A Service Virtualization Architecture for Efficient Multimedia Delivery

by

Elena Korotich

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.Sc. degree in
Computer Science

School of Information Technology and Engineering
Faculty of Computer Science
University of Ottawa

Abstract

This thesis provides a novel architecture for the creation and management of virtual multimedia adaptation services offered by a multimedia-enabled cloud.

The aim of the proposed scheme is to provide an optimal yet a transparent user access to adapted media contents while isolating them from the heterogeneity of the utilized devices, diversity of media formats, as well as the details of the adaptation services and performance variations of the underlying network. This goal is achieved through the development of service virtualization models that provide various levels of abstraction of the actual physical services and their performance parameters. Such virtual models offer adaptation functions by comprising adaptation services with accordance to their parameters. Additionally, parameters describing the functional specifics of the adaptation functions, as well as multimedia content features, are organized into a hierarchical structure that facilitates extraction of the virtual models capable of satisfying the conditions expressed by the user requests. At the same time the paramter/feature organization structure itself is flexible enough to allow users to specify media delivery requests at various levels of request details (e.g., summarize video vs. drop specific frames).

As a result, in response to a user request for a multimedia content, an optimal virtual service adaptation path is calculated, describing the needed media adaptation operations as well as the appropriate mapping to the physical resources capable of executing such functions. The selection of the adaptation path is done with the use of a novel performance-history based selection mechanism that takes into account the performance variations and relations of the services in a dynamically changing environment of multimedia clouds. A number of experiments are conducted to demonstrate the potential of the proposed work in terms of the enhanced processing time and service quality.

Acknowledgements

First and foremost I offer my sincerest gratitude to my supervisor, Dr Nancy Samaan, who has helped me to see beyond the implementation levels allowing me peer into the clouds of abstraction and who supported me throughout my thesis with her patience and knowledge.

Also my gratitude goes to all of the researchers, who have explored the endless digital fields of multimedia, adaptation and cloud computing before me, as their work and dedication inspired me at each step of my way.

Finally, I would like to extend my thanks to my family without whose support all this would not be possible. Especially to Shawn for explaining to me the possibilities of Perl, Eric for his patience, invaluable time, love and consultations about Unix. Edward for explaining the basics about technical writing and my son for giving me the light of inspiration when I need it the most.

Contents

List of Figures

List of Acronyms

DHT Dynamic Hash Table

FF First Fit

GOP Group of Pictures

IaaS Infrastructure as a service

MAPs Media Abstraction Parameters hierarchy

MCP Multi-Constraint path Problem

ML Machine Learning

MMSE Minimum Mean Square Error

OWL Web Ontology Language

PaaS Platform as a Service

PCA Principal Component Analysis

PG Performance Graphs

QoE Quality of Experience

QoS Quality of Service

RF Relevant Feedback

SaaS Software as a Service

SoA Service oriented Architectures

SON Service Overlay Network

SVA Service Virtualization Architecture

VM Virtual Manager

VSM Virtual Service Model

WSDL Web Service Description Language

Chapter 1

Introduction

1.1 Motivation

In recent years we have witnessed the introduction and fast expansion of powerful and complex multimedia enabled mobile devices such as smart-phones and tablet PCs. At the same time, technology innovations reached new domains that were never before associated with interactive media content and/or network connectivity; thus, creating smart home systems, interactive toys and electronic books capable of multimedia content creation, retrieval and delivery. As the availability of media-capable devices increases, so does the amount and variations of available content. This results in an abundance of content formats to deal with, some of which are unique and potentially proprietary. This content is usually personalized and adapted to user preferences, and device capabilities. This interest in personalized content has created a demand for methods that will quickly adapt media content while enhancing the user experience. Content adaptation provides a way to create a customized content that satisfies the robust device and network requirements, while accounting for the user preferences that sometimes can take a form of abstract specifications such as "the best visual quality". Among the multiple ways that allow for the production of such contents, dynamic adaptation methods stand out as they allow the generation of the needed content version from the same original file on a per-needed basis. At the same time, content adaptation and personalization marks the shift from the broadcasting of the same content to a group of users to the narrowcasting that still allows to distribute the same content but in formats that are tuned to user's personal needs and context.

Besides the wide variety of multimedia enabled devices available to the user, there also exists a diversity in the content delivery mechanisms that vary from satellite, radio broadcasting, cable, mobile, and copper using xDSL [56]. While the traditional content delivery model involves a user and a set of content servers controlled by a network administrator, in the popular, but less reliable peer-to-peer delivery model users are interconnected and act as the source of the media content for each other. At the same time intermediate network nodes that are used to route traffic from the source to the destination are extended with processing capabilities, transforming them into active and programmable network services united under the cloud computing paradigm. Cloud computing concepts comprise existing technologies such as wide spread service virtualization, utility computing and distributed computing to "outsource the provision of the computing infrastructure required to host services" [74]. By offering a variety of accessible and dynamically reconfigurable virtualized services on a pay-per-use basis, clouds present an attractive solution for resource-demanding content adaptation and customization. However, it becomes challenging to utilize the existing content adaptation and customization methods and models with the emerging cloud computing paradigm. As cloud customers pay for services on a per needed basis, it requires a dynamic and cost efficient solutions for locating effective sequences of adaptation services in a timely fashion, so that systems that lack processing capabilities can use them as assembly points for content adaptation and personalization purposes.

1.2 Thesis Objectives

In order to provide the needed level of customization, the systems need cloud resource management to expose the requested content through one or more adaptation/processing services, thus requiring a flexible architectural solution for selection and navigation through available virtualized services offered by the cloud. This includes the selection of the most appropriate adaptation functions among the wide variety of available services, as well as the fast discovery of their most efficient execution order.

In the context of the increasing complexity of content processing, delivery and multimedia consumption schemes, this thesis investigates the possibility of viewing all these actions in terms of the core low-level multimedia features. Such universal features are then used to create an appealing, to an average user, yet effective mechanisms to describe service requirements. They are also used for the extraction and ordering of appropriate

adaptation services in order to create customized content at need in response to the requested preferences and device capabilities. With the review of the current adaptation methods comes a solid basis for the second objective of the thesis, namely, the design of virtual service models. Due to virtualization, such models allow to separate details of physical service operation and performance from the actual goals of the virtual service. This allows the creation of reusable elements with seamless portability to a wide range of platforms and devices, while increasing service distribution and availability. With the use of the core low-level multimedia features, such service models are organized in a structure that allows the effective identification of the needed manipulations on the content for a particular user and his needs. The set of identified elements is then transformed into execution sequences with accordance to performance-history based selection scheme. The performance-based selection scheme takes into account the service performance, so that the ordering process is performed with accordance to the current state of the service behavioral specifics.

1.3 Thesis Contribution

Given the diversity of the available content, devices, content delivery mechanisms and user requirements, a content customization system faces the challenge of providing an effective mechanism to adapt the requested content to the desired form in a reasonable amount of time. In case of a multimedia clouds, the dynamic change of network components status and performance fluctuations may negate the effectiveness of service composition methods that rely only on the semantic compatibilities of it's elements.

The main contributions of this thesis are as follows [43];
- A novel service virtualization architecture (SVA) that hides the details of the service configurations offered over service clouds as well as the details of the service selection process. At the same time SVA provides a consolidated view to end-users through the utilization of virtual adaptation services.
- In order to utilize unified modeling techniques, different types of multimedia content are reviewed in terms of their core low-level multimedia features. Such core features are then used as base elements to effectively model adaptation requests, services and their functional dependencies.
- A novel performance-history based service selection scheme that takes into account the past performance of media processing services in order to service current and future

requests.

1.4 Thesis Outline

The remainder of this thesis is structured as follows:
Chapter 2, presents the background information on the multimedia content and content adaptation, allowing for an understanding of the existing research trends and approaches, their effectiveness and challenges. At the same time, relevant context, surrounding content adaptation is also discussed, such as the establishment of multimedia services and multimedia service clouds.
Chapter 3 presents an architecture for the creation and management of virtual multimedia adaptation services. The discussion of Service Virtualization Architecture starts with virtualization techniques for media services offered over service clouds. It then proceeds to gradual multimedia parameter abstraction in a hierarchical form, that acts as an interpreter between user defined terms and the low-level multimedia features, while providing a foundation to construct and sort the list of offered virtual service models. The chapter then focuses on specifics of each of the components and how they fit into the concept.
Chapter 4, will discuss the method of forming a chain of processing services by ordering them with accordance to their performance characteristics. The chapter will elaborate on the performance metrics themselves, as well as their collection methods and statistical models/methods allowing for the expression and use of such characteristics in the compact form of performance graphs.
Chapter 5 is dedicated to the implementation of the proposed architecture in order to evaluate it's performance. A set of experiments were performed on several video file formats (i.e. MJPG and MPEG2), that were processed through a selected set of services using random, first available queue and performance graphs service ordering methods. Evaluation was conducted with the use of single services as well as complex models, when the adaptation function is offered by several services.
Chapter 6 concludes the thesis. It summarizes the contributions, experimental results and appoints the future research plans and potential enhancements that can be made to the outlined content adaptation techniques.

Chapter 2

Related Work

2.1 Chapter Objectives

This chapter represents an overview of the background information used during the design of the Service Virtualization Architecture (SVA). It starts by introducing the notion of multimedia content and providing a brief overview of different types of multimedia contents, their parameters and service models. Next, the chapter discusses content adaptation and provides an overview of its types and approaches. The chapter then focuses on modeling techniques and solutions for content adaptation in a proxy-based system. The chapter is concluded with the review of the evolution of the multimedia functionality abstractions towards services and multimedia-aware cloud computing.

2.2 Multimedia. An Overview

Over the last decade, a family of small portable personal devices have been introduced. Among their features there is an ever increasing dependence on media content delivered through the Internet. Such devices have specifications that vary widely with regards to their display resolutions, processing capabilities, storage capacity and content support. Additionally, the delivered multimedia content itself keeps evolving towards richer content and interactive experiences. Multimedia itself can be described as a channel or a medium that allows humans to experience the real-world or it's artificial model through a combination of human senses. Defined by space and time, multimedia can be experienced through creation, manipulation, delivery and consumption [71]. If setting aside unique parameters defining format specifications and compression schemes (such auto-regressive

based texture synthesis or video compression based on kinetic Delaunay triangulation), the core of a multimedia content comprises one or a combination of following components [68]:
Text - represents a discrete sequence of meaningful characters. Textural information can be described by parameters such as a language used and the adapted font characteristics. It can be also combined into complex web objects encapsulating additional parameters such as color, shades and emboss.
Images and Graphics - represent a spatial presentation of real or virtual entities. In general, a digital image is formed by a plane of pixels, where each pixel is assigned a range of bits to store it's color values. Distinct parameters, identifying such images include spatial resolution and color encoding that is measured in bits per pixel. Graphics can utilize higher-level schemes for representations of primitives (such as lines and geometric shapes) and spatial regions that can be used to improve compression and manipulation methods.
Audio - is content the medium of which is formed by pressure waves and takes the form of music, speech or sequence of digitized sounds. Its peculiar parameters include frequency, volume, sampling rate (determines the intervals used to convert a continuous signal into a digital format) and number of bytes used to quantize the sample. The audio content can also originate in a digital form from electronics instruments or a computer.
Video - represents a continuous sequence of frames which semantics depend on the level of the relative change in the discrete values between video frames or of the whole continuum. Important measures include the frame aspect ratio, frame rate and color encoding details such as luminance and chrominance.
Complex Objects - This category combines specialized structures that utilize complex timings, spacial and semantical synchronization relations or use supplementary dimensions to define 2D and 3D shapes like holography, stereophony and element of virtual reality. The addition of a dynamic layer of human interaction in a form of feedback allows to add more value to such multimedia content, while utilizing existing delivery mechanisms to compose and generate it's required form [71].

Certain features and parameters describing multimedia content are consistent throughout the entire multimedia object (e.g., frame rate, font color and sound volume) allowing their retrieval to be a relatively simple process. Other sets of parameters vary and provide a significant value when summarized over a specific region (e.g., dominating color, motion intensity). As an example, color features are widely used in the image domain

and, depending on the system and user requirements, can be described by either color histograms (that characterize both the global and local distribution of colors but does not consider the spatial information of the pixel) or color coherence vector (that characterizes the color distribution of pixels and the spatial correlation of color pairs) [11].

2.2.1 Multimedia description

Multimedia contents can be modeled by a set of low-level features (e.g., color values, dimension values and frequency). An average user, on the other hand, in it's interactions and requests tends to recognize and use a finite set of high-level semantic concepts that are represented by keywords and text descriptors. For example, psychophysical experiments show that when exploring images for similarities, human perception can recognize about 40 low-level features including parameters such as number of regions, color composition, number of edges and presence of a central object . When modeling system requests and content descriptions a sematic gap between low-level features and high-level semantic should be addressed in order to allow the creation of the concepts that are accessible by a wider user-base. For instance, an average color calculated for all pixels in a given graphic at the system level is represented by a hexadecimal color space, while at the user-level it is assigned a semantic color name used in natural language. Such mapping between sets of low-level features and high-level semantics can be established by using supervised and unsupervised machine learning techniques [78].

In supervised machine learning techniques, a large number of labeled training samples are fed into a model, allowing to locate hyper-planes to differentiate between labeled categories. At the testing stage, the unlabeled low-level data features are processed by the same model and are associated to the high-level semantic concept, that gives the highest positive match. On the downside, such feature mapping is fixed and might have to be relearned with the introduction of the new concepts or domain variations. A bootstrapping approach allows the model to train from a smaller set of labeled data. In this case, at the testing-stage information is processed by two independent classifiers, that join their efforts in further co-training and annotation. With the use of relevant feedback (RF) from the user, a decision tree can be also built while processing the requested information. Such decision trees can be later used as a separate model to process similar queries or be translated into a set of policies allowing for an automatic feature mapping. The described above methods are most consistent with mapping nominal low-level features unlike sets

of multimedia properties with continuous values. With no labeled outcome values to guide the learning process, unsupervised learning techniques focus on attempts to cluster and categorize given input features [78]. The main idea is to group properties into separate clusters in a way that maximizes the similarity between the cluster elements, while at the same time minimizing the similarity between the clusters themselves. In this case, statistical variations of elements within each cluster are used as a criteria to guide the mapping process between low-level multimedia features and high-level semantics [78].

File format represents the most widespread and standardized form of describing a multimedia content and thus can be used to supply the basic low-level features and attributes for higher-level models. As an alternative, with it's 377 complex types, 417 attributes and 1182 elements, MPEG-7 standard provides description tools to annotate multimedia content at different levels of abstraction [31]. Web Ontology Language (OWL) [62] is often used to describe high-level semantics as it provides a large vocabulary for class and property descriptions, however it does not provide means to model complex temporal and spatial relationships between video and audio content [46]. As a result, the combination of low-level features with mapping to high-level semantics allows to create an extensive content description, that can be used for content retrieval and formulation of precise system requests. However, when it comes to the creation of complex multimedia structures and utilization of existing elements in service composition chains, such descriptions come short in providing the right level of details. In such cases, preference is given to multimedia models that, among other things, can recognize and record the internal structural and temporal relationships and capture the nature of inter-element relations.

2.2.2 Multimedia Models

Multimedia models and metamodels are designed to synthesize knowledge and common concepts from different domains into a uniform view [55]. Such models allow for an exploration of semantic, temporal/spatial relationships and composition of particular multimedia elements without the burden of the implementation details.

Earlier multimedia models in order to create a uniform abstraction for stored multimedia content were using conventional object-oriented data modeling approaches, representing multimedia as an active object, while extending them with methods to address,

compression/decompresion and other functionalities specific to multimedia. Dataflow models concentrated on modeling dataflow and transformations through directed graphs of operators [65]. In [52] all multimedia types are joined under an entity *Multimedia Type*.

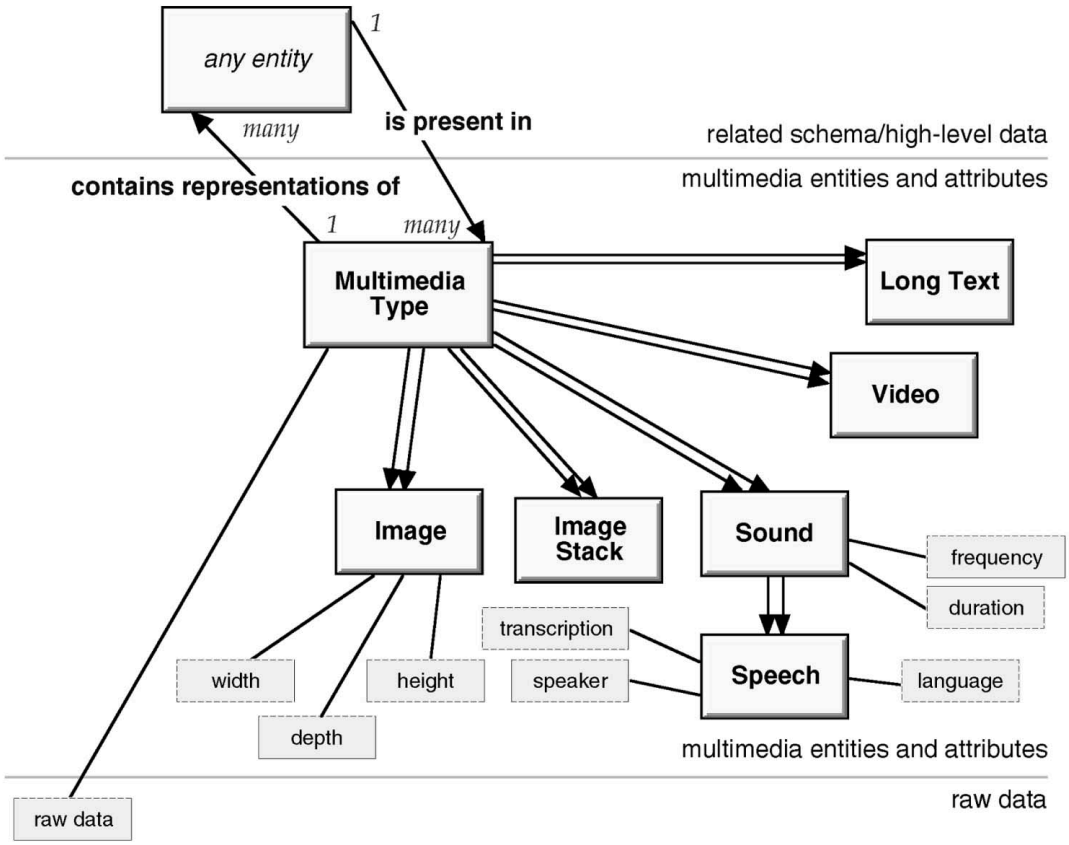


Figure 2.1: Different levels of a multimedia type model (from [52])

The low-level layer is described by a *raw data* attribute, corresponding to the raw sequence of bytes free of any interpretations or translation. Intermediate layers describe subclasses of the *Multimedia Type*, allowing to attach raw data to meaningful sub-components. Features and attributes of these sub-components are used to model service requests at different levels of abstractions. As shown in Figure 2.1, *Image* is an example of a basic multimedia type, that structurally corresponds to a two-dimensional array of pixels, each of which requires a particular number of bits to define a *depth* attribute. At the same time, *Image* can be described by the *dimension* parameter, that in it's turn is bound to attributes *width* and *hight*. Such abstractions can be further

extended with temporal models, that describe temporal dependencies between entities (such as event-based dependency) and/or spatial models, that allow to determine element positioning (such as absolute positioning or topological relations). To account for the element of time, multimedia content can be also modeled in the form of a general stream, where parameters specific to particular multimedia types (e.g., individual words for text, samples at given frequency for sound) are sequenced over time [16].

In the end, multimedia models keep shifting towards abstraction and increased support for high-level interactions and manipulation. This allows multimedia delivery systems to explore and perform needed manipulations using just the content attributes and features in an implementation-independent manner. Evolving from pre-composed streams into structured sets of objects composed and rendered at runtime, multimedia keep spreading into new domains (Figure 2.2) [33].

Domains	Example applications
Information management	Hypermedia, multimedia-capable databases, content-based retrieval
Entertainment	Computer games, digital video, and audio (MP3)
Telecommunication	Videoconferencing, shared workspaces, virtual communities
Information publishing/delivery	Online training, electronic books, and streaming media

Figure 2.2: Domains and example applications utilizing multimedia (from [33])

With the evolution of multimedia content, a challenging requirement emerges: how to offer a pleasant experience to end-users while providing an efficient delivery mechanism, in a way that suits their preferences, device capabilities and running application constraints, while considering performance variations of the underlying network [8, 26]. Along with context awareness and personalization of the user interfaces, multimedia content adaptation is identified as an essential feature for the realization of such requirement [49].

2.3 Introduction to Content Adaptation

The *content adaptation* research discipline studies the means to scale and transform a given media content in order to satisfy network and device constraints, evolving over the years to take into consideration user preferences in attempts to maximize user's quality of experience (QoE). An example of the situation when adaptation can be useful is when

a video material needs to be displayed on a smartphone. It needs to be resized to fit the display, tinted green to accommodate the user's vision challenges and converted to a different format in order to provide the correct encoding, while satisfying the bandwidth requirement.

2.3.1 Reasons and Approaches to Multimedia Content Adaptation

This section will provide an overview of the background information on the topic of the multimedia content adaptation, as well as an overview of the existing adaptation solutions and their challenges. In recent years, individual users have become a growing category in the multimedia consumption chain, driving an era where the content delivery is enhanced with varying levels of customization to target a more individualistic audience. In such a trend, several human factors become a driving point of the multimedia consumption. It all starts with the manual or automatic selection of the desired content (e.g., selecting the video title from the on-line digital theater service) while capturing and interpreting user's preferences in a standardized representation form. In the next step, the selected content is analyzed in terms of suitability to the relevant usage environment characteristics and/or context. Context can be seen as the information that describes/provides additional information about features and conditions of any given component of the multimedia delivery and consumption chains. Low-level contextual information includes and is not limited to: network conditions, user and device characteristics, as well as observations that can provide insight to the user's emotional and physical state. More advanced context-aware content adaptation systems try to explore the inter-relations among different types of the low-level contextual information, while at the same time using ontologies to infer high-level context [9]. In the end, the combination of user preferences and usage environment characteristics derive reasons for content adaptation [57]:

- *Technical limitations*, such as available bandwidth or device capabilities (such as processing power or resolution).

- *User semantic preferences*, that assign a value to a part or type of the content (e.g., during an on-line broadcast the user is interested in a news report, while requesting the weather report segment in the form of a screenshot).

- *Content provider multimedia consumption policies* with accordance to the user profile and/or network conditions. As an example, filtering the video content before delivering it to an underage user or changing the bitrate during peak-usage-hours.

- *Perception preferences and limitations* that are derived from the user's unique perception of the content in terms of senses such as vision or hearing.

At a higher scale, the adaptation approaches can be viewed as either *device-centric* or *end-user-centric*. A *Device-centric* approach produces the content tailored to the device capabilities. Nevertheless, in this case, the highest quality of the content in terms of the device requirements may not be viewed as such by the end-user of the content. On the other hand, *user-centric* approaches guide the adaptation process with accordance to the user preferences that are expressed in a form of user satisfaction metrics [44] or an acceptable range of one or more adaptation parameters. However, such an approach provides a challenge; negotiating an acceptable trade-off between the content quality requested by the user, and the version of the content that satisfies the device constraints. In order to provide adequate content, a combination of possible actions can be considered [57]:
- *Variation selection*, that provides access to a set of multiple variations of the same content (e.g., versions of the same video adapted for different screen sizes or coded at different bitrates).
- *Content Scalability*, that is achieved by a scalable coded bitstream that allows for the extraction of content representations in one or more scalability dimensions (such as temporal or spacial) from a single stream.
- *Content Transformation*, that is also commonly referred to as content adaptation, and represents a group of actions or operations that can be performed on the delivered content.
This work will focus on the last methodology of content transformation, since the former two do not scale well with the current volume of media demands.

2.3.2 Types of Content Transformations

A multimedia file can be viewed as a container with a specific amount of information represented by modality (e.g., datatype such as video, image, text and speech) and fidelity (e.g., quality or rendering requirement) [48]. Depending on the way transformations manipulate these parameters, they can be separated into the following categories [57]:

- *Transcoding* modifies fidelity of the content, while keeping the modality and the amount of information unchanged. This can be accomplished by the use of operations that manipulate such aspects of the multimedia content as color, resolution, quality etc.

- *Transmoding* is the process that changes the modality, while at the same time trying to preserve as much information as possible. Some of the transformations are quite direct (such as video to picture), other involve an amount or reasoning and interpretation (such as image to audio file).

- *Semantic Filtering* can be performed on such aspects of a multimedia file as duration, scene composition or spatial (as in region of interest) by reducing and thus rearranging the amount of information that they convey. Most common operation in this category being summarization.

Creation and delivery of customized content requires the consideration of multiple constraints; such as device capabilities, network parameters and user preferences. Incorporation of possibly contradicting requirements, such as the user's request for high quality content versus network aspiration for traffic reduction, may result in the need to perform a combination of complex content transformations. The first logical step to investigate the possible timeline of the adaptation process.

2.3.3 Adaptation Timeline: Static and Dynamic Adaptation

In terms of time, customized content can be created statically [61] beforehand or on the fly [45]. A static approach calls for the creation of multiple versions of the same content at different quality levels to fit a variety of devices. The original content is pre-adapted along the device dimensions (such as modality and fidelity), resulting in device-independent content versions generated with accordance to a set of adaptation planes bounded by values that model device dimensions [48]. Static methods allow great delivery performance in situations where the content provider is dealing with an established and consistent audience as in these cases, available content versions can be closely matched to the range of the adaptation requests or provide a set of acceptable equivalents. Static content versions can also be hand-tailored by guided transcoding process to the desired adaptation results. The major downside of this method includes storage costs, as well as the challenge to adapt the existing content to the changes in the consumption chain parameters, such as a more dynamic consumer base or introduction of new devices and formats. Other challenges present situations when the desired content

version falls in between two pre-adapted versions.

