TextAnalysisService* defines two variants of the function *addKeywords*, *addKeywords(T1 rdfKeywordsRes)* and *addKeywords(T1 rdfKeywordsRes, T2 text)*. The *addKeywords(T1 rdfKeywordsRes, T2 text)* function adds support to read the text provided by means of external text files. Both functions provide text analysis to identify keywords that represent the text. For the purposes of our prototype, we provided a simple implementation of the TF-IDF term weighting.

Similar to the *addDetectedFeatures* function provided by the interface *FeatureAnalysisService*, the *addKeywords* functions contribute to the *RDF-Graph* by linking to the resource *keyword* those recognized terms, that relate to the source media element.

The *SOIDetectionService* interface is implemented by two classes, these are the **TermSOIDetectionServiceImpl** and the **RuleBasedSOIDetectionServiceImpl.** We provide two implementations of this service to enable us to evaluate the results that we get when the rule-based approach is used to detect the SOI. In other words, the SOI analysis results provided by **TermSOIDetectionServiceImpl** class represent the control data that we can use to evaluate the results provided by the **RuleBasedSOIDetectionServiceImpl** class**.**

Both services use the resources *keywords* and *meta* that the interface *DocumentAnalysisService* and its different sub-services have added to the *RDF-Graph*. Both services provide an implementation for the function *detectSOI*.

In **TermSOIDetectionServiceImpl** class we followed the term frequency approach to identify the SOI. To narrow down the number of the detected SOI, we provided a configurable value for a threshold, SOI which satisfy the defined threshold can be added to the SOI list. Once the list of SOI is created, additional steps will be performed to create a RDF subgraph for each SOI in the list.

The **RuleBasedSOIDetectionServiceImpl** class follows a different approach by providing rules to detect the SOI. Figure 4-29 lists the rules that we have defined to detect the SOI.

```
[ruleMetadata: dcmc:meta (?root, ?metadata) AND (?metadata ?p ?metaValue)
 => dcmc:keyword (?root, ?metaValue) ]
[createSOI: dcmc:keyword (?root, ?metaValue) => createSOI(http://dcmc/sois,
?metaValue)]
[rateSOI: dcmc:soi(http:/dcmc/sois, ?soi) => rateSOI(?soi, dcmc:keyword, 1)]
[resourcesSOI: dcmc:keywords(?mediaElement, ?keywords) AND
               dcmc:keyword (?keywords, ?keyword) AND
               dcmc:soi (http://dcmc/sois, ?soiResource) AND
               dcmc:rating (?soiResource, ?rating) AND
               greaterThan(?rating, 1) AND
               soiEqual(?soiResource, ?keyword)
 => dcmc:mediaElement(?soiResource, ?mediaElement)]
```

Figure 4-29: Set of rules to detect one or more SOI

A rule consists of two parts, the antecedent and the consequent, more about the rules will be presented in section 4.9.1 below. Both parts of a rule can contain a list of elements, those elements can either be a triple of the form of (?S ?P ?O), a functor in form of $f(P)$, or another rule. In the course of defining rules for our rule-based SOI detection, we provided implementation for functors such as *rateSOI* and *soiEqual* that serve the SOI detection in DCMC, e.g., Figure 4-43 in section 4.9.3 presents the implementation of the functor *soiEqual.*

4.6 Development of the Selector

4.6.1 Underlying concepts

The *Selector* is responsible for selecting those media elements required for the composition, which correspond to a SOI. The preceding two phases of the *Analysis* and *Enrichment* contribute to the *RDF-Graph* by providing one or more SOI and a list of media elements that relate to those SOI. Those media elements stem from the analyzed multimedia documents and from the results of the enrichment process. The first step in the selection process is to find those media elements that are related to the set of SOI. The second step is to select the most relevant media element, which can be included in the *Composition* phase. In order to determine which media elements are more relevant with respect to the set of SOI, we use a Feature Vector Based Distance Measure [Grimnes et al. 2008] to find the distance between a SOI and a linked media element. Media elements found to have a distance within a specific threshold will be selected. As a basic rule to provide new information in addition to the original documents, original text elements will be selected only if no new text elements were retrieved from LOD resources during the *Enrichment* phase. Figure 4-30 represents the steps of the selection process and the *Selection* phase as part of the enhanced authoring process.

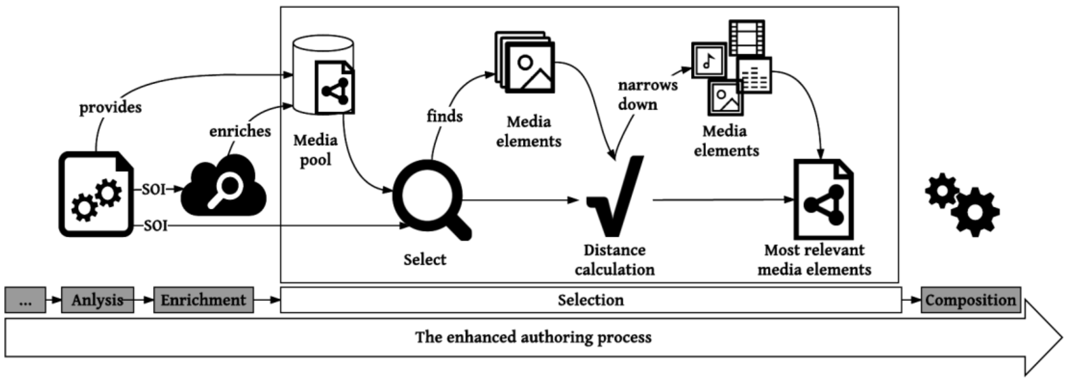


Figure 4-30: The Selection process

4.6.2 Design and specifications of the services and components

MultimediaElementSelectionService as depicted in Figure 4-31 provides a generic interface and adopts the same concepts of generic programming we used in other parts of the DCMC. The *MultimediaElementSelectionService* interface provides the function *selectMediaElements*. The *selectMediaElements* function expects an RDF model as input parameter, the result of the selection is stored in the source *RDF-Graph*. *MultimediaElementSelectionService* contributes to the *RDF-Graph* by linking those selected media elements to the resource that represents the SOI.

```
public interface MultimediaElementSelectionService<T1> {
   public void selectMediaElements(T1 rdfModel);
}
```

Figure 4-31: The MultimediaElementSelectionService interface

Figure 4-32 represents an example of the corresponding part of the *RDF-Graph* showing a set of media elements and a rating element linked to a SOI identified by the *subject* "tiger". The two elements defined by dcmc:mediaElement link another two rdf:resources that are represented by an image and a text. The dcmc:rating element provides and integer value that represent the raring of this SOI. This value of the has been set in the *Analysis* phase.

```
<rdf:Description rdf:about="http://univie.ac.at/dcmc/soi/tiger">
    <dcmc:mediaElement rdf:resource="http://univie.ac.at/dcmc/Tiger.smil/
     Tiger_in_Ranthambhore.JPG"/>
    <dcmc:mediaElement
        rdf:resource="http://univie.ac.at/dcmc/Tiger.smil/Tiger.txt"/>
    <dcmc:rating
        rdf:datatype="http://www.w3.org/2001/XMLSchema#int">3</dcmc:rating>
</rdf:Description>
```

Figure 4-32: Media elements linked to a SOI

The *DistanceCalculationService* interface depicted in Figure 4-33 provides the function *calculateDistance*. The *calculateDistance* function expects a parameter that represents the SOI. The type of the SOI entity is represented by the generic type <T>. The function *calculateDistance* implements the following logic to calculate the distance:

1. It creates a subgraph where the root element of the subgraph is the SOI. The subgraph can be created by using a dedicated algorithm such as the Concise Bounding Description [W3C Org. 2014a].
2. It calculates a feature vector [Grimnes et al. 2008] for each resource in the subgraph.
3. It uses the calculated feature vector to calculate the distance between the SOI and each element in the media elements list.

```
public interface DistanceCalculationService<T> {
   public void calculateDistance(T soiResource);
}
```

Figure 4-33: The CalculateDistanceService interface

4.6.3 Implementation of the services and components

We provided three different implementations for the interface *MultimediaElementSelectionService*. With these three variants we aim at observing the results of using these different implementations for the logic of the selection process. These different selection processes are summarized in the following list:

- A selection logic that selects elements linked to several SOI.
- A selection logic that selects elements linked to the highest rated SOI.
- A selection logic that selects elements linked to a randomly chosen SOI.

The **DefaultSOISelectionServiceImpl** class represents the default behavior of the DCMC and implements the first selection logic presented in the list. The **DefaultSOISelectionServiceImpl** class finds those relevant media elements for all SOI in a shortened list of SOI. This shortened list of SOI contains only those SOI with a rating greater than a minimum value. This minimum value is a system configurable variable.

The **SOISelectionServiceImpl** class implements the second selection logic presented in the list. It finds those relevant media elements for the highest rated SOI. The rating of the SOI is provided by the *Analyzer* based on the implementation of the *SOIDetectionService* presented in section 4.5.3.

The **RandomSOISelectionServiceImpl** class implements the third selection logic presented in the list. It finds those relevant media elements for a randomly selected SOI.

The *DistanceCalculationService* interface is implemented by the **FeatureVectorDistanceServiceImpl** class which provides the function *calculateDistance*. The *calculateDistance* function accepts a SOI of type RDF resource as input parameter. The *calculateDistance* function uses the **CBDImplementation** class that provides the function *cbdImpl,* the *cbdImpl* calculates the subgraph of a given SOI. The **FeatureVectorService** class provides the function *createFeatureVector* that is used to create the feature vector.
The **DistanceCalculator** class offers the function *calculateDistance* that is used to calculate the distance between the SOI and the media element.
The services and components of the *Selector* contribute to the *RDF-Graph* by linking media elements that relate to a SOI.

4.7 Development of the LOD API framework

4.7.1 Underlying concepts

Semantic Web and Linked Open Data standards provide specifications for data providers on how to create their services and represent their data. Data providers use different concepts for their endpoints and data representation schemes. The design of endpoints influences mainly the method of querying those endpoints and the response of those endpoints which can be represented in different data schemes. Hence, the *LOD API framework* for the DCMC must provide a generic design for its services and components to allow for communicating with various LOD APIs. Retrieving data from LOD APIs involves an interaction of (request/response) form. Data providers specify how their APIs can be called, and how the response of the call is represented. Based on these specifications the *LOD API framework* provides four generic modules to communicate with LOD APIs, these are: i) the Query API, ii) the Parser API, iii) the Mapping API, and iv) the HTTP API.

4.7.2 Design and specification of the framework

The *SOIEnrichmentService* interface provides the entry point for the enrichment process in DCMC. The design of the service follows that same generic concepts we used in our system. The interface *SOIEnrichmentService* provides the function *enrich* that accepts two parameters: i) one of type <T1> representing the SOI to the enrichment process, which can be used to construct the request. ii) one type of <T2> representing the resource to which the result of the enrichment process is to be linked. Figure 4-34 depicts the *SOIEnrichmentService*.

```
public interface SOIEnrichmentService<T1, T2> {
   public void enrich(T1 soi, T2 result);
}
```

Figure 4-34: The SOIEnrichmentService interface

The interface *SOIEnrichmentService* communicates with internal world of the DCMC system, to communicate the external world, i.e., the LOD APIs, the *LOD API framework* provides four generic and abstract components as presented earlier.

The Query API allows for constructing the query. It allows for implementing a generic Query provider that supports querying a certain LOD API. The query could be either a SPARQL query or a GET/POST call to a RESTful web service. Generally, the **Query API** supports any query type that can be transmitted over the HTTP protocol.

The Parser API allows to parse a LOD source document into internal objects. The **Parser API** follows that same concepts of *reflection* and *annotation* we presented in section 4.4.1. Hence, the two generic classes *XMLParser* and *JSONParser* are provided.

The Mapping API allows to map the parsed objects that represent the retrieved LOD document to the internal data model of the DCMC, i.e., to RDF triples.

The HTTP API allows for establishing connections over HTTP in order to send and receive data. This component builds on top of the *HTTPService* that the *NtF* provides and that has been presented in section 4.3.2.2.

4.7.3 Implementation of the framework and its components

For the purposes of a demonstration of the DCMC prototype we implemented the interfaces and the abstract classes of the *LOD API framework* to support DBpedia [DBpedia Org. 2017] and GeoNames [GeoNames 2017].

The **DBpediaEnrichmentServiceImpl** class implements the presented *SOIEnrichmentService* interface. The function *enrich* accepts two parameters: The first parameter is a string value that represents the SOI, the second parameter represents the RDF part that represents the SOI resource. The DBpedia API can be queried by means of SPARQL. Hence, the presented *QueryProvider* is implemented to construct the SPARQL query. Figure 4-35 provides an example of a sample query about a SOI that is represented by the *subject* "Lion". The query selects any property and its associated value that are linked to the resource "Lion". The query adds a language filter to narrow down the results to those values of properties provided in the English language.

```
http://dbpedia.org/sparql?default-graph-uri=
http://dbpedia.org&should-sponge=&format=xml&debug=on&timeout=&query=
SELECT ?property ?hasValue
WHERE {
   <http://dbpedia.org/resource/Lion> ?property ?hasValue
   FILTER (lang(?hasValue) = "en" )
}
```

Figure 4-35: SPARQL to request data from DBpedia API

The response of the call to DBpedia endpoint is a document about the requested resource represented by means of XML syntax. The **DBpediaResultParser** class extends the generic *XMLParser* class to enable de-serializing the results of calling the endpoint of DBpedia to an object model. The **DBpediaResultMapper** class provides the mapping logic to map the parsed results to the internal data model that is represented by means of RDF.

The **DBpediaEnrichmentServiceImpl** class contributes to the *RDF-Graph* by linking the SOI rdf:resource to the retrieved data by using the relation "*enrichment*".

The **GeoNamesEnrichmentServiceImpl** class implements the previously presented *SOIEnrichmentService* interface. GeoNames API can be queried by providing geo-coordinates. Hence, the function *enrich* accepts two parameters, the first parameter is a duple that provides the values of the geographic longitude and latitude representing the SOI, the second parameter provides the RDF part that represents the SOI resource. The rest of the components are implemented in a similar way as we did in the **DBpediaEnrichmentServiceImpl** class**.**

4.8 Development of the Composer

4.8.1 Underlying concepts

The *Composer* aims at creating the multimedia document, which is represented by means of RDF graph. The *Composer* makes use of the following input data:

- A set of selected media elements.
- The *RDF-Graph* that includes a representation of the original documents.
- Composition rules.
- Optionally, a set of multimedia document composition templates.

On the one hand, DCMC makes use of ZYX as an abstract and internal model for creating and manipulating multimedia documents. On the other hand, ZYX model distinguishes between the *structure* of the document and the *layout*. Hence, the

Composer incorporates several services for the composition of the multimedia documents in order to maintain a well-defined and well-structured authoring process.

The design of the *RDF-Graph* that we presented in section 4.4.3 encompasses five main resources: *mediaElements*, *meta*, *layout*, *structure*, and *keywords*. These resources are directly linked to the root resource, i.e., the multimedia document. Additional resources can be added cumulatively to the *RDF-Graph* by each phase of the authoring process. In the *Composition* phase however, the *Composer* creates a new root resource that represents the new multimedia document. This created *RDF-Graph* must adopt the same design specifications provided by the DCMC to encode multimedia documents by means of RDF.

