HOCHSCHULE KONSTANZ TECHNIK, WIRTSCHAFT UND GESTALTUNG
UNIVERSITY OF APPLIED SCIENCES

Galerie-Synchronisation

Thomas Heidrich

Konstanz, 13.04.2011

BACHELORARBEIT

Bachelorarbeit

zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

an der

Hochschule Konstanz

Technik, Wirtschaft und Gestaltung

Fakultät Informatik
Studiengang Software-Engineering

Thema: **Galerie-Synchronisation**

Bachelorkandidat: Thomas Heidrich
Martin-Schleyer-Str. 39
78465 Konstanz

1. Prüfer: Professor Dr. Oliver Eck
2. Prüfer: Diplom-Informatikerin (FH) Ute Hauth
miradlo Informatikdienstleistungen
Wollmatinger Str. 23
78467 Konstanz

Ausgabedatum: 01.01.2011
Abgabedatum: 13.04.2011

Zusammenfassung (Abstract)

Thema:	Galerie-Synchronisation
Bachelorkandidat:	Thomas Heidrich
Firma:	miradlo
Betreuerin:	Ute Hauth
Abgabedatum:	13.04.2011
Schlagworte:	asynchrone Kommunikation, Peer-To-Peer, Bilderverwaltung, Galerie, Synchronisation

In dieser Bachelorarbeit werden Synchronisationsprozesse entworfen, um ein Bilderverwaltungssystem entwickeln zu können, mit welchem Benutzer Bilder auf ihren eigenen Rechnern verwalten, bearbeiten und mit anderen Instanzen des Systems synchronisieren können.

Es werden adäquate Datenbestandsverwaltungen für den client- und serverseitigen Teil des Systems nach eingehender Überprüfung extrahiert.

Die Anforderungen an den Synchronisationsprozess zwischen den Datenbeständen werden analysiert. Die Wahl der zum Einsatz kommenden Technologien für das client- und serverseitige Teilsystem wird kritisch diskutiert und Möglichkeiten der Umsetzung aufgezeigt.

Das Bilderverwaltungssystem wird vielfältige Arten der Kommunikation mit anderen Systemen bieten. Dazu gehört unter anderem das Hinzufügen von Bildern per E-Mail und die Weitergabe von Bildern an Soziale Netzwerke., wie beispielsweise identi.ca, twitter oder Facebook.

Ehrenwörtliche Erklärung

Hiermit erkläre ich, Thomas Heidrich, geboren am 10.10.1986 in Reichenbach / Vogtland,

(1) dass ich meine Bachelorarbeit mit dem Titel:

"Galerie-Synchronisation"

im Betrieb unter Anleitung von Professor Dr. Oliver Eck und Diplom-Informatikerin (FH) Ute Hauth selbständig und ohne fremde Hilfe angefertigt habe und keine anderen als die angeführten Hilfen benutzt habe;

(2) dass ich die Übernahme wörtlicher Zitate, von Tabellen, Zeichnungen, Bildern und Programmen aus der Literatur oder anderen Quellen (Internet) sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe.

Mir ist bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, 13.04.2011

Inhaltsverzeichnis

Vorwort und Danksagung

Diese Bachelorthesis ist bei der Firma miradlo Informatikdienstleistungen in Konstanz entstanden. Bei miradlo war ich als Mitarbeiter und Bachelorand angestellt. Diese Arbeit bildet den Abschluss meines Studiums im Studiengang Software Engineering an der Hochschule für Technik Wirtschaft und Gestaltung in Konstanz.

Mein erster Dank geht an Herrn Professor Dr. Oliver Eck für die Möglichkeit, diese Arbeit unter seiner Betreuung anfertigen zu können. Während meines gesamten Studiums war Herr Eck als Studiengangsleiter und Dozent stets als Ansprechpartner für mich da. Durch sein Engagement und seine freundliche Art hat er es stets geschafft, meine Moral und die des gesamten Studiengangs hoch zu halten. Seine Unterstützung hat dazu beigetragen, dass ich mein Studium in der vorgegebenen Zeit abschließen konnte. Er stand mir mit hilfreichen Tipps während meiner Arbeit an diesem Dokument zur Seite. Mit der Bewilligung einer Verlängerung der Bearbeitungszeit um 14 Tage hat er es ermöglicht, dass die für mich ungünstigen Umstände im März 2011 keinerlei Einfluss auf die Qualität dieser Arbeit hatten.

Ich danke Herrn Diplom-Informatiker (FH) Roland Baldenhofer (miradlo) für die fachliche Unterstützung bei der Erstellung dieser Arbeit. Seine Geduld und sein Engagement bei der Diskussion von Problemstellungen und Entwicklung von Lösungsansätzen war beispiellos und hat erheblich zur Qualität dieses Dokuments beigetragen.

Ich danke Frau Diplom-Informatikerin (FH) Ute Hauth (miradlo) und Roland Baldenhofer für die enorme moralische Unterstützung. Beide haben mir gezeigt, was eine wissenschaftliche Arbeit ausmacht und welche Arbeitsmethoden für dessen Erstellung am günstigsten sind.

Ich bedanke mich bei Ute Hauth für ihre Engelsgeduld beim Korrekturlesen und die vielen hilfreichen Tipps zu Formulierungen und strukturellem Aufbau in einer Bachelorthesis.

Ein besonderer Dank geht an meine Eltern Brigitte und Stefan sowie meine Schwester Stefanie, ohne die mein Studium überhaupt nicht möglich gewesen wäre.

1 Einleitung

Digitale Bilder (Bilder) werden heutzutage auf unterschiedlichsten Geräten wie beispielsweise Mobiltelefonen, Laptops und Digitalkameras erzeugt.

Bei exzessiver Nutzung entsteht nach kurzer Zeit eine Flut an ungeordneten und kaum noch durchsuchbaren Bildern, siehe auch folgende Abbildung.

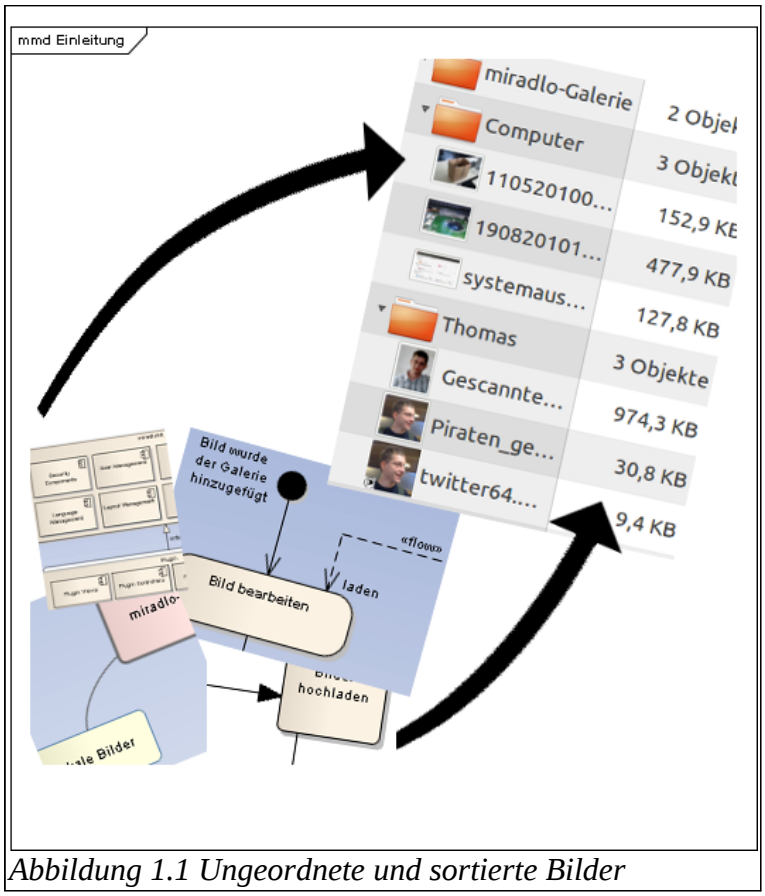

Abbildung 1.1 Ungeordnete und sortierte Bilder

Wenn Bilder zugeschnitten oder anders verändert werden, entstehen neue Bilder die wiederum verwaltet werden müssen. Um die Originale zu erhalten, müssen veränderte Versionen, wie zugeschnittene Bilder, an einer anderen Stelle abgespeichert werden.

Die Verwaltung und der Austausch von Bildern und zugehörigen Zusatzinformationen stellt für viele Benutzer eine große Herausforderung dar, da Wissen im Bereich der Dateiverwaltung und -formate notwendig ist.

Das Ziel ist eine selbst-gehostete Galerie, die sowohl online als auch offline nutzbar ist. Effizient und schnell werden auch große Mengen Bilder zwischen Benutzern ausgetauscht. Aus der Galerie heraus bietet sich die Möglichkeit mit beliebigen, aktuellen Sozialen Netzwerken zu kommunizieren. Ein Bild kann nicht nur Teil der Galerie sein, sondern ebenso Teil eines Blog-artikels, eines Postings in Facebook oder eines Tweets. Diese Galerie erweitert die kurzlebigen Zugriffe von sozialen Netzwerken mit nachhaltiger Verwaltung der Bilder damit diese über

längere Zeit schnell gesucht und gefunden werden können.

Abgerundet wird der Funktionsumfang durch die Möglichkeit, offline Bildergalerien zu bearbeiten und diese ohne Eingriff des Benutzers online zu stellen, falls der lokale Rechner an das Internet angebunden wird. Um das zu erreichen, muss diese Galerie einfach genutzt werden können. Die Verwaltung der einzelnen Bildergalerien muss intuitiv möglich sein. Suchmöglichkeiten unterstützen beim Wiederfinden des in diesem Moment gewünschten Bildes.

1.1 Aufbau dieses Dokuments

Eine Systemarchitektur wird immer für eine bestimmte Anwendergruppe erstellt. Aus diesem Grund gibt es unterschiedliche Sichtweisen auf die Architektur. Zunächst wird der Systemkontext betrachtet. Der Aufbau dieses Dokuments stellt die Geschäftsprozesse in den Vordergrund. Im *Kapitel 2, Geschäftssicht, Seite 7,* werden diese Geschäftsprozesse für das Bilderverwaltungssystem erarbeitet. Im *Kapitel 3, Problemstellung und Analyse, Seite 20,* werden die einzelnen Aspekte, die für die Synchronisation der Instanzen benötigt werden, entsprechend erarbeitet. Abschließend werden im *Kapitel 4, Technische Konzepte, Seite 76,* weitere Aspekte erläutert, die für die Erstellung des Bilderverwaltungssystems zusätzlich betrachtet werden müssen. Da die Anforderungen in der betrachteten Sicht des Systems nicht die Grundlage darstellen, werden diese nur im Anhang mit aufgeführt. Das gesamte Dokument orientiert sich am arc42-Template 4.0 [ARC42] und nutzt auch die dort üblichen Begriffe.

1.2 Systemkontext

Die in dieser Arbeit beschriebene Online-Galerie kann bei Bedarf von diversen anderen Systemen (Umsystemen) verwendet werden. Dies soll eine Weiterverarbeitung und universelle Nutzung der Bilder ermöglichen.

Die folgenden Umsysteme sind zum Zeitpunkt der Erstellung dieser Arbeit die potenziellen Hauptnutzer der Online-Galerie.

- Anwendungssysteme [5], wie Blogsysteme [3] nutzen die Online-Galerie als Datenquelle um Bilder dort geordnet darstellen zu können.
- Weitere Rechner, auf denen die Online-Galerie ebenfalls in Betrieb ist, teilen ihre Inhalte untereinander. Das Ziel dabei ist, auf jeder Online-Galerie die benötigten Bilder vorhalten zu können. Dadurch können spezielle Ansichten und Webauftritte mit den gleichen Bildern realisiert werden.
- Lokale Rechner können verwendet werden, um die lokal vorhandenen Bilder zu ordnen und zu bearbeiten. Diese Bilder sollen ebenfalls mit der Online-Galerie synchronisiert werden können. (siehe nächstes *Kapitel 1.3, Lokale Daten abgleichen, Seite 4*)
- Externe Dienste, wie Twitter [TWIT] können von der Online-Galerie aktiv mit Daten versorgt werden.
- Bilder, die auf Kameras, USB-Sticks und anderen Speichermedien [6] abgelegt sind sollen der Online-Galerie hinzugefügt werden können.
- Mithilfe von E-Mails [7] soll es möglich sein, Bilder und diverse Zusatzinformationen

der Online-Galerie zu senden.

- Auf die Online-Galerie kann jederzeit mit einem Webbrowser [8] zugegriffen werden, um Inhalte darzustellen und zu verändern.

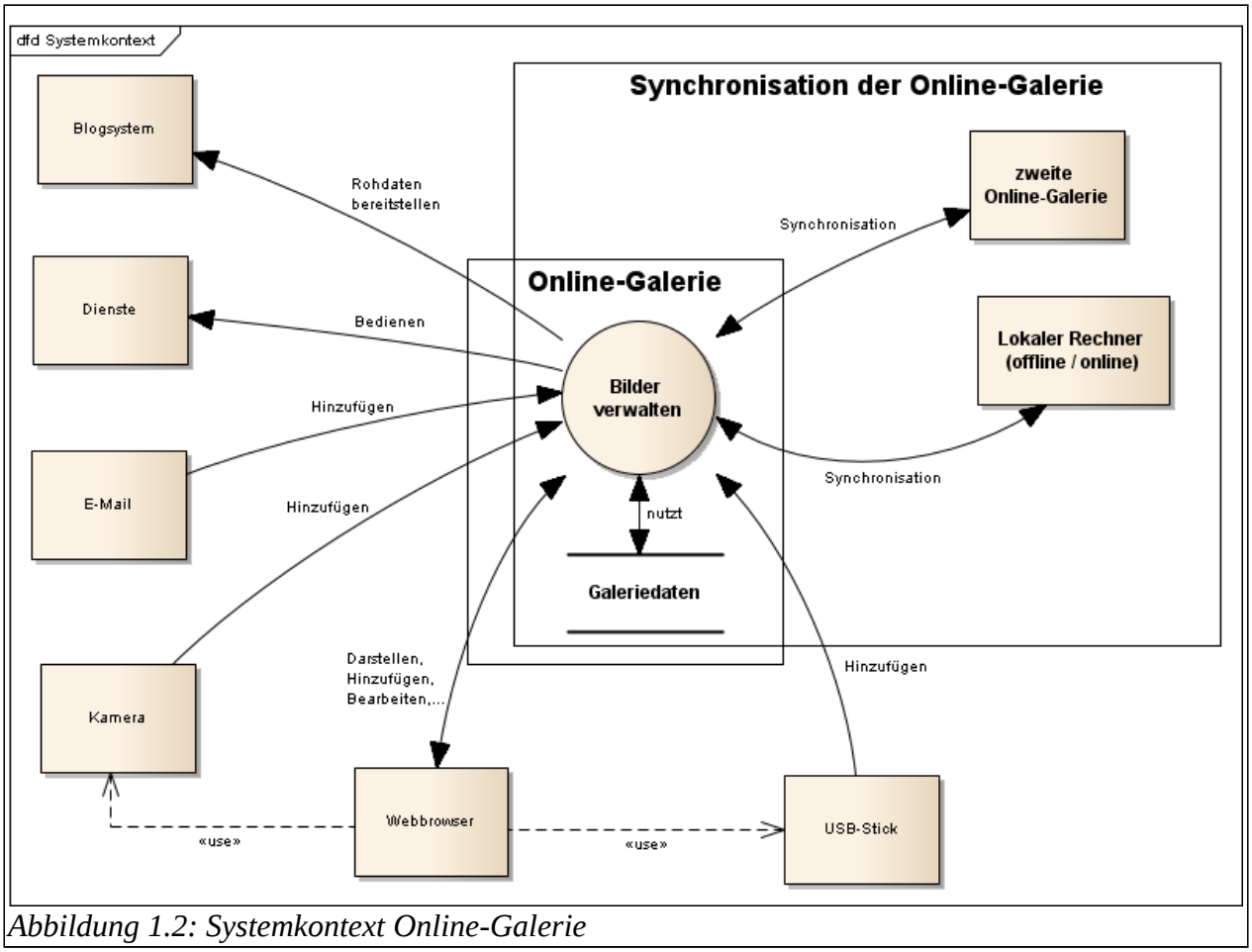

Abbildung 1.2: Systemkontext Online-Galerie

In der Mitte des abgebildeten Datenflussdiagramms [11] ist die zu entwickelnde Online-Galerie samt ihrer Beziehungen abgebildet. Sie besteht aus dem allgemeinen Bilderverwaltungsprozess und einer Datenspeichereinrichtung für die Galeriedaten. Rund um die Online-Galerie sind die beteiligten Umsysteme abgebildet.

Diagrammelement	**Beschreibung**
Synchronisation der Online-Galerie	Der Fokus der Bachelorthesis liegt auf der Synchronisation der einzelnen Server und Clients. Dies beinhaltet in erster Linie die Online-Galerie, die unterschiedlichen Server und die lokalen Rechner.

1.3 Lokale Daten abgleichen

Die Grundanforderung an eine Online-Galerie ist, dass die von einem Redakteur eingestellten Bilder den Besuchern jederzeit angezeigt werden können.

Die Verarbeitung von Bildern stellt gewisse Anforderungen an die Netzwerkverbindung zwischen dem Computer des Redakteurs und der Online-Galerie.

Bei einer herkömmlichen Online-Galerie muss der Redakteur während seiner Arbeit immer mit der Online-Galerie verbunden sein. Es findet ein stetiges Nachladen von Bildern statt, wenn z.B. die Größe eines Bildes verändert werden soll. Dies verursacht hohe Netzauslastung und der Redakteur muss unter Umständen lange auf das System warten.

Wenn keine Netzwerkverbindung besteht, wie beispielsweise beim Arbeiten im Zug, kann eine Bilderbearbeitung nicht stattfinden.

Konventionelle Online-Galerien erlauben es nicht, lokal ohne Netzwerkverbindung die Bilder zu verarbeiten. Wenn viele Bilder bearbeitet werden sollen, ist die Nutzbarkeit des Systems stark eingeschränkt.

