Architekturvorschlag für ein serverbasiertes Farbmanagementsystem für displaybasierte Endgeräte

Dissertation zur Erlangung des akademischen Grads Dr.-Ing.
an der Fakultät für Elektrotechnik, Informationstechnik und Medientechnik
der Bergischen Universität Wuppertal

vorgelegt von
Stefan Lahme, M.Sc.

Velbert, 4. Februar 2018

1. Gutachter: Prof. Dr. rer. nat. Stefan Brües

2. Gutachter: Prof. Dr.-Ing. Wolfgang Kühn

Mündliche Prüfung: 19. Juli 2018

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20180801-133352-6
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20180801-133352-6]

Danksagung

Mein besonderer Dank gilt meinem Doktorvater Herrn Prof. Dr. rer. nat. Stefan Brües für die Unterstützung dieses Promotionsvorhabens, die vertrauensvolle Zusammenarbeit, das stets fachkundige und hilfreiche Feedback sowie die Bereitstellung der benötigten Geräte und Arbeitsmittel.

Ebenfalls danke ich Herrn Prof. Dr.-Ing. Wolfgang Kühn für sein Interesse an diesem Thema sowie die Begutachtung dieser Arbeit. Bei allen weiteren Mitarbeiterinnen und Mitarbeitern der Fakultät für Elektrotechnik, Informationstechnik und Medientechnik möchte ich mich zudem für die freundschaftliche und kollegiale Zusammenarbeit bedanken.

Weiterhin danke ich allen Studierenden des Studiengangs Druck- und Medientechnologie, die im Rahmen ihrer Projekt- und Abschlussarbeiten an dieser Arbeit mitgewirkt haben.

Abschließend möchte ich mich vor allem bei meiner Frau und meiner Familie bedanken, die durch ihren Zuspruch und ihre immerwährende Unterstützung dazu beigetragen haben, diese Arbeit erfolgreich fertigzustellen.

„An architectural proposal for a server-based color management system suitable for display-based devices“

Abstract

Each display-based device model, such as a specific smartphone, has distinctive color rendering properties, which are influenced by the built-in display technology as well as the system-specific brightness and color settings.

This work is concerned with the approach of an architectural proposal for a server-based color management system, which is designed to ensure cross-platform color consistency and is particularly suitable for connected, display-based devices.

For this purpose, the causes for individual color rendering properties of display-based devices are explained initially. In addition, the main components of a color management system, which are required for the realization of the proposed design, are introduced.

In order to differentiate the proposed architectural design from other solutions, the existing approaches are introduced and classified regarding their purpose as well as their advantages and disadvantages.

The focus of this work is a detailed modelling of the proposed color management architecture. In this context, the individual layers, the contained modules and their interaction are explained. This includes the design of an object-oriented data model, the application logic as well as the interfaces to connected peripheral systems.

Subsequently, a proposal of an application programming interface based on modern communication standards is introduced. Furthermore, the architectural design is implemented prototypically using current tools and frameworks.

With the use of the implemented prototype, the color matching potential of the proposed system is examined using current, display-based devices. In this concern, both a visual and a measurement-based evaluation is conducted.

Finally, the main results of the work are summarized and classified according to the initial objective. In addition, a prospect on future issues as well as potential enhancements is given.

Kurzfassung

Jedes displaybasierte Gerätemodell, wie z.B. ein spezifisches Smartphone, besitzt charakteristische Farbwiedergabeeigenschaften, die einerseits von der zugrunde liegende Displaytechnologie als auch von den systemseitigen Helligkeits- und Farbeinstellungen beeinflusst werden.

Diese Arbeit befasst sich mit dem Architekturvorschlag für ein serverbasiertes Farbmanagementsystem, das eine plattformübergreifende, konsistente Farbwiedergabe gewährleisten soll und sich insbesondere für den Einsatz auf displaybasierten Endgeräten eignet.

Hierzu werden eingangs die Ursachen für die individuellen Farbwiedergabeeigenschaften displaybasierter Endgeräte erläutert sowie zentrale Systemkomponenten vorgestellt, die für die Umsetzung des Farbmanagementsystems erforderlich sind.

Um den vorgeschlagenen Architekturentwurf gegenüber anderen Systemen abzugrenzen, werden vorab bereits existierende Architekturansätze vorgestellt und im Hinblick auf ihren Einsatzzweck sowie ihre jeweiligen Vor- und Nachteile eingeordnet.

Den Kern der Arbeit bildet eine ausführliche Modellierung der vorgeschlagenen Farbmanagementarchitektur. Hierzu werden die einzelnen Schichten, die darin enthaltenen Module und deren Zusammenspiel erläutert. Unter anderem werden das zugrunde liegende, objektorientierte Datenmodell, die zentrale Anwendungslogik sowie Schnittstellen zu angebundenen Peripheriesystemen beschrieben.

Im Anschluss an die Architekturmodellierung erfolgt ein Vorschlag für die Umsetzung einer Programmierschnittstelle auf Basis gegenwärtiger Kommunikationsstandards, sowie eine prototypische Implementierung des vorgeschlagenen Systems unter Verwendung aktueller Werkzeuge und Frameworks.

Mit Hilfe des implementierten Prototypen wird das Farbanpassungspotenzial des vorgestellten Architekturentwurfs unter Verwendung aktueller, displaybasierter Testgeräte untersucht. In diesem Zusammenhang findet sowohl eine visuelle als auch eine messtechnische Auswertung der Farbanpassungseigenschaften statt.

Abschließend werden die wichtigsten Ergebnisse der Arbeit zusammengefasst und anhand der Zielsetzung eingeordnet. Zudem wird ein Ausblick auf zukünftige Fragestellungen sowie mögliche Erweiterungspotenziale gegeben.

Inhaltsverzeichnis

„Color management has a beauty and simplicity. It is astounding to see the results from a good color managed system. Once you understand how color management works, you cannot fail to be impressed.“

– Abhay Sharma

Author of „Understanding Color Management“

1 Einleitung

1.1. Motivation und Zielsetzung

Mit der Einführung des Smartphones im Jahr 2007 begann der Aufschwung und die Massentauglichkeit vernetzter, displaybasierter Endgeräte. Zahlreiche Geräteklassen wurden seitdem zur Marktreife gebracht. Hierzu gehören neben Smartphones insbesondere Tablets sowie vernetzte Fernsehgeräte und Armbanduhren. Aktuell stehen unter anderem Geräte aus den Bereichen Automobil und Haushalt im Fokus der Entwicklung.

Im Allgemeinen handelt es sich bei diesen Endgeräten um informationstechnisch aufgerüstete Alltagsgegenstände, die mit einem Benutzer oder anderen Gegenständen interagieren. Üblicherweise bestehen diese Geräte aus anwendungsspezifisch zugeschnittenen, in sich geschlossenen Systemkomponenten.

Eine zentrale Komponente vieler dieser Geräte ist das Display. Es dient als visuelles Ausgabemedium und wird mittels integrierter Berührungsempfindlichkeit bei vielen Endgeräten zudem als Eingabemedium genutzt.

Aufgrund des Erfolgs von Smartphones und Tablets, hat sich das Nutzungsverhalten digitaler Inhalte in den letzten Jahren grundlegend geändert. Diese Geräte erzeugen bereits einen Großteil der Anfragen an Onlinedienste, seien es Nachrichtenportale, Onlineshops oder Mediatheken.

Anbieter dieser Dienste optimieren ihre Inhalte daher zunehmend für vernetzte Geräteklassen. Die zentrale Herausforderung ist in diesem Zusammenhang die Anpassung von Inhalten an eine sehr heterogene Geräte- und Systemlandschaft.

Ein Punkt, der in vielen Anwendungsbereichen von großer Bedeutung ist, ist eine möglichst farbverbindliche Wiedergabe von digitalen Inhalten auf unterschiedlichen Endgeräten. Beispielsweise ist beim Verkauf eines Kleidungsstücks über einen Onlineshop die Übereinstimmung des digital dargestellten Farbtons mit dem physischen Farbton ein entscheidendes Kriterium für Kundenzufriedenheit.

Je nach gerätespezifischen Farbwiedergabeeigenschaften, hauptsächlich bedingt durch die zugrunde liegende Displaytechnologie, kann die Farbwiedergabe verschiedener Endgeräte jedoch stark variieren.

Auf konventionellen Computersystemen wie z.B. Windows PCs oder Macs, wird diese Problemstellung seit einigen Jahren mit Hilfe sogenannter Farbmanagementsysteme kompensiert. Auf kompakten Geräteklässen wird diese Komponente aufgrund des zusätzlichen Ressourcenbedarfs nach heutigem Stand nur in seltenen Fällen eingesetzt.

Aufgrund dessen, widmet sich diese Arbeit dem Entwurf einer neuartigen Architektur für ein Farbmanagementsystem, das sich speziell für die Nutzung mit vernetzten, displaybasierten Endgeräten eignet.

Das Ziel ist die Realisierung eines plattformunabhängigen Farbmanagementsystems, das eine konsistente Farbwiedergabe auf verschiedenen Endgeräten mit einem breiten Spektrum an Gerätemodellen und Betriebssystemen ermöglicht.

Hierzu soll ein abstrakter, technologieunabhängiger Architekturentwurf modelliert werden und dessen technische Realisierungbarkeit durch die Implementierung eines Prototypen sichergestellt werden. Darüber hinaus soll das implementierte System hinsichtlich seiner Farbanpassungseigenschaften untersucht und bewertet werden.

1.2. Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in acht Kapitel. Im Anschluss an die Einleitung folgt die Darstellung einiger technischer Grundlagen, die zum besseren Verständnis der Arbeit erforderlich sind.

Das dritte Kapitel stellt den aktuellen Stand von Architekturansätzen im Bezug auf die Integration von Farbmanagementsystemen dar. Die vorgestellten Ansätze werden hinsichtlich ihres Einsatzzwecks sowie ihrer Vor- und Nachteile eingeordnet.

Im viertel Kapitel erfolgt ein Vorschlag für eine neuartige, serverbasierte Farbmanagementarchitektur, die sich speziell für die Nutzung auf displaybasierten Endgeräten eignet. Hierzu werden die verschiedenen Systemschichten und -komponenten vorgestellt sowie die Interaktion der einzelnen Bestandteile erläutert. Beim Entwurfsprozess wird bewusst auf einen hohen Abstraktionsgrad sowie eine technologieneutrale Modellierung Wert gelegt.

Im fünften Kapitel erfolgt die Beschreibung einer Programmierschnittstelle zur Kopplung des serverbasierten Farbmanagementsystems mit vernetzten Clientgeräten auf Basis aktueller Kommunikationsstandards.

Um die technische Realisierbarkeit des Architekturentwurfs zu gewährleisten, wird im sechsten Kapitel die prototypische Implementierung mit Hilfe aktueller, technischer Werkzeuge vorgestellt.

Unter Verwendung des entwickelten Prototypen, wird im vorletzten Kapitel das Potenzial des vorgestellten Systementwurfs untersucht. In diesem Zusammenhang findet sowohl eine visuelle als auch eine messtechnische Einschätzung der Farbanpassungseigenschaften statt.

Abschließend werden die wichtigsten Ergebnisse der Arbeit zusammengefasst und anhand der Zielsetzung eingeordnet.

2 Technische Grundlagen

2.1. Displaytechnologien

Das Display ist die zentrale, visuelle Komponente displaybasierter Endgeräte und hat maßgeblichen Einfluss auf die Farbdarstellung. Jedes Display besteht aus einer definierten Anzahl an Pixeln, die wiederum, bestehend aus mehreren Subpixeln, in einer Matrix angeordnet werden. Üblicherweise besteht ein Pixel aus einem roten, grünen und blauen Subpixel, aus deren additiver Mischung sich das darstellbare Farbspektrum ergibt. Je nach Hersteller und Modell kommen alternative Ausprägungen von Pixelmatrizen zum Einsatz (vgl. Abb. 2-1).

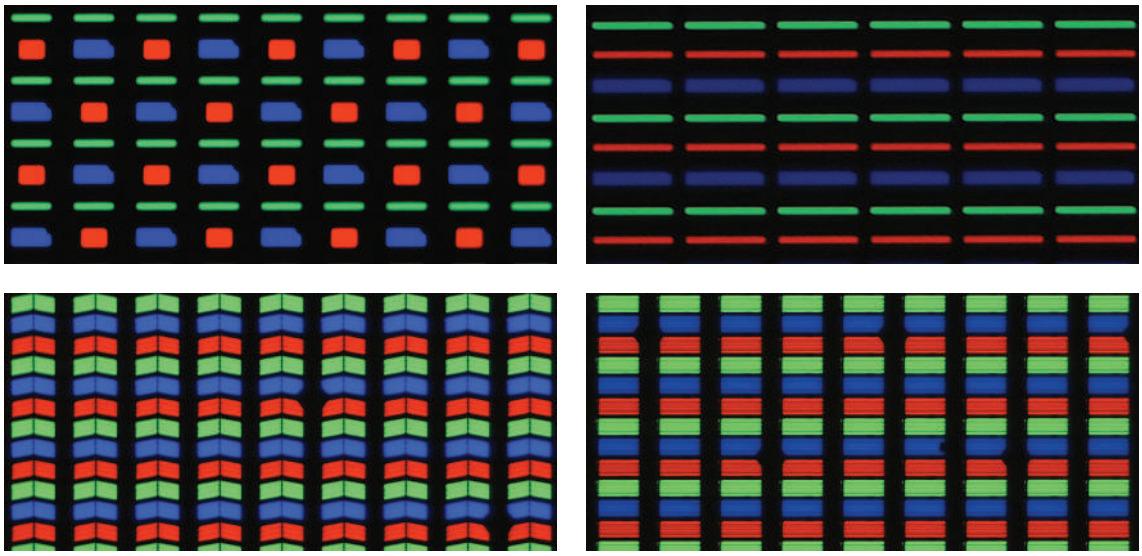


Abbildung 2-1: Mikrokopische Aufnahmen von Pixelmatrizen exemplarischer Smartphone-Displays

Die in aktuellen Geräten vorherrschenden Displaytechnologien lassen sich grob in zwei Kategorien unterteilen: hintergrundbeleuchtete sowie selbstleuchtende Displays. Nachfolgend werden diese beiden Ansätze am Beispiel aktueller Technologieansätze erläutert.

2.1.1. Hintergrundbeleuchtete Displays

Das Prinzip der Hintergrundbeleuchtung kommt heutzutage bei sogenannten Liquid Crystal Displays zum Einsatz. Bei diesen Displays wird Licht durch eine permanente, im Urzustand weiße, Hintergrundbeleuchtung erzeugt. Erst in Kombination mit Farbfiltern wird aus weißem Licht farbiges Licht. Die Intensität der jeweiligen Farbe wird durch eine Schicht von Flüssigkristallen gesteuert. Hierzu wird das Licht der Hintergrundbeleuchtung mit Hilfe von Polarisationsfiltern gezielt ausgerichtet (vgl. Abb. 2-2).

Es existieren verschiedene Technologieansätze von LCDs, die sich vorallem in der Ausrichtung des Lichts durch die Flüssigkristallschicht unterscheiden. Hierzu gehören insbesondere die Technologien *Twisted Nematic*, *Vertical Alignment* und *In Plane Switching*.

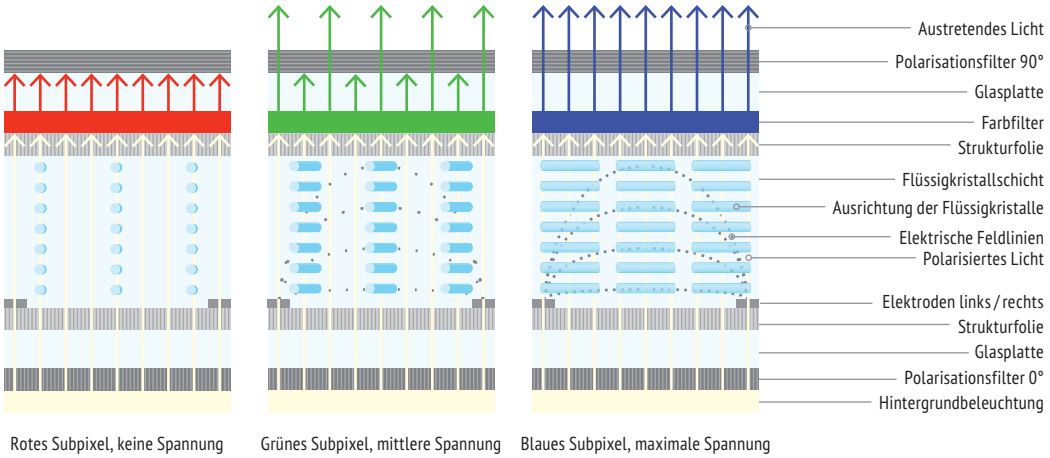


Abbildung 2-2: Schematische Darstellung des Schichtaufbau eines IPS-LCD-Pixels

Entscheidend für die Farbdarstellung eines Liquid Crystal Displays ist das Zusammenspiel aus Hintergrundbeleuchtung und Farbfiltern. Im Folgenden wird der Aufbau einer Hintergrundbeleuchtung sowie deren Interaktion mit dem Filtersystem betrachtet.

Aufgrund ihrer kompakten Bauweise und Energieeffizienz, werden nach heutigem Stand hauptsächlich sogenannte Edge-Lit-Hintergrundbeleuchtungen verwendet (vgl. [Bho08]). Die Lichtquelle liegt hierbei seitlich des Displays und wird mittels Reflektoren in eine Lichtführungsschicht geleitet. Diese Schicht verteilt das Licht gleichmäßig über die gesamte Displaybreite und gleicht bestehende Helligkeitsunterschiede mit Hilfe eines Diffusors aus (vgl. Abb. 2-3).

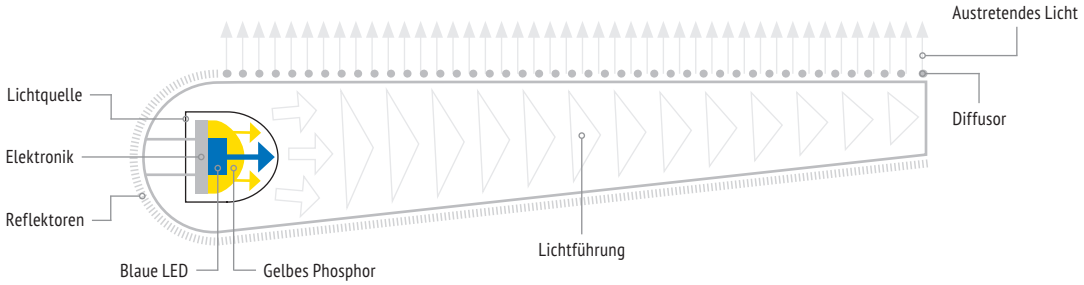


Abbildung 2-3: Technischer Aufbau einer Edge-Lit-Hintergrundbeleuchtung

Relevant für die Farbdarstellung der Hintergrundbeleuchtung ist insbesondere die genutzte Lichtquelle. In aktuellen, displaybasierten Endgeräten kommen hauptsächlich sogenannte Pseudo-White-LEDs zum Einsatz (vgl. [Che11]). Diese besitzen einen blau leuchtenden LED-Chip, welcher von gelb fluoreszierendem Phosphor umgeben ist. Die-

ser wandelt einen Teil des blaufarbenen Lichts in gelbfarbenes Licht um, sodass in der Mischung weißes Licht entsteht. In Kombination mit den Transmissionseigenschaften des zugehörigen Filtersystems entstehen die Primärfarbenspektren des Displays.

Aufgrund der permanenten Hintergrundbeleuchtung erreichen LCDs sehr hohe Leuchtdichten. Da die gesamte Displayfläche beleuchtet wird, scheinen jedoch auch nicht leuchtende Pixel zu geringen Teilen durch. Das macht sich insbesondere bei der Darstellung von schwarzen Flächen bemerkbar.

2.1.2. Selbstleuchtende Displays

Im Gegensatz zu hintergrundbeleuchteten Displays emittieren selbstleuchtende Displays das Licht nicht permanent und ganzflächig, sondern bei Bedarf pro Pixel. Dieser Ansatz wird nach heutigem Stand insbesondere von sogenannten *Organic Light Emitting Diodes* repräsentiert. OLED-Displays nutzen selbstleuchtende, organische Halbleiter zur Farberzeugung (vgl. [NSW14]). Prinzipiell werden gezielt Elektronen (-) und Löcher (+) in eine emittierende Schicht transportiert, um in ihrer Rekombination Energie in Form von Licht auszustrahlen (vgl. Abb. 2-4).

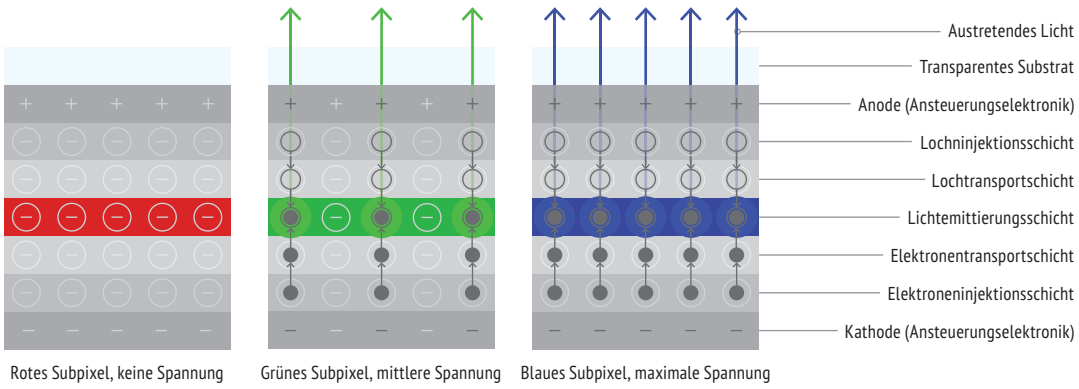


Abbildung 2-4: Schematische Darstellung des Schichtaufbaus eines OLED-Pixels

Da OLED-Displays nicht auf Farbfilter angewiesen sind, weisen sie insbesondere in den grünen und roten Wellenlängen eine deutlich höhere Farbsättigung als LCDs auf. Ebenso können OLED-Displays durch den Verzicht auf eine Hintergrundbeleuchtung reines Schwarz darstellen, erreichen jedoch nicht ganz so hohe Leuchtdichten wie LCDs.

Aus den Erläuterungen geht hervor, dass aktuell verbreitete Displaytechnologien charakteristische Unterschiede hinsichtlich Leuchtdichte, Schwarztiefe und Farbsättigung aufweisen. Für eine konsistente Farbwiedergabe auf unterschiedlichen Displaytypen wird daher zwingend eine kompensierende Instanz in Form eines Farbmanagementsystems benötigt.

2.2. Gerätespezifische Farbeinstellungen

Wie zuvor erläutert, legt das verbaute Displaymodul den Darstellungsspielraum eines displaybasierten Endgeräts fest und nimmt somit maßgeblichen Einfluss auf die Farbdarstellung. Innerhalb der Farbraumgrenzen des Displays, lässt sich die Darstellung wiederum durch gerätespezifische Farbeinstellungen beeinflussen.

In Kombination mit einem Farbmanagementsystem, bezweckt eine Farbeinstellung üblicherweise die Kalibrierung der Farbwiedergabe auf einen Referenzzustand und bildet somit die Grundlage für eine farbgetreue Darstellung.

Da zahlreiche, insbesondere kompakte Endgeräte über kein Farbmanagementsystem verfügen, verfolgen die vorhandenen Farbeinstellungsoptionen dieser Geräte andere Ziele. Je nach Gerätemodell dienen die Einstellungsoptionen entweder zur benutzerdefinierten Farbmodifikation oder zur Emulation standardisierter Wiedergabebedingungen.

In diesem Zusammenhang finden insbesondere die Farbräume sRGB, Adobe RGB (1998) sowie DCI-P3 häufige Anwendung (vgl. [IEC99], [Ado05], [SMP11]). Dabei dient sRGB als Farbraum für die Darstellung allgemeiner, multimedialer Inhalte, Adobe RGB (1998) wird für fotografische Inhalte genutzt, während DCI-P3 insbesondere der Wiedergabe von Filminhalten dient. Abbildung 2-5 vergleicht die Farbumfänge der angesprochenen Farbräume.

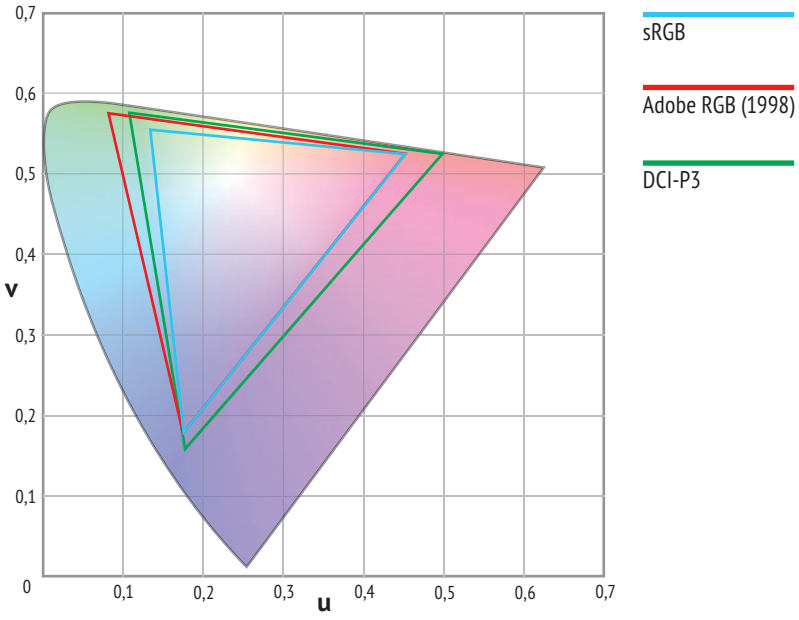


Abbildung 2-5: Farbumfangsvergleich der Farbräume sRGB, Adobe RGB (1998) und DCI-P3

Die Anpassung der Farbdarstellung eines Endgeräts an eine standardisierte Wiedergabebedingung setzt voraus, dass die darzustellenden Inhalte für eben diese Bedingung farblich aufbereitet wurden.

Ein weiterer Parameter ist in diesem Zusammenhang isoliert zu betrachten. Die Helligkeit eines Displays erfüllt auf mobilen Endgeräten wie z.B. Smartphones und Tablets in erster Linie die Aufgabe, die Leuchtstärke des Displays an die aktuellen Umgebungslichtbedingungen anzupassen.

Ein Großteil mobiler Endgeräte verfügt daher über einen Helligkeitssensor, der eine automatische Anpassung der Displayhelligkeit ermöglicht. Dennoch setzen bestimmte Situationen die präzise Kalibrierung der Displayhelligkeit auf einen bestimmten Luminanzwert voraus. Dies ist unter anderem der Fall, wenn Displays verschiedener Endgeräte zu Vergleichszwecken auf eine gemeinsame Leuchtstärke eingestellt werden müssen.

Eine externe Farbmanagementkomponente sollte jederzeit die Kontrolle über die farbliche Einstellung eines Endgeräts besitzen. Außerdem sollte sie das Gerät bei Bedarf in einen gewünschten Farbmodus sowie eine spezifische Leuchtdichte überführen können.

2.3. Farbmanagementsysteme

Im Gegensatz zur statischen Emulation einer bestimmten Wiedergabebedingung, erlauben Farbmanagementsysteme eine dynamische Farbkonvertierung digitaler Inhalte in den spezifischen Farbraum eines Ausgabegeräts.

Farbmanagementsysteme existieren in verschiedenen Spezifikationen. Aufgrund des Verbreitungsgrads wird im weiteren Verlauf dieser Arbeit auf die Version 4.3 des *International Color Consortium* Bezug genommen (vgl. [ICC10]).

Ein Farbmanagementsystem besteht nach den Vorgaben des ICC im Wesentlichen aus einem Farbmanangementmodul sowie entsprechenden Farbprofilen.

Farbmanagementmodule sind für die Umrechnung von Farbwerten im Kontext einer Farbtransformation verantwortlich und besitzen Kenntnis über die benötigten Umrechnungs- und Interpolationsalgorithmen. Ein Farbmanagementsystem besitzt grundsätzlich ein vorgegebenes, systemspezifisches Farbmanagementmodul *(Default CMM),* lässt sich jedoch um Farbmanagementmodule externer Anbieter *(3rd party CMMs)* erweitern.

Farbprofile sind binäre Dateien zur Beschreibung der farblichen Geräteeigenschaften und beinhalten die Parameter für die Transformation von Farbwerten mit Hilfe eines Farbmanagementmoduls.

2.3.1. Farbprofile

ICC-Farbprofile bestehen grundsätzlich aus drei Bestandteilen: einem Profilkopf *(Profile header)*, einer Tag-Tabelle *(Tag table)* sowie den damit verknüpften Tag-Daten *(Tagged element data).* Der Profilkopf beinhaltet beschreibende Informationen über das Farbprofil sowie das damit charaktarisierte Endgerät.

Die Auswahl der in einem Profilkopf vorhandenen Einträge deckt eine Vielzahl möglicher Einsatzzwecke ab. Zu den Informationen gehören beispielsweise das bevorzugte Farbmanagementmodul, Informationen zu Gerätehersteller und -modell sowie die zugrunde liegende Systemplattform.

Im Anschluss an den Profilkopf folgt die Tag-Tabelle. Diese lässt sich mit einem Inhaltsverzeichnis vergleichen, das die im Farbprofil hinterlegten Tags auflistet. Die Tag-Tabelle verweist wiederum auf die eigentlichen Tag-Daten (vgl. Abb. 2-6).

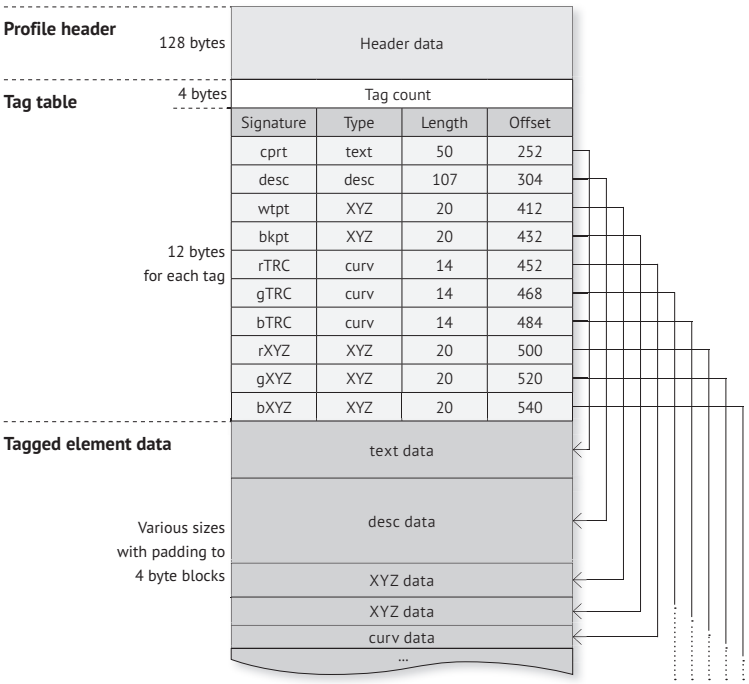


Abbildung 2-6: Aufbau eines ICC-Farbprofils am Beispiel von Adobe RGB (1998) (eigen. Darst. nach [ICC10])

Tags lassen sich in drei Kategorien unterteilen: *required tags*, *optional tags* und *private tags*. Wie die Bezeichnung erahnen lässt, werden erforderliche Tags für den grundlegenden Funktionsumfang vorausgesetzt. Optionale Tags sind hingegen nicht zwingend erforderlich, können jedoch zu einer höheren Qualität der Farbtransformation beitragen. Sowohl erforderliche als auch optionale Tags sind durch die ICC-Spezifikation definiert. Private Tags werden herstellerseitig festgelegt und benötigen daher ein herstellerspezifisches Farbmanagementmodul zur Verarbeitung.

Je nach zugrunde liegender Geräteklasse, kann die Art und Anzahl der erforderlichen und optionalen Tags variieren. Im weiteren Zusammenhang werden insbesondere Displayprofile sowie allgemeine Farbraumprofile thematisiert. Displayprofile beschreiben die Farbwiedergabecharakteristik von Displays bzw. Monitoren, bei allgemeinen Farbraumprofilen handelt es sich hingegen um geräteunabhängige Farbräume oder Wiedergabebedingungen wie beispielsweise sRGB, Adobe RGB (1998) oder DCI-P3.

2.3.2. Farbtransformationen

Eine Farbtransformation bezeichnet den Vorgang, bei dem Farbwerte aus einem Quellfarbraum in einen Zielfarbraum konvertiert werden. Für diese Umrechnung werden diverse Parameter benötigt. Hierzu gehören unter anderem der umzurechnende Farbort im Quellfarbraum, das Farbprofil des Quellfarbraums, das Farbprofil des Zielfarbraums sowie die Angabe des Umrechnungsverfahrens in Form eines sogenannten Rendering Intents (vgl. Tab. 2-1).

Nr.	Bezeichnung	Wiedergabeabsicht
0	Perzeptiv	Wahrnehmungsorientiert, Erhaltung der relativen Farbabstände
1	Relativ Farbmetrisch	Farbmetrisch genaue Umrechnung mit Anpassung des Weißpunkts
2	Sättigungserhaltend	Maximierung der Farbsättigung
3	Absolut Farbmetrisch	Farbmetrisch genaue Umrechnung ohne Anpassung des Weißpunkts

Tabelle 2-1: Rendering Intents der ICC-Spezifikation (vgl. [ICC10])

Das relativ farbmetrische Rendering Intent lässt sich darüber hinaus mit einer sogenannten Tiefenkompensierung kombinieren. Das Verfahren sorgt dafür, dass der Schwarzpunkt des Quellfarbraums bei der Farbkonvertierung an den Schwarzpunkt des Zielfarbraums angeglichen wird. Dadurch lässt sich die Zeichnung in dunklen Bildbereichen erhalten.

2.4. Kopplungstechnologien

Für die Umsetzung einer serverbasierten Farbmanagementarchitektur wird eine Kopplungstechnologie benötigt, die Server und Clients miteinander verbindet. Die Kommunikation zwischen diesen Parteien wird mit Hilfe eines Protokolls definiert, das Syntax, Semantik und Synchronisation des Nachrichtenaustauschs festlegt.

Serverbasierte Architekturen, deren Kopplung auf Basis von Webtechnologien erfolgt, werden als Webservice bezeichnet. Als konkrete Umsetzungen von Webservices sind insbesondere die Technologien *Representational State Transfer* und *Simple Object Access Protocol* zu nennen.

Bei SOAP erfolgt der Nachrichtenaustausch auf Basis von XML-Dokumenten, deren jeweiliger Aufbau mit Hilfe einer dedizierten Beschreibungssprache definiert wird. Die Kommunikation über eine REST-basierte Schnittstelle ist hingegen deutlich leichtgewichtiger und bietet mehr Flexibilität im Bezug auf die Auswahl des Austauschformats (vgl. [Til11]).

Ein zentrales Konzept des REST-Ansatzes ist die Abbildung von Inhalten in Form von Ressourcen. Dies geschieht mit Hilfe sogenannter *Uniform Resource Identifier*. Jeder URI folgt einem vorgegebenen Definitionsschema, welches sich in insgesamt fünf Elemente gliedert: *Scheme, Authority, Path, Query* und *Fragment* (vgl. Abb. 2-7).

Abbildung 2-7: Elemente eines Uniform Resource Identifiers (vgl. [Bel15])

Ein URI beinhaltet demnach das zugrunde liegende Kommunikationsprotokoll *(Scheme)*, die Domain und den Port des Host *(Authority)* sowie den Ressourcenpfad *(Path)*. Der Pfad lässt sich um sogenannte Query- und Fragment-Parameter erweitern.

Jeder Ressourcenpfad besteht aus mehreren, durch einen Schrägstrich getrennten Parametern, die durch ihre Reihenfolge einen hierarchischen Zusammenhang abbilden. Query-Parameter werden insbesondere für den Verweis auf spezifische Ressourceninhalte genutzt, während Fragment-Parameter auf einen spezifizierten Inhaltsabschnitt verweisen.

2.4.1. Kommunikationsprotokolle

Wie bereits erläutert, wird die Kommunikation im Rahmen einer serverbasierten Architektur mit Hilfe eines Kommunikationsprotokolls definiert. Webbasierte Kopplungstechnologien nutzen hierfür hauptsächlich das *Hypertext Transfer Protocol*.

HTTP liegt aktuell in den drei Protokollspezifikationen HTTP/1, HTTP/1.1 und HTTP/2 vor. Aufgrund des momentanen Verbreitungsgrads innerhalb von Entwicklungswerkzeugen, bezieht sich die Arbeit im weiterem Verlauf auf die Version HTTP/1.1.

Jede Interaktion über HTTP besteht aus einer Anfrage des Clients an den Server *(Request)* sowie einer Antwort des Servers an den Client *(Response)*. Jede dieser Interaktionen gliedert sich wiederum in einen Nachrichtenkopf und einen Nachrichtenrumpf. Der Nachrichtenkopf wird durch eine Anfragezeile eingeleitet und kann weitere, die Anfrage beschreibende Parameter beinhalten (vgl. List. 2-1).

```
1 GET /path/to/resource HTTP/1.1
2 Host: www.example.com
```

Listing 2-1: Exemplarische Anfrage in HTTP/1.1

Die Anfragezeile eines HTTP-Requests besteht grundsätzlich aus einer Kombination von Methode, URI und Protokollversion. Jede HTTP-Version besitzt eine konstante Menge an Methoden, die unter anderem dazu dienen Ressourcen auszulesen, hinzuzufügen, zu aktualisieren oder zu löschen. Die folgende Tabelle stellt den Methodensatz von HTTP/1.1 und deren jeweiligen Verwendungszweck dar.