4.8.2 Design and specifications of the services and component

The *Composer* aims at creating the multimedia document by defining the temporal and spatial relationships for the selected media elements, which are represented by the resource *mediaElements* in the *RDF-Graph*. The *Composer* comprises the following services:

(i) The interface *PresentationService*, which creates the RDF representation for the new multimedia document and adds the resources *keywords*, *metadata*, and *mediaElements*.

(ii) The interface *StructureCompositionService* adds the temporal relationships between the elements such as *parallel* and *sequential*.

(iii) The interface *LayoutCompositionService* adds the spatial relationships between the media elements to the RDF representation.

The *PresentationService* interface depicted in Figure 4-36 provides the function *createMultimediaPresentation* that creates the root resource of the new multimedia document. The function *createMultimediaPresentation* accepts a SOI resource of type<T>. The created root resource provides the main five resources of multimedia document, as mentioned above. Hence, the interface *PresentationService* contributes to the *RDF-Graph* by adding a new root resource. It contributes as well by linking related rdf:resources to the resources *mediaElements*, *meta*, and *keywords*.
The *structure* and *layout* will be contributed by the interfaces *StructureCompositionService* and *LayoutCompositionService*, respectively.

```
public interface PresentationService<T> {
  public void createMultimediaPresentation(T rdfModel);
}
```

Figure 4-36: The PresentationService interface

The interface *StructureCompositionService* depicted in Figure 4-37 provides the function *addStructure*. The function *addStructure* expects the root resource of the created multimedia document. It uses the composition rules to create the *structure* of the document. The *structure* of the document defines the temporal relationships between the different multimedia elements that are included in the composition.

```
public interface StructureCompositionService <T> {
   public void addStructure(T resourceDoc);
}
```

Figure 4-37: The StructureCompositionService interface

The interface *LayoutCompositionService* depicted in Figure 4-38 provides the function *addLayout*. Similar to *StructureCompositionService* interface above, the *addLayout* function expects the root resource of the created multimedia document. It uses the composition rules to create the *layout* of the document. The *layout* of the document defines the spatial relationships between the different multimedia elements that are included in the composition.

```
public interface LayoutCompositionService<T> {
   public void addLayout(T resourceDoc);
}
```

Figure 4-38: The LayoutCompositionService interface

4.8.3 Implementation of the services and components

The interface *PresentationService* creates a resource that represents the new multimedia document. It takes a SOI resource as input parameter. The created resource interlinks the main resources presented in section 4.2, i.e., *mediaElements*, *meta*, *layout*, *structure*, and *keywords*.

The resource *mediaElements* interlinks the media elements that relate to the SOI. Those media elements stem either from another multimedia document, the content provided by the user, or from the Web of Data.

The resources *meta* and *keywords* link additional information about the newly composed document and its individual media elements. This additional information is detected by the interface *DocumentAnalysisService* and its sub-services as presented in section 4.5.

The *PresentationService* interface is implemented by the **PresentationServiceImpl** class that provides the function *createMultimediaPresentation*. The *createMultimediaPresentation* function expects as input an RDF resource that represents the SOI. It creates a resource that represents the new multimedia document and links it to the SOI resource. The newly created resource links the five parts of the multimedia document as presented above. The **PresentationServiceImpl** class links the available information about the SOI and the media elements that will be included in the composition to the resources *mediaElements*, *meta*, and *keywords*, whereas *structure* and *layout* will be handled by *StructureCompositionService* and *LayoutCompositionService*, respectively. Figure 4-19 depicts the root structure of the multimedia document represented by means of *RDF-Graph*.

The *mediaElements* resource links those media elements associated with the SOI that the *Selector* has selected in the preceding phase of the authoring process, i.e., in *Selection* phase. Those media elements stem from another multimedia document, the content provided by the user, or from web resources.

The resources *meta* and *keywords* link additional information about the newly composed document and its media elements. That additional information is provided by the analysis process. It includes the results provided by the *DocumentAnalysisService* interface and its sub-services.

The interface *PresentationService* contributes to the *RDF-Graph* by adding a new resource representing the new multimedia document which comprises content related to the given SOI. It contributes as well by interlinking resources that relate to the resources *mediaElements*, *meta*, and *keywords*. Resources that relate to *structure* and *layout* will be contributed by the interfaces *StructureCompositionService* and *LayoutCompositionService,* respectively.

The interface *StructureCompositionService* expects the root resource of the created multimedia document as input. It uses composition rules to create the *structure* of the document. In ZYX model the *structure* of the document defines the temporal relationships between the multimedia elements. Temporal relationships can be described by means of *parallel* and *sequential*. The specification of *parallel* implies that all media elements are to be presented simultaneously. The specification of *sequential* implies that media elements are to be presented consecutively. Of course, each media element either has associated a start-time and an end-time, or start-time and duration of playback.

The interface *StructureCompositionService* is implemented by the **StructureCompositionServiceImpl** class that provides the function *addStructure*. The function *addStructure* expects a resource as an input parameter. This resource represents the multimedia document that the interface *PresentationService* has created. The *addStructure* aims at creating the temporal relationships of the media elements that are linked to the resource *mediaElements* of the created multimedia document.

The **StructureCompositionServiceImpl** class makes use of the three groups of rules as defined by the interface *StructureCompositionService*, i.e., the *structural*, *temporal* and *sub-temporal rules*.

Structural rules allow to infer an identifier (ID) based on a *possible combination* of types of the media elements interlinked with the resource *mediaElements.* The inferred ID represents the *structure* rule that will be used to create the temporal relationships between those media elements. For each *possible combination* and regardless of the actual number of the media elements of each type in the resource *mediaElements*, we defined the most appealing *structure* that fits to each certain combination in the context of our approach. Based on the inferred ID, further *temporal* rules can be applied to provide more structuring granularity (structuring template).

Table 4-1 depicts the *possible combinations* of the types of media elements, a (+) denotes the occurrence of at least one media element of a specific type, a (-) denotes no occurrence, e.g., the combination given in the 1st row suggests that the resource *mediaElements* contains one or more elements of each of the types Text and Image, and zero elements of the types Audio and Video. For each combination expressed in a row an identifier ID is assigned, e.g., Structure8 for the combination in the 8th row, which corresponds to a *structural rule* (e.g., equalstructure8) that tests a concrete multimedia document against the occurrence of that specific combination of media types. In addition, each combination has assigned a group of structuring templates, which provides alternative instructions for creating the temporal relationships of the involved media elements (see Table 4-2).

row	*Text*	*Image*	*Audio*	*Video*	*ID*	*Group of Structuring Templates*
1	+	+	-	-	Structure1	Temporal1
2	+	-	+	-	Structure2	Temporal1
3	+	-	-	+	Structure3	Temporal1
4	+	+	+	-	Structure4	Temporal2
5	+	+	-	+	Structure5	Temporal1
6	+	-	+	+	Structure6	Temporal1
7	+	+	+	+	Structure7	Temporal3
8	-	+	+	-	Structure8	Temporal4
9	-	+	-	+	Structure9	Temporal5
10	-	+	+	+	Structure10	Temporal5
11	-	-	+	+	Structure11	Temporal1
…	…	…	…	…	…	…

Table 4-1: Possible combinations of the types of media elements

For example, the 8th row in Table 4-1 can be interpreted as follows:

IF no elements of type ***Text***
AND one or more elements of type ***Image***
AND one or more elements of type ***Audio***
AND no elements of type ***Video***
THEN the ***ID*** **is** ***Structure8*** **and** ***Group of Structuring Templates*** **for creating the temporal relationships is** ***temporal4*****.**

Please note, Table 4-1 does not show four special cases: If the resource *mediaElements* contains only a single kind of media element type, i.e., only one of text, image, audio, or video, those special mono-media element configurations are treated by dedicated default templates.

Table 4-1 refers to the group of structuring templates, which provide a more detailed specification for handling different combinations depending on the concrete number of media elements present. The templates specified in such a group, e.g., *Temporal4*, see Table 4-2, guide the creation of temporal relationships used for the composition of the media elements. Each template tries to implement the most appealing structure for the composition. Of course, these templates can be designed according to individual needs of an application.

Table 4-2 depicts eight cases of template variants of *possible combinations of occurrences of media elements* that we have defined in the course of our demonstration application.

The column "Structuring Template" in Table 4-2 encodes the templates that will be used to create the temporal relationships.

No of images	***No of audios***	***Audio linked to images***	***Structuring Template***	***case***
1	1	True	composeSubTemporalPar: image and audio are placed in one slide	4A
1	1	False	composeSubTemporalSeq: image and audio are placed in two slides	4B
1	> 1	True	composeSubTemporal41: - Image and linked audio/s parallel in one slide - remained audios if any sequential each of which in a slide.	4C
1	> 1	False	composeSubTemporalSeq	4D
> 1	1	True	composeSubTemporal42: - One parallel element that contains a sequential element for the image(s) providing a slideshow, while the linked audio is played in parallel. - the remained unlinked media elements are displayed sequentially in a slideshow presentation	4E
> 1	1	False	composeSubTemporalSeq	4F
> 1	> 1	True	composeSubTemporal42	4G
> 1	> 1	False	composeSubTemporalSeq	4H

Table 4-2: The group of structuring Templates "Temporal4" – possible combinations

The template *composeSubTemporalPar* arranges the media elements in, e.g., one presentation slide, those elements are displayed simultaneously. The template *composeSubTemporalSeq* arranges the media elements in, e.g., presentation slides, each element has its own slide, and those slides are presented sequentially. The templates *composeSubTemporal41* and *composeSubTemporal42* arrange the media elements as described in more detail in the table.

The main part of the semantics of the composition logic is implemented by means of a rule-based reasoner that supports inference over RDF graphs. Our current implementation is based on the Jena general purpose rule engine [Apache Org. 2017c]. This includes all the structural rules (covering the logic indicated by Table 4-1) as well as rules for the realization of the groups of structuring templates as indicated, e.g., by Table 4-2.

In the following, we provide a detailed illustrating example using the combination given in the 8th row of Table 4-1 with the ID *Structure8* and referring to the group of structuring templates *Temporal4*. Figure 4-39 shows the rules which we defined to handle the combination with ID *Structure8,* and the specific rule for case 4E (see Table 4-2) in the group of structuring template *Temporal4* that can be applied to construct the temporal relationships of the media elements. This case 4E is characterized by more than one image and one audio file. This one audio file narrates these images.

```
// rule corresponding to ID Structure8
[equalStructure8:
      dcmc:mediaElements(?mmDoc, ?mediaElements)  AND
      equalStructure8(?mediaElements)
       =>  subTemporal4(?mmDoc dcmc:usesSubTemporal)
 ];
// rules corresponding to group of templates Temporal4
// … rules for cases 4A, 4B, … 4H go here
[rule_4E:
       dcmc:usesSubTemporal(?mmDoc, ?subTemporal) AND
       dcmc:mediaElements (?mmDoc, ?mediaElements) AND
       dcmc:structure (?mmDoc?, structure) AND
       equal(?subTemporal,'Temporal4-2)
       => composeSubTemporal42(?structure, ?mediaElements)
];
…
```

Figure 4-39: Sample rules to detect the structure ID Structure8 and to compose according to Group Temporal4

We provided implementations for functors such as *equalStructure8, subTemporal4,* and *composeSub-Temporal42* that serve creating the structure of the multimedia document.

The rule *equalStructure8* uses the functor *equalStructure8,* which evaluates to *true,* if the combination that corresponds to ID *Structure8* is fulfilled, i.e., the resource ?*mediaElements* contains one or more elements of the types *image* and *audio*, and no element of the types *video* and *text*. As a consequence of the rule *subTemporal4* is triggered, which adds a new assertion to the *RDF-Graph* encoding the template to be used for the composition. This is done by examining the number of occurrences of media elements of a specific type, e.g., as specified in Table 4-2 for case 4E, and asserting that in this case the multimedia document is to be composed following 'Temporal4-2".

Subsequently, since rule_4E evaluates to true, the composition is realized according to the template assigned. The functor *composeSubTemporal42* adds the media elements according to the corresponding template to the *RDF-Graph*, interlinking it with the resource *structure*. Figure 4-40 illustrates the *structure* that the *composeSubTemporal42* creates.

Finally, we apply the same logic we presented above about *structure8* to handle the other *structure, temporal,* and *sub-temporal rules*, i.e., to cover all the combination we presented in Table 4-1.

The *StructureCompositionService* contributes to the *RDF-Graph* by adding the structure information of the multimedia document to the resource *structure*. The resource *structure* represents the temporal relationships of the media elements.

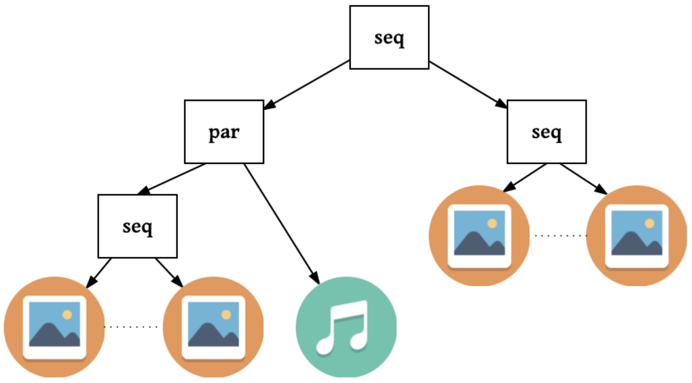


Figure 4-40: Structure created by *composeSubTemporal42*

The interface *LayoutCompositionService* - similar to the interface *StructureCompositionService* presented above - takes the root resource of the created multimedia document as input parameter. It uses composition rules to create the *layout* of the document. In ZYX model, the *layout* of the document defines the spatial relationships between the multimedia elements that are included in the composition. Hence, this service aims at creating the spatial relationships of the media elements that the resource *mediaElements* of the created multimedia document includes. *Layout* information specifies the position of a media element on the display. In ZYX model, a presentation element has one projector element attribute. This attribute points to a presentation area within the *layout* of the presentation. This presentation area has attributes that describe its position on the display. Our approach uses an absolute positioning system to create the *layout*. The absolute positioning system uses x and y positions or distance based on a fixed point in the presentation, e.g., left upper corner of the screen.

The interface *LayoutCompositionService* follows a similar approach to the one we used to create the *structure* of the document, i.e., we defined rules in order to conclude the layout of the document.

We provided two different implementations for the interface *LayoutCompositionService*. Those are the **TemplateCompositionServiceImpl** class and the **FreeStyleCompositionServiceImpl** class. Both classes provide the function *addLayout*. The *addLayout* function expects a resource as an input parameter. This input resource represents the multimedia document that the interface *PresentationService* has created. The function *addLayout* aims at creating the spatial relationships of the media elements that are linked to the resource *mediaElements* of the created multimedia document. *Layout* information specifies the position of a media element on the display.