In dynamic adaptation, only the original version of the multimedia content has to be stored, as the content is transformed on the fly with accordance to the request characteristics that may include device capabilities, network parameters and user preferences. Sets of pre-adapted files can be used as an input for the dynamic content adaptation framework, marking the collaboration of two adaptation methods in order to reduce the computational overhead [48]. To be sufficient on it's own, the dynamic approach needs to provide solutions that transform multimedia content at a reasonable computational cost, while taking into account the nature of the adaptation system, location of the adaptation being one of them.

2.3.4 Location of the Dynamic Adaptation

Dynamic content adaptation can be performed on the client side, on a content server or at a proxy-server located on the network. In case of a *client-side* adaptation, the content is transformed by the client's actual device, providing the most direct way for the user to control and specify their preferences. However, depending on the adaptation request, it can be challenging or even impossible to execute a content adaptation routine due to device resource constraints [80].

When transcoding is performed on the *content-server-side,* content providers have complete control over content transformation methods, as well as the authorization of the user's requests, thus allowing for the creation of a more secure adaptation process. This provides an optimal environment for creation and storage of the adapted content versions produced with static methods. Other benefits may include reduction of the network traffic, in cases when the modified files are of a more efficient format or lower quality before they are sent into the network [7]. It is also possible to reduce service load by separating tasks. As an example, by assigning the creation of an adaptation plan to the client side proxy, while the actual adaptation process is left to the content server [69]. However, in cases of multiple requests or as new adaptation functions are being added, additional computational load is introduced to the servers, as well as functional and maintenance complexity.

Proxy-servers provide an alternative, when needed transcoding operations are performed

on an intermediate platform located between the client and the content source. Since content transformation can be executed without involving client device or content server resources, it allows for a reduction of their design complexity. That said, such advantage over other methods can be lost in case of high latency connections between proxy servers and content server components. In general, physical proxy servers host a set of fixed logical transcoding functions which are capable of exchanging requests on the user's behalf. In cases when the needed operations can't be performed on the server, its functionality can be extended with the use of adaptation tools developed externally by an automatic download of the needed plug-in components [38]. A good example of such design is Gamma : a content-adaptation server for wireless multimedia applications [81]. In addition to providing transcoding operation for videos, images, audio and documents, it can expend its capabilities by installing executables into the existing system and adding the corresponding entries to it's configuration table.

Such flexibility allows to consider the proxy-based framework platform for the creation of an elaborate adaptation system, that can provide a dynamic adaptation solution accessible by multiple organizations and users. Due to its dynamic nature, such systems require a flexible adaptation architecture, that, among other things, can address unexpected system changes, such as the failing, leaving or introduction of new transcoding services. This can become especially important in cases when complex adaptation operations are performed through combined functionalities of several adaptation proxies. Given the number of possible configurations, it present a challenge to find the chain among available services offered by transcoding servers, that will create the needed content with accordance to the adaptation request.

2.3.5 Models for Proxy-based Content Adaptation

In current proxy-based content adaptation techniques two notable research streams can be characterized - modeling content adaptation as an optimization problem or viewing it as process of creation of an adaptation path from available services.

In the optimization problem model, the optimal delivery media parameters (e.g., frame rate, frame size) are calculated while minimizing resource utilization and satisfying the users and applications requirements. The value of the adapted content version is measured by a utility function that represents a user's ease of access to the content and

content perceived quality [12]. The adaptation operations to achieve the desired output parameters are then performed on a dedicated server (e.g., [26], [21]). Authors in [27] investigated simple scenarios in order to evaluate the cost of implementing machine learning (ML) techniques (such as Multilayer perception and Bayesian Inference) to analyze user contextual information in order to define the set of most adequate media stream re-encoding parameters. Overall it is an attractive approach, allowing in a single transcoding step to produce the desired result, however, the main limitation is it's complexity and the high dimensionality of the optimization problem.

The second stream of the research efforts targets distributed adaptation through finding an optimal path of adaptation services. A service path is an ordered sequence of adaptation services between the content source and destination device that, as a result, creates a more complex function to produce a desired content version (e.g., scaling then transcoding followed by the color change). This approach is effective in cases, when a single adaptation service is not able to generate the required content version. Differences between the descriptions of the available multimedia object and required multimedia object provide a way to monitor the proximity of reaching the required content version at each step of the adaptation path. This approach implies a unified description model (such as XML schemas in MPEG-21 Digital Item Adaptation specification [70]) for multimedia content, for both adaptation services and content providers [37], thus allowing an easier construction of support mechanisms (such as service composition) to search for an adaptation path.

2.3.6 Service Composition

The problem of composing complex functional elements with the use of smaller independent entities is well studied in fields such as web services and can be generalized into a sequence of four stages (Figure 2.3) [39]:

With the use of semantic matching rules or ontologies, *translation* provides mapping between service requests and internal system specifications/logical models. When utilizing common services and functional models, this stage can be omitted. Once produced, a service request is used as a guideline to *generate* needed functionality by composing available services using semantic or syntactical matching techniques. Composition of available services can produce several composition plans in a form of graphs or formal descriptions. Composition plans are then subjected to *evaluation* in terms of given con-

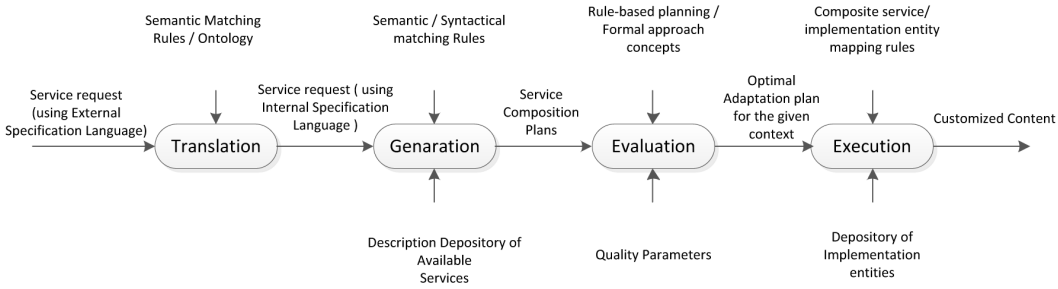


Figure 2.3: Service Composition stages (based on [39])

text, network conditions, non functional quality parameters (such as accuracy, reliability and scalability) and other properties define by the architectural logic of the processing system. Evaluation can be conducted in accordance with a rule-based plan or a more dynamic formal approach such as Petri nets that identifies basic aspects of the system, both mathematically and conceptually. During the *execution* of the selected composition plan, an appropriate implementation entity capable of realizing the identified composite service is selected. It ranges from a simple successive invocation of a list of entities to a complex contextual discovery and configuration of suitable network elements.

Restrictions driven by the nature of the multimedia content have to be considered when trying to use composition techniques for an adaptation path construction. Multimedia is usually presented in a form of a continuous data flow such as video or audio, so when services are composed together to form an advanced processing function, they have to satisfy not only the functional dependency by matching outputs with corresponding inputs, but also maintain consistent QoS to maintain a certain level of spatial and temporal quality of the delivered content. This means that in order to maintain the expected time for service instantiation (besides typical network performance measures such as bandwidth, jitter and packet loss), system have to contemplate parameters such as number of proxy hops and transmission delays, that at the same time greatly depend on selection of the composition mechanism. As an example, concurrent composition manipulates several continuous flows in parallel (Figure 2.4), thus requiring additional service monitoring in order to provide a strong functional and time synchronization.

In successive service composition functionally dependent services are invoked in a sequence (Figure 2.5). In order to process service requests within the established time frame, each successive service has to be available to service the corresponding processing stage at a relevant time.

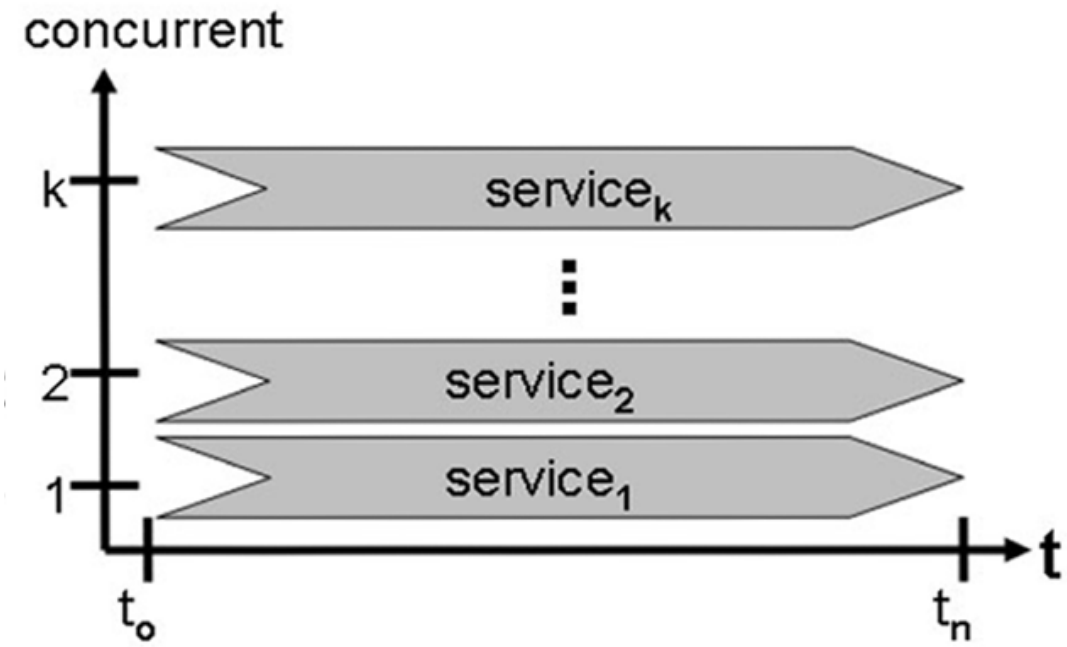


Figure 2.4: Concurrent Service composition [54]

Search for a replacement service element during service failures introduces additional delays that can stress tight execution time constraints in a successive service chain, while in a concurrent it might not have an effect on the successful task execution (i.e., when failure of close caption extraction happens in parallel with video transcoding) [54].

In order to provide system flexibility or when the quality of the end result is an essence, a set of policies can be considered to guide the ordering for certain groups of composition elements based on requested content type and/or properties specified by it's meta-data descriptions. As an example, a category of non-functional services (e.g., compression and sampling) that manipulate quality of the media content affects data volume, thus becoming a desirable option when attempting to administer transmission delays [58]. Ordering of some service functions might also be based on the effectiveness of their mutual interactions. For instance, it is less expensive to translate a summarized version rather than the whole text, making it beneficial to perform language translation and format/modality conversion as a last operation in the adaptation chain. "Object extraction" from an uncompressed image tends to provide better quality results, compared to when it is performed after a lossy compression [14]. A combination of "caching" and

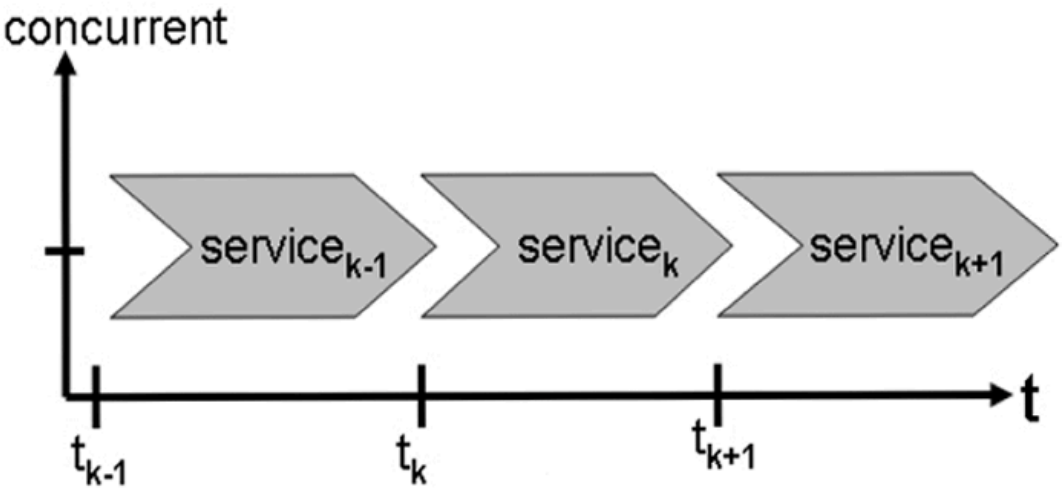


Figure 2.5: Successive Service composition [54]

"re-encoding into a streaming format" services might be given a priority or inserted into a predefined adaptation sequence in order to better interrelate service requests with network conditions when network load exceeds a set threshold. Directed acyclic graphs (DAG) are often used as solid support structures to express and explore such service dependencies in multiple chains of adaptation services. There exist many well-known algorithms that can be then applied to such graphs in order to optimize generation and evaluation of the service composition stage.

2.3.7 Adaptation Graphs

In essence, service graphs contain all available network adaptation services in a form of nodes, while directed edges represent feasible physical and/or logical ordering of such services [28], [14]. The start node represents a service which outputs a file format corresponding to the requested media file. Edges connect it to the invocation of the needed adaptation functions and the end/sink nodes represent services that, as a result, accept the desired media format. As an example, authors in [50] when considering an adaptation of an oral video presentation into a PDF document, construct such a graph by consulting a service registry. In this case the adaptation path to produce the desired format contains a total of three adaptation services: sound extraction, transmoding sound to text followed by transmoding text to image in a PDF format (Figure 2.6).

One of the methods to create adaptation graphs is to connect all output links of the

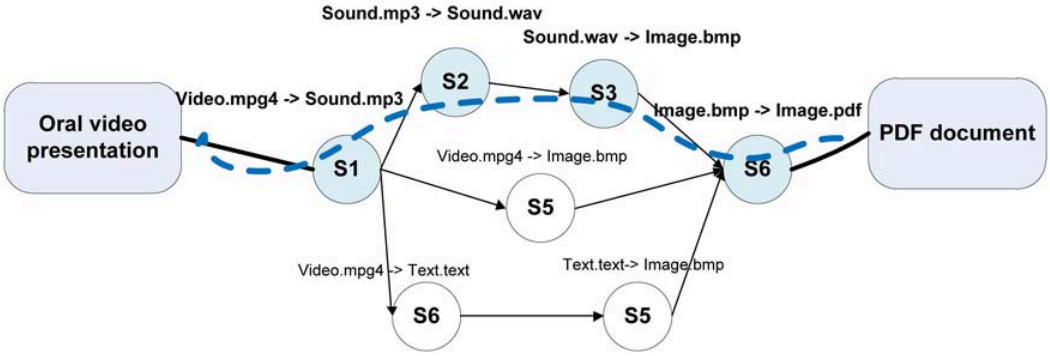


Figure 2.6: Example of an adaptation graph (redrawn from [50])

start node to all input links of all other nodes that accept that format as the input. The same procedure is repeated for all other nodes, while keeping the resulting graph acyclic by verifying the uniqueness of the structural file format along the way [28]. It is also possible to use backward chaining techniques, when services outputting the desired file format are identified first. In this case the selection process is repeated until the input of the last selected node can be matched to the configuration corresponding to the requested file. If the last candidate in the adaptation chain can not be matched to any other nodes, the selection is revoked and the search process backtracks to the previously selected element [40]. During concurrent creation of an adaptation path from start and end nodes, "forward" path selections rely on the profile obtained from the requested file, while the "backward" path creation depends on the combination of request, client and device profiles. In this case, the system is constantly checking for the existence of a common node linking two paths together, while maintaining the acyclicity of the adaptation trees [69]. In order to reduce computational time and complexity, the set of candidate services can be adjusted to consider only services that are available at the time of the adaptation request. This can be done by maintaining a queue for each of the services with elastic limits based on service capacity and traffic flow. When the limit is reached, service can no longer be considered in the adaptation candidate set [69].

Another solution that can help to reduce computational complexity is to create graphs by using the "guidance structures". Such structures focus just on the qualified services and can be stored for the later use for cases of multiple resemblant requests. One example of such structures is an ordered string of adaptation operators (i.e., a transformation prescript), that is formed before the actual file processing to meet request constraints. It

abstracts from execution details and provides input/output comparability only at a media level [13]. The adaptation graph itself is then constructed from the set of candidate adaptation services, that can perform operations specified by the prescript, at which point each connection is evaluated in terms of quality, consistency and input/output value compliance. In cases when the needed path can not be completed, unconnected candidate services are removed from the graph and/or supplementary, semantically neutral operators are inserted. Service ordering for such adaptation paths is based on the prescript, that in it's turn is based on the context profile without consideration of the adaptation service specifics and correlations. This can potentially result in omission of efficient (in terms of desired QoS) service sequences, as well as potentially lead to violations of certain operation requirements (example: compression has to occur before and not after encryption in order to comply with the system privacy standards) [41].

2.3.8 Path Selection in Adaptation Graphs

Selection of an adaptation path in a graph among multiple alternatives is based on the score vector of quality parameters associated with each node in the adaptation graph. Quality vectors can include parameters such as time, cost and service availability. The overall path value is calculated as a sum of the scores of it's meaningful vector components and can be additionally weighted to account for preferences and quality criteria defined by the user request and system objectives [13]. Opposed aspirations of the user for the best quality content and the system to persevere it's resources calls for a tradeoff, that can be viewed as a multi-constraint path problem (MCP), which is known to be NP-complete. Such situation arise in [58], where delay (sum of transmission between services + total processing time) is a quality factor for the user, while service path components are defined by the data input/output ratio change, resulting in a dynamic processing time, dependant on the data volume. In this case, authors use a heuristic method to resolve the problem by applying a hop-by-hop path construction technique, where each successive service link selection in based on a low delay and small input/output ratio.

Other techniques to identify adaptation paths include brute force methods. They are used in [41] where an "automatic path creator" utilize breadth-first search and in [14], an adaptation path is located through an exhaustive search according to the maximum quality score value of each of the available paths. More sophisticated methods include Dijkstra's shortest path algorithm [24] or as in [37], where services are selected from

a set ordered with accordance to the number of mismatched descriptions found by the comparison function with the results and then sorted according to the distance of the adaptation services from the requesting device. Due to high computational costs, such search methods are used with the assumption, that the diameter of the adaptation graph will not exceed a certain threshold.

The scalability and computational overhead challenges in the search for an acceptable chain of adaptation services increases as the number of functions offered by the proxy-servers and the variations in their performance expand. It becomes a time-consuming process in the face of the shortest path algorithm, thus calling for a further search for new design solutions.

2.3.9 Stages of an Adaptation Path Search

Logical Adaptation Path

The adaptation path search can be viewed as a two stage process; during the first stage, a logical adaptation path is composed from compatible logical adaptation operators, combined by the correspondence of their input and output parameters. The main challenge of this stage is the discovery of the needed compatible operators that, in the worst case scenario, requires an exhaustive search among all service descriptions. Time needed for the discovery of compatible service components can be reduced with the introduction of Decentralized Dynamic Hash Table (DHT) structures [34]. DHT stores and maps service components meta-data with accordance to the adaptation functions that they provide. However, constant updates with the combination of direct lookup in the DHT table can cause system message overhead, that, in it's turn, can be improved with the use of a semantic approach. In a semantic approach, service functions are grouped based on a criteria predefined during the development phase. Similarly video files before processing can be separated into categories based on the features reflecting a spatiotemporal complexity of the files, as there exist a strong correlation between video adaptation behavior and the low-level content features [79]. In order to guide the adaptation process for future requests, a representative operation preference (based on the statistical characteristic of the entire group) for a given set of parameters is summarized for each of such categories. As an example, authors in [8] group media ports within geographical subareas into hierarchies based on types of services that they offer. Services in such groups (like adaptation services or routing services) can be further classified according to their

properties. The challenge arise as dynamic service properties (such as cost, location and availability) might change over time, forcing the system to reassess the correctness of the service logical positioning in the classification group. In addition, the introduction of a supplementary classification criteria refines hierarchical structure at a high message overhead and maintenance cost and, in certain cases, decrease the effectiveness of the hole structure. Such effects are explored in [76], where authors select a limited number of useful parameters from media feature set of an MC EZBC coded video for a further classification-based prediction.

Physical Adaptation Path

In the second stage of the adaptation path search, the logical path is transformed into a physical one. This is done by mapping each logical operator to a specific proxy-server providing a corresponding functionality (Figure 2.7).

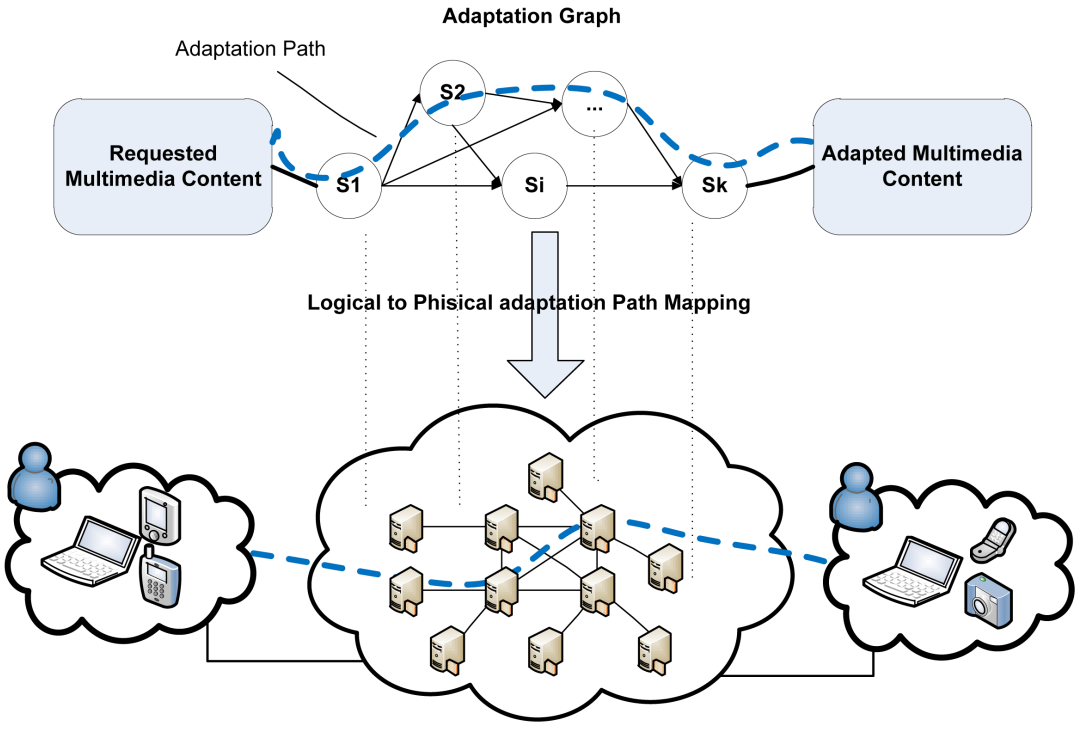


Figure 2.7: Mapping logical to a physical adaptation path

In this case, the selection of an optimal path among a set of available paths is based on a performance criteria and current system runtime parameters. As an example, in [34] authors propose a structure, where multimedia application units providing service

functions are mapped to service components in a Service Overlay Network (SON). One of the challenges is, that once an adaptation service is assigned to a network proxy and is allocated for a media stream, both network and server resources are dedicated to that stream for its duration. Such resource reservation might result in service and network overload or under-utilization. Moreover, even single service degradation may result in degradation in the quality of the delivered media content. There have been several efforts to overcome these limitations; for example, Al-Oqily et al. [8] build service overlay networks as efficient chordal rings structures that, due to it's properties, allow for an easier adaptation path identification, while simplifying service replacement process.

2.3.10 Summary and Open Issues

In order to provide the best experience in a dynamically changing environment, along with the pressure of heterogeneous nature of user requests and constantly growing numbers of available devices and services, systems using service composition for a multimedia adaptation path construction have to provide flexible solutions at each stage of the composition process.

In order to take advantage of all available service functionalities, user request for a specific multimedia content (in a form of references, QoS and device requirements) have to be mapped to a common description model (semantics and/or ontologies) used to describe services and available content. Attempts to develop a universal multimedia access methodology [72] target only the semantic content representation and fail to address the gap between content semantics and low-level parameters. On the outside, systems can explore the advantages of an execution environments such as cloud computing, that acts as a single container for applications, services and other components allowing seamless access to resources without the knowledge of their location and implementation details. On the inside, services have to allow for a contextual discovery of the most appropriate candidates without heavy costs of semantic matching by hierarchically grouping them with accordance to the types of offered services [8]. However, due to the status changes of network components and performance fluctuations reflected by dynamic service properties (such as cost, location and availability), the system might need to reassess the correctness of the service logical positioning in such classification groups.

In the next step, logical adaptation paths are assembled from a set of discovered candidate

adaptation services in a form of adaptation graphs by evaluating connections in terms of their quality, consistency and input/output value compliance. If the construction of an adaptation graph is based only on the context profile without the consideration of the implementation details and adaptation service correlations, it can lead to semantically correct, but inefficient service combinations, potentially violating operation prerequisite requirements [41], [13]. At the same time, if services involved in a successive composition are functionally dependent, then in order to process the service request within the established time frame, each successive service has to be available for the corresponding processing stage at the relevant time. This requires the development of mechanism to estimate workflow timelines to guide and optimize the usage of the system resources. An advanced user can refine a composite service plan himself by enhancing an existing service graph with descriptions of service functions needed to complete adaptation tasks and/or with specifications of the inter-service dependencies, that can be later used by the system to extract different composition patterns [34], [58]. However, an average user, when submitting requests at the application level, is more interested in the final outcome (without the need or want to understand the underlying process). This calls for a need to include the possibility of expressing preferences with the use of high-level semantics in order to create an approachable for an average user, yet precise enough method to allow adaptation system to refine and navigate the adaptation graph on the user's behalf.