Methode	Beschreibung
`GET`	Fordert eine Ressource an
`POST`	Erstellt eine neue Ressource
`PUT`	Ersetzt eine vorhandene Ressource
`PATCH`	Ersetzt Teile einer vorhandenen Ressource
`DELETE`	Entfernt eine vorhandene Ressource
`HEAD`	Fordert den Nachrichtenkopf einer Ressource an
`TRACE`	Liefert eine Anfrage exakt so zurück, wie sie gestellt wurde
`OPTIONS`	Fordert eine Liste aller unterstützen Methoden an
`CONNECT`	Stellt eine verschlüsselte Verbindung her

Tabelle 2-2: Übersicht der Methoden von HTTP/1.1 (vgl. [Bel15])

Verarbeitete Anfragen werden seitens des Servers mit einem Statuscode beantwortet und signalisiert, ob eine Anfrage erfolgreich war oder zu einem Fehler geführt hat. Der Anfragekopf wird zudem um eine kurze Beschreibung des Statuscodes sowie die zugrunde liegende Protokollversion ergänzt (vgl. List. 2-2, Zeile 1). Eine vollständige Liste aller Statuscodes von HTTP/1.1 sowie der jeweiligen Bedeutung ist in Anhang A ersichtlich.

```
1 HTTP/1.1 200 OK
2 Content-Length: 51
3 Content-Type: application/json
4 {
5    "content": "This is an examplary response"
6 }
```

Listing 2-2: Exemplarische Antwort in HTTP/1.1

Angeforderte Inhalte werden im Nachrichtenrumpf der Antwort unter Angabe von Inhaltslänge und -typ übertragen. Inhaltstypen werden bei einer HTTP-basierten Kommunikation mit Hilfe des MIME-Standards *(Multipurpose Internet Mail Extension)* kategorisiert (vgl. Tab. 2-3).

Kategorie	Beschreibung	Beispiel
`text`	Textdateien	`text/html`
`image`	Grafik- und Bilddateien	`image/jpeg`
`video`	Videodateien	`video/mpeg`
`audio`	Audiodateien	`audio/x-wav`
`application`	Programmgebundene Dateien	`application/vnd.iccprofile`
`multipart`	Mehrteilige Daten	`multipart/form-data`
`message`	Nachrichten	`message/http`
`model`	Mehrdimensionale Strukturen	`model/vrml`

Tabelle 2-3: Kategorien von MIME-Typen (vgl. [Fre96])

2.4.2. Austauschformate

Für den textbasierten Informationsaustausch werden üblicherweise strukturierte Datenformate eingesetzt. Aufgrund ihrer großen Verbreitung sind in diesem Zusammenhang insbesondere die Formate JSON und XML zu nennen. Anhand eines Beispiels werden die markanten Unterschiede dieser beiden Formate veranschaulicht. Die hierfür dargestellten Listings 2-3 und 2-4 bilden die Beschreibung des Medienweißpunkts eines ICC-Farbprofils mit den beiden genannten Formaten ab.

```
1 {
2    "wtpt": {
3       "description": "Media white point XYZ values",
4       "type": "XYZ",
5       "content": [0.9504547, 1, 1.0890503]
6    }
7 }
```