The **TemplateCompositionServiceImpl** class makes use of layout templates, which we created for this service. Those templates correspond to one of the *possible combinations* of the structure as presented in Table 4-1. As an example, Figure 4-41 depicts the template that corresponds to the combination of ID *Structure8*, group of structural templates *Temporal4,* and the structuring template *composeSubTemporal4-2*, which we presented above. The template encompasses two slides that are presented sequentially, those are Slide1 and Slide2. Slide1 presents in parallel two presentation elements, a sequential element that presents the images in a slideshow, and an audio element that is played in parallel with the images. Slide2 presents image items sequentially.

The **FreeStyleCompositionServiceImpl** class follows a different approach to create the layout of the presentation. The *mediaElements* contains elements that stem either from another multimedia document or new elements retrieved from the LOD servers. Media elements from another document have layout information.

The **FreeStyleCompositionServiceImpl** class makes use of this information as a starting point to compose the layout of the presentation.

In the course of creating the structure as presented in the *StructureCompositionService*, the relation *usesSubTemporal* is defined, this relation provides information about the structure of the document. The **FreeStyleCompositionServiceImpl** class detects how many slides the presentation contains, and creates a layout for each slide considering the included media elements.

The interface *LayoutCompositionService* contributes to the *RDF-Graph* by adding the layout information of the multimedia document to the resource *layout*. The resource *layout* represents the spatial relationships of the media elements.

To combine both interfaces *StructureCompositionService* and *LayoutCompositionService* in the above illustrating example, we assume that the resource *mediaElements* contains 4 images and 1 audio. Two images are linked to the audio, the other two are not. Given this input information, the rule *equalStructure8* is fulfilled resulting in triggering *subTemporal4*. The *subTemporal4* examines the *possible combinations* depicted in **Table** 4-2, e.g., case 4E, which finally results in using *composeSubTemporal42*. The upper part of Figure 4-41 illustrates the structure that the *composeSubTemporal42* creates. The lower part shows the layout of the document.

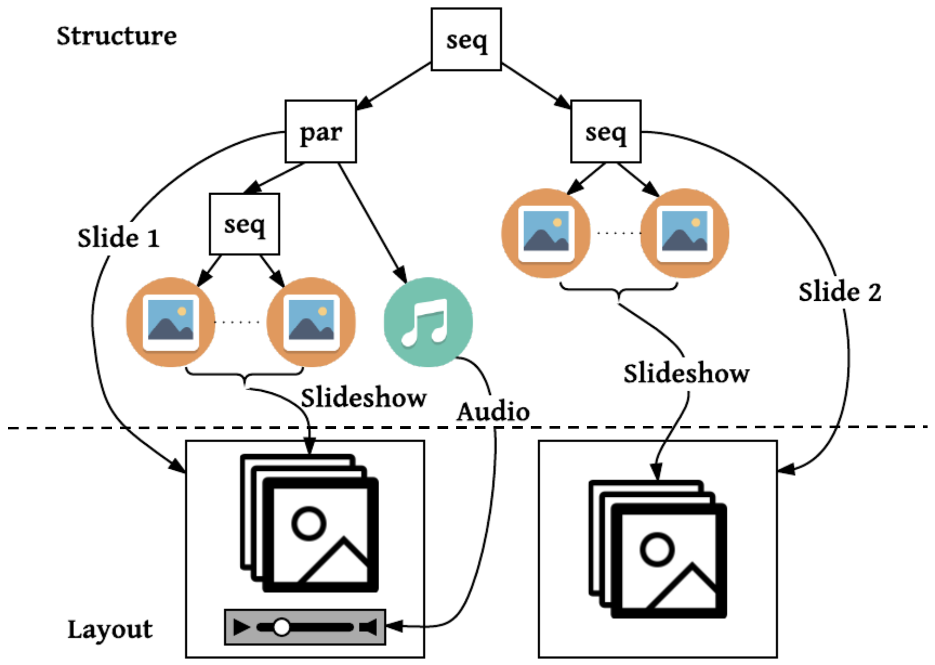


Figure 4-41: Template for Structure8 – Temporal4-2

4.9 Development of the Inference Engine

4.9.1 Underlying concepts

The Semantic Web Rule Language (SWRL) [W3C - SWRL 2010] provides a definition and specifications for the rule language. A rule consists of an antecedent (body) and a consequent (head). A rule can be read as follows: if the conditions specified in the antecedent is true, then the conditions specified in the consequent must also be true. Both the antecedent (body) and consequent (head) consist of zero or more atoms. Multiple atoms are treated as a conjunction. Atoms in these rules can be of the form C(x), P(x,y), sameAs(x,y) or differentFrom(x,y), where C is an OWL description, P is an OWL property, and x,y are either variables, OWL individuals, or OWL data values [W3C - SWRL 2010].

4.9.2 Design and specification of the component

The *InferenceEngineService* depicted in Figure 4-42 provides a generic interface that can be implemented to support a specific type of RDF model and Rules. The function *run(T1 model, T2 rules)* that the interface provides accepts two parameters. Those are an RDF model of type <T1> and set of Rules of type <T2>. This generic interface allows us to provide multiple implementations for the *InferenceEngineService*, each implementation can bind a specific Inference Engine and rule format. The result of applying the rules on the RDF model is added to the model.

```
public interface InferenceEngineService<T1, T2> {
   public void run(T1 model, T2 rules);
}
```

Figure 4-42: The InferenceEngineService interface

The *Inference Engine* component adds custom functions that can be used either in the body or in the head of the rule. Those custom functions support the inferencing logic that the authoring process of DCMC applies to accomplish its sub-tasks. Figure 4-43 presents the functor *SOIEqual* as an example of those custom functions. The SOIEqual function can be used only in the body of the rule. It examines whether two SOI are equal or not.

4.9.3 Implementation of the services and components

The interface *InferenceEngineService* is implemented by the class *InferenceEngineServiceImpl*. The *Inference Engine* in DCMC is built on top of Jena framework [Apache Org. 2017b]. Hence, the **InferenceEngineServiceImpl** class uses the Jena RDF Model type for <T1> and a String type for <T2>, this implies as well that the function *run* that the **InferenceEngineServiceImpl** class implements accepts the RDF model of type jena.Model as its first input parameter, and accepts a String value that represents the syntax of the inferencing rules as its second parameter. However, the design of the inferencing component does not impose any constraints on using any specific implementation of the *Inference Engine*, i.e., the component is not bonded to Jena framework.

```
public class SOIEqual extends BaseBuiltin {
      @Override
   public boolean bodyCall(Node[] args, int length, RuleContext context) {
      checkArgs(length, context);
      Node n1 = getArg(0, args, context); //soiResource
      Node n2 = getArg(1, args, context); //?keyword
      String soiResource = n1.getLocalName();
      String keyword = n2.getLiteralLexicalForm();
      return soiResource.equalsIgnoreCase(keyword);
   }
}
```

Figure 4-43: Class implementing the custom functor (SOIEqual)

4.10 SMIL player

We have opted to provide an implementation of the components and frameworks of the DCMC to support SMIL documents. To experiment those implemented components and frameworks of DCMC, we required a SMIL player. Mobile systems, e.g., Android, provide a built-in SMIL player that is embedded in the MMS application to enable viewing the MMS messages. However, this built-in SMIL player is strongly coupled with the MMS application and is optimized to support a subset of SMIL specifications that the MMS specifications adopt. Hence the embedded SMIL player cannot be used to test our approach without being extracted and extended. This means, to be able to use the SMIL player to experiment our approach, we had to: (i) decouple and extract that SMIL player from the MMS application, and (ii) extend the SMIL player to add support to element containers such as *parallel* and *sequence*. The MMS application that the Android platform provides is an open source project and is available for download at [The Android Open Source Project 2017].

In the following, we provide a simplified explanation of the mechanism of presenting a SMIL document by using the adapted SMIL player application, providing a full documentation about that SMIL player is out of the scope of this work:

i. The application provides two main classes to present a SMIL document, these are the *SMILPlayer* and the *Presenter*.
ii. The *SMILPlayer* class is responsible for creating the timeline of the presentation and adding the individual media elements to that timeline, those media elements can be either element containers such as parallel and sequence, or region elements such as text, image, video, and audio.
iii. Once the *SMILPlayer* class is initiated, it creates a list that represents the timeline of the presentation. This list holds all the elements of the presentation and is sorted by the value of the "*offsetTime*". The *offsetTime* attribute defines the point of time when an action that corresponds to a media element can be taken, an action can be either displaying or hiding an element.
iv. The *SMILPlayer* class starts playing the presentation by sequentially fetching the media elements from the timeline and dispatching those elements as timeline events to the *Presenter* class. A timeline event informs the *Presenter* class about the action to be performed, i.e., begin or end of presenting an element.
v. The *Presenter* class displays the element that is linked to timeline *begin* event on the screen based on the values of attributes such as *X,Y, height,* and *width*. The displayed element remains visible on the screen till the presenter receives the *end* event.
vi. The *SMILPlayer* class runs until all elements have been displayed and there are no more elements in the timeline. However, the *SMILPlayer* class supports user's interaction with the presentation by providing presentation's controls such as *pause*, *resume*, *stop*, *forward*, *backward,* and *replay*.

We used this adapted version of the SMIL Player in our experiments to test the implemented components and frameworks of DCMC. In section 4.11 we present our illustrating experiment.

4.11 Experiments and explanation of results

For the purpose of illustrating the authoring and sharing process of multimedia documents in the context of PAN by means of a software application, we provide the DCMC prototype, which implements the framework and components we presented in chapter 3 and in this chapter.

The DCMC prototype is implemented in Java and was developed by using Android Studio. The prototype makes use of external libraries such as Apache Jena framework [Apache Org. 2017b], Apache Lucene [Apache Org. 2017a], and Clarifai [Clarifai 2017]. The Apache Jena framework is designed to run on server environments that support J2EE, which imposed constraints on running the framework on Android platform. Thanks to several projects which provided a porting for Jena framework to Android, we were able to include and use Jena framework in our prototype. The version we used in our prototype is the AndroJena 0.5 [AndroJena 2017], which is provided under the Apache license v.2.0.

The design and architecture of DCMC allow for incorporating different networking technologies, W3C standards, online services, and LOD APIs. In the implemented prototype we opted to use Bluetooth as networking technology, SMIL as multimedia presentation format, and DBpedia and GeoNames as LOD sources.

Our illustrating experiment uses a pair of two devices and four multimedia documents. Two documents on each device.

As presented above, the *Inference Engine* is built on top of Jena framework and is invoked during the composition process by other components of DCMC. In this experiment the *Inference Engine* is invoked in the phases of *Analysis*, *Selection*, and *Composition*. The input are RDF statements and a set of rules and SPARQL statements, the output differs according to phase, e.g., in the *Analysis* phase the output is a set of SOI.

In the following, we present the phases of the experiment in details:

Phase (1): *PAN establishment*

In this phase a smartphone and a tablet form the PAN.

Input: A smartphone (S1) and a tablet (T1).

Output: PAN.

Process: The tablet sets a listener for incoming connections, the smartphone searches for nearby devices and discovers the tablet. It initiates a connection; the tablet accepts the incoming connection. Consequently, the PAN is formed and the two devices can exchange data.

Phase (2): *Document to RDF*

Besides the implemented services and components we provided to support SMIL documents, *FMMDR* framework has several modules and utilities: The local storage scanning service on a device is a File I/O service that searches for files of the types .smil or .sml in the shared documents' folder on a device and captures a list of available SMIL documents. Figure 4-44 depicts the discovered SMIL documents as displayed on the smartphone.

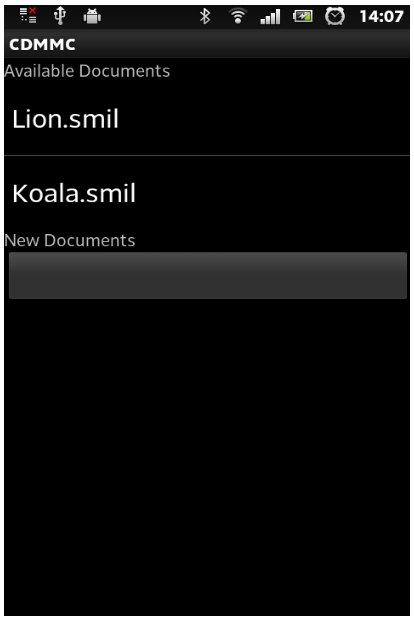


Figure 4-44: List of documents - smartphone

The implementation of the interface *DocumentToRDFService* provides a process to convert a SMIL into an equivalent RDF document. This conversion process consists of three steps: i) the conversion of the XML data that represent the SMIL document to its equivalent Document Object Model, e.g., SMIL-DOM, ii) the conversion of the SMIL-DOM to ZYX-DOM, and iii) the conversion of the ZYX-DOM to RDF.

In our experiment, this phase can be further described in terms of input, output, and process:

Input: A set of SMIL multimedia documents.

Output: *RDF-Graph* that represents all the multimedia documents provided as input and captures all the required information for the authoring process.

Process: Each SMIL document is de-serialized into a DOM representation. The root subject of the R*DF-Graph* is the document itself. All attributes of the root document and its sub-elements are represented as RDF triples. The right side of Figure 4-45 depicts the layout of a multimedia document as displayed on the screen, the left side depicts the equivalent representation by means of *RDF-Graph* of that layout.

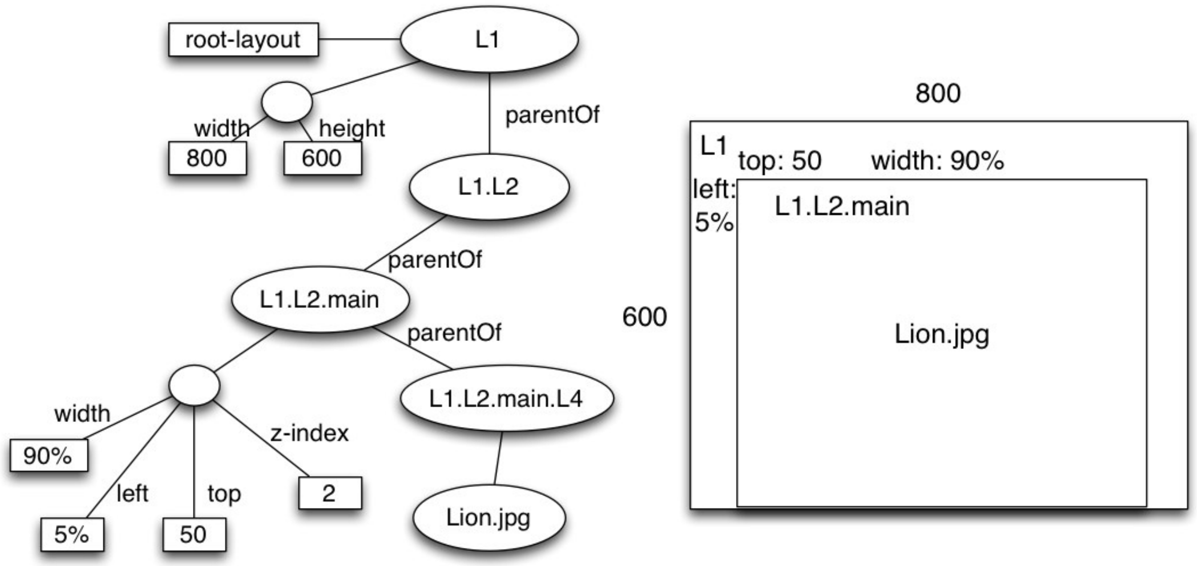


Figure 4-45: RDF/Document layout mapping

Phase (3): *Analysis*

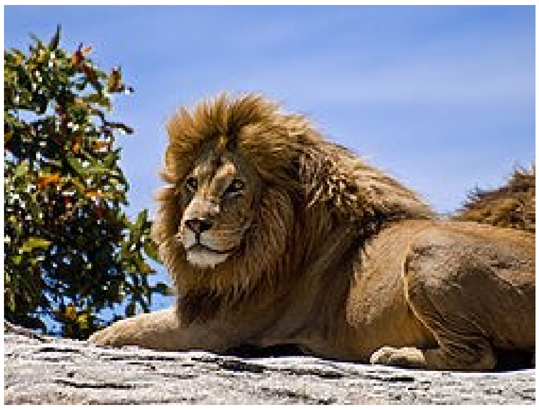
The *Analyzer* is responsible for detecting a set of SOI common to all or some of the shared documents. Again, the analysis process can be described in terms of input, output, and process as follows:

Input: RDF documents, inference rules, and SPARQL statements.