It's been noted, that multimedia content demonstrates similar behavior and operation preferences based on the similarity of statistical characteristics of the content spatial and/or temporal parameters [79]. However, till now there is no clear investigations on how this information about multimedia content low-level parameters can be used to create service ordering rules and guiding structures.

2.4 Future of the Multimedia Services

2.4.1 Services

Over the years, the level of multimedia functionality abstraction progressed from modules, to objects, then components and finally to services. In essence, services comprise identifiable functions, whose principle of operation does not depend on the context or state of the other services. While combining functional descriptions and behaviors, a service hides the implementation details at the same time providing a simple interface

that allows for the creation of external connections with other services and system components. Service oriented Architectures (SoA) employ the flexibility of services and with the combination of sets of policies and frameworks, create a concept specialized in a specific domain or business task [15]. Applied to multimedia processing, SoA concepts are extended to support large real-time feeds and are capable of analyzing, searching and retrieving multimedia content from existing databases, while maintaining the coherence of media formats and profiles [32].

Web services are programmatic interfaces that provide a generic and flexible way to shift storage and computational load from acquisition and consumption devices to the network. Web services can be published, registered and discovered by service requests and are designed to support interpretable machine-to-machine interactions over the network, resulting in services that a SoA concept can use and expose via web service interfaces. A generic web-service for content analysis includes a controller, that transforms incoming requests into a functional requests that is then used by the underlying system.

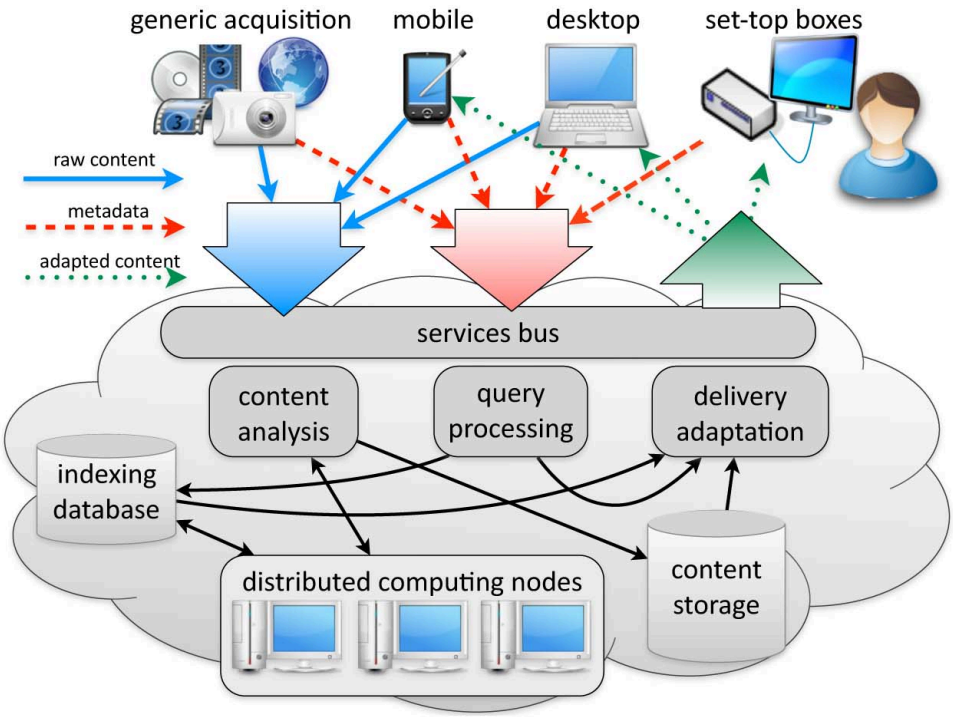


Figure 2.8: Content processing system using web services [32]

A typical web-service comprise a set of either physical or virtual computers that pro-

vide one or more content processing operations and can be added in an ad-hoc manner. Indexing databases (possibly distributed) maintain an internal set of low-level components that describe the content stored by the web service (Figure 2.8). Such web-service concepts naturally fit into cloud computing, where processing resources are assumed to be satisfied by the service providers.

2.4.2 Service Clouds

As the multimedia content evolves, management methods for hardware and software components supporting the creation, retrieval and delivery methods evolve concurrently. In the light of the rapid increase in the amounts of consumed data that happens in recent years, there exists a need for a powerful, massively scalable system solution that is accessible on demand and provides different levels of services. Cloud computing is a large-scale paradigm, where resources (i.e. computing power, storage), as well as third party software and applications are offered to customers as services in a seamless manner over the Internet via a service cloud. Such concept expects the end-users to work from anywhere, using any device, platform or network [10].

The cloud architecture can be defined as a layered structure [30]. The fabric resources (i.e. raw hardware resources such as storage and network resources) are abstracted/encapsulated (usually by virtualization) into a unified resource layer that appears to the end-user in a form of integrated resources (such as virtual computer/cluster and database system). Virtualization is widely used in clouds as a cost-effective means to run an abundance of applications, providing support to configure each individual application, while at the same time maintaining the necessary abstraction to create a unity of the underlying fabric resources. Due to it's properties, virtualization provides a possibility to run multiple applications on the same server, while performing maintenance procedures (e.g., service migration, monitoring, provisioning and service recovery) in the background without the need of service interruption [30]. Such integrated resources are then used by applications and middleware and are accessible through standard protocols (WSDL, SOAP and some advanced Web 2.0 technologies) via an abstract user interface.

Infrastructure as a service (IaaS) offers hardware, software, and storage resources (mostly at the unified resource and fabric layers) that can scale with accordance to resource needs of the requesting application. Software as a Service (SaaS) is the most

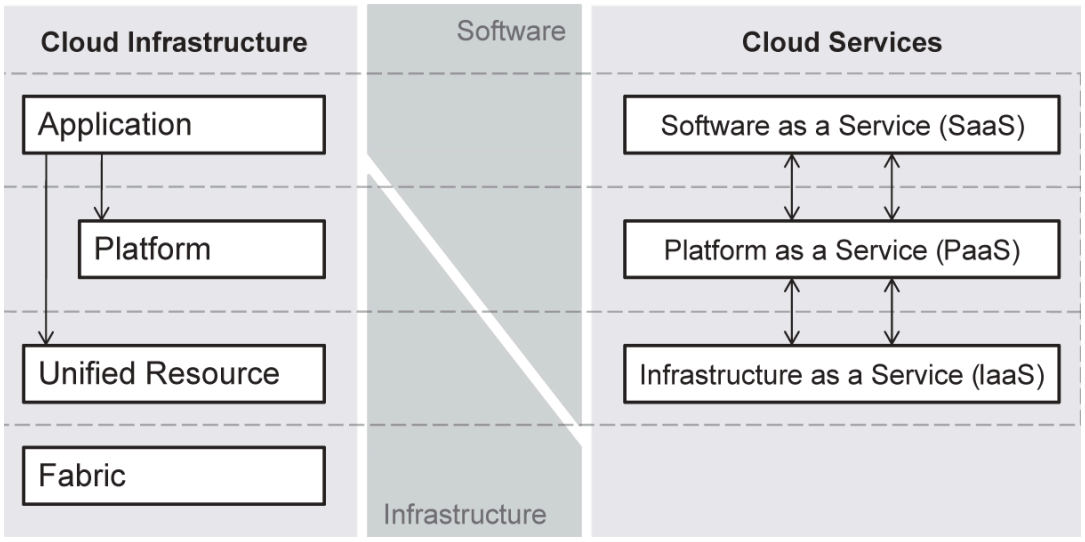


Figure 2.9: An abstract overview of a cloud infrastructure [67]

visible and at the same time most restricted layer of a Cloud Computing paradigm that offers special-purpose software that is remotely accessible by consumers through the Internet with a usage-based pricing model. Examples of such services include Google Apps such as Google Mail, Google Docs and Spreadsheets. Platform as a Service (PaaS) represents a a high-level integrated environment targeting software developers as it provides a means to build, test, and deploy custom applications. Examples are the Google App Engine, which allows applications to be run on Googles infrastructure. The PaaS acts as an abstraction layer between the infrastructure (IaaS) that virtualizes access to the available resources and software applications (SaaS) that relay on Paas standardized interfaces and development platforms [67]. Although Clouds provide services at three different levels (IaaS, PaaS, and Saas)(Figure 2.9), standards for interfaces to these different levels are still yet to be defined [30].

2.4.3 Multimedia Service Clouds

Performing editing, processing, adaptation and delivery tasks on rich media content in a general purpose computing cloud may lead to a quick service degradation due to the stringent quality of service (QoS) and quality of experience (QoE) requirements, the high resource utilization of these services and the bursty nature of media streams. Multimedia-aware clouds [83] provide the elasticity of the cloud computing resources allowing at the same time to manage a wide range of demands with relatively low initial

capital expenses, providing a solid base to perform compute intensive tasks with audio/video information [23]. However, the current state of public cloud capabilities and price structures (such as bandwidth costs), given ever increasing demand for multimedia services, requires architectural solutions that enable efficient data processing while utilizing the combination of effective adaptation algorithms and services offered by the cloud (e.g,. [26]) in order to address several challenges before achieving such a vision.

After processing a user request on the cloud, the final content has to be encoded and delivered over regular/wireless networks to the end-user. In order to provide a pleasant experience to the multimedia-aware cloud customers, a wide range of services have to be enabled to adapt the delivered media content to suit each individual user preferences, device capabilities, the nature of the running application (e.g., voice over IP versus image-based rendering and mobile gaming), as well as to address the variations in the underlying and intermediate network resource availability [8,83]. It becomes challenging, as the industrial focus on building efficient user devices has resulted in a wide spectra of memory, processing, display and battery power device capacities. What complicates the task of selection of the most appropriate set of services that will fit the individual user device, is that multimedia content standardization efforts have introduced even more diversity in the content formats (e.g., H.264 / MPEG-4 AVC coding standards [53]).

2.5 Chapter Summary

An overview of the current content adaptation and composition techniques allows to identify the stages and challenges that have to be addressed in order to create an effective content adaptation system. This chapter reviewed the notion of multimedia content adaptation. It also reviewed possible content transformations, as well as different adaptation approaches it terms of timeline (dynamic and static), relation to the user (user-centric and device centric) and location where the adaptation process takes place (client slide, server side or proxy-based). The chapter then focused on modeling techniques, solutions, as well as the challenges of the content adaptation in a proxy-based system. The chapter concludes with an overview of the multimedia services and their role in a cloud computing paradigm. The next chapter will present a novel service architecture that addresses the challenges outlined in the overview and at the same time can be seamlessly integrated into a cloud computing concept.

Chapter 3

Service Virtualization Architecture

3.1 Chapter Objectives

After reviewing in the previous chapter the existing methods of managing composite adaptation paths, it is possible to identify the objectives that should be considered at each development stage of a novel adaptation architecture. Then, the overview of such Service Architecture solution is presented, followed by detailed descriptions of each of the architectural elements.

3.2 Problem Formulation

The media adaptation/processing chain involves a client (i.e. user), a content provider and a media service cloud. The content provider hosts multimedia data servers, providing a storage environment for various types of multimedia content. A stored multimedia file M is described by a set of parameters and features $\boldsymbol{p}^c_M=(p^c_1, \cdots, p^c_n)$ such as frame rate, frame dimensions and audio frequencies. When users request a multimedia file M, they express their preferences by specifying an acceptable range of multimedia parameters, while potentially prioritizing some of them. After all needed rectifications to insure that the defined parameter ranges do not conflict with device constraints such as supported formats, screen resolution and processing power are performed, a desired description of the requested multimedia file is formed $\boldsymbol{p^u}_M=(p^u_1, \cdots, p^u_n)$. If similarity measurement between the existing content and the requested indicates that the two do not match $sim(p^c_M, p^u_M) \neq 0$, the server has to locate and then apply a set of available adaptation functions in order to deduce the existing content to the desired format.

From this sample scenario, it is possible to identify a set of challenges, that should be addressed during the development of a sophisticated multimedia processing system:
- The system should provide support mechanisms to allow users to express their preferences not only by a set of multimedia parameters, but also in a form of abstract quality requirements such as "best quality" or "fastest delivery time".
- Provide an efficient way to locate a set of adaptation functions capable of adjusting one or more of the mismatched multimedia parameters among the vast amount of available adaptation services.
- Compose discovered services in an adaptation sequence that upon execution will transform the content into a desired format.
- When constructing an adaptation sequence, a system should allow to factor in a degree of flexibility in order to adjust to the dynamic nature of network conditions and service availability.
- Proposed multimedia processing architecture should be scalable and portable.
- It should provide support for a wide range of existing multimedia file formats and features at the same time allowing an easy addition of the new formats.
- The system should provide the architectural support for the possibility to process content in a real time manner, which includes the integration of service reliability and monitoring mechanisms.

The multimedia service clouds inherit the aforementioned challenges, however they become secondary, in light of the introduced necessity for the efficient management of the large number of offered multimedia processing and adaptation services. The proposed Service virtualization architecture (SVA) envisions the solution with the introduction of the following components:
- Use of the service virtualization techniques [51] through the utilization of virtual adaptation services over media-enabled service clouds [83] to provide a consolidated view to the end-users by separating details of physical service operation and performance from the actual goals of the virtual service. This achieves seamless portability across a wide range of applications, platforms and devices, while increasing service distribution and availability, creating a serviceable and scalable solution for customized content on a pay-per-use basis.
- The introduction of a gradual multimedia parameter abstraction, provides a binding of low-level multimedia parameters to the high-level semantics, thus permitting the user

to specify his preferences at various detalization levels. User and device-centric request parameters are used to form the functional groups of candidate operations and later used in the creation of an effective adaptation sequence.
- Construct the adaptation sequence with the use of a performance-history based service selection scheme that takes into account the past performance of individual and composed media processing services.
- Hierarchical structuring of available adaptation services, makes possible the creation of clusters of pre-assembled high-demand logical chain segments, thus reducing the number of candidate services for qualified groups of requests. At the same time, prospection of the lower hierarchy layers, allows high customizability by providing a range of possible alternatives in situations of drastic changes in service availability/performance and when service request specifications can not be satisfied with pre-assembled chains.

3.3 Overview of the Proposed Architecture

It is assumed that a multimedia-service cloud comprises sets of data server clusters and proxies that offer various types of media processing/adaptation services. A *multimedia adaptation service* is defined as $S_f = (st, dy)$, where f is a multimedia processing function offered by the component. A processing function is represented by a downloadable application extension or a plug-in and is described by a set of static st and dynamic dy parameters. Static parameters st describe the services perpetual properties and include such parameters as ranges of acceptable input/output values and specifications of multimedia attributes that are directly affected by the manipulations of the offered adaptation function. The set of dynamic parameters dy describes values that can change over time and include parameters such as service location on the network, service availability status and statistical information about service performance (ratio of input/output data, average processing time per data unit). Even though the presented work does not impose any particular language for the model description; nevertheless, OWL for services (OWL-S) or web service description language (WSDL) are good candidates [62]. In order to efficiently employ such services for a wide range of multimedia adaptation tasks, a novel service virtualization architecture (SVA) is introduced in Figure 3.1, while Figure 3.2 unveils the underlying logical stages.

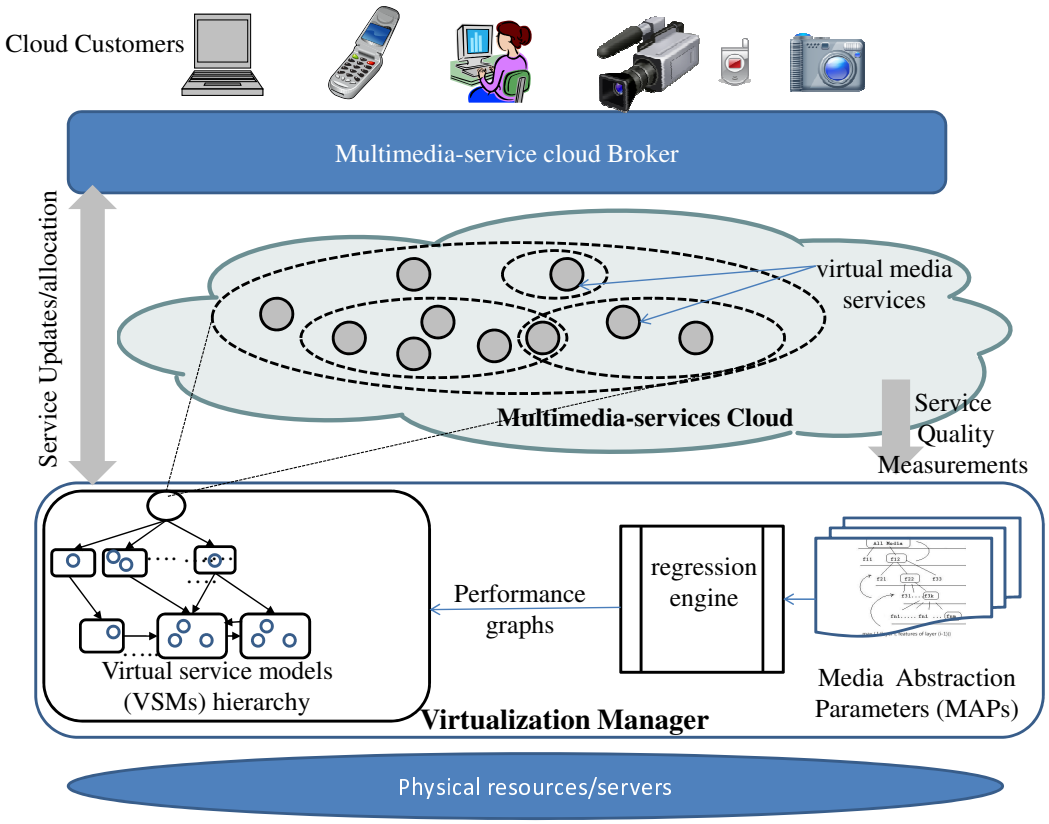


Figure 3.1: Schematic description of the proposed service virtualization architecture (SVA)

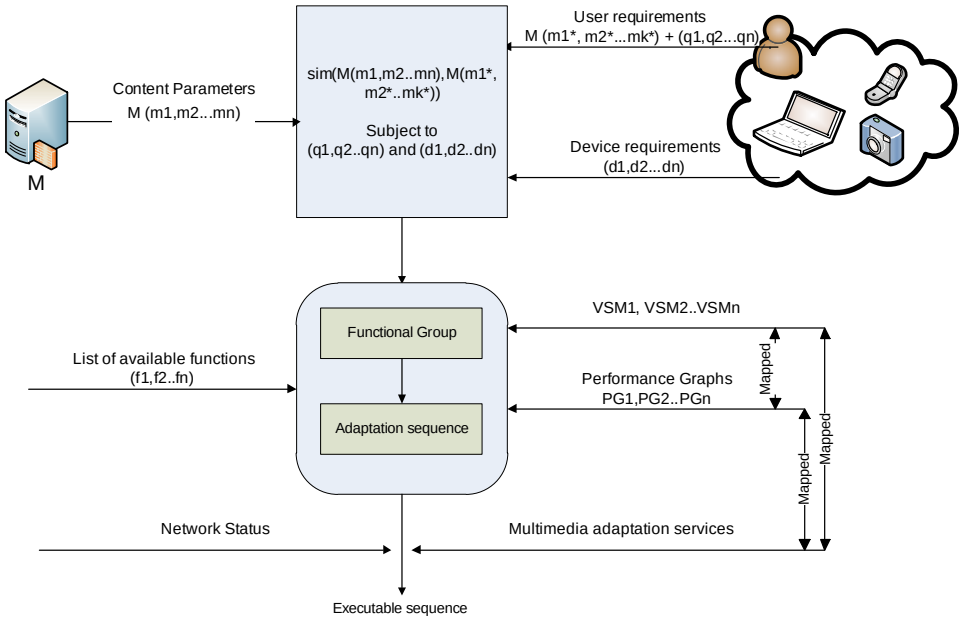


Figure 3.2: Adaptation process in SVA

As can be seen in Figure 3.1, through a web-browser the user can request, at different levels of detalization, a media processing service or a specific configuration of a media file (e.g., by specifying low-level features such as display resolution and frame rate or by using high-level abstract descriptions such as "audio quality" or "display"). Once the user selects a desired configuration of the media content, the request is submitted to the cloud service broker. The broker allocates the request to the appropriate virtual service model (VSM), capable of performing the required input/output media parameter manipulations, determined by a set of physical executable services associated with the VSM. In cases when a suitable VSM is not available, the broker forwards the request to the virtualization manager (VM).

The virtualization manager (VM) is the centralized element in the proposed architecture, as it maintains a hierarchy of virtual service models (VSMs). The VSM hierarchy is recursively built by gradually combining VSMs from lower layers with accordance to the effects that service models have on the media contents. Leaf nodes of such hierarchy are directly mapped to individual media processing services located on servers/proxies in a cloud, while VSMs at the higher levels are mapped to sequences of such physical adaptation services. The rationale behind this aggregation is that often media processing services are not requested in isolation, but rather as a sequence of processing operations

that, when performed on an input file, produce a media stream that meets a repeatedly requested set of user, device and network constraints. To facilitate the conformity of the mapping process, the VM manager also maintains a Media Abstraction Parameters (MAPs) hierarchy. The MAPs hierarchy is based on gradual media feature abstraction, that accounts for correlation between low-level multimedia content features. At higher hierarchy levels MAPs manages terms, values and artifacts used by an average user. Lower levels of the hierarchy expand such terms to provide granularity and in the end are mapped to groups of relevant low-level multimedia content features, used at a system level.

Essentially, a user request, comprising user and device constraints is defined using parameters specified by the elements in MAPs, that in it's turn, acts as a "guidance structure" in selection of a functional group. *Functional group* is a collection of unordered candidate VSMs, each of which performs manipulations on identified file parameters, progressing the state of the requested multimedia content towards the desired format. At the next step, candidate VSMs from the functional group are used to form an adaptation sequence. *Adaptation sequence* is a ordered sequence of VSMs, that are taken from the functional group and are ordered with accordance to the performance characteristics associated with the VSMs. Performance measurements, trends and adaptation specifics of each individual VSM are derived by it's available multimedia adaptation services hosted on the network servers/proxies. In essence, construction of an adaptation sequence is based on exploration of general functional dependencies, when, depending on the network conditions, each element at run time can be potentially mapped to one or several services. Since the VM communicates with service proxies in the cloud, it continuously obtains service performance measurements from each of the network services. Needed dynamic properties of the adaptation services (such as cost and availability) are collected in a form of low-level contextual information using sensors. These performance measurements are combined together to form high-level performance indications to create a consolidated service performance view for each of the individual VSMs in a form of *performance graphs* (PG). Altogether, using the MAPs hierarchy and latest performance graphs, the VM manager creates an effective adaptation sequence from a functional group of VSMs that satisfies the user request and sends the result to the broker.

Furthermore, the introduction of various layers of service virtualization allows to combine distributed adaptation services with the same logical functionality into one or more

VSM components. This way they are formed and managed without a prior knowledge of the underlying physical network and without the need to perform complex restructuring procedures of system's logical components. Details on the functionality of different components of the proposed SVA architecture are provided in the following sections.

3.4 Service Virtualization Architecture Components

3.4.1 Virtual Service Models

There has been several attempts to virtualize network components, data centers and devices (e.g., [22]), however, the presented work is the first to investigate the applicability of virtualization techniques to media services offered over service clouds. Inspired by [60], virtual service models (VSM), a core component of the Service Virtualization Architecture (SVA) are adopted as a means to create an abstract structure that will reflect all capabilities and limitation of the network, while permitting user requests to be analyzed in order to identify the efficient adaptation sequence without utilizing the actual hosted services.

A Virtual service model (VSM) is comprised of one or more currently available multimedia services, each of which offer a comparable behavioral function. Each VSM is formulated as $VSM_i = (I_i, O_i, M_i, E_i)$. The I_i and O_i sets represent the input/output parameter ranges of the i-th VSM respectively.

Parameter M_i reflects a generalized function that unifies active services consolidated by the given VSM. It formalizes the manipulations that have to be performed on I_i to obtain O_i. Parameter set E_i describes the performance behavior of services associated with the virtual model and is expressed by means of performance indicators such as mean processing time, resulting quality, etc.