Listing 2-3: Beschreibung eines Medienweißpunkts mit Hilfe eines JSON-Dokuments

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <wtpt>
3     <description>Media white point XYZ values</description>
4     <type>XYZ</type>
5     <content>
6        <item>0.9504547</item>
7        <item>1</item>
8        <item>1.0890503</item>
9     </content>
10 </wtpt>
```

Listing 2-4: Beschreibung eines Medienweißpunkts mit Hilfe eines XML-Dokuments

Die dargestellten JSON- und XML-Dokumente besitzen eine Zeichenanzahl in Höhe von 132 bzw. 230 Zeichen. In 8-bit kodierter Form entsprechen diese Zahlen zudem der Dokumentengröße in Bytes.

JSON-Dokumente sind insgesamt leichtgewichtiger als XML-Dokumente und lassen sich daher schneller übertragen. Dies liegt insbesondere am unterschiedlichen Aufbau von Listen, die bei JSON-Dokumenten deutlich weniger Overhead erzeugen (vgl. List. 2-2 Zeile 5 sowie List. 2-3 Zeile 6-8).

Aus diesem Grund werden JSON-Strukturen für den einfachen Datentransfer im Web und insbesondere für die Verwendung im Zusammenhang mit REST-basierten Schnittstellen bevorzugt.

3 Existierende Systemansätze

Die Anbindung eines Farbmanagementsystems erfolgt grundsätzlich über eine Schnittstelle, die einen definierten Funktionsumfang zur Verfügung stellt und die Zugriffsregeln für ausführende Applikationen bzw. für Grafik- und Bildbibliotheken festlegt. Eingehende Anfragen werden von der Schnittstelle an die entsprechenden Module des Farbmanagementsystems weitergeleitet und verarbeitet (vgl. Abb. 3-1).

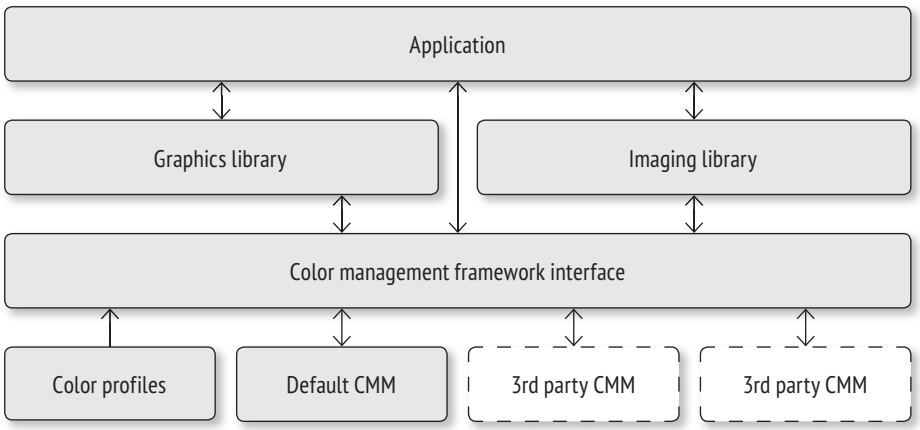


Abbildung 3-1: Allgemeines Architekturschaubild von ICC-basierten Farbmanagementsystemen (eigen. Darst. nach [ICC10])

Die Verteilung und Ausprägung der dargestellen Komponenten kann je nach Architekturansatz variieren. Nachfolgend werden verschiedene Szenarien für die Integration eines Farbmanagementsystems vorgestellt und im Hinblick auf ihren Einsatzzweck eingeordnet.

3.1. Betriebssystemseitiger Ansatz

Bei der Integration eines Farbmanagementsystems auf Betriebssystemebene handelt es sich um die ursprüngliche Form der Anbindung. In diesem Fall steht das Farbmanagementsystem allen Applikationen der Plattform zur Verfügung. Darüber hinaus sind die systemseitigen Grafik- und Bildbibliotheken bereits über die benötigten Schnittstellenfunktionen mit dem Farbmanagementsystem verbunden (vgl. Abb. 3-2).

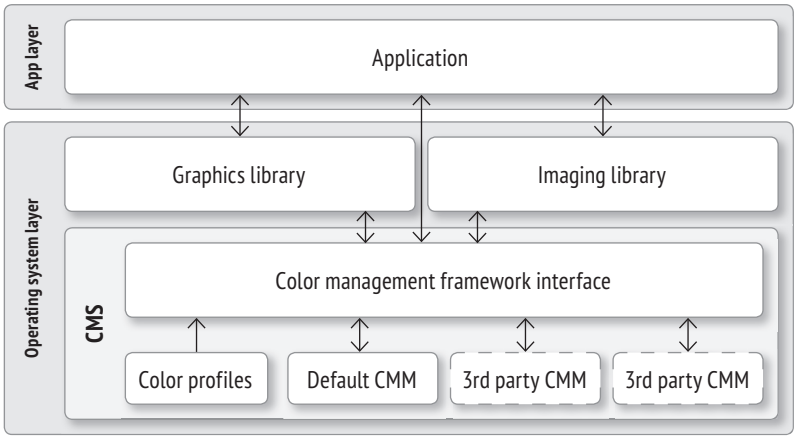


Abbildung 3-2: Integration eines Farbmanagementsystems auf Betriebssystemebene

Der grundlegende Vorteil dieses Ansatzes ist einerseits, dass die Bereitstellung von Farbanpassungsfunktionen nicht dem Anwendungsentwickler obliegt. Über die Schnittstelle des Farbmanagementsystems können die bestehenden Farbkalibrierungs- und transformationsfunktionen applikationsübergreifend genutzt werden. Weiterhin profitiert die Ausführung eines Farbmanagementsystems auf Betriebssystemebene von der direkten Unterstützung des Grafikprozessors.

Grundsätzlich beansprucht die systemseitige Ausführung eines Farbmanagementsystems dauerhaft Rechen-, Speicher- und Energieressourcen. Daher eignet sich dieser Ansatz nur für Endgeräte, bei denen diese Ressourcen in ausreichender Kapazität zur Verfügung stehen.

Systemseitige Farbmanagementsysteme werden seit vielen Jahren auf konventionellen Betriebssystemen (z.B. in Mac OS bzw. Windows) eingesetzt. In Plattformen für mobile Endgeräte wurde zugunsten der Akkulaufzeit viele Jahre auf den Einsatz von Farbmanagementsystemen verzichtet. Mittlerweile verfügen zumindest die Systeme iOS (seit 03/2016, Version 9.3) bzw. Android (seit 11/2017, Version 8.0) über entsprechende Funktionen (vgl. [App16], [Ope17]).

3.2. Applikationsinterner Ansatz

Bei der applikationsinternen Einbindung eines Farbmanagementsystems sind die entsprechenden Komponenten integraler Bestandteil der Anwendung. Durch diesen Ansatz ist die Applikation nicht auf das Farbmanagementsystem des ausführenden Betriebssystems angewiesen, kann systemseitig vorhandene Farbmanagementkomponenten jedoch bei Bedarf nutzen (vgl. Abb. 3-3).

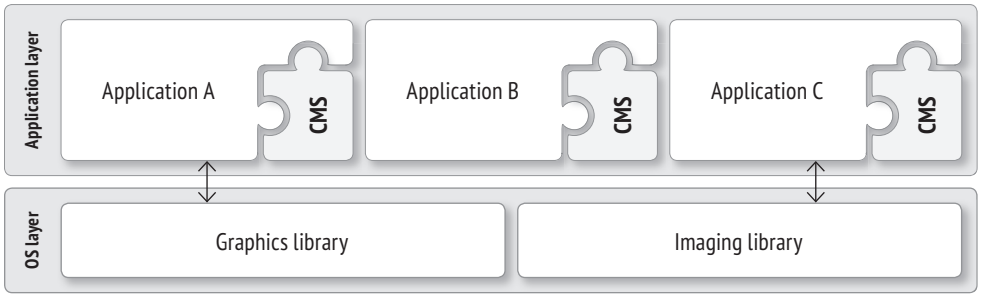


Abbildung 3-3: Applikationsinterne Integration eines Farbmanagementsystems

Dieser Ansatz hat zwei zentrale Vorteile: Einerseits ist die betreffende Applikation nicht auf ein systemseitiges Farbmanagementsystem angewiesen, andererseits kann die Applikation sicherstellen, dass systemübergreifend identische Farbmanagementfunktionen zur Verfügung stehen.

Nachteilig ist hingegen, dass das integrierte Farbmanagementsystem zusätzlichen Speicherplatz beansprucht und eine Aktualisierung des Farbmanagementsystems ebenfalls eine Aktualisierung aller Applikationen voraussetzt, die dieses System integrieren. Eine zentrale Verwaltung wie beispielsweise auf Betriebssystemebene ermöglicht dieser Ansatz nicht.

Ein häufiger Einsatzzweck für applikationsinternes Farbmanagement sind Webbrowser. Beispielsweise setzt Mozilla Firefox einen solchen Ansatz ein, um entsprechende Funktionen auch auf Betriebssystemen ohne Farbmanagementsystem (hierzu gehören insbesondere linuxbasierte Plattformen) nutzen zu können.

Weiterhin werden applikationsinterne Farbmanagementsysteme häufig in Grafik- und Bildbearbeitungsprogrammen eingesetzt. Beispielsweise integriert Adobe das proprietäre Farbmanagementsystem *Adobe Color Engine* in zahlreiche Anwendungen, um systemübergreifend eine konsistente Farbkonvertierung und -wiedergabe zu gewährleisten.

3.3. Serverbasierter Ansatz

Beim serverbasierten Systemansatz wird das Farbmanagementsystem als externe Komponente zur Verfügung gestellt und über ein Protokoll mit ausführenden Clients verbunden. Der schematische Aufbau eines Farbmanagementsystems als serverbasierte Architektur wird in Abbildung 3-4 dargestellt. Dieses Szenario schließt weder die zusätzliche Verwendung eines betriebssystemsseitigen, noch eines applikationsinternen Farbmanagementsystems aus.

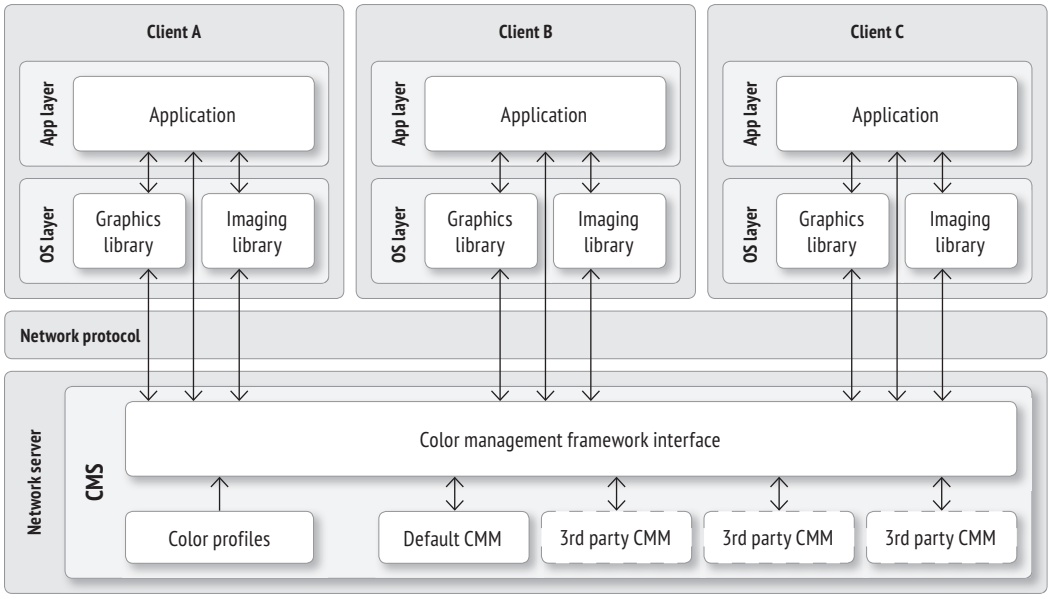


Abbildung 3-4: Integration eines Farbmanagementsystems als serverbasierte Architektur

Die dargestellte Architekturvariante erlaubt eine applikations- und systemübergreifende Einbindung des Farbmanagementsystems und kombiniert somit grundlegende Vorteile der zuvor genannten Ansätze. Hinzu kommt, dass sich das Farbmanagementsystem (analog zum betriebssystemseitigen Ansatz) an zentraler Stelle verwalten und aktualisieren lässt.

Weiterhin werden ausführende Clientgeräte durch die Auslagerung des Farbmanagementsystems hinsichtlich Speicherplatz, Rechenleistung und Energieverbrauch entlastet. Dies begünstigt insbesondere mobile und rechenschwache Endgeräte wie z.B. Smartphones und Tablets.

Grundsätzlich birgt dieser Ansatz, insbesondere durch die verteilte Architektur, zusätzliche Herausforderungen. Beispielsweise setzt die Ausführung eines solchen Farbmanagementsystems eine Netzwerk- bzw. Internetverbindung voraus, die entsprechend der zu übertragenden Datenmengen ausreichend breitbandig sein sollte. Die verteilte Architektur erfordert zudem geeignete Autorisierungs- und Sicherheitsmechanismen.

Eine weitere, zentrale Anforderung an eine solche Architektur ist ein erforderliches Maß an Flexibilität, um die Anbindung an eine enorm heterogene Geräte- und Systemlandschaft zu ermöglichen.

Es existieren diverse, kommerzielle Ausprägungen dieses Architekturansatzes. Das Patent *„Web enabled color management service system and method“* von Xerox beschreibt ein serverbasiertes Farbmanagementsystem für printbasierte Ausgabegeräte (vgl. [Kes09]).

Die Anbindung an das Farbmanagementsystem erfolgt über eine Verbindungsschicht, die unterschiedliche Kommunikations- und Sicherheitsprotokolle berücksichtigt. Dabei kommen insbesondere internetbasierte Protokolle zum Einsatz.

Der Ansatz setzt eine Authentifizierung von Systemnutzern voraus, die über ein zentrales Datenbanksystem verwaltet werden. Es wird zwischen sogenannten *Basic Users* und *Color Critical Users* unterschieden, denen unterschiedliche Konfigurationsoptionen zur Verfügung stehen. Farbkritische Nutzer haben beispielsweise die Möglichkeit benutzerdefinierte Farbprofile anzulegen, um inbesondere druckspezifische Parameter wie Unbuntaufbau oder maximalen Farbauftrag anzupassen.

Aufgrund der starren Ausrichtung auf printbasierte Ausgabegeräte bleiben wesentliche Anforderungen displaybasierter Endgeräte jedoch unberücksichtigt. Ein wesentlicher Kritikpunkt ist, dass es sich bei diesem Ansatz um einen weitestgehend manuellen, benutzergetriebenen Prozess handelt, der nicht für eine, wie bei displaybasierten Endgeräten üblich, hintergrundbasierte Ausführung ausgelegt ist.

Ein weiterer, serverbasierter Ansatz lautet *„Method and system for improved internet color“* (vgl. [Hil13]). Das beschriebene System bezweckt eine Verbesserung der Farbtreue in internetbasierten Anwendungen und sieht hierzu die Ausgabe auf displaybasierten Endgeräten vor. Es wird inbesondere auf den konkreten Anwendungsfall „Onlineshopping im Webbrowser“ hingewiesen. Aufgrund der direkten Bezugnahme auf Webbrowser, berücksichtigt das System neben Bildern unter anderem bildreferenzierende Dokumente wie z.B. HTML und XML. Als Kommunikationsprotokoll liegt HTTP zugrunde.

Das Farbmanagementsystem bietet die Möglichkeit, generische Ausgabefarbprofile (wie z.B. sRGB) zu nutzen, oder eigene, gerätespezifische Farbprofile zu erstellen. Hierzu wird eine systemseitige, netzwerkbasierte Kalibrierung und Charakterisierung des ausgebenden Displays angeboten. Dies setzt jedoch einen mehrminütigen, interaktiven Prozess sowie das Vorhandensein geeigneter, farbmesstechnischer Geräte voraus. Ein systemseitiges Angebot an modellspezifischen Farbprofilen, die sich anhand der verwendeten Hardware automatisiert ermitteln lassen und einen qualitativen Kompromiss zwischen generischen und benutzerspezifischen Profilen darstellen, sieht der Ansatz nicht vor.

Der aktuellste, serverbasierte Ansatz lautet *„Color management for web server based applications“* (vgl. [Chu15]). Wie im zuletzt genannten Beispiel, dient ein Farbmanagementserver als zentrale Instanz, um digitale Inhalte für eine konsistente Farbwiedergabe auf displaybasierten Endgeräten aufzubereiten. Der Ansatz sieht inbesondere die Verwendung mobiler Endgeräte wie z.B. Smartphones und Tablets vor (vgl. Abb. 3-5).

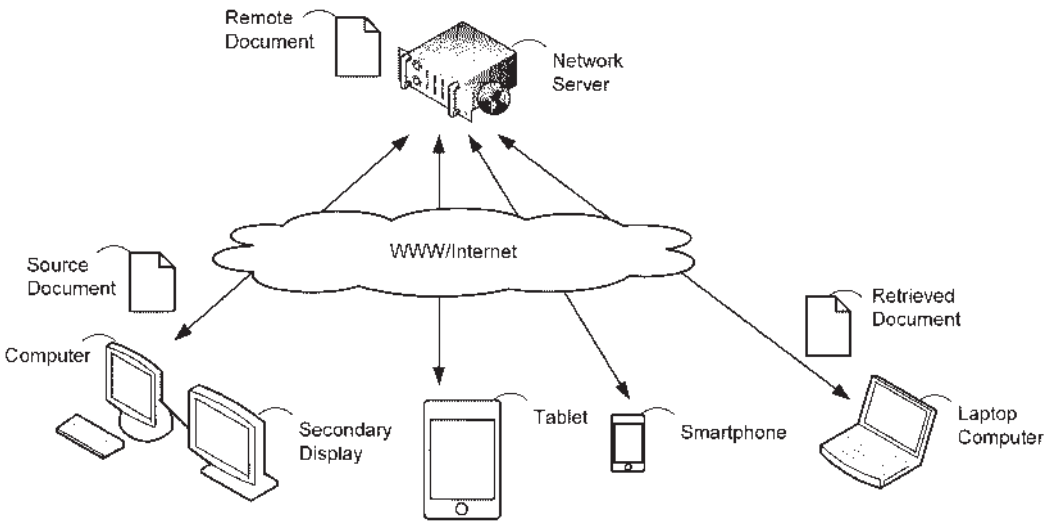


Abbildung 3-5: Architekturschaubild des Ansatzes nach [Chu15]

Der Ansatz bezieht sich allgemein auf die Übertragung von seitenbasierten Dokumenten, die aus einzelnen Seitenelementen bestehen und Farbdaten wie z.B. Bilder entweder referenzieren oder einbetten können. Die im Ansatz beschriebenen Anwendungsfälle beziehen sich wiederum gezielt auf die Darstellung von Farbinhalten in Webbrowsern.

Je nach Übertragungs- bzw. Konvertierungsstatus werden die Dokumente als *Source Document*, *Remote Document* bzw. *Retrieved Document* bezeichnet. Es wird ausdrücklich darauf hingewiesen, dass Quell- und Zielort eines Dokuments nicht zwingend identisch sein müssen. Ein Dokument kann somit auch von einer dritten Instanz angefordert und für die Darstellung auf einem spezifischen Wiedergabegerät aufbereitet werden.

Der serverseitige Prozess wird grob in drei Module gegliedert: *Receiving Module*, *Processing Module* und *Transmitting Module*. Diese Module nehmen Dokumente entgegen, verarbeiten sie und übertragen sie wiederum an einen Client (vgl. Abb. 3-6).

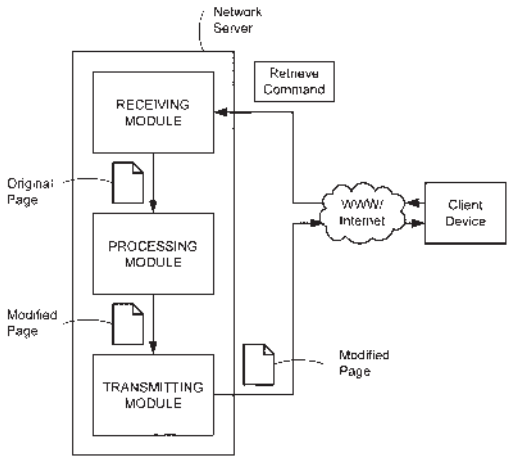


Abbildung 3-6: Modularisierung und Prozessfluss des Ansatzes nach [Chu15]

Weiterhin beschreibt der Ansatz eine serverseitige Prozesslogik, die spezifische Farbeigenschaften darstellender Endgeräte erfasst und den Prozessablauf anhand dieser Informationen anpasst. Bei bereits vorhandenem, clientseitigem Farbmanagement lässt sich eine serverseitige Farbumwandlung deaktivieren.

Die Gliederung in die Komponenten *Receiving*, *Processing* und *Transmitting* lässt leider nur allgemeine Rückschlüsse auf das jeweils zugehörige Aufgabengebiet zu. Darüber hinaus wird die Kalibrierung von wiedergebenden Endgeräten bei diesem Ansatz nicht berücksichtigt.

Alle vorgestellten Ansätze unterscheiden Farbprofile hinsichtlich ihrer Qualität entweder als benutzerspezifisch angepasst oder vollständig generisch. Profile, die einen qualitativen Kompromiss zwischen diesen beiden Kategorien darstellen (wie z.B. gerätemodellspezifische Farbprofile), werden nicht vorgeschlagen.

Obwohl es die allgemeine Architektur des ICC vorschlägt, ist keiner der vorgestellten, serverbasierten Ansätze ausreichend flexibel, um benutzerdefinierte Farbmanagementmodule hinzuzufügen. Benutzer sind daher stets an vorhandene Farbmanagementmodule und die von ihnen unterstützten Farbprofilspezifikationen gebunden.

Da alle existierenden Architekturansätze mit Nachteilen für den Nutzer bzw. das nutzende System behaftet sind, befasst sich diese Arbeit mit dem Entwurf eines neuartigen Lösungsansatzes, der Ideen bestehender Ansätze aufgreift, erweitert und versucht existierende Systemeinschränkungen zu minimieren.

4 Architekturvorschlag

In diesem Kapitel wird ein neuartiger Architekturansatz für ein serverseitiges Farbmanagementsystem vorgeschlagen. Die Architekturbeschreibung legt den Fokus auf die Verwendung mit displaybasierten Endgeräten. Grundsätzlich ist der modellierte Entwurf jedoch ausreichend flexibel, um auch von anderen, nicht-displaybasierten Ausgabemedien genutzt zu werden.

Nachfolgend werden die einzelnen Schichten des Entwurfs, die darin enthaltenen Komponenten sowie deren Zusammenspiel vorgestellt und erläutert. Ein wesentlicher Schwerpunkt liegt auf einer möglichst technologieunabhängigen Beschreibung sowie einem hohen Grad an Abstraktion.

Im folgenden Schaubild wird der Architekturansatz schematisch dargestellt und ein Überblick über die beteiligten Akteure, Komponenten und deren Kommunikationswege gegeben (vgl. Abb. 4-1).

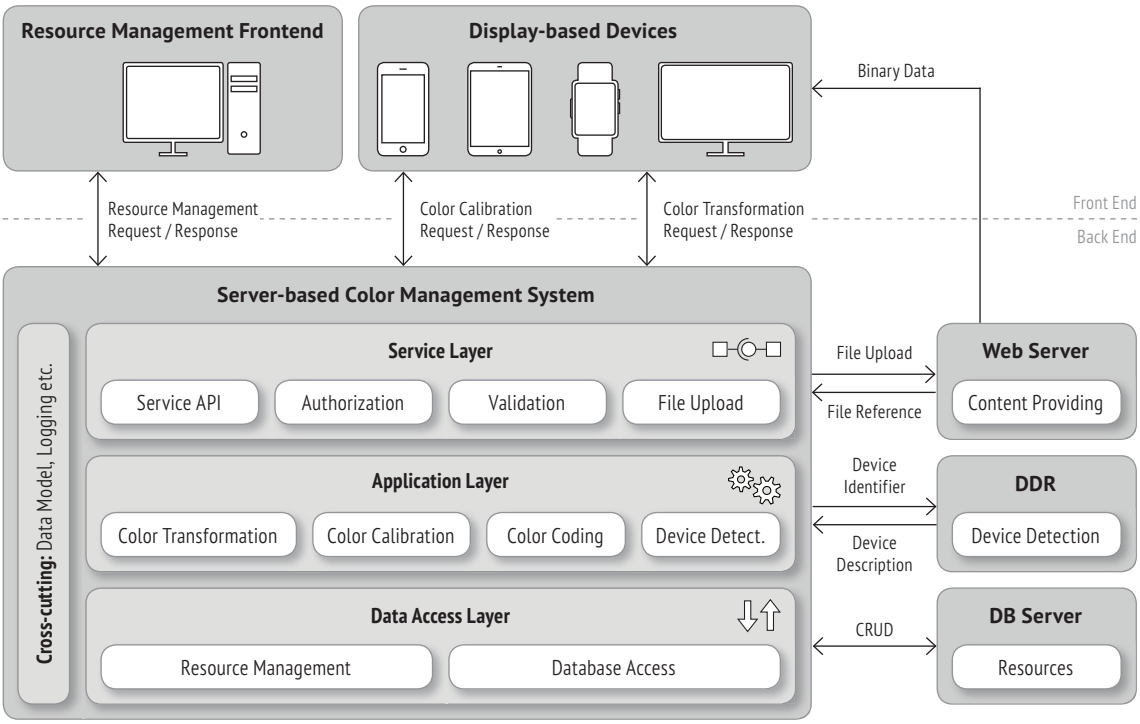


Abbildung 4-1: Schaubild des vorgeschlagenen Architekturansatzes für ein serverbasiertes Farbmanagementsystem

Die Abbildung gliedert sich in Frontend und Backend. Das Frontend dient zur visuellen Präsentation der vom Farbmanagementsystem zur Verfügung gestellten Inhalte. Hierzu gehören in erster Linie Endgeräte, die Farbkalibrierungseinstellungen anwenden sowie

farbtransformierte Inhalte wie z.B. Bilder und Farbflächen abbilden. Weiterhin dienen insbesondere konventionelle Rechner zur Darstellung einer grafischen Benutzeroberfläche für die Verwaltung von Systemressourcen.

Das Backend besteht aus drei Schichten. Die oberste Schicht, die sogenannte Serviceschicht, dient zur Kommunikation mit dem Frontend. Sie stellt die Funktionen des Farbmanagementsystems über eine API zur Verfügung und prüft eingehende Anfragen hinsichtlich Benutzerrechten und Datenvalidität.

Valide Anfragen werden an die Anwendungsschicht weitergeleitet. Diese Schicht bildet die Prozesslogik ab und repräsentiert daher den zentralen Aufgabenbereich des Farbmanagementsystems. Es existiert jeweils eine Komponente für die Prozesse Farbkalibrierung und -transformation sowie damit verbundene Hilfskomponenten zur Geräteerkennung bzw. zur De- und Enkodierung von eingehenden Farbdaten und Dokumentenformaten.

Die von der Anwendungsschicht benötigten, persistenten Ressourcen werden über die Datenzugriffsschicht abgerufen und verwaltet. Sie verknüpft das Farbmanagementsystem darüber hinaus mit zugrunde liegenden, physischen Datenbanken und gewährleistet somit eine Entkopplung zwischen Anwendungsdomäne und Datenhaltung.

Einige Komponenten des Farbmanagementsystems lassen sich keiner expliziten Schicht zuordnen. Hierzu gehören neben den klassischen schichtübergreifenden Aspekten wie z.B. dem Logging insbesondere das zentrale, objektorientierte Datenmodell des Farbmanagementsystems. Das Datenmodell, die angesprochenen Systemschichten sowie deren beinhaltete Komponenten werden in den nachfolgenden Abschnitten erläutert.

4.1. Datenmodell

Das Datenmodell beschreibt die im Farbmanagementsystem abgebildeten Ressourcen. Es unterscheidet zwischen persistenten und nicht-persistenten Ressourcen. Zu den persistenten, also den im System gespeicherten Ressourcen, gehören neben Farbprofilen ebenso Informationen zur Beschreibung von Endgeräten, Betriebssystemen und Farbmanagementmodulen. Eingehende Farbpixel und -transformationseinstellungen stellen nicht-persistente Ressourcen dar.

Das Datenmodell basiert auf der abstrakten Klasse *Resource*, die den grundlegenden Typen zur Definition einer Ressource angibt. Von dieser Klasse erben wiederum die abstrakten Klassen *PersistentResource* und *NonPersistentResource*.

Die Klasse *PersistentResource* definiert die beiden Attribute *id* und *description*. Hierbei handelt es sich um eine eindeutige Kennung, über die Ressourcen im System zu identifizieren sind, sowie eine allgemeine, textbasierte Beschreibung.

Von der Klasse *PersistentResource* erben die vier konkreten Typen *IccProfile*, *Cmm*, *Device* sowie *OpSystem*. Hierbei handelt es sich um eine objektorientierte Beschreibung von Farbprofilen, Farbmanagementmodulen, Endgeräten sowie deren zugrunde liegenden Betriebssystemen.

Die Klasse *NonPersistentResource* vererbt an die Klassen *Pixel* und *TSettings* zur Beschreibung von Farbwerten und -transformationseinstellungen. Ressourcenobjekte werden grundsätzlich mittels Fabrikmethode erzeugt.

Die Klassenhierarchie des Datenmodells sowie der sequenzielle Ablauf zum Erstellen von Ressourcen werden in den Abbildungen 4-2 sowie 4-3 dargestellt. Der detaillierte Aufbau der genannten, konkreten Ressourcentypen wird nachfolgend genauer beschrieben.

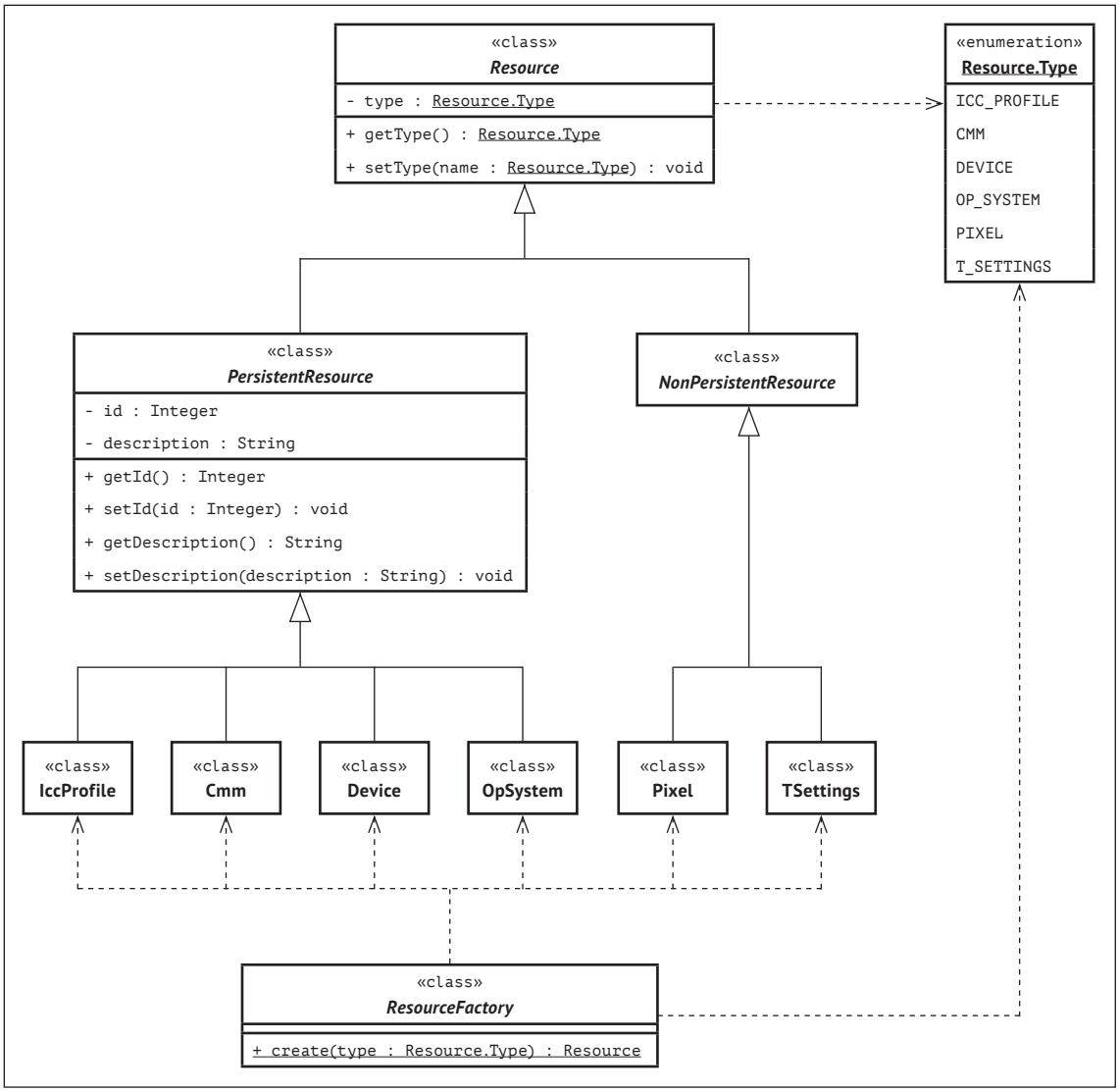


Abbildung 4-2: Klassendiagramm des Datenmodells