Output: A set of SOI.

Process: We defined a set of rules to identify one or more SOI as presented in section 4.5.3. The *Analyzer* analyzes two parts of the RDF document: the keyword part, which is the result of performing text analysis on text content and low level feature analysis on image content, and the metadata part, which is extracted from the metadata tags of the multimedia document and its media elements. The individual media elements (e.g., Text, Image) found in the SMIL documents are analyzed and added to the *RDF-Graph*. Analyzing the text results in identifying keywords according to the subjects covered by the SMIL document. For example, in our illustrating experiment we run the analysis services on a SMIL document named "Lion.smil". This SMIL document encompasses two media elements, an image and a text that provides information about the object presented by the image. Figure 4-47 presents the image resource which is named "Lion_on_Rock.JPG" and the results of the different analysis steps we run on that image resource. The EXIF data and other metadata of the images are also incorporated in the analysis. Figure 4-46 presents the content of the text resource which is named "Lion.txt". Figure 4-48 presents the result of the text analysis on this text resource (to keep it simple for the purposes of illustrating the text analysis service, we abstract from the variations of the terms like lion (sg.)/lions(pl.)).

```
<rdf:Description
rdf:about="http://.../dcmc/Lion.smil/
Lion_on_Rock.jpg/keywords">
    <dcmc:keyword>safari</dcmc:keyword>
    <dcmc:keyword>barbaric</dcmc:keyword>
    <dcmc:keyword>lion</dcmc:keyword>
    <dcmc:keyword>portrait</dcmc:keyword>
    <dcmc:keyword>wild</dcmc:keyword>
    <dcmc:keyword>majestic</dcmc:keyword>
    <dcmc:keyword>danger</dcmc:keyword>
    <dcmc:keyword>large</dcmc:keyword>
    <dcmc:keyword>fur</dcmc:keyword>
    <dcmc:keyword>carnivore</dcmc:keyword>
    <dcmc:keyword>wildlife</dcmc:keyword>
    <dcmc:keyword>mane</dcmc:keyword>
    <dcmc:keyword>travel</dcmc:keyword>
    <dcmc:keyword>no person</dcmc:keyword>
    <dcmc:keyword>predator</dcmc:keyword>
    <dcmc:keyword>nature</dcmc:keyword>
    <dcmc:keyword>cat</dcmc:keyword>
    <dcmc:keyword>animal</dcmc:keyword>
    <dcmc:keyword>outdoors</dcmc:keyword>
    <dcmc:keyword>mammal</dcmc:keyword>
</rdf:Description>
```



[Wikimedia 2017a]

Figure 4-47: The image resource "Lion_on_Rock.JPG" and the result of the analysis

The **lion** (Panthera leo) is one of the four big cats in the genus Panthera, and a member of the family Felidae. With some **males** exceeding 250 kg in weight, it is the second-largest living cat after the tiger. **Wild lions** currently exist in Sub-Saharan **Africa** and in Asia with a critically endangered remnant population in Gir Forest **National** Park in India, having disappeared from North **Africa**, and Southwest Asia in historic times. Until the late Pleistocene, which was about 10,000 **years** ago, the **lion** was the most widespread large land mammal after **humans**. They were found in most of **Africa**, much of Eurasia from western Europe to India, and in the Americas from the Yukon to Peru. **Lions live** for around 10–14 **years** in the **wild**, while in captivity they can **live** over 20 **years**. In the **wild**, **males** seldom **live** longer than ten **years**, as injuries sustained from continuous fighting with rival **males** greatly reduce their longevity. They typically inhabit savanna and grassland, although they may take to bush and forest. **Lions** are unusually social compared to other cats. A pride of **lions** consists of related females and offspring and a small number of adult **males**. Groups of female **lions** typically hunt together, preying mostly on large ungulates. **Lions** are apex and keystone predators, although they will scavenge if the opportunity arises. While **lions** do not typically hunt **humans** selectively, some have been known to become man-eaters and seek human prey. The **lion** is a vulnerable species, having seen a possibly irreversible population decline of 30 to 50 percent over the past two decades in its African range. **Lion** populations are untenable outside of designated reserves and **national** parks. Although the cause of the decline is not fully understood, habitat loss and conflicts with **humans** are currently the greatest causes of concern. **Lions** have been kept in menageries since Roman times and have been a key species sought for exhibition in zoos the world over since the late eighteenth century. Zoos are cooperating worldwide in breeding programs for the endangered Asiatic subspecies. Visually, the **male lion** is highly distinctive and is easily recognized by its mane. The **lion**, particularly the face of the **male**, is one of the most widely recognized animal symbols in human culture. Depictions have existed from the Upper Paleolithic period, with carvings and paintings from the Lascaux and Chauvet Caves, through virtually all ancient and medieval cultures where they historically occurred. It has been extensively depicted in literature, in sculptures, in paintings, on **national** flags, and in contemporary films and literature.

Figure 4-46: Content of a text resource named "Lion.txt"

```
<rdf:Description rdf:about="…/dcmc/Lion.smil/Lion.txt/keywords">
    <dcmc:keyword>humans</dcmc:keyword>
    <dcmc:keyword>live</dcmc:keyword>
    <dcmc:keyword>lion</dcmc:keyword>
    <dcmc:keyword>national</dcmc:keyword>
    <dcmc:keyword>years</dcmc:keyword>
    <dcmc:keyword>wild</dcmc:keyword>
    <dcmc:keyword>males</dcmc:keyword>
    <dcmc:keyword>africa</dcmc:keyword>
    <dcmc:keyword>lions</dcmc:keyword>
</rdf:Description>
```

Figure 4-48: The result of the analysis on the text resource "Lion.txt"

The results of the analysis are added to the RDF dataset as keyword attributes and linked to the corresponding resource element. The interface *SOIDetectionService* uses this list of keywords to detect one or more SOI. Figure 4-49 represents the detected SOI in our experiment. The SOI rated highest is explicitly identified by the relation "maxRatedSOI".

```
<rdf:Description rdf:about="http://univie.ac.at/dcmc/sois">
    <dcmc:maxRatedSOI rdf:resource="http://univie.ac.at/dcmc/soi/mammal"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/tree"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/panda"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/koala"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/nature"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/wood"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/tiger"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/lion"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/outdoors"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/animal"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/mammal"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/wildlife"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/zoo"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/portrait"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/carnivore"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/large"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/safari"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/danger"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/cat"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/fur"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/one"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/predator"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/wild"/>
    <dcmc:soi rdf:resource="http://univie.ac.at/dcmc/soi/person"/>
</rdf:Description>
```

Figure 4-49: The detected SOI in this experiment

The analysis process creates for each detected SOI an rdf:resource and adds it to the graph. This rdf:resource links additional information that can be used in subsequent phases of the authoring process. Figure 4-50 presents two rdf:resources that have been created for two detected SOI represented by the terms "wild" and "mammal". These rdf:resources link the related media elements to the detected terms in addition to the rating of each of the terms. However, the rdf:resource is the result of the *Analysis* phase and will be further processed in the subsequent phases of *Enrichment* and *Selection*. The

Enrichment phase will add the rdf:resource "enrichment" and the *Selection* phase will select a subset of the linked media elements as we will present in the following phases.

```
<!-- rdf:resource "soi" for the term "wild"  -->
<rdf:Description rdf:about="…/dcmc/soi/wild">
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Tiger.smil/Tiger_in_Ranthambhore.JPG"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Panda.smil/Grosser_Panda.JPG"/>
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Koala.smil/Friendly_Female_Koala.JPG"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Lion.smil/Lion_on_Rock.jpg"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Lion.smil/Lion.txt"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Panda.smil/Panda.txt"/>
    <dcmc:rating
     rdf:datatype="http://www.w3.org/2001/XMLSchema#int">6</dcmc:rating>
</rdf:Description>

<!-- rdf:resource "soi" for the term "mammal" -->
<rdf:Description rdf:about="…/dcmc/soi/mammal">
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Tiger.smil/Tiger_in_Ranthambhore.JPG"/>
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Panda.smil/Grosser_Panda.JPG"/>
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Koala.smil/Friendly_Female_Koala.JPG"/>
    <dcmc:mediaElement

     rdf:resource="…/dcmc/Lion.smil/Lion_on_Rock.jpg"/>
    <dcmc:rating

   rdf:datatype="http://www.w3.org/2001/XMLSchema#int">8</dcmc:rating>
</rdf:Description>

    <!-- other detected terms -->
  ...
```

Figure 4-50: RDF-Graph for the resources representing the SOI "wild" and "mammal"

Phase (4): *Enrichment*

The LOD component realizes the retrieval of data from LOD sources. The process can be described as follows:

Input: A set of SOI.

Output: Media elements retrieved from LOD.

Process: The enrichment service receives a set of SOI and tries to collect for each SOI additional information from the Web of Data using the provided API implementations. For this specific experiment, we used the **DBpediaEnrichmentServiceImpl** class presented in section 4.7.3 to retrieve additional information from DBpedia about the detected SOI represented by the term "mammal". For illustrating purposes, Figure 4-51 shows the rdf:resource "soi" represented by the term “mammal”, the added rdf:resource “enrichment”, and some part of the retrieved information about the SOI “mammal” linked to the rdf:resource “enrichment”.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dbponto="http://dbpedia.org/ontology/"
    xmlns:dcmc="http://dcmc.univie.ac.at/mmdoc-rdf-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/" >
    <rdf:Description rdf:about="http://.../dcmc/soi/mammal">
    <dcmc:mediaElement
        rdf:resource="…/dcmc/Tiger.smil/Tiger_in_Ranthambhore.JPG"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Panda.smil/Grosser_Panda.JPG"/>
    <dcmc:mediaElement
        rdf:resource="…/dcmc/Koala.smil/Friendly_Female_Koala.JPG"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Lion.smil/Lion_on_Rock.jpg"/>
    <dcmc:rating
        rdf:datatype="http://www.w3.org/2001/XMLSchema#int">8</dcmc:rating>
    <dcmc:enrichment rdf:resource="http://.../dcmc/soi/mammal/enrichment"/>
    <rdf:Description rdf:about="http://.../dcmc/soi/mammal/enrichment">
        <dbpprop:classis>Mammalia</dbpprop:classis>
        <foaf:name>Mammals</foaf:name>
        <rdfs:label>Mammal</rdfs:label>
        <dbponto:abstract>Mammals (formally Mammalia) are a class of vertebrate,
            air-breathing animals whose females are characterized by the
            possession of mammary glands while both males and females are
            characterized by sweat glands, hair and/or fur, three middle ear
            bones used in hearing, and a neocortex region in the brain. …
        </dbponto:abstract>
        …
    </rdf:Description>
</rdf:RDF>
```

Figure 4-51: Part of the *Enrichment* results about the SOI “mammal”

Phase (5): *Selection*

The *Selector* is responsible for selecting the media elements required for the composition of the new document based on the semantic meaning of the SOI.

The description of the experiment in terms of input, output, and process is as follows:

Input: a set of SOI and inference rules.

Output: a set of media elements.

Process: As presented in section 4.6.3, we provided three implementations for the interface *SOISelectionService***.** The results presented in this experiment are those provided by the **SOISelectionServiceImpl** class, which selects the SOI rated highest. The first step in the selection process is to find the media elements that are related to the set of SOI. The second step is to select the most relevant media content to each SOI, which will be included in the composed document. In order to determine which media elements are more relevant with respect to the set of SOI, we calculate the distances between the feature vectors and the SOI. Media elements found to have a distance within a specific threshold will be selected. In this experiment, the selection considers only the media elements of type Image and Text. Quite an obvious rule to provide new information in addition to the original documents is as follows: Original text elements will be selected only if no new text elements were retrieved from LOD resources during the *Enrichment* phase. Figure 4-52 presents the result of the *Selection* phase, which includes two images that stem from the provided multimedia documents and one text that is retrieved from the DBpedia about the resource "mammal".

```
<rdf:Description rdf:about="http://univie.ac.at/dcmc/soi/mammal">
    <dcmc:mediaElement rdf:resource="…/dcmc/Panda.smil/Grosser_Panda.JPG"/>
    <dcmc:mediaElement rdf:resource="…/dcmc/Lion.smil/Lion_on_Rock.jpg"/>
    <!-- The rdf:resource /enrichment is the same rdf:resource /enrichment -->
    <!-- presented in Figure 4-51 -->
    <dcmc:enrichment rdf:resource="http://.../dcmc/soi/mammal/enrichment"/>
    <dcmc:rating
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">8</dcmc:rating>
</rdf:Description>
```

Figure 4-52: The results of selection process

Phase (6): *Composition*

The Composer realizes the composition of the final document and can be described as follows:

Input: The set of selected media elements, the RDF representations of the original documents, composition rules, and optionally a set of multimedia document composition templates.

Output: RDF document that represents the final composed multimedia document.

Process: As presented in section 4.8.3, the *Composer* implements and makes use of three services to create a new multimedia document. The interface *DocumentCompositionService* created the new document in RDF format and linked the required parts, which are represented by the resources (*mediaElements*, *meta*, *layout*, *structure* and *keywords*). The interface *StructureCompositionService* created the structure of the multimedia document. Since the resource *mediaElements* contains media elements of the types *image* and *text*. The structure rules concluded to use the combination *Strucute1* and *Temporal1*. See Table 4-1. The *Temporal1* rule uses the structure that the *composeSubTemporalPar* presents, i.e., the media elements are included in a parallel container and presented simultaneously. The interface *LayoutCompositionService* created the layout of the document. Figure 4-53 represents the multimedia document about the SOI "mammal" encoded as SMIL format.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
    <head>
        <meta name="name" content="Mammal" />
        <meta name="classis" content="Mammal" />
        <layout type="text/smil-basic-layout">
            <root-layout height="600" id="root-layout" width="800" />
            <region id="slide1">
                <region top="0" id="region-text1" left="0" height="100"
                 width="100%" />
                <region top="100" id="region-image1" left="0" height="100"
                 width="100%" />
                <region top="200" id="region-image2" left="0" height="100"
                 width="100%" />
            </region>
        </layout>
    </head>
    <body id="default-body">
        <par id="par1" max="indefinite">
            <text id="text1" region="region-text1" src="Mammal.txt" />
            <img id="image1" region="region-image1" src="Grosser_Panda.JPG" />
            <img id="image2" region="region-image2" src="Lion_on_Rock.jpg" />
        </par>
    </body>
</smil>
```

Figure 4-53: The composed multimedia document represented in SMIL format

Phase (7): *Transformation*

The implementation of the interface *RDFToDocumentService* provides the reverse process to Document-To-RDF, where the RDF document can be converted back to a SMIL document. The component can be described again in terms of input, output, and process as follows:

Input: RDF document.