Hier soll das in dieser Bachelorthesis entwickelte Bilderverwaltungssystem Abhilfe schaffen. Im folgenden Diagramm wird der Aufbau einer konventionellen Online-Galerie dargestellt.

Im folgenden Datenflussdiagramm wird vereinfacht dargestellt, wie ein Redakteur auf seinem Laptop Bilder von einem USB-Stick in die Online-Galerie überträgt. Nachdem der Redakteur die Bilder eingestellt und bearbeitet hat, werden sie mit dem Webserver [10] der Online-Galerie synchronisiert und können Besuchern präsentiert werden.

Das Laptop des Redakteurs und der Webserver sind vollständig autonome Systeme, welche über eine Netzwerkverbindung kommunizieren. Der Grund dieser Kommunikation ist ausschließlich die Benutzung der Online-Galerie.

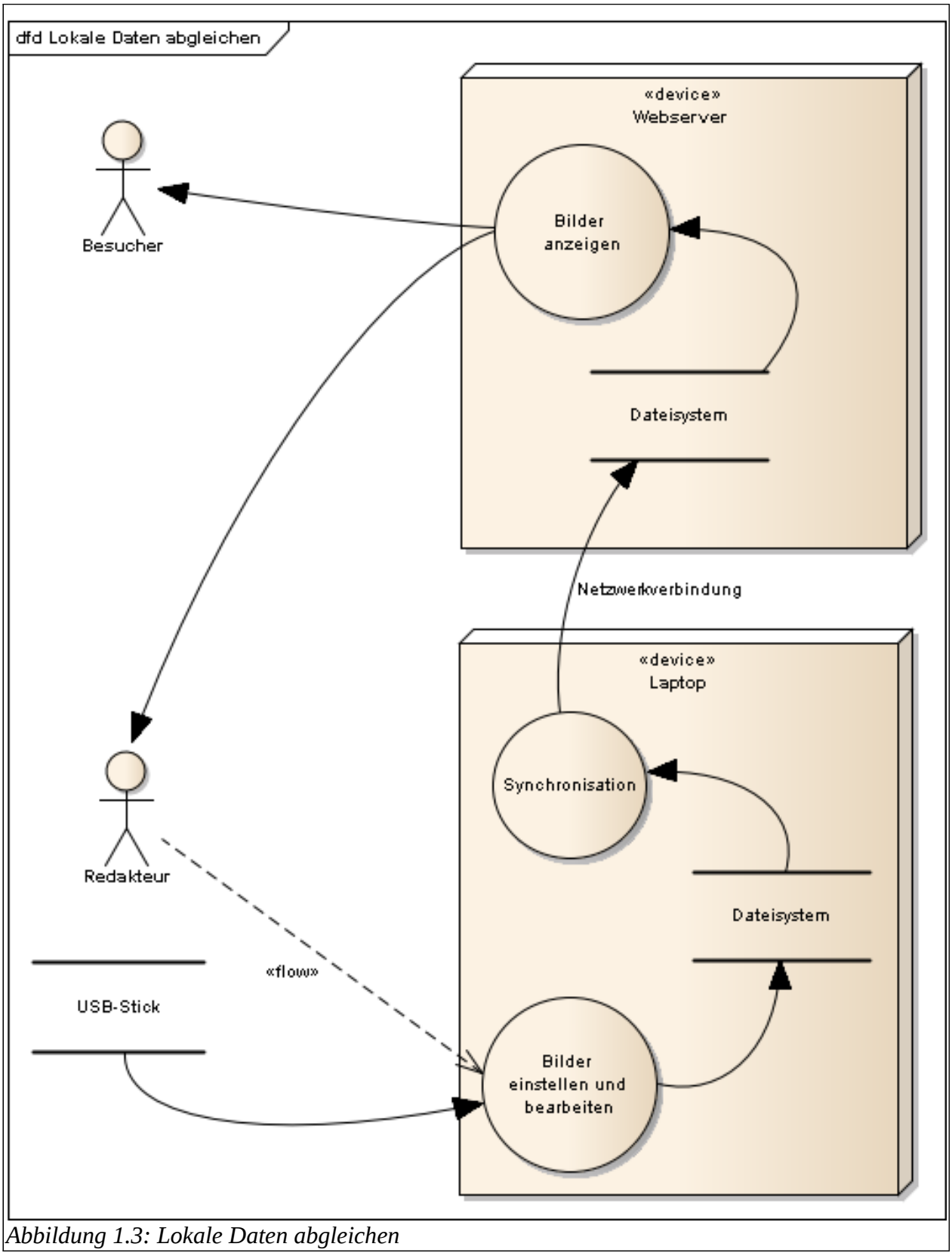

Abbildung 1.3: Lokale Daten abgleichen

1.4 Fokus der Bachelorthesis

Diese Bachelorthesis beschreibt Möglichkeiten der Synchronisation von mehreren miradlo-Galerien. Es werden verschiedene Synchronisationsmechanismen beschrieben und auf die Problemstellungen der persistenten Datenspeicherung eingegangen. "Persistenz ist das Speichern von Objekten auf einem nicht flüchtigen Medium." [12] Weiter werden die Arbeitsweisen im On- und Offlinebetrieb und damit verbundene Abgleiche beleuchtet. Nicht Inhalt dieser Arbeit ist die konkrete Umsetzung des Systems.

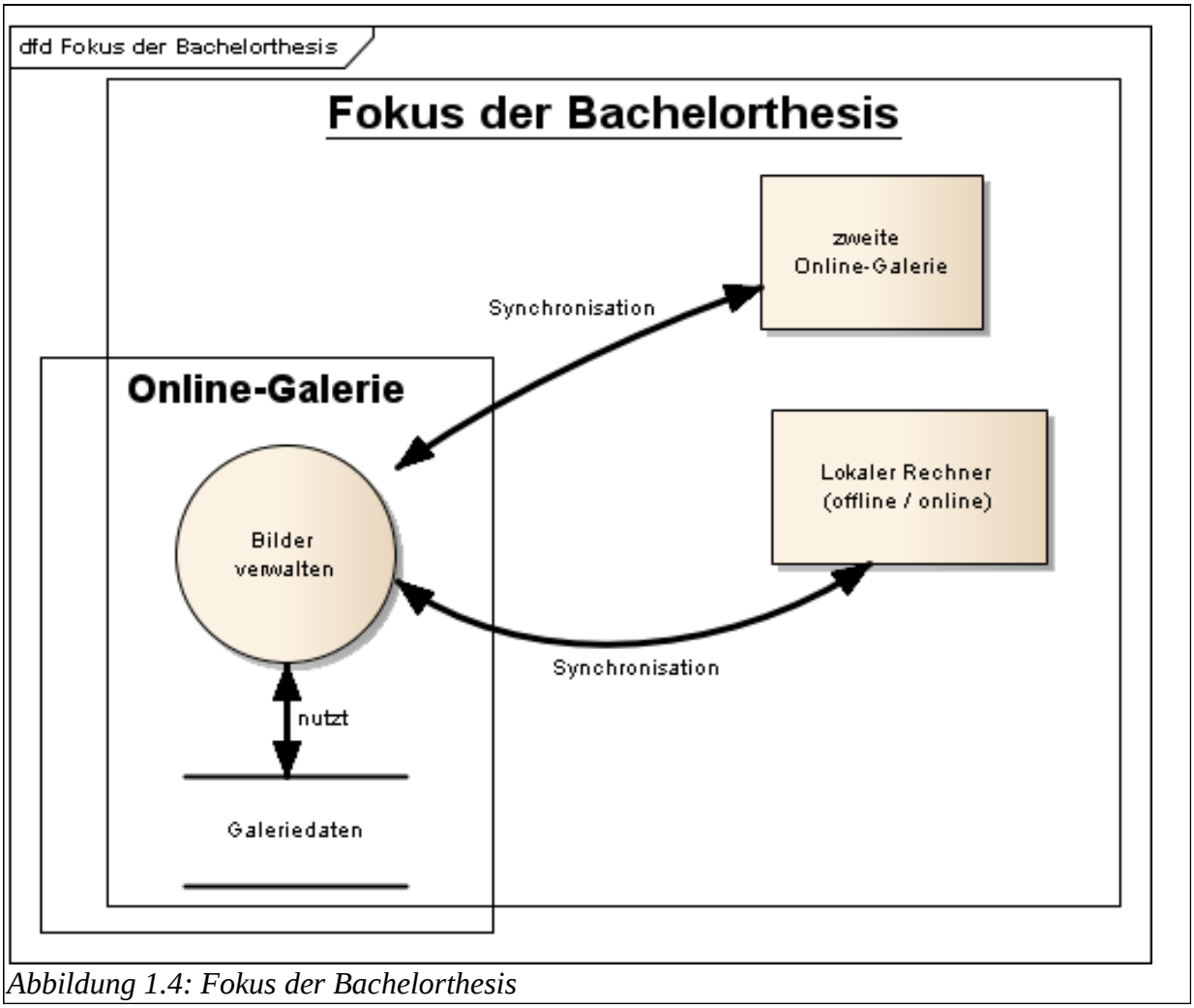

Abbildung 1.4: Fokus der Bachelorthesis

In der Mitte des gezeigten Datenflussdiagramms [11] ist die zu entwickelnde Online-Galerie abgebildet.

Diagrammelement	Beschreibung
Fokus der Bachelorthesis	Der Hauptaugenmerk der Bachelorthesis liegt auf der Synchronisation der einzelnen Server und Clients. Dies beinhaltet in erster Linie die Online-Galerie, die unterschiedlichen Server und die lokalen Rechner.

2 Geschäftssicht

Die Firma miradlo, welche der Auftraggeber dieser Arbeit ist, hat ein detailliertes Pflichtenheft [0] angefertigt. In diesem ist beschrieben, welcher grundlegenden Vision die miradlo-Galerie folgen soll. Außerdem sind Anwendungsfallgruppen aufgeführt, die in der Arbeit zu berücksichtigen sind. In dieser Bachelorthesis sind auf dieser Basis einzelne Anforderungen an die miradlo-Galerie extrahiert worden. Diese finden sich im Anhang wieder und dienen als Grundlage für die in diesem Kapitel beschriebenen Geschäftsprozesse. Das folgende Bild skizziert die Möglichkeiten des Systems aus Nutzersicht.

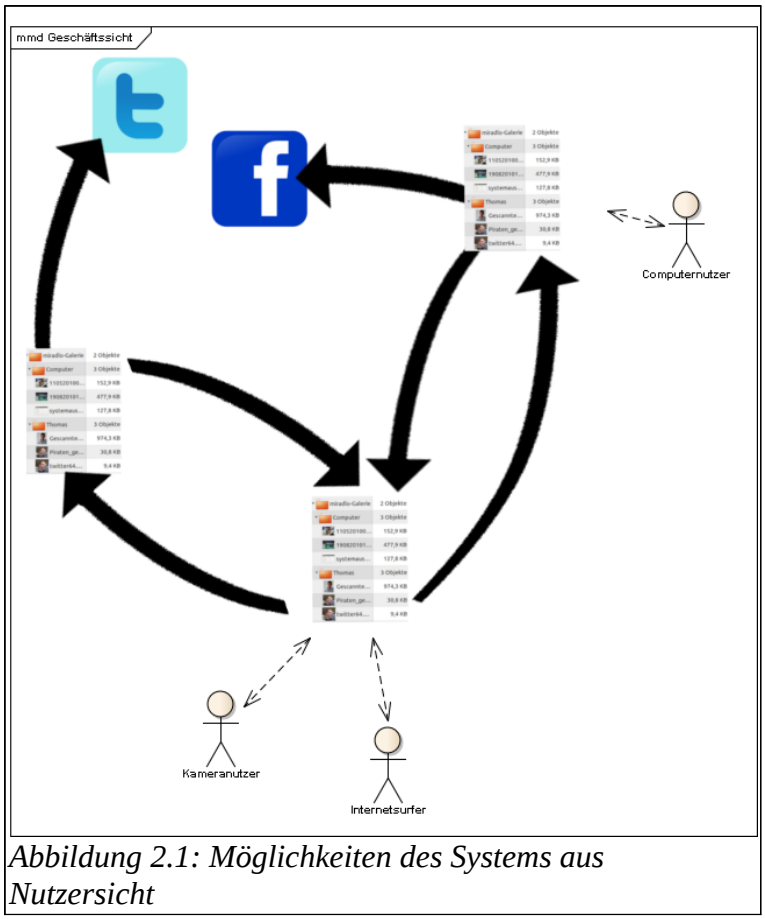

Abbildung 2.1: Möglichkeiten des Systems aus Nutzersicht

"Ein Geschäftsprozess (Abk. 'GP') beschreibt eine Folge von Einzeltätigkeiten, die schrittweise ausgeführt werden, um ein geschäftliches oder betriebliches Ziel zu erreichen. Im Gegensatz zum Projekt kann der Prozess öfter durchlaufen werden. Ein Geschäftsprozess kann Teil eines anderen Geschäftsprozesses sein oder andere Geschäftsprozesse enthalten bzw. diese anstoßen. Geschäftsprozesse gehen oft über Abteilungen und Betriebsgrenzen hinweg und gehören zur Ablauforganisation eines Betriebs. Diese Definition leitet sich aus den Definitionen von Geschäft im engeren Sinn (wirtschaftliche Tätigkeit) und Prozess her." [1]

Die Anforderungen und Geschäftsprozesse wurden während ihrer Erstellung mit der Firma miradlo diskutiert. Erst nachdem sämtliche dieser Punkte geklärt wurden, ist mit der Analyse der vorherrschenden Problemstellungen begonnen worden.

2.1 Bilder in miradlo-Galerie laden

Der Benutzer einer miradlo-Galerie fügt mit diesem Geschäftsprozess Bilder in die miradlo-Galerie ein.

Die Bilder werden ausgewählt, eingestellt und mit Zusatzinformationen (Metadaten) versehen. Metadaten umfassen unter anderem eine Bildbeschreibung, die Dateigröße, den Speicherort, einen Titel und weitere beschreibende Informationen.

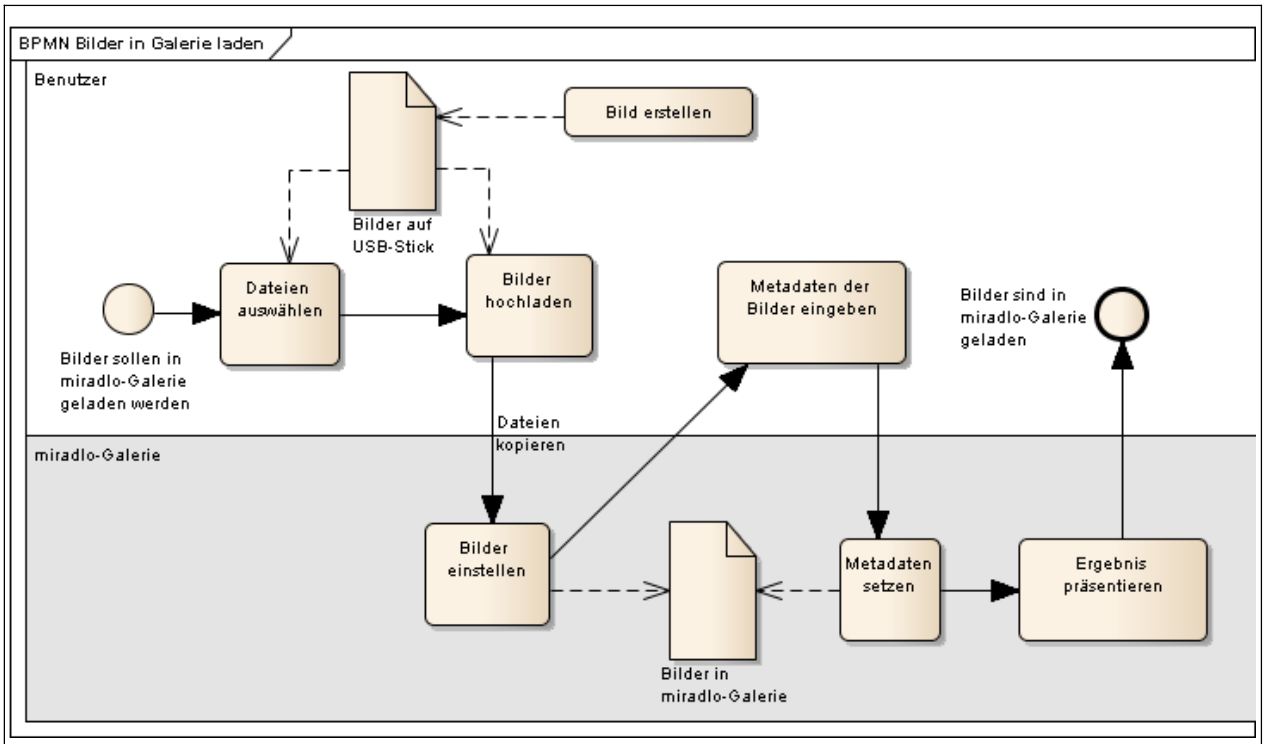

Abbildung 2.2: Bilder in miradlo-Galerie laden

Das vorstehende Geschäftsprozessdiagramm [26] illustriert die einzelnen Aktivitäten von Benutzer und miradlo-Galerie, die nacheinander ablaufen müssen, um Bilder der miradlo-Galerie hinzuzufügen. Es ist zu sehen, dass die Bilder zuerst hochgeladen und dann mit Metadaten versehen werden. Dieser Ablauf ist darin begründet, dass die miradlo-Galerie die Eingabemaske der Metadaten benutzerfreundlicher darstellen kann, wenn sie bereits über die neuen Bilder verfügt.