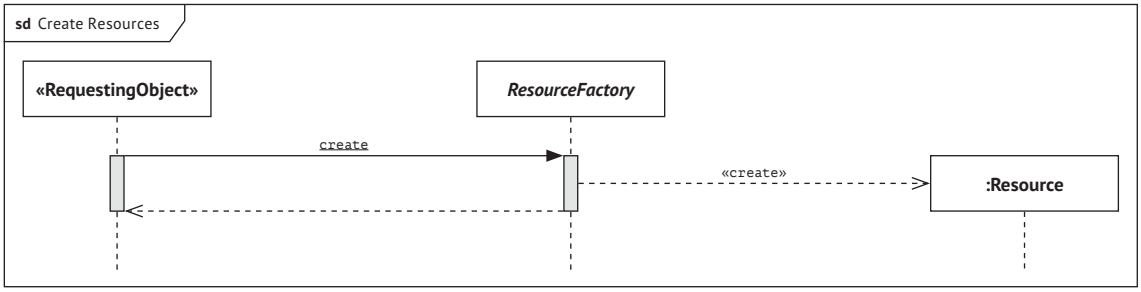


Abbildung 4-3: Sequenzdiagramm für das Erzeugen von Ressourcen

4.1.1. Endgeräte

Die im Datenmodell abgebildeten Endgeräte werden mit Hilfe der Klasse *Device* beschrieben. Neben den beiden geerbten Einträgen *id* und *description*, verfügt ein Endgerät über folgende, weitere Attribute (vgl. Tab. 4-1).

Attribut	Datentyp	Beschreibung
`manufacturer`	`String`	Angabe des Geräteherstellers
`model`	`String`	Angabe des Gerätemodells
`type`	`Enumeration`	Geräteklasse des Endgeräts, z.B. Smartphone oder Tablet
`category`	`Enumeration`	Kategorisierung in spezifische bzw. generische Endgeräte
`ddrIdMap`	`Map<String, String>`	Verknüpfung mit verbundenen Device Detection Repositories

Tabelle 4-1: Attribute von Endgeräten

Die Attribute *manufacturer* und *model* geben den Hersteller sowie das Modell des Endgeräts an. Darüber hinaus besitzt jedes hinterlegte Gerät eine Beschreibung der Geräteklasse (z.B. Smartphone oder Tablet), um eine diesbezügliche Sortierung der im System hinterlegten Endgeräte zu gewährleisten.

Weiterhin werden Endgeräte in spezifische und generische Geräte kategorisiert. Ein spezifisches Endgerät beschreibt ein konkretes Gerätemodell, ein generisches Gerät hingegen eine abstrakte Gruppe von Gerätemodellen. Auf diese Weise lassen sich mehrere Gerätemodelle mit ähnlichen Eigenschaften zusammenfassen und mit einem einzelnen, generischen Geräteprofil beschreiben. Weiterhin lässt sich auf diese Weise eine Zuordnung von Farbprofilen zu Geräten gewährleisten, die sich mittels Geräteerkennung nicht als konkretes Gerätemodell identifizieren lassen.

Das Attribut *ddrIdMap* beinhaltet die jeweiligen Identifikationsschlüssel des beschriebenen Endgeräts für die mit dem Farbmanagementsystem verbundenen Device Detection Repositories. Auf diese Weise lässt sich der Rückgabewert einer Geräteerkennung einem entsprechenden Endgerät zuordnen. Das Modell der Klasse *Device* wird in Abbildung 4-4 dargestellt.

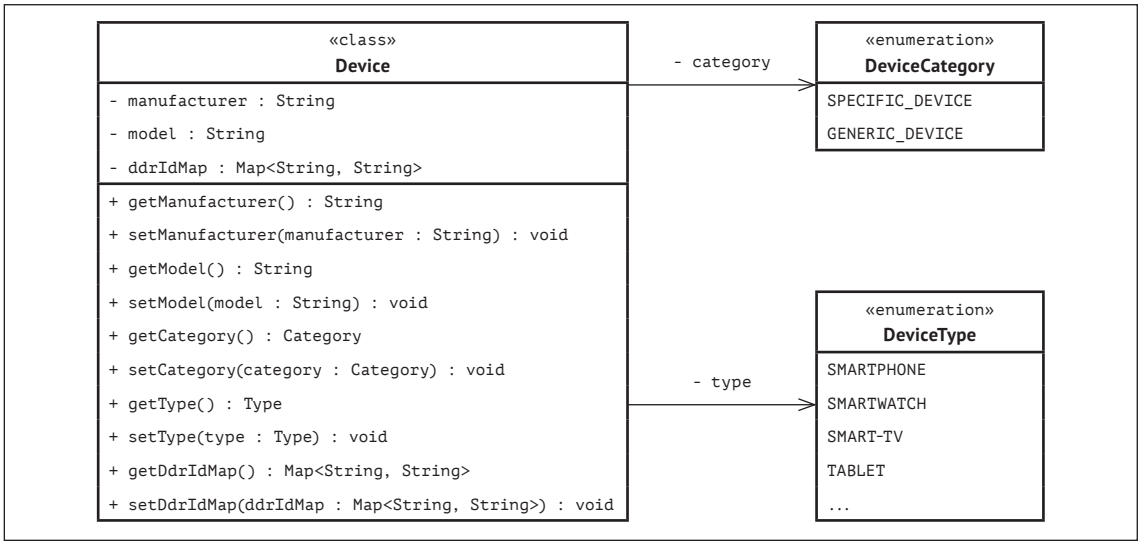


Abbildung 4-4: Klassendiagramm zur Beschreibung von Endgeräten

4.1.2. Betriebssysteme

Jedem Endgerät liegt ein Betriebssystem zugrunde. Diese Betriebssysteme werden im Kontext des Farbmanagementsystems durch die Klasse *OpSystem* abgebildet. Neben den bereits erläuterten, geerbten Eigenschaften verfügt ein Betriebssystem zusätzlich über folgende, weitere Einträge (vgl. Tab. 4-2).

Attribut	Datentyp	Beschreibung
`manufacturer`	`String`	Hersteller des Betriebssystems
`version`	`String`	Version des Betriebssystems
`cmsAvailable`	`Boolean`	Angabe, ob das Betriebssystem bereits ein CMS besitzt
`calibrationScheme`	`Map`	Kalibrierungsschema des Betriebssystems
`ddrIdMap`	`Map<String, String>`	Verknüpfung mit verbundenen Device Detection Repositories

Tabelle 4-2: Attribute von Betriebssystemen

Die Attribute *manufacturer* und *version* beschreiben den Hersteller sowie die Version des zugehörigen Betriebssystems.

Der Eintrag *cmsAvailable* gibt an, ob das beschriebene Betriebssystem bereits über ein internes Farbmanagementsystem verfügt. Bei bereits vorhandenem Farbmanagementsystem, obliegt der aufrufenden Applikation die Wahl zwischen systemseitiger bzw. externer Farbanpassung.

Das Attribut *calibrationScheme* beschreibt die zur Verfügung stehenden Farbeinstellungsoptionen eines Betriebssystems, indem es die verfügbaren Kalibrierungsparameter, deren jeweiligen Wertebereich sowie einen Vorgabewert definiert. Mit Hilfe dieses Kalibrierungsschemas kann sichergestellt werden, dass die zugrunde liegenden Kali-

brierungseinstellungen innerhalb eines Farbprofils den Optionen des Betriebssystems entsprechen. Darüber hinaus können fehlende Angaben durch hinterlegte Vorgabewerte ergänzt werden.

Da Betriebssysteme über eine Vielzahl verschiedener Versionen verfügen können, diese sich aber nicht zwingend auf farbbestimmende Eigenschaften auswirken, gelten hinterlegte Versionen als aufwärtskompatibel. Demnach ist ein aktueller Betriebssystemeintrag nur bei Änderung der Attribute *cmsAvailable* oder *calibrationScheme* erforderlich.

Analog zur Klasse *Device* besitzen Objekte der Klasse *OpSystem* das Attribut *ddrIdMap*. Das zugrunde liegende Klassendiagramm wird in Abbildung 4-5 dargestellt.

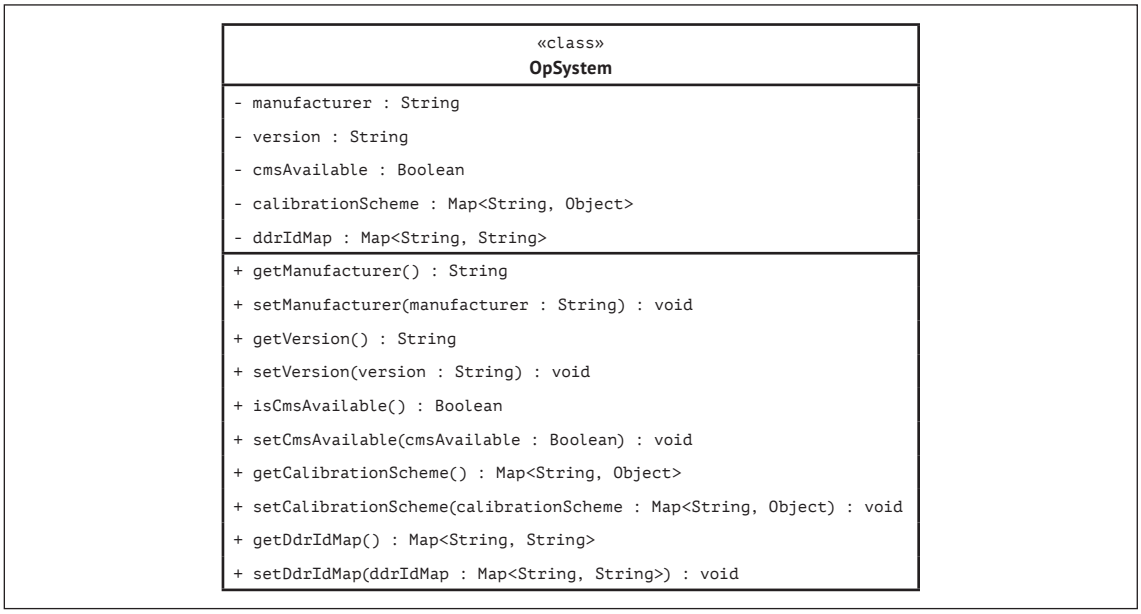


Abbildung 4-5: Klassendiagramm zur Beschreibung von Betriebssystemen

4.1.3. Farbprofile

Wie in den vorangegangenen Kapiteln erläutert, können mit Farbprofilen die Farbeigenschaften verschiedenster Ein- und Ausgabebedingungen beschrieben werden. Im Rahmen dieses Architekturvorschlags bezieht sich der Einsatz von Farbprofilen vor allem auf displaybasierte Endgeräte sowie standardisierte Wiedergabebedingungen.

Der Architekturvorschlag des Farbmanagementsystems legt keine bestimmte Farbprofilspezifikation fest, bezieht sich im weiteren Verlauf jedoch auf ICC Version 4.3 (vgl. [ICC10]).

Bei der Klasse *IccProfile* handelt es sich um einen Wrapper, der die zugrunde liegenden binären Farbprofildaten speichert und um zusätzliche Attribute ergänzt (vgl. Tab. 4-3).

Attribut	Datentyp	Beschreibung
`data`	`Byte[]`	Binärdaten des zugrunde liegenden Farbprofils
`calibration`	`Map<String, Object>`	Farbkalibrierungseinstellungen des zugehörigen Endgeräts
`deviceId`	`Integer`	Identifikationsnummer des zugehörigen Endgeräts
`opSystemId`	`Integer`	Identifikationsnummer des zugehörigen Betriebssystems
`category`	`Enumeration`	Profilkategorie (vgl. Tab. 4-5)

Tabelle 4-3: Attribute von Farbprofilen

Das Attribut *data* speichert das Farbprofil und die darin enthaltenen Farbcharakterisierungsdaten in seiner ursprünglichen Kodierungsform. Dieses Attribut wird unter anderem für die Verarbeitung des Farbprofils im Zuge einer Farbtransformation benötigt. Weiterhin lassen sich aus diesem Datenbestand einzelne Profilbestandteile (z.B. *Header*, *Tag Table* und *Tagged Element Data*) auslesen.

Die bei der Erstellung des Farbprofils zugrunde liegenden Kalibrierungseinstellungen werden mit Hilfe des Eintrags *calibration* als Liste aus Schlüssel-Wert-Paaren gespeichert.

Das Attribut *deviceId* verknüpft das Farbprofil mit einem im System hinterlegten Endgerät. Analog dazu kann ein Farbprofil mit Hilfe des Eintrags *opSystemId* mit einem bestimmten Betriebssytem verknüpft werden. Dies ist notwendig, um die im Farbprofil hinterlegten Farbkalibrierungseinstellungen beim Hinzufügen des Farbprofils mit dem Kalibrierungsschema des zugehörigen Betriebssystems zu validieren.

Weiterhin gliedern sich die im Farbmanagementsystem hinterlegten Farbprofile in eine der nachfolgenden vier Kategorien. Diese wird mit Hilfe des Eintrags *category* definiert (vgl. Tab. 4-4).

Profilkategorie	Beschreibung
Specific Device Display	Farbprofile für Displays eines spezifischen Gerätemodells
Generic Device Display	Allgemeine Farbprofile für Displays einer Gerätemodellklasse
Custom Device Display	Benutzerdefinierte Farbprofile
Generic Color Space	Allgemeine Farbraumprofile / Standardisierte Wiedergabebedingungen

Tabelle 4-4: Kategorisierung von Farbprofilen

Die Kategorie *Specific Device Display* beschreibt Farbprofile für Displays eines spezifischen Gerätemodells. Profile dieser Kategorie werden bevorzugt für Geräte eingesetzt, die im Zuge einer Farbkalibrierung und -transformation automatisch ermittelt werden sollen. Modellspezifische Farbprofile stellen einen qualitativen Kompromiss zwischen benutzerdefinierten und generischen Farbprofilen dar.

Bei Farbprofilen der Kategorie *Generic Device Display* handelt es sich um Profile für eine Reihe von Gerätemodellen, die aufgrund vergleichbarer Farbwiedergabeeigenschaften zu einer Klasse zusammengefasst werden können. Mit generischen Profilen lassen sich mehrere Endgeräte mit nur einem Farbprofil hinreichend gut beschreiben. Farbprofile dieser Kategorie werden mit entsprechenden, generischen Endgeräten verknüpft.

Die Kategorie *Custom Device Display* beschreibt Farbprofile für Displays eines spezifischen Gerätemodells mit benutzerdefinierten Modifikationen. Falls ein Endgerät beispielsweise mit individuellen Farbkalibrierungseinstellungen eingesetzt werden soll, kann für diesen Fall ein benutzerdefiniertes Farbprofil hinzugefügt werden. Dieser Ansatz eignet sich insbesondere für Anwendungsfälle, in denen eine individuelle, möglichst präzise Farbwiedergabe vorausgesetzt wird.

Die Profilkategorie *Generic Color Space* beschreibt generische Farbräume sowie standardisierte Wiedergabebedingungen wie z.B. sRGB, Adobe RGB (1998) oder DCI-P3, die sowohl als Eingabe- und Ausgabefarbprofil eingesetzt werden können. Da diese Profilkategorie keine Gerätemodelle bzw. -modellklassen beschreibt, sind weder Kalibrierungseinstellungen noch die Verweise auf Endgerät und Betriebssystem erforderlich. Das Diagramm der Klasse *IccProfile* wird in Abbildung 4-6 dargestellt.

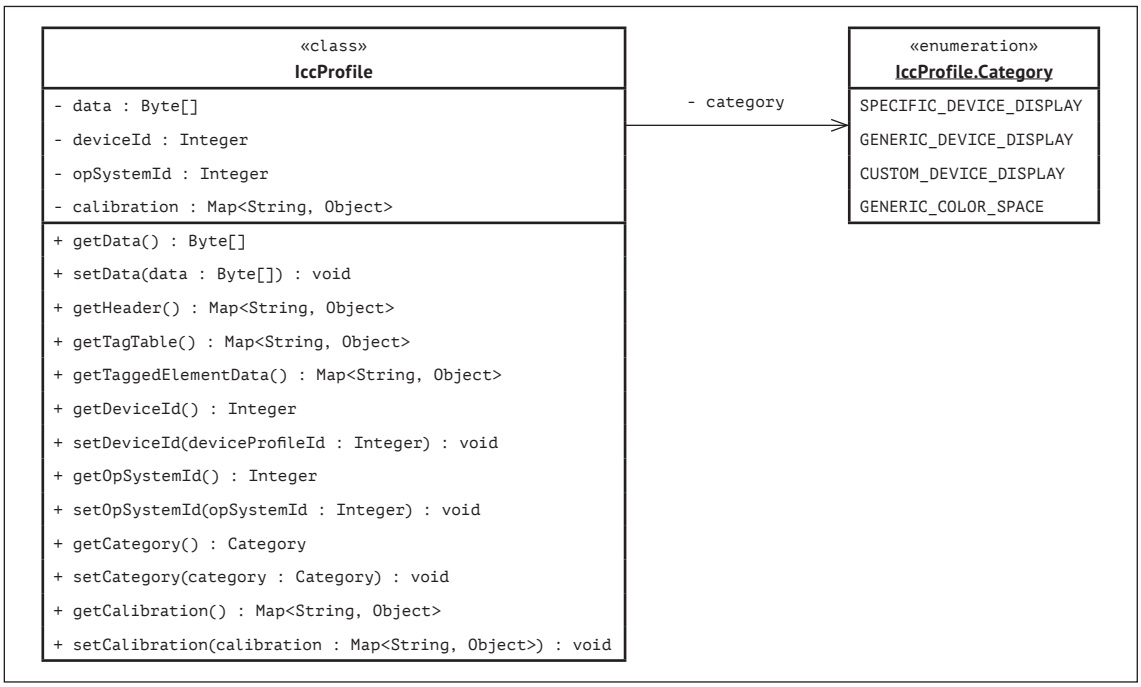


Abbildung 4-6: Klassendiagramm zur Beschreibung von Farbprofilen

4.1.4. Farbmanagementmodule

Die Klasse *Cmm* beschreibt die im Farbmanagementsystem hinterlegten Farbmanagementmodule. Instanzen dieser Klasse besitzen folgende, weitere Attribute (vgl. Tab. 4-5).

Attribut	Datentyp	Beschreibung
`manufacturer`	`String`	Hersteller des Farbmanagementmoduls
`version`	`String`	Version des Farbmanagementmoduls
`specification`	`String`	Unterstützte Farbprofilspezifikation
`abbreviation`	`String`	Kürzel des Farbmanagementmoduls
`capabilities`	`Map<String, Object>`	Erweiterte Farbmanagementfunktionen des Moduls
`data`	`Byte[]`	Binärdaten des zugrunde liegenden Farbmanagementmoduls

Tabelle 4-5: Attribute von Farbmanagementmodulen

Die Einträge *manufacturer, version* und *specification* beschreiben den Hersteller, die Version sowie die unterstützte Farbprofilspezifikation eines Farbmanagementmoduls.

Das Attribut *abbreviation* beinhaltet das Kürzel, unter dem das Farbmanagementmodul im Header-Eintrag *Preferred CMM* eines Farbprofils angegeben ist. Auf diese Weise kann eine Zuordnung von Farbprofil und hinterlegtem Farbmanagementmodul stattfinden und die Auswahl des Farbmanagementmoduls bei der Farbtransformation automatisiert werden.

Der Eintrag *capabilities* beschreibt eine Liste erweiterter Funktionen eines Farbmanagementmoduls, die über die erforderlichen und optionalen Funktionen einer Farbprofilspezifikation hinausgehen. Als konkrete Beispiele sind in diesem Zusammenhang die Tiefenkompensierung sowie auf *private tags* basierende Zusatzfunktionen zu nennen.

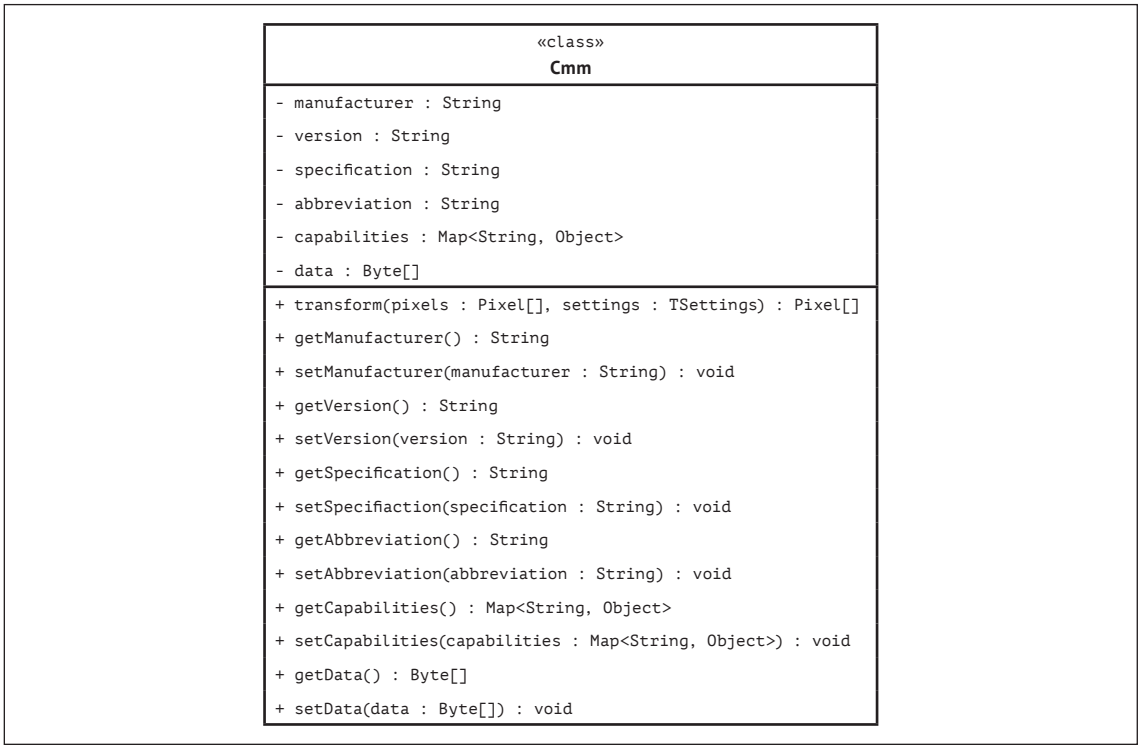


Abbildung 4-7: Klassendiagramm zur Beschreibung von Farbmanagementmodulen

Das Attribut *data* beinhaltet die Binärdaten des eigentlichen Farbmanagementmoduls, welches zur Umrechnung von Farbwerten auf Basis zugrunde liegender Farbtransformationseinstellungen dient. Die Ansteuerung der Farbtransformation erfolgt mit Hilfe der Methode *transform*, die von jedem hinterlegten Farbmanagementmodul zu implementieren ist (vgl. Abb. 4-7).

4.1.5. Pixel

Die Klasse *Pixel* beschreibt einen Farbwert bestehend aus einer definierten Anzahl an Subpixeln. Jedes Pixel besitzt eine bestimmte Farbkodierung zur Angabe der Farbtiefe sowie des Kodierungsschemas (vgl. Abb. 4-8). Pixelwerte, die zur Beschreibung von Farbfächen, Schriften oder sonstigen Seitenelementen dienen, können als strukturiertes Datenformat an das Farbmanagementsystem übertragen werden. Ebenso werden jedoch auch Bilder bzw. seitenbasierte Dokumente, die an das Farbmanagementsystem übertragen werden, zerlegt und in das beschriebene Pixelformat überführt.

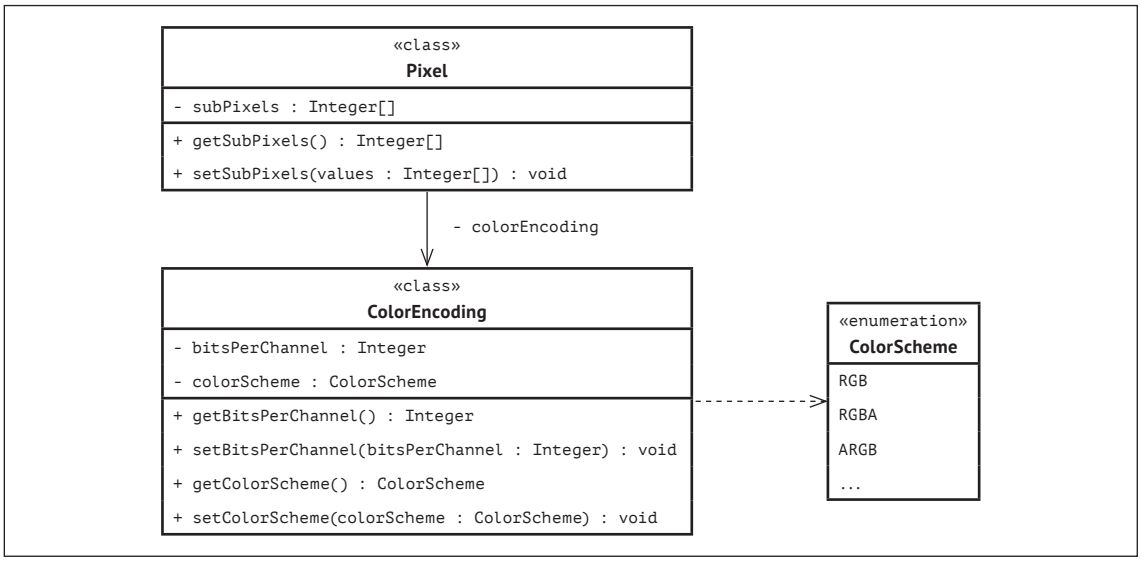


Abbildung 4-8: Klassendiagramm zur Beschreibung von Pixeln

4.1.6. Farbtransformationseinstellungen

Die für die Farbtransformation benötigten Einstellungen werden durch Objekte der Klasse *TSettings* beschrieben. Diese beinhaltet die Attribute *inputProfile*, *outputProfile*, *renderingIntent, cmm* sowie den Eintrag *parameterMap*, welcher zusätzliche, optionale Transformationsparameter beinhalten kann (vgl. Abb. 4-9).

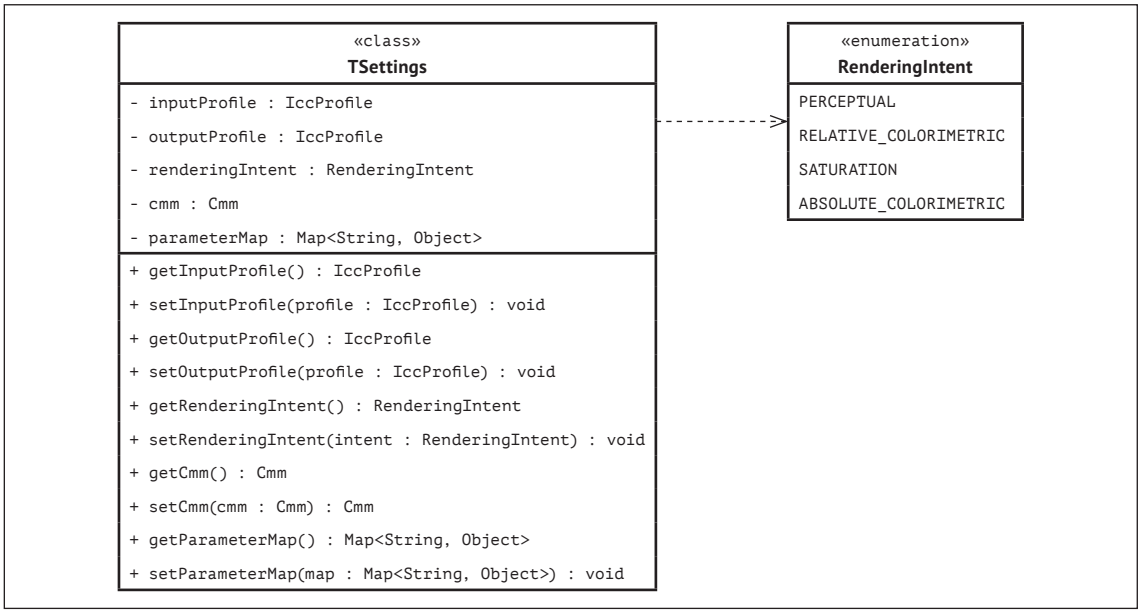


Abbildung 4-9: Klassendiagramm zur Beschreibung von Farbtransformationseinstellungen

4.2. Dienstschicht

Die Dienstschicht stellt die oberste Schicht des Backends dar. Sie besteht aus den Komponenten *ServiceApi*, *Authorization, Validation* sowie *FileUpload*. Der Einsatz einer solchen Schicht gewährleistet Flexibilität im Bezug auf die Wahl der Kommunikationstechnologie, des Autorisierungsverfahrens sowie des Nachrichtenformats für den Datenaustausch zwischen Client und Server. Nachfolgend werden die oben genannten Komponenten im Einzelnen vorgestellt.

4.2.1. Dienstzugang

Die Komponente *ServiceApi* definiert die Programmierschnittstelle, über die Applikationen auf das Farbmanagementsystem zugreifen können. Sie stellt Methoden für die zentralen Anwendungsfälle Farbkalibrierung, Farbtransformation und Ressourcenverwaltung zur Verfügung. Diese Methoden werden mit Hilfe einer Schnittstelle deklariert und von einer technologiespezifischen Instanz implementiert, die jeweils einen bestimmten Servicetyp repräsentiert (z.B. REST, SOAP etc.).

Hierzu werden generische Datentypen genutzt, die je nach Wahl des Kommunikationsprotokolls zu spezifischen Typen gebunden werden. In den Beispielen der folgenden Abbildung handelt es sich dabei jeweils um Instanzen der Klassen *HTTP-Request* bzw. *HTTP-Response*.

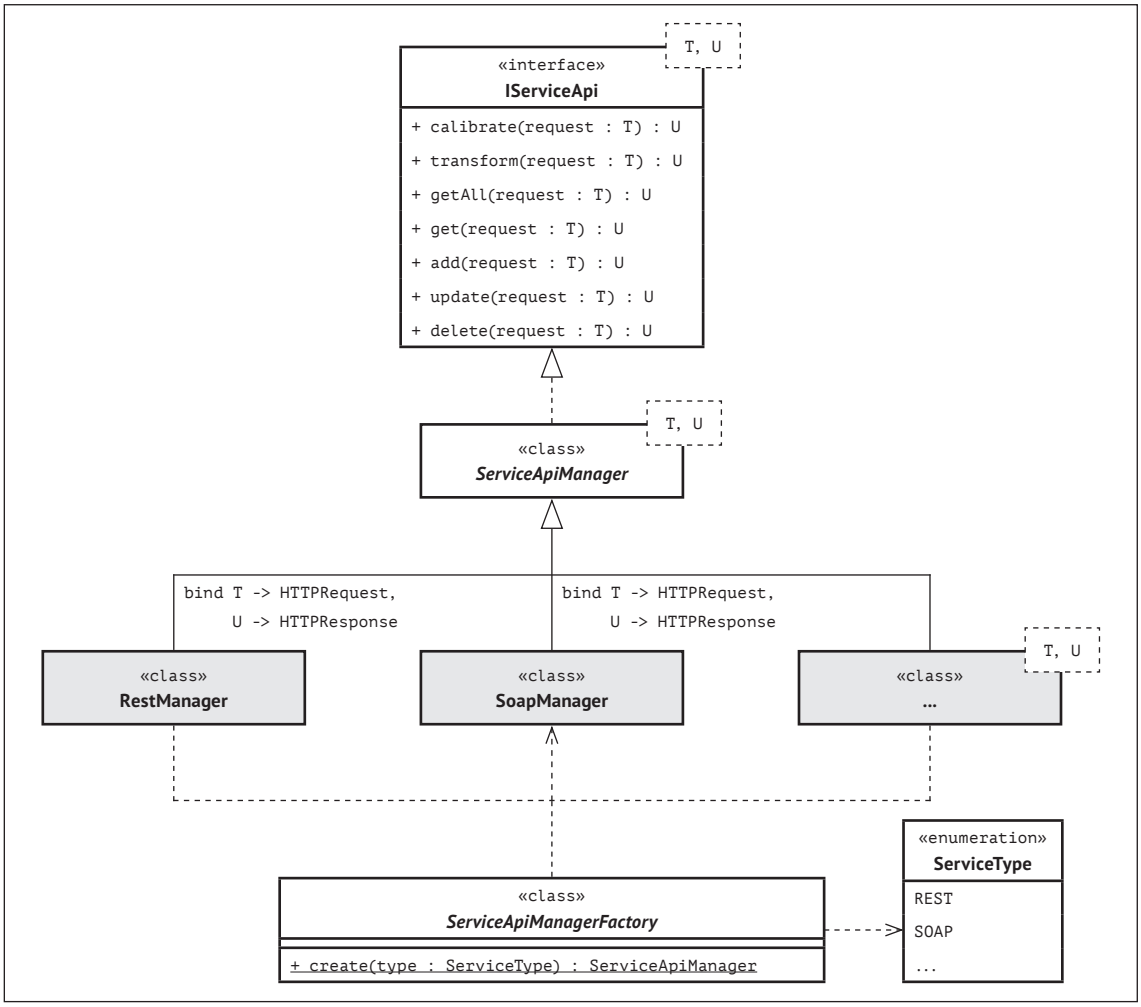


Abbildung 4-10: Klassendiagramm der Komponente »ServiceApi«

Grundsätzlich wird jeder Prozess von einer konkreten Instanz einer Subklasse von *ServiceApiManager* eingeleitet, die mit Hilfe einer Fabrikmethode erzeugt und als Einstiegspunkt für den Dienstzugriff festgelegt wird (vgl. Abb. 4-11).

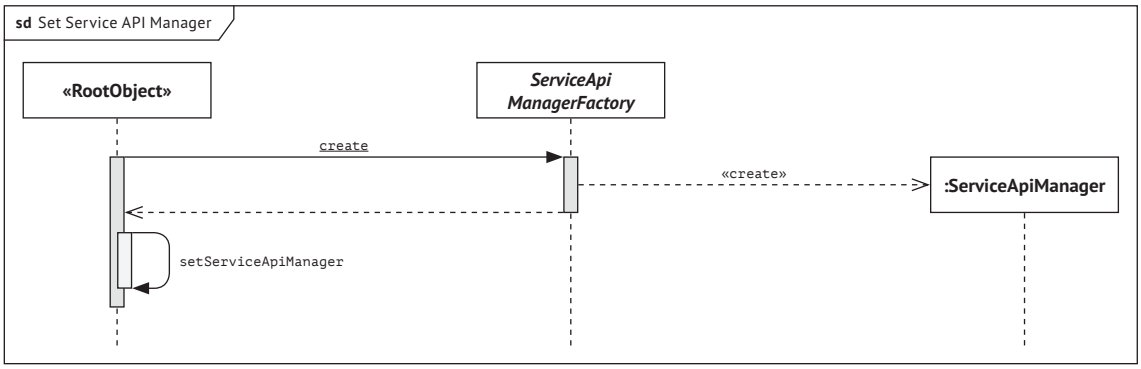


Abbildung 4-11: Sequenzdiagramm für die Erzeugung und Festlegung des Einstiegsobjekts für den Dienstzugriff

4.2.2. Autorisierung

Die Komponente *Authorization* stellt sicher, dass ein Systemnutzer für eine konkrete Anfrage berechtigt ist. Nachdem sich ein Benutzer bzw. eine Applikation mit den entsprechenden Anmeldedaten authentifiziert hat, wird mit Hilfe der Autorisierung festgestellt, ob für die angefragte Aktion (z.B. das Hinzufügen eines Farbprofils) eine Berechtigung existiert.

Eine Autorisierung, also die Entscheidung, ob ein Benutzer eine bestimmte Aktion ausführen darf oder nicht, kann grundsätzlich zum Teil von der Infrastruktur übernommen werden. Der zugrunde liegende Autorisierungsmechanismus lässt sich jedoch systemseitig festlegen.

Die Struktur dieser Komponente folgt dem bereits vorgestellten Schema, nämlich der Deklaration einer Schnittstelle, deren Implementierung mittels abstrakter Managerklasse sowie der Erzeugung einer konkreten Subklasse auf Basis eines bestimmten Autorisierungsmechanismus (vgl. Abb. 4-12).

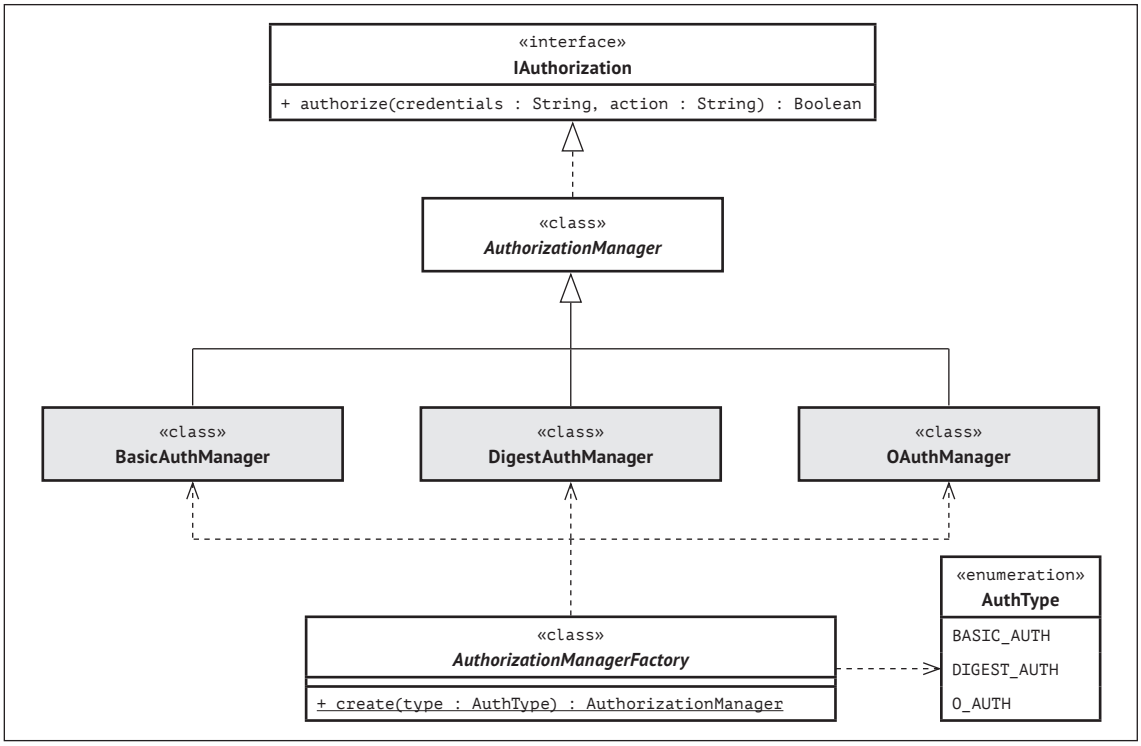


Abbildung 4-12: Klassendiagramm der Komponente »Authorization«

Falls die Ausführung eines Prozesses die Autorisierung von Benutzerrechten erfordert, wird die Methode *authorize* der technologiespezifischen Klasse des entsprechenden Autorisierungsmechanismus aufgerufen. Diese Methode nimmt die Anmeldedaten eines Benutzers entgegen und prüft, ob er für die angegebene Aktion autorisiert ist. Diese Methode liefert folglich einen Wahrheitswert *true* oder *false* zurück (vgl. Abb. 4-13).

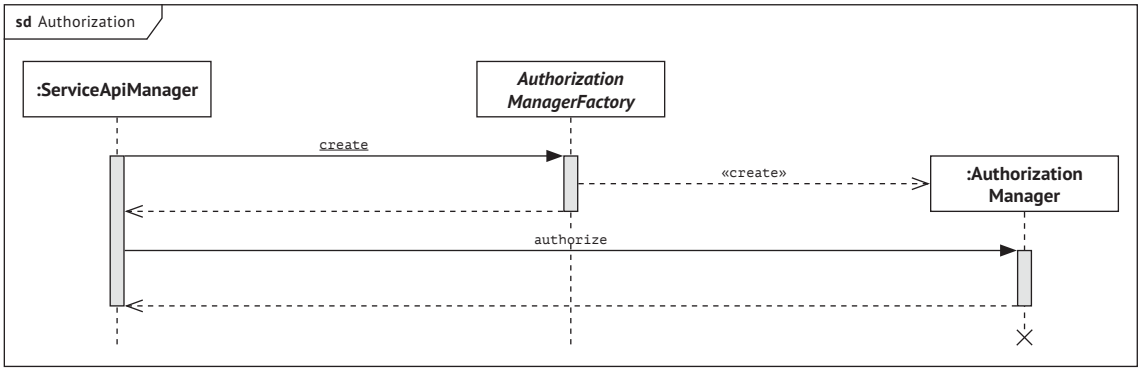


Abbildung 4-13: Sequenzdiagramm des Autorisierungsprozesses

4.2.3. Validierung

Die Komponente *Validation* soll sicherstellen, dass die vom Nutzer übermittelten Eingangsdaten syntaktisch korrekt und systemkompatibel sind. Die Eingangsdaten werden in Form eines strukturierten Dokumentenformats übermittelt und anschließend mit Hilfe eines Validierungsschemas geprüft.

Die Architektur gewährleistet die flexible Wahl des gewünschten Austauschformats. Das entsprechende Klassendiagramm der Komponente *Validation* wird in Abbildung 4-14 dargestellt.

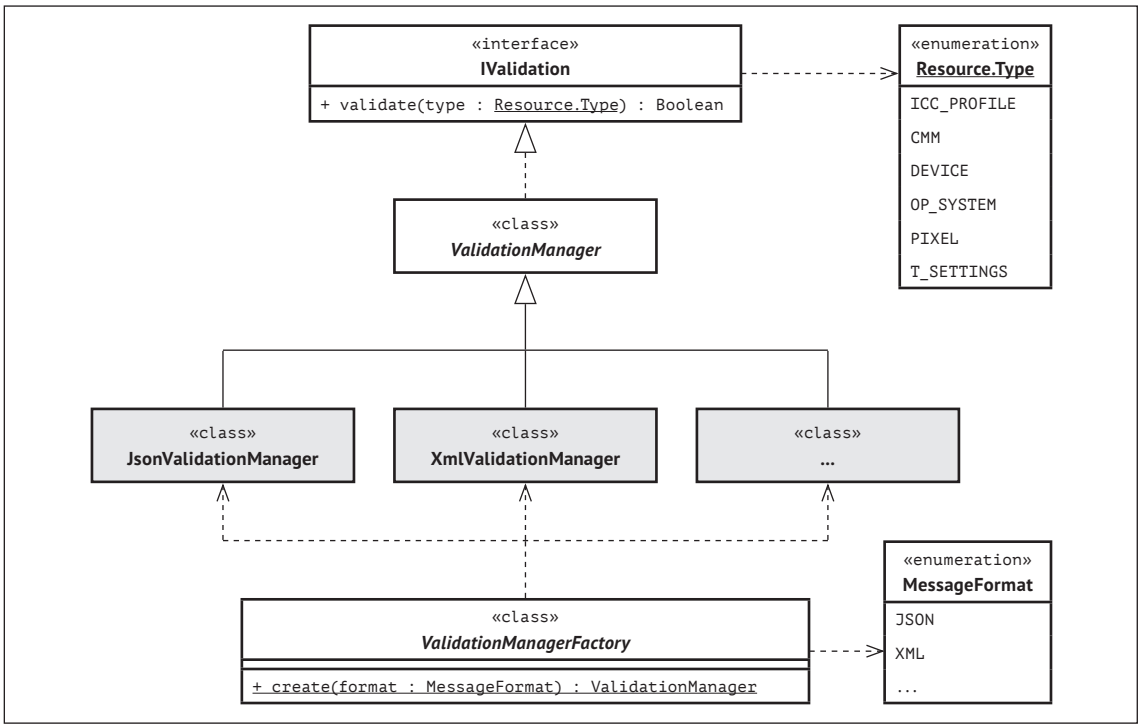


Abbildung 4-14: Klassendiagramm der Komponente »Validation«

Sowohl die Ressourcenverwaltung als auch die Farbtransformation basieren auf eingehenden Daten und setzen daher eine Validierung dieser Daten voraus. Hierzu wird eine konkrete Subklasse von *ValidationManager* für das verwendete Austauschformat mittels Fabrikmethode erzeugt und die zugehörige Methode *validate* unter Angabe des zu validierenden Typs aufgerufen. Diese prüft die Eingangsdaten anhand eines entsprechenden Validierungsschemas und liefert wiederum den Wahrheitswert *true* oder *false* zurück (vgl. Abb. 4-15).

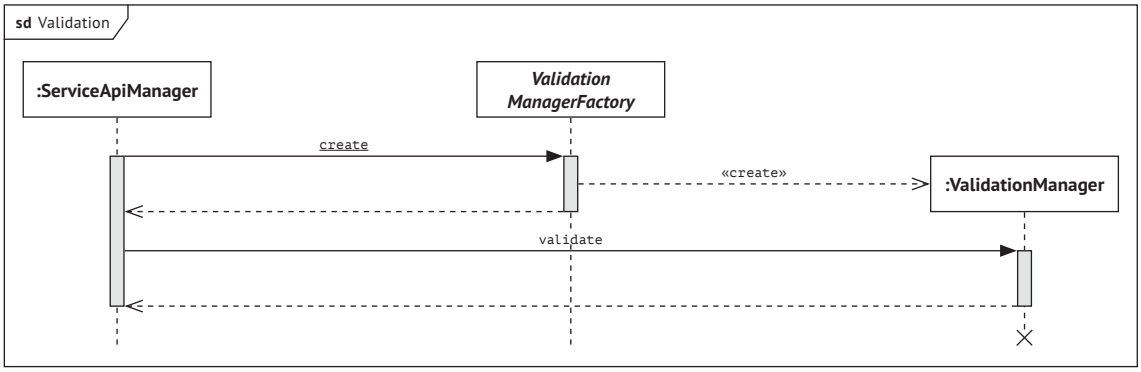


Abbildung 4-15: Sequenzdiagramm des Validierungsprozesses

4.2.4. Dateiupload

Aufgabe der Komponente *FileUpload* ist es, zu transformierende Farbinhalte (wie z.B. Bilddaten) die mittels URL-Referenzierung an das Farbmanagementsystem übertragen wurden, auf einen angeschlossenen Webserver hochzuladen und im Anschluss an die Farbtransformation wiederum als URL zur Verfügung zu stellen.

Hierzu deklariert diese Komponente eine Methode *upload*, die eine binäre Datei entgegen nimmt, hochlädt und die entsprechende URL an die aufrufende Klasse zurückgibt. Diese Methode wird nach dem bereits vorgestellten Schema von einer konkreten Klasse, die vom Typ *FileUploadManager* erbt, implementiert (vgl. Abb. 4-16).

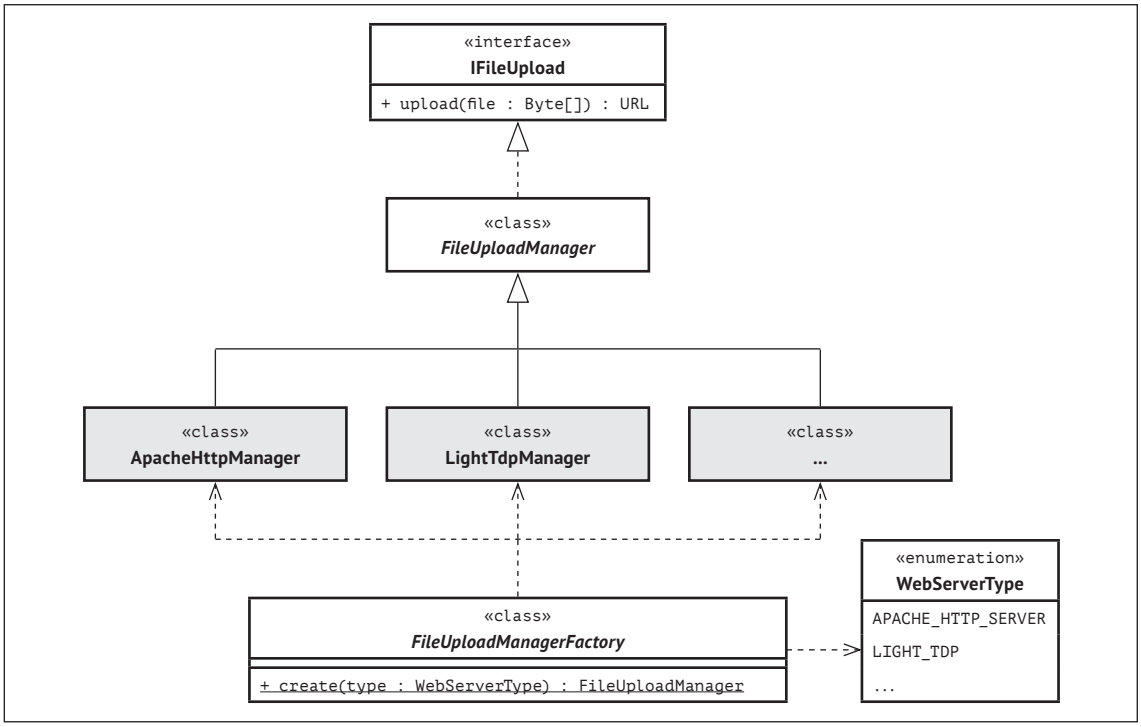


Abbildung 4-16: Klassendiagramm der Komponente »FileUpload«

Diese konkrete Managerklasse wird wiederum mittels Fabrikmethode erzeugt und repräsentiert den Typ des verbundenen, spezifischen Webservers. Der sequenzielle Ablauf für das Hochladen und Zurverfügungstellen von farbtransformierten Dateien wird im folgenden Diagramm abgebildet.