Output: SMIL document.

Process: The RDF triples are converted to SMIL elements. Table 4-3 shows an illustrating snippet of a SMIL document showing the <root-layout> element, and its RDF and SMIL representations.

RDF	*SMIL*
`<rdf:Description rdf:about="…/Lion.smil/L1"> <dcmc:id>L1</dcmc:id> <dcmc:width>800</dcmc:width> <dcmc:height>600</dcmc:height> <dcmc:type>root-layout</dcmc:type> </rdf:Description>`	`<root-layout id="L1" height="600" width="800"/>`

Table 4-3: RDF2SMIL - Transformation of RDF to SMIL

Table **4-4** summarizes the input and output of each phase of the authoring process for a demonstration of the prototype.

Phase	***Input***	***Output***
PAN Establishment	Smartphone (S1) Tablet (T1)	PAN
Document to RDF	Lion.smil Koala.smil Panda.smil Tiger.smil	S1-RDF T1-RDF
Analysis	S1-RDF T1-RDF SPARQL Rules	Set of SOI = {Mammal, Wild, …}
Enrichment	Set of SOI = {Mammal}	DBpedia Article about Mammals
Selection	Rules Text: Mammals, Lion, Koala, Panda, Tiger Images: Lion, Koala, Panda, Tiger	Text: Mammals Images: Lion, Panda
Composition	Text: Mammals Images: Lion, Panda, Rules S1-RDF T1-RDF	New Multimedia content. Subgraph: Mammals.rdf
Transformation	Subgraph: Mammals.RDF	Document Mammals.smil

Table 4-4: The phases of the authoring process and their input and output

Sample Screenshots capturing some phases described before

Table 4-5 provides screenshots taken from the devices, which we used during this illustrating experiment. The first and the third rows show two lists of multimedia documents of type ".smil" on the two devices before and after the authoring process. The second row shows an opened multimedia document on the smart phone and on the tablet. The last row shows the new authored multimedia document. The presented multimedia documents in this table illustrate the layout of the composed document depicting the spatial relationships between the media elements, i.e., a text element followed by one or more images, thus, the text element shown serves only the purpose of the illustration.

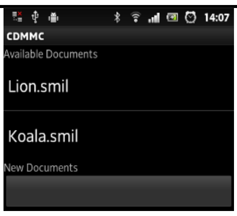
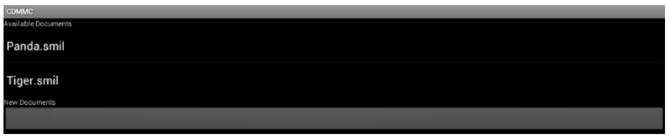
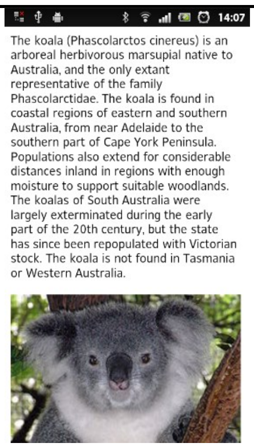
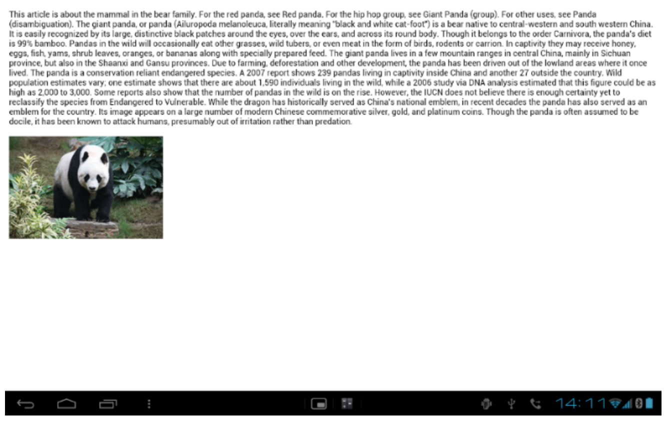
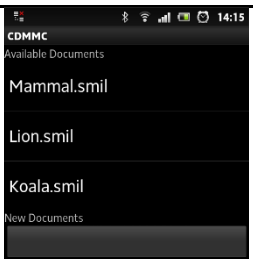
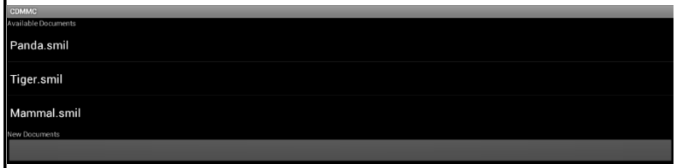
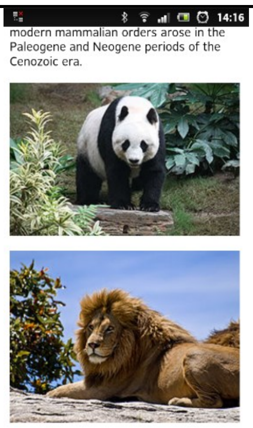
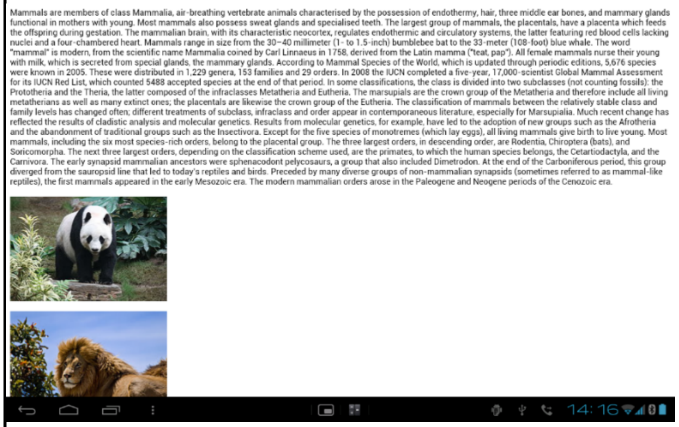
Description	*Smartphone (S1)*	*Tablet (T1)*
List of documents (initial state)	 CDMMC Available Documents Lion.smil Koala.smil New Documents	 CDMMC Available Documents Panda.smil Tiger.smil New Documents
Opened multimedia document	 The koala (Phascolarctos cinereus) is an arboreal herbivorous marsupial native to Australia, and the only extant representative of the family Phascolarctidae. The koala is found in coastal regions of eastern and southern Australia, from near Adelaide to the southern part of Cape York Peninsula. Populations also extend for considerable distances inland in regions with enough moisture to support suitable woodlands. The koalas of South Australia were largely exterminated during the early part of the 20th century, but the state has since been repopulated with Victorian stock. The koala is not found in Tasmania or Western Australia.	 This article is about the mammal in the bear family. For the red panda, see Red panda. For the hip hop group, see Giant Panda (group). For other uses, see Panda (disambiguation). The giant panda, or panda (Ailuropoda melanoleuca, literally meaning "black and white cat-foot") is a bear native to central-western and south western China. It is easily recognized by its large, distinctive black patches around the eyes, over the ears, and across its round body. Though it belongs to the order Carnivora, the panda's diet is 99% bamboo. Pandas in the wild will occasionally eat other grasses, wild tubers, or even meat in the form of birds, rodents or carrion. In captivity they may receive honey, eggs, fish, yams, shrub leaves, oranges, or bananas along with specially prepared feed. The giant panda lives in a few mountain ranges in central China, mainly in Sichuan province, but also in the Shaanxi and Gansu provinces. Due to farming, deforestation and other development, the panda has been driven out of the lowland areas where it once lived. The panda is a conservation reliant endangered species. A 2007 report shows 239 pandas living in captivity inside China and another 27 outside the country. Wild population estimates vary; one estimate shows that there are about 1,590 individuals living in the wild, while a 2006 study via DNA analysis estimated that this figure could be as high as 2,000 to 3,000. Some reports also show that the number of pandas in the wild is on the rise. However, the IUCN does not believe there is enough certainty yet to reclassify the species from Endangered to Vulnerable. While the dragon has historically served as China's national emblem, in recent decades the panda has also served as an emblem for the country. Its image appears on a large number of modern Chinese commemorative silver, gold, and platinum coins. Though the panda is often assumed to be docile, it has been known to attack humans, presumably out of irritation rather than predation.
Extended list of documents after composition process	 CDMMC Available Documents Mammal.smil Lion.smil Koala.smil New Documents	 CDMMC Available Documents Panda.smil Tiger.smil Mammal.smil New Documents
The new composed document corresponding to SOI "Mammal"	 modern mammalian orders arose in the Paleogene and Neogene periods of the Cenozoic era.	 Mammals are members of class Mammalia, air-breathing vertebrate animals characterised by the possession of endothermy, hair, three middle ear bones, and mammary glands functional in mothers with young. Most mammals also possess sweat glands and specialised teeth. The largest group of mammals, the placentals, have a placenta which feeds the offspring during gestation. The mammalian brain, with its characteristic neocortex, regulates endothermic and circulatory systems, the latter featuring red blood cells lacking nuclei and a four-chambered heart. Mammals range in size from the 30–40 millimeter (1- to 1.5-inch) bumblebee bat to the 33-meter (108-foot) blue whale. The word "mammal" is modern, from the scientific name Mammalia coined by Carl Linnaeus in 1758, derived from the Latin mamma ("teat, pap"). All female mammals nurse their young with milk, which is secreted from special glands, the mammary glands. According to Mammal Species of the World, which is updated through periodic editions, 5,676 species were known in 2005. These were distributed in 1,229 genera, 153 families and 29 orders. In 2008 the IUCN completed a five-year, 17,000-scientist Global Mammal Assessment for its IUCN Red List, which counted 5488 accepted species at the end of that period. In some classifications, the class is divided into two subclasses (not counting fossils): the Prototheria and the Theria, the latter composed of the infraclasses Metatheria and Eutheria. The marsupials are the crown group of the Metatheria and therefore include all living metatherians as well as many extinct ones; the placentals are likewise the crown group of the Eutheria. The classification of mammals between the relatively stable class and family levels has changed often; different treatments of subclass, infraclass and order appear in contemporaneous literature, especially for Marsupialia. Much recent change has reflected the results of cladistic analysis and molecular genetics. Results from molecular genetics, for example, have led to the adoption of new groups such as the Afrotheria and the abandonment of traditional groups such as the Insectivora. Except for the five species of monotremes (which lay eggs), all living mammals give birth to live young. Most mammals, including the six most species-rich orders, belong to the placental group. The three largest orders, in descending order, are Rodentia, Chiroptera (bats), and Soricomorpha. The next three largest orders, depending on the classification scheme used, are the primates, to which the human species belongs, the Cetartiodactyla, and the Carnivora. The early synapsid mammalian ancestors were sphenacodont pelycosaurs, a group that also included Dimetrodon. At the end of the Carboniferous period, this group diverged from the sauropsid line that led to today's reptiles and birds. Preceded by many diverse groups of non-mammalian synapsids (sometimes referred to as mammal-like reptiles), the first mammals appeared in the early Mesozoic era. The modern mammalian orders arose in the Paleogene and Neogene periods of the Cenozoic era.

Table 4-5: Sceenshots taken from the interconnected devices during an experiment

4.12 Conclusion

In this chapter, we presented the underlying concepts, the specifications, and the implementation of the different components and frameworks of the DCMC. To cover the different aspects of the functional and non-functional requirements of DCMC, we followed the concepts of generic software design and architecture. The design of generic components and frameworks is very challenging and can be optimized only after several iterations, experiments, and several system level and component level tests.

We defined software design patterns that we used to specify the different components and services provided by DCMC. By applying those design patterns on the different components and services, we achieved a consistent behavior to those components and services, and fulfilled the requirement of the reuse of design and components.

The DCMC provides a clear partitioning of its services into two logically separated parts: the one that covers the networking required by authoring process to exchange the data, and the one that covers all the services needed for the multimedia documents authoring and sharing.

The Networking Framework (*NtF*) provides the transport layer for the authoring process to exchange data. It provides different services and components to support PAN management, and allows DCMC to incorporate different devices using different PAN technologies in the authoring and sharing process.

The enhanced authoring and sharing process of multimedia documents consists of several phases. Each phase is handled by a dedicated system component and service. These components and services make use of Semantic Web technologies to achieve a flexible yet efficient authoring process. The *RDF-Graph*, which DCMC uses as internal and canonical data model, provided a flexible, extendable, and transparent data model for the different services. The rule-based authoring that the different services used provides an efficient authoring process. Besides its flexibility, it enabled creating multiple multimedia documents at the same time.

As presented in section 4.8, the *StructureCompositionService* provides rules to guide creating the structure of the multimedia document. Applying those rules on the complete RDF dataset of DCMC resulted in creating several multimedia documents for each detected SOI. The rule-based approach enables us to use alternative sets of rules, which are used by different services within the life cycle of the authoring process. Alternating the rules enables the manipulation of the behavior of the system and the composition logic without being forced to change the implemented classes or to provide new classes. Such flexibility not only provides an efficient and convenient experimenting environment for testing and configuring the whole system, but also provides a well-defined mechanism for adapting the system to specific needs of other applications.

5. Conclusion

Having presented the conceptual design and the implementation of the Dynamic Cross-Model Composition (DCMC) approach, we discuss in this chapter the results achieved in the course of this thesis, our observations represented by the lessons learned, the limitations of our approach, and some ideas on future work.

5.1 Results and Contributions

Multimedia content authoring is a challenging topic and has been researched in the last two decades. Research efforts yielded different authoring paradigms, languages, and tools. However, an approach like our approach has not been proposed yet. The research results of this work contribute to various fields: multimedia content analysis, authoring, sharing, requirements engineering and application architecture and design. The DCMC developed and presented in the course of this thesis is built upon the contributions to those research areas. The DCMC provides an enhanced multimedia content authoring process supported by a flexible and generic framework for rule-based content composition driven by the Subject of Interest of the user (SOI).

In chapter 2 we discussed various concepts and technologies related to this work. We presented the networking technologies intended to enable forming PAN and interconnecting various types of devices. We presented the mobile platforms and frameworks and provided information on how these mobile platforms support multimedia application design and development. Then we discussed W3C standards concerning Semantic Web technologies and multimedia document models and formats. Finally we provided an in-depth analysis for the related work following a systematic approach based on the defined comparison criteria. These criteria enabled two levels of analysis: At the first level of analysis we analyzed the related work considering the authoring process as a whole, at the second level of analysis we looked at each phase of the authoring process. This approach in analyzing the related work resulted in (i) providing a cross-application criterion-based analysis, and (ii) defining patterns used by approaches and applications.