Diagrammelement	Beschreibung
Dateien auswählen	Der Benutzer wählt die hochzuladenden Bilder im lokalen Dateisystem aus. In diesem Fall liegen die Bilder auf einem USB-Stick. Die Auswahl wird durch das Betriebssystem des Benutzers oder durch zu erstellende Auswahlwerkzeuge unterstützt.
Bild erstellen	Ein Bild wird z.B. mit einer Digitalkamera erstellt. Dieses Bild liegt danach im Speicher der Kamera oder wurde auf dem lokalen Rechner abgelegt. Als weitere Datenträger kommen USB-Sticks und diverse andere handelsübliche Massenspeicher in Frage.
Bilder hochladen	Die ausgewählten Bilder werden der miradlo-Galerie übergeben. Im Normalfall werden bei diesem Vorgang die Bilder über ein Netzwerk in die miradlo-Galerie kopiert. Hierfür kann sowohl der Webbrowser also auch eine spezielle Kommunikationssoftware eingesetzt werden.
Bilder einstellen	Die miradlo-Galerie überträgt die Bilder für die weitere Verarbeitung in ihren persistenten [12] Speicher. Dabei werden die soeben hochgeladenen Bilder speziell als Originale markiert, um eine Veränderung zu verhindern. Diese Bilder werden im Folgenden als Originalbilder bezeichnet.
Metadaten der Bilder eingeben	Der Benutzer fügt jedem Bild Metadaten hinzu. Diese Daten dienen zur Verbesserung der Bilddarstellung für Benutzer.
Metadaten setzen	Die miradlo-Galerie legt die Metadaten jedes Bildes im persistenten Speicher ab.
Ergebnis präsentieren	Es werden die neuen Bilder samt ihrer Metadaten präsentiert. Dabei werden alle eingegebenen Daten und die Originalbilder gemeinsam dargestellt.
Bilder sollen in miradlo-Galerie geladen werden	Der Benutzer besitzt Bilder, die er der miradlo-Galerie jetzt hinzufügen möchte. Die Bilder wurden zuvor beispielsweise mit einer Digitalkamera erstellt.
Bilder sind in miradlo-Galerie geladen	Zu diesem Zeitpunkt sind die Bilder in der miradlo-Galerie gespeichert und können dort angesehen oder weiter bearbeitet werden.
Bilder auf USB-Stick	Der USB-Stick beinhaltet eine Vielzahl unterschiedlicher Bilder, welche nicht alle der miradlo-Galerie hinzugefügt werden sollen.
Bilder in miradlo-Galerie	Dieses Datenobjekt stellt eine Kopie der vom Benutzer ausgewählten und in die miradlo-Galerie hochgeladenen Bilder dar. Diese Originalbilder können nicht mehr verändert werden. Lediglich Metainformationen können bei Bedarf angepasst werden.

2.2 Bilder zwischen miradlo-Galerien synchronisieren

In diesem Geschäftsprozess werden Bilder zwischen zwei miradlo-Galerien synchronisiert. Bei diesem Vorgang findet ein Austausch von Inhalten statt. Das Ziel ist es dabei, am Ende des Prozesses auf beiden Systemen den gleichen Datenbestand vorliegen zu haben.

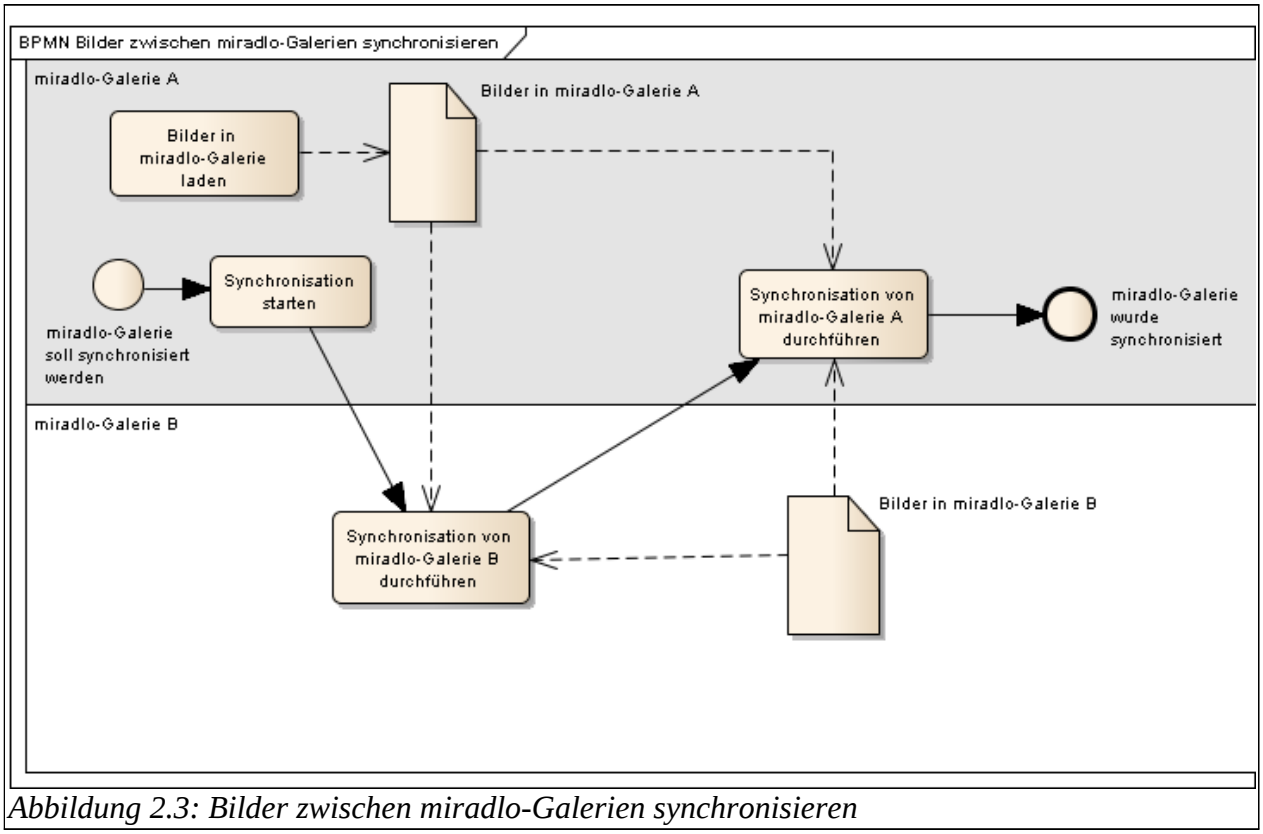

Abbildung 2.3: Bilder zwischen miradlo-Galerien synchronisieren

Das vorstehende Geschäftsprozessdiagramm [26] illustriert den Geschäftsprozess der Synchronisation zweier miradlo-Galerien. Die miradlo-Galerie A ist dabei der Initiator des Prozesses. Die miradlo-Galerie B ist der Synchronisationspartner. Nachdem miradlo-Galerie A und miradlo-Galerie B ihre Datenbestände abgeglichen haben, ist die Synchronisation abgeschlossen.

Diagrammelement	**Beschreibung**
Bilder in miradlo-Galerie laden	Bilder werden z.B. mit einer Digitalkamera erstellt. Danach werden sie der miradlo-Galerie hinzugefügt. (siehe Geschäftsprozess *Bilder in miradlo-Galerie laden*)
Synchronisation starten	Die miradlo-Galerie A sendet der miradlo-Galerie B eine Synchronisationsanfrage, um sicherzustellen, dass diese verfügbar und zu diesem Vorgang bereit ist.
Synchronisation von miradlo-Galerie B durchführen	Unter Berücksichtigung des Datenbestands aus miradlo-Galerie A und dem eigenen Datenbestand, führt miradlo-Galerie B eine Synchronisation durch. Dabei werden die Inhalte aus miradlo-Galerie A mit den eigenen Inhalten verglichen und fehlende Teile im eigenen Datenbestand ergänzt.

Diagrammelement	Beschreibung
Synchronisation von miradlo-Galerie A durchführen	Unter Berücksichtigung des Datenbestands aus miradlo-Galerie B und dem eigenen Datenbestand, führt miradlo-Galerie A eine Synchronisation durch. Dabei werden die Inhalte aus miradlo-Galerie B mit den eigenen Inhalten verglichen und fehlende Teile im eigenen Datenbestand ergänzt.
miradlo-Galerie soll synchronisiert werden	Die miradlo-Galerie A soll mit miradlo-Galerie B synchronisiert werden. Dies kann beispielsweise der Fall sein, wenn ein Benutzer neue Bilder der miradlo-Galerie A hinzugefügt hat. Die miradlo-Galerie A hat festgestellt, dass sie über eine funktionierende Netzwerkverbindung verfügt und beginnt nun mit der Synchronisation.
miradlo-Galerie wurde synchronisiert	Die miradlo-Galerie A wurde synchronisiert. miradlo-Galerie A und miradlo-Galerie B besitzen nun den gleichen Datenbestand.
Bilder in miradlo-Galerie A	Dieses Objekt stellt sämtliche Bilder dar, die zur miradlo-Galerie A gehören.
Bilder in miradlo-Galerie B	Dieses Objekt stellt sämtliche Bilder dar, die zur miradlo-Galerie B gehören.

2.3 Bilder suchen und ansehen

Ein Benutzer sucht in diesem Geschäftsprozess mithilfe eines Suchbegriffs. Die miradlo-Galerie stellt danach das Suchergebnis dar und der Benutzer kann die Bilder betrachten.

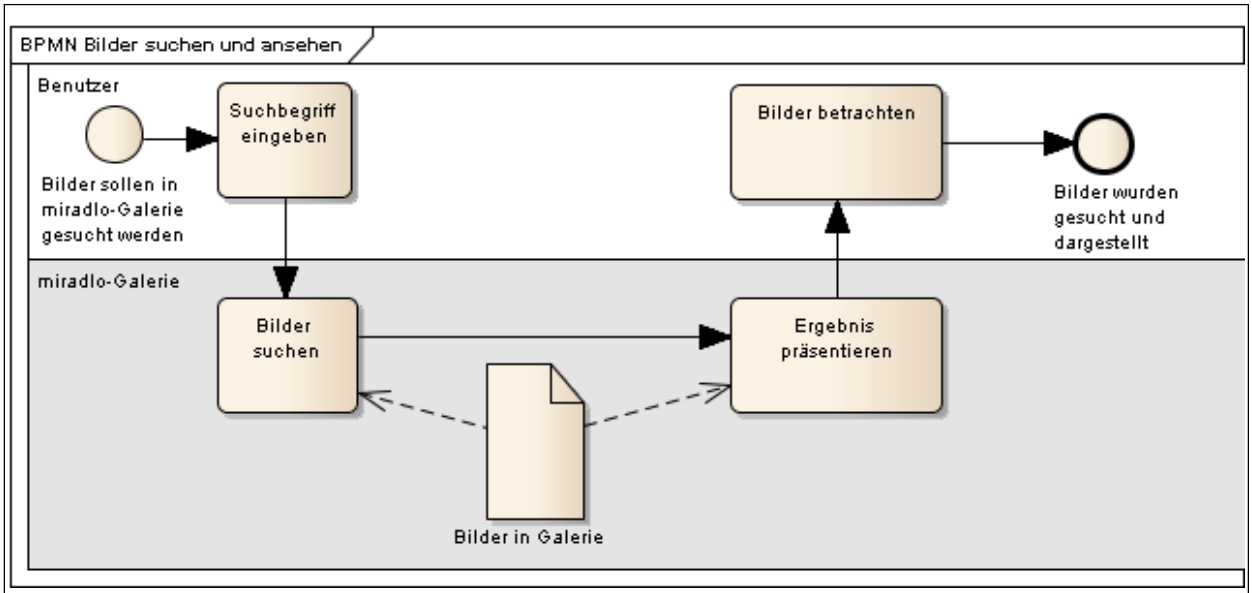

Abbildung 2.4: Bilder suchen und ansehen

Das vorstehende Geschäftsprozessdiagramm [26] illustriert, dass die miradlo-Galerie nach dem Empfangen des Suchbegriffs vom Benutzer, eine Suche durchführt. Dabei werden alle Bilder des eigenen Datenbestandes durchsucht. Jedes Bild im Datenbestand ist mit einem Titel und einer Beschreibung versehen, welche beim Suchvorgang berücksichtigt werden. Nachdem die Suche abgeschlossen ist, werden die Ergebnisse dem Benutzer präsentiert, welcher daraufhin einzelne Bilder betrachten kann.

Diagrammelement	**Beschreibung**
Suchbegriff eingeben	Der Benutzer gibt einen Suchbegriff ein, mit dem bestimmte Bilder in der miradlo-Galerie gefunden werden sollen. Der Suchbegriff könnte beispielsweise "Piratenpartei Infostand März" lauten.
Bilder suchen	Die miradlo-Galerie führt die Suche auf Basis ihres Datenbestandes durch.
Ergebnis präsentieren	Es werden die gefundenen Bilder samt ihrer Metadaten dem Benutzer präsentiert. Die Bilder sind dabei standardmäßig nach ihrer Übereinstimmung mit dem Suchbegriff sortiert.
Bilder betrachten	Die einzelnen Bilder der Suchanfrage werden vom Benutzer betrachtet.
Bilder sollen in miradlo-Galerie gesucht werden	Der Benutzer besitzt Bilder in der miradlo-Galerie, die er jetzt durchsuchen möchte. Die Bilder wurden zuvor beispielsweise mit einer Digitalkamera erstellt und der miradlo-Galerie zugeführt.
Bilder wurden gesucht und	Der Suchvorgang ist abgeschlossen und der Benutzer kann mit

Diagrammelement	Beschreibung
dargestellt	dem Suchergebnis seine Arbeit fortsetzen. Beispielsweise kann nun eine Bearbeitung der gefundenen Bilder erfolgen.
Bilder in Galerie	Dieses Objekt stellt die Bilder der miradlo-Galerie dar.

2.4 Bild bearbeiten

In diesem Geschäftsprozess werden Bilder durch einen Benutzer mithilfe der miradlo-Galerie bearbeitet. Beispielsweise kann in diesem Vorgang ein Benutzer ein bestimmtes Bild um 90° gegen den Uhrzeigersinn drehen.

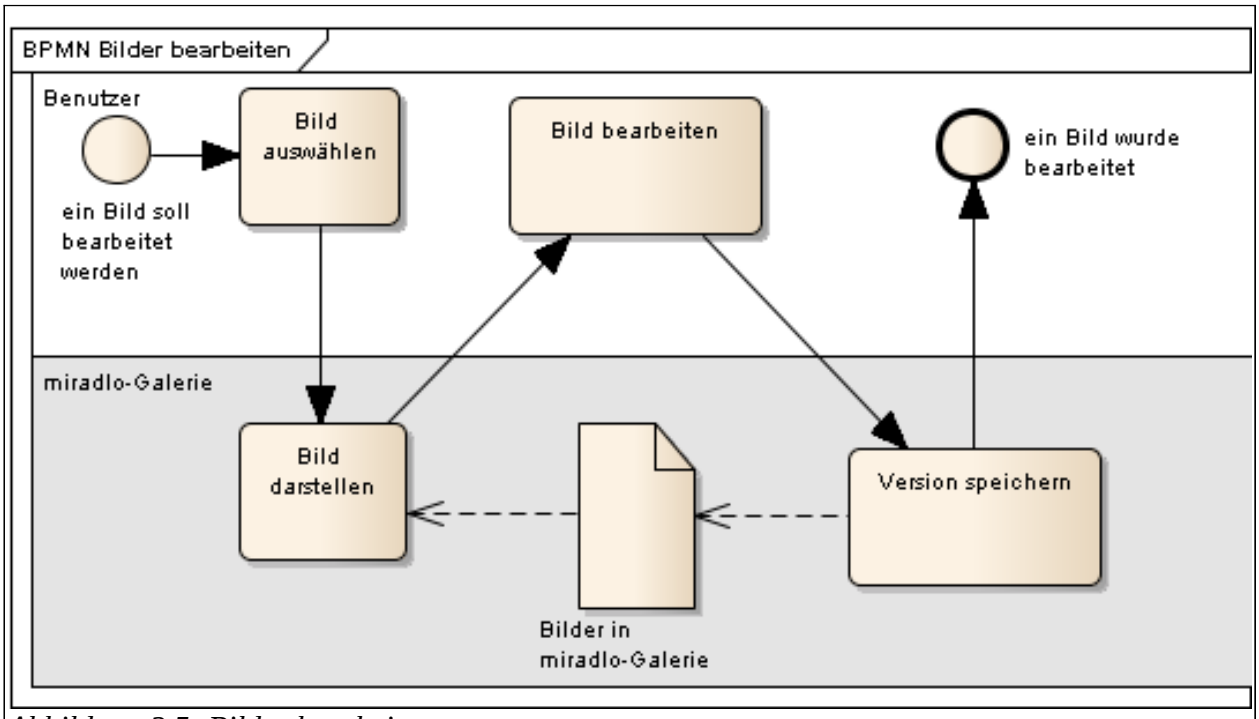

Abbildung 2.5: Bilder bearbeiten

Im vorstehenden Geschäftsprozessdiagramm [26] ist zu sehen, dass ein Benutzer nach dem Auswählen eines Bildes, eine Bearbeitung vornehmen kann. Die miradlo-Galerie speichert daraufhin die Eingabe.

Diagrammelement	Beschreibung
Bild auswählen	Der Benutzer wählt das zu bearbeitende Bild aus. Dieses Bild ist bereits in der miradlo-Galerie.
Bild darstellen	Die miradlo-Galerie zeigt das Bild dem Benutzer auf dem Bildschirm, um die erfolgreiche Auswahl zu bestätigen.
Bild bearbeiten	Der Benutzer führt die Bearbeitung durch. Dies umfasst die Eingabe eines Bearbeitungsschritts. Beispielsweise könnte dieser das Rotieren des Bildes gegen den Uhrzeigersinn auslösen.
Version speichern	Die miradlo-Galerie speichert die Version und schließt damit den

Diagrammelement	Beschreibung
	Prozess ab.
ein Bild soll bearbeitet werden	Der Benutzer besitzt Bilder in der miradlo-Galerie, die er jetzt durchsuchen möchte. Die Bilder wurden zuvor beispielsweise mit einer Digitalkamera erstellt.
ein Bild wurde bearbeitet	Das vom Benutzer ausgewählte Bild wurde bearbeitet.
Bilder in miradlo-Galerie	Dieses Objekt stellt die Bilder der miradlo-Galerie dar.

2.5 Bilder in externer Applikation nutzen

In diesem Geschäftsprozess fragen externe Applikationen bestimmte Bilder ab. Beispielsweise kann dadurch ein Blog eine bestimmte Bilderserie für einen seiner Artikel integrieren.