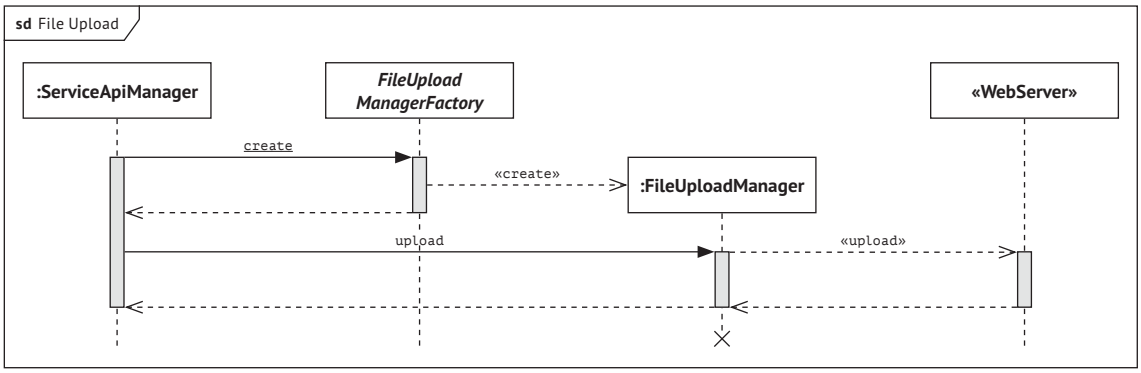


Abbildung 4-17: Sequenzdiagramm des Prozesses zum Dateiupload

4.3. Applikationsschicht

Die Applikationsschicht repräsentiert die zentrale Prozesslogik des Farbmanagementsystems. Sie beinhaltet Komponenten für die Farbkalibrierung und Farbtransformation sowie Hilfskomponenten zur Geräteerkennung und zur Kodierung von eingehenden Binärdokumenten.

Bei letzterem werden insbesondere Pixeldaten sowie etwaige, eingebettete Farbprofile aus Farbdateien extrahiert und für die Farbtransformation aufbereitet.

4.3.1. Geräteerkennung

Die Komponente *DeviceDetection* hat die Aufgabe, das aufrufende Gerätemodell sowie dessen zugrunde liegendes Betriebssystem zu identifizieren und somit die Ermittlung der für die Farbkalibrierung und -transformation benötigten Parameter sowie die Automatisierung von Farbanpassungsvorgängen zu ermöglichen.

Hierzu nimmt die Komponente einen gerätespezifischen Identifizierungsschlüssel entgegen und leitet diesen an eine Geräteerkennungsdatenbank *(engl. Device Detection Repository)* weiter. Anhand der erzeugten Rückgabewerte und unter Verwendung des Attributs *ddrIdMap* der Klassen *Device* und *OpSystem,* lassen sich die spezifischen *ids* des aufrufenden Endgeräts und Betriebssystems ermitteln.

Diese Angaben werden wiederum für die Zuweisung des Ausgabefarbprofils benötigt, das unter anderem die Farbkalibrierungs- und transformationsparameter beinhaltet. Die Klassenhierarchie dieser Komponente wird in Abbildung 4-18 dargestellt.

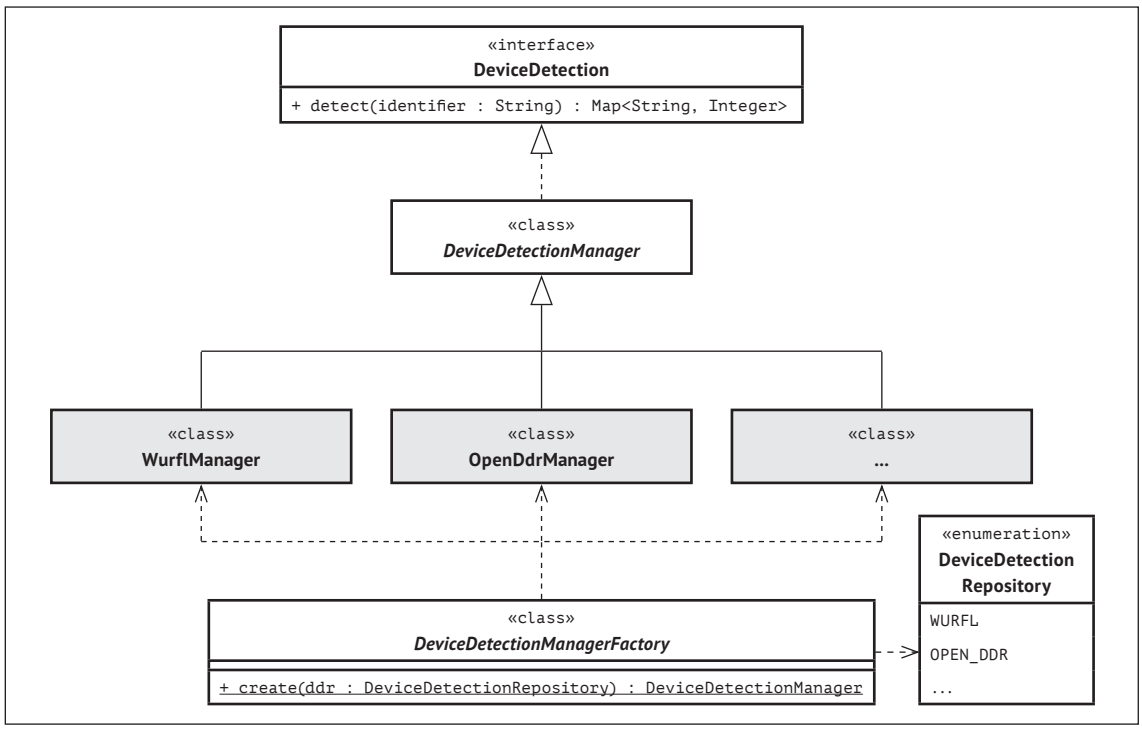


Abbildung 4-18: Klassendiagramm der Komponente »DeviceDetection«

Die beschriebene Logik dieser Komponente wird durch die Methode *detect* abgebildet. Diese wird wiederum von einer konkreten Managerklasse implementiert, die die Anfrage an eine spezifische Geräteerkennungsdatenbank weiterleitet. Der sequenzielle Ablauf des beschriebenen Prozesses wird in Abbildung 4-19 dargestellt.

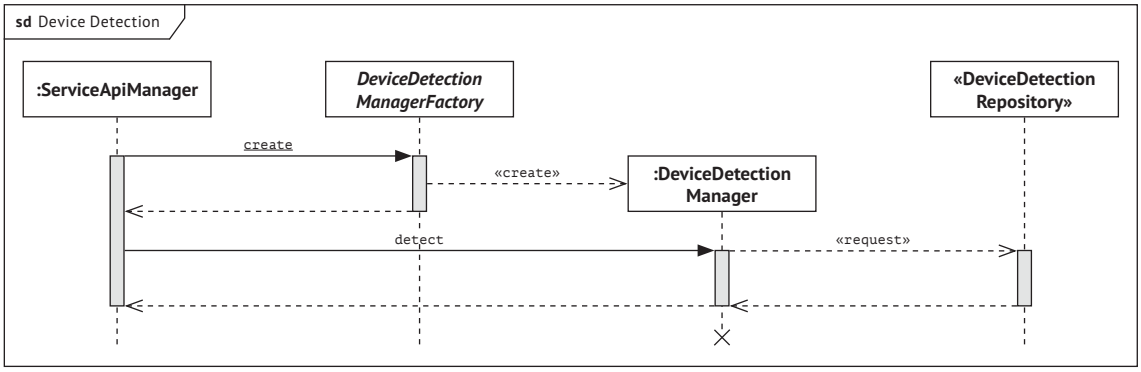


Abbildung 4-19: Sequenzdiagramm des Geräteerkennungsprozesses

4.3.2. Farbdatenkodierung

Aufgabe der Farbdatenkodierung ist die Verarbeitung von Farbdaten vor und nach der Farbtransformation. Hierzu werden innerhalb der Komponente *ColorCoding* zwei Schnittstellen definiert.

Die Schnittstelle *IImageCoding* deklariert Methoden zur Dekodierung von eingehenden Bilddateien in formatunabhängige Bildobjekte sowie zur Enkodierung in das ursprüngliche Bildformat. Darüber hinaus deklariert sie eine Methode zur Extrahierung von eingebetteten Eingabefarbprofilen.

Die Schnittstelle *IPixelCoding* stellt wiederum Methoden zur Verfügung, um aus den angesprochenen, formatunabhängigen Bildobjekten die enthaltenen Farbdaten als Zahlenwerte zu entnehmen. Diese Farbwerte können anschließend mittels eines ausgewählten Farbmanagementmoduls transformiert und in das ursprüngliche Bildobjekt eingefügt werden.

Da die Komponente über zwei Schnittstellen verfügt, wird der Zugang mit Hilfe einer vorgeschalteten Fassade zusammengefasst. Dies hat zur Folge, dass diese Komponente für zugreifende Objekte einen einheitlichen Ansprechpartner zur Verfügung stellt und die einzelnen Managerklassen somit nicht manuell instanziiert werden müssen (vgl. Abb. 4-20).

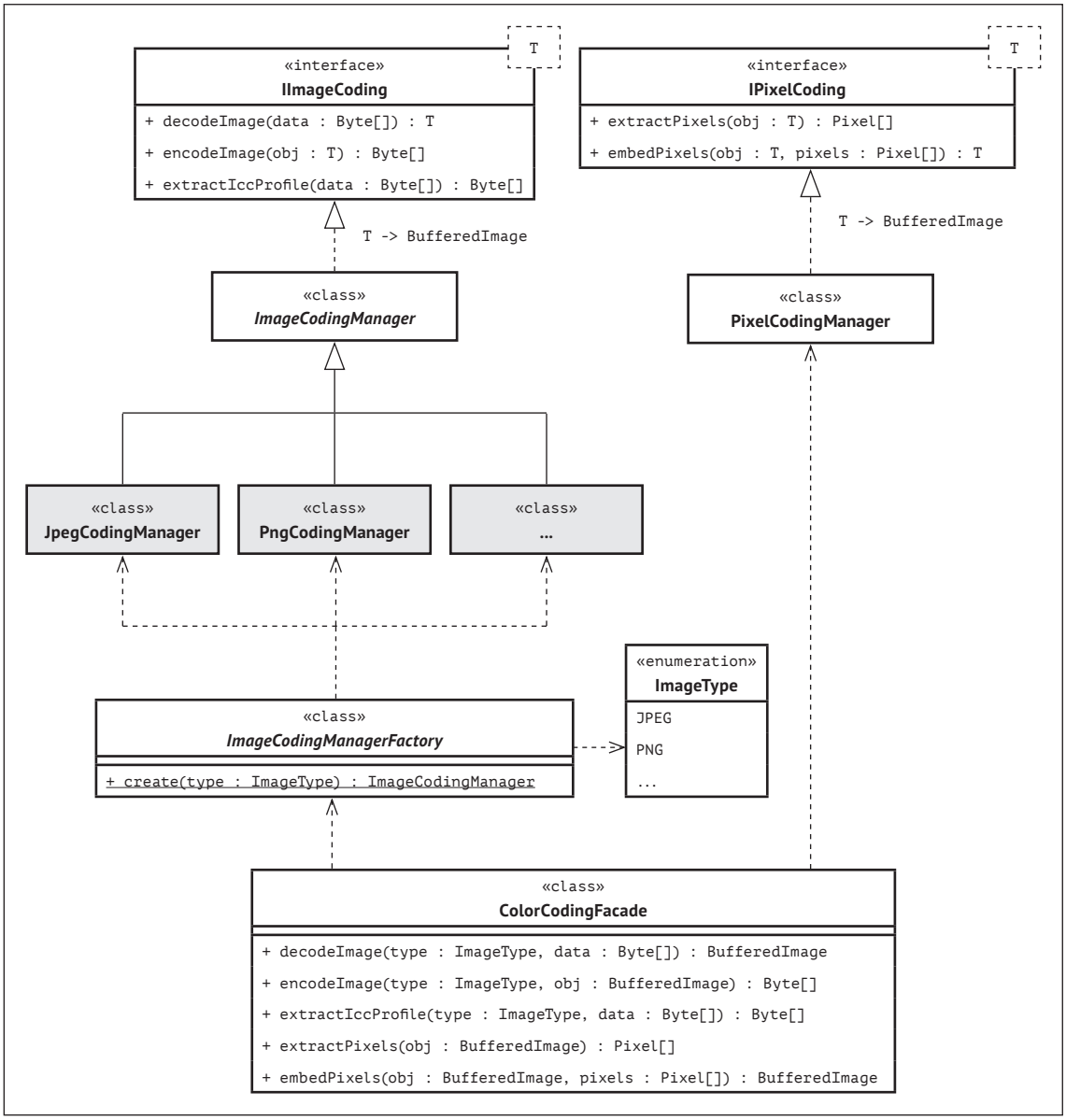


Abbildung 4-20: Klassendiagramm der Komponente »ColorCoding«

Nachfolgend werden die sequenziellen Abläufe der Prozesse zur Extrahierung eingebetteter Farbprofile, zum De- und Enkodieren von Bilddaten sowie zum Entnehmen der Pixelwerte für die eigentliche Farbumrechnung vorgestellt.

Moderne Bilddatenformate bieten die Möglichkeit zur Einbettung von Farbprofilen. Diese Farbprofile können vor dem Farbtransformationsprozess extrahiert und als Eingabefarbprofil genutzt werden. Hierzu wird eine entsprechende Anfrage an die Klasse *ColorCodingFacade* gestellt. Diese erzeugt mittels Fabrikmuster eine konkrete Subklasse von *ImageCodingManager* für das ausgewählte Bilddatenformat, um die binären Farbprofildaten zu extrahieren und an die Fassade zurückzugeben. Auf Basis der entnommenen Farbprofildaten wird eine Instanz der Klasse *IccProfile* erzeugt (vgl. Abb. 4-3, S. 22), die zur späteren Verwendung im Farbtransformationsprozess genutzt wird (vgl. Abb. 4-21).

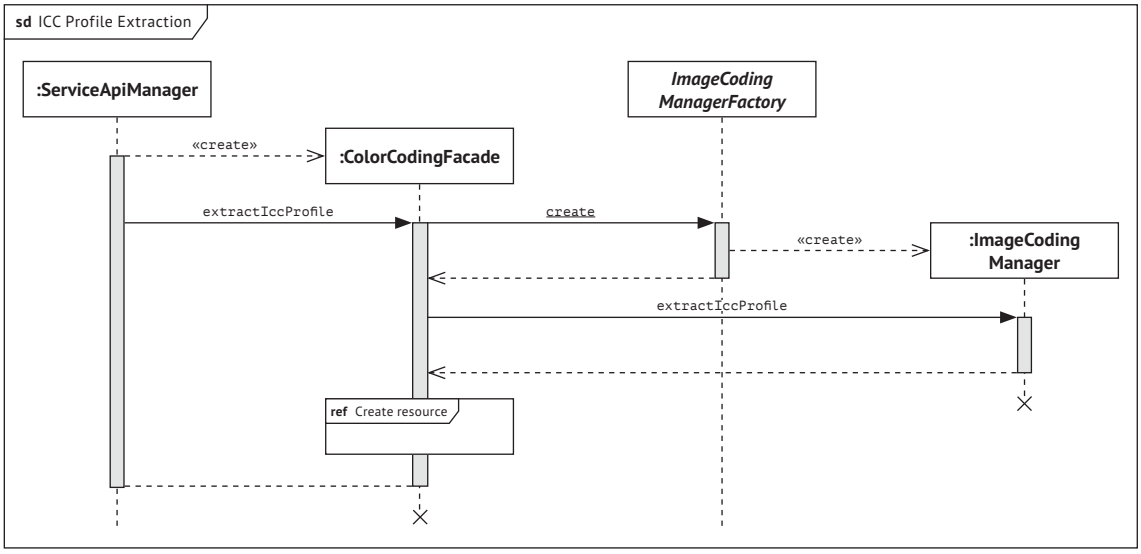


Abbildung 4-21: Sequenzdiagramm für das Extrahieren von Farbprofilen aus Bilddaten

Eine Farbtransformation basiert grundsätzlich auf einem oder mehreren Eingabefarbwerten. Daher müssen binärkodierte Datenformate vor der Farbtransformation dekodiert und die benötigten Pixelwerte extrahiert werden.

Hierzu wird eine Instanz einer Subklasse von *ImageCodingManager* für das zugrunde liegende Bildformat erzeugt und die eingehenden Bilddaten unter Verwendung der Methode *decodeImage* in ein formatneutrales Bildobjekt umgewandelt. Anschließend werden dem Bildobjekt die beinhalteten Pixelwerte unter Verwendung der Klasse *PixelCodingManager* sowie der zugehörigen Methode *extractPixels* entnommen (vgl. Abb. 4-22).

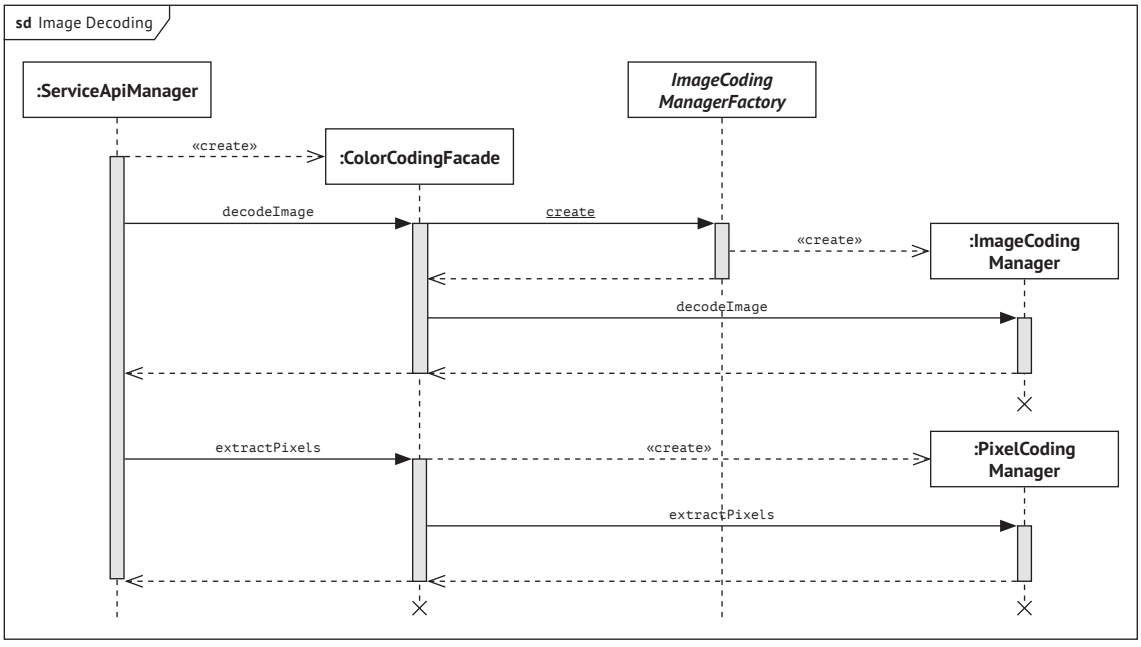


Abbildung 4-22: Sequenzdiagramm für die Dekodierung von Bilddaten und Extrahierung der enthaltenen Pixelwerte

Im Anschluss an die Farbtransformation lassen sich die umgerechneten Pixelwerte mit Hilfe der Methode *embedPixels* in das zuvor erzeugte Bildobjekt einbetten, welches wiederum unter Verwendung der Methode *encodeImage* in das ursprüngliche Bilddatenformat umgewandelt wird (vgl. Abb. 4-23).

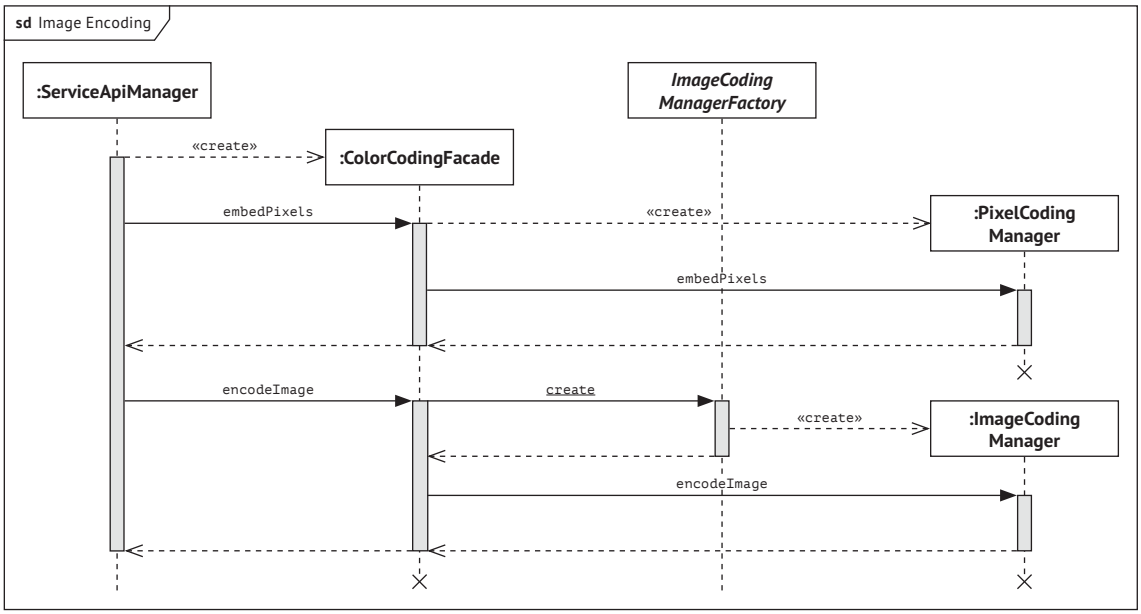


Abbildung 4-23: Sequenzdiagramm für das Einbetten von Pixelwerten in Bildobjekte und die Enkodierung als binäres Bildformat

Die beschriebene Komponente bildet im dargestellten Abschnitt ausschließlich die De- und Enkodierung von Bilddaten und Pixelwerten ab. Grundsätzlich ist jedoch auch die Unterstützung weiterer Dokumentenformate wie z.B. Bewegtbilddaten oder seitenbasierter Dokumente möglich. In diesem Fall muss für das vorliegende Dokumentenformat eine zusätzliche Managerklasse implementiert werden, die den Datenbestand in Bilder bzw. Pixelwerte zerlegt. Die weitere Vorgehensweise bleibt hiervon unberührt.

4.3.3. Farbkalibrierung

Die Aufgabe der Farbkalibrierung besteht darin, ein aufrufendes Endgerät vor der Farbtransformation in einen farblichen Referenzzustand zu überführen. Hierzu werden die benötigten Farbkalibrierungseinstellungen dem entsprechenden Ausgabefarbprofil entnommen.

Unter bestimmten Umständen ist es zudem erforderlich, dass die Farbkalibrierung eine definierte Leuchtdichte auf dem ausgebenden Display erzielt. Dies trifft beispielsweise zu, wenn ein Endgerät unter konstanten Umgebungslichtbedingungen betrieben wird oder mehrere Endgeräte zu Vergleichszwecken auf einen gemeinsamen Luminanzwert kalibriert werden müssen.

Hierzu stellt die Komponente *ColorCalibration* die Schnittstellenmethode *calibrate* zur Verfügung. Diese übernimmt die beiden optionalen Parameter *outputProfileId* und *luminance*. Der erste Parameter beinhaltet die *id* des Ausgabeprofils, das die benötigten Kalibrierungseinstellungen beinhaltet. Unter Verwendung des zweiten Parameters kann eine zu erzielende, absolute Leuchtdichte definiert werden. Hierzu wird der luminanzbestimmenden Farbkalibrierungsparameter auf den entsprechenden Wert eingestellt.

Im nachfolgenden Klassendiagramm wird die Schnittstelle und die implementierende Managerklasse der Komponente *ColorCalibration* dargestellt. Da in diesem Zusammenhang keine technologiespezifischen Komponenten existieren, besteht die Komponente lediglich aus den zwei angesprochenen Elementen (vgl. Abb. 4-24).

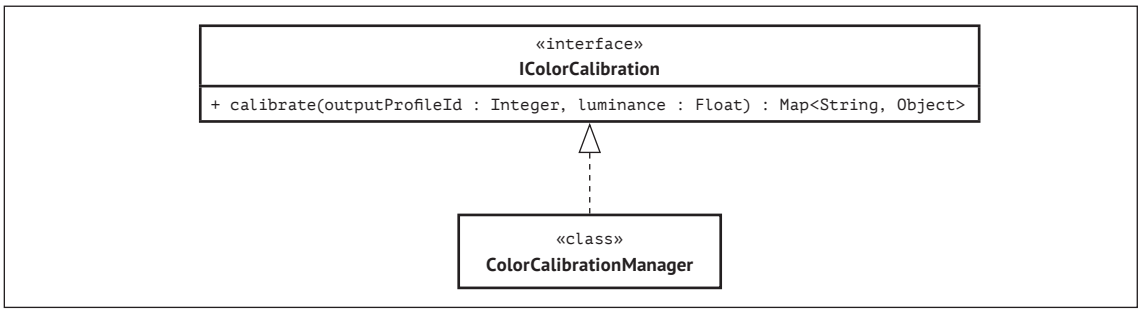


Abbildung 4-24: Klassendiagramm der Komponente »ColorCalibration«

Wie zuvor angedeutet, sind die Parameter der Methode *calibrate* optional. Bei fehlender Angabe des Ausgabefarbprofils, erfolgt die Ermittlung per Geräteerkennung. Die Farbkalibrierungseinstellungen werden über die Datenzugriffsschicht aus der Datenbank bezogen. Der sequenzielle Ablauf sowie die Modellierung als Aktivitätsdiagramm wird in den nachfolgenden Abbildungen 4-25 und 4-26 dargestellt.

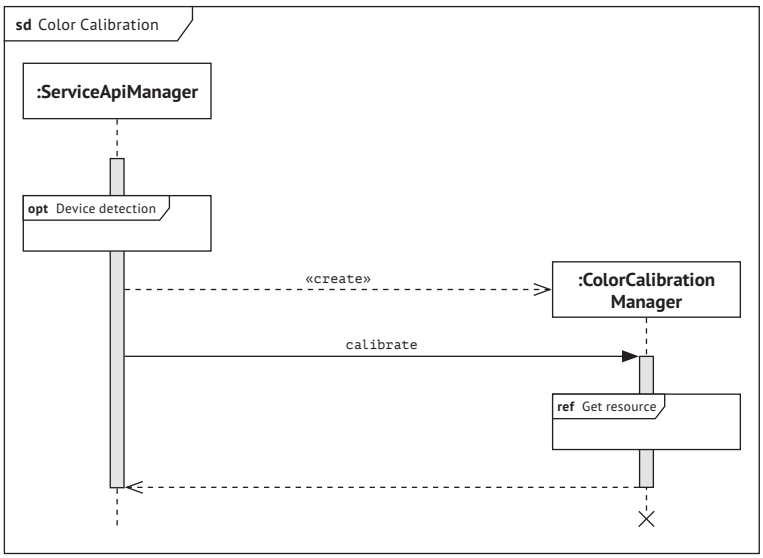


Abbildung 4-25: Sequenzdiagramm des Farbkalibrierungsprozesses

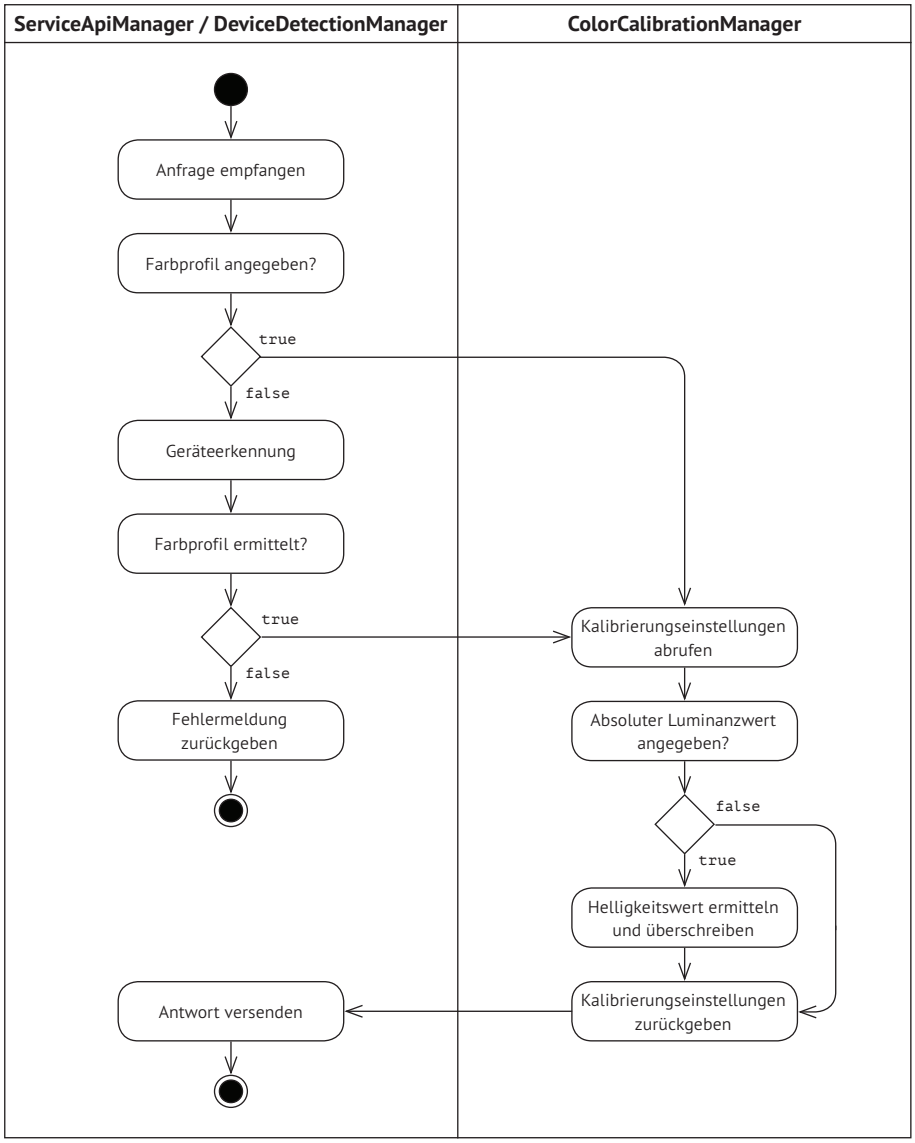


Abbildung 4-26: Aktivitätsdiagramm des Farbkalibrierungsprozesses

4.3.4. Farbtransformation

Für die Umrechnung von Farbwerten im Kontext des Farbmanagementsystems ist die Komponente *ColorTransformation* verantwortlich. Sie hat die Aufgabe, die ihr übermittelten Farbdaten unter Angabe entsprechender Parameter mit Hilfe eines Farbmanagementmoduls umzurechnen.

Hierzu deklariert die Schnittstelle der Komponente die Methode *transformWithCmm*, die wiederum von der Klasse *ColorTransformationManager* implementiert wird. Die Methode erwartet die Angabe von einem oder mehreren Pixelwerten sowie einem Parametersatz an Farbtransformationseinstellungen. Als Rückgabewert werden farbtransformierte Pixel erzeugt (vgl. Abb. 4-27).

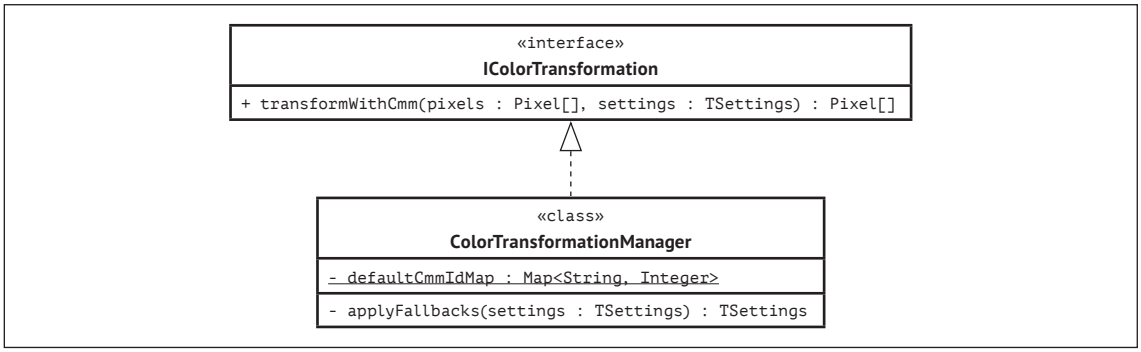


Abbildung 4-27: Klassendiagramm der Komponente »ColorTransformation«

Wie in Abschnitt 2.3. erläutert, verfügt jedes Farbmanagementsystem über ein vorgebenes Farbmanagementmodul. Diese Festlegung trifft die Klasse *ColorTransformationManager* mit Hilfe des Parameters *defaultCmmIdMap*. Hierbei handelt es sich um eine Liste von Schlüssel-Wert-Paaren, die für eine Farbprofilspezifikation jeweils ein vorgegebenes Farbmanagementmodul definiert.

Darüber hinaus besitzt die Managerklasse die private Methode *applyFallbacks*, mit dessen Hilfe sich gewünschte Ausweichmechanismen auf die zugrunde liegenden Farbtransformationseinstellungen anwenden lassen. Dies ist im Fall einer unvollständigen, fehlerhaften oder inkonsistenten Parameterbelegung erforderlich.

Der sequenzielle Ablauf einer Farbtransformation beginnt grundsätzlich mit der Validierung der Eingangsdaten. Je nach Parametrisierung erfolgt anschließend der Prozess der Geräte- bzw. Betriebssystemerkennung.

Liegen einer Farbtransformation Bilddaten zugrunde, wird anschließend geprüft, ob ein eingebettetes Farbprofil vorhanden ist bzw. extrahiert werden soll. Danach folgt eine Dekodierung der Bilddaten in pixelbasierte Farbwerte.

Die Pixeldaten sowie die benötigten Parameter werden anschließend der Klasse *ColorTransformationManager* übergeben. Diese prüft die Parametrisierung auf Vollständigkeit, Konsistenz sowie Kompatibilität mit dem angegebenen Farbmanagementmodul und wendet mit Hilfe der Methode *applyFallbacks* gegebenenfalls notwendige Ausweichmechanismen an.

Im nächsten Schritt werden die Farbdaten sowie die -transformationseinstellungen an die Methode *transform* der jeweiligen CMM-Instanz weitergeleitet und transformiert. Die Auswahl des CMM wird über die angegebene bzw. evaluierte Parametrisierung festgelegt. Bei zugrunde liegenden Bilddaten werden die transformierten Pixeldaten anschließend wieder in ihr ursprüngliches Bildformat überführt (vgl. Abb. 4-28).

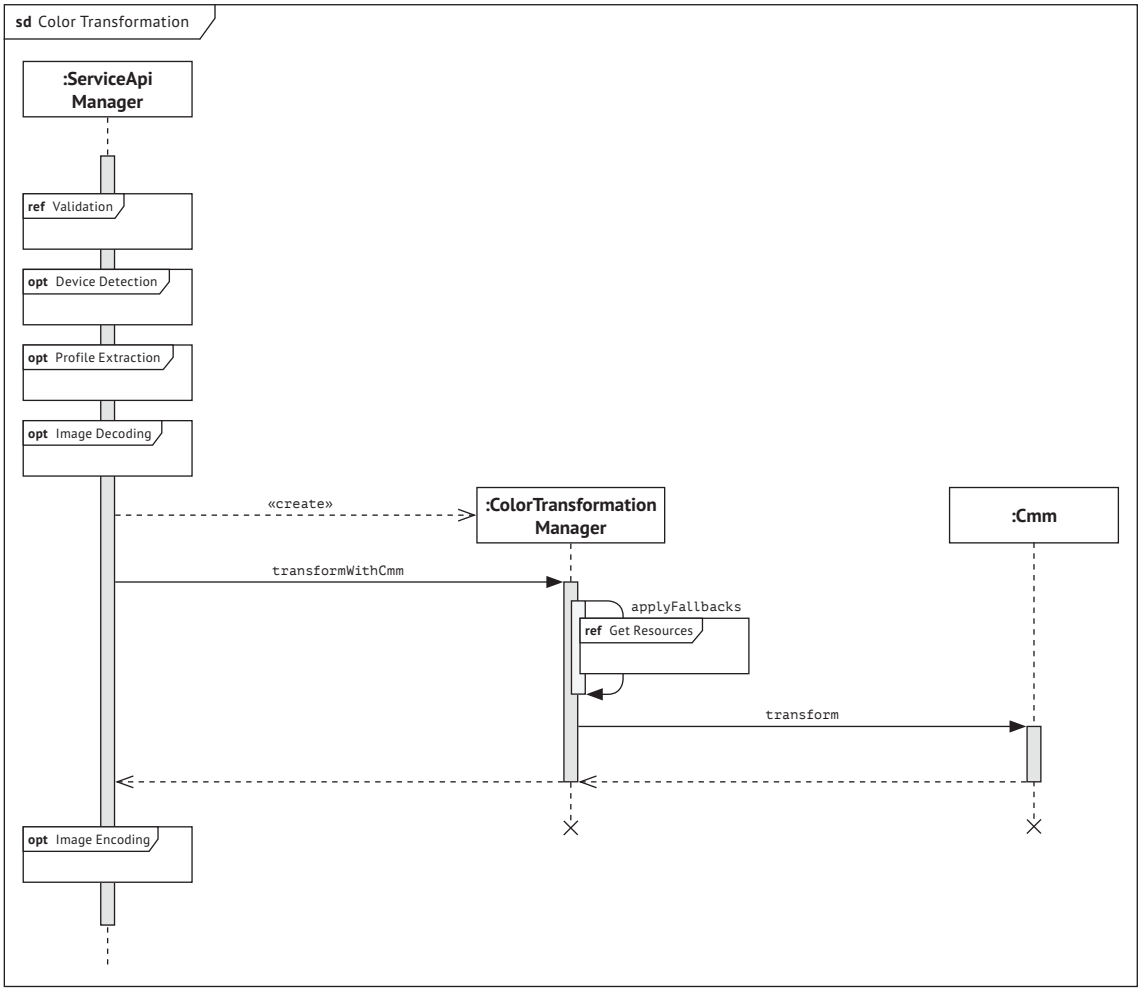


Abbildung 4-28: Sequenzdiagramm des Farbtransformationsprozesses

4.4. Datenzugangsschicht

Die Datenzugangsschicht verbindet die Applikationslogik des Systems mit einer oder mehreren physischen Datenbanken und verhindert somit einen starren Verbund beider Schichten. Sie dient der Verwaltung und insbesondere dem Abrufen von persistenten Ressourcen. In den vorherigen Abschnitten wurde auf diese Komponente bereits mehrfach referenziert.

Die Verwaltungsfunktionen der Komponente *DataAccess* werden über eine Schnittstelle zur Verfügung gestellt und berücksichtigen das Auslesen, Hinzuzufügen, Aktualisieren und Löschen von Datensätzen. Diese Funktionen werden jeweils von einer datenbankspezifischen Subklasse des Typs *PersistentResourceManager* implementiert.

Durch die Verwendung einer Fabrikmethode, lässt sich die Managerklasse für das entsprechende Datenbankmanagementsystem flexibel anfordern (vgl. Abb. 4-29).

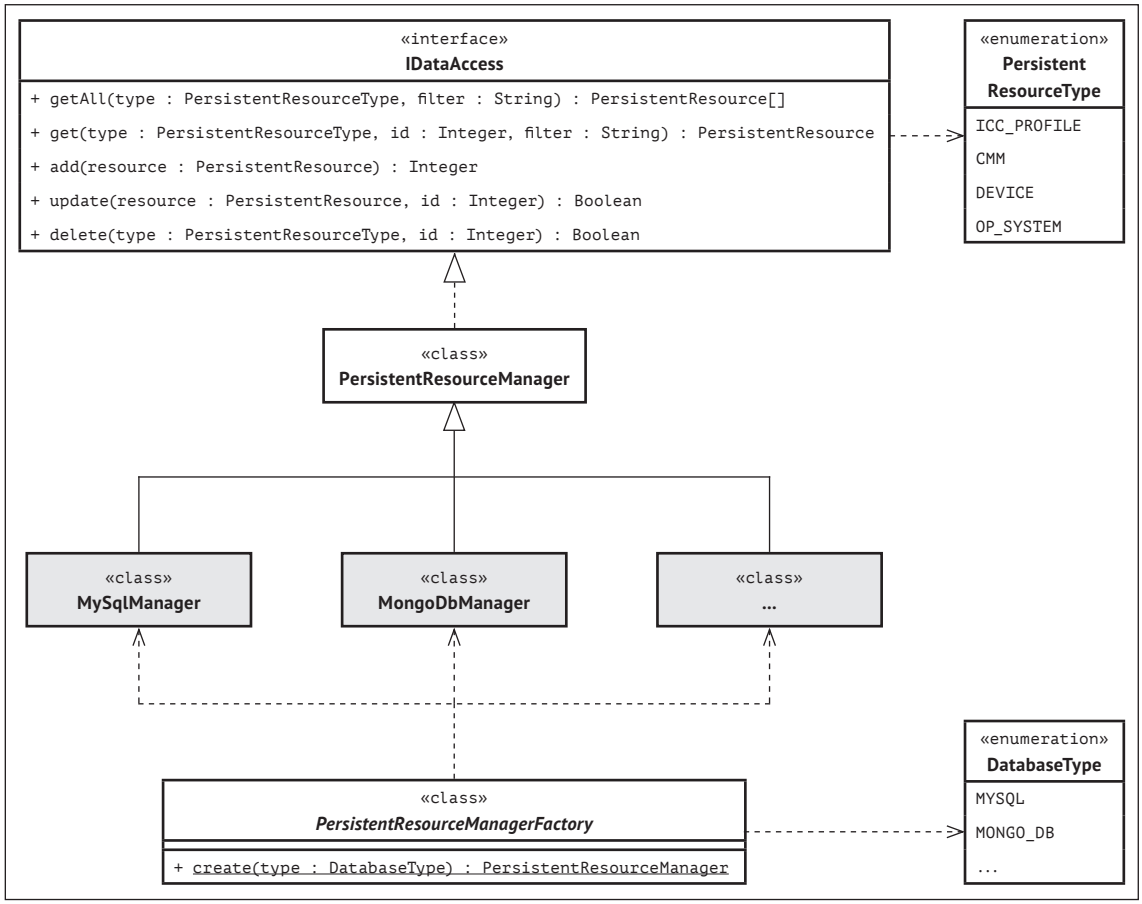


Abbildung 4-29: Klassendiagramm der Komponente »DataAccess«

Beim Auslesen von Ressourcen wird zwischen den Methoden *getAll* und *get* unterschieden. Diese Methoden greifen auf alle Ressourcen eines bestimmten Typen bzw. auf einzelne Ressourcen unter Verwendung einer *id* zu. Beispielsweise lässt sich auf diese Weise die Ausgabe von Endgeräten auf eine spezifische Geräteklasse einschränken oder einzelne Einträge eines Farbprofils herausfiltern (vgl. Abb. 4-30).

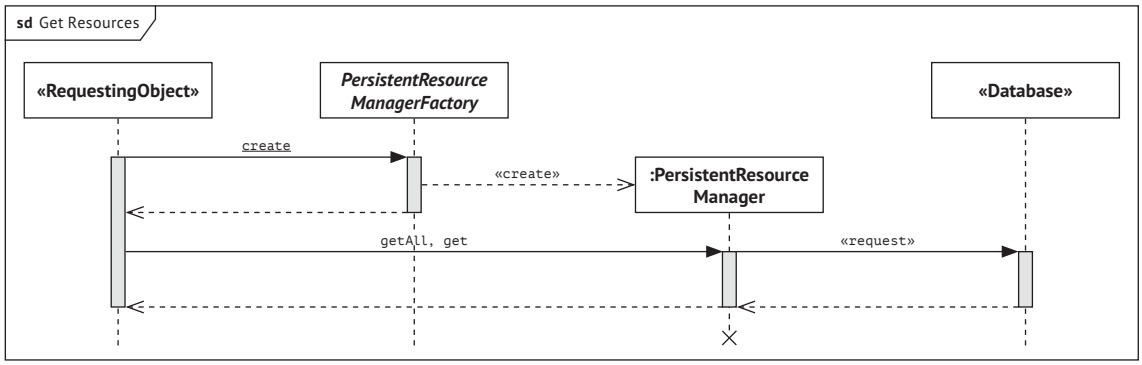


Abbildung 4-30: Sequenzdiagramm für das Abrufen von persistenten Ressourcen

Über die Methode *add* werden Ressourcen zum System hinzugefügt. Als Rückgabewert wird eine *id* erzeugt, unter der die Ressource in der Datenbank hinterlegt wird. Bestehen-

de Ressourcen können mit Hilfe der Methode *update* unter Angabe ihrer *id* aktualisiert werden. Der zurückgegebene Wahrheitswert bestätigt in diesem Fall, ob die Aktion erfolgreich war. Das Auslesen von Ressourcen erfolgt entweder über entsprechende Managerklassen der Applikationsschicht oder direkt über die Serviceschicht.

Das Hinzufügen bzw. Aktualisieren von Ressourcen erfolgt hingegen stets über eine technologiespezifische Klasse der Serviceschicht und setzt sowohl eine Autorisierung des Nutzers als auch eine Validierung der Eingangsdaten voraus. Bei erfolgreicher Absolvierung beider Prozesse, wird aus den Eingangsdaten ein entsprechendes Ressourcenobjekt erzeugt (vgl. Abb. 4-3) und zur angebundenen Datenbank hinzugefügt (vgl. Abb. 4-31).

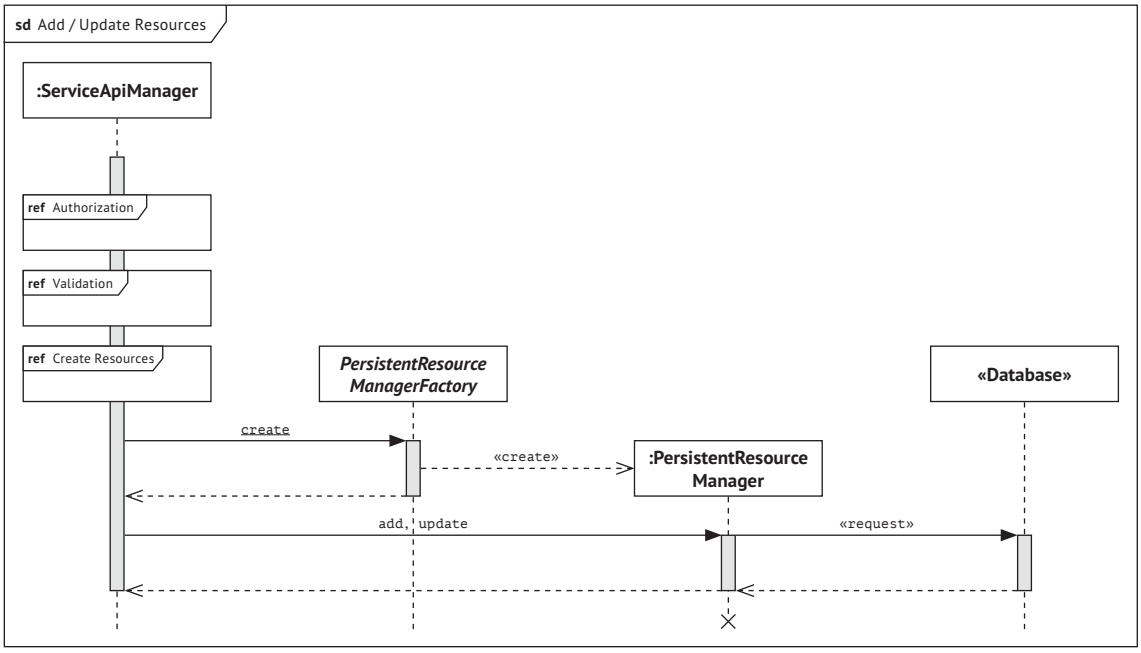


Abbildung 4-31: Sequenzdiagramm für das Hinzufügen und Aktualisieren von Ressourcen

Das Löschen von Ressourcen erfolgt unter Verwendung der Methode *delete*. Diese erwartet die Angabe des zu löschenden Ressourcentyps sowie dessen *id*. Dieser Prozess wird ebenfalls durch eine Subklasse vom Typ *ServiceApiManager* initiiert und setzt eine erfolgreiche Autorisierung des Benutzers voraus (vgl. Abb. 4-32).

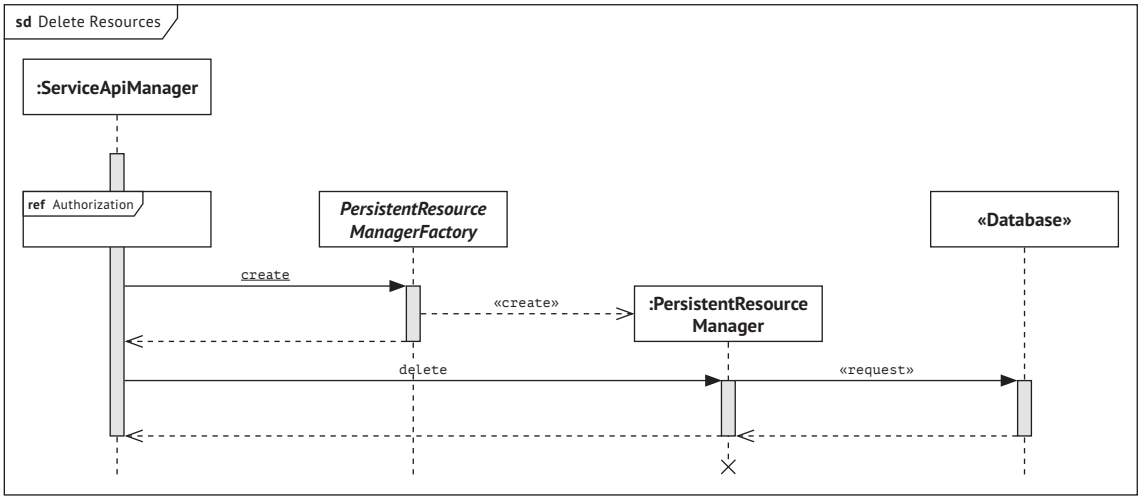


Abbildung 4-32: Sequenzdiagramm für das Löschen von persistenten Ressourcen

5 Vorschlag einer REST-basierten API

In diesem Kapitel wird der Entwurf einer REST-basierten Programmierschnittstelle für den beschriebenen Architekturvorschlag vorgestellt und im Bezug auf die zentralen Anwendungsfälle Farbkalibrierung und Farbtransformation sowie die Ressourcenverwaltung erläutert.

Hierzu werden exemplarische HTTP-basierte Anfragen und Antworten dargestellt, die jeweils einem einheitlichen Schema folgen (vgl. Abb 5-1). Aufgrund des Verbreitungsgrads liegt HTTP/1.1 als Protokollversion zugrunde.

Abbildung 5-1: Schema für die Darstellung von HTTP-Listings

Jeder Anwendungsfall besitzt individuelle Parametersätze, die hauptsächlich aus Pfad- und vereinzelt aus Query-Parametern bestehen. Darüber hinaus existieren drei allgemeine Header-Parameter, die in mehreren Anfragen zum Einsatz kommen (vgl. Tab. 5-1).

Parameter	Datentyp	Beschreibung
`Authorization`	`String`	Autorisierungschlüssel des Benutzers
`User-Agent`	`String`	Identifikationsschlüssel des aufrufenden Clients
`Host`	`String`	Domainname des Servers

Tabelle 5-1: Liste der allgemeinen Header-Parameter

Der Parameter *Authorization* beinhaltet einen Schlüssel, mit dessen Hilfe sich ein Benutzers für die Nutzung bestimmter Funktionen des Farbmanagementsystems autorisiert. In den nachfolgenden Beispielen wird zum besseren Verständnis das leichtgewichtige Verfahren *HTTP Basic Auth* angewendet. Hierbei werden Benutzername und Passwort als Base64-kodierte Zeichenkette übertragen.

Der Parameter *User-Agent* dient als Identifikationsschlüssel des aufrufenden Clients. Unter Verwendung eines Device Detection Repositories, lassen sich mit Hilfe dieses Parameters Angaben zu Gerätehersteller und -modell sowie des zugrunde liegenden Betriebssystems ermitteln.

Der Parameter *Host* ist in allen Anfragen obligatorisch und gibt den Domainnamen des Servers an. In den folgenden Beispielen wird die Domain *www.example.com* gewählt.

Auf den Domainnamen folgt im weiteren Verlauf grundsätzlich der Wurzelpfad *(cms)* sowie die Version der Programmierschnittstelle (*v1)*. Die Formulierung der Pfade richtet sich nach aktuell geläufigen Gestaltungsrichtlinien REST-basierter Schnittstellen (vgl. [Mas11]).

5.1. Farbkalibrierung

Das Anfragen von Farbkalibrierungseinstellungen für ein Endgerät erfolgt unter Verwendung der Methode `GET` in Kombination mit dem Pfadparameter *calibrate*. Diese Anfrage lässt sich durch zwei optionale Query-Parameter erweitern (vgl. Tab. 5-2).

Abfrage	Datentyp	Beschreibung
`profile_id`	`Integer`	Explizite Angabe des zu verwendenden Farbprofils
`luminance`	`Float`	Angabe eines absoluten Luminanzwerts in cd/m²

Tabelle 5-2: Optionale Query-Parameter einer Farbkalibrierung

Durch Angabe des Parameters *profile_id* wird das zugrunde liegende Farbprofil explizit ausgewählt. Dies ist insbesondere bei clientseitiger Geräte- und Betriebssystemerkennung sinnvoll. Ohne diese Angabe erfolgt eine automatische Zuweisung des Farbprofils.

Der Parameter *luminance* ermöglicht die Angabe eines zu erzielenden Luminanzwerts in cd/m². Hierzu wird für den zugehörigen Kalibrierungsparameter des Endgeräts (üblicherweise der Helligkeitsparameter) der Wert ermittelt, der bei der Ausgabe den angeforderten Luminanzwert erzielt.

Im nachfolgenden Beispiel wird eine exemplarische Anfrage ohne Query-Parameter dargestellt. Die Identifikation des aufrufenden Endgeräts sowie des zugrunde liegenden Betriebssystems erfolgt mit Hilfe des Header-Parameters *User-Agent*. Durch die ermittelten Angaben kann jeweils die *id* von Endgerät und Betriebssystem und somit die *id* des zugehörigen Farbprofils festgestellt werden. (vgl. List. 5-1).