Chapter 3 represents the results of the conceptual work of this thesis. As mentioned, the DCMC approach aims at the automatic analysis, authoring, and sharing of augmented multimedia presentations driven by a user's detected Subject of Interest (SOI). Besides this main goal, the DCMC approach considers the change in role of the users from being only passive content consumers to content producers and consumers at the same time. Thus, we followed the notion of User Stories to identify the requirements of the system. User Stories allowed us to focus on the users, consequently, the identified requirements represented by these User Stories target audiences with non-technical background.

Having identified the requirements, the DCMC approach reflected those requirements by:

(i) Reconsidering the flow of the authoring process by providing an analysis phase to detect user's SOI. These detected SOI are used to select and enrich related multimedia elements with discovered information from the Web of Data, which can be included in the authoring process. As a result, we defined an enhanced authoring process that consists of eight phases. This enhanced authoring process focuses on the user and targets content authoring in PAN.

(ii) Presenting a rule-based multimedia composition system that incorporates Semantic Web technologies intended to extract knowledge from a RDF-Graph. This rule-based composition allowed for alternating and configuring the behavior of the authoring process and its results, respectively, by adapting or providing new sets of composition rules.

The User Stories and the enhanced authoring process provided input to the design of the services following the principals of a service-oriented architecture (SOA). The SOA realized by our approach follows well-defined Software Engineering principles. This resulted in having a modular and generic architecture that consists of seven logically separated components. These seven components provide various services that support the enhanced authoring process.

The envisioned Distributed Presentation Player (DisPly) presented in section 3.5 is motivated by the advances of smart devices and networking technologies. We will discuss this topic in more details in the section on future work below.

The results of chapter 4 can be summarized by having a fully implemented functional system, which represents the proof of concept of DCMC approach. We followed a systematic approach to develop each of the seven components of our SOA, for each component this systematic approach comprises the underlying concepts, the specifications of the components, and the implementation of the various parts of the components. Section 4.11 provided a walk through a short illustrating application experiment of the system. For each of the phases of the enhanced authoring process we explained the input, the internal process, and the output. We provided snippets of system log entries and screenshots taken from the interconnected devices we used during the experiments.

5.2 Lessons learned

The DCMC approach is motivated by four aspects, the capabilities of smart devices, the Semantic Web technologies and applications, the social network platforms, and the networking technologies. In the course of trying to gain benefit from using these technologies from the four areas in an integrated manner, we tackled research topics and concepts addressing multimedia content authoring and sharing, and we presented the related technologies and standards. Consequently, each of these areas has its impact on our work.

5.2.1 A service-oriented architecture can support various authoring processes

A service-oriented architecture provides decoupled standalone services. A service is designed to accomplish a specific task. It defines an Application Programming Interface (API) that allows for interacting with the service by means of a request/response model that makes use of well-defined input and output data structures. This allows for separating between the service layer represented by the standalone services and the service-orchestration layer represented by an application that is built on top of and consumes these services. By applying these concepts, the DCMC approach separates between the services that the various components of the system provide and the application that makes use of the provided services. This application implements the enhanced authoring process that the DCMC approach defines. The enhanced authoring process defines the flow of authoring tasks that can be achieved by making use of the provided services. This implies that we can reuse and extend these services to implement new applications. These new applications can implement different authoring processes that are tailored to fulfill various content authoring requirements.

5.2.2 Benefits of using RDF

The enhanced authoring and sharing process of multimedia documents consists of several phases, each phase is handled by a dedicated service. These services make use of Semantic Web technologies to support a flexible and efficient authoring process. The *RDF-Graph*, which DCMC approach uses as internal and canonical data model, provides a flexible, extendable, and transparent data model for the different services. During the lifecycle of the authoring process these services return the results of calling their APIs encoded by means of RDF. These RDF encoded results are added to the main *RDF-Graph* contributing to a global dataset that holds all the information about the input multimedia content and the content to be composed. The DCMC approach benefited from using RDF in many ways: (i) It allowed for using inferencing techniques that can be applied on *RDF-Graphs* to extract knowledge enabling a rule-based multimedia content authoring system. (ii) It provided a flexible and transparent data model that allowed the various services to add data to the *RDF-Graph* during the lifecycle of the authoring process. And (iii), since the inferencing rules are applied on the complete *RDF-Graph*, it is possible to author more than one multimedia document about various SOI simultaneously.

5.2.3 Rule-based composition allows for wide range of options

The DCMC approach introduced a rule-based authoring system realized by various services. The rules that these various services make use of can be stored in and provided to the system by separate data resources. This implies that changing the set of rules that these data resources provide allows for alternating and configuring the behavior of the authoring process and its results by adapting or providing new set of composition rules. Despite a limitation regarding this technique (see section 5.3.3), which has been imposed by the implementation of the used *Inferencing Engine*, the DCMC approach benefited from this technique by achieving a separation between the inferencing-logic and the implementation of the various services. The inferencing - logic is realized by the rules and represents the logic that is used to complete the authoring process using the available services. The various services make use of the provided rules to complete their tasks.

5.3 Limitations of our approach

The DCMC approach aims at providing a flexible and extendable rule-based authoring process, supported by a well-designed service-oriented architecture. This authoring process incorporates Semantic Web technologies and targets multimedia presentation composition in a distributed authoring environment. This distributed authoring environment encompasses different types of interconnected devices that use various networking technologies to interconnect with other devices. In the course of implementing the DCMC prototype, we had to make some assumptions regarding the following issues: (i) The availability of frameworks that implement the Semantic Web technologies targeting mobile computing platforms. (ii) The simplicity of establishing a Bluetooth connection between two devices. And (iii) The ability to provide custom functions to the *Inference Engine* on the fly. Hence, these assumptions helped us to address the limitations represented by these issues. In the following, we discuss these assumptions and the related limitations.

5.3.1 Availability of Semantic Web frameworks targeting mobile platforms

The concepts of the Web of Data and the Semantic Web defined by the W3C target server side computing, i.e., Web servers. This has influenced the development of the frameworks that implement these W3C standards and the technologies used in the various implementations. Consequently, these frameworks were designed and implemented to run and to be hosted on Web servers. The lack of a framework that implements Semantic Web standards targeting mobile computing environments has urged us to use one of the implementations that has been ported to run on Android platform (see section 4.11), which originally was developed to run on Web servers. Using this ported version of the framework enabled us to implement and test the DCMC approach, however we faced difficulties in compiling un-necessary framework components which resulted in an overweighed application. In addition to that, it has limited the support of the DCMC to Android platform only. DCMC support to other mobile platforms, such as Windows phone and iOS, depends on the availability of frameworks that target these platforms and implement the Semantic Web technologies that serve knowledge extraction from interlinked data, e.g., RDF, SPARQL, OWL, or SKOS.

5.3.2 The simplicity of establishing a Bluetooth connection between two devices

In the course of implementing the DCMC prototype presented in this thesis, we have opted to use the Bluetooth technology to interconnect our test devices as presented in section 4.11. The Bluetooth standards define two types of connections that can be established between the Bluetooth-enabled devices. These types are secured and unsecured connections. When developing the Bluetooth libraries that the mobile platforms provide, i.e., Android, Windows, or iOS, platform providers defined the security policies regarding their Bluetooth implementation and connection. Those platforms require a secure Bluetooth connection. This implies that to be able to establish a Bluetooth connection between any two devices, a preceding pairing process must take place in advance. This pairing process imposed a burden on the flow of using the system and on the testing process. In addition to that it has limited the possibility of a spontaneous testing of the approach outside the labor. Such required preparation steps prior to establishing a connection between the PAN-enabled devices can affect the usability of the system and the user experience.

5.3.3 The ability to provide custom functions to the *Inference Engine* on the fly

In sections 4.5.3 and 4.8.3 we presented the inferencing rules we used in the *Analyzer* and the *Composer*. These rules include custom functions that can be used either in the body or in the head of the rule. However, the implementation of these rules is provided programmatically in the code of the *Inference Engine* component. This implies on the one hand that the *Inference Engine* component must be re-compiled every time the implementation of these functions changes or when a new function is required. On the other hand, providing an equivalent implementation to the code implementing these rules can result in a very complex rule syntax that is hard to read, debug, and maintain. Providing the rules as external resources is a key advantage of our approach, since it allows for alternating and configuring the behavior of the authoring process and its results, respectively, by adapting or providing a new set of composition rules. To overcome such limitations, a Software Engineering approach is required to enable the *Inference Engine* to parse and interpret the implementation of the custom rules when provided by an external data resources.

5.4 Future work

In the course of developing the concepts and the prototypical implementation of the DCMC we realized the potential of some interesting further extensions that might be subject of future work. We briefly sketch three of these ideas, distributed playback of multimedia presentations over multiple PAN-enabled devices, composing 3D presentations, and deploying the framework on computing machines.

5.4.1 Distributed playback of multimedia presentation over multiple PAN-enabled devices

Nowadays, users possess various types of consumer electronic devices, such as loudspeakers, smart-watches, projectors, printers, etc. These consumer electronic devices support various types of networking technologies, such as Wi-Fi, Bluetooth, NFC, Body Area Network (BAN) [Karulf 2008], etc. This enables these devices to easily establish peer-to-peer connections. However, the connection between these various types of devices can go beyond the peer-to-peer connection to a sort of a personal area network that interconnects multiple devices and facilitates the communication between them. Having this type of PAN interconnecting various types of electronic devices opens the door to a new generation of multimedia presentation players. This new type of players aims at making use of the various types of the interconnected devices to playback a multimedia presentation. For example, in a PAN that interconnects a smartphone, a loudspeaker, a LED light projector, and a smart-TV, a multimedia presentation can be played back using these available output channels in this PAN. One can imagine that in this setup, the presentation could be played back as follows: the smart-TV can be used to playback the visual elements such as images. The Bluetooth-enabled loudspeakers can be used to playback the acoustic elements such as the audio, and via the smartphone one can control the LED light projector to add suitable lighting effects to the room. This type of presentation players has special requirements regarding the synchronization of the playback of the different media elements over the network, and regarding the communication between the interconnected devices using different networking technologies.

5.4.2 Presentation composition in 3D

The enhancement in the rendering capabilities of the graphical processing units (GPU) that come with the new generation of the smart devices, and existence of new 3D viewers such as the Cardboard from Google [Google 2016] and more sophisticated devices such as the Hololense from Microsoft [Microsoft 2017a], increase the demand of the users on having more attracting multimedia presentations. A 3D effect can be achieved by applying one of the available 3D techniques such as the Anaglyph or the Polarization systems. However, besides the required graphics rendering capabilities of the display device, additional hardware is usually required to be able to display a 3D presentation, such as the Virtual Reality Headsets or the archetypal 3D glasses. The functionality of the DCMC can be extended to add an explicit support for 3D presentations.

5.4.3 Deploy the DCMC framework on special computing machines

The modularity and generality of the service-oriented architecture of DCMC enable the deployment of the system on computing machines such as robots, robots can provide a suitable runtime environment for our approach. Since robots are equipped with different input and output devices, they can interact with users during the lifecycle of the authoring process to collect additional user's preferences that can be used as input and accordingly can influence the quality of the authored document. This requires extending and adapting the authoring process by enabling the interaction with the user and updating the input of the different authoring phases according to the information that the user provides through the interaction with the robot.

List of Figures

List of Tables

List of Abbreviations

1G	First generation of wireless telephone technology
2G	Second generation of wireless telephone technology
3D	Three-dimensional space
3G	Third generation of wireless telephone technology
3GP (3GPP)	Multimedia container format defined by the Third-Generation Partnership Project
4G	Fourth generation of wireless telephone technology
A2DP	Advanced Audio Distribution Profile
AAC/AAC+	Advanced Audio Coding
ADT	Android Development Tools
AMRnb	Adaptive Multi-Rate audio codec
API	Application Programming Interface
ART	Android Runtime
BIP	Basic Imaging Profile
CDC	Connected Device Configuration
CLDC	Connected Limited Device Configuration
CLR	Common Language Runtime
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DCMC	Dynamic Cross-Model Composition

DCMI	Dublin Core Metadata Initiative
DOM	Document Object Model
DTD	Document Type Definition
DVD	Digital Versatile Disc
EFUS	Epics, Features, and User Stories
EXIF	Exchangeable Image File Format
FMMDR	Framework for Multimedia and RDF
FOAF	Friend of a Friend
GAVDP	The Generic Audio/Video Distribution Profile
GEO	Geospatial Vocabulary
GHZ	Gigahertz
GPS	Global Positioning System
GPU	Graphical Processing Unit
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
H263	Video Compression Standard
H264	MPEG-4 Part 10
HE-AAC	High-Efficiency Advanced Audio Coding
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifier
IDE	Integrated development Environment
IDF	Inverse Document Frequency
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things

IP	Internet Protocol
IRI	Internationalized Resource Identifier
IT	Information Technology
J2EE	Java 2 Enterprise Edition
J2ME	Java 2 Micro Edition
J2SE	Java 2 Standard Edition
JIT	Just-in-time compilation
JPEG/JPG	Joint Photographic Experts Group
KOS	Knowledge Organization Systems
LAN	Local Area Network
LBS	Location Based Service
LOD	Linked Open Data
MAC	Media Access Control
MBIT/S	Megabit per second
MIDP	Mobile Information Device Profile
MMS	Multimedia Messaging Service
MP3	MPEG-1 or MPEG-2 Audio Layer III
MP4	MPEG-4 Part 14 format
MPEG-21	Moving Picture Experts Group standards ISO/IEC 21000
MPEG-7	A multimedia content description standard (ISO/IEC 15938)
MSE	Media Source Extension
NAP	Network Access Point
NDK	Native Development Kit
NFC	Near Field Communication
NFR	Non-functional Requirements

NtF	Networking Framework
OGD	Open Government Data
OMA	Open Mobile Alliance
OOP	Object Oriented Programming
OS	Operating System
OWL	Web Ontology Language
OWL-S	Web Ontology Language for Web Services
P2P	Peer-to-peer
PAN	Personal Area Network
PANU	Personal Area Network User
PC	Personal Computer
PDA	Personal Digital Assistant
PNG	Portable Network Graphics
POI	Point of Interest
QR	Quick Response Code
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RESTful	Representational State Transfer
RFCOMM	Radio Frequency Communication
RS-232	Recommendation standard for serial port communication
RSTP	Rapid Spanning-Tree Protocol
SDK	Software Development Kit
SDP	Service Discovery Protocol
SIG	Special Interest Group
SIM	Subscriber Identity Module

SKOS	Simple Knowledge Organization Systems
SMIL	Synchronized Multimedia Integration Language
SMS	Short Message Service
SOA	Service-Oriented Architecture
SoC	Separation of Concerns
SOI	Subject of Interest
SPARQL	Protocol and RDF Query Language
SSP	Serial Port Protocol
SQL	Structured Query Language
SVG	Scalable Vector Graphics
TCP/IP	Transmission Control Protocol/Internet Protocol.
TF	Term Frequency
TV	Television
UGC	User Generated content
UIQ	User Interface Quartz
URI	Uniform Resource Identifier
USB	Universal Serial Bus
VDP	Video Distribution Profile
W3C	World Wide Web Consortium
WHATWG	Web Hypertext Application Technology Working Group
WI-FI	Wireless local area networking technology
WLAN	Wireless Local Area Network
WMA	Windows Media Audio
WML	Wireless Markup Language
WMV	Windows Media Video

WPAN	Wireless Personal Area Network
WWW	World Wide Web
XAML	Extensible Application Markup Language
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language

Index

References

ABU-NAIM, B., AND KLAS, W. Smart authoring and sharing of multimedia content in personal area networks based on Subject of Interest. In *2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 1–6.