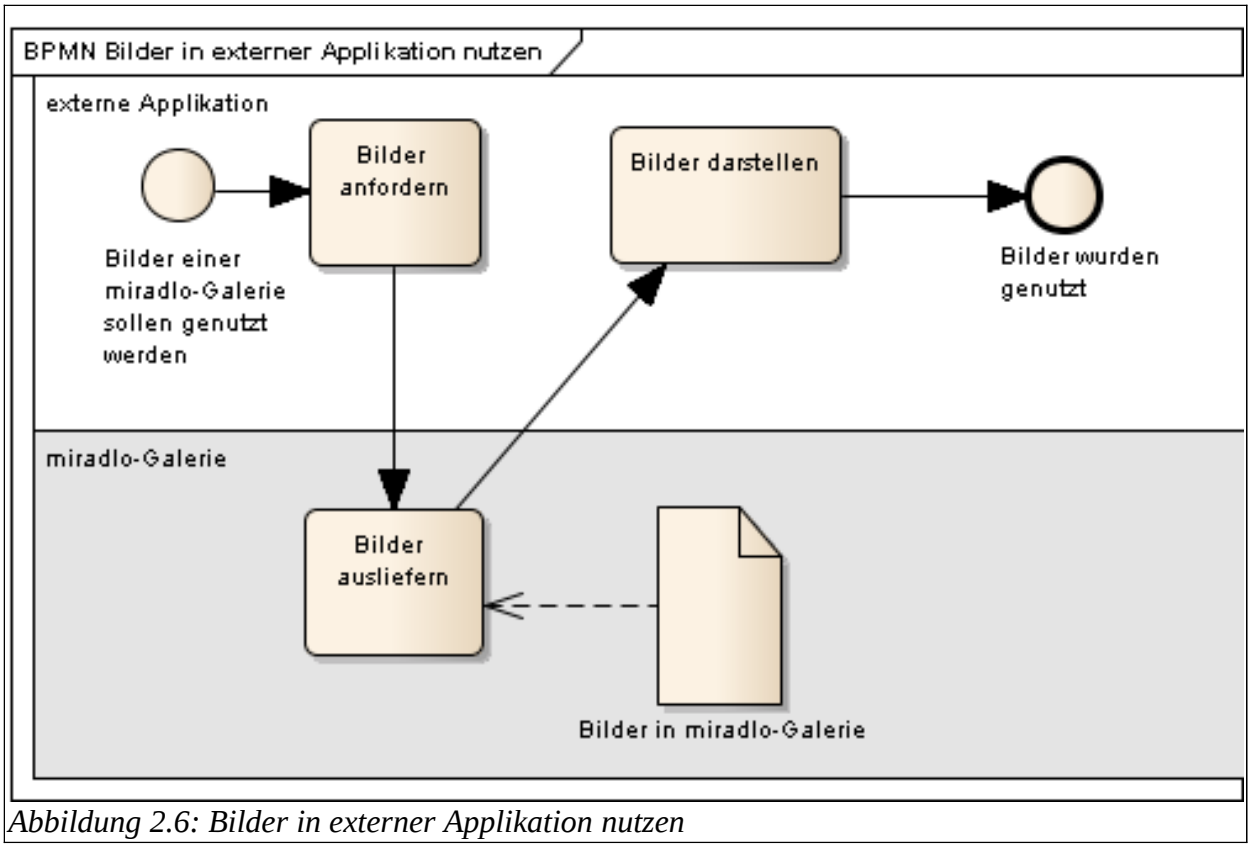

Abbildung 2.6: Bilder in externer Applikation nutzen

Das vorstehende Geschäftsprozessdiagramm [26] illustriert, dass eine externe Applikation bei Bedarf an der miradlo-Galerie bestimmte Bilder anfordert. Die miradlo-Galerie übermittelt die Bilder an die Applikation, wo sie dargestellt werden.

Diagrammelement	Beschreibung
Bilder anfordern	Die externe Applikation fordert bestimmte Bilder von der miradlo-Galerie an. Beispielsweise kann durch diesen Prozess eine Online-Zeitung bestimmte Bilder in ihren Artikeln integrieren.
Bilder ausliefern	Die miradlo-Galerie verarbeitet die Anfrage und liefert die Bilder an die externe Applikation. Beispielsweise kann dieser Vorgang eine Sammlung an Adressen zu den angeforderten Bildern umfassen.
Bilder darstellen	Die externe Applikation verarbeitet die Daten von der miradlo-Galerie und stellt die enthaltenen Bilder entsprechend dar.
Bilder einer miradlo-Galerie sollen genutzt werden	Eine externe Applikation möchte Bilder einer miradlo-Galerie verwenden. Dieser Vorgang kann bei Bedarf angestoßen werden.
Bilder wurden genutzt	Die externe Applikation hat die Bilder der miradlo-Galerie für ihre eigenen Zwecke eingesetzt. Es folgt eine Rückmeldung an die miradlo-Galerie.
Bilder in miradlo-Galerie	Dieses Objekt stellt die Bilder der miradlo-Galerie dar.

2.6 Bilder an externe Applikation senden

Eine miradlo-Galerie interagiert in diesem Geschäftsprozess mit einer externen Applikation. Dabei wird von der miradlo-Galerie beim Eintreten von vorher spezifizierten Ereignissen eine externe Applikation aufgerufen. Ereignisse können dabei beispielsweise das Vorhandensein eines neuen Bildes oder die Registrierung eines neuen Benutzers sein. Konkret kann in diesem Geschäftsprozesses eine automatisierte Meldung über Twitter [TWIT] verbreitet werden, dass es ein neues Bild gibt.

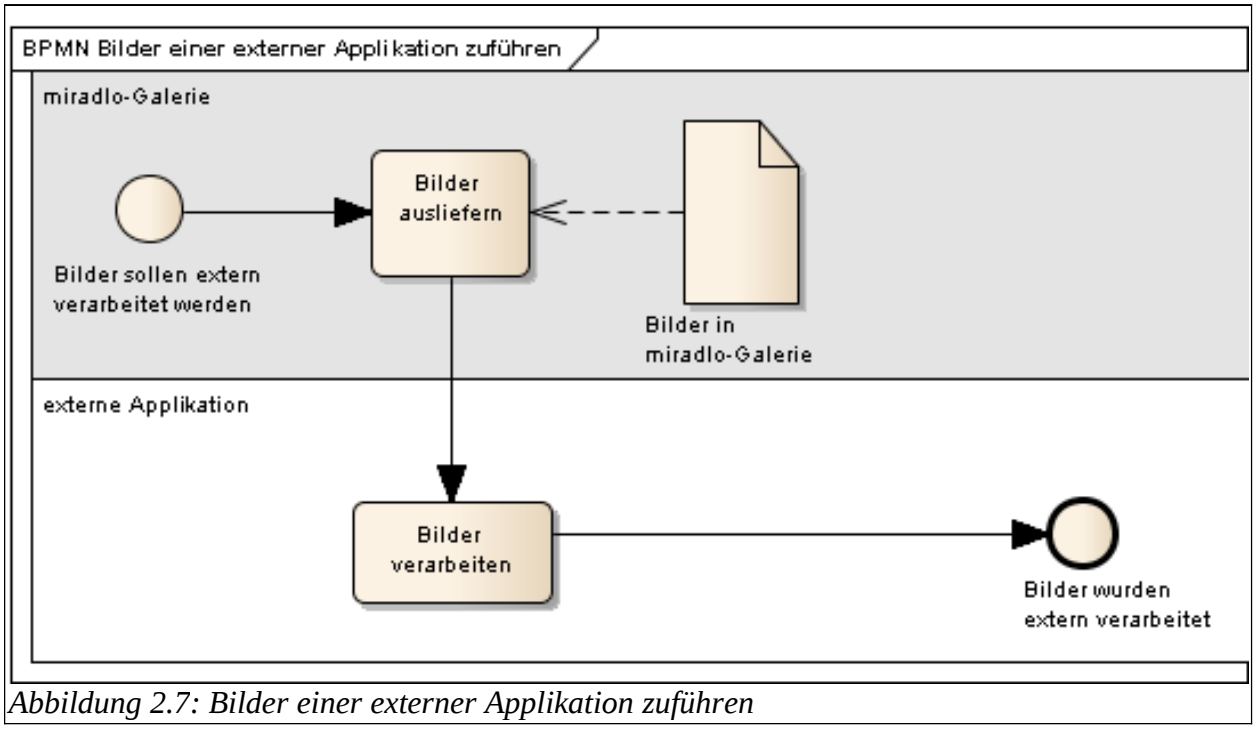

Abbildung 2.7: Bilder einer externer Applikation zuführen

Im oberhalb gezeigten Geschäftsprozessdiagramm [26] ist zu sehen, dass die miradlo-Galerie die Auslieferung der Bilder initiiert. Die externe Applikation nimmt die Bilder entgegen.

Diagrammelement	**Beschreibung**
Bilder ausliefern	Abhängig von den Einstellungen der miradlo-Galerie werden bestimmte Bilder an externen Anwendungen übermittelt.
Bilder verarbeiten	Der externen Applikation werden die Bilder übermittelt und können daraufhin verarbeitet werden. Konkret kann beispielsweise eine externe Applikation wie Twitter [TWIT] in diesem Vorgang Bilder verbreiten.
Bilder sollen extern verarbeitet werden	Je nach Einstellung der miradlo-Galerie werden beim Eintreten von bestimmten Ereignissen Bilder externen Applikationen zugesandt. Beispielsweise kann jede Stunde eine Zusammenstellung neuer Bilder an eine externe Applikation weitergeleitet werden.

Diagrammelement	Beschreibung
Bilder wurden extern verarbeitet	Die externe miradlo-Galerie hat die Bilder verarbeitet. Die miradlo-Galerie wird nicht informiert inwiefern dieser Vorgang erfolgreich verlaufen ist oder nicht.
Bilder in miradlo-Galerie	Dieses Objekt stellt die Bilder der miradlo-Galerie dar.

2.7 Stakeholder und Akteure

Die miradlo-Galerie steht mit mehreren Personen und Personengruppen in Verbindung. Deren Zufriedenheit entscheidet, ob die miradlo-Galerie ein Erfolg wird oder nicht.

Es "*können alle Individuen oder Gruppen als Stakeholder (Anspruchsgruppen) bezeichnet werden, die in der Unternehmung einen materiellen oder immateriellen Anspruch („stake“) haben. Als „Anteilseigner“ haben sie Anteil am bzw. Einfluss auf den Erfolg und Misserfolg eines Unternehmens. Der Anspruch eines Stakeholders am Unternehmen beruht im wesentlichen auf der Zurverfügungstellung von Ressourcen.*" [27]

Das folgende Schaubild zeigt alle Stakeholder und Akteure dieser Bachelorthesis.

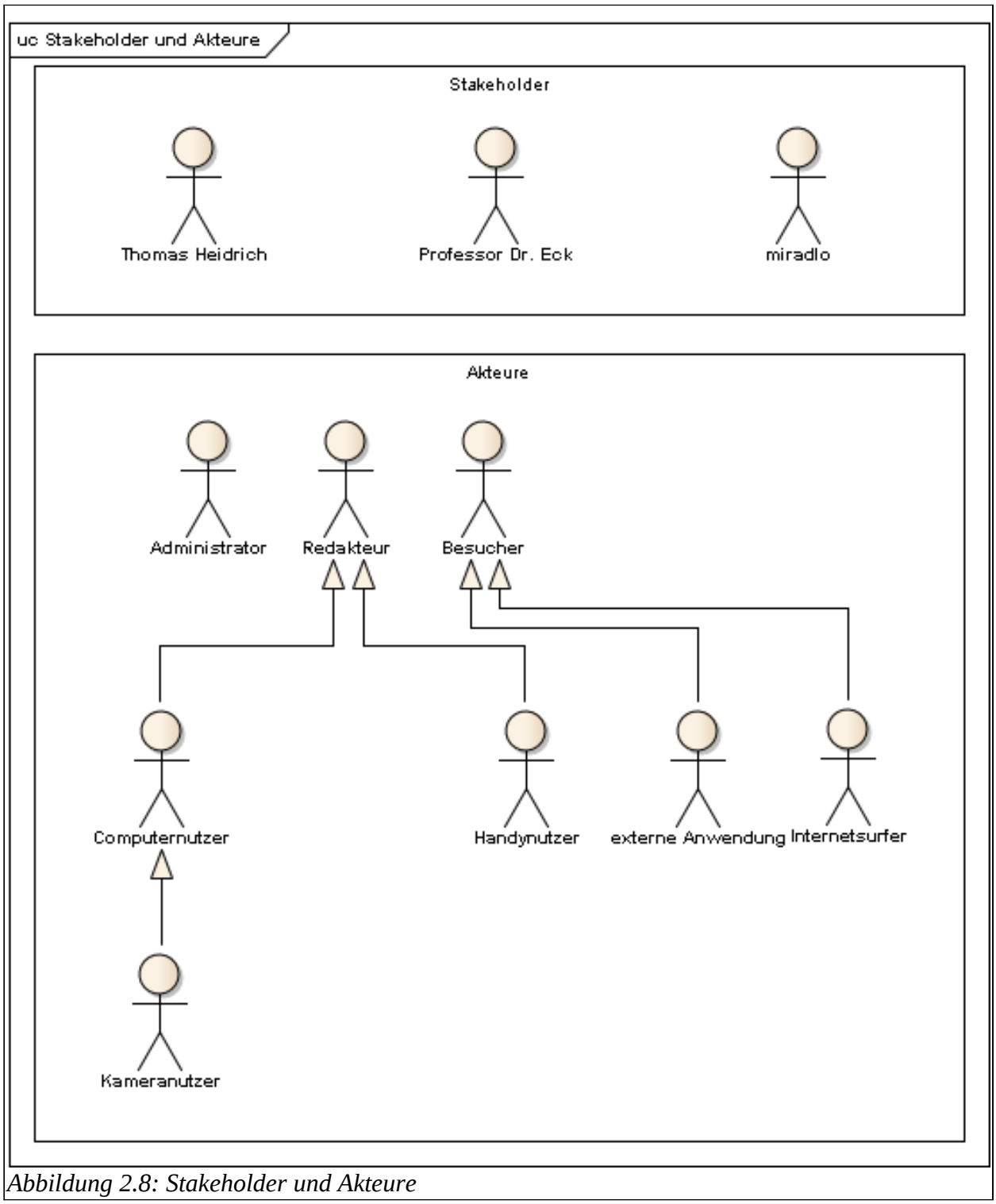

Abbildung 2.8: Stakeholder und Akteure

Diagrammelement	Beschreibung
Thomas Heidrich	Dieser Stakeholder arbeitet bei miradlo und ist Hauptentwickler der miradlo-Galerie. Seine Bachelorarbeit handelt vom Entwurf der Synchronisation der miradlo-Galerie.
Professor Dr. Eck	Dieser Stakeholder ist erster Prüfer der auf dem System beruhenden Bachelorarbeit. Er beurteilt die Qualität der Arbeit.
miradlo	Dieser Stakeholder ist Auftraggeber der miradlo-Galerie und zweiter Prüfer der auf dem System beruhenden Bachelorarbeit. Er wird die Weiterentwicklung koordinieren und Third-Level-Support anbieten.
Administrator	Dieser Akteur ist für die Verwaltung einer oder mehrerer miradlo-Galerien zuständig. Dies umfasst vor allem die Konfiguration und diverse Housekeeping-Aufgaben.
Redakteur	Dieser Akteur meldet sich an der miradlo-Galerie an, bevor er mit seiner Arbeit beginnt. Seine Arbeit besteht hauptsächlich aus dem Hinzufügen und Bearbeiten von Bildern.
Besucher	Dieser Akteur benutzt die miradlo-Galerie grundsätzlich ohne angemeldet zu sein. Dies bedeutet, dass er nur Zugriff auf Bereiche bekommt, die keinen Zugriffsbeschränkungen unterliegen.
Computernutzer	Dieser Akteur lädt eine große Anzahl an Bildern von seiner lokalen Festplatte in die miradlo-Galerie, um sie dort zu bearbeiten, ordnen und anderen zur Verfügung zu stellen.
Handynutzer	Dieser Akteur erstellt Bilder mit seinem Handy und sendet sie via E-Mail zur miradlo-Galerie, um sie dort anderen zur Verfügung zu stellen.
externe Anwendung	Dieser Akteur kann Bilder der miradlo-Galerie nutzen. Hierfür kann er selbst bei der miradlo-Galerie Bilder abrufen oder bei Bedarf von der miradlo-Galerie Bilder automatisch zugesandt bekommen.
Internetsurfer	Es handelt sich hier um den gemeinen Internetnutzer. Dieser ruft die miradlo-Galerie auf, schaut sich verschiedene Bilder an und verlässt die Seite direkt wieder.
Kameranutzer	Dieser Akteur lädt Bilder von seiner Digitalkamera auf seinen Computer und verarbeitet sie dort lokal. Danach lädt er die Bilder in die miradlo-Galerie und stellt sie damit anderen zur Verfügung.

3 Problemstellung und Analyse

Das Ziel der miradlo-Galerie ist es, vielen Benutzern, die auf unterschiedlichen Computern arbeiten, die Möglichkeit zu geben, ihre Bilder untereinander auszutauschen und in Bildergalerien ordnen zu können.

Eine Anforderung an das System ist, dass es damit umgehen kann, wenn die Kommunikation zwischen den verschiedenen Computern nur sporadisch möglich ist. Beispielsweise soll ein Benutzer im Zug seine Bilder einstellen, bearbeiten und später mit den anderen miradlo-Galerien abgleichen können, siehe folgendes Bild.