Based on the complexity of internal service relationships, VSMs can be divided into VSM of a primitive and compound services (Figure 3.4). The VSM of a primitive service is comprised of a single unique processing or adaptation operation and unites basic services that evoke effects (similar to the ones described by the adaptation operation) in the parameters of the processed files. An example of such basic operations: converting a media file from one format to another, media transcoding, size reduction or color change. On the other hand, VSMs of compound services, treat it's elements not

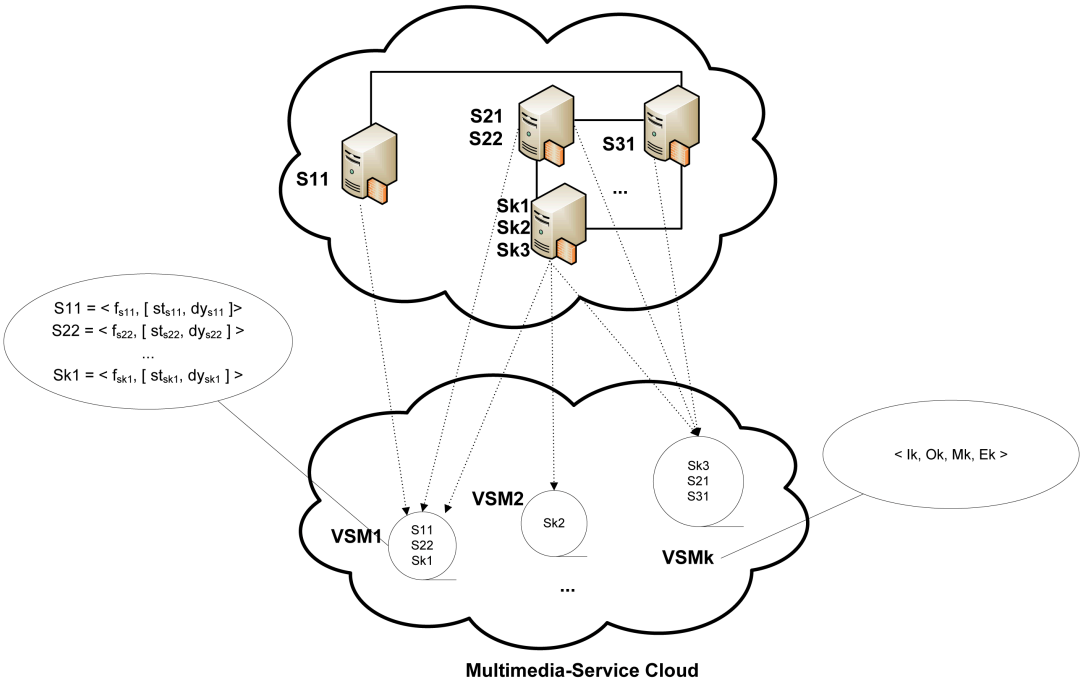


Figure 3.3: Formal relations of VSM and multimedia adaptation Services

as individual services, but rather as groups of services that might combine more than one basic processing operations. One category of such VSMs are formed by conserving popular adaption functions in a form of a functional group that, at runtime, can be quickly assembled into flexible adaptation sequences. Other category of such VSMs groups fixed adaptation chains, treating them as reusable elements for defined objectives.

The main idea of VSMs of compound services, is to form groups of nodes that are in high demand for their functionality and/or processing capabilities and encourage the initial search for adaptation services to be performed on such groups. This is the case, when virtualization is used to allow for the separation of the semantic of the task from the execution details, and does not require a prior knowledge of the network topology. Once an adaptation service is bound to a specific VSM (primitive or a compound), this association is maintained regardless of the network activity. The VSM treats it's associated adaptation services as a single entity by establishing a generalized behavior model. The generalized behavior model is accomplished by cumulatively aggregating and then averaging metrics of individual VSM members (such as ranges of accepted input/output

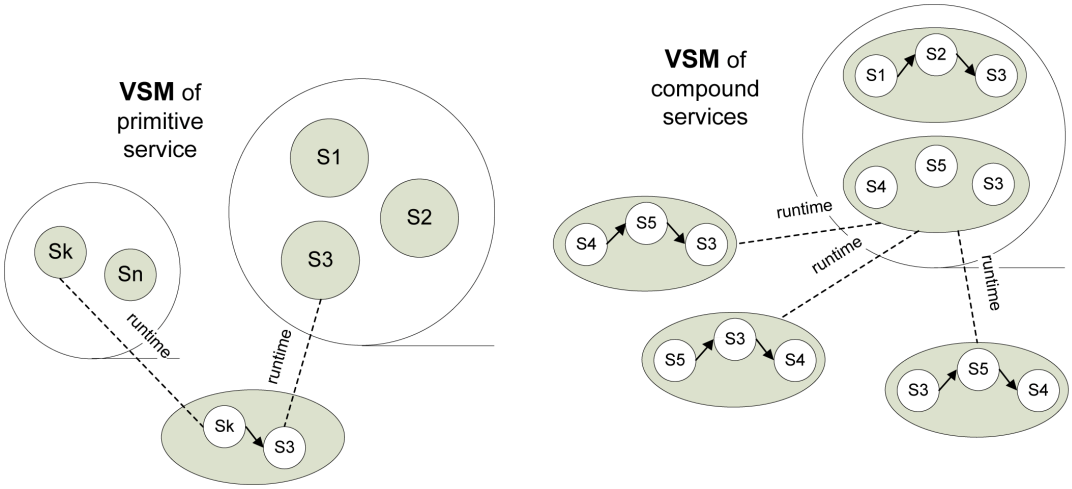


Figure 3.4: VSM Types

multimedia parameters). Similar process is repeated for statistical performance behaviors of individual elements and performed manipulation functions.

In established local networks, it might be sufficient to explicitly associate adaptation services to an appropriate VSM during system initialization by keeping corresponding logical relations in a registry of available services. However, this will not hold in a case of dynamic network infrastructures with frequently updated functions and changing topological structure (such as a mobile network) due to high maintenance and lookup costs. Searches for nodes required to satisfy the adaptation request might be performed with the use of patterns of deliberate criteria such as physical location, functionality or "distance parameters" (i.e. network delays, number of hops etc.) used to time limit the search. A similar network management system is used in [29] for a framework providing multimedia services, when the multimedia file source or the location region of the receiving device is known.

When responding to the system request, first the search for a compound VSM to form the needed adaptation sequence from it's reusable chain components or functional groups is performed. In cases when available compounds can not provide the needed functionality or precision, search is expanded into available sets of primitive VSMs that provide comparable, but distinct adaptation functions. In order to make such search even more efficient, architectural solutions such as automatization of the classification process and improvement of the query mechanism can be considered. In SVA, VSMs are organized in

a hierarchical manner with the use of Media Abstraction Parameters (MAPs) hierarchy as the support structure.

3.4.2 Media Abstraction Parameters (MAPs) Hierarchy

The Media Abstraction Parameters (MAPs) hierarchy is built to aid the Virtual Manager (VM) to efficiently bind physical services to virtual models (VSMs). After the initial network set-up, any addition of new services and data formats introduces variations into the description schemes. These variations are derived by content syntactic/semantic representations, their structural specifics and input/output descriptions. In order to preserve inter-operability and description compatibility of existing and newly introduced services, a seamless translation from one description model to the other should be performed, while services themselves should consider utilizing a uniform access interface and fundamental description schemes that will less likely to be effected by the innovations. If abstracted from the format specifications and compression schemes, a fundamental description of any specific multimedia content type is a mix of primary physical parameters and it's basic features.

Primary Physical Parameters and Features

In essence, multimedia primary features are common properties that describe the physical characteristics of the media stream or specifications of the encoded format. Examples of such basic parameters (also referred as low-level features), include but are not limited to motion magnitude, frame width, dominant color, frame rate, audio frequencies and text size. Low-level feature extraction techniques have been widely used before, primarily in semantic content analysis for classification purposes. Since low-level features are consistent through the entire content type description (and are not affected by the format specification), they can be used as a criteria to separate different subsets of data (i.e number of colors, font types, audio volume, overall motion level).

Each such low-level parameter can be extracted directly from the content metadata and file specifications or estimated from a combination of multiple frames. As an example, when utilizing MPEG-7 low-level content descriptors for color and textures, it can result in a 60-dimensional feature vector for a video stream description [77]. Computation of the mean and standard deviation of motion magnitude for video vectors adds two additional dimensions to such feature vector, allowing for an extensive description of the

imaging content of a single video frame. Reviewing a video segment with a 25 frames per second rate will now result in a 1550 feature vector and if audio features are considered in addition to the video descriptors, even higher dimensional space is reached. One way to reduce the dimension space in order to create a more compact feature vector, is to explore spacial and temporal redundancies in the visual features and adjacent video frames with the use of Principal component analysis (PCA) and by capturing significant eigenvectors. A unique variation of vector creation from multiple descriptors with the use of merging and fusion techniques are described in [66].

In cases when media content does not provide built-in descriptors to facilitate easy feature extraction, other methods can be used. In a video file for example, sequence motion descriptors or inter-frame relations can be obtained directly from video frames. The analyzing of macroblocks can provide information on the intensity and spatial distribution of local motion in the video shot. Spatial information, such as color in the form of a Color Histogram or Color Correlogram (that takes into account relative distances between pixels of all colors) can be done with the use of the keyframes [82]. No mater what low-level feature extraction method is used, the main criteria for such method is that the extracted set captures distinctive characteristics of the multimedia stream and particular properties of a specific domain.

Lower Layers of (MAPs) Hierarchy

In view of the low-level feature concept, all media types (e.g., video, audio and text) can be treated as subclasses of a single media artifact class. Hence, it forms the root of the MAPs hierarchy and accounts for all possible features of all media content types. The leaves of such hierarchy outline features necessary to describe a media streams primary physical parameters (e.g., frame rate, frame width and height, audio frequencies and text size). Lower hierarchy layers i are formed from detailed low-level parameters using feature selection technique. Feature selection is a process that selects a subset of the original feature set. A typical feature selection process starts with declaration of a search strategy to produce a candidate feature subsets. Resulting sets of features are then evaluated and gradually refined until a selected stopping criterion is satisfied. The search for the needed group of features, depending on availability, can start with an empty subset with subsequent cumulation of best parameters, from a full set and be gradually reduced in size, or it can utilize a combination of both approaches. In case of

MAPs, the extent of the influence that a basic feature has on other features is selected as an evaluation criteria [47].

Using principle component analysis, training samples of various types of multimedia content are clustered and then K dominant parameters from each of the clusters are extracted. Principal Component analysis (PCA) is a statistical technique, which allows the extraction of the most important information from the data set and reduction of it's dimensionality by keeping only the most significant elements. The main idea is to compute a set of "principal components" (i.e. linear combinations of the original variables), where the first component contains the largest possible variance (i.e. inertia) and therefore this component will explain or extract the largest part of the inertia of the data table [5]. The next component is created as the one with the next largest possible inertia while being orthogonal to the first component. The other components are computed likewise. It is a common practise to use PCA to project data to a lower dimensional subspace so that then the K-means clustering methods can be applied [25]. In case of the proposed architecture, the data set is represented by the multimedia low-level features. The emphasis is put on the features that were outlined during the psychophysical experiments as being the most influential in the human perception of recognition of the multimedia content [78]. The low-level features are rated based on how well they can represent and relate to the outlined content aspects, as well as on the range of their influence on other parameters. The influence can be derived from the definite logical feature relationships derived from the structure of a specific type of multimedia content. As an example, the total data rate is the amount information in one unit of time. For a video file that is the audio and video throughput combined together. The video throughput duration itself is effected by frame rate and the number of frames. Manipulations with video frames will have an effect on the overall file size. Due to the utilized clustering strategy, some of the parameters will form "control sets", indicating their sensitivity to adjustments of a wide range of content parameters. In cases, when the nature of the correlation between features cannot be easily traced, statistical modeling techniques (such as regression) can be applied to collect and express the behavioral patterns of different basic features.

Altogether, the most significant (dominant) features from the lower $i+1$ layer feature space are selected to represent new nodes of layer i. Let $P_i = (p_{11}, p_{12}, \cdots, p_{ij})$ represent the set of features at each layer i. Feature p_{ij} is assigned to layer i when it has the maximum covariance with other features in group $P_{(i+1)}$ in layer $i+1$ of the hierarchy.

This utilized technique is close to the feature-based classification technique in [76].

$$p_{ij} = \arg \max_{p_{i+1h} \in P_{i+1}} cov(p_{i+1h}, p_{i+1k}), \quad \forall p_{i+1h} \in P_{i+1},$$

The end result is a structure formed by the basic multimedia feature/parameters that are gradually clustered, culminating in sets of parameters that are affected by previous (lower) hierarchy levels (Figure 3.5). As an example, the "video file size" feature is affected by changes in any of the following parameters: "compression rate", "frame size", "frame count". In it's turn, out of the three mentioned parameters only "frame size" will be affected by the changes in either "width" or "height", thus automatically changing the "file size". At the same time manipulation of the "frame count" have an influence on "file duration" that also changes the file size.

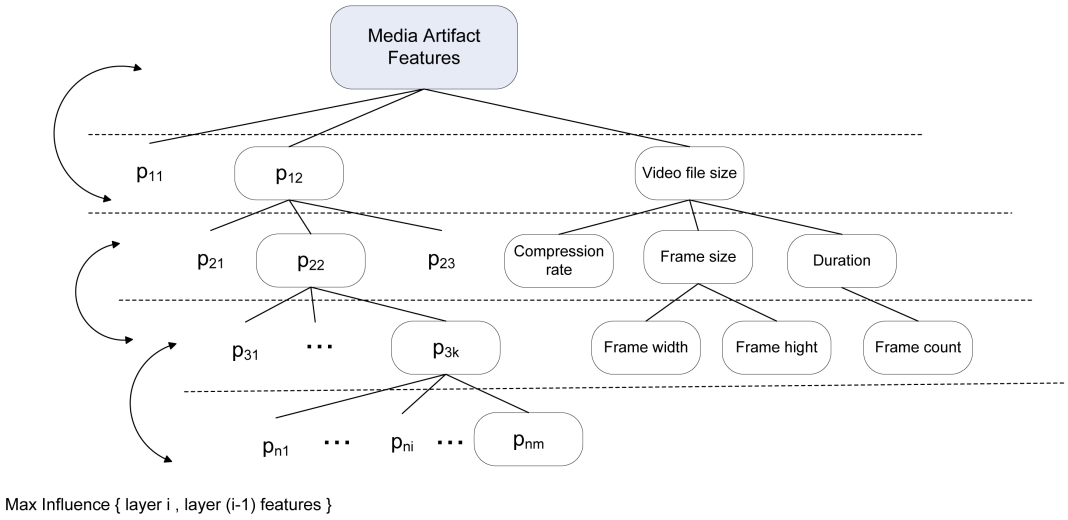


Figure 3.5: Media artifact hierarchy

Higher Layers of (MAPs) Hierarchy

An average user, when submitting a request at the application level, is more interested in the final outcome without the need to understand the underlying adaptation parameters such as file format, codecs and network conditions that have to be considered for successful file transformations. While structuring media content parameters, the higher layers of the MAPs hierarchy are designed with the intent of closing the gap between parameters used by the internal system processes and features that are used by an average user when specifying his request preferences. One or more features from the intermediate layer i

of the MAPs hierarchy are combined to produce a new abstract feature at layer $i + 1$. As an example, "frame width" and "frame height" form the "resolution" parameter, but they can be also mapped to an abstract feature "display size", which in it's turn takes the values "full", "medium screen" and "cinemascope". Abstract features at higher hierarchy levels, with some reasoning about the feature interpretation, incorporate user friendly terms, that are then easily mapped and referenced by a standard user interface. Such feature mapping is achieved by following a set of rules that are ether determined by experimental observations of statistical and logical relations in the domain, or are manually defined during the system design stage.

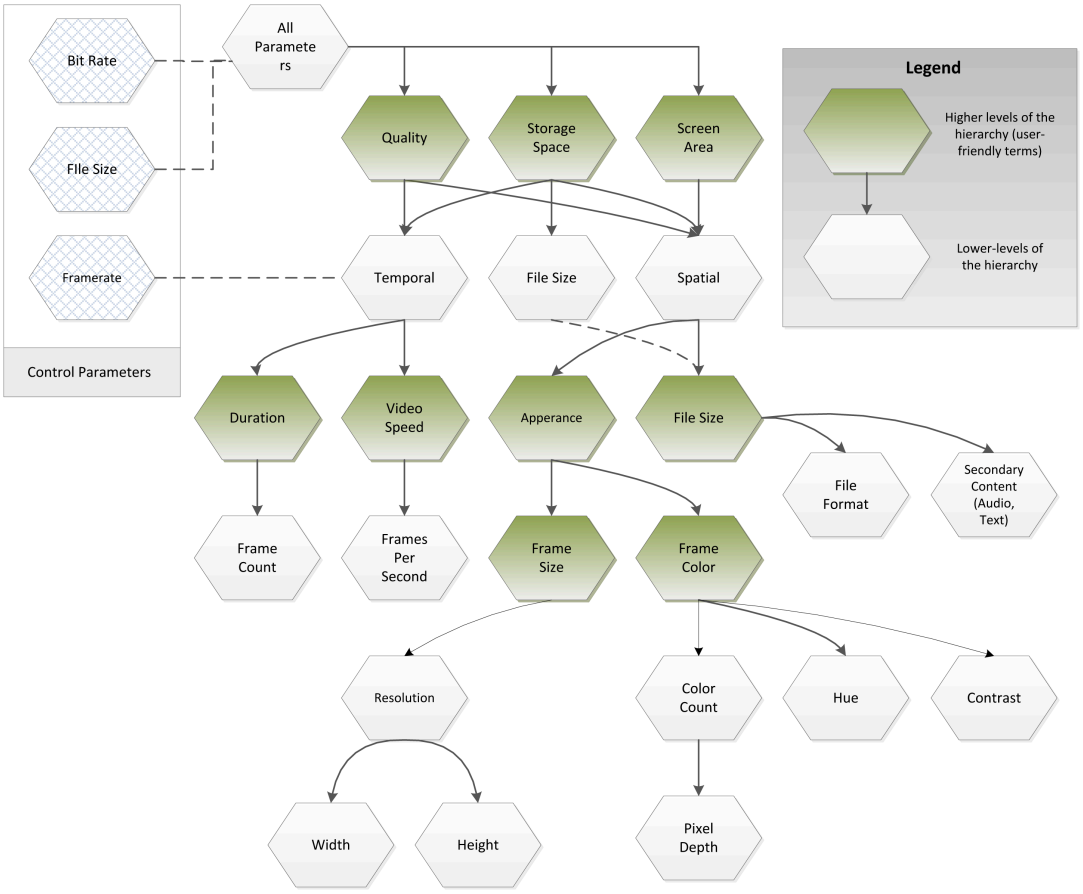


Figure 3.6: An example of the MAP's hierarchy fragment for a video content

Essentially the entire MAPs hierarchy can be logically separated into two sections (Figure 3.6). The first section provides a user-friendly interpretation for various multimedia parameter distributions and functional logic, acting at the same time as a gateway to the next section, that provides the actual systematization of the multimedia features

and parameters. Each layer in this second section hosts groups of dominant multimedia features, with dominance based on the influence that they have on the features located on the sub-layers. With this setup, Media abstraction parameters (MAPs) hierarchy acts as an interpreter between user defined terms and the low-level multimedia features, while providing a foundation for Virtualization manger (VM) to construct and sort the list of offered virtual services (VSMs) into their own hierarchy. It is initially constructed off-line but can be easily adapted as new media artifact types become available.

3.4.3 Virtual Service Models (VSM) Hierarchy

One possible way to manage a variation of numerous adaptation services is to form a layered structure. Such structure maintains all possible mappings of available adaptation services to VSM formations, matched with accordance to their corresponding input/output parameters. Clearly, this approach is not scalable as it produces high processing redundancy. Another way, is to gradually structure VSMs as needed or when responding user requests, there arises a need for a unique composed adaptation function. While this particular approach reduces the number of produced VSMs and hence simplifies their management, the drawback is that at the time of the request, the Virtual Manager (VM) will have no accumulated history of the consolidated performance of the group of adaptation services associated with the newly constructed VSM. Moreover, after continuous operation, such an approach might result in service redundancies. Instead of these two extreme ends, the proposed system relies on MAPs hierarchy for a scalable yet efficient method of organizing already existing VSMs in a hierarchical manner, while allowing for the direct addition of the new functional combinations.

Depending upon the sphere of influence of the transformation effects, parameters manipulated by the adaptation services can be separated into either spatial, temporal or "detail" categories. The direct manipulation of parameters related to any of these three categories affects the number of coded bits (i.e. content primary features). These coded bits could, on the other hand, indirectly effect other content features affiliated with the sets of parameters that are used as input requirements for the subsequent processing operations. Manipulations of the spatial resolution plane parameters are one of the most used in mobile devices, as they tend to have smaller screens and require tailoring of the content towards the correct resolution or frame fragment. While producing smaller frames, spacial adjustments, as a side-effect, also reduce bit rate, which automatically

leads to the reduction of the required bandwidth, buffer memory and energy consumption. At the same time, bit rate reduction can be controlled by adjustments of the frame rate (temporal resolution plane) and/or quantization factors such as detail resolution [19].

Static description parameters of the multimedia processing services include the specifications of multimedia attributes that are directly affected by the manipulations of the adaptation function offered by the service. Given that low-level parameter dependencies are explored by MAP's, they can be used as a reference to categorize VSMs performing corresponding parameter transformations, thus tying manipulated parameters to the related features in MAP'S hierarchy. For example, the lowest level in MAPs hierarchy contains a media frame *width* and *height* features that are linked to the VSMs that independently manipulate each of the mentioned parameters. At a higher level, MAPs hosts frame size that is represented by a parameter *resolution* formed as a combination of the *height* and frame *width* features (Fugure 3.7). In this case, the *resolution* parameter in MAPs can be aligned with a VSM of a primitive comprising services that manipulate resolution directly and/or to an alternative VSM of a compound, that represents a service chain that successively adjusts both "height" and "width" parameters.

The location of the identified parameters in the MAP's hierarchy suggests the structural importance of VSMs that manipulate the same parameters, as well as their dependencies and interactions. This property allows for the creation of the core layering of the VSM hierarchy, that is later expanded by recursively combining available VSMs into VSMs of a compound service. The expansion is imposed by the fact that often media processing services are not requested in isolation, but rather as a sequence of processing operations performed on an input file in order to produce a final media stream.

3.4.4 Adaptation Sequence

After the requested parameters (user and device-centric) are expressed in a form of features from MAPs, they are used to extract the corresponding VSMs from the VSM hierarchy. Extracted elements form the core of the *functional group*, containing functions that can perform manipulations on the parameters identified by the mismatch between requests and specifications of the requested content. Depending on the context, the core elements of the functional group can be further refined. As an example, elements of the *functional group* can either be replaced or expanded by an already existing VSM of a compound service that comprises a functional chain or a functional group that duplicates

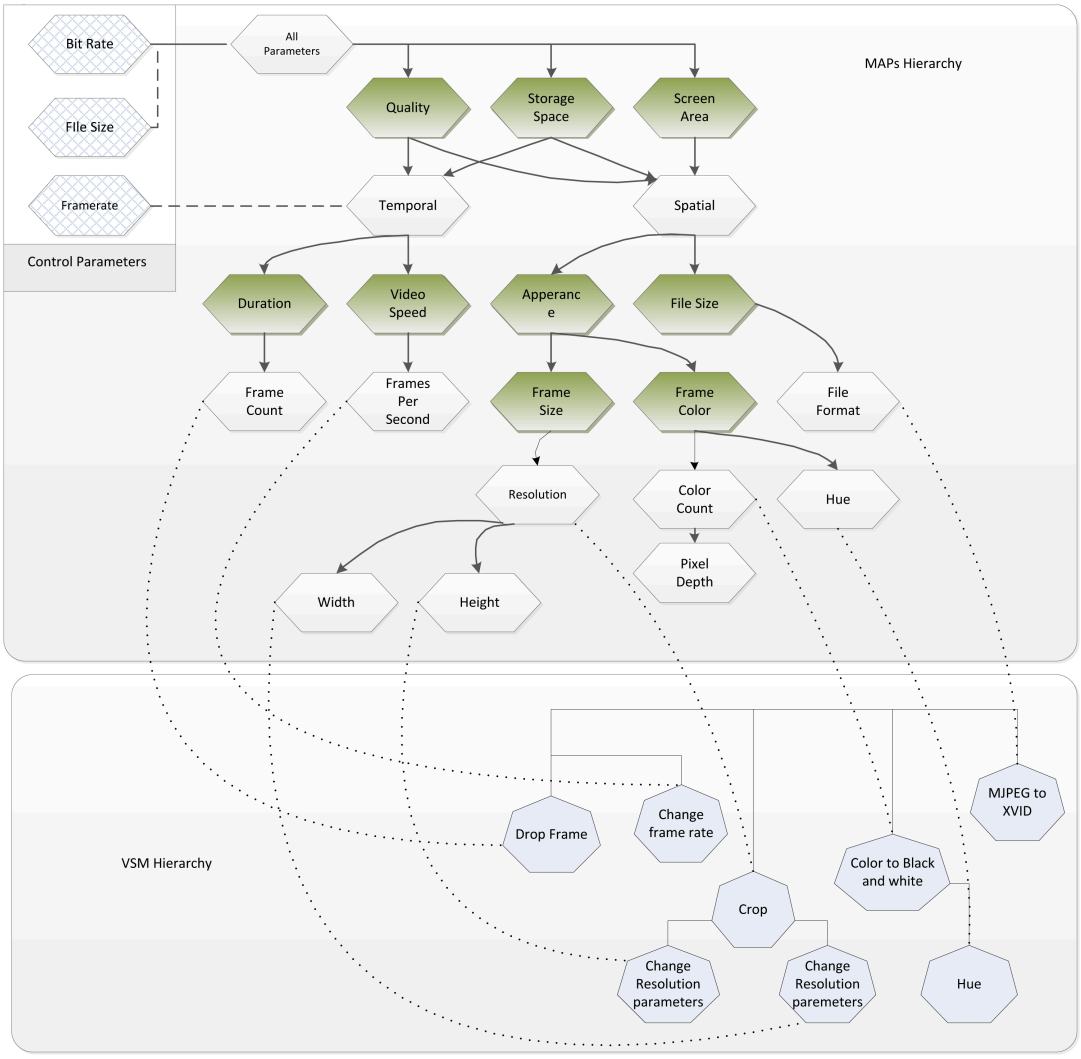


Figure 3.7: Example of mapping MAP's to VSMs

the needed elements. In cases when a core element of a functional group does not support the needed range of parameters or it is not available at the time of the request, it is replaced by a set of alternative VSMs from the lower hierarchy levels, which coordinated efforts can perform the needed manipulation.