AHLERS, D., AND BOLL, S. 2009. Beyond position - spatial context for mobile information retrieval systems. In *2009 6th Workshop on Positioning, Navigation and Communication*. I E E E, Piscataway, 129–134.

ALLEN, J.F. 1983. Maintaining Knowledge About Temporal Intervals. *Commun. ACM 26*, 11, 832–843. http://doi.acm.org/10.1145/182.358434.

ANDROJENA. 2017. *lencinhaus/androjena*. https://github.com/lencinhaus/androjena/releases. Accessed 24 January 2017.

ANJA JENTZSCH, RICHARD CYGANIAK,CHRIS BIZER. 2011. *State of the LOD Cloud*. http://lod-cloud.net/state/. Accessed 19 May 2015.

APACHE ORG. 2017a. *Apache Lucene - Welcome to Apache Lucene*. https://lucene.apache.org/. Accessed 24 January 2017.

APACHE ORG. 2017b. *JENA Framework*. https://jena.apache.org/. Accessed 9 February 2017.

APACHE ORG. 2017c. *Apache Jena - Reasoners and rule engines: Jena inference support*. https://jena.apache.org/documentation/inference/. Accessed 27 February 2017.

APPLE INC. 2015a. *Cocoa (Touch)*. https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/Cocoa.html. Accessed 10 January 2017.

APPLE INC. 2015b. *Siri*. http://www.apple.com/ios/siri/?cid=oas-us. Accessed 19 May 2015.

APPLE INC. 2017a. *Apple*. http://www.apple.com/. Accessed 10 January 2017.

APPLE INC. 2017b. *iPhone*. http://www.apple.com/iphone/. Accessed 10 January 2017.

APPLE INC. 2017c. *Swift - Apple Developer*. https://developer.apple.com/swift/. Accessed 10 January 2017.

ARC ELECTRONICS. 2016. *RS232 Tutorial on Data Interface and cables*. http://www.arcelect.com/rs232.htm. Accessed 25 July 2016.

AYAAN MOHAMUD. 2011. *Mobile Social Networking Audience Grew 44 Percent Over Past Year in EU5*. http://www.comscore.com/Insights/Press-Releases/2011/11/Mobile-Social-Networking-Audience-Grew-44-Percent-Over-Past-Year-in-EU5. Accessed 19 May 2015.

AZEVEDO, R.G.D.A., SANTOS, R.C.M., ARAÚJO, E.C., SOARES, L.F.G., AND SOARES NETO, C.D.S. 2013. Multimedia authoring based on templates and semi-automatic generated wizards. In *Proceedings of the 2013 ACM symposium on Document engineering*, S. MARINAI AND K. MARRIOTT, Eds. ACM, New York, NY, 205.

BARRENHO, F., ROMÃO, T., MARTINS, T., AND CORREIA, N. 2006. InAuthoring Environment: Interfaces for Creating Spatial Stories and Gaming Activities. In *Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*. ACM, New York, NY, USA.

BERTOLOTTI, P., GAGGI, O., AND SAPINO, M.L. 2006. Dynamic context adaptation in multimedia documents. In *Applied computing 2006. The 21st annual ACM Symposium on Applied Computing : proceedings of the 2006 ACM Symposium on Applied Computing : Dijon, France 2006*, H. M. HADDAD, Ed. ACM Press, New York, New York, USA, 1374.

BIZER, C., HEATH, T., AND BERNERS-LEE, T. 2009. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems 5*, 3, 1–22.

BLUETOOTH SIG. 2016. *Bluetooth Technology Website*. https://www.bluetooth.com/. Accessed 6 July 2016.

BLUETOOTH SIG - CORE 5.0. 2017. *Bluetooth® 5 quadruples range, doubles speed, increases data broadcasting capacity by 800%*. https://www.bluetooth.com/news/pressreleases/2016/06/16/-bluetooth5-quadruples-rangedoubles-speedincreases-data-broadcasting-capacity-by-800.

BLUETOOTH SIG - CORE SPECIFICATIONS. 2017. *Bluetooth core specifications*. https://www.bluetooth.com/specifications/bluetooth-core-specification/technical-considerations.

BLUETOOTH SIG - PAN PROFILE. 2016. *Personal Area Networking Profile | Bluetooth Development Portal*. https://developer.bluetooth.org/TechnologyOverview/Pages/PAN.aspx. Accessed 23 July 2016.

BLUETOOTH SIG - SPP PROFILE. 2016. *Serial Port Profile | Bluetooth Development Portal*. https://developer.bluetooth.org/TechnologyOverview/Pages/SPP.aspx. Accessed 23 July 2016.

BOLL, S., AND KLAS, W. 1999. ZYX -- A Semantic Model for Multimedia Documents and Presentations. In *Database Semantics: Semantic Issues in Multimedia Systems*, R. MEERSMAN, Z. TARI AND S. STEVENS, Eds. Springer US, Boston, MA, 189–209.

BOOCH, G., AND BOOCH, G.O.-O.A.A.D.W.A. 2007. *Object-oriented analysis and design with applications*. The Addison-Wesley object technology series. Addison-Wesley, Upper Saddle River, NJ, Harlow.

BULTERMAN, D.C.A., CESAR, P., AND GUIMARÃES, R.L. 2013. Socially-aware multimedia authoring. *ACM Trans. Multimedia Comput. Commun. Appl. 9*, 1s, 1–23.

CESAR, P., VAISHNAVI, I., KERNCHEN, R., MEISSNER, S., HESSELMAN, C., BOUSSARD, M., SPEDALIERI, A., BULTERMAN, D.C., AND GAO, B. 2008. Multimedia adaptation in ubiquitous environments. In *DocEng'08. Proceedings of the Eight ACM Symposium on Document Engineering São Paulo, Brazil, September 16-19, 2008*, D. C. A. BULTERMAN, Ed. ACM Press, New York NY, 275.

CHEVERST, K., DAVIES, N., MITCHELL, K., FRIDAY, A., AND EFSTRATIOU, C. Developing a context-aware electronic tourist guide. In *the SIGCHI conference*, T. TURNER AND G. SZWILLUS, Eds., 17–24.

CLARIFAI. 2017. *Clarifai API*. https://developer.clarifai.com/. Accessed 19 January 2017.

COHN, M. 2004. *User stories applied. For agile software development / Mike Cohn*. The Addison-Wesley signature series. Addison-Wesley, Boston, Mass., London.

D'SOUZA, M., POSTULA, A., BERGMANN, N., AND ROS, M. 2005. A Multimedia Guidebook Implementation Using a Bluetooth Wireless Information Point Network. In *Proceedings of the 3rd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*. ACM, New York, NY, USA, 33–38.

DASIOPOULOU, S., KOMPATSIARIS, I., AND NIXON, LYNDON J. B. 2007. A Framework for Ontology-driven Multimedia Analysis and Composition. In *Proceedings of the Eight International Workshop on Image Analysis for Multimedia Interactive Services*. IEEE Computer Society, Washington, DC, USA, 47-.

DBPEDIA ORG. 2017. *DBpedia*. http://wiki.dbpedia.org/. Accessed 25 January 2107.

DUBLIN CORE - TERMS. 2015. *FAQ/DC and DCTERMS Namespaces - DCMI_MediaWiki*. http://wiki.dublincore.org/index.php/FAQ/DC_and_DCTERMS_Namespaces. Accessed 30 May 2015.

DUBLIN CORE METADATA INITIATIVE (DCMI). 2016. *DCMI Home: Dublin Core® Metadata Initiative (DCMI)*. http://dublincore.org/. Accessed 12 August 2016.

ECONOMOU, D., GAVALAS, D., KENTERIS, M., AND TSEKOURAS, G.E. 2008. Cultural Applications for Mobile Devices: Issues and Requirements for Authoring Tools and Development Platforms. *SIGMOBILE Mob. Comput. Commun. Rev. 12*, 3, 18–33. http://doi.acm.org/10.1145/1462141.1462145.

EIDENBERGER, H., BOLL, S., CHRISTODOULAKIS, S., DIVOTKEY, D., LEOPOLD, K., MARTIN, A., PEREGO, A., SCHERP, A., AND TSINARAKI, C. 2007. Integrated Authoring, Annotation, Retrieval, Adaptation, Personalization, and Delivery for Multimedia. In *Proceedings of the 1st International Conference on Digital Libraries: Research and Development*. Springer-Verlag, Berlin, Heidelberg, 87–103.

EXIF.ORG. 2016. *EXIF.org | EXIF and related resources*. http://www.exif.org/. Accessed 12 August 2016.

FAN, M., LIU, Q., TANG, H., AND CHIU, P. 2015. POLI: Interactive multimedia authoring and retrieval on mobile phone by leveraging its audio channel and coded light. In *IEEE International Conference on Multimedia & Expo workshops (ICMEW). June 29, 2015 - July 3, 2015, Turin, Italy*. IEEE, Piscataway, NJ, 1–6.

FITZEK, FRANK H. P., REICHERT, FRANK (Eds.). 2007. *Mobile Phone Programming. and its Application to Wireless Networking*. Springer Netherlands.

FOUNDATION, E., AND INC. 2017. *Eclipse - The Eclipse Foundation open source community website.* https://eclipse.org/. Accessed 10 January 2017.

GEONAMES. 2017. *GeoNames API*. http://www.geonames.org/. Accessed 19 January 2017.

GOOGLE. 2016. *Google Cardboard – Google VR*. https://vr.google.com/cardboard/. Accessed 4 April 2017.

GOOGLE. 2017. *Android*. https://www.android.com/. Accessed 27 February 2017.

GRADY, R.B. 1992. *Practical software metrics for project management and process improvement*. Prentice Hall, Englewood Cliffs, NJ.

GRIMNES, G., EDWARDS, P., AND PREECE, A. 2008. Instance Based Clustering of Semantic Web Resources. In *The Semantic Web: Research and Applications*, S. BECHHOFER, M. HAUSWIRTH, J. HOFFMANN AND M. KOUBARAKIS, Eds. Springer Berlin Heidelberg, 303–317.

GUNYHO, G., AND GUTIÉRREZ PLAZA, J. 2011. Evolution of Longer-Term Planning in a Large Scale Agile Project – F-Secure’s Experience. In *Agile processes in software engineering and extreme programming. 12th international conference, XP 2011, Madrid, Spain, May 10-13, 2011, proceedings / [edited by] Alberto Sillitti … [et al.]*, A. SILLITTI, Ed. Springer, Heidelberg, 306–315.

HAUSENBLAS, M. 2011. Utilising linked open data in applications. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics, WIMS'11. Sogndal, Norway, May 25-27, 2011*, R. AKERKAR, Ed. ACM, New York, 1.

HELLER, R.S., MARTIN, C.D., HANEEF, N., AND GIEVSKA-KRLIU, S. 2001. Using a Theoretical Multimedia Taxonomy Framework. *J. Educ. Resour. Comput. 1*, 1es. http://doi.acm.org/10.1145/376697.376701.

HOFFER, J.A., GEORGE, J.F., AND VALACICH, J.S. 2002. *Modern systems analysis and design*. Prentice Hall, Upper Saddle River, N.J., Great Britain.

HONG, C., AND KIM, Y. 2012. The multimedia authoring in collaborative e-learning system. In *2012 6th International Conference on New Trends in Information Science, Service Science and Data Mining (ISSDM2012)*, 158–161.

HOSOKAWA, Y. 2008. A Location-aware Information Browser Implemented on BREW-based Mobile Phones. In *Proceedings of the 2008 ACM Symposium on Applied Computing*. ACM, New York, NY, USA, 1878–1883.

HTML - PUBLIC DOCUMENT. 1992. *Tags used in HTML.* https://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html. Accessed 11 January 2017.

HTML HISTORY. 2017. *HTML5 Introduction*. http://www.w3schools.com/html/html5_intro.asp. Accessed 11 January 2017.

IEEE 802.11 WLAN. 2017. *IEEE 802.11. The Working Group Setting the Standards for Wireless LANs*. http://www.ieee802.org/11/. Accessed 11 January 2017.

IEEE 802.15 - WPAN. 2008. *IEEE 802. The Working Group Setting the Standards for Wireless PANs*. http://www.ieee802.org/15/pub/TG2.html. Accessed 10 January 2017.

IEEE STANDARDS COMMITTEE. 2016. *LMSC, LAN/MAN Standards Committee (Project 802)*. http://www.ieee802.org/. Accessed 29 August 2016.

INTELLIJ. 2017. *IntelliJ IDEA*. https://www.jetbrains.com/idea/.

JANNACH, D., LEOPOLD, K., TIMMERER, C., AND HELLWAGNER, H. 2006. A knowledge-based framework for multimedia adaptation. *Appl Intell 24*, 2, 109–125.

JAVA. 2017. *Java technology. Oracle Technology Network for Java Developers | Oracle Technology Network | Oracle*. http://www.oracle.com/technetwork/java/index.html. Accessed 11 January 2017.

JAVA ME. 2017. *Java ME Technology - CDC*. http://www.oracle.com/technetwork/java/javame/tech/index-jsp-139293.html. Accessed 10 January 2017.

JOKELA, T., LEHIKOINEN, J.T., AND KORHONEN, H. 2008. Mobile Multimedia Presentation Editor: Enabling Creation of Audio-visual Stories on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 63–72.

JOURDAN, M., LAYAÏDA, N., ROISIN, C., SABRY-ISMAÏL, L., AND TARDIF, L. Madeus, and authoring environment for interactive multimedia documents. In *the sixth ACM international conference*, W. EFFELSBERG AND B. C. SMITH, Eds., 267–272.

KARULF, E. 2008. *Body Area Networks (BAN)*. http://www.cse.wustl.edu/~jain/cse574-08/ftp/ban/index.html. Accessed 4 April 2017.

KENTERIS, M., GAVALAS, D., AND ECONOMOU, D. 2009. An innovative mobile electronic tourist guide application. *Personal and Ubiquitous Computing 13*, 2, 103–118. http://dx.doi.org/10.1007/s00779-007-0191-y.

KERAMANE, C., AND DUDA, A. 1997. Operator based composition of structured multimedia presentations. In *From Multimedia Services to Network Services: 4th International COST 237 Workshop Lisboa, Portugal, December 15-19, 1997 Proceedings*, A. DANTHINE AND C. DIOT, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 1–32.

KLAS, W., AND SHETH, A. 1998. *Multimedia data management. Using metadata to integrate and apply digital media / Amit Sheth, editor, Wolfgang Klas, editor*. McGraw-Hill series on data warehousing and data management. McGraw-Hill, New York, London.