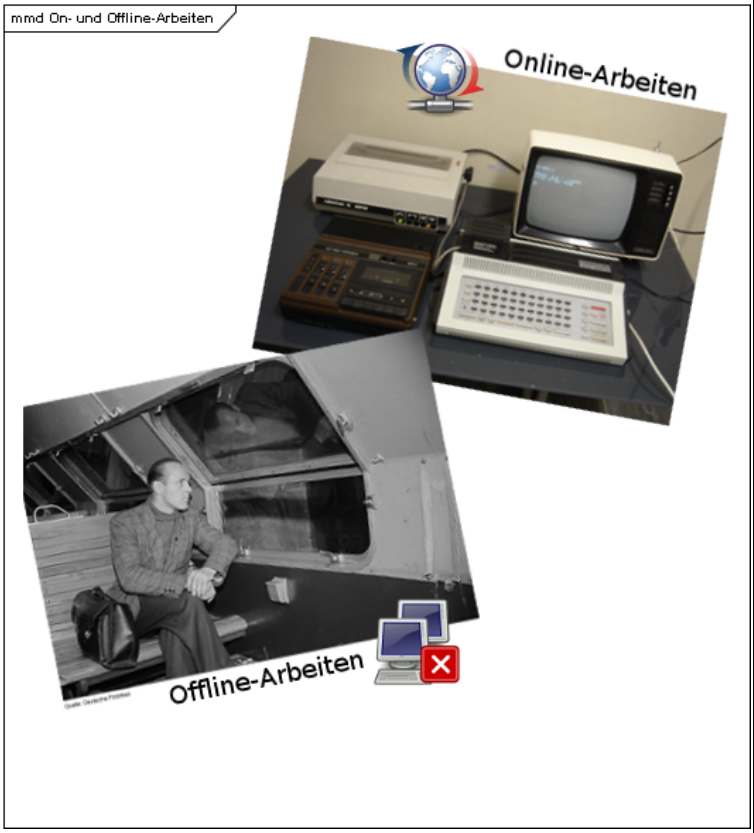

Abbildung 3.1: On- und Offline-Arbeiten

Die Synchronisation der miradlo-Galerien hat dabei ohne Nutzereingriff abzulaufen. Sobald Bildergalerien auf einem Computer freigegeben sind, können diese auch von anderen genutzt werden.

In diesem Kapitel werden alle auf der Hand liegenden Problemstellungen aufgezeigt, Lösungsvarianten analysiert und Designentscheidungen getroffen. Weiter werden diverse Anforderungen an das System ausgearbeitet. Diese Anforderungen finden sich im Anhang wieder.

3.1 Grundlagen

Die Firma miradlo [MIRAD] ist ein in Konstanz ansässiges Softwareunternehmen, welches sich auf die Realisierung von kundenspezifischen Webapplikationen spezialisiert hat. Um die meist

heterogenen Kundenanforderungen schnell und stabil umsetzen zu können, hat miradlo das Framework miradlokit entwickelt.

"*Bei miradlokit handelt es sich um einen auf CakePHP [CAKE] aufbauenden Webapplikations-baukasten, der diverse Basisanforderungen abdeckt.* [...]

Mit miradlokit werden alle Funktionen für miradlo-Kunden erstellt, die nicht in eingesetzter Standardsoftware enthalten sind. Weiterhin wird miradlokit für die Integration von unterschiedlichen Standardsoftwarekomponenten, z.B. unterschiedlichen Blogsysteme, verwendet. miradlokit ist ein komponentenbasiertes System, das aus einem allgemeinen Kern besteht, der durch spezialisierte Plugins erweitert wird." [0]

Es ist in miradlokit auf diese Weise möglich, verschiedene Inhaltselemente durch allgemeine Funktionen zu erweitern. Beispielsweise kann ein Plugin ein Bild bereitstellen, welches durch eine interaktive Kommentarfunktion ergänzt wird.

Mithilfe von miradlokit wurden unter anderem bisher Online-Zeitungen, Postsendungsverwaltungen, Kandidaten- und private Webseiten realisiert.

Aufgrund dieser Umstände wurde entschieden, die miradlo-Galerie als Plugin für miradlokit zu realisieren, siehe Anforderung Integration in miradlokit. Das folgende Diagramm illustriert den Aufbau von miradlokit. Das Diagramm ist an die Abbildung [28] des Pflichtenheftes [0] angelehnt.

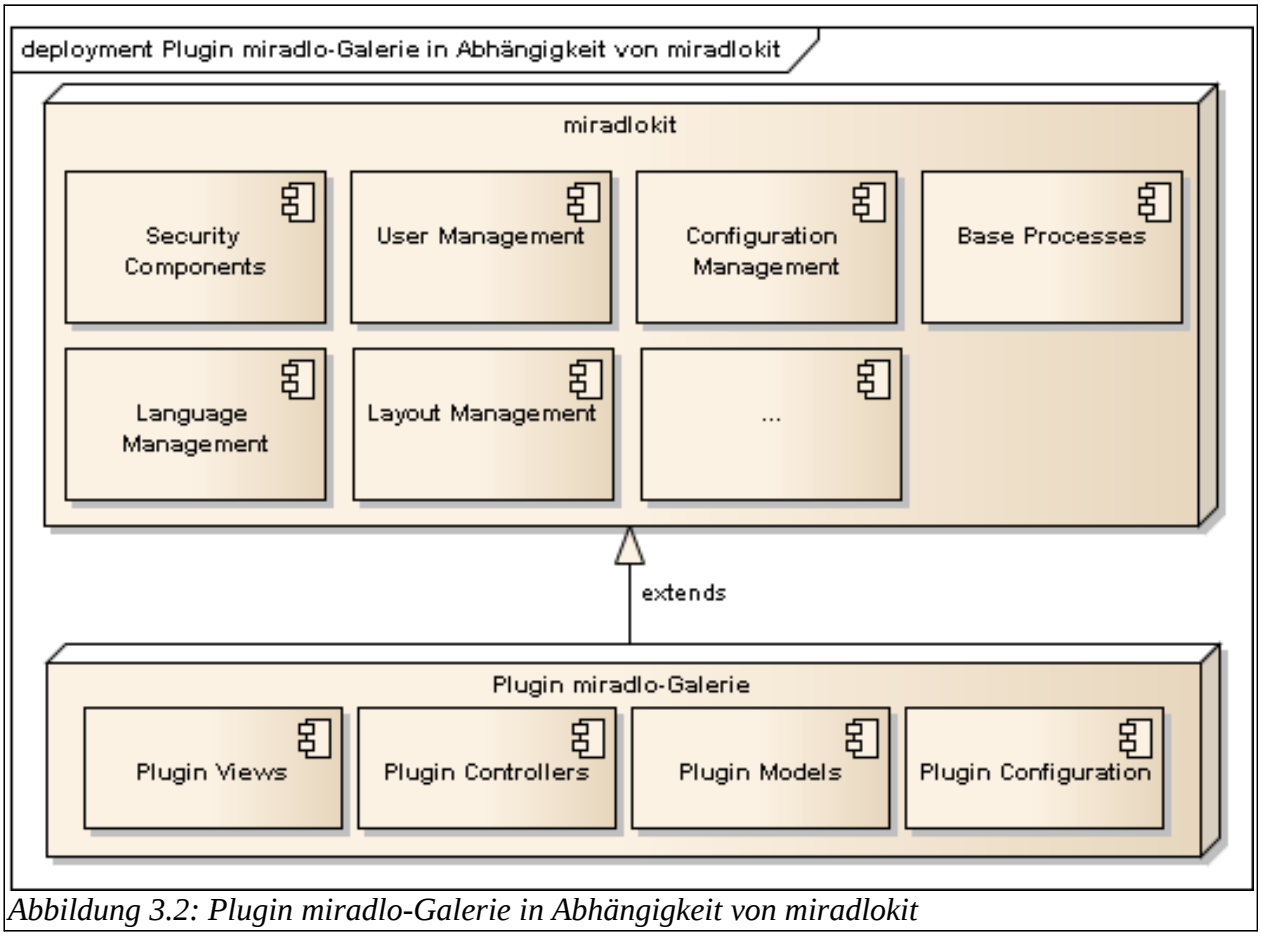

Abbildung 3.2: Plugin miradlo-Galerie in Abhängigkeit von miradlokit

3.2 Versionsverwaltung

Die miradlo-Galerie ist ein Verteiltes System. Bezugnehmend auf die *Einleitung* wird bei reger Nutzung erwartet, dass nach kurzer Zeit sehr viele verschiedene Bilder in unterschiedlichen Versionen entstehen.

Ein Bild besteht im Kontext der miradlo-Galerie aus dem Originalbild, Versionen und diversen Metadaten die das Bild genauer beschreiben. Eine Version eines Bildes entsteht, sobald ein Bild verändert und das Ergebnis abgespeichert wird. Beispielsweise wird eine Version erstellt, wenn ein Benutzer ein Bild um 90° gegen den Uhrzeigersinn dreht. Diese Version muss getrennt vom Originalbild abgespeichert und zusammen mit dem Originalbild verwaltet werden.

Die miradlo-Galerie muss die unterschiedlichen Versionen eines Bildes speichern und mit unterschiedlichen miradlo-Galerien synchronisieren können.

Bilder sind binäre Daten, die bei einer Synchronisation schnell zu hohen Datenvolumen führen können, die übertragen werden müssen. Der Umfang dieser Daten muss so weit wie möglich eingeschränkt werden, um einen performanten Abgleich zwischen den miradlo-Galerien zu ermöglichen.

In den folgenden Kapiteln werden verschiedene Varianten für die Versionsverwaltung beschrieben und bewertet.

3.2.1 Direkte Änderung

In einem Bildbearbeitungsprogramm, wie Microsoft Paint [PAINT] oder GIMP [GIMP], wird das Originalbild verändert, wenn ein Bild geöffnet, bearbeitet und in der selben Datei gespeichert wird.

Im Fall einer freigegebenen Bildergalerie müsste nun das veränderte Bild an alle miradlo-Galerien übertragen werden.

Weiterhin würde die Qualität des Bildes nach jeder Verarbeitung verschlechtert, falls verlustbehaftete Formate wie JPEG verwendet werden. "JPEG (Joint Photographic Expert Group) ist ein Format mit verlustbehafteter Kompression. Es wird für die Wiedergabe von Fotos im WWW eingesetzt." [13]

Vorteile

Wenn immer nur eine Version eines Bildes im System vorhanden ist, kann der Verwaltungsaufwand entsprechend reduziert werden.

Es ist eine begrenzte Anzahl an Bildern vorhanden. Eine Inflation an Versionen wird nicht entstehen.

Nachteile

Auf das ursprünglich in die miradlo-Galerie eingeführte Originalbild kann nicht mehr zugegriffen werden, da die binären Daten verändert wurden.

Fazit

Dieses Vorgehen kann in Erwägung gezogen werden, wenn weder eine Historie der Bearbeitungen noch die Ursprungsversion eines Bildes benötigt wird. Bei jeder Veränderung muss das dadurch neu entstandene Bild in vollem Umfang an alle miradlo-Galerien übermittelt werden.

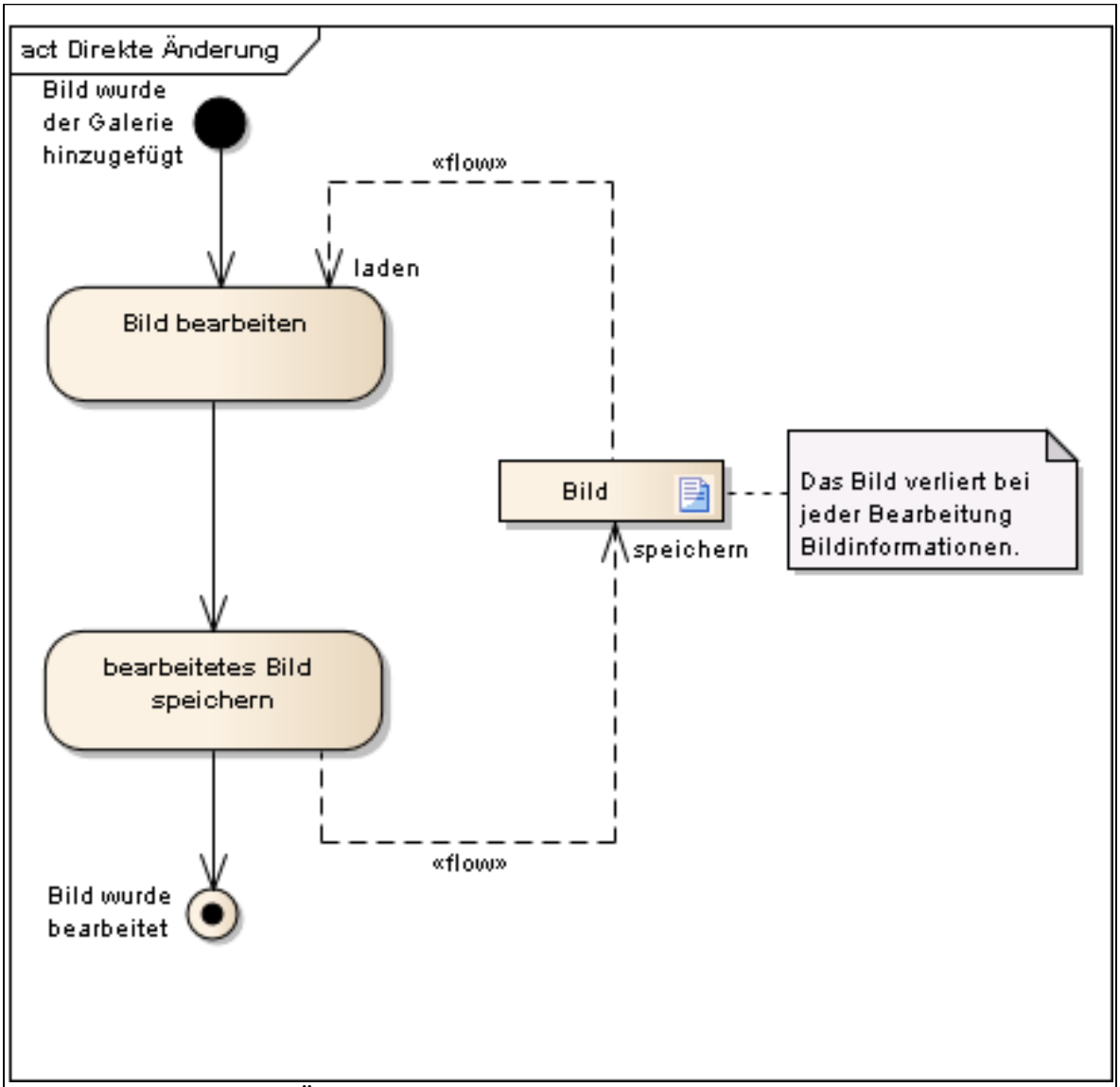

Abbildung 3.3: Direkte Änderung

Im vorstehenden Diagramm wird illustriert, wie die Bearbeitung eines Bildes innerhalb der miradlo-Galerie abläuft, wenn jede Änderung direkt auf das Bild angewendet wird.

Bei jeder Bearbeitung werden die folgenden Schritte ausgeführt.

- Bearbeiten des bereits in der miradlo-Galerie befindlichen Bildes
- Ersetzen des Bildes durch dessen bearbeitete Version

Durch die wiederkehrende Verarbeitung wird das Originalbild stetig verändert und verliert nach und nach Bildinformationen und somit seine Qualität.

Diagrammelement	Beschreibung
Bild bearbeiten	In diesem Vorgang wird eine Änderung auf dem Bild angewendet. Beispielsweise könnte dies das Skalieren des Bildes auf 50% seiner ursprünglichen Größe darstellen.
bearbeitetes Bild speichern	Bei diesem Vorgang wird das ursprüngliche Bild gelöscht und von der soeben bearbeiteten Version ersetzt.
Bild	Dieses Objekt stellt ein in die miradlo-Galerie geladenes binäres Bild samt seiner Metadaten dar. Da dieses Bild erneut verwendet wird, können Verluste entstehen.
Bild wurde der Galerie hinzugefügt	Der Ausgangspunkt ist ein in der miradlo-Galerie existierendes Bild, welches bearbeitet werden soll.
Bild wurde bearbeitet	Das Bild wurde bearbeitet und es existiert keine Möglichkeit, die ursprüngliche Version wiederherzustellen.

3.2.2 Originalbild mit bearbeiteten Versionen

Wenn eine mögliche Abnahme der Bildqualität bei Bildern mit verlustbehafteter Kompression nicht akzeptabel ist, kann diese Variante der Versionierung in Betracht gezogen werden.

Das Hinzufügen eines Bildes in die miradlo-Galerie hat zur Folge, dass es unmittelbar gesondert als Originalbild abgespeichert wird und ab diesem Zeitpunkt nur noch für lesenden Zugriff zur Verfügung steht.

Wenn eine Bearbeitung durchgeführt werden soll, basiert diese immer auf einem Originalbild. Die daraus entstehende Version wird dabei mit dem Originalbild verknüpft abgelegt.

Dieses Vorgehen garantiert, dass ein möglicher verlustbehafteter Komprimierungsalgorithmus nie auf ein Originalbild angewendet wird.

Damit das zu übertragende Datenvolumen bei einer Synchronisation klein bleibt, kann zusätzlich nur eine Version zugelassen werden.

Vorteile

Der Vorteil dieser Variante ist, dass das Original eines einst der Galerie hinzugefügten Bildes in seiner Form erhalten bleibt und ein kontinuierlicher Qualitätsverlust nicht stattfindet.

Nachteile

Jede neue Version muss in vollem Umfang abgespeichert werden. Ein weiterer Nachteil ist, dass durch dieses Vorgehen weitere Bearbeitungen von Versionen von vornherein ausgeschlossen sind, was möglicherweise Nutzer irritieren könnte.

Fazit

Dieses Vorgehen bietet sich an, wenn eine Historie der Bearbeitungen und die Ursprungsversion eines Bildes benötigt wird.

Bei jeder Veränderung muss jedoch das neu entstandene Bild in vollem Umfang an alle miradlo-Galerien übertragen werden. Falls mehrere Versionen zugelassen werden, steigt die zu übertragende Datenmenge bei einer Synchronisation stark an.