When a functional group is refined to include the most relative VSM services, with the use of performance graphs an adaptation sequence is formed. Performance graphs provide a formal description of trends and adaptation specifics of the multimedia adaptation services at various detalization levels. Through continuous updates of performance

measurements collected from the physical services, performance graphs dynamically reflect the expected performance fluctuations for the VSM (the one that comprises the corresponding physical services). Performance graphs are used as an evaluation criteria in shortest path algorithms [20], [24] to locate an optimal sequence of functions realized by VSMs. At runtime such adaptation sequence acts as a guidance structure for the execution order of the adaptation services that should be applied to the requested content in order to produce the desired result. Each physical service is executed with accordance to the place of it's managing VSM in the adaptation chain. If service degradation is reported, it is automatically reflected in the corresponding performance graph. In this case, the virtual manager (VM) uses the updated performance graph to reorganize and/or reorder virtual services in the adaptation path. The same graph can be used at execution time to help map the elements of the adaptation path to an alternative physical service of a needed performance. The next chapter will introduce one of the possible methods of creation of such performance graphs.

3.5 Chapter Summary

At the beginning of this chapter, the content adaptation problem was reviewed to identify the challenges and objectives of the proposed architecture. After an overview of the developed system, its subcomponents were discussed in details. It started with the review of the architectural elements, the virtual service model (VSM), that is adopted as a means to create an abstract overlay structure to organize and interact with multimedia adaptation servers in a cloud. Media Abstraction Parameters (MAPs) hierarchy section discussed the features/parameters that describe multimedia content and how they can be structured and used in conjunction with the VSMs. At the end of the chapter, methods to form functional groups with components extracted from the VSM hierarchy were introduced, followed by the discussion on how they are used to obtain an executable adaptation sequence. The following chapter will propose a performance-history based adaptation service selection scheme that utilize a formal description of trends and adaptation specifics of the multimedia adaptation services in a form of performance graphs.

Chapter 4

Performance Graphs

4.1 Chapter Objectives

This chapter will review the challenges that arise when performing media adaptation service ordering. It will then introduce a statistical model that is used to first gather and then to utilize performance measurements to create a performance-history based selection mechanism to produce an efficient service ordering for the most current state of a dynamically changing environment of multimedia clouds.

4.2 Creation of an Adaptation Sequence

After the main components of the multimedia content request are identified (user preferences and device requirements), with the use of MAPs they are interpreted and mapped, first to the corresponding multimedia features and then to the appropriate virtual service models (VSMs) that are then combined into a *functional group*. In essence a *functional group* is an unordered set of VSM's, each of which performs manipulations on identified file parameters. Since the role of each element of the functional group is to advance the state of the requested multimedia content towards the desired format, the desired content adaptation is achieved by a collective usage of all elements of the group, however service ordering that will lead to the shortest execution time or best obtained quality is not always apparent.

As seen in Figure 4.1, a set of the same four adaptation services combined in three different ways generate output files that have the same format and duration, but differ

in the file size, as well as in the time that it took to produce them.

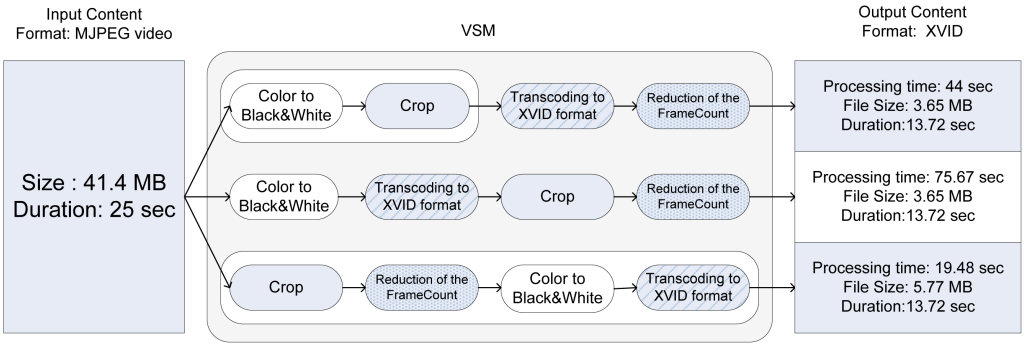


Figure 4.1: Example of different adaptation paths formed from a same functional group

It is rather intuitive, that on average, the execution of more complex and profound manipulations (such as transcoding) at the end of the execution chain provides faster results compared to when they are performed on an unaltered media file (assuming that each adaptation step reduces the size or quality of the original content). The effective ordering of the services of a lesser processing complexity is still not very clear. When tested with multiple different multimedia file formats and sizes, it is revealed that optimal service ordering tends to hold within the group of multimedia files united by the range of fundamental parameters such as file size or motion complexity.

As seen in Figure 4.2, crop operation is the first choice for the fastest processing for MJPG files between 2 and 60 MB, while the discoloration service would be preferred for file sizes between 60 and 90 MB. The same operations change their most effective ranges when performed on a different file format (Figure 4.3).

Our objective is to automate the service ordering process so it would lead to some desirable properties, such as a faster adaptation time or better quality of the final content. In order to achieve this goal, the system has to utilize methods that allow to accurately convey and predict behavioral patterns of the adaptation services.

In an idealistic model (Figure 4.4) performance measurements are first acquired from the underlying adaptation services and then combined in order to create a performance prediction model. An idealistic performance model creates a precise prediction of the effects (i.e., time and end-result of the adaptation) that the modeled services have on the multimedia content. With the use of such service performance effects, it becomes possible to predict and compare the outcomes that different service orderings will have on the requested content, thus allowing for a quick and precise estimation of the most

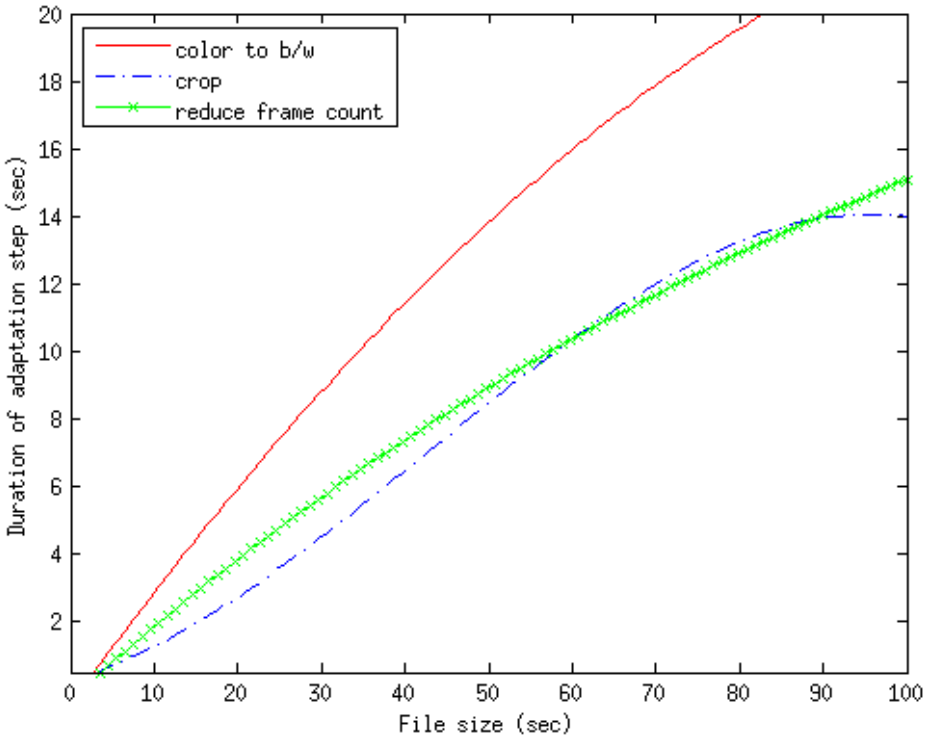


Figure 4.2: Example of operation cooperativeness for MJPG file format

effective execution sequence for a particular multimedia file. However, in real life this idealistic model will not hold, as there exist multiple factors affecting the precision of the results delivered by the prediction model. One such factor is the implementation of the performance parameters collection and delivery mechanisms.

In order keep up with the availability and performance changes of the services offered in a multimedia-aware cloud, the virtual manager (VM) (introduced in Chapter 3), continuously communicates with each of the underlying physical servers/proxies. The performance measurement collection frequency is based on the proxy's average workload, as well as on variations of the workload patterns. If availability patterns of servers/proxies can be separated into distinct periods (i.e., based on the time of the day or day of the week), service message frequency can be reduced by caching relevant performance measurements and extracting them at the appropriate time. In this case, a full update routine will have to be performed only in the case of a major change in a service behavior. In order to speed up the process of information collection and consolidation,

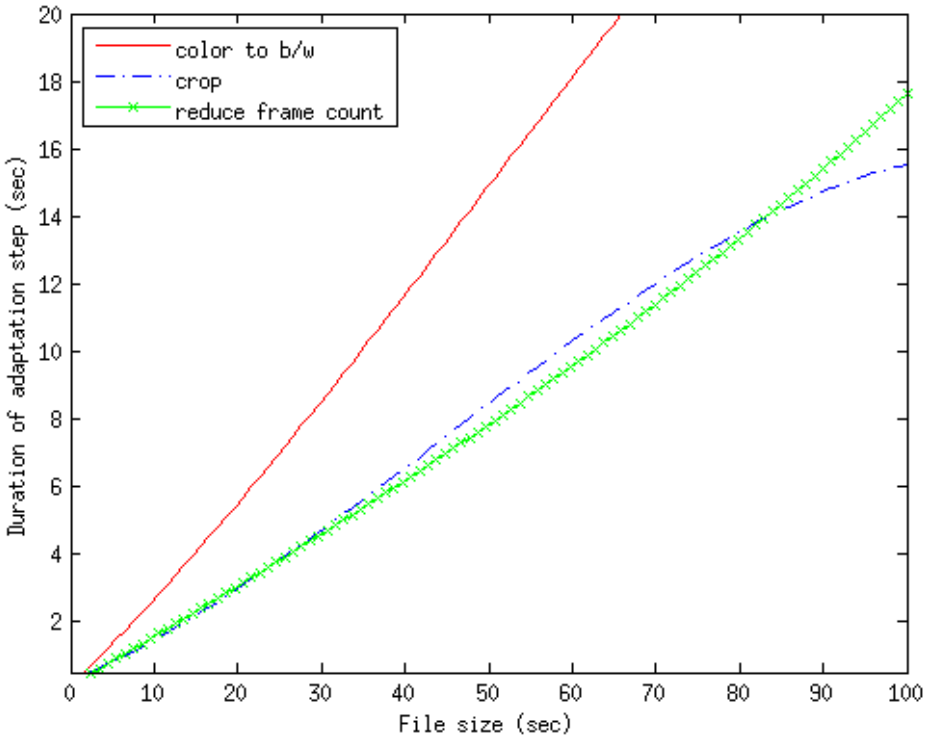


Figure 4.3: Example of operation cooperativeness for MPEG2 file format

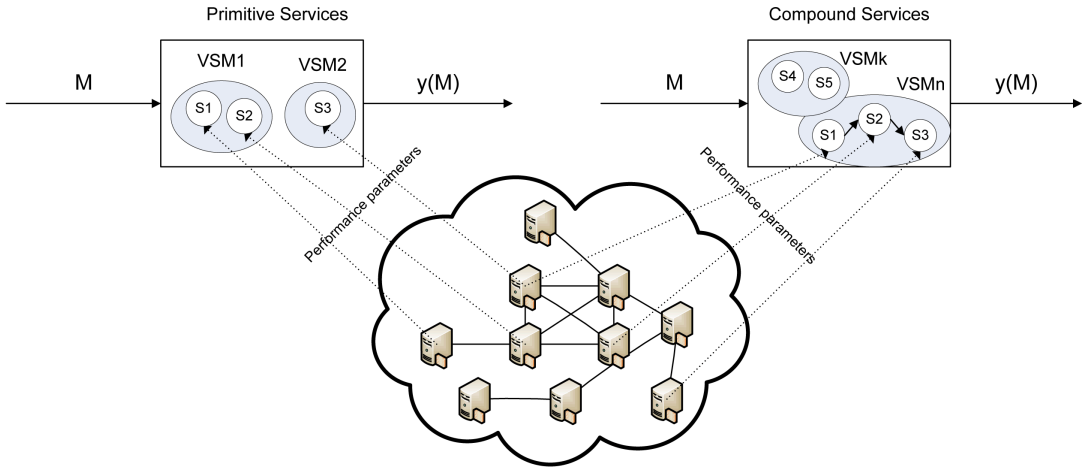


Figure 4.4: Idealistic prediction model

servers can also be configured to perform computation and data collection in a parallel and/or asynchronous manner. The accuracy of the collected measurements depends on the frequency of the service message exchanges. More frequent messages reflect a more accurate status at a higher message overhead cost. The other factor that effects the precision of the prediction results is the realization of the prediction model itself.

4.3 Statistical Models

Statistical methods provide efficient means to extract existing trends from collected data. These trends can be further used to construct mathematical models to predict the overall system behavior [6]. The selection of the specific mathematical model depends on the input variables, system goals, data collection methods and required accuracy. The white-box modeling method, for example, utilizes an expert-based knowledge of the underlying structure in order to define parameter relations. As shown in Figure 4.5, a black-box statistical approach, on the other hand, requires less prior knowledge about the underlaying structure, as it relies purely on the observations to discover parameters that effectively predict the needed measurements [18].

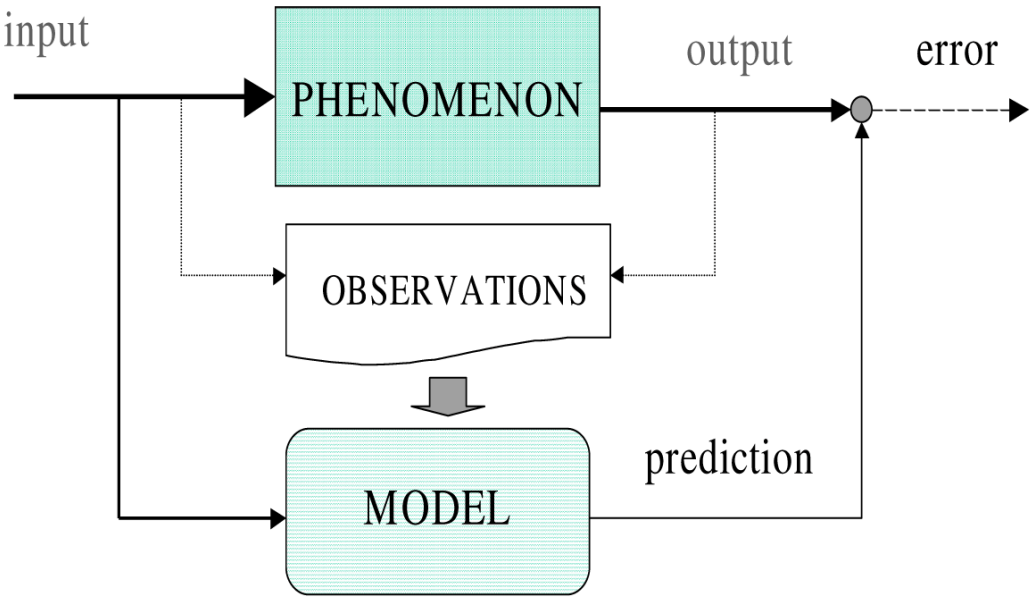


Figure 4.5: Data-driven modeling of an input/output phenomenon [18]

The main goal of a statistical model is to obtain a generalized estimation model,

calibrated by the basis of a finite set of observations. Information collected during observations is used to determine the mean of the observed parameter changes by using predictors or explanatory variables as well as coefficients, which control the behavior of the model [59]. While the problem of coefficient estimation can be reduced to fitting the prediction function close to the observed data, the function fitting itself introduces an error component that is derived from the probability distribution and estimation of each unnecessary parameter. This causes the increase in the variance of the prediction error. The stability of the prediction model can be evaluated against the prediction results of new observations. Once fed a value of the input parameters, a stable prediction model should return an accurate prediction of the dependent parameters even when processing previously unobserved values.

Regression methods were selected as a mechanism to form performance graphs for the media adaptation services as they represent a group of statistical models that allow to interpret the observations using a reasonable number of parameters [63]. It uses the relationships between the variable to be predicted (such as the time that it takes to process a file) and the parameters that explain its variation (such as the input file size or content complexity). Regression allows for the flexibility in describing such variations. In case of multiple parameters, they can be combined together in order to create one general predictor. In case of substantial influence of multiple parameters on the variable of interest, they can be weighted by additional coefficients to account for their correlation. Transformations such as the natural logarithm or reciprocal transformations can also be applied to the both types of variables (dependent and independent) in order to even further simplify the prediction model.

One of the advantages of the statistical methods is that once the prediction technique has been established, the corresponding models are fully reproducible and can be automated. However, with the increasing accuracy of the prediction results comes the costs of data acquisition, processing time and implementation difficulty.

4.4 Performance Parameters

The first step in the creation of an estimation model of the behavior of an adaptation service, is to accumulate a set of needed observations. The Service Virtualization Architecture (SVA) as part of it's maintenance routine tasks collects the performance

measurements for each of the adaptation services. That includes collection of statistics and/or aggregation of the collected data.

The SVA maintains statistics of it's services with the following performance parameters:
- Measurements that convey the performance specifics of an adaptation service. Such as the measurement of the intensity of the service manipulations. It is expressed as a range of changes that the service processing creates in multimedia content parameters. The emphasises is made on parameters that can be mapped to the corresponding components in the Media Abstraction Parameters (MAPs) hierarchy. As an example, the intensity of the manipulation of the service that provides the approximation of the image fragment can be measured as a reduction percentage of the original "frame size". In this case the "frame size" corresponds to a parameter in MAPs.
- Measurements that describe the performance variations of the server/proxy hosting the adaptation service. As an example, the average processing time of the file, changes in the available bandwidth, memory load and number of CPU cycles required required to process a media file.

In general, performance specifics of an adaptation service can be determined using operation descriptions provided by the service creator/publisher, as they convey a set of the essential parameters targeted by the operation. If such descriptions do not provide all of the necessary details, service functionality can be determined by observing the intensity of it's manipulations. That is by performing a comparative analysis on the processed file input and output parameters. At the same time, in order to reduce the extent of data collection, existing specifics of the multimedia types and file format properties are considered. As an example, it is possible to trace the effects that MPEG video encoding parameters (i.e., frame scale, Q-scale) have on the CPU and energy resources of a transcoding server [64]. Another effective linear model in it's prediction of the transcoding time of independent segments of an MPEG stream uses the notion of "group of pictures" (GOP) [35]. The GOPs due to the variation of the transcoding time for a given group size are partitioned into regions, that are then defined by the mean value of the region.

Observations that describe the server/proxy performance variations are collected as a part of the maintenance routine, allowing to promptly detect and correct prediction er-

rors of the existing model (the minimum mean square error (MMSE) is often used as one of the prediction accuracy criterions). In which case, the statistical model is updated to accommodate the current performance changes. If stored in the form of a table, performance measurements can quickly expand causing performance degradation. To overcome this limitation, the average performance data is maintained in a compact form of performance graphs.

4.5 Forming Performance Graphs

Performance graphs provide a formal description of performance specifics of the adaptation services as well as performance trends and variations of the server/proxy hosting the adaptation service. Through continuous updates, performance measurements are collected from the physical services modeled by the VSM and through regression are then transformed into a compact form. After the *functional group* of unordered VSM's (selected with accordance to the request preferences and content parameters) is formed, performance graphs are used as an evaluation criteria to locate an optimal sequence of functions realized by the VSMs. At runtime, using the performance graphs of individual adaptation servers/proxies and defined adaptation sequence as a guideline, the physical services are mapped to the corresponding functions.

In order to support an efficient selection process, Virtual service models (VSMs) are organized in a hierarchical structure. The VSM comprise one or more currently available physical executable multimedia services capable of performing the required input/output media parameter manipulations. For each service j that is modeled by a VSM_{ij}, on a layer i of the VSM hierarchy and that during it's manipulations on a media content M affects n_i features (that can be mapped to corresponding features in MAPs), we build a regression model. Regression is calculated for each performance measurement $y(M)$ (e.g., mean processing time and change in media quality) using the following form:

$$\hat{y}(\delta P(M)) = \sum_{k=1}^{n_i} \alpha_{ik} \delta p_{ik}(M) + \alpha_{i0} \tag{4.1}$$

Here $\hat{y}(\delta P(M))$ is an estimate for a performance measurement y (i.e., processing time), that is resulted from changes of the parameter p_{ik} (e.g., frame size) by δp_{ik}, where δp_{ik} itself is calculated as the difference between the parameter value of the original file and the p_{ik} specified by the the user request. The coefficient vector $\alpha = (\alpha_{i0}, \cdots \alpha_{in_i})$

is calculated using ℓ media files ($y(M_h)$, $\delta p_{i1}(M_h), \ldots, \delta p_{in_i}(M_h)$), where $h = 1, \cdots, \ell$. The corresponding measurements were collected after processing each of the ℓ media files M_h by services modeled by VSM_{ij}. More precisely, the coefficients for the performance model are calculated such as they minimize the mean squared error, i.e.,

$$\boldsymbol{\alpha}^* = arg\ \min_{\alpha} ||\delta \mathbf{P}\boldsymbol{\alpha} - Y|| \tag{4.2}$$

where

$$\delta \mathbf{P} = \begin{bmatrix} \delta p_{i1}(M_1) & \delta p_{i2}(M_1) & . & . & . & \delta p_{in_i}(M_1) \\ \delta p_{i1}(M_2) & \delta p_{i2}(M_2) & . & . & . & \delta p_{in_i}(M_2) \\ . & . & & & & . \\ \delta p_{i1}(M_\ell) & \delta p_{i2}(M_\ell) & . & . & . & \delta p_{in_i}(M_\ell) \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} y(M_1) \\ y(M_2) \\ \\ y(M_\ell) \end{bmatrix}$$

The higher layers of the VSM hierarchy, gradually combine the virtual models of primitive services into more complex VSMs of a compound service. When a VSM comprise multiple services that are behaviorally close in terms of influenced parameters (i.e they can be mapped to the same feature f_{ij} in MAPs), in order to represent it's behavior at a higher, less detailed perspective, a prediction function is fitted to the aggregated observations for all services that are associated with the VSM. In this case performance parameters, in a form of regression coefficients $\boldsymbol{\alpha}^*$, from "local" nodes (i.e., nodes forming the VSM) are transmitted to the "group leader" (the selection of the leader can be realized through any appropriate leader election technique [75]), where they are used to calculate new coefficients $\boldsymbol{\beta}^* = (\beta_{i0} ... \beta_{im})$ for a group level regression model (Figure 4.6). Coefficients for a group level performance model $\hat{f}_{group}$, are calculated in a manner that is similar to the one used by the individual services. In this case m is the number of services forming the VSM and $A^* = (\boldsymbol{\alpha}^*_{\mathbf{1}}, \cdots, \boldsymbol{\alpha}^*_{\boldsymbol{m}})$ represents the coefficients propagated from m "local" nodes.

$$\hat{f}_{group}(A^*) = \sum_{k=1}^{m} \beta_{ik} \hat{y}(\boldsymbol{\alpha}^*{}_k) + \beta_{i0} \tag{4.3}$$

At a higher i layer of the VSM hierarchy, each service is comprised of a sequence of two or more VSMs from the layer $i + 1$ to form a highly-demanded adaption function chain. In this case regression coefficients $\boldsymbol{\alpha}^*$ are calculated for the the whole group. An example of such sequence creation is a VSM of cropping services, based on services that can adjust just width or height parameters. Figure 4.7 shows the performance graph of two services that can adjust independently width and height parameters, as well as the service that simultaneously manipulates hight and width (representing the goal for the combined VSM of cropping service).

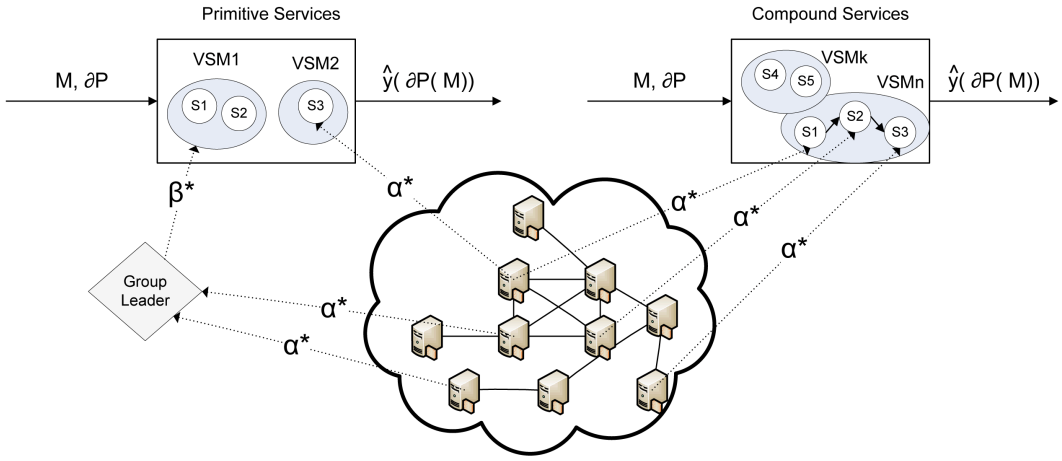


Figure 4.6: Realistic prediction scenario

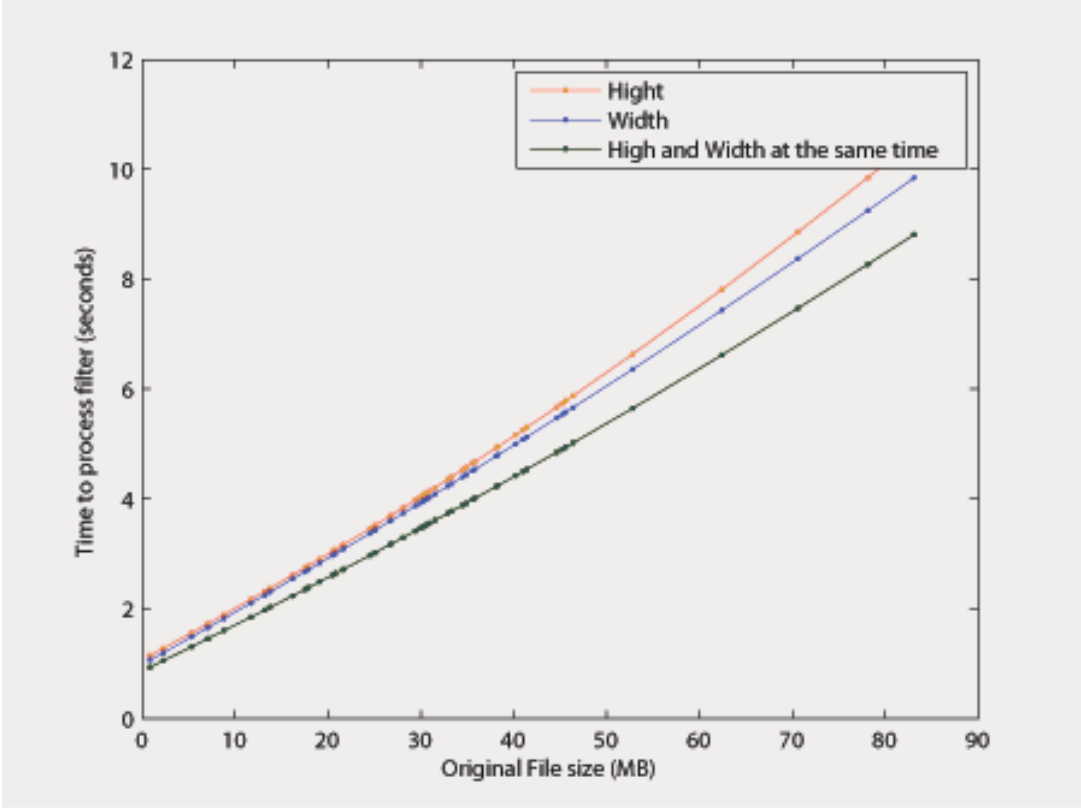


Figure 4.7: Processing time of individual and grouped crop services

Even though the end result is the same (i.e cropped picture), performance parameters vary with accordance to the order in which the sub-functions are processed. In Figure 4.8 the processing time of service that adjusts *Hight* and *Width* at the same time is compared to the processing time of chains of individual services. One chain first adjusts *Width* and then *High*, while the other chain does it in a reverse order (i.e. *Hight* and then *Width*).