KRÖSCHE, J., BALDZER, J., AND BOLL, S. 2004. MobiDENK-Mobile Multimedia in Monument Conservation. *IEEE MultiMedia 11*, 2, 72–77. http://dx.doi.org/10.1109/MMUL.2004.1289043.

LABORIE, S., EUZENAT, J., AND LAYAÏDA, N. 2007. Multimedia document summarization based on a semantic adaptation framework. In *Proceedings of the 2007 international workshop on Semantically aware document processing and indexing*, M. NANARD, Ed. ACM, New York, NY, 87.

LAIOLA GUIMARÃES, R., BULTERMAN, D., CESAR, P., AND JANSEN, J. 2014. Synchronizing Web Documents with Style. In *WebMedia'14. Proceedings of the 20th Brazilian Symposium on Multimedia and the Web : November 18-23, 2014, João Pessoa, Brazil*, R. KULESZA AND T. TAVARES, Eds. ACM, New York, 151–158.

LEFFINGWELL, D. 2011. *Agile software requirements. Lean requirements practices for teams, programs, and the enterprise / Dean Leffingwell*. The Agile software development series. Addison-Wesley, Boston, Mass., London.

LIMO FOUNDATION. 2017. *LiMo Foundation - Wikipedia*. https://en.wikipedia.org/w/index.php?oldid=749309941. Accessed 10 January 2017.

LITTLE, T. 2003. Multimedia. In *Encyclopedia of Computer Science*, A. RALSTON, E. D. REILLY AND D. HEMMENDINGER, Eds. John Wiley and Sons Ltd, Chichester, West Sussex, England, Hoboken, NJ, USA, 1196–1202.

LOD. 2014. *State of the LOD Cloud*. http://lod-cloud.net/state/. Accessed 11 January 2017.

LOD 2011. 2011. *The Linking Open Data cloud diagram*. http://lod-cloud.net/. Accessed 11 January 2017.

LOUI, A., KRAUS, B., AND RIEK, J. 2005. Rhythmpix: A Multimedia Composition and Albuming System for Consumer Images. In *2005 IEEE international conference on multimedia and expo (ICME)*. IEEE, 494–497.

MAEMO OS. 2007. *maemo.org - Intro: The Home of the Maemo Community*. http://maemo.org/intro/. Accessed 10 January 2017.

MALAKA, R., AND ZIPF, A. 2000. DEEP MAP: Challenging IT Research In The Framework Of A Tourist Information System. In *Information and Communication Technologies in Tourism 2000*, D. R. FESENMAIER, S. KLEIN AND D. BUHALIS, Eds. Springer Vienna, Vienna, 15–27.

MICROSOFT. 2014a. *Microsoft officially welcomes the Nokia Devices and Services business*. http://news.microsoft.com/2014/04/25/microsoft-officially-welcomes-the-nokia-devices-and-services-business/. Accessed 8 December 2014.

MICROSOFT. 2014b. *MSDN - Windows Phone. Multimedia*. http://msdn.microsoft.com/en-us/library/windows/apps/bg182883.aspx.

MICROSOFT. 2015a. *Cortana*. http://windows.microsoft.com/en-us/windows-8/cortana. Accessed 19 May 2015.

MICROSOFT. 2015b. *Hello World: Windows 10 Available on July 29*. https://blogs.windows.com/windowsexperience/2015/06/01/hello-world-windows-10-available-on-july-29/#JsTVZpVRpducXRwy.97. Accessed 20 March 2017.

MICROSOFT. 2017a. *Microsoft HoloLens*. https://www.microsoft.com/microsoft-hololens/en-us. Accessed 4 April 2017.

MICROSOFT. 2017b. *Windows Continuum for Windows 10 Phones & Mobile | Microsoft*. https://www.microsoft.com/en-us/windows/continuum. Accessed 20 March 2017.

MURRAY, L., CARRINGTON, D., AND STROOPER, P. 2004. An Approach to Specifying Software Frameworks. In *Proceedings of the 27th Australasian Conference on Computer Science - Volume 26*. Australian Computer Society, Inc, Darlinghurst, Australia, Australia, 185–192.

MURTHY, U., AHUJA, K., MURTHY, S., AND FOX, E.A. 2006. SIMPEL. In *Opening information horizons*. *6th ACMIEEE-CS Joint Conference on Digital Libraries*, G. MARCHIONINI, M. L. NELSON AND C. C. MARSHALL, Eds. ACM, New York, 377.

NFC. 2016. *Near Field Communication*. http://nearfieldcommunication.org/. Accessed 10 January 2017.

NOKIA. 2009. *Python_for_S60. Runtimes*. about:neterror?e=dnsNotFound&u=http%3A//www.forum.nokia.com/Tools_Docs_and_Code/Tools/Runtimes/Python_for_S60&c=UTF-8&f=regular&d=Firefox%20can%E2%80%99t%20find%20the%20server%20at%20www.forum.nokia.com. Accessed 24 December 2009.

OMA. 2017. *Open Mobile Alliance. Mobile Phone Standards & Specifications | OMA*. http://www.openmobilealliance.org/. Accessed 10 January 2017.

OMA - MMS. Multimedia Messaging Service.

OPENMOKO OS. 2013. *Openmoko*. http://wiki.openmoko.org/wiki/Main_Page. Accessed 10 January 2017.

ORACLE. 2016a. *Generic Types (The Java™ Tutorials > Learning the Java Language > Generics (Updated)). Generic Types*. https://docs.oracle.com/javase/tutorial/java/generics/types.html. Accessed 29 August 2016.

ORACLE, J. 2016b. *The Annotations API*. https://docs.oracle.com/javase/tutorial/java/annotations/basics.html. Accessed 18 January 2017.

ORACLE, J. 2016c. *The Reflection API*. https://docs.oracle.com/javase/tutorial/reflect/. Accessed 18 January 2017.

PARNAS, D.L. 1972. On the criteria to be used in decomposing systems into modules. *Commun. ACM 15*, 12, 1053–1058.

PAYTHON. 2017. *Here's What Python Is*. http://python.about.com/od/gettingstarted/ss/whatispython.htm#step2. Accessed 11 January 2017.

POSPISCHIL, G., UMLAUFT, M., AND MICHLMAYR, E. 2002. Designing LoL@, a Mobile Tourist Guide for UMTS. In *Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction*. Springer-Verlag, London, UK, UK, 140–154.

QTOPIA OS. 2006. *Qtopia*. http://qpe.sourceforge.net/. Accessed 10 January 2017.

RABBATH, M., SANDHAUS, P., AND BOLL, S. 2010. Automatic creation of photo books from stories in social media. In *WSM '10. Proceedings of the 2nd ACM SIGMM Workshop on Social Media : October 25, 2010, Firenze, Italy*, S. BOLL AND R. JIN, Eds. Association for Computing Machinery, New York, N.Y., 15.

S. BOLL, AND W. KLAS. 2001. ZYX-a multimedia document model for reuse and adaptation of multimedia content. *IEEE Transactions on Knowledge and Data Engineering 13*, 3, 361–382.

S. BOLL, W. KLAS, AND U. WESTERMANN. A Comparison of Multimedia Document Models Concerning Advanced Requirements.

S. BOLL, W. KLAS, AND U. WESTERMANN. 1999. Multimedia document models-sealed fate or setting out for new shores? In *Multimedia Computing and Systems, 1999. IEEE International Conference on*, 604-610 vol.1.

SCHERP, A. 2008. Canonical processes for creating personalized semantically rich multimedia presentations. *Multimedia Systems 14*, 6, 415–425.

SCHERP, A., AND BOLL, S. 2005a. Context-driven Smart Authoring of Multimedia Content with xSMART. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*. ACM, New York, NY, USA, 802–803.

SCHERP, A., AND BOLL, S. 2005b. *MM4U: A Framework for Creating Personalized Multimedia Content*. IGI Global. http://www.igi-global.com/ViewTitle.aspx?TitleId=25976.

SCHERP, A., AND BOLL, S. 2005c. Paving the Last Mile for Multi-Channel Multimedia Presentation Generation. In *Proceedings of the 11th International Multimedia Modelling Conference*. IEEE Computer Society, Washington, DC, USA, 190–197.

SCHMIDT, D.C., AND BUSCHMANN, F. 2003. Patterns, Frameworks, and Middleware: Their Synergistic Relationships. In *Proceedings of the 25th International Conference on Software Engineering*. IEEE Computer Society, Washington, DC, USA, 694–704.

SHAW, M. 2011. Modularity for the modern world, 1.

SIKOS, L.F. 2015. *Mastering structured data on the Semantic Web. From HTML5 microdata to linked open data*. The expert's voice in web development. Apress, [Berkeley, CA].

SNAP INC. 2016. *Snapchat*. https://www.snapchat.com/. Accessed 24 October 2016.

SOMMERVILLE, I. 2001. *Software engineering*. International computer science series. Addison-Wesley, Harlow, England.

STEPANOV, A., AND MUSSER, D. 2017. *Generic Programming*. http://www.generic-programming.org/. Accessed 13 February 2017.

SYMBIAN. 2016. *Symbian Foundation*. http://licensing.symbian.org/. Accessed 11 January 2017.

TANENBAUM, A.S., AND WETHERALL, D. 2011. *Computer networks*. Pearson Prentice Hall, Boston.

TARR, P., HARRISON, W., OSSHER, H., FINKELSTEIN, A., NUSEIBEH, B., AND PERRY, D. 2001. Workshop on multi-dimensional separation of concerns in software engineering (workshop session), 809–810.

TECHTERMS. 2016. *Serial Port Definition*. http://techterms.com/definition/serialport. Accessed 25 July 2016.

TENG, C.-M., WU, C.-I., CHEN, Y.-C., CHU, H.-H., AND HSU, J.Y.-J. 2004. Design and Evaluation of mProducer: A Mobile Authoring Tool for Personal Experience Computing. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia*. ACM, New York, NY, USA, 141–148.

THE ANDROID OPEN SOURCE PROJECT. 2017. *MMS Application. platform/packages/apps/Mms - Git at Google*. https://android.googlesource.com/platform/packages/apps/Mms/+/master. Accessed 17 January 2017.

TUMMALA, H., AND JONES, J. 2005. Developing Spatially-aware Content Management Systems for Dynamic, Location-specific Information in Mobile Environments. In *Proceedings of the 3rd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*. ACM, New York, NY, USA, 14–22.

TWITTER INC. 2016. *Twitter. It's what's happening*. https://twitter.com/?lang=en. Accessed 24 October 2016.

W3C - SWRL. 2010. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. https://www.w3.org/Submission/SWRL/#2.2. Accessed 18 January 2017.

W3C ORG. 2009. *W3C Document Object Model*. https://www.w3.org/DOM/. Accessed 14 April 2017.

W3C ORG. 2011. *Scalable Vector Graphics (SVG) 1.1 (Second Edition)*. https://www.w3.org/TR/SVG/. Accessed 11 July 2016.

W3C ORG. 2014a. *Concise Bounding Description*. https://www.w3.org/Submission/CBD/. Accessed 25 January 2104.

W3C ORG. 2014b. *HTML5*. https://www.w3.org/TR/html5/. Accessed 11 July 2016.

W3C ORG. 2014c. *OWL 2 Web Ontology Language. New Features and Rationale (Second Edition)*. https://www.w3.org/TR/2012/REC-owl2-new-features-20121211/. Accessed 11 January 2017.

W3C ORG. 2014d. *RDF 1.1 Primer*. https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/#section-triple. Accessed 11 July 2016.

W3C ORG. 2014e. *RDF Schema 1.1*. https://www.w3.org/TR/2014/REC-rdf-schema-20140225/. Accessed 11 July 2016.

W3C ORG. 2014f. *Synchronized Multimedia Integration Language (SMIL 3.0)*. https://www.w3.org/TR/REC-smil/. Accessed 11 July 2016.

W3C ORG. 2016a. *Data - W3C*. https://www.w3.org/standards/semanticweb/data. Accessed 26 July 2016.

W3C ORG. 2016b. *Semantic Web - W3C*. https://www.w3.org/standards/semanticweb/. Accessed 26 July 2016.

W3C ORG. 2016c. *SPARQL Current Status - W3C*. https://www.w3.org/standards/techs/sparql#w3c_all. Accessed 25 July 2016.

WEIß, D., DUCHON, M., FUCHS, F., AND LINNHOFF-POPIEN, C. 2008. Context-aware personalization for mobile multimedia services. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*, G. KOTSIS, Ed. ACM, New York, NY, 267.

WHATSAPP INC. 2016. *WhatsApp Web*. https://web.whatsapp.com/. Accessed 24 October 2016.

WI-FI. 2017. *Wi-Fi Alliance*. http://www.wi-fi.org/. Accessed 10 January 2017.

WIKIMEDIA. 2017a. *File:Male Lion on Rock.jpg - Wikimedia Commons*. https://commons.wikimedia.org/wiki/File:Male_Lion_on_Rock.jpg. Accessed 30 May 2017.

WIKIMEDIA. 2017b. *File:Tiger in Ranthambhore.jpg - Wikimedia Commons*. https://commons.wikimedia.org/wiki/File:Tiger_in_Ranthambhore.jpg. Accessed 30 May 2017.

WIKIPEDIA. 2014. *Mobile Operating System*. http://en.wikipedia.org/wiki/Mobile_operating_system. Accessed 8 December 2014.

WIKIPEDIA. 2016a. *Digital container format - Wikipedia*. https://en.wikipedia.org/w/index.php?oldid=744719939. Accessed 11 January 2017.

WIKIPEDIA. 2016b. *User story - Wikipedia, the free encyclopedia*. https://en.wikipedia.org/w/index.php?oldid=725116416. Accessed 16 July 2016.

WIKIPEDIA. 2016c. *Windows Media Video - Wikipedia*. https://en.wikipedia.org/w/index.php?oldid=755488941. Accessed 11 January 2017.

WIRFS-BROCK, R.J., AND JOHNSON, R.E. 1990. Surveying current research in object-oriented design. *Commun. ACM 33*, 9, 104–124.

XIAO, J., LYONS, N., ATKINS, C.B., GAO, Y., CHAO, H., AND ZHANG, X. 2010. iPhotobook: Creating Photo Books on Mobile Devices. In *Proceedings of the International Conference on Multimedia*. ACM, New York, NY, USA, 1551–1554.

XMP. 2016. *ISO 16684-1:2012 - Graphic technology -- Extensible metadata platform (XMP) specification -- Part 1: Data model, serialization and core properties*. http://www.iso.org/iso/catalogue_detail?csnumber=57421. Accessed 12 August 2016.

YIN, W., MEI, T., AND CHEN, C.W. 2013. Automatic generation of social media snippets for mobile browsing. In *MM '13. Proceedings of the 2013 ACM Multimedia Conference : October 21-25, 2013, Barcelona, Spain*, A. (JAIMES, N. SEBE, N. BOUJEMAA, D. GATICA-PEREZ, D. A. SHAMMA, M. WORRING AND R. ZIMMERMANN, Eds. ACM, New York, 927–936.

ZUBY EZEASOR, MARTIN GORM PEDERSEN. 2013. *Guideline: Writing Requirements Statements*. http://epf.eclipse.org/wikis/openup/core.tech.common.extend_supp/guidances/guidelines/writing_good_requirements_48248536.html. Accessed 3 August 2016.