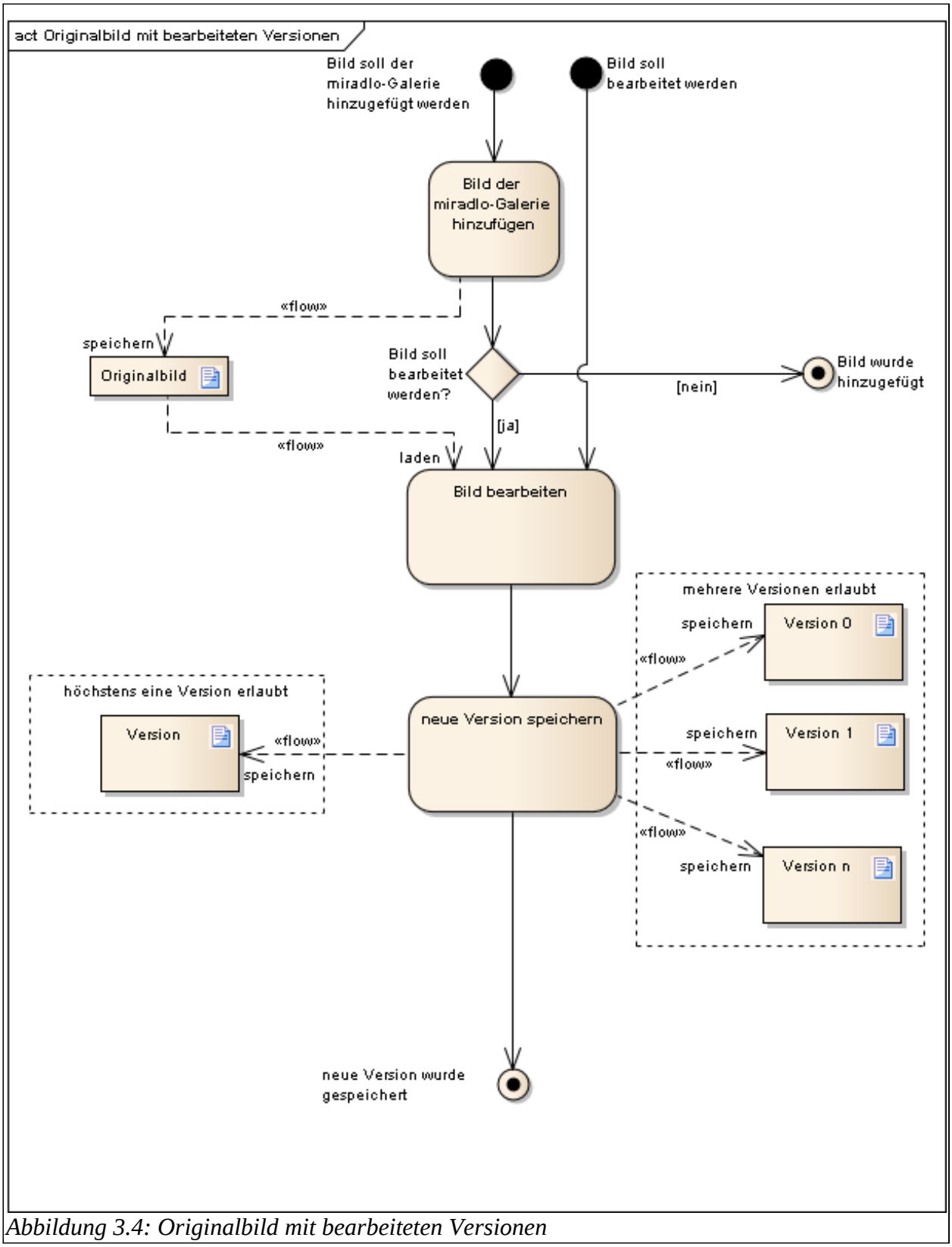

Abbildung 3.4: Originalbild mit bearbeiteten Versionen

In obigem Diagramm wird illustriert, wie Bearbeitungen eines Bildes innerhalb der miradlo-

Galerie ablaufen, wenn diese immer vom Originalbild ausgehen und entweder zu exakt einer Version oder zu mehreren Versionen führen.

Sobald ein neues Bild der miradlo-Galerie hinzugefügt wird, legt diese es als Originalbild ab. Auf dieses Original wird nur noch lesender Zugriff erlaubt und es wird bei eventuell nachfolgenden Bearbeitungen immer als Ausgangspunkt verwendet.

Falls höchstens eine Version erlaubt ist, findet bei jeder Bearbeitung eines Originalbildes eine Ersetzung der bereits existierenden Version statt.

Falls jedoch mehrere Versionen erlaubt sind, wird bei jeder Bearbeitung eine neue Version erzeugt und separat abgespeichert.

Diagrammelement	**Beschreibung**
Bild der miradlo-Galerie hinzufügen	Es wird ein Bild der miradlo-Galerie übergeben, welches daraufhin als Originalbild gesondert abspeichert wird.
Bild bearbeiten	In diesem Vorgang wird eine Änderung auf dem Originalbild angewendet. Beispielsweise könnte dies das Zuschneiden des Bildes auf den wesentlichen Inhalt darstellen.
neue Version speichern	Es wird bei diesem Vorgang eine neue Version gespeichert. Falls höchstens eine Version erlaubt ist, so wird die evtl. bereits existierende Version des Originalbildes überschrieben.
Version 0, Version 1, Version n	Dies isind die zeitlich gesehen erste, zweite und n-te Version des Originalbildes.
Originalbild	Dieses Objekt stellt ein in die miradlo-Galerie geladenes Bild samt seiner Metadaten dar. Die Besonderheit daran ist, dass nach dem Erstellen nur noch lesender Zugriff erlaubt ist, solange sich das Bild in der miradlo-Galerie befindet.
Version	Dieses Objekt stellt die Version eines Originalbildes samt seiner Metadaten dar. Die Besonderheit daran ist, dass eine erneute Bearbeitung des Originalbildes diese Version überschreibt.
Bild soll bearbeitet werden?	An dieser Verzweigung wird entschieden, ob das eben hinzugefügte Originalbild weiter bearbeitet werden soll.
Bild wurde hinzugefügt	Das Originalbild wurde der miradlo-Galerie hinzugefügt und kann zu einem späteren Zeitpunkt bearbeitet werden.
Bild soll bearbeitet werden	In diesem Ausgangspunkt soll ein in der miradlo-Galerie existierendes Bild bearbeitet werden.
Bild soll der miradlo-Galerie hinzugefügt werden	Der Ausgangspunkt ist, dass ein Bild der miradlo-Galerie hinzugefügt werden soll.
neue Version wurde gespeichert	Das vom Originalbild abgeleitete Bild wurde gespeichert.

3.2.3 Originalbild mit Transformationsbeschreibungen

Der Ansatz *Originalbild* mit bearbeiteten Versionen ist unter Berücksichtigung der Anforderung *REQ1007 - Kein Qualitätsverlust bei der Manipulation von Bildern, Seite 103* nicht in der Lage, mehrere Bearbeitungsschritte in Folge zu erlauben. Die erneute und möglicherweise verlustbehaftete Kompression des Bildes führt sonst zu einer stetigen Abnahme der Bildqualität. Da die Einschränkung auf einen Bearbeitungsschritt für viele Benutzer nicht akzeptabel wäre, wurde die Anforderung *REQ1004 - miradlo-Galerie Name und Beschreibung mehrsprachig Seite 97* gestellt und der folgende Ansatz entwickelt.

Neben einer Version wird bei einer Bearbeitung eine Transformationsbeschreibung erzeugt. Diese Beschreibung enthält Informationen, auf welchen Bearbeitungsschritten eine bestimmte Version beruht. Wird eine Version erneut bearbeitet, so wird eine neue Transformationsbeschreibung erzeugt. Diese enthält nun Bearbeitungsschritte, die sowohl von der bereits existierenden Version als auch von der eben durchgeführten Bearbeitung stammen.

Dies ermöglicht der miradlo-Galerie, beim Speichern der neuen Version nur eine Dekodierung und erneute Kodierung durchführen zu müssen.

Vorteile

Die Galerie kann die momentan nicht benötigten Versionen löschen, um freien Speicherplatz zu gewinnen. Gelöschte Versionen können mithilfe der vorhandenen Transformationsbeschreibungen und Originalbilder jederzeit wieder erzeugt werden.

Zwischen den einzelnen miradlo-Galerien werden nur die Originalbilder und die Transformationsbeschreibungen abgeglichen.

Die Versionen können durch die Transformationsbeschreibungen in jeder miradlo-Galerie bei Bedarf selbst erstellt werden.

Nachteil

Neben den Originalbildern müssen auch deren Transformationsbeschreibungen und die daraus resultierenden Versionen verwaltet werden.

Fazit

Dieses Vorgehen stellt eine umfassende Versionsverwaltung dar, welche mit Historien von Bearbeitungen und Ursprungsversionen von Bildern umgehen kann. Die Versionsverwaltung benötigt dabei lediglich die Änderungen zwischen den Bildern. Damit kann sie alle Versionen eines vorhandenen Originalbildes rekonstruieren.

In folgendem Diagramm wird illustriert, wie Bearbeitungen eines Bildes innerhalb der miradlo-Galerie ablaufen, wenn Transformationsbeschreibungen nach jedem Bearbeitungsprozess erzeugt werden.

Wird ein neues Bild der miradlo-Galerie hinzugefügt, so wird es als Originalbild abgelegt und ist ab diesem Zeitpunkt nur noch lesend zugreifbar.

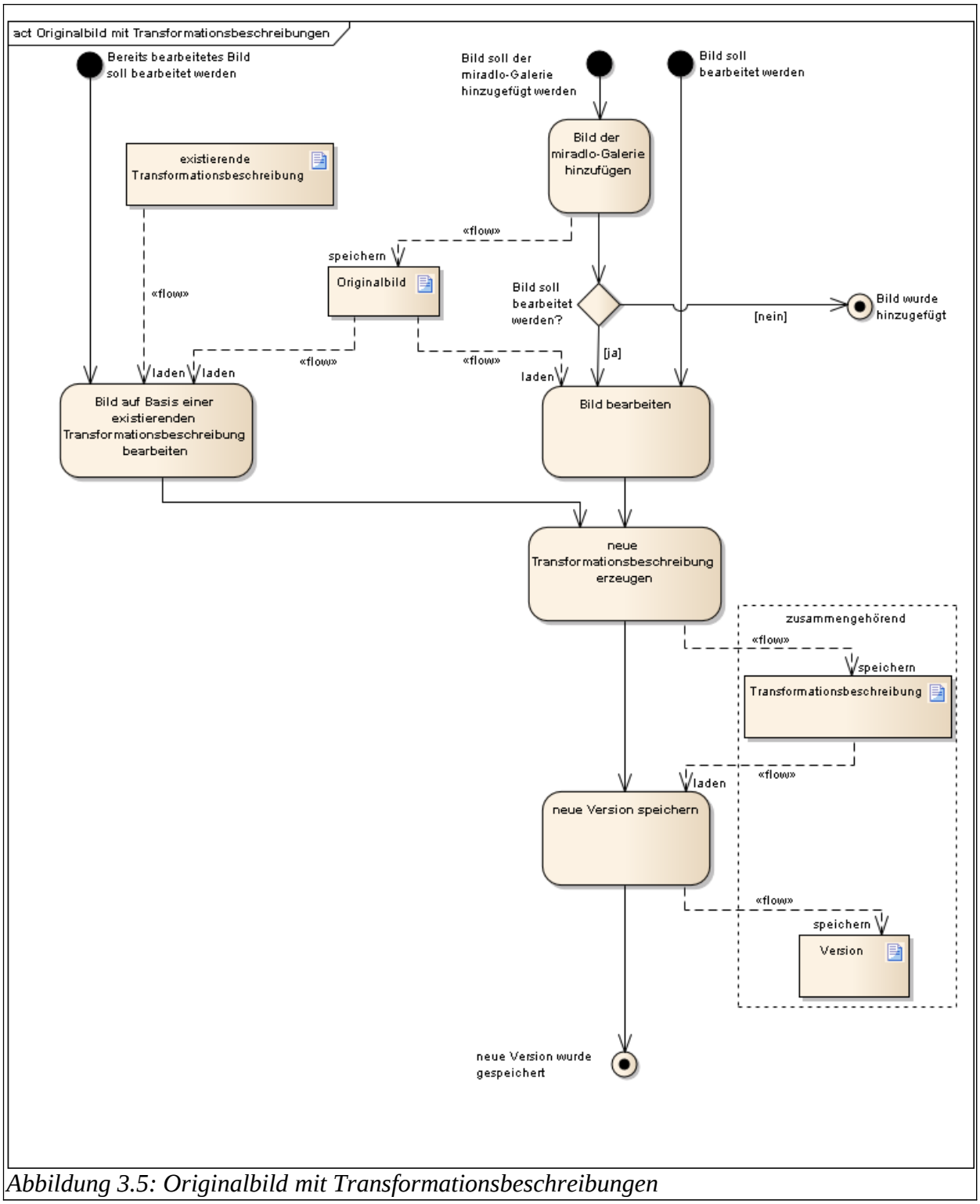

Abbildung 3.5: Originalbild mit Transformationsbeschreibungen

Es ist möglich, ein bereits bearbeitetes Bild weiter zu bearbeiten. Hierfür wird eine existierende Transformationsbeschreibung als Ausgangspunkt einer Bearbeitung genommen. Die aus dem Vorgang resultierende Transformationsbeschreibung stellt folglich eine Erweiterung der existierenden dar.

Immer, wenn eine neue Transformationsbeschreibung erzeugt wurde, speichert die miradlo-Galerie eine neue Version, um das Resultat des Bearbeitungsprozesses sichtbar zu machen.

Diagrammelement	Beschreibung
Bild auf Basis einer existierenden Transformationsbeschreibung bearbeiten	Es wird eine Änderung auf einer Version angewendet. Dabei werden die Bearbeitungsschritte aus der zur Version gehörenden Transformationsbeschreibung als Ausgangspunkt verwendet.
neue Version speichern	Es wird die bearbeitete Version eines Originalbildes als neue Version gespeichert. Um dies zu erreichen, werden die einzelnen Bearbeitungsschritte der im vorherigen Vorgang erzeugten Transformationsbeschreibung nacheinander auf das Originalbild angewendet.
neue Transformationsbeschreibung erzeugen	Es werden die im vorherigen Vorgang durchgeführten Bearbeitungsschritte in eine neue Transformationsbeschreibung überführt.
Transformationsbeschreibung	Dieses Objekt stellt eine Beschreibung aller Bearbeitungsschritte dar, die auf einem Originalbild durchgeführt wurden. Im Diagramm illustrierten Beispiel, kann diese Beschreibung den folgenden Inhalt haben. •verwendetes Originalbild •Erstellungsdatum der Transformationsbeschreibung •Bildtitel •Bildbeschreibung •Transformationsschritt 1: Rotieren um 90° im Uhrzeigersinn •Transformationsschritt 2: Skalieren auf 50% •resultierende Bilddimension (Höhe, Breite)
Version	Dieses Objekt stellt eine bearbeitete Version eines Originalbildes samt seiner Metadaten dar. Die Besonderheit daran ist, dass es auf Basis der durchgeführten Bearbeitungsschritte erstellt wurde. Diese Schritte sind in der Transformationsbeschreibung gespeichert.
existierende Transformationsbeschreibung	Dieses Objekt stellt eine bereits existierende Beschreibung aller Bearbeitungsschritte dar, die bereits einmal auf einem Originalbild durchgeführt wurden. Im Diagramm illustrierten Beispiel, kann diese Beschreibung den folgenden Inhalt haben. •verwendetes Originalbild •Erstellungsdatum der Transformationsbeschreibung •Bildtitel •Bildbeschreibung •Transformationsschritt 1: Rotieren um 90° im Uhrzeigersinn •resultierende Bilddimension (Höhe, Breite)
Bereits bearbeitetes Bild soll bearbeitet werden	Es existiert bereits eine Version eines Originalbildes in der miradlo-Galerie. Diese Version soll nun als Ausgangspunkt einer weiteren Bearbeitung dienen.

3.2.4 Bewertung der Versionsverwaltungsansätze

Die miradlo-Galerie ist ein Verteiltes System und die drei kritischen Ressourcen sind der Verbrauch an Netzwerkbandbreite und -verbindungen, Speicherplatz und Prozessorzeit [CPUT].

"*Rund zehn Jahre nach Einführung der ersten DSL-Anschlüsse knacken erste Angebote die 100-MBit/s-Marke. Die sind noch lange nicht überall verfügbar, die Provider bauen aber massiv aus. In den Städten werden schnelle Anschlüsse bald Standard und eröffnen neue Nutzungsmöglichkeiten.*" [25]

Auch wenn in der heutigen Zeit nahezu allen Benutzern Breitbandanschlüsse zur Verfügung stehen, ist eine hohe Netzwerklast der Faktor eines Verteilten Systems, der am kritischsten zu betrachten ist. Je höher die Belastung des Netzwerks im regulären Betrieb ist, desto kleiner ist die Anzahl der miradlo-Galerien, mit denen gleichzeitig kommuniziert werden kann. Weiter steht besonders bei Mobilgeräten nur sporadisch eine relativ langsame Netzwerkanbindung zur Verfügung, wodurch ein Abgleich so zeitsparend wie möglich erfolgen muss. Die Anzahl der Netzwerkverbindungen ist besonders dann relevant, wenn komplexe zustandsabhängige Transportprotokolle eingesetzt werden. In diesem Fall muss das Betriebssystem des Rechners, der die miradlo-Galerie ausführt, sämtliche kommunikationsrelevanten Datenstrukturen bis zur Terminierung der Verbindung im Speicher halten.

Nach der Netzwerklast ist die Speicherlast als kritisch anzusehen. Die Größe von verfügbarem Festplatten- und Hauptspeicher ist auf jedem System unterschiedlich. Eine völlige Auslastung einer oder beider Ressourcen hat erhebliche Leistungseinbußen zur Folge und führt im schlimmsten Fall zum Ausfall der miradlo-Galerie. Heutzutage ist Speicher verhältnismäßig günstig und kann bei Bedarf nachgerüstet werden, insofern der Administrator frühzeitig über die Speicherknappheit informiert wird.

Die Prozessorlast ist auf heutigen Systemen am unkritischsten. Mehrere Prozessorkerne mit Caches im Megabytebereich und mehreren Hauptspeicherbussen stellen für gewöhnlich ausreichend Rechenleistung zur Verfügung. Bei miradlo wird auf der Produktionsmaschine ein handelsüblicher Allzweck-Mehrkernprozessor mit AMD64-Mikroarchitektur eingesetzt.