```
1  GET /cms/v1/calibrate HTTP/1.1
2  Host: www.example.com
3  User-Agent: Mozilla/5.0 (Linux; Android 6.0; SAMSUNG SM-G930F Build/MMB29K)
```

```
4  HTTP/1.1 200 OK
5  Content-Length: 110
6  Content-Type: application/json
7  {
8     "calibration": [
9     {
10       "name": "screenMode",
11       "value": ["native"]
12    },
13    {
14       "name": "brightness",
15       "value": [1]
16    }]
17 }
```

Listing 5-1: Anfrage und Antwort einer Farbkalibrierung ohne Query-Parameter

Im oberen Beispiel liefert die Antwort zwei Kalibrierungsparameter im JSON-Format zurück, die der ausführenden Applikation mitteilen, sowohl den nativen Darstellungsmodus als auch eine Displayhelligkeit von 100% einzustellen.

Eine erfolgreiche Kalibrierungsanfrage wird durch den Status *»200 OK«* bestätigt (vgl. List. 5-1, Zeile 4). Fehlermeldungen werden hingegen mit einem Statuscode im Bereich von 400 bis 417 angegeben. Die Definition der einzelnen Statuscodes ist in Anhang A ersichtlich.

Die in Listing 5-2 dargestellte Anfrage fordert eine Farbkalibrierung auf Basis eines definierten Farbprofils sowie einer absoluten Luminanz in Höhe von 320 cd/m^2 an. Die Angabe des Parameters *User-Agent* ist in diesem Fall nicht erforderlich.

```
1  GET /cms/v1/calibrate?profile_id=13&luminance=320 HTTP/1.1
2  Host: www.example.com
```

```
3  HTTP/1.1 200 OK
4  Content-Length: 64
5  Content-Type: application/json
6  {
7     "calibration": [{
8        "name": "brightness",
9        "value": [0.597]
10    }]
11 }
```

Listing 5-2: Anfrage und Antwort einer Farbkalibrierung mit Query-Parametern

Die zugehörige Antwort gibt einen entsprechenden Helligkeitswert in Höhe von 59,7% zurück. Weitere Einstellungen sind in diesem Beispiel nicht erforderlich. Je nach Betriebssystemumgebung können Anzahl, Bezeichnung und Wertebereich der zurückgegebenen Kalibrierungsparameter variieren.

Das Ergebnis einer Helligkeitskalibrierung wird in nachfolgender Vergleichsabbildung zweier Displays mit unterschiedlicher maximaler Leuchtstärke veranschaulicht.

Abbildung 5-2: Veranschaulichung einer Helligkeitskalibrierung am Beispiel eines IPS-LCD- und OLED-Displays[1]

Das rechte Motiv stellt ein IPS-LCD mit ca. 555 cd/m² Luminanz dar, das linke Motiv hingegen ein OLED-Display mit ca. 320 cd/m². Auf dem mittleren Motiv wurde die Luminanz des IPS-LCD an den Luminanzwert des OLED-Displays angeglichen. Da in diesem Beispiel noch keine Farbtransformation stattgefunden hat, weichen die Darstellungen in puncto Farbton und Farbsättigung nach wie vor voneinander ab.

Die einzelnen Interaktionsschritte zwischen Client und Server im Rahmen einer Farbkalibrierung werden im nachfolgenden Aktivitätsdiagramm dargestellt (vgl. Abb. 5-3).

1 Diese und alle weiteren, dargestellten Motive wurden mit einer digitalen Spiegelreflexkamera unter folgenden Bedingungen aufgenommen: ISO 400, Belichtungszeit 1/125 s, Blende *f*/5.6, Umgebungslicht ca. 30 cd/m²

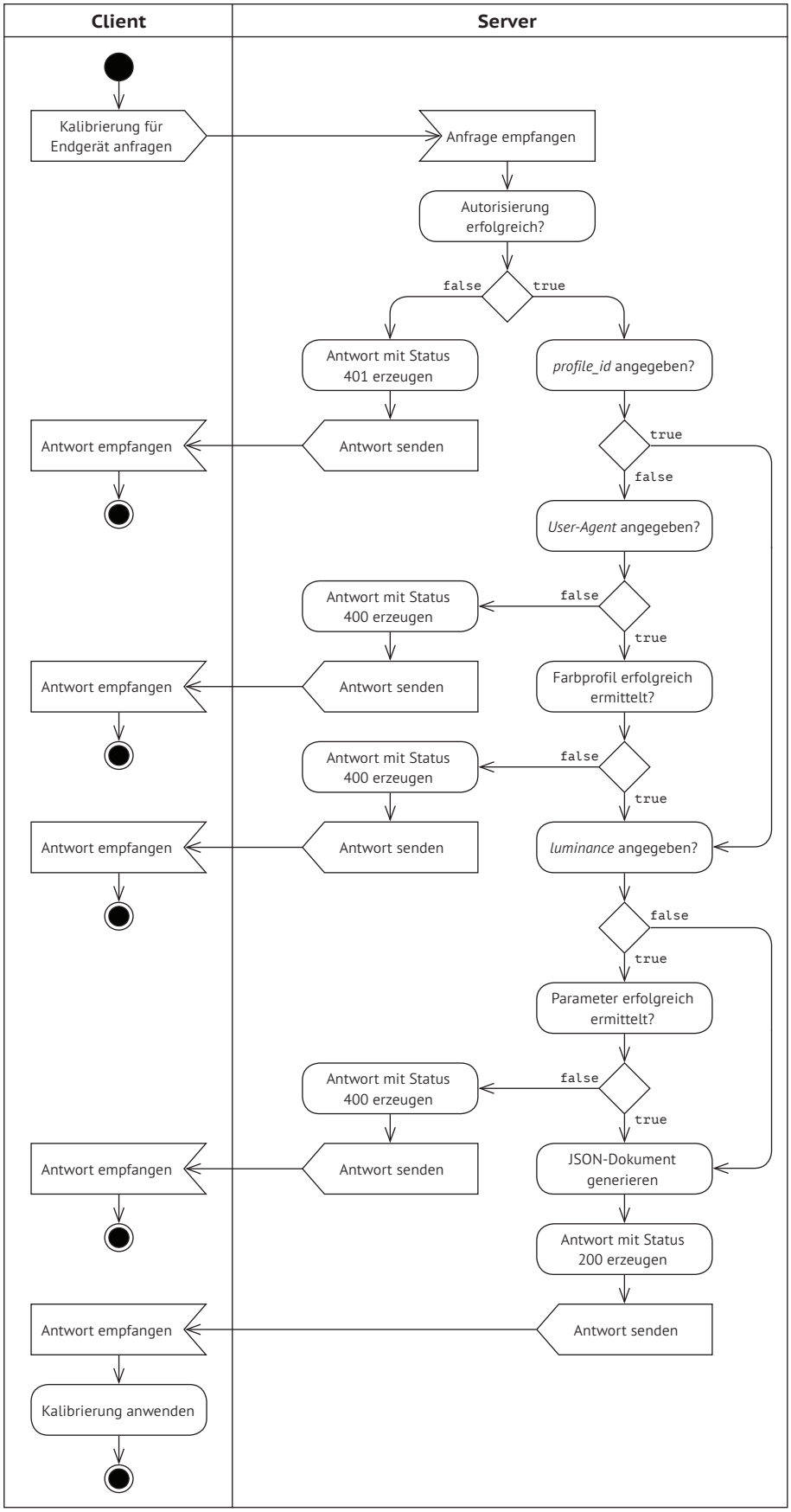


Abbildung 5-3: Interaktion zwischen Client und Server im Rahmen der Farbkalibrierung

5.2. Farbtransformation

Sobald sich ein Endgerät in einem kalibrierten Referenzzustand befindet, können die darzustellenden Farbinhalte mit Hilfe einer Farbtransformation für den entsprechenden Ausgabezweck aufbereitet werden.

Da bei einer Farbtransformation digitale Inhalte an das Farbmanagementsystem übertragen werden, erfolgt die Anfrage unter Verwendung der Methode `POST`. Der zugehörige Pfadparameter lautet *transform*.

Die Schnittstelle unterscheidet zwischen einer Farbtransformation von reinen Pixelwerten sowie von Bilddateien. Die Transformation von Pixelwerten dient unter anderem der farblichen Anpassung von Farbpaletten für Seitenelemente wie z.B. Text oder Hintergrundflächen. Sie wird über den Pfadzusatz *pixels* adressiert.

Bei Bilddateien setzt sich der Pfad aus den variablen Komponenten *transfer* und *mimetype* zusammen, die einerseits festlegen, auf welchem Weg eine Datei zum Farbmanagementsystem übertragen wird und andererseits den zu transformierenden Dateityp angeben (vgl. Tab. 5-3).

Methode	Pfad	Beschreibung
`POST`	`/cms/v1/transform/pixels`	Farbtransformation von Pixelwerten
`POST`	`/cms/v1/transform/{transfer}/{mime-type}`	Farbtransformation von Bilddateien

Tabelle 5-3: Pfad- und Methodentabelle der Farbtransformation

Hinsichtlich der Übertragungsmethode unterschiedet die Programmierschnittstelle, ob eine Datei von einer externen Quelle referenziert oder auf direktem Weg zum Farbmanagementsystem übertragen wird. Die Referenzierung externer Quellen erfolgt mittels URL unter Angabe des Parameters *href*, eine direkte Übertragung als binärer Datenstrom wird durch den Parameter *binary* gekennzeichnet (vgl. Tab.5-4).

Parameter	Beschreibung
`href`	Angabe einer Referenz mittels URL
`binary`	Direkte Übertragung von Binärdaten

Tabelle 5-4: Übertragungsparameter der Farbtransformation

Bei einer Farbtransformation werden neben den zu transformierenden Farbdaten auch Transformationseinstellungen übermittelt. Der Anfragerumpf wird daher in sogenannte Multiparts gegliedert. Die Übermittlung von textbasierten Inhalten findet im JSON-Format statt, Binärdaten verbleiben in ihrer ursprünglichen Kodierungsform.

5.2.1. Pixelwerte

Die Anfrage einer Pixeltransformation gliedert sich in drei Abschnitte. Der Anfragekopf enthält die Anfragezeile, den obligatorischen Host-Parameter sowie die Angabe des *User-Agent*. Der Anfragerumpf wird in zwei Abschnitte unterteilt und besteht zum einen aus den zu transformierenden Pixelwerten, zum anderen aus den zugehörigen Transformationseinstellungen (vgl. List. 5-3).