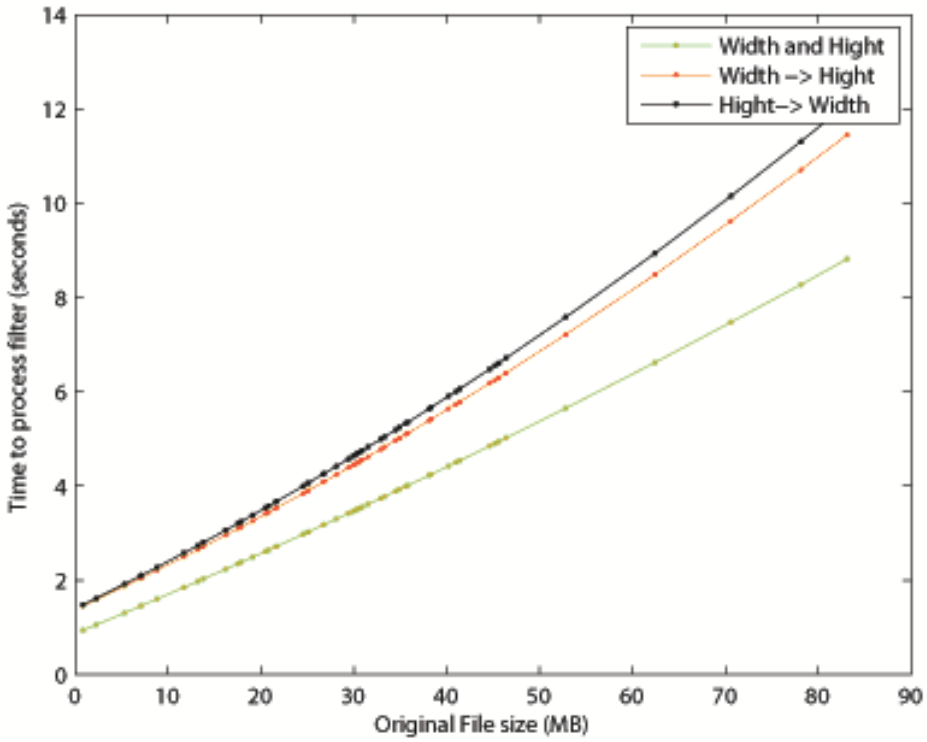


Figure 4.8: Processing time of individual vs processing time of chains of cropping services

Creation of the generalized function that account for performance variations of such processing sequences provides certain challenges due to the following reasons: it is relatively easy to evaluate a chain of two entities by assigning weight coefficients (as an example, weighing more services causing a significant file size reduction), which will automatically bias the decision towards the most appropriate sequencing. However, longer chains (involving three or more services) pose a challenge, as the estimate for the first service in the chain depends only on the incoming request, while the estimate for the subsequent processing actions rely on the preceding results. More generally, the estimate

of n-th member will depend on our estimates of (n-1) member, that in it's turn is affected by (n-2) processed results and so on. Thus, when creating a larger chain, this cumulative dependence effect has to be taken into account, resulting in a sharp increase in the estimation complexity. Therefore, to create a more accurate performance estimate for a processing chain, a more holistic approach has to be considered, rather than estimating each result step by step. Essentially, the combined function from the user point of view, is a black box (instead of a chain of processing services) producing the desired outcome. As an example, an automated picture visual correction filter and a combination of image sharpening and contrast adjustments, produce a visually close result. Using the black-box approach allows the creation of a generalized function on the higher layers of the hierarchy, thus deriving an estimator for it's adaptation functions (i.e. estimating the outcome of a combined function).

In order to provide a better system flexibility, a set of policies can be also considered to guide the ordering process for certain groups of combined functions or in cases when the quality importance of the end result is not an essence. As an example, when network load exceeds a set threshold, a combination of "caching and re-encoding to a streaming format" chain might be given a priority in a predicted sequence. Another example is a semantically confined ordering, like using encryption or performing language translation and format/modality conversion as a last operation, allowing for the production of a better results (in terms of selected QoS like time or security) [14]. Reasoning being that it is less expensive to translate a summarized and not the whole text, as well as the object extraction from an uncompressed image will produce a better quality result, as when it is performed after a lossy compression.

4.6 Chapter Summary

After the discussion of the influence of service ordering on content processing, this chapter reviewed performance parameters and statistical models that can be used to automate the service ordering. Performance parameters were reviewed in terms of their type and collection methods. Regression was selected as a statistical methods allowing to comprise such performance characteristics into a compact form of performance graphs for the effective processing of the adaptation requests. The next chapter will use such performance graphs in the implementation of the proposed architectural solutions.

Chapter 5

Implementation and Results

5.1 Experimental Setup

In order to evaluate the performance of the proposed architecture, a simulation model was constructed within a standard Unix environment. A simulated environment represents a single network domain, hosting adaptation services that can process video files. Each video is processed in a non-streaming manner, when the file is processed completely before becoming available for the manipulations of the next service. Services are confined to the following four basic functions: frame count reduction, frame resize, color/hue adjustments and file format conversion. Each of these basic services were combined from filters available through Mencoder, a free command line filtering/decoding tool [1]. Several video file formats where processed through a selected set of services. An MPEG format [17] commonly used for video streaming on the Internet was selected to represent the current trend of devices that use block-based video codecs with motion compensation and discrete cosine transfer for video compression. MJPEG [2] was selected as an uncommon format to simulate a scenario of exploring the use and integration of the new services into the system.

A training set of files of a selected format, ranging in sizes from 5 to 100Mb is used to obtain performance measurements to bootstrap the system and to form the initial performance graphs for each layer of the virtualized adaptation services hierarchy. Using the obtained results, a representation of the Multimedia Artifacts Hierarchy (MAPs) was created. In this implementation, all parameters of the MAPs "higher hierarchy levels" are divided into the three major categories (Quality, Storage Space and Screen Area) for

the user to select his adaptation preferences. Once the user selects the category, he will be provided with the granular options, for the case if he wishes to refine the adaptation details for a selected category. As an example, the user requests a video file for adaptation and specifies the "Frame rate" as the quality criteria. In the "Storage" category, the user is given the option to emphasize a criteria which is most important to him (video speed, duration or file size), when creating a more compact form of the content. After the user submits the query, his selection is analyzed and a sub-tree of the corresponding multimedia features is calculated. The sub-tree contains all available multimedia features that affect the selected criteria. To extract the sub-tree, the pre-ordered tree traversal technique is used. This technique in usually applied to store the hierarchical data in the Databases [73]. As an example, the selection of Appearance will produce two sub-trees that contain six features, while Frame color will capture only three (Figure 5.1).

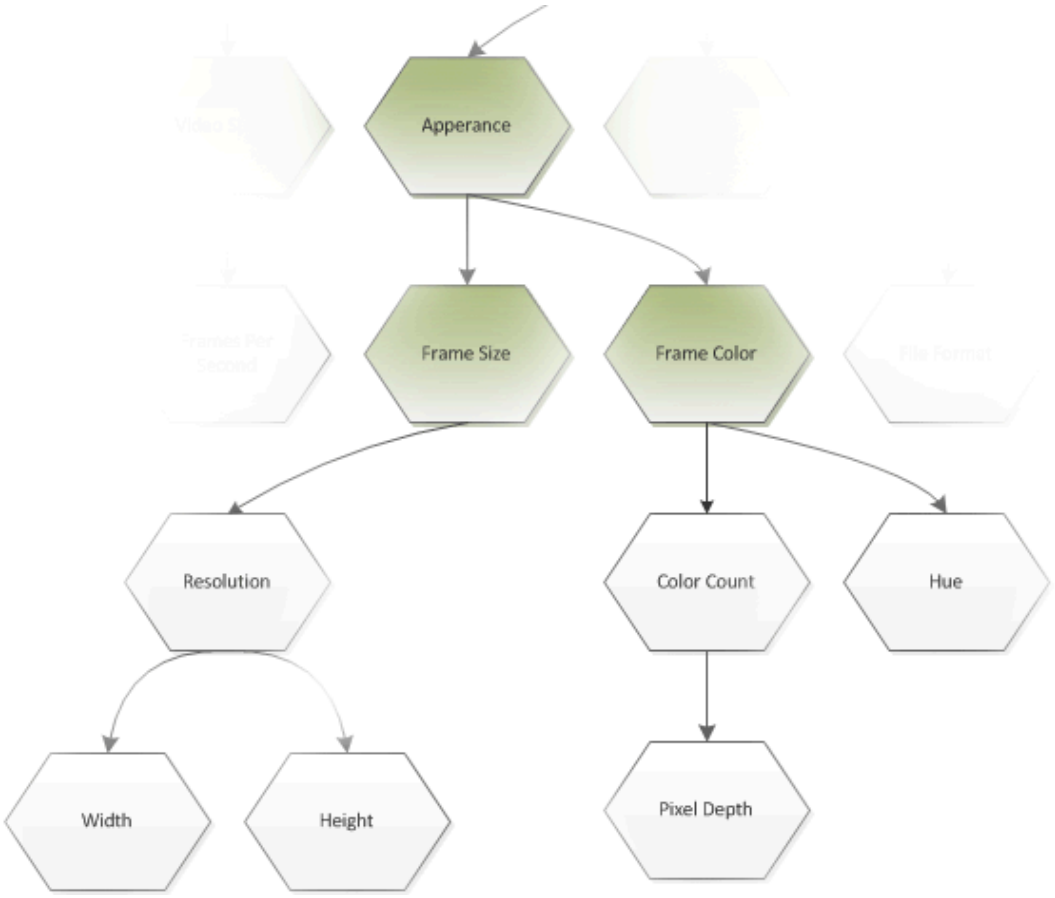


Figure 5.1: Sub-tree that is calculated as a result of the user preference selection

With the combination of the device constraints, the extracted sub-trees are reduced

to the features needed to process the request. Corresponding services in the VSM hierarchy are then located to form the functional group.

In the simulation, the arrival of user requests is generated using a poisson distribution, so they can be then processed by a selected set of four adaptation services (crop, change hue, drop frame and transcoding) using the following three algorithms:
1) Adaptation services are randomly selected to create an adaptation path.
2) "First available queue selection" that was inspired by the First Fit (FF) - a non-prediction-based load balancing algorithm for media clusters, shown to produce higher throughput than the simple round-robin method [35]. In this method the first service whose corresponding queue has a vacancy is selected in a round-robin way in order to become the next "step" in the adaptation path. In a case when all services from the needed set of are occupied, the request is sent to the service, which queue has the least amount of waiting requests.
3) In "Regression" method an adaptation path is assembled with the use of performance graphs that are modeled using regression. During the first run, the simulation wa using the initial performance graphs, obtained from a training set. Subsequent runs used statistical feedback information captured from the video transformations (such as processing time, changes in file size and video parameters) that was then used to correct performance graphs to reflect the current system productivity.
Results are explored for cases when processing services are represented with only one function, and when a service is comprised from several functions that are close in nature (i.e. transcoding service targeting different bitrates) or are formed as combined chains (i.e. function that extracts an area of interest and a function that is created by combining functions that separately crop width and height).

5.2 Experimental Results of MJPG

The MJPG file format uses a compression scheme, where each frame is an independently compressed JPEG still-image. It provides the possibility to create videos of a higher visual quality with minimal level of image degradation (depending upon sampling and compression-ratio of the individual frames). When compared to the video formats that use motion compensation frame prediction techniques, using frame independency facilitates editing while increasing storage costs. The MJPEG video file format is mostly seen on PC's and devices such as sports action and underwater cameras, making it a good

choice for a scenario where conversion to a different codec will have to take place in order to accommodate it for multimedia content streaming.

5.2.1 Characterising the Behavior of VSMs

In order to reduce the extent of data collection, needed for the creation of the performance graphs and corresponding entries in MAPs, the specifics of the MJPG format properties were investigated.

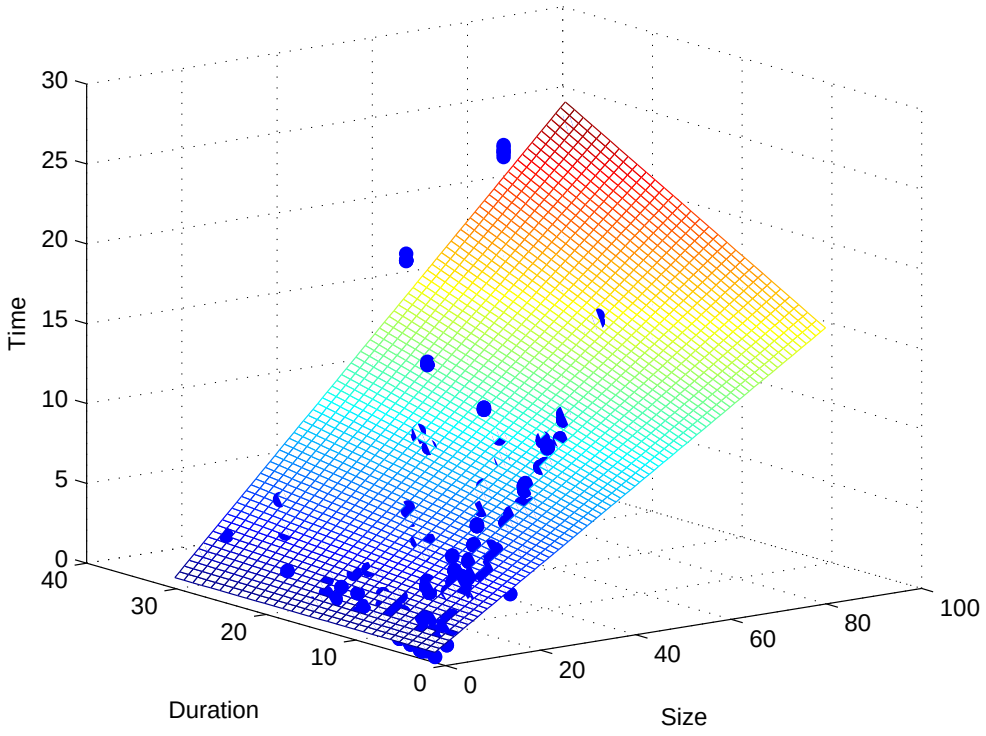


Figure 5.2: Dependance of the discoloration time on the duration and file size (MJPG)

As can be seen from the use of the discoloration service (Figure 5.2), two MJPG videos may share the same input size in terms of bytes, but depending on their duration, the service will produce vastly different output results (as they are also dependent on the multimedia content complexity variation and motion relations). The same will not hold when using the same service with a different file format (Figure 5.3). This can be

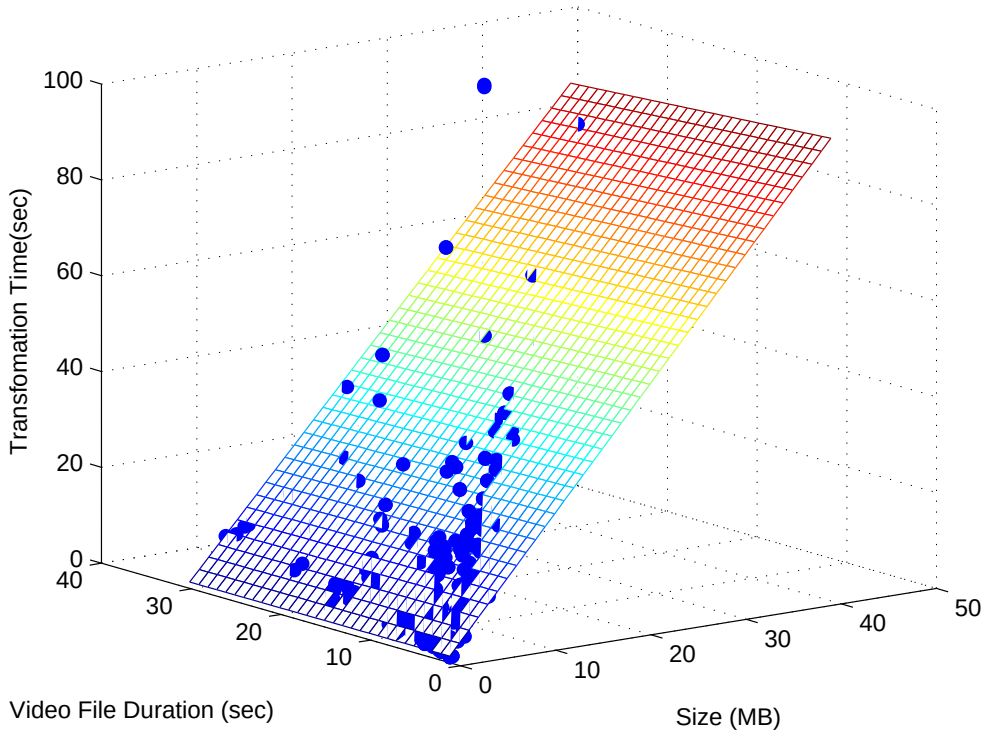


Figure 5.3: Dependance of the discoloration time on the duration and file size (XVID)

explained by the fact that the output byte size and the transcoding time for a JPEG picture can be approximated by a function of the number of pixels in the input image [36]. This suggests, that in case of an MJPG video the number of frames and video resolution might need to be collected in order to create a more precise prediction of the service manipulation results. However this might not be reliable, as in case of MJPG file format there does not exist a unified format specifications, leaving structural enforcement to the various wrapping formats (i.e QuickTime, AVI and ASF). With the assumption that the detailed information about the file format behavior is unknown (even though the AVI file format was selected for the experiments [2]), the general video file information such as duration, overall bit rate, frame rate is obtained from the file tags with the use of "MediaInfo" utility [3].

5.2.2 Use of Primitive VSMs

Using obtained statistical information, regression is used to study the preliminary transcoding effects for each of the individual services. Figure 5.4 shows the average hue adjustment time for MJPG files with regards to their initial size. A polynomial function was selected as a fitting form for the regression. The observations for the "old regression" were taken during the initial system run with the test file set (i.e. only one service was running at any given time), while the the "new regression" reflects parameter relationships, that are captured during continuous observations during the simulation of a continuous stream of user requests.

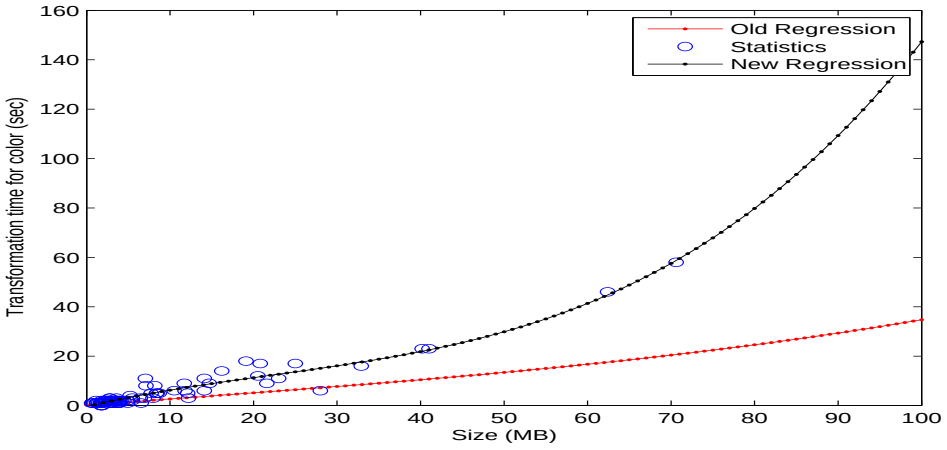


Figure 5.4: Initial regression, observations and refined regression for Hue adjustment

For this particular service, if the system load does not experience sudden changes in performance, the updated regression only effects the ordering of the filters that demonstrate close and resemblant behavior in particular sections of the monitored parameter range. Other services from the pre-defined set demonstrate similar behavior and thus are not presented here in detail.

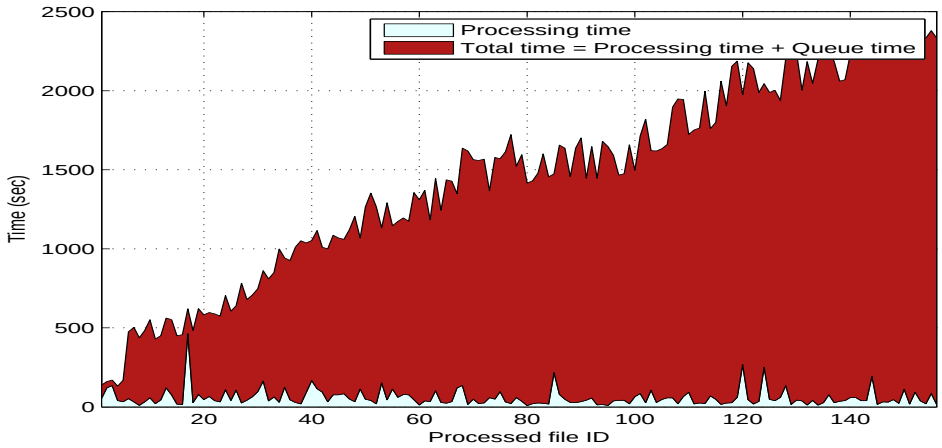
Ordering of the transcoding services have a direct impact on the processing time of the individual files. Presented results study the ordering effectiveness for a functional group containing "crop", "change hue", "drop frame" and "transcoding to XVID services". The requested video files are cropped to fit a predefined dimension, as well as their hue is changed to a predefined value. *Drop frame* service reduces the frame count in half, thus creating a noticeably shorter content version in a concentrated form that

visually looks like fast-forwarding. The Xvid is a MPEG-4 video codec, that achieves high compression rates by removing information irrelevant to human perception, while preserving high visual quality [4].

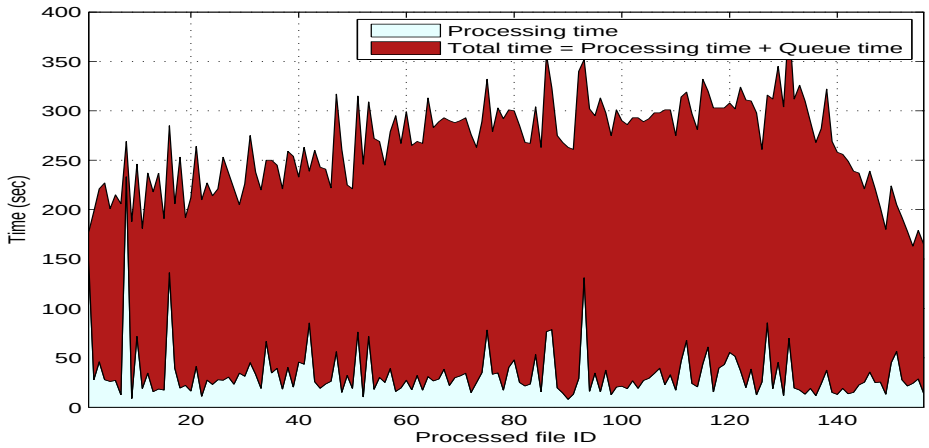
When "Random method" is used to manage the service ordering (Figure 5.5(a)), the queue time continuously increases as the only transcoding service keeps getting occupied by an inefficient processing of large un-resized media files, creating a bottleneck. In the "First Available queue" method (Figure 5.5(b)), the requests are sent to either the first unoccupied service or to the queue that has the least amount of waiting requests. This will automatically balance the queues and direct the requests to services that process the multimedia files quicker. The overall processing time is reduced, since the most time-consuming transcoding service will tend to get to process files that have already been highly reduced by other transformations (each new adaptation step is set to reduce the overall size of the multimedia file). When selecting the service ordering, the "Regression" method (Figure 5.5(c)), with the use of the latest performance graphs directs not only the effective placement of the most time-consuming transcoding service to the end of the transformation chain, but also enforces a more stable and efficient ordering for the current processing conditions of the other services.