Für die Realisierung der miradlo-Galerie wird die Variante *Originalbild mit Transformationsbeschreibungen* eingesetzt. Sie benötigt zur Erstellung der einzelnen Versionen eines Originalbildes lediglich das Originalbild selbst und die Transaktionsbeschreibungen. Dieses Vorgehen bietet somit die geringste zu übertragende Datenmenge pro Version eines Bildes. Bei der Erstellung einer Version auf Basis einer Transaktionsbeschreibung und dem Originalbild entsteht CPU-Last. Die Anforderung *REQ0021 - Überlastung bei der Erstellung einer Version verhindern*, *Seite 111*, beschreibt diesen Punkt genauer und hält fest, dass der Überlastfall bei der Realisierung zu bedenken ist.

3.3 Datenverwaltung

Im *Kapitel 3.2, Versionsverwaltung, Seite 22,* wurde definiert, dass auf jedem Originalbild mehrere Versionen basieren können.

Um eine große Anzahl an Bildern auf Persistenzebene zu verwalten, bedarf es eines definierten Konzeptes. Mit dessen Hilfe muss das System in die Lage versetzt werden, Bilder so zu sortieren, dass sie bei einer Suche schnell wiedergefunden werden. Es muss darüberhinaus Möglichkeiten bereitstellen, Bilder von der Synchronisation mit anderen miradlo-Galerien auszuschließen.

Im folgenden werden diese Punkte in verschiedenen Konzepten betrachtet und bewertet.

3.3.1 Bildergalerien

In dieser Variante dienen Bildergalerien als Mittel der Strukturierung.

Jede miradlo-Galerie besitzt Bildergalerien. Jede Bildergalerie besitzt wiederum viele Bilder, wobei jedes Bild exakt einer Bildergalerie zugeordnet ist.Es wird grundsätzlich zwischen freigegebenen und lokalen Bildergalerien unterschieden. Nur freigegebenen Bildergalerien wird eine Synchronisation ermöglicht.

Die Suche erfolgt auf der Ebene der Bildergalerien und auf der Ebene der Bilder. Dies bedeutet, dass auf beiden Ebenen eine Textsuche auf Basis von Titeln und Beschreibungen der jeweiligen Elemente stattfindet.

Vorteile

Bei diesem Ansatz ist es möglich, ein Originalbild mehrfach in verschiedene Bildergalerien zu laden. Die Benutzer von verschiedenen Bildergalerien sind somit schwach gekoppelt und eine Mehrfachverwendung in unterschiedlichem Kontexten ist möglich.

Nachteile

Bildergalerien bilden Strukturen, die verwaltet werden müssen. Die Akzeptanz von Benutzern kann geschmälert werden, da grobe Strukturen vorgegeben werden. Dieser Effekt ist besonders stark, wenn die Administration vernachlässigt wird. Bildergalerien sind ein starres und konventionelles Konzept, um Bilder einzuordnen. Den Benutzern werden dadurch verhältnismäßig wenig Freiräume eingeräumt.

Fazit

Dem Benutzer werden durch Bildergalerien starre Strukturen vorgegeben. Originalbilder können mehrfach hinzugefügt werden, wenn sie auf verschiedene Bildergalerien aufgeteilt werden.

Das folgende Diagramm illustriert diesen Aufbau.

Im folgendem Diagramm sind die miradlo-Galerien A und B abgebildet. Die miradlo-Galerie A besitzt sowohl lokale als auch freigegebene Bildergalerien. Die miradlo-Galerie B besitzt lediglich die Bildergalerie 2, welche mit der miradlo-Galerie A synchron gehalten wird.

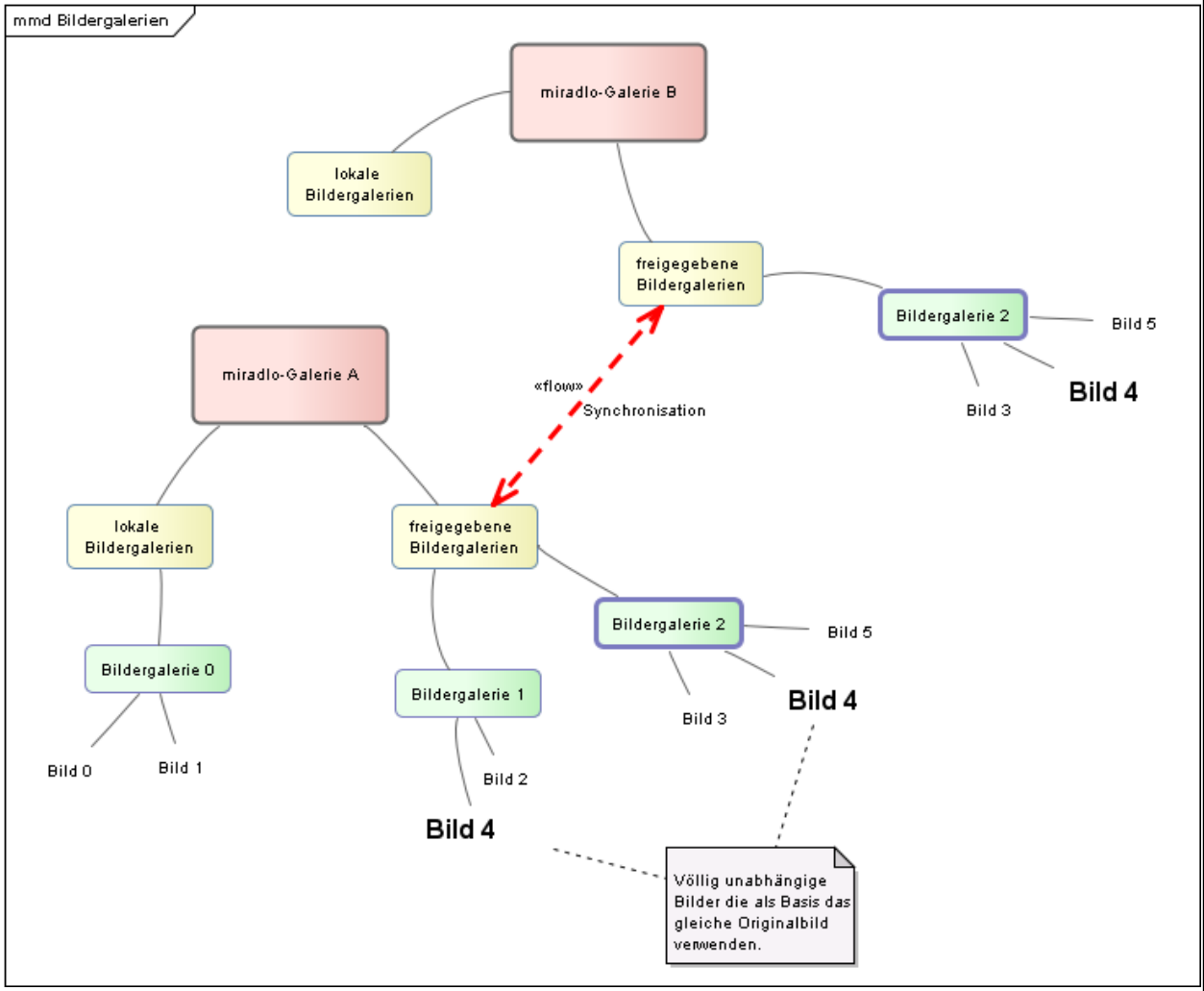

Abbildung 3.6: Bildergalerien

3.3.2 Tags

Die Strukturierung findet in dieser Variante auf Basis von Tags [TAG] statt.

In einer miradlo-Galerie wird grundsätzlich nur zwischen lokalen und freigegebenen Bildern unterschieden. Nur freigegebene Bilder werden zwischen miradlo-Galerien abgeglichen. Lediglich Tags, wie "Piratenpartei", "Urlaub", "25.11.2010" werden bei der strukturierten Darstellung und Suche berücksichtigt.

Vorteile

Es werden praktisch keine Strukturen vorgegeben. Der Benutzer kann die Gestaltung so vornehmen, wie er es möchte. Durch diese individuell anpassbare Struktur kann die Akzeptanz verhältnismäßig hoch sein.

Nachteile

Durch die fehlenden Strukturen wird der Benutzer nicht geführt und praktisch allein gelassen. Besonders bei großen thematisch zusammenhängenden Bildmengen wird es schnell unübersicht-

lich. Die Suche hängt vollkommen von Tags ab. Ohne das Wissen der vergebenen Tags, ist eine erfolgreiche Suche schwer. Tags müssen außerdem über alle miradlo-Galerien einheitlich verwendet werden. Beispielsweise könnte der Tag "PIRATEN" auf einer miradlo-Galerie für Bilder der Piratenpartei stehen. Auf einer anderen miradlo-Galerie könnte dieser Tag stattdessen an Bildern eines Fastnachtsumzugs hängen.

Fazit

Es existiert praktisch keine vorgegebene Struktur. Dies kann je nach Benutzer von Vor- oder von Nachteil sein. Tags sollten bei diesem Ansatz semantisch auf allen miradlo-Galerien gleich verwendet werden, um qualitativ hochwertige Suchergebnisse mit geringem Aufwand zu ermöglichen.

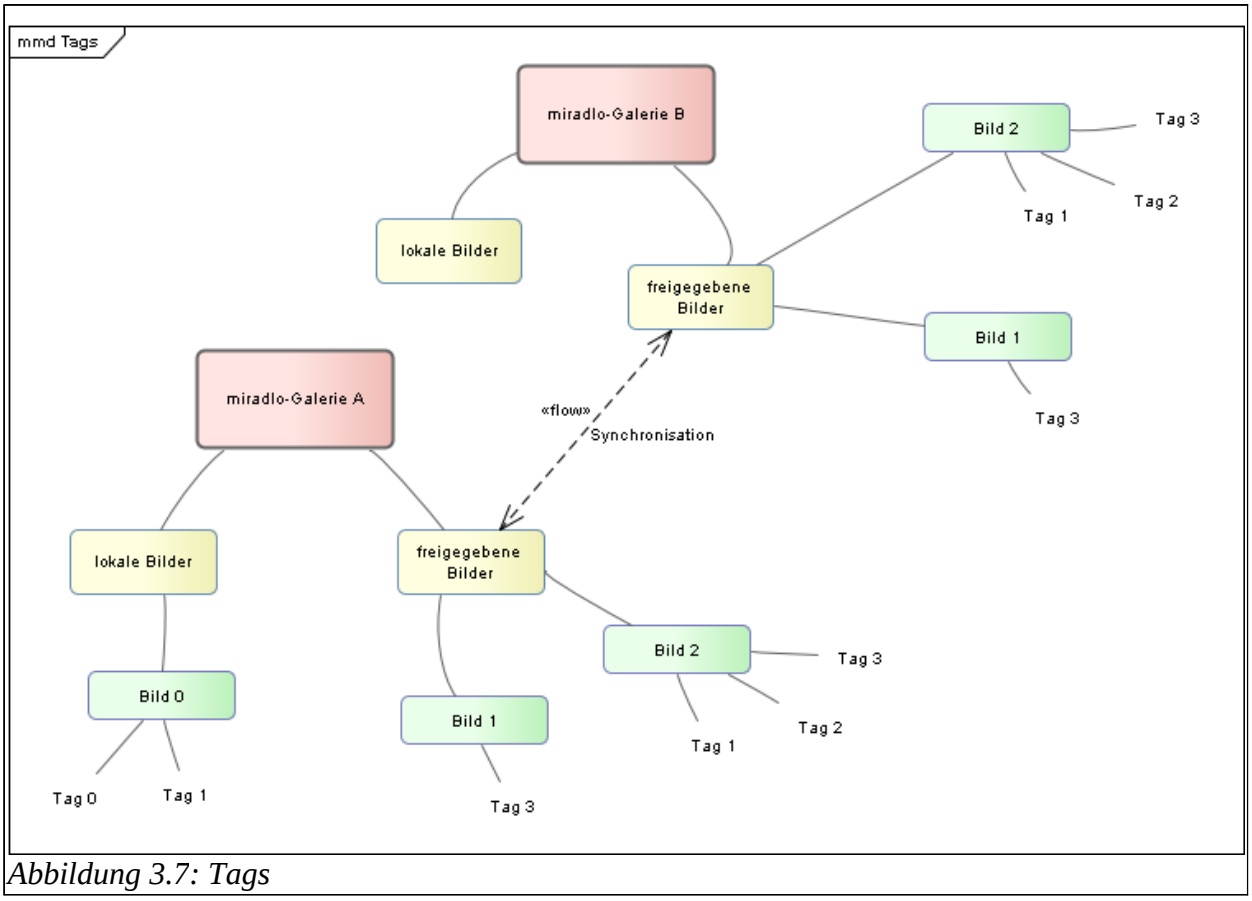

Abbildung 3.7: Tags

Im Diagramm zu den Tags sind die miradlo-Galerien A und B abgebildet. Die miradlo-Galerie A besitzt sowohl lokale als auch freigegebene Bilder. Die miradlo-Galerie B besitzt lediglich die zwei freigegebene Bilder, welche mit der miradlo-Galerie A synchron gehalten werden.

3.3.3 Bewertung der Datenverwaltungsansätze

Die vergleichsweise lose Strukturierung beim Ansatz *Tags* hat zur Folge, dass prinzipiell alle freigegebenen Bilder zwischen Bildergalerien synchronisiert werden. Dieser Punkt lässt sich durch die Beschränkung der Synchronisation auf Bilder mit bestimmten Tags abschwächen. Dabei wird dem Tag-Mechanismus jedoch zu viel Verantwortung übertragen. Durch die Nutzung eines Tagsystems für die inhaltliche Suche und die Konfiguration der Synchronisation wird die Benutzbarkeit stark eingeschränkt. Theoretisch müsste ein Benutzer vor dem Hinzufügen eines Tags zu einem Bild prüfen, ob das Tag im Netzwerk aus miradlo-Galerien eine bereits bestehende Konfiguration beeinflusst.

Durch die stärkere Strukturierung auf Basis von Bildergalerien werden die miradlo-Galerien in die Lage versetzt, ihre Synchronisation auf Basis der Bildergalerien zu koordinieren. Die inhaltliche Suche kann darüberhinaus auf Textelementen, wie Titel und Beschreibung erfolgen.

Durch die Kombination von beiden Konzepten kann eine noch mächtigere Datenverwaltung geschaffen werden. Wenn Bilder des Ansatzes der Bildergalerien noch zusätzlich mit Tags versehen werden, kann die schnelle Durchsuchbarkeit erhöht werden. Durch die Kombination wird eine weitere Möglichkeit der Strukturierung unterhalb der Bildergalerien eingeführt. Dies ermöglicht es, Bildergalerien mit einer verhältnismäßig großen Anzahl an Bildern durchsuchbar zu halten.

Die Realisierung der Datenverwaltung wird in der miradlo-Galerie mithilfe der Kombination aus Bildergalerien und Tags umgesetzt.

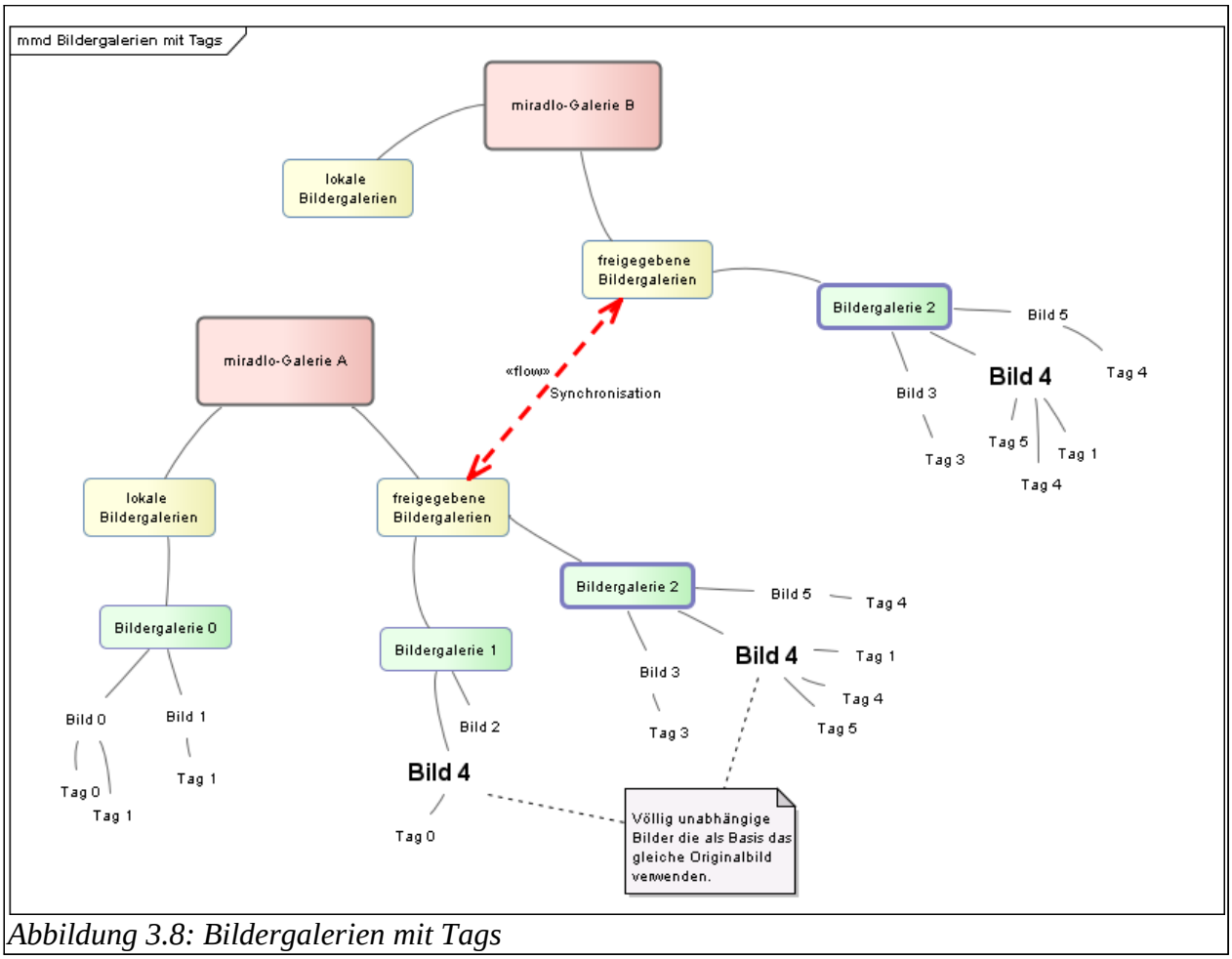

Abbildung 3.8: Bildergalerien mit Tags

In obigem Diagramm sind die *miradlo-Galerien A* und *B* abgebildet. Die *miradlo-Galerie A* besitzt sowohl lokale als auch freigegebene Bildergalerien. Die *miradlo-Galerie B* besitzt lediglich die *Bildergalerie 2*, welche mit der miradlo-Galerie A synchron gehalten wird.