```
1  POST /cms/v1/transform/pixels HTTP/1.1
2  Host: www.example.com
3  User-Agent: Mozilla/5.0 (Linux; Android 6.0; SAMSUNG SM-G930F Build/MMB29K)
4  Content-Disposition: form-data; name="pixels"; filename="rgb-palette.json"
5  Content-Type: application/json
6  {
7     "pixels": [[200, 0, 0], [0, 100, 200]],
8     "colorEncoding": {
9        "colorScheme": "RGB",
10       "bitsPerChannel": "8"
11    }
12 }
13 Content-Disposition: form-data; name="settings"; filename="settings.json"
14 Content-Type: application/json
15 {
16    "inputProfileId": 1
17 }

18 HTTP/1.1 200 OK
19 Content-Length: 41
20 Content-Type: application/json
21 {
22    "pixels": [[234, 0, 0], [0, 100, 204]]
23 }
```

Listing 5-3: Farbtransformation von Pixelwerten

Die Angabe der Farbkodierung bestehend aus Farbschema und Farbtiefe, teilt dem Farbmanagementmodul mit, wie die Werte der übergebenen Pixel zu interpretieren sind (vgl. List. 5-3, Zeile 7-10).

Die Parametrisierung der Farbtransformation besteht im oben dargestellten Beispiel lediglich aus der Angabe eines Eingabefarbprofils (vgl. Zeile 16). Diese Angabe ist obligatorisch, da bei reinen Farbwerten keine Farbprofile eingebetten werden können.

Das Ausgabefarbprofil wird mit Hilfe der Geräteerkennung ermittelt. Alle weiteren Einstellungen, wie z.B. das zu verwendende Farbmanagementmodul oder Rendering Intent, können dem Ausgabefarbprofil entnommen werden.

Das Farbmanagementsystem bestätigt eine erfolgreiche Verarbeitung der Anfrage durch den entsprechenden Status und gibt die transformierten Farbwerte entsprechend des Anfrageschemas zurück (vgl. Zeile 22).

5.2.2. Bilddaten

Die Anfrage einer Farbtransformation für Bilddaten gliedert sich ebenfalls in drei Abschnitte. Die Anfragezeile beinhaltet die Übertragungsform sowie den MIME-Type der zu transformierenden Bilddatei, der sich aus der Bezeichnung *image* sowie der Dateiendung (z.B. *png*) zusammensetzt.

Wie eingangs erwähnt, wird im Zusammenhang mit Bilddaten zwischen applikationsexternen und -internen Quellen unterschieden. Externe Quellen werden durch die Angabe *href* gekennzeichnet. Das Farbmanagementsystem ruft die Bildquelle unter der angegebenen URL ab (vgl. List. 5-4).

```
1  POST /cms/v1/transform/href/image/png HTTP/1.1
2  Host: www.example.com
3  User-Agent: Mozilla/5.0 (Linux; Android 6.0; SAMSUNG SM-G930F Build/MMB29K)
4  Content-Disposition: form-data; name="href"; filename="image.json"
5  Content-Type: application/json
6  {
7     "href": "http://www.domain.com/path/to/image.png"
8  }
9  Content-Disposition: form-data; name="settings"; filename="settings.json"
10 Content-Type: application/json
11 {
12    "parameterMap": {
13       "getLog": true
14    }
15 }
```

Listing 5-4: Farbtransformation auf Basis einer Bildreferenz (Anfrage)

Die übermittelten Transformationseinstellungen (Zeile 9-15) beinhalten im oben dargestellten Beispiel lediglich den Eintrag *getLog*, um der Antwort eine Zusammenfassung der evaluierten Transformationseinstellungen beizufügen. Auf diese Weise lässt sich das Zustandekommen einzelner Parameter nachvollziehen, beispielsweise welche Parameter automatisch ermittelt wurden bzw. ob ggf. Fallbackmechanismen zum Einsatz gekommen sind. Diese Einstellung dient lediglich dem Debugging und wird in Produktionsumgebungen deaktiviert.

```
HTTP/1.1 200 OK
Content-Length: 358
Content-Type: application/json
{
   "href": "http://www.example.com/path/to/transformed/image.png",
   "log": {
      "inputProfile": "Detected: Adobe RGB (1998)",
      "outputProfile": "Detected: Samsung Galaxy S7 (Android 6.0)",
      "renderingIntent": "Detected: Relative Colorimetric",
      "cmm": "Fallback: Little CMS",
      "parameterMap": {
         "bpc": "Fallback: false",
         "quality": "Detected: Normal",
         "getLog": true
      }
   }
}
```

Listing 5-5: Farbtransformation auf Basis einer Bildreferenz (Antwort)

Das Farbmanagementsystem bestätigt eine erfolgreiche Verarbeitung der Anfrage wiederum mit dem entsprechenden Statuscode (vgl. List. 5-5). Da es sich bei der ursprünglichen Bildquelle um eine Referenz handelt, wird die transformierte Datei ebenfalls als URL zur Verfügung gestellt. Der anschließende Log-Eintrag fasst die verwendeten Parameter der Farbtransformation zusammen und gibt dabei an, ob ein Parameter automatisch ermittelt wurde oder ein Fallbackmechanismus zum Einsatz kam.

Die Farbtransformation von applikationsinternen Bilddaten erfolgt hingegen durch binäre Datenübertragung. Hierzu werden die zugrunde liegenden Binärdaten im Rumpf der Anfrage bzw. der Antwort eingefügt (vgl. List. 5-6). Bei binären Bilddaten findet die Validierung im Rahmen der Dekodierung in der Applikationsschicht statt.

```
POST /cms/v1/transform/binary/image/png HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Linux; Android 6.0; SAMSUNG SM-G930F Build/MMB29K)
Content-Disposition: form-data; name="binary"; filename="image.png"
Content-Type: image/png
                                    <binary data>
                                         ...
HTTP/1.1 200 OK
Content-Length: 3495
Content-Type: image/png
                                    <binary data>
```

Listing 5-6: Farbtransformation auf Basis einer binär übertragenen Bilddatei

Das Ergebnis einer Farbtransformation wird in der nachfolgenden Vergleichsabbildung veranschaulicht. Das rechte Motiv stellt ein IPS-LCD mit sRGB-Farbraum dar, das linke Motiv hingegen ein OLED-Display mit nativer, deutlich gesättigterer Farbdarstellung. Auf dem mittleren Motiv wurde die Farbdarstellung des OLED-Displays zu sRGB transformiert. Die Farbabweichung zwischen dem mittleren und dem rechten Motiv ist nach der Farbtransformation erkennbar geringer.

Abbildung 5-4: Veranschaulichung der Farbtransformation am Beispiel eines IPS-LCD- und OLED-Displays

Die einzelnen Interaktionsschritte zwischen Client und Server im Rahmen einer Farbtransformation werden im nachfolgenden Aktivitätsdiagramm dargestellt (vgl. Abb. 5-5).

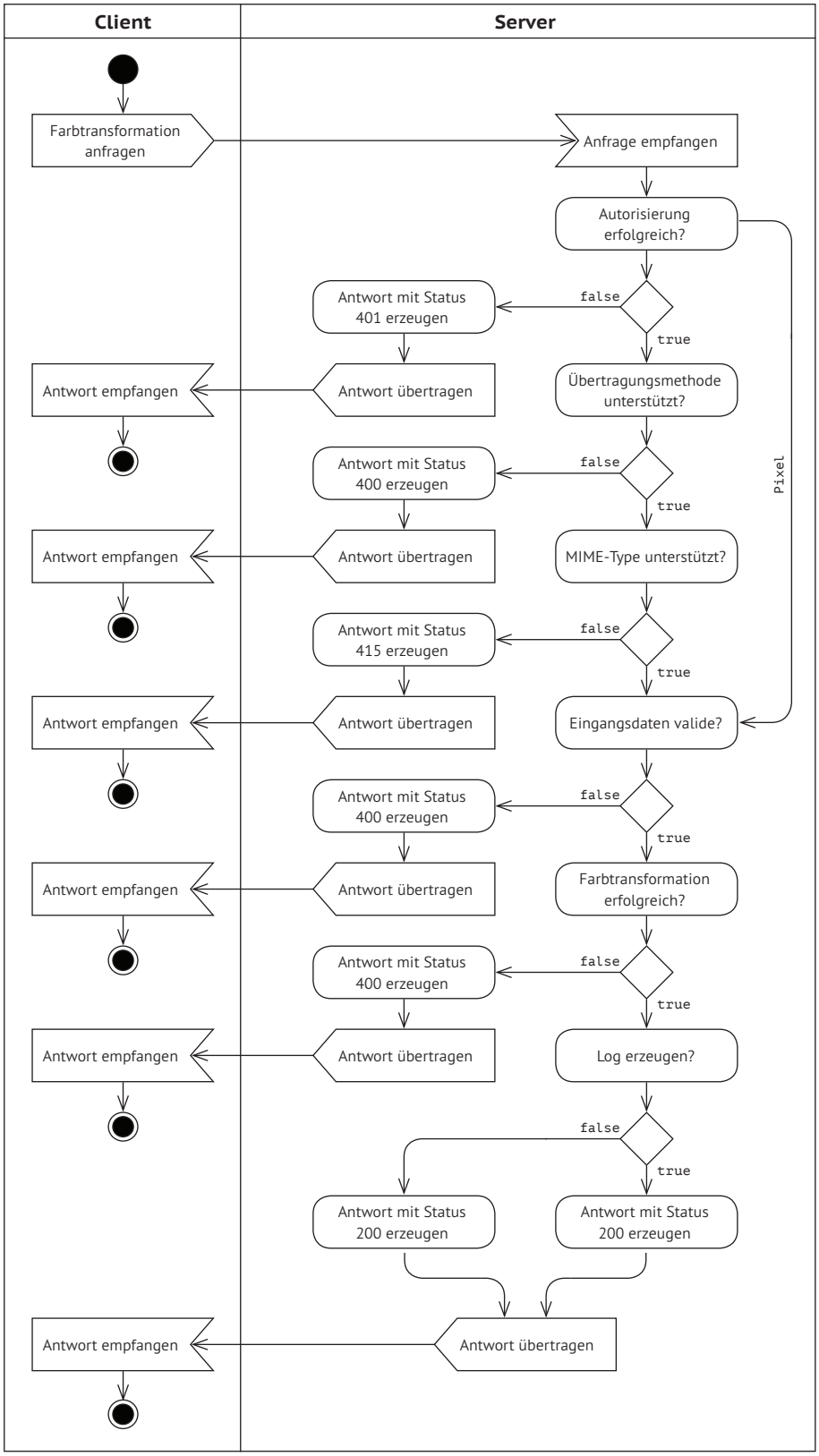


Abbildung 5-5: Interaktion zwischen Client und Server im Rahmen der Farbtransformation

5.3. Ressourcenverwaltung

Die Ressourcenverwaltung beinhaltet das Ausgeben, Hinzufügen, Aktualisieren und Entfernen von Ressourcen. Für jeden dieser Fälle existiert eine zugehörige HTTP-Methode, die jeweils mit einem entsprechenden Ressourcenpfad kombiniert wird. Die nachfolgenden vier Abschnitte veranschaulichen die in Tabelle 5-5 dargestellten Anfragen anhand einiger Beispiele.

Path-Parameter	GET	POST	PUT/PATCH	DELETE
`/cms/v1/profiles`	Farbprofile auflisten	Farbprofil hinzufügen	–	–
`/cms/v1/profiles/{filter}`	Farbprofile gefiltert auflisten	–	–	–
`/cms/v1/profiles/{id}`	Farbprofil ausgeben	–	Farbprofil aktualisieren	Farbprofil löschen
`/cms/v1/profiles/{id}/{filter}`	Farbprofil gefiltert ausgeben	–	–	–
`/cms/v1/cmms`	CMM auflisten	CMM hinzufügen	–	–
`/cms/v1/cmms/{filter}`	CMM gefiltert auflisten	–	–	–
`/cms/v1/cmms/{id}`	CMM ausgeben	–	CMM aktualisieren	CMM löschen
`/cms/v1/cmms/{id}/{filter}`	CMM gefiltert ausgeben	–	–	–
`/cms/v1/devices`	Endgeräte auflisten	Endgeräte hinzufügen	–	–
`/cms/v1/devices/{filter}`	Endgeräte gefiltert auflisten	–	–	–
`/cms/v1/devices/{id}`	Endgeräte ausgeben	–	Endgeräte aktualisieren	Endgeräte löschen
`/cms/v1/devices/{id}/{filter}`	Endgeräte gefiltert ausgeben	–	–	–
`/cms/v1/systems`	Betriebssysteme auflisten	Betriebssysteme hinzufügen	–	–
`/cms/v1/systems/{filter}`	Betriebssysteme gefiltert auflisten	–	–	–
`/cms/v1/systems/{id}`	Betriebssysteme ausgeben	–	Betriebssysteme aktualisieren	Betriebssysteme löschen
`/cms/v1/systems/{id}/{filter}`	Betriebssysteme gefiltert ausgeben	–	–	–

Tabelle 5-5: Pfad- und Methodentabelle der Ressourcenverwaltung

5.3.1. Ausgeben von Ressourcen

Unter Verwendung der Methode `GET` können Ressourcen eines Typs aufgelistet sowie einzelne Ressourcen ausgegeben werden. Für das Auflisten von Datensätzen, richtet sich die Anfrage grundsätzlich an den Ressourcenpfad des aufzulistenden Typs (vgl. List. 5-7).

```
GET /cms/v1/profiles HTTP/1.1
Host: www.example.com
```

```
HTTP/1.1 200 OK
Content-Length: …
Content-Type: application/json
{
   1: ["Adobe RGB (1998)", "Generic Color Space"],
   2: ["sRGB (IEC61966-2.1)", "Generic Color Space"],
   3: ["DCI-P3 (SMPTE-RP431-2)", "Generic Color Space"],
   …
}
```

Listing 5-7: Listenausgabe von Ressourcen

Die oben dargestellte Anfrage fordert das Farbmanagementsystem auf, eine Liste aller im System hinterlegten Farbprofile auszugeben. Der Server antwortet auf diese Frage mit dem entsprechenden Statuscode und gibt die Liste als JSON-Struktur zurück.

Falls bei einer Listenanfrage ausschließlich Farbprofile einer bestimmten Kategorie von Interesse sein sollten, lässt sich die Anfrage mit einer Erweiterung des Ressourcenpfads präzisieren. Das nachfolgende Listing verdeutlicht diesen Anwendungsfall am Beispiel der Ausgabe von Farbprofilen einer bestimmten Kategorie.

```
GET /cms/v1/profiles/category/specific_device_display HTTP/1.1
Host: www.example.com
```

```
HTTP/1.1 200 OK
Content-Length: …
Content-Type: application/json
{
   10: ["Apple iPad 1. Generation", …, 1, 4, "Specific Device Display", …, …, …],
   11: ["Apple iPad 2. Generation", …, 2, 4, "Specific Device Display", …, …, …],
   12: ["Apple iPad 3. Generation", …, 3, 4, "Specific Device Display", …, …, …],
   …
}
```

Listing 5-8: Gefilterte Listenausgabe von Ressourcen

Durch die Erweiterung der Anfragezeile um zusätzliche Pfadparameter, wird die Ausgabe auf Farbprofile der Kategorie *Specific Device Display* eingeschränkt. Das Ausgeben von Listen dient insbesondere der tabellarischen Darstellung von Ressourcen.

Einzelne Ressourcen lassen sich durch die Angabe der entsprechenden *id* innerhalb des Ressourcenpfads adressieren. In der Ausgabe übersetzen ICC-Profile ihre binärkodierten Bestandteile Header, Tag-Table und Tagged-Element-Data in eine textbasierte Darstellung (vgl. List. 5-9).

```
GET /cms/v1/profiles/10 HTTP/1.1
Host: www.example.com
```

```
HTTP/1.1 200 OK
Content-Length: …
Content-Type: application/json
{
   "id": 10,
   "description": "Apple iPad 1. Generation",
   "calibration": […],
   "deviceProfileId": 1,
   "systemProfileId": 4,
   "category": "Specific Device Display",
   "header": […],
   "tagTable": […],
   "taggedElementData": […]"
}
```

Listing 5-9: Ausgabe einzelner Ressourcen

Die Ausgabe einzelner Ressourcen lässt sich ebenso filtern, wie die Ausgabe von Listen. Hierzu wird der Ressourcenpfad um einen entsprechenden Pfadparameter erweitert. Auf diese Weise lässt sich die Übertragung nicht benötigter Datenmengen vermeiden. Im folgenden Beispiel wird die Ausgabe auf den Header eines Farbprofils beschränkt (vgl. List. 5-10).

```
GET /cms/v1/profiles/1/header HTTP/1.1
Host: www.example.com
```

```
HTTP/1.1 200 OK
Content-Length: …
Content-Type: application/json
{
   "header": [
      {
         "Size": 560
      },
      {
         "CMM": "ADBE"
      },
      {
         "Specification": "2.1.0"
      },
      {
         "Class": "mntr"
      }, ...
   ]
}
```

Listing 5-10: Gefilterte Ausgabe einzelner Ressourcen

5.3.2. Hinzufügen von Ressourcen

Während sich Anfragen zur Ausgabe von Ressourcen auf die Übertragung eines Anfragekopfs beschränken, wird für die Übertragung von hinzuzufügenden Datensätzen ein zusätzlicher Anfragerumpf benötigt.

Werden beim Hinzufügen eines Datensatzes zwei oder mehr Dokumente übertragen, wird der Anfragerumpf mit Hilfe von Multiparts in mehrere Abschnitte unterteilt. Dies ist z.B. beim Hinzufügen von ICC-Profilen notwendig, da sowohl das binäre Farbprofil als auch ein textbasiertes Dokument mit zusätzlichen Informationen übertragen wird (vgl. List. 5-11).

```
1  POST /cms/v1/profiles HTTP/1.1
2  Host: www.example.com
3  Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
4  Content-Disposition: form-data; name="icc-profile-binary"; filename="profile.icc"
5  Content-Type: application/vnd.iccprofile
6                                        <binary data>
7  Content-Disposition: form-data; name="profile"; filename="profile.json"
8  Content-Type: application/json
9  {
10    "description": "HTC One (M8) Display Profile",
11    "calibration": [...],
12    "deviceId": 31,
13    "systemId": 12,
14    "category": "Specific Device Display",
15 }

16 HTTP/1.1 201 Created
17 Content-Length: 11
18 Content-Type: application/json
19 {
20    "id": 41
21 }
```

Listing 5-11: Hinzufügen von Ressourcen

Das Hinzufügen von Ressourcen erfolgt mit Hilfe der Methode `POST` und richtet sich an den Pfad des hinzuzufügenden Ressourcentyps. Aus Sicherheitsgründen ist für das Hinzufügen von Datensätzen eine Autorisierung erforderlich (vgl. Zeile 3).

Der Anfragerumpf des oben dargestellten Auszugs besteht aus zwei Abschnitten. Der erste Abschnitt beinhaltet die Binärdaten des zugrunde liegenden Farbprofils (im Beispiel gekennzeichnet durch einen Platzhalter). Der zweite Abschnitt liefert ergänzende Informationen als JSON-Dokument. Diese Informationen ordnen das Farbprofil unter anderem einem Endgerät sowie einem Betriebssystem zu und ergänzen es um etwaige Farbkalibrierungseinstellungen.

Die Antwort des Farbmanagementsystems bestätigt das erfolgreiche Hinzufügen des Datensatzes durch den Status *»201 Created«* und gibt die *id* zurück, unter der das hinzugefügte Farbprofil zu adressieren ist.

5.3.3. Aktualisieren von Ressourcen

Das Aktualisieren von Ressourcen erfolgt durch die Verwendung der Methoden PUT und PATCH. Während die Methode PUT eine bestehende Ressource vollständig ersetzt, wird durch die Methode PATCH lediglich der übertragene Ressourcenteil erneuert. Der entsprechende Ressourcenpfad adressiert jeweils die zu modifizierende Ressource.

```
1  PUT /cms/v1/devices/31 HTTP/1.1
2  Host: www.example.com
3  Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
4  Content-Disposition: form-data; name="device"; filename="device.json"
5  Content-Type: application/json
6  {
7     "description": "HTC One M8",
8     "manufacturer": "HTC",
9     "model": "HTC6525LVW",
10    "category": "Specific Device",
11    "type": "Smartphone"
12 }

13 HTTP/1.1 202 Accepted
```

Listing 5-12: Ersetzen von Ressourcen

In Listing 5-12 wird ein bestehendes Endgerät mit Hilfe des beigefügten JSON-Dokuments vollständig ersetzt. In Listing 5-13 werden hingegen lediglich die angegebenen Informationen aktualisiert. Beide Fälle setzen eine Autorisierung der Anfrage voraus. Das Farbmanagementsystem bestätigt erfolgreiche Anfragen jeweils mit dem Status *»202 Accepted«* und beinhaltet darüber hinaus keine weiteren Rückgabewerte.

```
1  PATCH /cms/v1/devices/31 HTTP/1.1
2  Host: www.example.com
3  Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
4  Content-Disposition: form-data; name="device"; filename="device.json"
5  Content-Type: application/json
6  {
7     "description": "HTC One (M8)",
8  }

9  HTTP/1.1 202 Accepted
```

Listing 5-13: Aktualisieren von Ressourcen

5.3.4. Entfernen von Ressourcen

Das Entfernen von Ressourcen erfolgt mit Hilfe der Methode `DELETE` in Kombination mit dem Ressourcenpfad des zu löschenden Datensatzes. Das Entfernen von Ressourcen setzt ebenfalls eine Autorisierung der Anfrage voraus. Das Farbmanagementsystem bestätigt das Löschen einer Ressource ohne weitere Rückgabewerte mit dem Status *»202 Accepted«* (vgl. List. 5-14).

```
1 DELETE /cms/v1/devices/31 HTTP/1.1
2 Host: www.example.com
3 Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=