5.2.3 Compound VSMs

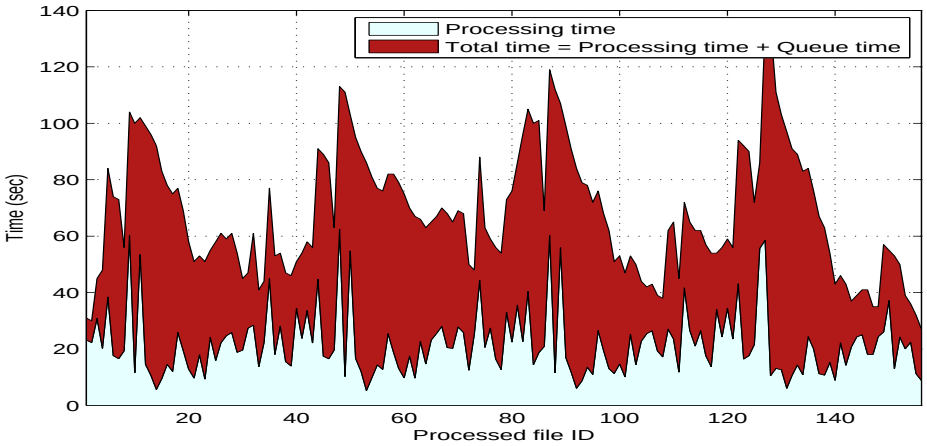
In the next set of experiments each processing function is offered by multiple services; the supplementary "change hue" and "drop frame" service are represented by the duplicates of the original services, while the "transcoding to XVID service" is now offered with three different bitrates and "crop" can be additionally performed by the chains combining two independent width and height modification services. The introduction of supplementary services did not effect the service order selection principles of the "Random" (Figure 5.6(a))and "First available queue"(Figure 5.6(b)) methods. On the other hand, in case of "Regression" method (Figure 5.6(c)), selection of each new step of the adaptation path is performed with the use of the combined performance graph that unites all functions forming the virtual service model. Before processing the actual file, such combined performance graph is used for the creation of the tentative adaptation sequence, that is then used as a guidance structure to process the multimedia content. When the content has to be processed by the VSM, the priority is first given to the service that shows the best results in terms of the QoS criteria requested by the user (during the experiments,



(a) Random path selection method



(b) First available queue method



(c) Performance Graphs

Figure 5.5: Service time for primitive VSMs (MJPG)

fastest processing time was selected as that criteria). If the most productive service, representing the adaptation function, is not available to process the request right away, the request is sent to the next best (in terms of the QoS criteria) available function. When all of the services of a particular VSM are occupied, the "First Available Queue" method is used to determine the function with least requests in the queue. In the case when a desired processing service is formed by the "chain" of functions, the request is forced to follow the sequence determined by the "chain", ignoring the service availability or the queue load.

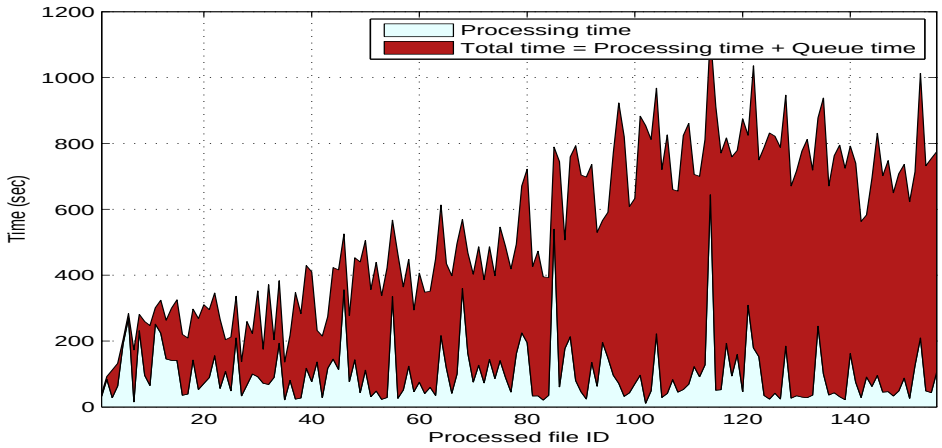
5.2.4 Primitive vs Compound VSMs

Looking at the average processing time for adaptation functions represented by a primitive VSM (Figure 5.7(a)), it can be seen that the ordering suggested by the "Random" method produce the worst outcome in terms of total time needed to process the requested multimedia files. For the same services "First available queue" method is able to improve upon the processing time results, however the best time is obtained using "Regression" methods.

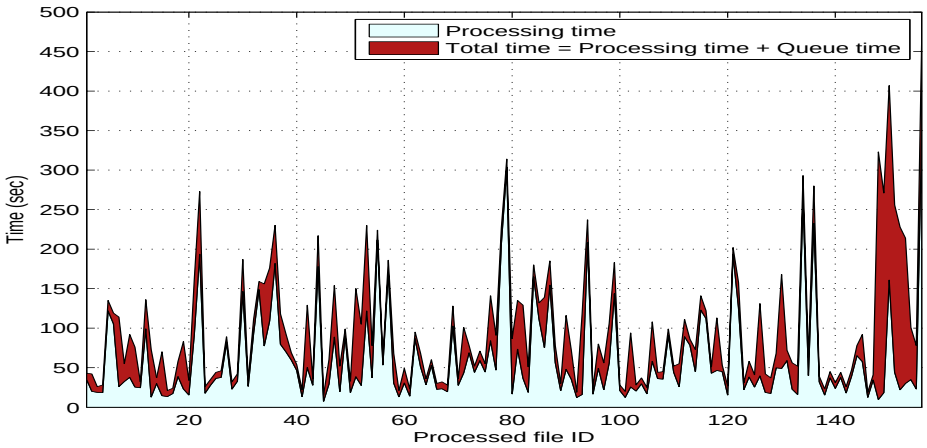
With the introduction of grouped services the most noticeable changes are in the reduction of the processing time for the "Random method" and the increase of the processing time for the other two methods (Figure 5.7(b)). In case of the "First available queue", the processing time is now burdened by the revision of the queue occupancy for all newly introduced services. The processing time increase for the "Regression" method is also caused by the revision times of the additional queues. However, unlike the "First available queue" method, at each service selection step it has to only review queue availability for services that are grouped within one function (i.e. virtual model), overall resulting in a less extensive increase of the processing time.

The queue time is the worst for the "Random" method and minimal for the "Regression method" (Figure 5.8(a)), with the introduction of grouped services the same arrangement holds given the overall reduction of the time that the content now have to spend in the queue (Figure 5.8(b)).

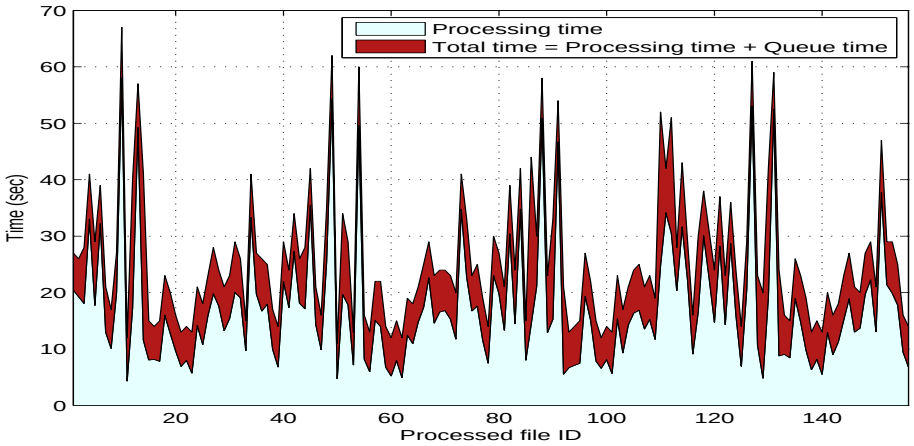
Altogether, the "Regression" method shows good results when working with MJPG file format, when compared to "Random" and "First available queue" methods, it consistently selects the fastest service ordering that produce desired content with the minimum delays even in the light of the increasing number of available services.



(a) Random path selection method

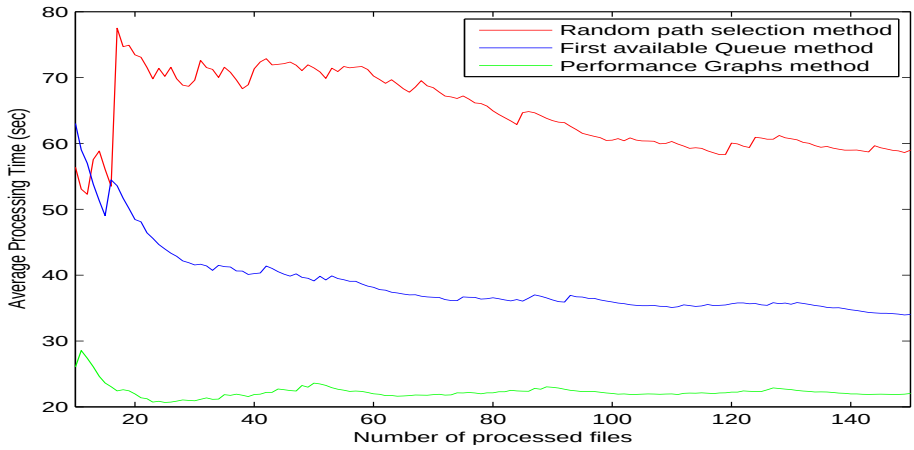


(b) First available queue

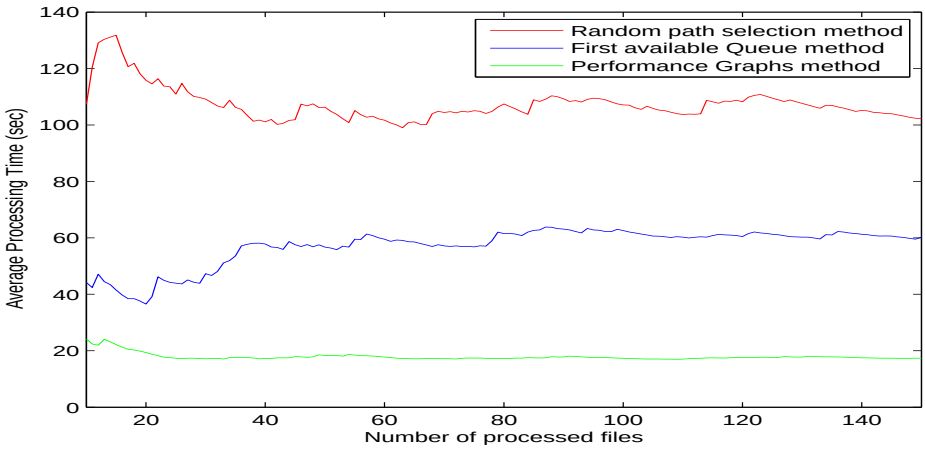


(c) Using Performance Graphs

Figure 5.6: Service time for compound VSMs (MJPG)



(a) Primitive VSMs (MJPG)

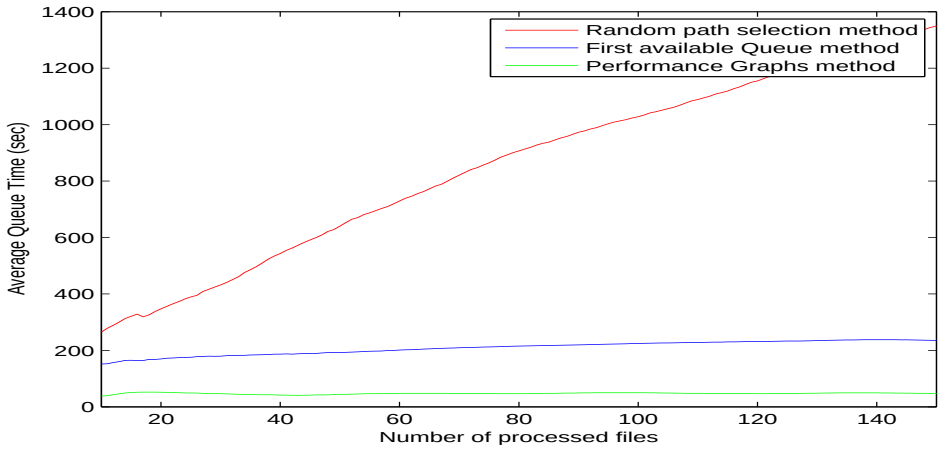


(b) Compound VSMs

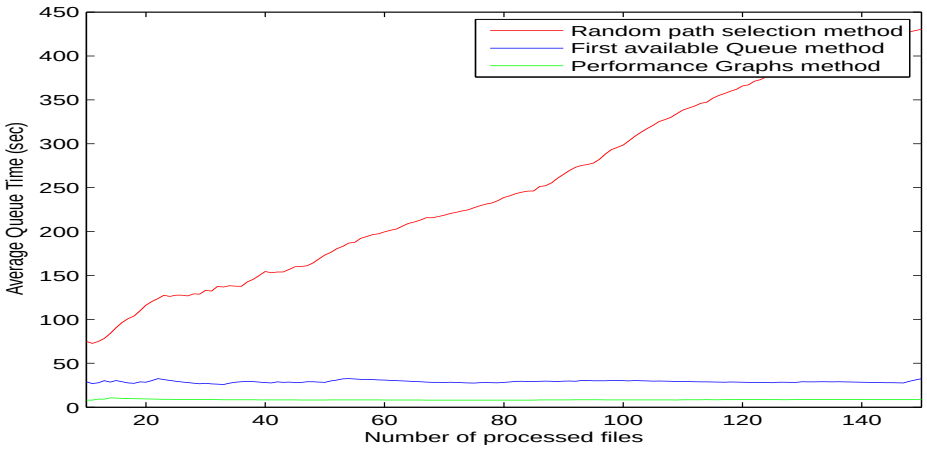
Figure 5.7: Average Processing Time (MJPG)

5.3 Experimental Results of MPEG

When compared to MJPG, the MPEG codec family uses superior compression algorithms allowing it to provide high quality content for digital TV broadcasting and DVD content with a relatively smaller file size. MJPG codec, due to it's properties, allows easy frame-by-frame access/editing as well as robustness to bit-errors during transitions. The distinction of MPEG codec is that it exploits temporal redundancies by utilizing motion compensation and spatial redundancies with the DCT transformations [42]. The



(a) Primitive VSMs



(b) Compound VSMs

Figure 5.8: Average Queueing Time (MJPG)

MPEG2 coding scheme that was selected for the experiments, is a good representation for a current trend of devices for video streaming on the Internet. In this set of experiments, ordering techniques are applied to a functional group containing "crop", "change hue", "drop frame" and "transcoding to H264 file format" services. Essentially, these services offer the exact same functionalities as in the case of MJPG content, with the exception of the transcoding service. Although H.264 employs a hybrid coding approach similar to that of MPEG-2 in concept, it is different in realization techniques (such is the use of an integer transform with energy compaction properties similar to that of the

DCT, instead of the actual DCT [42]), allowing to offer good video quality at a lower rate then MPEG2.

5.3.1 Use of Primitive VSMs

The "Random" method demonstrates trends similar to the ones observed in case of MJPG format, when the ineffective service ordering causes high occupancy of the slow-processing transcoding service, resulting in ever increasing queue times (Figure 5.9(a)). The "First Available queue" method experiences some spikes in the processing time at the beginning (i.e when the only unoccupied service to process the unmodified file is the time-consuming transcoding), but then, once again, it is stabilized and further demonstrates consistent queue and processing time (Figure 5.9(b)). With the use of the latest performance graphs, the "Regression" method selects the service ordering that provides the fastest delivery of the requested content (Figure 5.9(c)), while demonstrating stable queue time. It can be noted, that at first the queue time keeps increasing, but then it stabilizes, which can be clearly seen in the comparison of the average processing (Figure 5.10(a)) and queue times (Figure 5.10(b)) for the three methods.

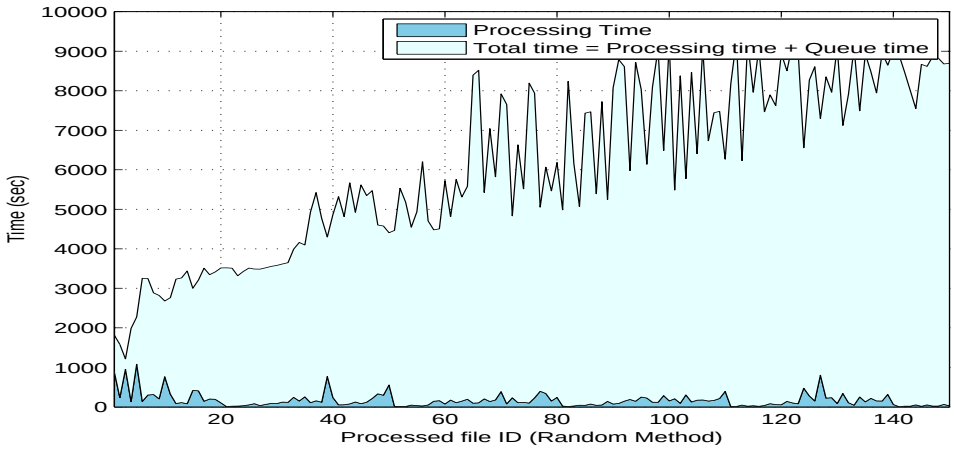
5.3.2 Compound VSMs

Once again, for experiments when each processing function is offered by multiple services, the supplementary "change hue" and "drop frame" are represented by duplicates of the original services, while "transcoding" is expended with services using different bitrates and "crop" is supplemented with chains of services.

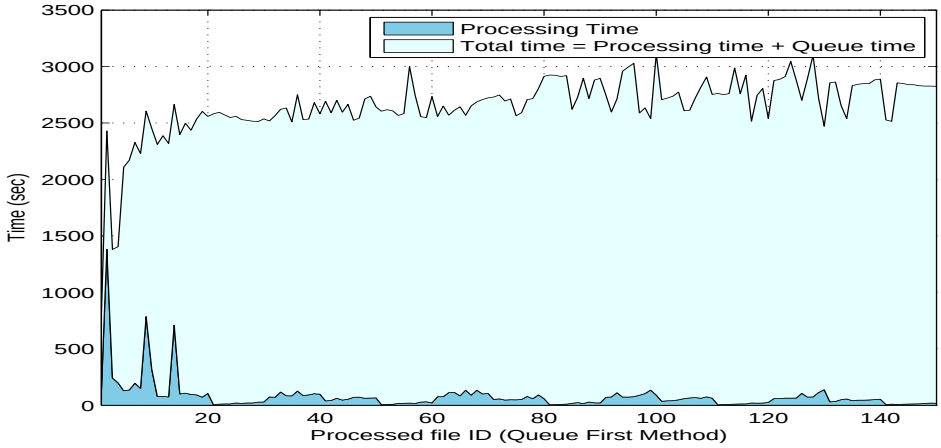
At the first glance, there is not much difference in the results obtained from primitive VSMs (Figure 5.10(a), 5.10(b)) and compound VSMs (Figure 5.12(a), 5.12(b)). However, when graphed together (in this case for the "Regression" method), it can be seen that grouped services do improve the queue time (Figure 5.13(b)). At the same time the improvement of the processing time is almost negligible (Figure 5.13(a)). However, considering that the simulation concentrated on the creation of separate processes and not completely independent processing services (i.e. independent processing resources), even such small improvement in the processing time indicates the potential for the service grouping.

5.4 Chapter Summary

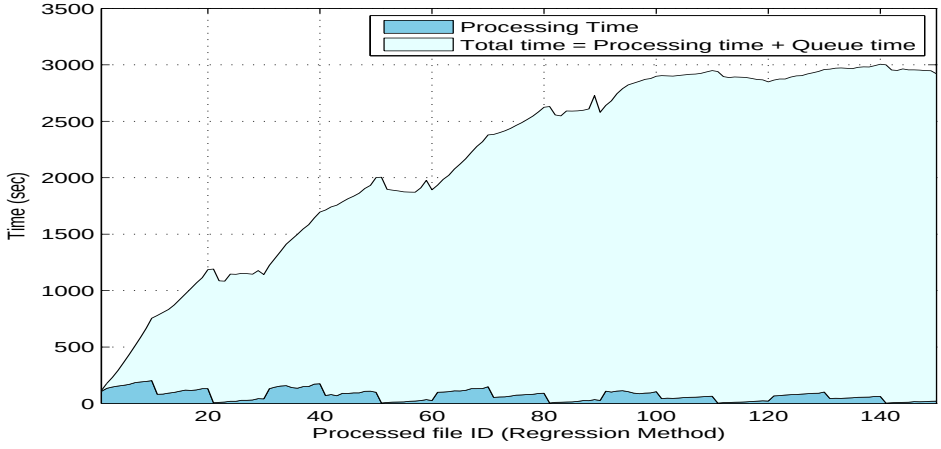
This chapter presented the results of the experiments that were conducted on video files coded in MJPG and MPEG formats. Each group of test files was processed by a similar set of services with accordance to random, first available queue and performance graphs service ordering algorithms. The algorithms performances were also evaluated for the cases of stand-alone services, as well as for the cases when a desired functionality was offered by several different services. It is shown that overall performance graph method shows the best results in terms of the time needed to obtain the desired results, as well as for the queue time experienced during such processing. The next chapter will summarize the experimental results and elaborate on the future work.



(a) Random path selection method

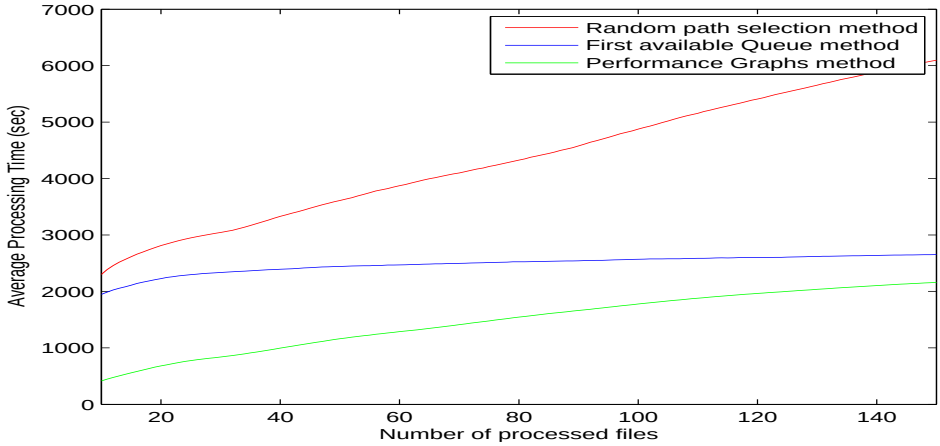


(b) First available queue

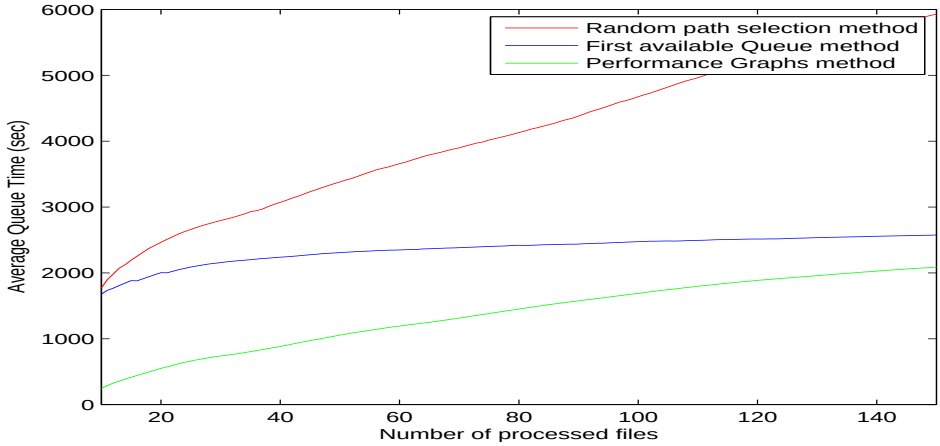


(c) Using Performance Graphs

Figure 5.9: Service time for primitive VSMs (MPEG)