4 HTTP/1.1 202 Accepted
```

Listing 5-14: Löschen von Ressourcen

Die einzelnen Interaktionsschritte zwischen Client und Server im Rahmen der Ressourcenverwaltung werden im nachfolgenden Aktivitätsdiagramm dargestellt (vgl. Abb. 5-6).

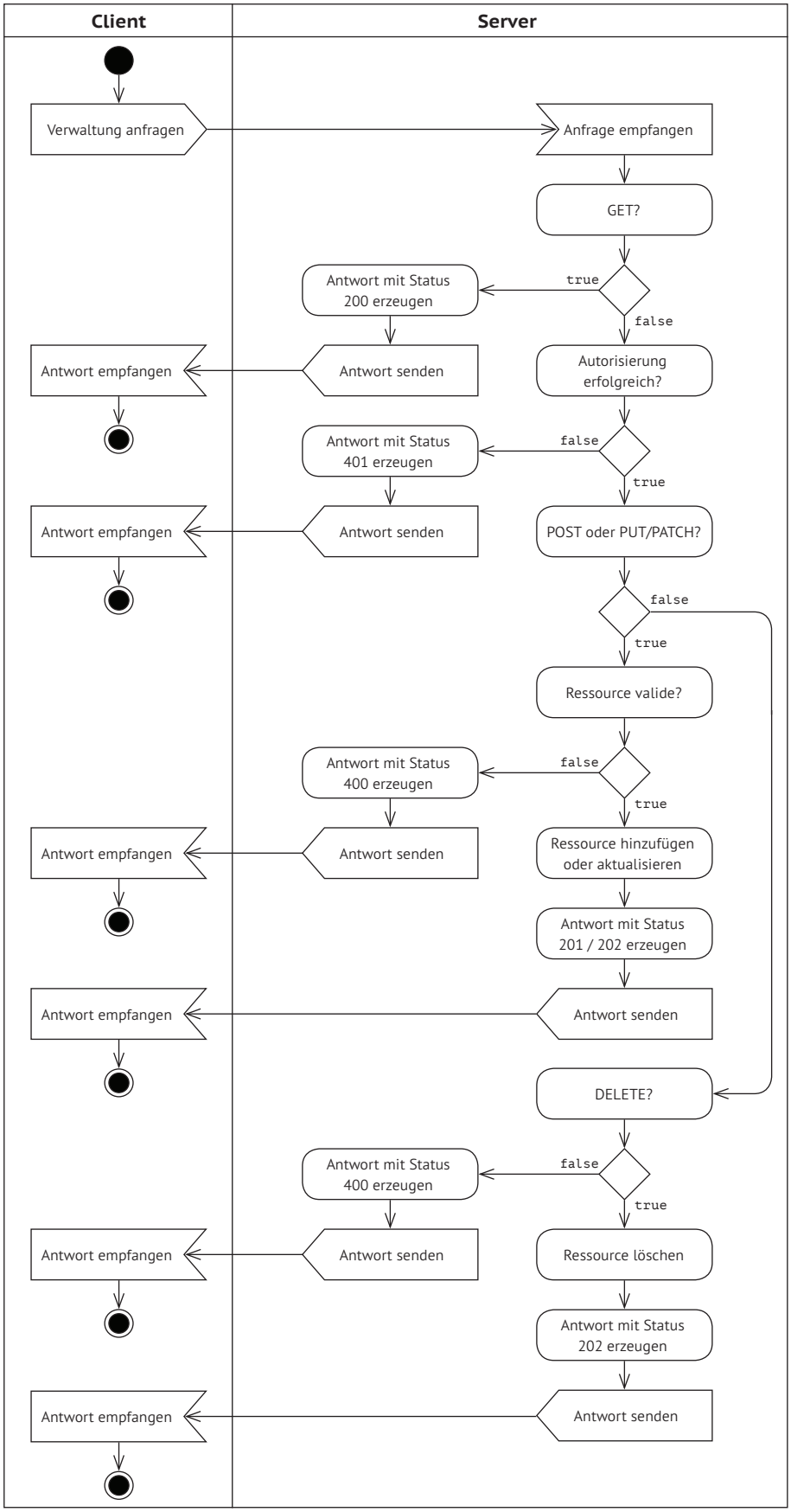


Abbildung 5-6: Interaktion zwischen Client und Server im Rahmen der Ressourcenverwaltung

6 Prototypische Implementierung

Um die technische Realisierbarkeit des Architekturvorschlags sicherzustellen, wurde der abstrakte Entwurf auf Basis konkreter Technologiestandards implementiert. Hierzu wurde ein Application Server mit REST-Schnittstelle aufgesetzt.

Dieser Server kommuniziert mit diversen Peripheriesystemen. Hierzu gehören ein externes Device Detection Repository zum Identifizieren von aufrufenden Endgeräten, ein Datenbankserver zum Speichern von Systemressourcen sowie ein Webserver zum Bereitstellen von farbtransformierten Bildinhalten (vgl. Abb. 6-1).

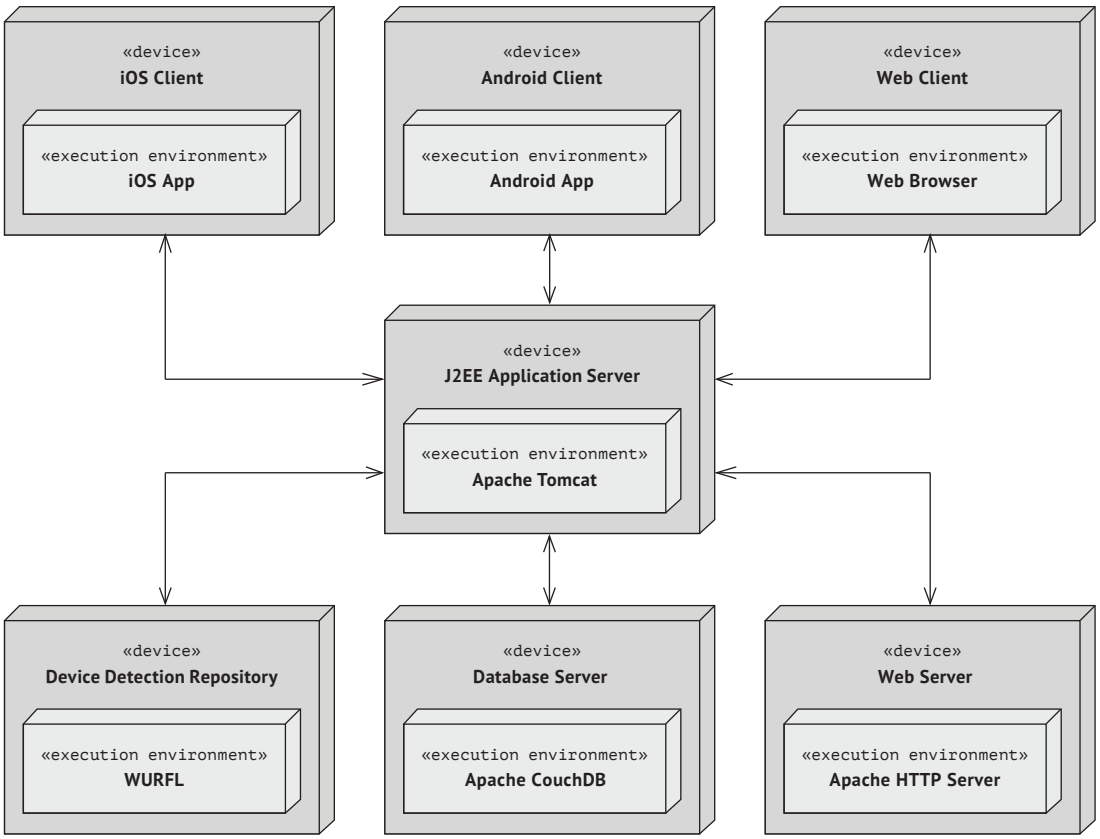


Abbildung 6-1: Verteilungsdiagramm der prototypischen Implementierung

Um die bereitgestellten Funktionen des Application Servers nutzen und demonstrieren zu können, wurden zudem drei verschiedene Clientapplikationen erstellt. Hierzu gehören zwei native Applikationen für die mobilen Plattformen iOS und Android sowie eine Webapplikation für die Nutzung im Browser. Die Webapplikation dient zudem als grafische Benutzeroberfläche für die Verwaltung von Systemressourcen.

6.1. Application Server

Der Application Server stellt den Kern des Backends dar und beinhaltet die implementierten Systemschichten sowie das Datenmodell. Die vorgeschlagene Systemstruktur wurde, entsprechend der Modellierung in Kapitel 3, mit Hilfe von Java (J2EE) umgesetzt. Der Application Server wird mittels Apache Tomcat[1] bereitgestellt.

Die Programmierschnittstelle wurde ebenfalls anhand des vorgeschlagenen Entwurfs mit Hilfe von REST umgesetzt. Als Autorisierungsverfahren wird die leichtgewichtige Technologie HTTP Basic Auth eingesetzt und mit SSL ergänzt. Als Datenübertragungsformat wird JSON eingesetzt. Die Validierung erfolgt dementsprechend mit Hilfe von JSON-Schema.

Insgesamt wurden 102 Farbprofile von 52 displaybasierten Endgeräten generiert. Für das Erstellen von Farbprofilen wurde die Kalibrierungs- und Profilierungssoftware DisplayCAL[2] genutzt. Auf Basis dieser Gerätefarbräume wurden zusätzliche, generische Farbprofile erzeugt, um allgemeine Farbraumklassen abzubilden. Die Liste der untersuchten Geräte ist in Anhang B einzusehen.

Weiterhin wurden Farbmanagementmodule auf Basis der Frameworks *Argyll CMS*, *ColorSync, Little CMS*, *Qcms* und *Sample ICC* integriert (vgl. [Gil16], [App05], [Sag16], [Mui16], [ICC16]). Hierzu wurden die zugrunde liegenden C/C++ Quellcodes mit Hilfe von JNI[3] in Java-kompatible Bibliotheken übersetzt.

6.2. Peripheriesysteme

Für die Identifikation von aufrufenden Endgeräten wird das Device Detection Repository *WURFL Cloud*[4] genutzt. Für die Ermittlung von Gerätemodell und zugehörigem Betriebssystem dient der User-Agent-Parameter im HTTP Request Header.

Das Speichern von persistenten Systemressourcen erfolgt mit Hilfe von Apache CouchDB[5]. Hierbei handelt es sich um ein dokumentenorientiertes Datenbankmanagementsystem mit REST-Schnittstelle, das zu persistierende Datensätze als JSON-Dokumente speichert. Binäre Daten wie z.B. Farbprofile können als Anhang beigefügt werden.

Die Bereitstellung von farbtransformierten Bilddaten auf Basis von Hyperlinks erfolgt mit Hilfe eines Apache HTTP Servers[6]. Sowohl die Bereitsstellungsdauer der Hyperlinks als auch das Caching der Bilddaten auf dem Webserver sind konfigurierbar.

---

1 Apache Tomcat: https://tomcat.apache.org/

2 DisplayCAL: https://displaycal.net/

3 Java Native Interface: https://docs.oracle.com/javase/8/docs/technotes/guides/jni/

4 WURFL Cloud: https://www.scientiamobile.com/page/wurfl-cloud

5 Apache CouchDB: https://couchdb.apache.org/

6 Apache HTTP Server: https://httpd.apache.org/

6.3. Clientapplikationen

Im Rahmen der prototypischen Implementierung wurden Applikationen für die mobilen Plattformen iOS und Android sowie für das Web realisiert. Diese dienen zum Testen des Farbmanagementsystems sowie zur Demonstration des Funktionsumfangs.

Der Funktionsumfang der mobilen Applikationen umfasst unter anderem die Farbkalibrierung von Endgeräten sowie die Farbtransformation von Pixelwerten und Bilddaten. Entsprechend der vorgeschlagenen REST-API, lassen sich zu transformierende Bilddaten als binärer Datenstrom übertragen sowie mittels Hyperlink referenzieren.

Weiterhin werden persistente Ressourcen wie z.B. Farbprofile und Endgeräte tabellarisch dargestellt. Zudem lassen sich einzelne Ressourcenbestandteile wie z.B. Header oder Tags eines Farbprofils im Detail ausgeben.

Darüber hinaus dient ein umfangreiches Menü zum Konfigurieren von Farbtransformationseinstellungen, dem Testen von unterschiedlichen Parametersätzen und den daraus resultierenden, möglichen Fallbackstrategien.

Der strukturelle Aufbau beider mobiler Applikationen ist identisch. Die nachfolgende Abbildung stellt exemplarische Auszüge der iOS-Applikation dar.

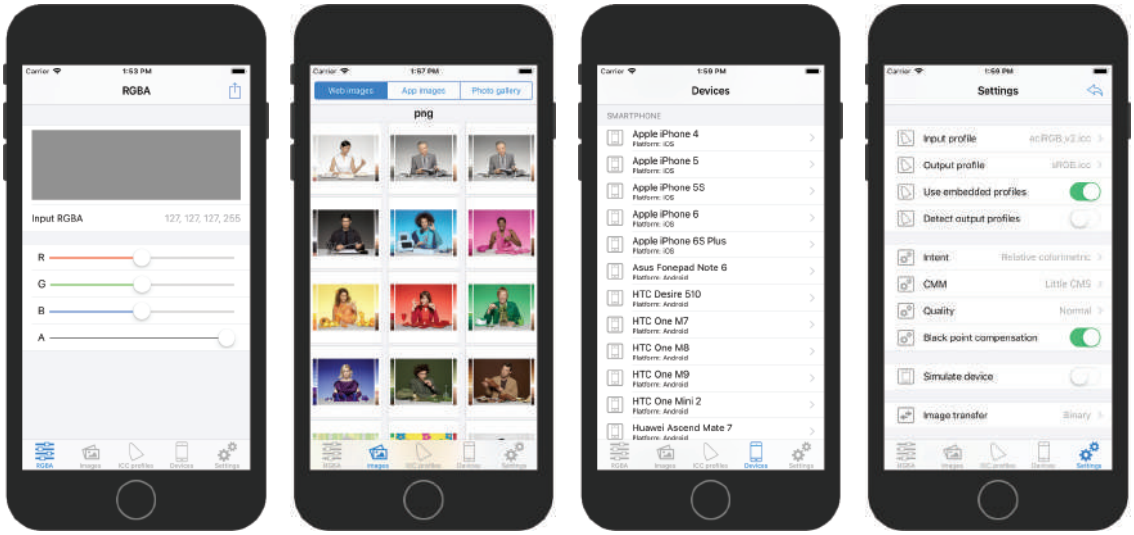


Abbildung 6-2: Exemplarische Ansichten der implementierten iOS-Applikation

Die umgesetzte Webapplikation stellt neben dem angesprochenen Funktionsumfang eine umfangreiche Verwaltungsoberfläche zur Verfügung. Auf diese Weise lassen sich Ressourcen wie z.B. Farbprofile zum Farbmanagementsystem hinzufügen, aktualisieren oder löschen. Die Startansicht der Webapplikation wird in Abbildung 6-3 dargestellt.

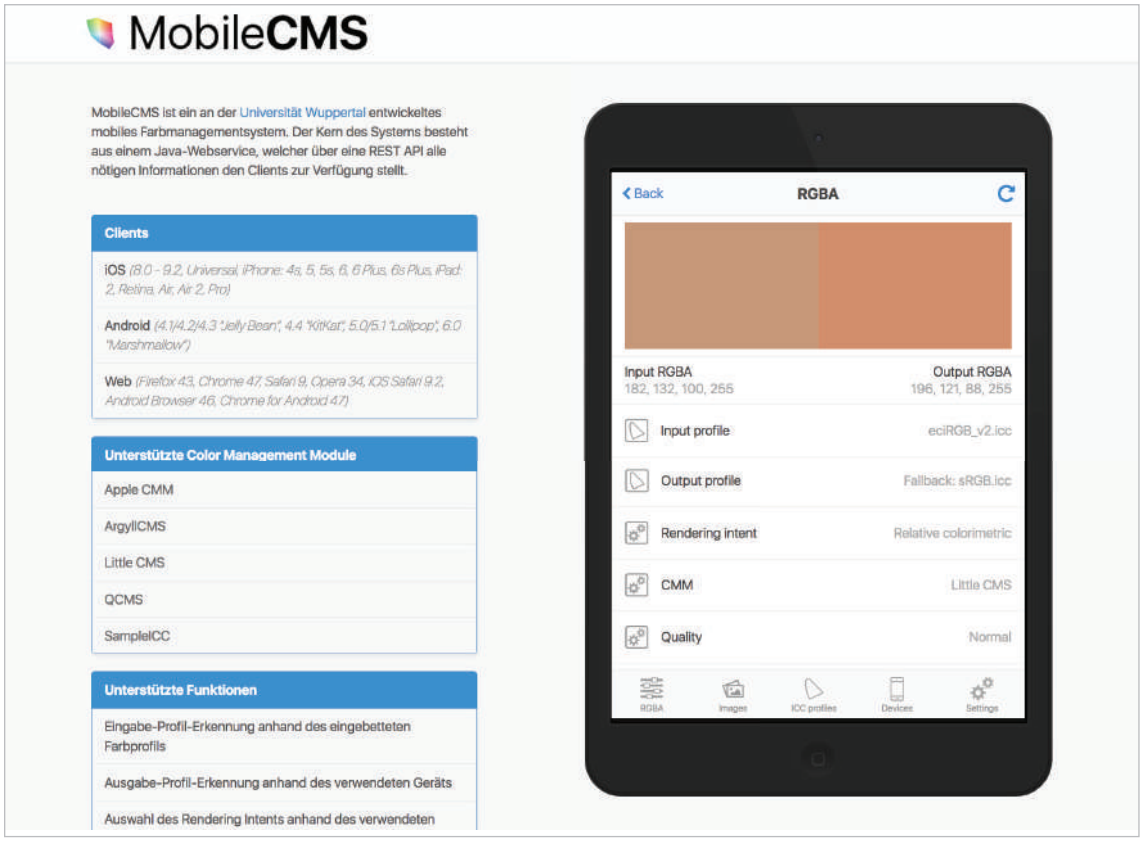


Abbildung 6-3: Startansicht der implementierten Webapplikation

7 Untersuchung der Farbwiedergabe

In diesem Kapitel wird untersucht, in welchem Maß sich die Farbtreue auf aktuellen, displaybasierten Endgeräten durch die Anwendung des vorgestellten Systementwurfs optimieren lässt. Für die Untersuchung wurde die zuvor beschriebene, prototypische Implementierung genutzt.

Die Untersuchung besteht aus einem visuellen und einem messtechnischen Teil. Die Farbdarstellung zu untersuchender Endgeräte wird in beiden Teilen mit Hilfe des umgesetzten Farbmanagementsystems an eine Referenzbedingung angeglichen. Abschließend wird die Güte dieser Angleichung visuell und messtechnisch bewertet.

Für diese Untersuchung werden Testgeräte benötigt, die ein möglichst breites Wiedergabespektrum abdecken. Bei der Erstellung generischer Farbprofile auf Basis von 102 Gerätefarbräumen von insgesamt 52 Endgeräten hat sich herausgestellt, dass sich aktuelle, displaybasierte Endgeräte in etwa vier Farbraumklassen einordnen lassen. Dazu gehört neben den Standardfarbräumen sRGB, Adobe RGB (1998) und DCI-P3 eine LCD-spezifische Farbraumklasse, mit gesättigtem Rot- und Blaubereich (vgl. [Rit15]).

Im Rahmen dieser Untersuchung wird für jede dieser Farbraumklassen ein repräsentatives Endgerät ausgewählt. Zudem wird darauf geachtet, dass sowohl Geräte mit IPS-LCD- als auch mit OLED-Technologie zum Einsatz kommen.

Neben den zu begutachtenden Endgeräten, wird für die Untersuchung ein Referenzdisplay benötigt, das über ein konventionelles, betriebssystemseitiges Farbmanagementsystem angesteuert wird und in der Lage ist, einen möglichst großen Teil der angesprochenen Farbraumklassen abzudecken.

Die an der Untersuchung beteiligten Endgeräte sowie deren Wiedergabeeigenschaften werden in Tabelle 7-1 dargestellt.

Endgerät	Displaytechnologie	Displaygröße	Luminanz	Modus	Farbraum
Endgerät 1	IPS-LCD	10,1 Zoll	ca. 450 cd/m²	–	Proprietär
Endgerät 2	IPS-LCD	7,9 Zoll	ca. 450 cd/m²	–	ca. sRGB
Endgerät 3a	OLED	4,5 Zoll	ca. 330 cd/m²	DCI-P3	ca. DCI-P3
Endgerät 3b				Adobe RGB	ca. Adobe RGB (1998)
Referenzdisplay	IPS-LCD	27 Zoll	ca. 300 cd/m²	Adobe RGB	ca. Adobe RGB (1998)

Tabelle 7-1: Testgeräte der visuellen und messtechnischen Untersuchung

Das OLED-basierte Endgerät aus Tabelle 7-1 wird in zwei verschiedenen Farbdarstellungsmodi angesteuert. Die Bezeichnung wird daher nachfolgend in Endgerät 3a und 3b unterteilt.

Das Referenzdisplay repräsentiert einen IPS-LCD-Monitor, der ebenso über die Farbdarstellungsmodi sRGB und Adobe RGB verfügt. Um einen möglichst breites Spektrum der Gerätefarbräume abzudecken, wird das Referenzdisplay im Modus Adobe RGB angesteuert (vgl. Abb. 7-1).

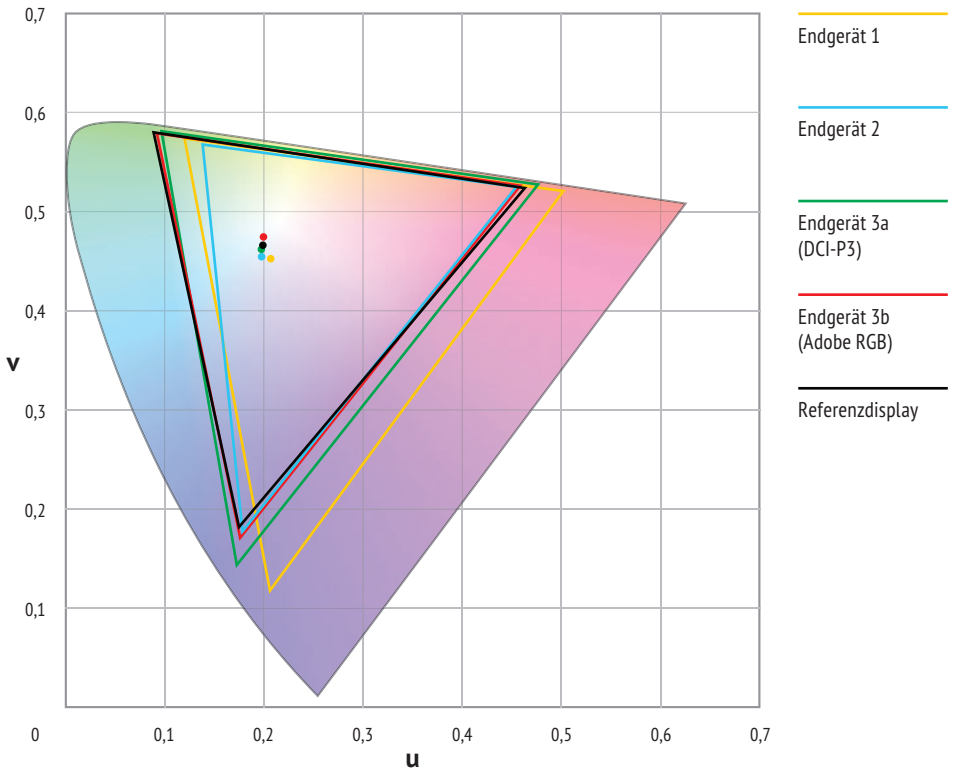


Abbildung 7-1: Farbumfang und Weißpunkt der Testgeräte

Bei der Betrachtung der Gerätefarbräume in Abbildung 7-1 fällt auf, dass die Endgeräte 1 und 2 größere farbliche Unterschiede zur Referenz aufweisen. Beide Geräte besitzen eine geringere Sättigung im grünen Farbbereich. Endgerät 1 ist zudem im roten und blauen Farbbereich stärker gesättigt und besitzt einen deutlich rötlicheren Weißpunkt.

Die Differenz zwischen den Endgeräten 3a und 3b und dem Referenzdisplay ist deutlich geringer. Dies trifft aufgrund des gleichen Darstellungsmodus besonders für Endgerät 3b zu. Es fällt jedoch auf, dass sich Farbumfang und Weißpunkt dennoch in geringem Maß von der Referenzbedingung unterscheiden.

Neben den angesprochenen Farbraumdifferenzen, besitzen die beteiligten Geräte zudem unterschiedliche maximale Leuchtdichten. Um eine Vergleichbarkeit herzustellen, werden alle Geräte auf den kleinsten gemeinsamen Luminanzwert von 300 cd/m² kalibriert. Das Kalibrierungsschema der Testgeräte wird in Abbildung 7-2 dargestellt.

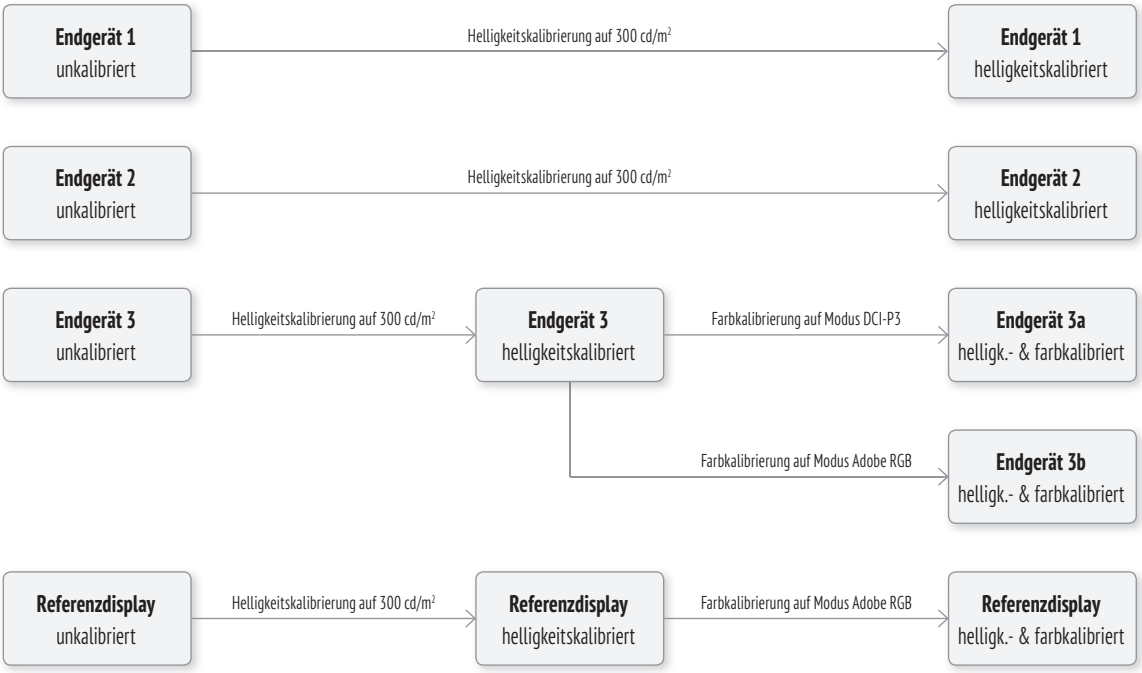


Abbildung 7-2: Kalibrierungssschema der visuellen und messtechnischen Untersuchung

Um die Farbwiedergabeeigenschaften der kalibrierten Endgeräte an die Referenz anzugleichen, folgt im nächsten Schritt eine Farbtransformation der visuell zu begutachtenden Motive sowie der messtechnisch auszuwertenden Testelemente.

Bei den Endgeräten erfolgt die Farbtransformation in zwei Schritten. Im ersten Schritt werden die Testelemente relativ farbmetrisch in den Referenzfarbraum übertragen. Um den Informationsverlust bei der Farbtransformation so gering wie möglich zu halten, werden alle Testelemente bereits im Farbraum Adobe RGB (1998) aufbereitet.

Im zweiten Schritt erfolgt eine absolut farbmetrische Transformation in den jeweiligen Ausgabefarbraum. Auf diese Weise wird der Farbraum der Referenzbedingung und insbesondere dessen Weißpunkt auf dem jeweiligen Endgerät simuliert. Dementsprechend entfällt der zweite Transformationsschritt beim Referenzdisplay (vgl. Abb. 7-3).

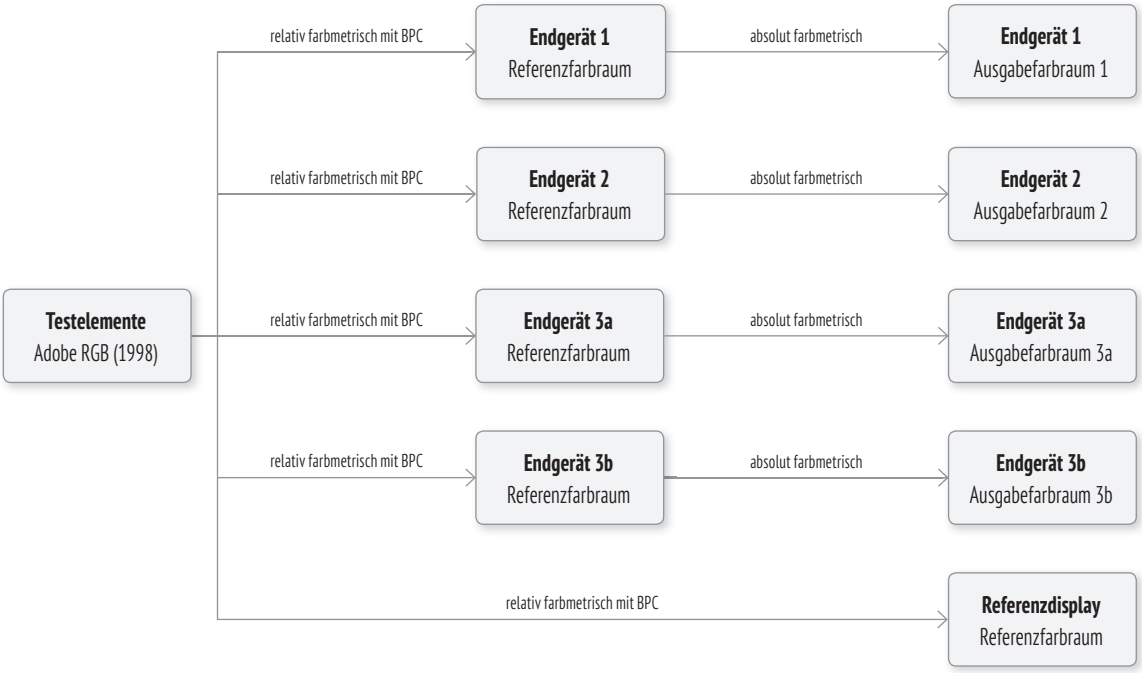


Abbildung 7-3: Farbtransformationsschema der visuellen und messtechnischen Untersuchung[1]

7.1. Visuelle Untersuchung

Die visuelle Untersuchung basiert auf der Begutachtung ausgewählter Testmotive durch zwölf Teilnehmer. Aufgrund ihres fachlichen Hintergrunds, sind alle Probanden hierfür ausreichend qualifiziert und erfahren.

Die Auswahl der Motive orientiert sich an Gegenständen, die häufig online erworben werden und bei denen eine hohe Farbtreue von großer Bedeutung ist. Hierzu gehören Artikel aus den Bereichen Bekleidung und Textilien, Möbel- und Innenaustattung sowie Kosmetik (vgl. [Sta15]). Darüber hinaus bezweckt die Motivauswahl die Darstellung eines breiten Farbspektrums sowie von wahrnehmungssensiblen Farbbereichen wie Haut- und Neutraltönen (vgl. Abb. 7-4).

Die Testmotive werden auf den Testgeräten sowohl mit, als auch ohne Farbtransformation dargestellt. Hierdurch lässt sich eine Relation zwischen dem untransformierten Urzustand und der farbtransformierten Optimierung erhalten. Das Referenzdisplay stellt dabei grundsätzlich den Soll-Zustand dar. Außerdem sind alle Geräte in beiden Fällen helligkeits- und farbkalibriert.

1 Die Erstellung der Farbprofile als *Custom Device Display Profiles* erfolgte mit Hilfe der Profilierungssoftware *DisplayCAL 3.1.7* (vgl. [Hoe16]) unter Verwendung des Messgeräts *i1 Publish Pro 2* (vgl. [XR16b]). Für die Farbumrechnung wurde das Farbmanagementmodul *Little CMS 2.8* verwendet (vgl. [Sag16]).

Es erfolgen demnach vier Durchgänge je Proband mit jeweils zwölf zu begutachtenden Motiven pro Endgerät. Um einen Lerneffekt zu verhindern, wird die Reihenfolge der dargestellten Motive zwischen den einzelnen Durchgängen variiert. Für die Probanden ist daher nicht ersichtlich, ob das zu begutachtende Motiv farbtransformiert ist oder nicht.

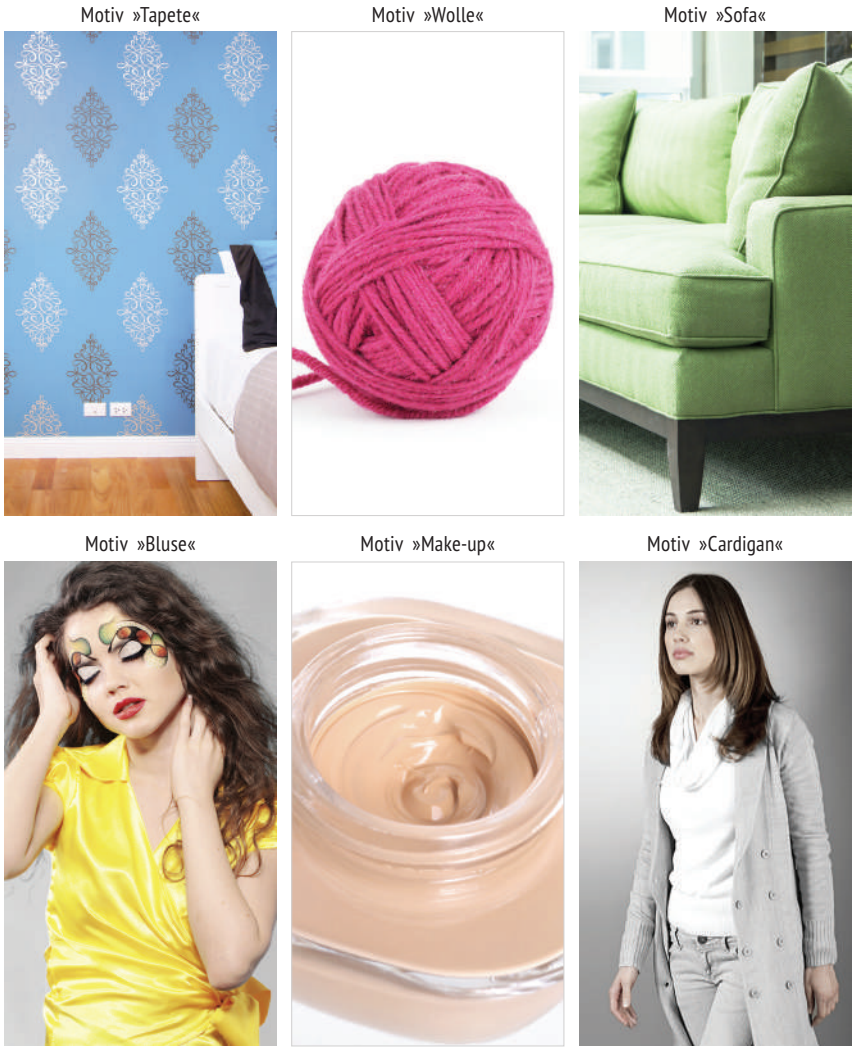


Abbildung 7-4: Motive der visuellen Untersuchung

Auch wenn der Einfluss der Umgebungsbeleuchtung auf Selbstleuchter mit einer Leuchtstärke von 300 cd/m² sehr gering ist (vgl. [Brn14]), wird der Raum für die visuelle Begutachtung abgedunkelt. Hierdurch werden unter anderem Reflektionen der Umgebungsbeleuchtung auf den Displays verhindert.

Um den Sichtabstand zwischen dem zu begutachtenden Testgerät und dem Referenzdisplay möglichst gering zu halten, wird das jeweilige Testgerät seitlich am Referenzdisplay befestigt. Zusätzlich werden alle nicht relevanten Elemente neutralgrau maskiert, um Markenassoziationen sowie ungewollte Farbreize auszuschließen (vgl. Abb. 7-5).

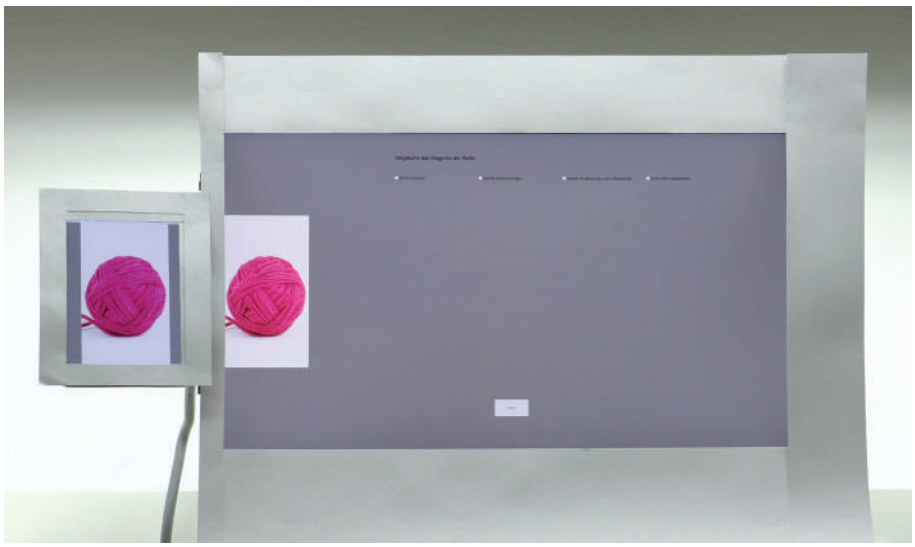


Abbildung 7-5: Versuchsaufbau der visuellen Untersuchung

Um trotz der subjektiven Einschätzung einzelner Probanden eine möglichst vergleichbare Aussage zu erhalten, wird ein einheitliches, vierstufiges Wertungsschema festgelegt. Jeder Proband muss daher bewerten, ob das zu begutachte Motiv im Verhältnis zur Referenz keine wahrnehmbare Farbabweichung aufweist (1), eine geringe Farbabweichung wahrnehmbar ist (2), eine deutliche, jedoch akzeptable Farbabweichung wahrnehmbar ist (3) oder ob die Farbabweichung als inakzeptabel (4) empfunden wird (vgl. Tab. 7-2).

Wertungstufe	Beschreibung
1	Eine farbliche Abweichung ist nicht wahrnehmbar
2	Eine geringe farbliche Abweichung ist wahrnehmbar
3	Eine deutliche, jedoch akzeptable farbliche Abweichung ist wahrnehmbar
4	Eine nicht mehr akzeptable farbliche Abweichung ist wahrnehmbar

Tabelle 7-2: Wertungsschema der visuellen Untersuchung

Um mögliche Farbwahrnehmungsschwächen eines Betrachters vor der visuellen Begutachtung auszuschließen, wurden alle teilnehmenden Probanden einem Farbfehlsichtigkeitstest unterzogen (vgl. [Mun16]). Bei keinem der zwölf Probanden wurde eine Farbfehlsichtigkeit festgestellt.

Die Auswertung der Ergebnisse der visuellen Begutachtung wird nachfolgend dargestellt. Es erfolgt jeweils eine isolierte Ergebnisbetrachtung der untransformierten und transformierten Motive sowie ein Vergleich beider Resultate.

Bei der Bewertungsbetrachtung der untransformierten Motive fällt erwartungsgemäß auf, dass die Motive auf jenen Geräten als farblich ähnlicher eingestuft werden, auf denen die Farbraumdifferenz im Verhältnis zur Referenzbedingung kleiner ist.

Während auf den Endgeräten 1, 2 und 3a die Farbabweichung mit einem Gesamtdurchschnitt > 3 als überwiegend nicht akzeptabel eingestuft wird, liegt der Wert bei Endgerät 3b bei 2,42 und damit überwiegend im akzeptablen Bereich (vgl. Tab. 7-3).

	Motiv »Tapete«				Motiv »Wolle«				Motiv »Sofa«				Motiv »Bluse«				Motiv »Make-up«				Motiv »Cardigan«				Durchschnitt gesamt
Wertungsstufe	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	
Endgerät 1	0	0	0	12	0	0	2	10	0	0	7	5	0	1	10	1	0	0	5	7	0	1	7	4	Ø 3,51
	Ø 4,00				Ø 3,83				Ø 3,42				Ø 3,00				Ø 3,58				Ø 3,25				
Endgerät 2	0	0	1	11	0	0	2	10	0	0	4	8	0	2	7	3	0	0	8	4	0	0	5	7	Ø 3,57
	Ø 3,92				Ø 3,83				Ø 3,67				Ø 3,08				Ø 3,33				Ø 3,58				
Endgerät 3a	0	5	3	4	0	1	7	4	0	0	4	8	0	3	7	2	0	1	1	10	0	6	5	1	Ø 3,18
	Ø 2,92				Ø 3,25				Ø 3,67				Ø 2,92				Ø 3,75				Ø 2,58				
Endgerät 3b	0	4	8	0	2	6	3	1	1	8	3	0	2	8	2	0	0	7	3	2	0	4	6	2	Ø 2,42
	Ø 2,67				Ø 2,25				Ø 2,17				Ø 2,00				Ø 2,58				Ø 2,83				

Tabelle 7-3: Visuelle Bewertung der untransformierten Motive

Die in Tabelle 7-3 dargestellten Reihen der vier Endgeräte enthalten in Kombination mit einem begutachteten Motiv jeweils fünf Zellen. Die oberen vier Zellen stellen die Anzahl der abgegebenen Wertungen pro Wertungsstufe dar, die untere Zelle stellt den arithmetischen Mittelwert aller Wertungen dar.

Im Gegensatz zur Bewertung der untransformierten Motive, wurde die Farbabweichung der transformierten Motive auf allen Endgeräten überwiegend als gering eingestuft. Signifikant ist insbesondere die Verbesserung der Farbdarstellung der Endgeräte 1 und 2 (vgl. Tab. 7-4).

Während beispielsweise der Farbabstand des Motivs »Tapete« im untransformierten Zustand auf den Geräten 1 und 2 insgesamt 23 mal als nicht akzeptabel eingestuft wurde, ist dies im transformierten Zustand nur noch einmal der Fall.

	Motiv »Tapete«				Motiv »Wolle«				Motiv »Sofa«				Motiv »Bluse«				Motiv »Make-up«				Motiv »Cardigan«				Durchschnitt gesamt
Wertungsstufe	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	
Endgerät 1	0	6	5	1	0	6	6	0	0	9	3	0	6	5	1	0	2	7	3	0	1	6	5	0	Ø 2,22
	Ø 2,58				Ø 2,50				Ø 2,25				Ø 1,58				Ø 2,08				Ø 2,33				
Endgerät 2	1	10	1	0	1	9	2	0	1	9	2	0	1	8	3	0	3	6	3	0	0	5	6	1	Ø 2,17
	Ø 2,00				Ø 2,08				Ø 2,08				Ø 2,17				Ø 2,00				Ø 2,67				
Endgerät 3a	2	10	0	0	0	3	7	2	4	6	1	1	8	4	0	0	2	6	4	0	3	7	2	0	Ø 2,02
	Ø 1,83				Ø 2,92				Ø 1,92				Ø 1,33				Ø 2,17				Ø 1,92				
Endgerät 3b	0	11	0	1	1	7	3	1	1	8	2	1	5	6	1	0	5	5	2	0	2	9	1	0	Ø 2,02
	Ø 2,17				Ø 2,33				Ø 2,25				Ø 1,67				Ø 1,75				Ø 1,92				

Tabelle 7-4: Visuelle Bewertung der transformierten Motive

Im Durchschnitt verbessert sich die Farbdarstellung auf den Endgeräten 1, 2 und 3a um 1,28 Wertungsstufen. Auf Endgerät 3b ist eine Verbesserung um 0,4 Stufen festzustellen. Die grafische Tendenz der Ergebnisse der visuellen Untersuchung wird in Abbildung 7-6 dargestellt.

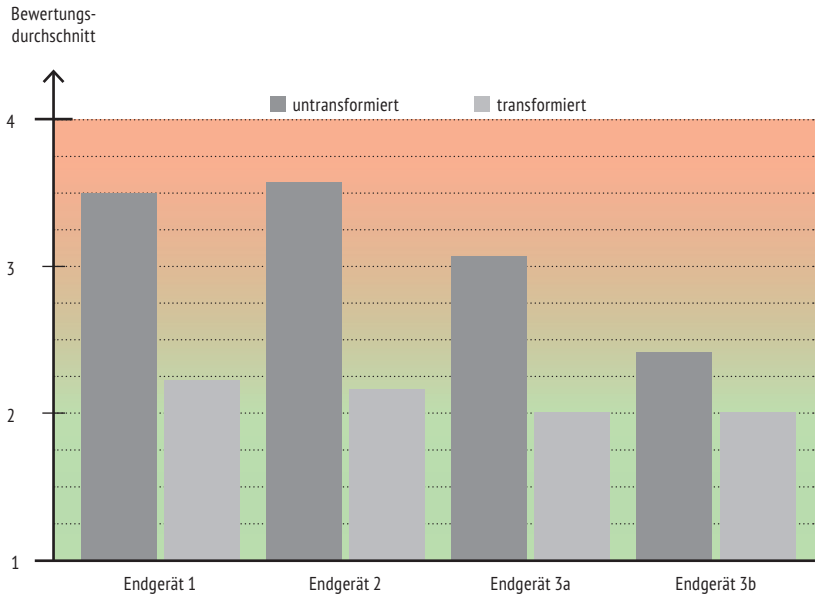


Abbildung 7-6: Ergebnis der visuellen Untersuchung

Das Ergebnis der visuellen Untersuchung lässt die Schlussfolgerung zu, dass sich potenzielle Farbabweichungen auf aktuellen, displaybasierten Endgeräten mit Hilfe des umgesetzten Farbmanagementsystems auf einen als gering einzustufenden Zustand kompensieren lassen.

Die Darstellung aller transformierten und untransformierten Motive auf den untersuchten Endgeräten ist in Anhang C einzusehen.

7.2. Messtechnische Untersuchung

Der zweite Teil der Untersuchung stellt eine messtechnische Bewertung der Farbwiedergabe dar. Diese dient als objektive Kontrolle der auf subjektiven Sinneseindrücken basierenden, visuellen Untersuchung.

Anstelle von Bildmotiven, wird auf den Endgeräten eine Auswahl von Farbfeldern dargestellt. Diese werden spektralfotometrisch vermessen und es werden Farbabstandswerte zum Referenzdisplay gebildet. Die Darstellung erfolgt gemäß des vorgestellten Farbkalibrierungs- und Farbtransformationsschemas (vgl. Abb. 7-2 / 7-3).

Analog zur visuellen Untersuchung findet zu Vergleichszwecken ebenfalls eine Messung der untransformierten Farbfelder statt.

Die Vorlage der Farbfelder bildet das Messtarget »ColorChecker Classic«, welches sich aus Primär-, Sekundär- und Tertiärfarben sowie Neutraltönen zusammensetzt (vgl. [XR16a]). Das Farbspektrum der ausgewählten Farbfelder ist vergleichbar mit dem der Bildmotive aus der visuellen Untersuchung (vgl. Abb. 7-7).

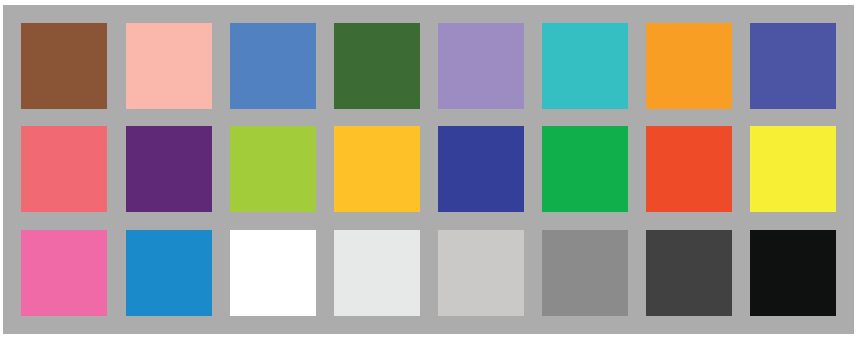


Abbildung 7-7: Farbfelder der messtechnischen Untersuchung

Nachfolgend werden die kummulierten Abstandswerte aller Farbfelder zwischen den untersuchten Endgeräten und dem Referenzdisplay gemäß der Farbabstandsformel ΔE_{00} dargestellt. Dies erfolgt jeweils im untransformierten und transformierten Zustand (vgl. Tab. 7-5 u. Abb. 7-8). Die Messdaten und Farbabstandswerte der einzelnen Farbfelder aller Endgeräte sind in Anhang D ersichtlich.

Endgerät	Durchschnittlicher Farbabstand ΔE_{00} zum Referenzdisplay		Differenz ΔE_{00}
	untransformiert	transformiert	
Endgerät 1	6,36	2,24	4,11
Endgerät 2	4,37	1,85	2,52
Endgerät 3a	3,63	1,55	2,07
Endgerät 3b	2,99	1,60	1,39

Tabelle 7-5: Ergebnis der messtechnischen Untersuchung

Die Ergebnisse der messtechnischen Untersuchung unterstreichen das Resultat der visuellen Untersuchung. Die durschnittlichen Farbabstände zum Referenzdisplay liegen bei den Endgeräten 1 bis 3b im untransformierten Zustand bei 6,36, 4,37, 3,63 bzw. 2,99 ΔE_{00}.

Die durchschnittlichen Farbäbstande zwischen Endgeräten und Referenzdisplay liegen im farbtransformierten Zustand bei 2,24, 1,85, 1,55 sowie 1,60 ΔE_{00}.

Angelehnt an die Wertungsstufen der visuellen Untersuchung, gelten gemessene Farbabstände erst ab einem Wert von 1 ΔE_{00} als wahrnehmbar. Farbabstände zwischen 1 und 2 ΔE_{00} werden als gering und Werte zwischen 2 und 4 ΔE_{00} als deutlich eingestuft. Ab einem Wert von 4 ΔE_{00} werden Farbabstände von Betrachtern üblicherweise nicht mehr toleriert (vgl. [Sch02]).

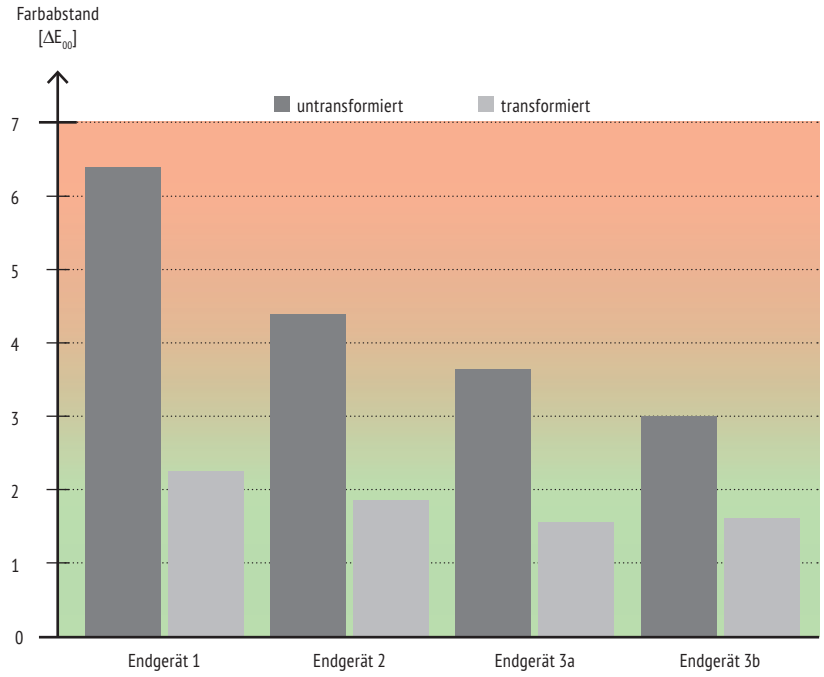


Abbildung 7-8: Ergebnis der messtechnischen Untersuchung

Die messtechnische Untersuchung unterstreicht somit die Aussage der visuellen Untersuchung. Die Farbdarstellung lässt sich auf den getesteten, displaybasierten Endgeräten durch die Anwendung des umgesetzten Farbmanagementsystems verbessern, sodass sich aus deutlichen und zum Teil inakzeptablen Farbabweichungen, eine überwiegend geringe, insgesamt akzeptable Farbdarstellung ergibt.

8 Fazit

Die vorliegende Arbeit begründet die Notwendigkeit eines Farbmanagementsystems, das aufgrund der heterogenen Geräte- und Systemlandschaft von displaybasierten Endgeräten eine serverbasierte, plattformunabhängige Architektur voraussetzt.

Der vorgestellte Entwurf legt den Fokus auf eine technologieneutrale, modulare und skalierbare Architektur des Systems. Komponenten zur Autorisierung und Validierung gewährleisten zudem die Benutzer- und Datenintegrität.

Die Applikationsdomäne des Systems verfügt aufgrund der verwendeten Geräte- und Systemerkennung über zahlreiche Mechanismen zur Prozessautomatisierung. Fallbackstrategien sichern diese Prozesse ab und werden durch das Konzept generischer Endgeräte und Farbprofile ergänzt. Die systemseitig hinterlegten Ressourcen lassen sich zentral verwalten und flexibel ergänzen.

Der Entwurf der vorgestellten Programmierschnittstelle veranschaulicht die Nutzung des Systems unter Verwendung eines verbreiteten Kopplungsmechanismus. Die prototypische Implementierung des Architekturvorschlags stellt darüber hinaus die Realisierbarkeit mit Hilfe aktueller, technischer Werkzeuge sicher.

Die abschließende visuelle und messtechnische Untersuchung des farblichen Anpassungspotenzials stellt zudem dar, dass sich Farbdifferenzen, die mehrheitlich als inakzeptabel eingestuft werden, auf eine akzeptable und insgesamt zu vernachlässigende Darstellung kompensieren lassen.

Der vorgestellte Architekturentwurf ermöglicht daher grundsätzlich eine Verbesserung der Farbkonsistenz, unabhängig von Geräteklasse und Displaytechnologie und erfüllt somit die in der Einleitung formulierte Zielsetzung.

Das realisierte Farbmanagementsystem basiert auf der aktuellen Spezifikation des ICC. Aufgrund der bewusst technologieneutralen und modularen Gestaltung, ist der Ansatz ausreichend flexibel, um ihn hinsichtlich zukünftiger Spezifikationsversionen zu erweitern. Dies könnte insbesondere den Einsatz von Farberscheinungsmodellen zur Berücksichtigung dynamischer Betrachtungs- und Beleuchtungsbedingungen ermöglichen.

Anhang

A – HTTP/1.1–Statuscodes

Code	Nachricht
100	Continue
101	Switching Protocols
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
307	Temporary Redirected
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden

Code	Nachricht
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Time-out
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Request Resource Too Large
414	Request-URI Too Large
415	Unsupported Media Type
416	Requested Range not satisfiable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Time-out
505	HTTP Version not supported

Tabelle A-1: Liste der Statuscodes von HTTP/1.1 (vgl. [Fie99])

B – Farbprofilierte Testgeräte

Nr.	Gerätemodell	Geräteklasse	Plattform	Displaytechnologie	Auflösung
1.	Apple iPad 1	Tablet	iOS	IPS-LCD	1024x768
2.	Apple iPad 2	Tablet	iOS	IPS-LCD	1024x768
3.	Apple iPad 3	Tablet	iOS	IPS-LCD	2048x1536
4.	Apple iPad 4	Tablet	iOS	IPS-LCD	2048x1536
5.	Apple iPad Air	Tablet	iOS	IPS-LCD	2048x1536
6.	Apple iPad Air 2	Tablet	iOS	IPS-LCD	2048x1536
7.	Apple iPad Mini 2	Tablet	iOS	IPS-LCD	2048x1536
8.	Apple iPad Mini 3	Tablet	iOS	IPS-LCD	2048x1536
9.	Apple iPhone 4	Smartphone	iOS	IPS-LCD	960x640
10.	Apple iPhone 5	Smartphone	iOS	IPS-LCD	1136x640
11.	Apple iPhone 5S	Smartphone	iOS	IPS-LCD	1136x640
12.	Apple iPhone 6	Smartphone	iOS	IPS-LCD	1334x750
13.	Asus Fonepad Note 6	Smartphone	Android	IPS-LCD	1920x1080
14.	Kazam Tornado 348	Smartphone	Android	OLED	1280x720
15.	HTC Desire 510	Smartphone	Android	TN-LCD	854x480
16.	HTC One M7	Smartphone	Android	IPS-LCD	1920x1080
17.	HTC One M8	Smartphone	Android	IPS-LCD	1920x1080
18.	HTC One M9	Smartphone	Android	IPS-LCD	1920x1080
19.	HTC One Mini 2	Smartphone	Android	IPS-LCD	1280x720
20.	Huawei Ascend Mate 7	Smartphone	Android	IPS-LCD	1920x1080
21.	Huawei Ascend P7 Mini	Smartphone	Android	TN-LCD	960x540
22.	Huawei Ascend Y530	Smartphone	Android	TN-LCD	854x480
23.	Huawei P8	Smartphone	Android	IPS-LCD	1920x1080
24.	LG G4	Smartphone	Android	IPS-LCD	2560x1440
25.	LG Google Nexus 4	Smartphone	Android	IPS-LCD	1280x720
26.	LG Google Nexus 5	Smartphone	Android	IPS-LCD	1920x1080
27.	Motorola Google Nexus 6	Smartphone	Android	OLED	2560x1440
28.	Motorola Moto G	Smartphone	Android	IPS-LCD	1280x720
29.	Nokia Lumia 830	Smartphone	Windows Phone	IPS-LCD	1280x720
30.	Nokia Lumia 930	Smartphone	Windows Phone	OLED	1920x1080
31.	Samsung Galaxy A3	Smartphone	Android	OLED	960x540
32.	Samsung Galaxy A5	Smartphone	Android	OLED	1280x720
33.	Samsung Galaxy Nexus	Smartphone	Android	OLED	1280x720
34.	Samsung Galaxy Note 3	Smartphone	Android	OLED	1920x1080
35.	Samsung Galaxy Note 4	Smartphone	Android	OLED	2560x1440
36.	Samsung Galaxy Note 8	Smartphone	Android	TN-LCD	1280x800

37.	Samsung Galaxy Note 10.1	Tablet	Android	IPS-LCD	2560x1600
38.	Samsung Galaxy Note Edge	Smartphone	Android	OLED	2560x1600
39.	Samsung Galaxy S4	Smartphone	Android	OLED	1920x1080
40.	Samsung Galaxy S5	Smartphone	Android	OLED	1920x1080
41.	Samsung Galaxy S6 Edge	Smartphone	Android	OLED	2560x1440
42.	Samsung Galaxy Tab 2	Tablet	Android	TN-LCD	1024x600
43.	Samsung Galaxy Tab 3	Tablet	Android	TN-LCD	1024x600
44.	Samsung Galaxy Tab 4	Tablet	Android	TN-LCD	1280x800
45.	Samsung Galaxy Tab Pro	Tablet	Android	IPS-LCD	2560x1600
46.	Sony Xperia Tablet Z	Tablet	Android	IPS-LCD	1920x1080
47.	Sony Xperia Tablet Z2	Tablet	Android	IPS-LCD	1920x1200
48.	Sony Xperia Z	Smartphone	Android	IPS-LCD	1920x1080
49.	Sony Xperia Z Compact	Smartphone	Android	IPS-LCD	1280x720
50.	Sony Xperia Z2	Smartphone	Android	IPS-LCD	1920x1080
51.	Sony Xperia Z3	Smartphone	Android	IPS-LCD	1920x1080
52.	Sony Xperia Z3 Compact	Smartphone	Android	IPS-LCD	1280x720

Tabelle B-1: Liste der farbprofilierten Endgeräte

C – Motive der visuellen Untersuchung

Abbildung C-1: Motive der visuellen Untersuchung (Wolke/Tapete)

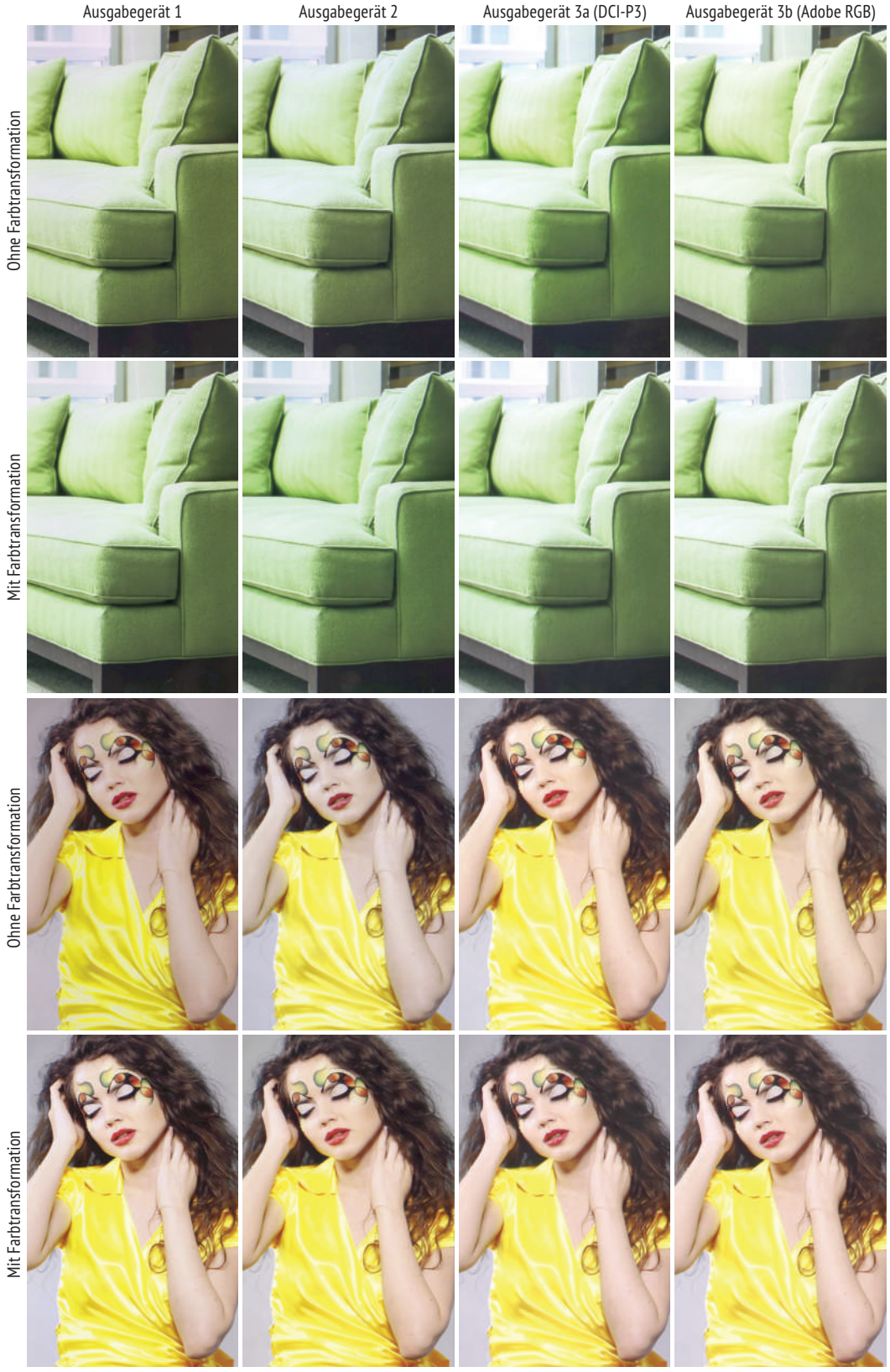


Abbildung C-2: Motive der visuellen Untersuchung (Sofa/Bluse)

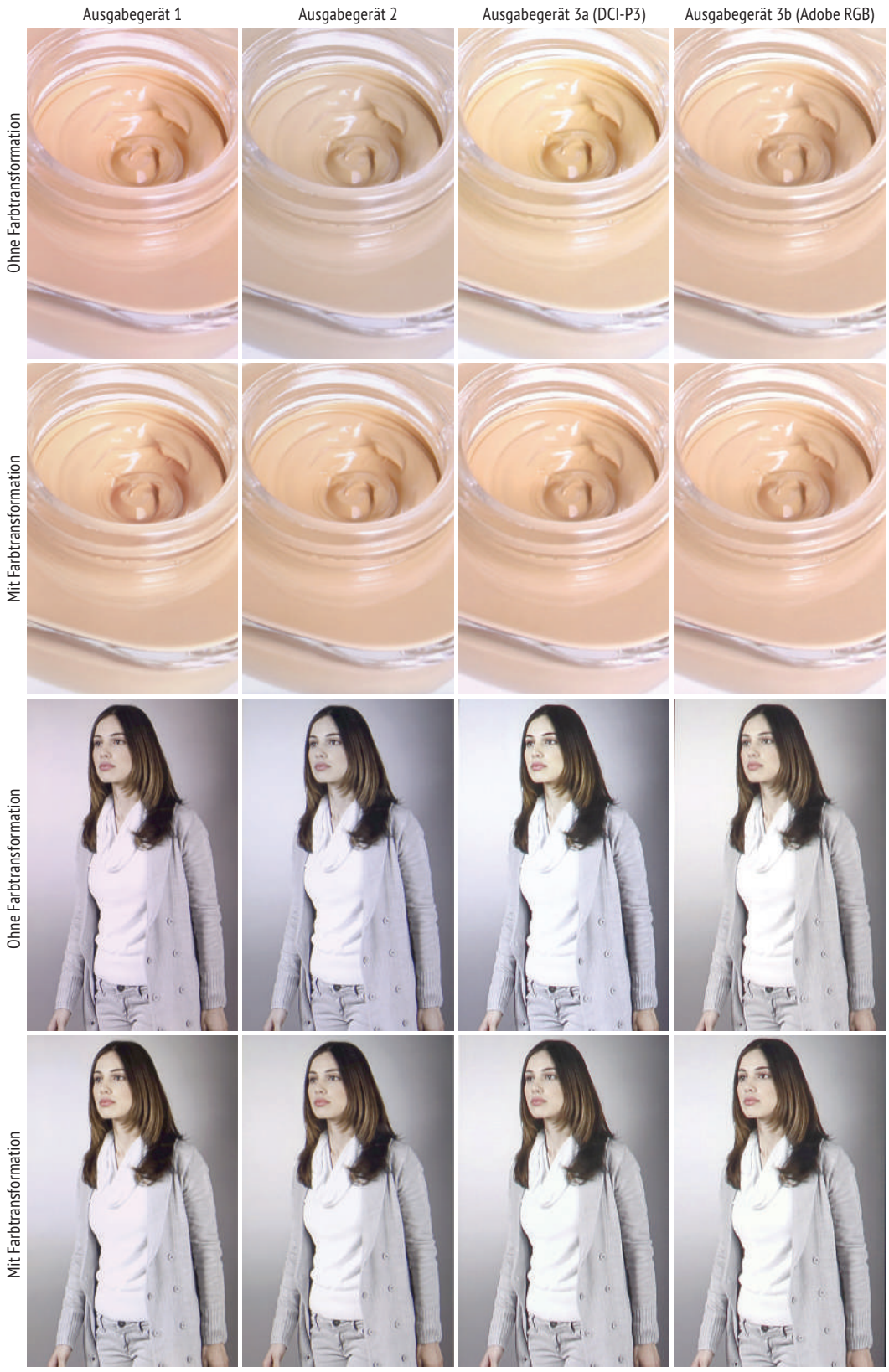


Abbildung C-3: Motive der visuellen Untersuchung (Make-up/Cardigan)

D – Daten der messtechnischen Untersuchung

Feld	Farbe	L*	a*	b*
1		56,16	16,09	6,11
2		91,26	15,95	10,79
3		71,59	1,18	-46,68
4		63,11	-22,74	16,53
5		77,62	20,13	-54,47
6		98,15	-42,90	-14,14
7		83,85	37,46	62,65
8		59,24	21,59	-71,79
9		71,70	56,15	7,43
10		47,91	32,73	-41,87
11		99,74	-39,45	64,66
12		96,40	11,20	75,08
13		46,39	32,34	-79,00
14		77,69	-55,22	32,95
15		60,15	62,48	21,36
16		109,48	-11,53	91,03
17		72,26	62,10	-34,21
18		71,78	-24,50	-48,79
19		135,59	-2,71	-29,85
20		110,62	-2,36	-20,16
21		91,95	-1,85	-17,38
22		73,48	-0,66	-14,90
23		54,48	-1,34	-10,83
24		35,65	1,35	-9,35

Tabelle D-1: Messwerte des Referenzdiplays

		Ohne Farbtransformation					Mit Farbtransformation				
Feld	Farbe	L*	a*	b*	ΔE_{76}	ΔE_{00}	L*	a*	b*	ΔE_{76}	ΔE_{00}
1		54,26	16,18	0,80	9,66	7,45	56,16	16,09	6,11	4,26	3,48
2		87,18	16,27	-2,88	11,40	7,53	91,26	15,95	10,79	5,88	4,44
3		71,19	8,07	-51,38	8,32	4,26	71,59	1,18	-46,68	2,43	1,61
4		63,88	-18,95	14,38	3,24	1,61	63,11	-22,74	16,53	1,24	0,68
5		75,47	24,98	-60,57	8,77	3,95	77,62	20,13	-54,47	3,07	2,18
6		95,38	-26,48	-19,18	16,32	7,69	98,15	-42,90	-14,14	4,69	2,73
7		77,94	27,91	56,80	16,01	7,94	83,85	37,46	62,65	5,15	3,15
8		55,44	31,89	-77,75	14,62	7,93	59,24	21,59	-71,79	3,15	2,47
9		64,10	52,28	-5,97	16,90	10,43	71,70	56,15	7,43	3,41	2,43
10		44,03	36,10	-51,56	8,74	4,76	47,91	32,73	-41,87	2,47	0,82
11		95,64	-34,43	60,49	8,33	4,18	99,74	-39,45	64,66	3,83	1,88
12		91,47	9,20	69,71	13,00	6,67	96,40	11,20	75,08	6,42	3,68
13		42,84	44,82	-87,35	13,38	6,25	46,39	32,34	-79,00	3,91	1,34
14		76,97	-43,00	30,17	11,62	4,07	77,69	-55,22	32,95	2,75	1,32
15		51,61	61,28	8,96	18,19	11,66	60,15	62,48	21,36	3,58	1,82
16		105,32	-11,11	88,66	10,62	5,45	109,48	-11,53	91,03	7,41	3,86
17		63,79	64,28	-51,73	17,24	9,87	72,26	62,10	-34,21	5,24	2,54
18		71,36	-12,34	-50,17	14,27	7,11	71,78	-24,50	-48,79	3,91	2,08
19		128,32	5,09	-35,58	13,57	4,45	135,59	-2,71	-29,85	3,97	2,06
20		104,61	4,26	-28,41	12,90	6,39	110,62	-2,36	-20,16	4,24	2,20
21		87,24	3,85	-24,57	10,30	6,30	91,95	-1,85	-17,38	3,26	1,89
22		71,21	3,61	-21,09	7,91	5,55	73,48	-0,66	-14,90	1,22	0,91
23		52,18	3,50	-17,11	6,53	5,52	54,48	-1,34	-10,83	1,96	1,67
24		34,58	3,34	-12,89	5,86	5,58	35,65	1,35	-9,35	2,12	2,63

Tabelle D-2: Mess- und Farbabstandswerte von Endgerät 1

		Ohne Farbtransformation					Mit Farbtransformation				
Feld	Farbe	L*	a*	b*	ΔE_{76}	ΔE_{00}	L*	a*	b*	ΔE_{76}	ΔE_{00}
1		56,37	9,30	3,65	7,95	5,53	56,36	13,41	8,07	2,08	1,42
2		94,12	8,16	0,13	11,88	8,72	93,47	14,05	8,39	4,93	3,84
3		76,05	2,70	-51,48	4,64	1,95	72,93	-1,16	-45,48	3,31	0,88
4		65,60	-18,61	14,65	3,94	2,39	63,45	-24,05	17,67	2,85	1,29
5		81,88	18,83	-61,27	5,61	2,85	79,61	18,10	-53,36	2,84	0,78
6		104,53	-28,25	-19,56	13,95	6,46	100,86	-44,75	-14,08	4,63	2,56
7		85,47	21,15	59,85	16,78	8,19	88,30	34,43	67,33	3,38	1,94
8		62,71	22,49	-77,40	5,72	1,56	61,03	19,14	-70,96	1,94	0,63
9		70,91	43,26	-3,68	16,36	7,03	74,14	53,42	8,10	2,48	1,13
10		46,94	30,55	-49,93	6,81	4,52	47,18	32,49	-42,35	2,33	0,98
11		104,57	-33,97	62,39	4,02	1,52	102,36	-41,26	66,93	5,07	1,44
12		99,57	2,34	71,64	14,13	7,88	100,38	10,98	78,62	5,52	3,12
13		48,06	35,97	-88,48	6,65	1,83	46,32	31,66	-80,03	3,06	0,65
14		82,90	-42,58	32,27	12,46	4,99	79,47	-57,31	32,40	3,34	0,88
15		55,92	53,91	13,15	16,14	7,31	61,08	62,35	24,17	2,56	1,13
16		113,01	-14,46	88,28	9,60	4,85	112,96	-9,82	93,62	4,70	2,22
17		71,35	53,03	-50,49	15,88	7,57	74,38	60,89	-35,22	4,04	1,22
18		78,42	-15,06	-50,61	11,97	6,08	73,83	-27,99	-47,55	4,07	1,51
19		136,46	-3,34	-31,04	5,43	2,92	134,13	-4,46	-21,61	5,21	2,52
20		115,00	-2,14	-27,36	5,39	2,72	113,40	-3,15	-20,05	2,79	1,78
21		95,82	-1,69	-23,44	4,28	2,24	93,73	-3,38	-16,67	3,48	2,52
22		75,86	-1,52	-18,96	3,74	2,29	74,40	-3,16	-12,51	3,74	3,18
23		55,18	-1,42	-14,56	2,35	1,88	54,40	-3,58	-9,88	4,17	4,37
24		34,63	-0,57	-10,75	2,29	1,61	34,38	-1,90	-6,67	2,42	2,41

Tabelle D-3: Mess- und Farbabstandswerte von Endgerät 2

		Ohne Farbtransformation					Mit Farbtransformation				
Feld	Farbe	L*	a*	b*	ΔE_{76}	ΔE_{00}	L*	a*	b*	ΔE_{76}	ΔE_{00}
1		55,26	9,80	8,83	4,88	3,99	56,73	14,85	6,60	3,31	2,58
2		93,13	10,58	10,04	8,78	7,39	93,80	18,59	4,73	1,40	1,00
3		72,63	-4,32	-44,98	6,05	2,75	72,12	3,18	-51,73	4,58	1,32
4		61,89	-23,69	16,60	2,67	1,77	62,52	-21,42	15,27	1,37	1,01
5		79,91	13,76	-51,97	6,72	2,20	79,33	20,51	-60,85	5,14	1,62
6		99,85	-44,75	-12,17	6,30	3,66	100,76	-44,34	-20,44	4,04	1,51
7		86,08	31,19	62,10	6,77	2,92	87,87	38,30	62,89	2,60	1,43
8		61,61	16,86	-71,33	2,96	1,13	60,46	22,77	-75,59	4,42	1,41
9		73,26	52,06	9,75	4,76	2,32	74,46	59,70	3,20	5,63	2,20
10		47,10	25,48	-39,78	9,45	3,56	46,41	37,58	-46,65	4,79	2,02
11		100,87	-42,03	62,97	5,48	2,42	102,99	-39,73	62,13	3,45	1,54
12		99,11	7,26	73,97	8,88	5,03	100,52	14,59	75,29	1,36	0,67
13		47,98	27,95	-80,49	5,57	2,52	45,08	33,86	-82,48	1,83	1,56
14		77,88	-55,36	34,97	4,47	1,79	78,08	-55,14	27,11	3,94	1,81
15		60,06	62,73	24,21	2,78	1,79	60,42	68,11	20,23	5,01	2,51
16		111,54	-14,73	88,94	9,87	5,05	114,45	-9,03	94,41	4,38	1,81
17		73,11	56,69	-35,23	7,37	2,18	73,96	66,10	-39,57	3,40	1,06
18		74,23	-31,80	-44,87	8,41	3,45	73,61	-27,49	-50,78	1,45	0,75
19		132,14	-7,13	-17,41	10,51	6,01	133,80	-2,27	-27,87	2,26	1,15
20		112,36	-6,56	-15,78	8,34	5,96	113,66	-0,99	-23,85	1,95	0,83
21		93,14	-5,61	-14,09	6,93	5,42	94,08	-2,96	-19,67	1,67	1,69
22		72,00	-4,37	-10,88	6,28	5,33	73,02	-2,62	-15,97	2,27	2,21
23		52,70	-3,60	-8,48	5,56	5,32	53,68	-0,22	-11,82	1,28	1,13
24		33,66	-2,57	-7,35	2,77	3,05	34,32	-0,04	-11,88	3,49	2,46

Tabelle D-4: Mess- und Farbabstandswerte von Endgerät 3a (DCI-P3)

		Ohne Farbtransformation					Mit Farbtransformation				
Feld	Farbe	L*	a*	b*	ΔE_{76}	ΔE_{00}	L*	a*	b*	ΔE_{76}	ΔE_{00}
1		56,99	10,39	8,97	3,95	3,11	57,40	18,13	5,21	6,06	4,88
2		94,95	12,45	9,10	6,76	5,42	96,13	18,58	5,13	2,50	1,55
3		74,33	-3,57	-46,08	4,87	2,46	74,96	0,87	-50,30	2,64	1,25
4		63,61	-23,40	16,60	1,77	0,85	65,25	-21,26	14,80	2,10	1,52
5		81,60	14,77	-53,53	5,23	2,27	81,21	19,94	-59,07	3,39	1,26
6		101,97	-45,30	-12,70	6,08	3,44	102,39	-43,65	-20,63	3,93	1,68
7		88,13	32,73	63,04	4,57	1,84	89,59	39,43	62,50	3,77	2,20
8		63,38	17,41	-72,31	2,99	2,15	61,99	23,25	-77,64	6,21	1,31
9		75,23	53,11	8,82	3,08	1,37	76,49	58,17	5,98	3,36	1,47
10		48,57	27,39	-41,66	6,93	2,87	48,85	35,34	-45,43	2,20	1,05
11		103,05	-42,39	62,57	5,65	2,43	104,17	-39,17	62,50	3,16	1,50
12		101,15	9,28	74,38	6,66	3,74	103,00	12,58	74,53	4,07	2,15
13		49,23	28,88	-82,43	4,77	3,45	47,31	33,62	-86,79	4,09	1,35
14		79,48	-54,76	34,17	3,39	1,30	81,24	-55,06	30,09	2,26	1,47
15		61,48	64,22	24,29	0,97	0,58	62,04	68,74	19,60	5,67	2,54
16		114,21	-14,37	90,16	8,96	4,64	114,93	-7,52	90,97	2,40	1,21
17		74,99	57,65	-36,05	5,92	1,43	76,41	65,13	-39,93	2,92	1,22
18		75,76	-31,53	-46,27	7,36	3,19	75,78	-29,03	-53,53	3,90	1,60
19		135,12	-6,97	-18,18	9,38	5,43	134,63	-2,22	-27,34	1,43	0,80
20		114,95	-6,19	-16,42	7,41	5,34	115,21	-2,30	-23,19	1,69	1,22
21		94,79	-4,47	-15,16	5,27	3,97	96,05	-2,16	-19,77	1,82	1,30
22		73,78	-3,58	-12,11	4,36	3,77	75,82	-1,62	-15,94	1,54	1,27
23		54,45	-2,90	-9,35	4,06	3,87	55,05	0,00	-13,12	0,77	0,65
24		34,28	-2,54	-7,58	2,38	2,80	34,89	0,37	-10,74	2,41	1,94

Tabelle D-5: Mess- und Farbabstandswerte von Endgerät 3b (Adobe RGB)

Literaturverzeichnis

[Ado05] **Adobe Systems, Inc.:** *„Adobe® RGB (1998) – Color Image Encoding“,* Mountain View (CA, USA), 2005

[App05] **Apple, Inc.:** *„ColorSync on Mac OS X“,* Cupertino (CA, USA), 2005

[App16] **Apple, Inc.:** *„iOS 9.3 API Diffs“,* https://developer.apple.com/library/content/releasenotes/General/iOS93APIDiffs/Swift/CoreGraphics.html, Abgerufen am 31.10.2017

[Bel15] **Mike Belshe, Roberto Peon, Martin Thomson:** *„Hypertext Transfer Protocol Version 2 (HTTP/2)“*, IETF, Fremton (CA, USA), RFC 7540, 2015

[Bho08] **Achintya K. Bhowmik, Zili Li, Philip J. Bos:** *„Mobile Displays – Technology and Applications“*, Wiley, Chichester (UK), 1. Auflage, 2008

[Brn14] **Hauke Brüning, Sarah Gremm, Tobias Sobkowiak:** *„Umgebungslichtsensorik in mobilen Geräten“*, Wahlpflichtfach Vertiefung Color Management, Bergische Universität Wuppertal, 2014 (unveröffentlicht)

[Brü99] **Stefan Brües, Liane May, Dietmar Fuchs:** *„Postscriptum on Color Management“*, LOGO / GretagMacbeth, Steinfurt, 1. Auflage, 1999

[Bun07] **Bundesverband Druck + Medien:** *„roman16 bvdm Referenzbilder“*, Wiesbaden, 2007

[Che11] **Robert H. Chen:** *„Liquid Crystal Displays: Fundamental Physics and Technology“*, Wiley, Hoboken (NJ, USA), 1. Auflage, 2011

[Chu15] **Alan Chu, Sing Yeung Lai, On Loy Sung:** *„Color management for web server based applications“*, Patent US9076367B2, Veröffentlicht am 07.07.2015

[Etl13] **Fabian Etling:** *„Realisierung von Colormanagement auf Anwendungsebene unter Android“,* Projektarbeit, Bergische Universität Wuppertal, 2013 (unveröffentlicht)

[Fie00] **Roy T. Fielding:** *„Architectural Styles and the Design of Network-based Software Architectures“*, Dissertation, Irvine (CA, USA), 2000

[Fie99] **Roy T. Fielding, James Gettys, Jeffrey. C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul Leach, Tim Berners-Lee:** *„Hypertext Transfer Protocol – HTTP/1.1“*, IETF, Fremton (CA, USA), RFC 2616, 1999

[Fre96] **Ned Freed, Nathaniel, Borenstein:** *„Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types“*, IETF, Fremton (CA, USA), RFC 2046, 1996

[Gil16] **Graeme Gill:** *„Argyll Color Management System“*, https://www.argyllcms.com/, abgerufen am 17.10.2016

[Gio08] **Edward J. Giorgianni, Thomas E. Madden:** *„Digital Color Management“,* Wiley, Chichester (UK), 2. Auflage, 2008

[Gre10] **Phil Green:** *„Understanding and Using ICC Profiles“*, Wiley, Chichester (UK), 1. Auflage, 2010

[Hil13] **William J. Hilliard:** *„Method and system for improved internet color “*, Patentnr. US8345060B2, veröffentlicht am 01.01.2013

[Hoe16] **Florian Höch:** *„DisplayCAL – Open Source Display Calibration and Characterization“*, https://displaycal.net/, Abgerufen am 09.12.2016

[ICC10] **International Color Consortium:** *„Specification ICC.1:2010 – V4.3“*, Reston (VA, USA), 2010

[ICC16] **International Color Consortium:** *„SampleICC“*, http://sampleicc.sourceforge.net/, abgerufen am 10.12.2016

[IEC99] **IEC: „IEC 61966-2-1:** *Multimedia systems and equipment, Colour measurement and management, Part 2-1: Default RGB colour space, sRGB“*, Genf (CH), 1999

[ISO03] **ISO/IEC 15948:2003:** *„Portable Network Graphics Specification“*, W3C, Cambridge (MA, USA), 2. Auflage, 2003

[Jos06] **Simon Joseffson:** *„The Base16, Base32, and Base64 Data Encodings“*, IETF, Fremton (CA, USA), RFC 4648, 2006

[Kes09] **Lalit Keshav, Peter Stanley Fisher:** *„Web enabled color management service system and method“*, Patent US20090299905A1, Veröffentlicht am 03.12.2009

[Mas11] **Mark Massé:** *„REST API Design Rulebook“*, O'Reilly, Sebastopol (CA, USA), 1. Auflage, 2011

[Mui16] **Jeff Muizelaar:** *„Qcms – Quick Color Management System“*, https://github.com/jrmuizel/qcms, Abgerufen am 10.12.2016

[Mun16] **Munsell Color:** *„Farnsworth Munsell 100 Hue Test“*, http://munsell.com/color-products/color-vision-tests, Abgerufen am 22.09.2016

[Nuy14] **Oskar Nuyken, Heidi Samarian, Ilse Wurdack:** *„Organische Leuchtdioden“*, Wiley, Berlin, 1. Auflage, 2014

[Ope17] **Open Handset Alliance:** *„ColorSpace Reference – API Level 26“,* https://developer.android.com/reference/android/graphics/ColorSpace.html, Abgerufen am 14.01.2018

[Per12] **Luiz Pereira:** *„Organic Light Emitting Diodes“*, Pan Standford Publishing, Boca Raton (FL, USA), 1. Auflage, 2012

[Rit15] **Sven Ritzmann:** *„Klassifizierung der Farbeigenschaften mobiler Endgeräte“,* Bachelor Thesis, Bergische Universität Wuppertal, 2015 (unveröffentlicht)

[Rit16] **Sven Ritzmann, Britta Reppel, Janina Hoffmann:** *„Mobile CMS im Praxistest“*, Präsentation, Bergische Universität Wuppertal, 2016 (unveröffentlicht)

[Sag16] **Marti Maria Saguer:** *„Little CMS – Open Source Color Management Engine“*, http://www.littlecms.com, abgerufen am 22.09.2016

[Sch02] **Kurt Schläpfer:** *„Farbmetrik in der grafischen Industrie“*, Ugra, St. Gallen, 3. Auflage, 2002

[Sha03] **Abhay Sharma:** *„Understanding Color Management“*, Cengage Learning, Clifton Park (NY, USA), 1. Auflage, 2003

[SMP11] **SMPTE:** *„RP 431-2:2011 – Digital Cinema Quality – Reference Projector and Environment“*, 2011

[Sta15] **Statista:** *„E-Commerce in Deutschland“*, Statista GmbH, Hamburg, 2015

[Til11] **Stefan Tilkov:** *„REST und HTTP“*, dpunkt, Heidelberg, 2. Auflage, 2011

[XR16a] **X-Rite, Inc.:** *„ColorChecker Classic“*, http://xritephoto.com/colorchecker-classic, abgerufen am 09.12.2016

[XR16b] **X-Rite, Inc.:** *„i1 Publish Pro 2“*, http://www.xrite.com/categories/calibration-profiling/i1publish-pro-2, abgerufen am 09.12.2016

Abbildungsverzeichnis

Tabellenverzeichnis

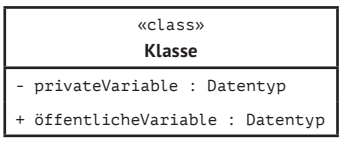
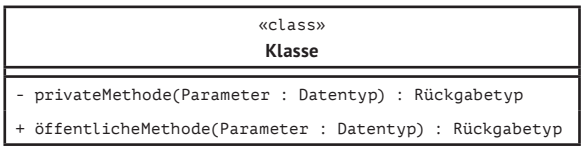
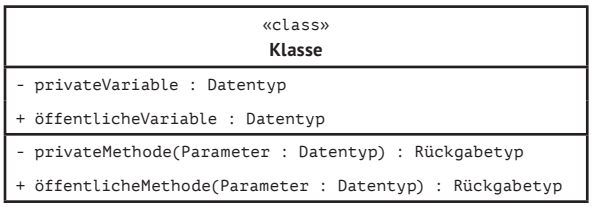
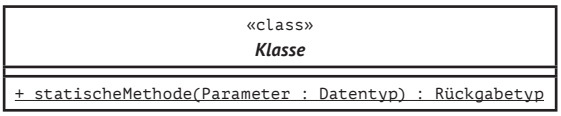
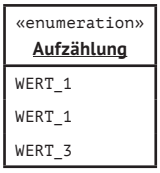
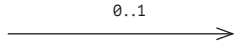
Listingverzeichnis

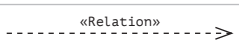
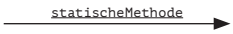
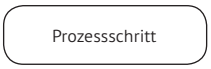
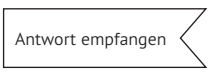
Abkürzungsverzeichnis

API	Application Programming Interface
BPC	Black Point Compensation
CIE	Commision Internationale de l'Éclairage
CMM	Color Management Module
CMS	Color Management System
CRUD	Create Read Update Delete
DBMS	Database Management System
DDR	Device Detection Repository
DIN	Deutsches Institut für Normung
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
ICC	International Color Consortium
IEC	International Electotechnical Commission
IETF	Internet Engineering Task Force
IPS	In Plane Switching
ISO	International Organization for Standardization
J2EE	Java Enterprise Edition
JNI	Java Native Interface
JPEG	Joint Photographic Expert Group
JSON	JavaScript Object Notation
LCD	Liquid Crystal Display
LUT	Look Up Table
MIME	Multipurpose Internet Mail Extension
OLED	Organic Light Emitting Diode
PCS	Profile Connection Space
PNG	Portable Network Graphic
REST	Representational State Transfer
RFC	Request for Comments
RGB	Red Green Blue
RI	Rendering Intent
SMPTE	Society of Motion Picture and Television Engineers
SOAP	Simple Object Access Protocol
SSL	Secure Socket Layer

sRGB	Standard Red Green Blue
TRC	Tone Reproduction Curve
UML	Unified Modelling Language
WURFL	Wireless Universal Resource File
XML	Extensible Markup Language

Notationsverzeichnis

	Klasse
	Klasse mit Angabe von Attributen
	Klasse mit Angabe von Methoden
	Klasse mit Angabe von Attributen und Methoden
	Technologiegebundene Klasse
	Abstrakte Klasse
	Abstrakte Klasse mit statischer Methode
	Schnittstelle
	Aufzählung
	Statische Aufzählung
	Relation
	Vererbung
	Implementierung
	Aggregation mit Angabe von Multiplizität

	Verwendung generischer Datentypen
T -> Datentyp	Bindung von generischem zu konkretem Datentyp
	Klasse
	Abstrakte Klasse
	Objekt
	Prozess
	Lebenszyklus
	Beendeter Lebenszyklus
	Relation
	Aufruf einer statischen Methode
	Aufruf einer öffentlichen Methode
	Aufruf einer privaten Methode
	Referenzierung einer Prozesskette
	Optionale Referenzierung einer Prozesskette
	Prozessbeginn
	Prozessende
	Prozessschritt
	Entscheidung
	Anfrage senden
	Anfrage empfangen
	Antwort senden
	Antwort empfangen
	Kante

Lebenslauf

Der Lebenslauf ist in dieser Version aus Datenschutzgründen nicht enthalten.