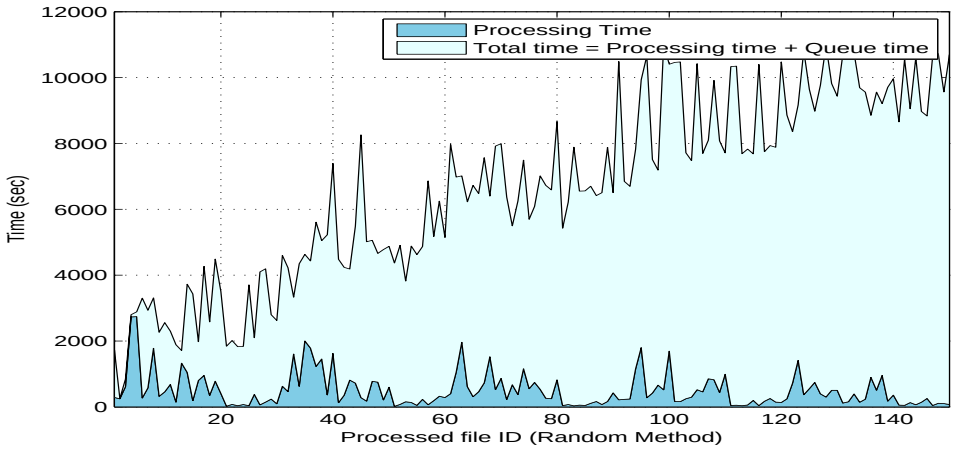


(a) Average Processing Time

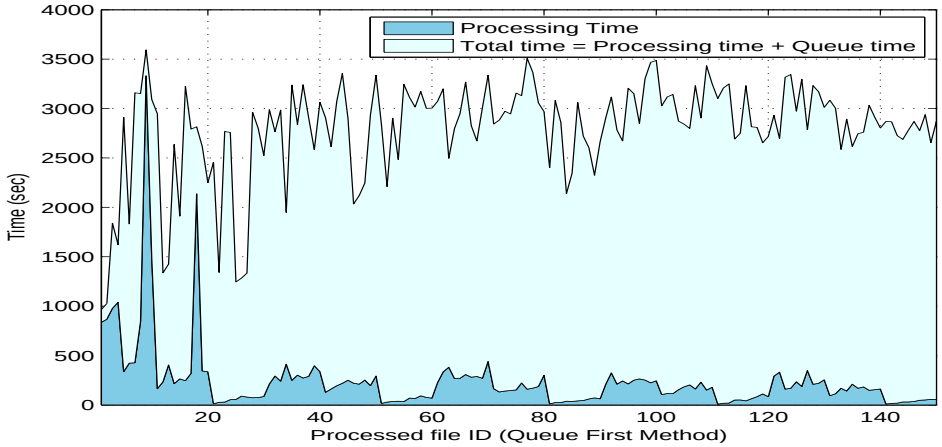


(b) Average Queue Time

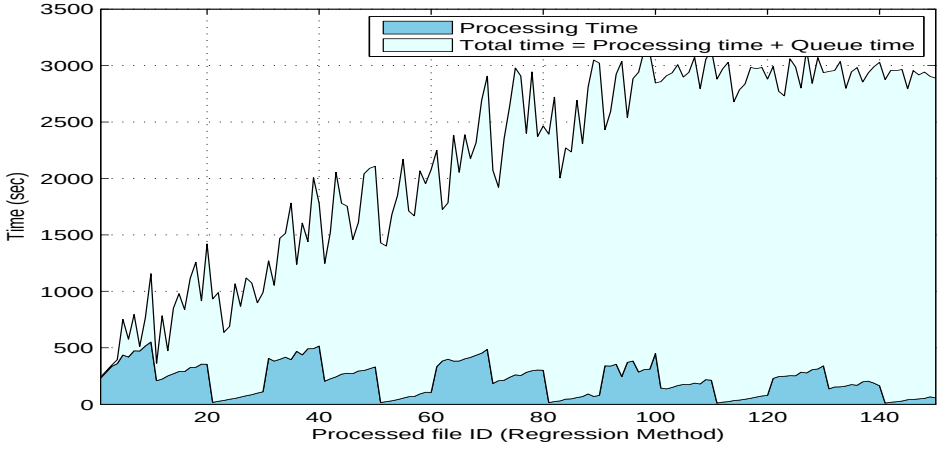
Figure 5.10: Performance of single services (MPEG)



(a) Random path selection method

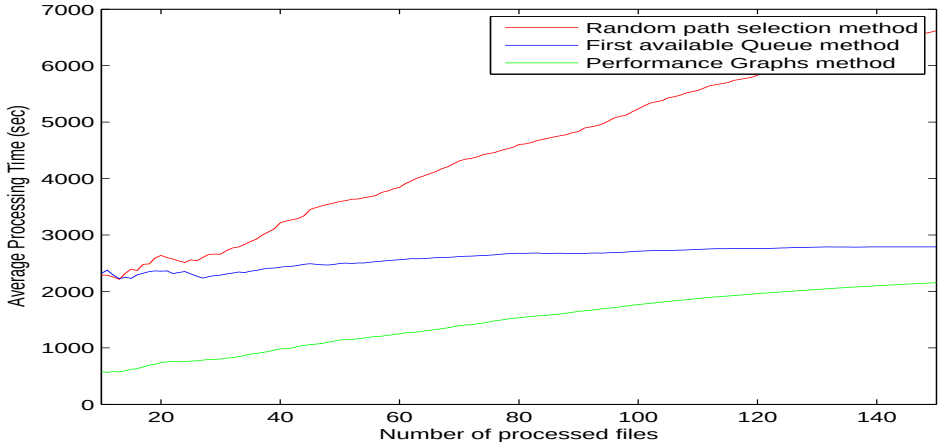


(b) First available queue

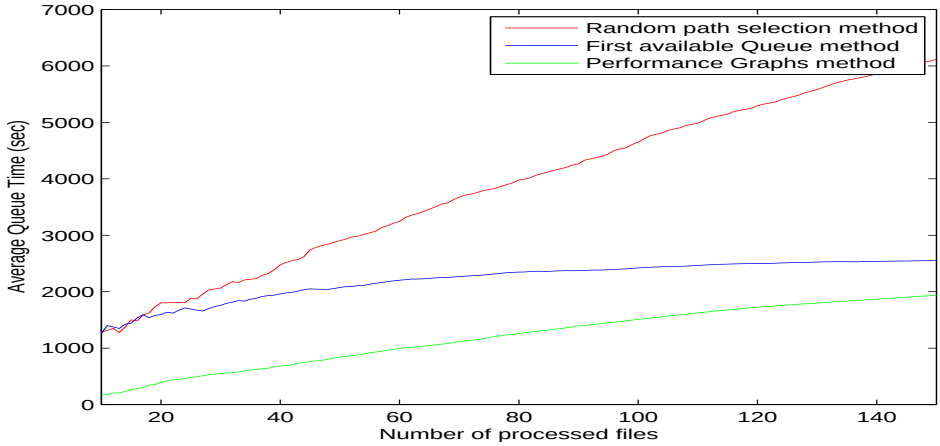


(c) Using Performance Graphs

Figure 5.11: Service time for compound VSMs (MPEG)

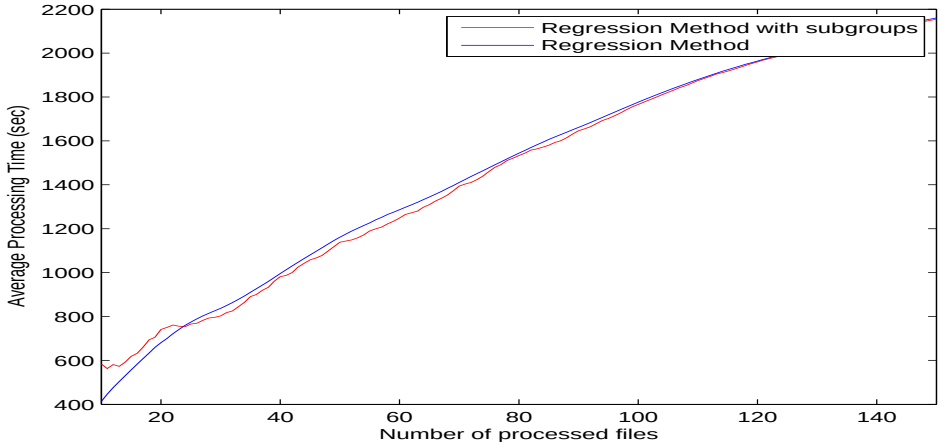


(a) Average Processing Time

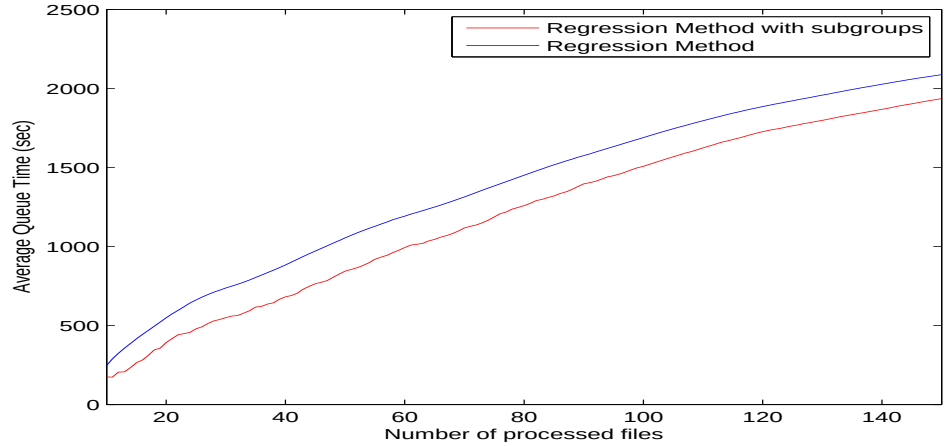


(b) Average Queue time

Figure 5.12: Performance of compound VSMs (MPEG)



(a) Processing time



(b) Queue time

Figure 5.13: Performance of single and grouped services Using Regression (MPEG)

Chapter 6

Conclusion and Future Work

6.1 Conclusion and Thesis Contribution

The goal of this work has been the exploration of the possibility of integrating content adaptation techniques with the emerging cloud computing paradigm. The search for a solution of locating the effective sequences of adaptation services in a timely fashion started with the revision of the diversity of the available content, content delivery and adaptation mechanisms. Inspired by the existing techniques, a novel service virtualization architecture (SVA) was introduced.

In order to embrace the possibilities offered by the cloud computing concept, this work explored creation of dynamically reconfigurable virtualized service models that hide the details of the service configurations offered over service clouds. Representation of different types of the multimedia content in terms of the core features (i.e., features consistent through entire multimedia types and not susceptible to media format and re-encodings changes) allowed to create a unified modeling technique for effective structuring multimedia content features and adaptation services. With the use of such structures the user requests can be easily expressed and mapped to the multimedia features that then help identify services needed to obtain the desired outcome. Performed experiments provide another affirmation that the end-results of the adaptation depend on the selected service ordering in such parameters as overall processing time and the file size of the adapted content. The introduced performance graph technique utilizes the performance-history based data to order the identified adaptation services with accordance to their relations with each other and the particular content. The use of the regression methods was

suggested for the possible creation of such performance graphs. Coupled with the mechanisms for tracking dynamic changes of network components status and performance fluctuations, they allow the creation of compact and effective description for the need service performance relations.
The proposed architectural solutions were implemented in order to evaluate their performances. The multimedia feature hierarchy was tested in terms of the user-interactions (i.e. tested the generation of the sub-tree of features corresponding to the criteria selected by the user at different detalization levels). To test the effectiveness of the proposed ordering technique MJPG and MPEG2 video formats were selected. A set of four adaptation functions (i.e. "crop", "change hue", "drop frame" and "transcoding") were tested with the use of the performance graphs (realized using regression) along with the "random" and "first available queue" methods. The proposed ordering technique shows the best results in terms of the time needed to obtain the desired results as well as in the queue time experienced during such processing. The same results hold for both MJPG and MPEG formats when adaptation functions are modeled with one or multiple services.

6.2 Future Work

Conducted experiments show that the use of the core multimedia features has the potential to create quick estimations of the effective service ordering for the set of selected candidate services. It showed good results for video files and it would be interesting to adapt and test the proposed architecture to work with media streams. That would be possible with the addition of the mechanisms capable of capturing and reacting to changes of system parameters and/or user preferences over the life of the stream [53]. The other expansion of the proposed architecture lies in the area of user privacy and it's protection, as when using intermediate servers for content adaptation the traditional ways of end-to-end security solution might not be always applicable (i.e. content has to be transcoded before it is encrypted). In such cases, adaptation authorization becomes an additional constraint to represent content owner's rights [9], possibly in a form of rules and policies in order to further guide the processing of user requests in Service Virtualization Architecture (SVA).

Bibliography

[1] `http://www.mplayerhq.hu/`. Accessed Aug 2012.

[2] `http://www.fileformat.info/format/bmp/spec/`. Accessed Aug 2012.

[3] `http://mediainfo.sourceforge.net/en`. Accessed Aug 2012.

[4] `http://www.xvid.org/Home-of-the-Xvid-Codec.1.0.html`. Accessed Aug 2012.

[5] Herv Abdi and Lynne J. Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4), 2010.

[6] B. Abraham and J. Ledolter. *Statistical methods for forecasting*. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1983.

[7] Velibor Adzic, Hari Kalva, and Borko Furht. A survey of multimedia content adaptation for mobile devices. *Multimedia Tools and Applications*, 51:379–396, January 2011.

[8] I. Al-Oqily and A. Karmouch. Sord: A fault-resilient service overlay for mediaport resource discovery. *Parallel and Distributed Systems, IEEE Transactions on*, 20:1112 –1125, Aug 2009.

[9] Maria Teresa Andrade, Hemantha Kodikara Arachchi, Sabih Nasir, Safak Dogan, Halil Uzuner, Ahmet M. Kondoz, Jaime Delgado, Eva Rodrguez, Anna Carreras, Tim Masterton, and Rachel Craddock. Using context to assist the adaptation of protected multimedia content in virtual collaboration applications. In *CollaborateCom'07*, pages 233–242.

[10] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[11] V. K. Banga Avneet Kaur. Color based image retrieval. *PSRC Proceedings Pattaya Conferences*, 2011.

[12] Vidhya Balasubramanian and Nalini Venkatasubramanian. Server transcoding of multimedia data for cross-disability access.

[13] Girma Berhe, Lionel Brunie, and Jean-Marc Pierson. Distributed content adaptation for pervasive systems. In *Proceedings of the International Conference on Information Technology: Coding and Computing*, ITCC '05, pages 234–241, Washington, DC, USA. IEEE Computer Society.

[14] Girma Berhe, Lionel Brunie, and Jean-Marc Pierson. Modeling service-based multimedia content adaptation in pervasive computing. In *Proceedings of the 1st conference on Computing frontiers*, CF '04, pages 60–69, New York, NY, USA. ACM.

[15] N.S. Bhuvaneswari, N.S.B.S. Sujatha, and S. Sujatha. *Integrating Soa and Web Services*. River Publishers, 2011.

[16] Susanne Boll and Wolfgang Klas. Zyx-a multimedia document model for reuse and adaptation of multimedia content. *IEEE Transactions on Knowledge and Data Engineering*, 13(3):361–382.

[17] Piero A. Bonatti, Carsten Lutz, Aniello Murano, and Moshey. Vardi. Iso/iec 13818-2 mpeg-2. information technology - generic coding of moving pictures and associated audio information: video. In *ICALP 2006. LNCS*, pages 540–551. Springer, 2006.

[18] Gianluca Bontempi and Gauthier Lafruit. Enabling multimedia qos control with black-box modelling. In *Proceedings of the First International Conference on Computing in an Imperfect World*, London, UK, UK, 2002. Springer-Verlag.

[19] Jens Brandt, Lars Wolf, and Paal Halvorsen. Multidimensional transcoding for adaptive video streaming. *SIGMultimedia Rec.*, 1:16–17.

[20] K. Mani Chandy and J. Misra. Distributed computation on graphs: shortest path algorithms. *Communications of the ACM*, 25(11):833–837.

[21] S.F. Chang and A. Vetro. Video adaptation: concepts, technologies, and open issues. *Proceedings of the IEEE*, 93(1):148–158, 2005.

[22] N.M. Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 54(5):862–876.

[23] S. Dey. Cloud mobile media: Opportunities, challenges, and directions. In *International Conference on Computing, Networking and Communications (ICNC)*, pages 929 –933, Feb 2012.

[24] EW Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[25] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. page 29, 2004.

[26] I. Djama, T. Ahmed, A. Nafaa, and R. Boutaba. Meet in the middle cross-layer adaptation for audiovisual content delivery. *IEEE Transactions on Multimedia*, 10(1):105–120, Jan 2008.

[27] Robson Eisinger, Roseli A. F. Romero, and Rudinei Goularte. Machine learning techniques applied to dynamic video adapting. *Fourth International Conference on Machine Learning and Applications*, pages 819–822, 2008.

[28] Khalil El-Khatib, Gregor von Bochmann, and Abdulmotaleb El-Saddik. A qos-based service composition for content adaptation. In *ICDE Workshops'07*, pages 331–338.

[29] Mercus Brunner Eskindir Asmare, Stefan Schmid. Setup and maintenance of overlay networks for multimedia services in mobile environments. *Proceedings of the 8th international conference on Management of Multimedia Networks and Services*, pages 82–95, 2005.

[30] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. pages 1–10.

[31] R. Garca and O. Celma. Semantic integration and retrieval of multimedia metadata. volume 185, pages 69–80. CEUR Workshop Proceedings, 2005.

[32] David Gibbon, Andrea Basso, Lee Begeja, Zhu Liu, Bernard Renger, Behzad Shahraray, and Eric Zavesky. Large-scale analysis for interactivemedia consumption. *TV Content Analysis: Techniques and Applications*, June 2012.

[33] Ruben Gonzalez. Disciplining multimedia. *IEEE MultiMedia*, 7(3):72–78, 2000.

[34] Xiaohui Gu and Klara Nahrstedt. Distributed multimedia service composition with statistical qos assurances. *IEEE Transactions on Multimedia*, 8(1):141–151, 2006.

[35] Jiani Guo and Laxmi N. Bhuyan. Load balancing in a cluster-based web server for multimedia applications. *IEEE Transactions on Parallel and Distributed Systems*, 17(11):1321–1334, 2006.

[36] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas. Dynamic adaptation in an image transcoding proxy for mobile web browsing. *IEEE Personal Communications*, 5(6):8 –17, Dec 1998.

[37] Stephen Herborn, Yoann Lopez, and Aruna Seneviratne. A distributed scheme for autonomous service composition. In *Proceedings of the first ACM international workshop on Multimedia service composition*, MSC '05, pages 21–30, New York, NY, USA. ACM.

[38] Jung-Lee Hsiao, Hao-Ping Hung, and Ming-Syan Chen. Versatile transcoding proxy for internet content adaptation. *IEEE Transactions on Multimedia*, 10(4):646–658.

[39] Noha Ibrahim and Frederic Le Mouel. A survey on service composition middleware in pervasive environments.

[40] Steven Ihde, Paul P. Maglio, Jrg Meyer, and Rob Barrett. Intermediary-based transcoding framework. *IBM Systems Journal*, pages 179–179, 2001.

[41] Sirish Chandrasekaran. Sirish Ch. Samuel Madden. Mihut Ionescu. Ninja paths: An architecture for composing services over wide area networks. *University of California Berkley technical report*, 2000.

[42] H. Kalva. Issues in h.264/mpeg-2 video transcoding. In *First IEEE Consumer Communications and Networking Conference*, pages 657 – 659, Jan 2004.

[43] E. Korotich and N. Samaan. A novel architecture for efficient management of multimedia-service clouds. In *IEEE Global Communications Conference, Enabling Green Wireless Multimedia Communications*, pages 723 –727, Dec. 2011.

[44] Janine Lachner, Andreas Lorenz, Bernhard Reiterer, Andreas Zimmermann, and Hermann Hellwagner. Challenges toward user-centric multimedia. *International Workshop on Semantic Media Adaptation and Personalization*, 0:159–164, 2007.

[45] Zhijun Lei and N.D. Georganas. Context-based media adaptation in pervasive computing, 2001.

[46] Qin Li, Zhao Lu, Yunfei Yu, and Lu Liang. Multimedia ontology modeling: An approach based on mpeg-7. In *3rd International Conference on Advanced Computer Control (ICACC), Jan 2011*, pages 351 –356.

[47] Huan Liu and Lei Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, April 2005.

[48] Wai Yip Lum and Francis C. M. Lau. On balancing between transcoding overhead and spatial consumption in content adaptation. In *MOBICOM'02*, pages 239–250.

[49] Wai Yip Lum and Francis C. M. Lau. User-centric content negotiation for effective adaptation service in mobile computing. *IEEE Transactions on Software Engineering*, 29:1100–1111, December 2003.

[50] Derdour Makhlouf, Philippe Roose, Marc Dalmau, and Nacira Ghoualmi Zine. An adaptation platform for multimedia applications csc (component, service, connector). *Journal of Systems and Information Technology*, 14(1), 2012.

[51] M.F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis. Virtualization for high-performance computing. *ACM SIGOPS Operating Systems Review*, 40(2):11, 2006.

[52] Dionisio John David N. and Cardenas Alfonso F. A unified data model for representing multimedia, timeline, and simulation data. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):746–767, 1998.

[53] Klara Nahrstedt and Wolf-Tilo Balke. Towards building large scale multimedia systems and applications: challenges and status. In *MSC '05: Proceedings of the first ACM international workshop on Multimedia service composition*, pages 3–10, New York, NY, USA. ACM.

[54] Klara Nahrstedt and Wolf-Tilo Balke. A taxonomy for multimedia service composition. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 88–95. ACM, 2004.

[55] Zeljko Obrenovic, Dusan Starcevic, and Bran Selic. A model-driven approach to content repurposing. *IEEE MultiMedia*, 11(1):62–71.

[56] F. Pereira and I. Burnett. Universal multimedia experiences for tomorrow. *IEEE Signal Processing Magazine*, 20(2):63–73, March 2003.

[57] FERNANDO PEREIRA. Multimedia content adaptation: May one fit all? In K. Wojciechowski, B. Smolka, H. Palus, R.S. Kozera, W. Skarbek, and L. Noakes, editors, *Computer Vision and Graphics*, volume 32 of *Computational Imaging and Vision*, pages 337–342. Springer Netherlands, 2006.

[58] Zhuzhong Qian, Minyi Guo, Sheng Zhang, and Sanglu Lu. Service-oriented multimedia delivery in pervasive space. In *Proceedings of the IEEE conference on Wireless Communications and Networking Conference*, pages 3012–3017, Piscataway, NJ, USA, 2009.

[59] J.O. Rawlings, S.G. Pantula, and D.A. Dickey. *Applied Regression Analysis: A Research Tool*. Springer Texts in Statistics. Springer, 1998.

[60] John Reumann, Ashish Mehra, Kang G. Shin, and Dilip Kandlur. Virtual services: a new abstraction for server consolidation. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 10–10, Berkeley, CA, USA, 2000. USENIX Association.

[61] J.R.Smith R.Mohan and C.-S. Li. Adapting multimedia internet content for universal access. *IEEE Trans. Multimedia*, 1:104–114, March 1999.

[62] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web service modeling ontology. *Applied Ontology*, 1(1):77–106, 2005.

[63] T.P. Ryan. *Modern regression methods*. Wiley series in probability and statistics. Wiley, 2009.

[64] Vidyut Samanta, Ricardo V. Oliveira, Advait Dixit, Parixit Aghera, Petros Zerfos, and Songwu Lu. In *First International Conference on Communication System Software and Middleware, Comsware 2006*.

[65] G.A. Schloss and M.J. Wynblatt. Using a layered paradigm to model multimedia. *Hawaii International Conference on System Sciences*, 1995.

[66] Evaggelos Spyrou, Herv Le Borgne, Theofilos P. Mailis, Eddie Cooke, Yannis S. Avrithis, and Noel E. O'Connor. Fusing mpeg-7 visual descriptors for image classification. In *ICANN (2)'05*, pages 847–852, 2005.

[67] Katarina Stanoevska-Slabeva and Thomas Wozniak. Cloud basics an introduction to cloud computing. In Katarina Stanoevska-Slabeva, Thomas Wozniak, and Santi Ristol, editors, *Grid and Cloud Computing*, pages 47–61. Springer Berlin Heidelberg, 2010.

[68] R. Steinmetz and K. Nahrstedt. *Multimedia: computing, communications, and applications*. Innovative technology series. Prentice Hall, 1995.

[69] Elias Susan, Raj Suprema, Lakshmanan Uma, Premkumar Sunaina, Easwarakumar K. S., and Chbeir Richard. Enabling dynamic content adaptation in distributed multimedia systems. In *1st International Conference on Digital Information Management, 2006*, pages 75–80.

[70] Information technology. Multimedia framework (mpeg-21) – part 7: Digital item adaptation. *ISO/IEC 21000-7:2007*.

[71] A.G. Tescher. Multimedia is the message. *IEEE Signal Processing Magazine*, 16:44–54, Jan 1999.

[72] J.A.I. Thomas-Kerr, I.S. Burnett, C.H. Ritz, D. De Schrijver, R. Van de Walle, and S. Devillers. Is that a fish in your ear? a universal metalanguage for multimedia. *IEEE Multimedia*, 14:72–77, 2007.

[73] Gijs Van Tulder. Storing hierarchical data in a database. `http://www.sitepoint.com/hierarchical-data-database/`. 2003.

[74] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55.

[75] Sudarshan Vasudevan, Jim Kurose, and Don Towsley. Design and analysis of a leader election algorithm for mobile ad hoc networks. In *Proceedings of the 12th IEEE International Conference on Network Protocols*, pages 350–360, Washington, DC, USA, 2004. IEEE Computer Society.

[76] Yong Wang, Mihaela van der Schaar, Shih-Fu Chang, and Alexander C. Loui. Classification-based multidimensional adaptation prediction for scalable video coding using subjective quality evaluation. *IEEE Transactions on Circuits and Systems for Video Technology*, 15:1270–1279, 2005.

[77] Li-Qun Xu and Yongmin Li. Video classification using spatial-temporal features and pca. *IEEE International Conference on Multimedia and Expo*, 3:485–488, 2003.

[78] Liu Ying, Zhang Dengsheng, Lu Guojun, and Ma Wei-Ying. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1), January 2007.

[79] Er C. Loui Yong Wang, Shih-fu Chang. Content-based prediction of optimal video adaptation operations using subjective quality evaluation. *Columbia University ADVENT Technical Report 202-2004-2, January 2004.*

[80] Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler. Using smart clients to build scalable services. In *Proceedings of the USENIX Technical Conference*, pages 105–117, 1997.

[81] Chandranmenon Girish Yui-Wah Lee and Scott C. Miller. Gamma: A content-adaptation server for wireless multimedia applications. *Lucent Technologies white paper.*

[82] Markos Zampoglou, Theophilos Papadimitriou, and Konstantinos I. Diamantaras. From low-level features to semantic classes: Spatial and temporal descriptors for video indexing. *Signal Processing Systems*, 61:75–83, 2010.

[83] Wenwu Zhu, Chong Luo, Jianfeng Wang, and Shipeng Li. Multimedia cloud computing. *IEEE Signal Processing Magazine*, 28(3):59 –69, May 2011.