Es ist zu sehen, dass fast alle Bilder mit Tags versehen wurden. Die Tags selbst haben keinerlei Abhängigkeiten untereinander. Bilder müssen obendrein nicht mit Tags versehen sein.

3.4 Datenspeicherung

Im *Kapitel 3.2, Versionsverwaltung, Seite 22*, wurde erarbeitet, dass es einer leistungsfähigen Verwaltung für Originalbilder, Transformationsbeschreibungen und Versionen bedarf.

Transformationsbeschreibungen sind stattdessen Sammlungen von Metadaten und Bearbeitungsschritten, die in textueller Form dargestellt werden können. Ein Originalbild kann viele Transformationsbeschreibungen haben. Jede Transformationsbeschreibung gehört zu einem Originalbild und beschreibt exakt eine Version. Die Abbildung dieser Abhängigkeiten ist äußerst wichtig, um später jederzeit einen konsistenten Datenbestand vorliegen zu haben.

In den folgenden Kapiteln werden verschiedene Varianten für die Realisierung der Datenspeicherung beschrieben und bewertet.

3.4.1 Datenbank

Die Speicherung von potenziell veränderlichen Daten erfolgt in miradlokit grundsätzlich in einer relationalen Datenbank auf einem MySQL-Server [MYSQL]. Lediglich einige Grundeinstellungen und Seitenlayouts werden im Dateisystem gespeichert.

Datenbank-Management-Systeme, wie der MySQL-Server, sind auf die Verwaltung großer Datenmengen spezialisiert. Sie erlauben es, Daten unter Beachtung derer Abhängigkeiten effizient abzulegen und abzurufen.

Die in miradlokit umgesetzten Webapplikationen realisieren ihre Benutzeroberflächen in der Regel mithilfe von HTML-Seiten. Benutzer sind dadurch in der Lage, die Applikationen mit einem Webbrowser ihrer Wahl zu nutzen.

Vorteile

Datenbanken verringern den Wartungsaufwand, da alle Daten von einer eigens dafür konzipierten Instanz verwaltet werden. Programme können mithilfe der Datenbanksprache SQL komfortabel benötigte Daten abrufen und zu speichernde Ergebnisse übergeben. "SQL (Structured Query Language) ist eine Abfrage- und Manipulationssprache für relationale Datenbanken. [...] SQL ist eine deklarative, mengenorientierte Sprache." [14]

Nachteile

Es ist bei HTML-Seiten normal, dass Bilder in verlinkter Form vorliegen und bei Bedarf vom jeweiligen Browser angefordert werden können. Dies bedeutet, dass ein Browser jedes benötigte Bild einzeln anfordert, was zu einer Vielzahl an parallelen Einzelanfragen führt. Ein Link zu einem solchen Bild würde nach dem Muster *http://webauftritt/get.php?id=3* aufgebaut sein. Die auf diese Weise ausgelieferten Bilder werden nicht im Cache des Browsers gespeichert, da dieser nur Dateien des Musters *http://webauftritt/bild.jpg* zwischenspeichern kann. Jede Einzelanfrage führt beim Webserver zu einem Start des PHP-Interpreters [4] und zum Aufbau einer eigenen Datenbankverbindung für diese Anfrage. PHP "ist eine weit verbreitete Open Source Skriptsprache, die sich in der Webprogrammierung [...] etabliert hat". [2] Dieser Overhead erzeugt bei einer Webseite mit beispielsweise 500 Bildern eine erhebliche Serverlast. Bei 20 aktiven Nutzern dieser Webseite würden somit theoretisch 10.000 aktive Interpretersessions erzeugt, was zu einer Überlastung des Webservers führen würde.

Zur Lösung dieser Probleme besteht die Möglichkeit, Bilder in Base64 kodierter Form in die HTML-Seite einzubetten. Das Auflisten von 500 Bildern auf einer Webseite würde so mit insgesamt einer Interpretersession auskommen. Die Umsetzung dieser Lösung ist jedoch nicht praktikabel, da jeder Browser gezwungen würde, alle Bilder einer Seite am Stück zu empfangen. Besonders bei mobilen Endgeräten und textbasierten Browsern ist eine solche Art der Auslieferung inakzeptabel.

Fazit

Datenbanken sind bei der Speicherung von großen Datenmengen das Mittel der Wahl.

Im Fall von miradlo kann die Datenbank nicht zur Speicherung von Bildern verwendet werden, da die Produktionsumgebung nicht dafür ausgelegt ist.

3.4.2 Dateisystem

"*Ein Dateisystem (file system) umfasst die Menge aller von einem Betriebssystem in derselben Art verwalteten Dateien einschließlich aller dafür erforderlichen Verwaltungsinformationen.*" [15]

Dateisysteme bestehen in der Regel aus mehreren Dateiverzeichnissen und den eigentlichen Dateien.

"*Ein Dateiverzeichnis [(Verzeichnis)] (Ordner, Katalog, directory) ist eine Datenstruktur mit Informationen über Dateien zu dem Zweck, diese effizient und geordnet auf einem Datenträger verwalten zu können.*" [16]

"*Eine Datei (file) ist eine Menge von logisch zusammenhängenden Daten, die auf einem geeigneten Medium permanent gespeichert werden kann und über einen Bezeichner identifizierbar ist.*" [17]

Dateien stellen die eigentlichen Datencontainer dar.

Die Quelltexte von miradlokit, Grundeinstellungen und Seitenlayouts liegen grundsätzlich im Dateisystem.

Die Produktionsumgebung von miradlo nutzt zur Auslieferung ihrer Webseiten den Apache HTTP Server (Apache). Die Freie Software "Apache stellt ca. 2/3 aller weltweit eingesetzten Web-Server [...]. [Sie] ist modular aufgebaut, wird ständig verbessert [und] ist hervorragend dokumentiert." [18] Diese Software ist dafür zuständig, Anfragen von Browsern wie z.B. *http://miradlo.de/* zu beantworten. Bei jeder Anfrage prüft der Webserver, ob die Datei, die sich hinter dem angegebenen URL verbirgt, interpretiert werden muss oder direkt ausgeliefert werden kann. Wenn auf miradlokit-Quelltexte verwiesen wird, startet Apache einen PHP-Interpreter und liefert dessen Ergebnis aus.

Vorteile

Es gelingt dem Webserver wesentlich schneller, eine Datei direkt auszuliefern, als sie erst von einen PHP-Interpreter verarbeiten zu lassen.

Nachteile

Die Semantik von Daten und deren Abhängigkeiten untereinander kann in Dateisystemen nur unzureichend abgebildet werden.

Fazit

Besonders für Elemente wie Bilder, die sehr oft durch den Webserver ausgeliefert werden müssen und sich nur selten verändern, bietet es sich auf der Produktionsumgebung von miradlo an, sie direkt im Dateisystem verfügbar zu machen.

3.4.3 Bewertung der Datenspeicherungsansätze

Die Produktionsumgebung von miradlo ist für die Wahl der Technologie maßgeblich.

Durch die vorhergehenden Betrachtungen hat es sich herausgestellt, dass Datenbanken im miradlokit-Umfeld nicht zur Speicherung von binären Objekten mit hohen Einzelzugriffen eingesetzt werden können. Der Aufwand des Starts von PHP-Interpreter und Datenbankverbindung ist wesentlich höher als die direkte Auslieferung einer Datei durch den Webserver.

Für die Umsetzung der miradlo-Galerie wird die Datenbank für die Verwaltung der Transformationsbeschreibungen eingesetzt. Der komfortable Zugriff und die garantierte referentielle Integrität von Informationen zu Bildergalerien, Originalbildern und Versionen bietet hier einen klaren Vorteil. Die Originalbilder und Versionen werden im Dateisystem abgelegt, um dem Webserver eine schnelle Auslieferung zu ermöglichen.

Im folgenden Diagramm wird vereinfacht dargestellt, wie eine Anfrage bei einem Webserver abgearbeitet wird. Es ist zu sehen, dass verschiedene Wege für die Abarbeitung existieren.

Falls eine existierende Datei aus dem Dateisystem des Webservers angefragt wird, kann diese nach dem Öffnen ausgeliefert werden.

Es kann jedoch auch ein zu interpretierendes Skript angefragt werden. In diesem Fall muss der Webserver einen Interpreter starten, welcher das Skript verarbeitet. Falls das Skript Daten aus einer Datenbank benötigt, muss der Interpreter eine Datenbankverbindung herstellen und diverse Datenbankanfragen absetzen. Die Verarbeitung eines Skripts mit Datenbankverbindung und die Auslieferung der Ergebnisse benötigt mehr Zeit als die Auslieferung einer Datei.

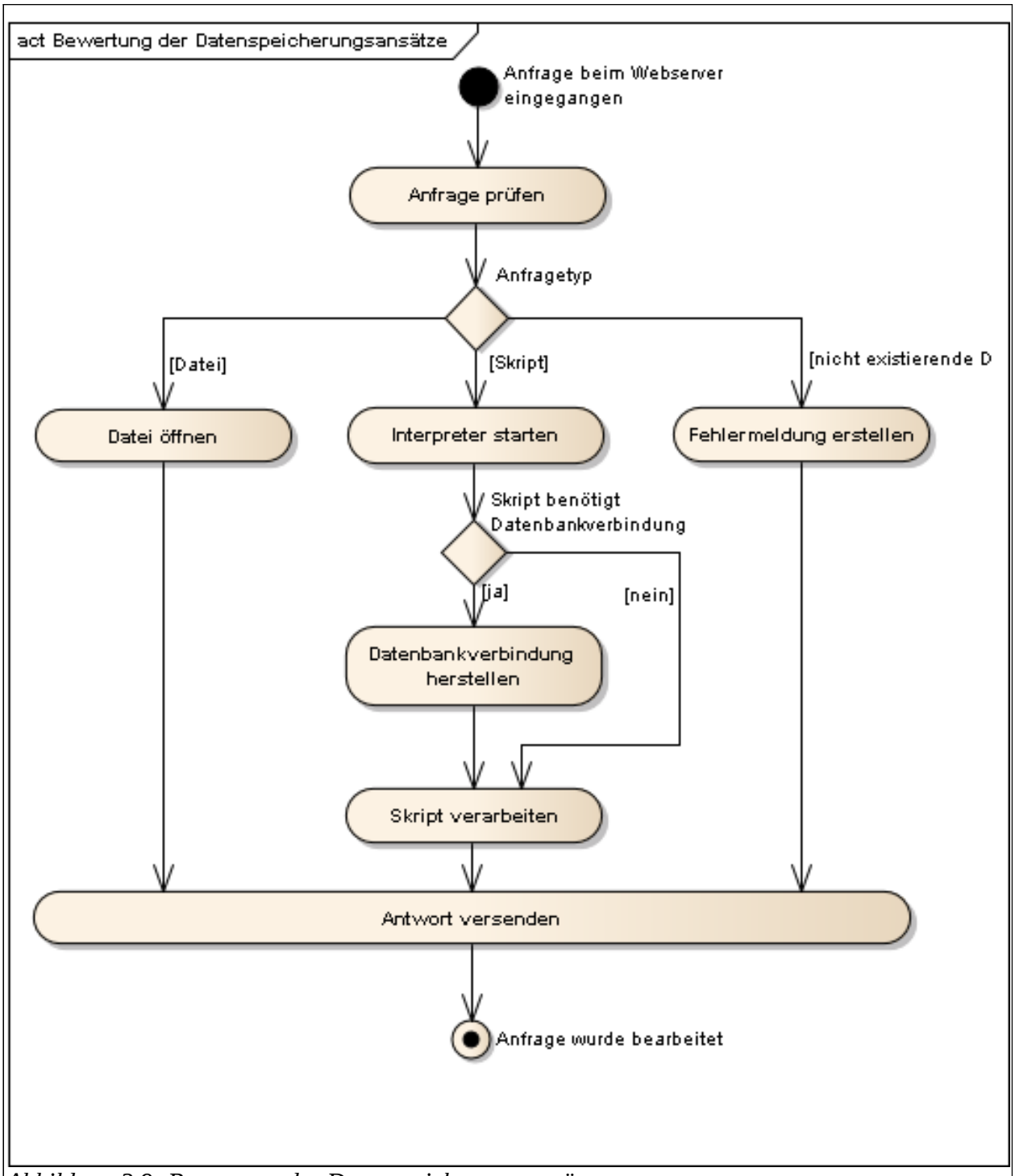

Abbildung 3.9: Bewertung der Datenspeicherungsansätze

Diagrammelement	Beschreibung
Anfrage prüfen	Die Anfrage wird vom Webserver geprüft und es werden Maßnahmen ergriffen, um die angefragten Inhalte zu liefern.
Datei öffnen	Die Datei wird vom Webserver geöffnet.
Interpreter starten	Der Interpreter wird vom Webserver gestartet und es wird begonnen, das Skript zu interpretieren.
Datenbankverbindung herstellen	Die Datenbankverbindung wird vom PHP-Interpreter hergestellt und eine oder mehrere Anfragen abgesetzt. Die Datenbank liefert daraufhin die Ergebnisse an den PHP-Interpreter. Mithilfe dieser Daten kann der PHP-Interpreter die Verarbeitung des Skripts abschließen.
Skript verarbeiten	Dies Verarbeitung des Skripts wird abgeschlossen. Die Ergebnisse des Skripts ergeben die Antwort, welche der Webserver dem anfragenden Computer zusendet.
Fehlermeldung erstellen	Falls eine Ressource angefragt wird, die der Webserver nicht ausliefern kann, wird eine Fehlermeldung erstellt.
Antwort versenden	Die im Vorfeld erstellte Antwort auf die Anfrage wird versendet.
Anfrage beim Webserver eingegangen	Ein Webbrowser setzt eine Anfrage an einen Webserver ab.
Anfrage wurde bearbeitet	Die Anfrage wurde erfolgreich verarbeitet und eine Antwort versendet.
Anfragetyp	Abhängig vom Anfragetyp muss vom Webserver die Antwort auf die Anfrage erstellt werden. Die folgenden Anfragetypen werden hier betrachtet. •Anfrage nach existierender Datei im Dateisystem •Anfrage nach existierendem PHP-Skipt im Dateisystem •Anfrage nach nicht existierender Datei im Dateisystem
Skript benötigt Datenbankverbindung	Abhängig vom Skript, wird eine Datenbankverbindung hergestellt. Die Datenbank liefert über diese Verbindung Daten, die zur Abarbeitung des Skripts benötigt werden.

3.5 Verteilung

In den *Kapiteln 3.2, Versionsverwaltung, Seite 22* und *3.4, Datenspeicherung, Seite 35,* wurde beschrieben, dass Originalbilder mithilfe von Transaktionsbeschreibungen in verschiedene Versionen überführt werden können. Dabei werden die Bilder als solche im Dateisystem und die Transaktionsbeschreibungen in der Datenbank gespeichert.

Die miradlo-Galerie ist von vornherein als Verteiltes System konzipiert. Die Art und Weise der Verteilung ist jedoch offen. Bisher ist lediglich beschrieben, dass Bilder von zumindest einem Teilsystem präsentiert werden können. Ein anderes Teilsystem muss mindestens in der Lage

sein, neue Bilder entgegen zu nehmen und dem präsentierenden Teilsystem zuzuführen. vergleiche Abbildung 1.3: Lokale Daten abgleichen, Seite 5.

In den folgenden Kapiteln werden verschiedene Varianten für die Realisierung der Verteilungsstrategie analysiert und bewertet.

3.5.1 Verteilung in miradlokit

Bevor begonnen werden kann, Lösungen für die miradlo-Galerie zu finden, müssen existierende Strukturen analysiert werden.

Anwendungen, die auf miradlokit basieren, sind von vornherein ein Verteiltes System. Die Serverkomponente bildet dabei miradlokit, welches im Kontext eines Apache HTTP Servers (Webserver) läuft.

"*Ein Server (Diensterbringer) ist eine Komponente, die Aufträge von Clients entgegennimmt, diese Aufträge bearbeitet und eine Antwort [...] an den Client zurücksendet.*" [20]

"*Ein Client (Kunde, Auftragnehmer) ist eine Komponente, die von einem Server eine bestimmte Dienstleistung anfordert [...] und auf eine Antwort wartet.*" [21]

miradlokit obliegt dabei die Datenspeicherung, die Anwendungslogik und der Aufbau der Benutzeroberfläche (GUI). Mögliche Clients verbinden sich mithilfe eines Webbrowsers zum Webserver, welcher wiederum miradlokit aufruft. Der Webbrowser stellt die von miradlokit generierte GUI dar und nimmt Eingaben entgegen. Der Webbrowser ist somit als Thin Client zu bezeichnen, da er neben der GUI auch eigene Zustandsinformationen verwaltet und kleine Programme wie eine automatische Eingabevervollständigung selbst ausführt. "Ein Client, der lediglich eine Benutzeroberfläche und Funktionen zur Kommunikation mit seinem Server enthält, wird als Thin Client [...] bezeichnet." [19]

Alles in allem ist zu sagen, dass bei miradlokit eine klassische Client-Server-Architektur gewählt wurde, bei der viele Clients auf einen Server zugreifen. Die Architektur besteht dabei aus drei Schichten. Die oberste Schicht umfasst den Client, welcher die Präsentation übernimmt. Die mittlere und untere Schicht verwaltet der Server. In der mittleren Schicht befindet sich die Anwendungslogik, welche von miradlokit gestellt wird. In der unteren Schicht befindet sich die Datenspeicherung. Diese wird von einem MySQL-Server bzw. dem Dateisystem übernommen.

Das folgende Diagramm erläutert diesen Punkt näher.

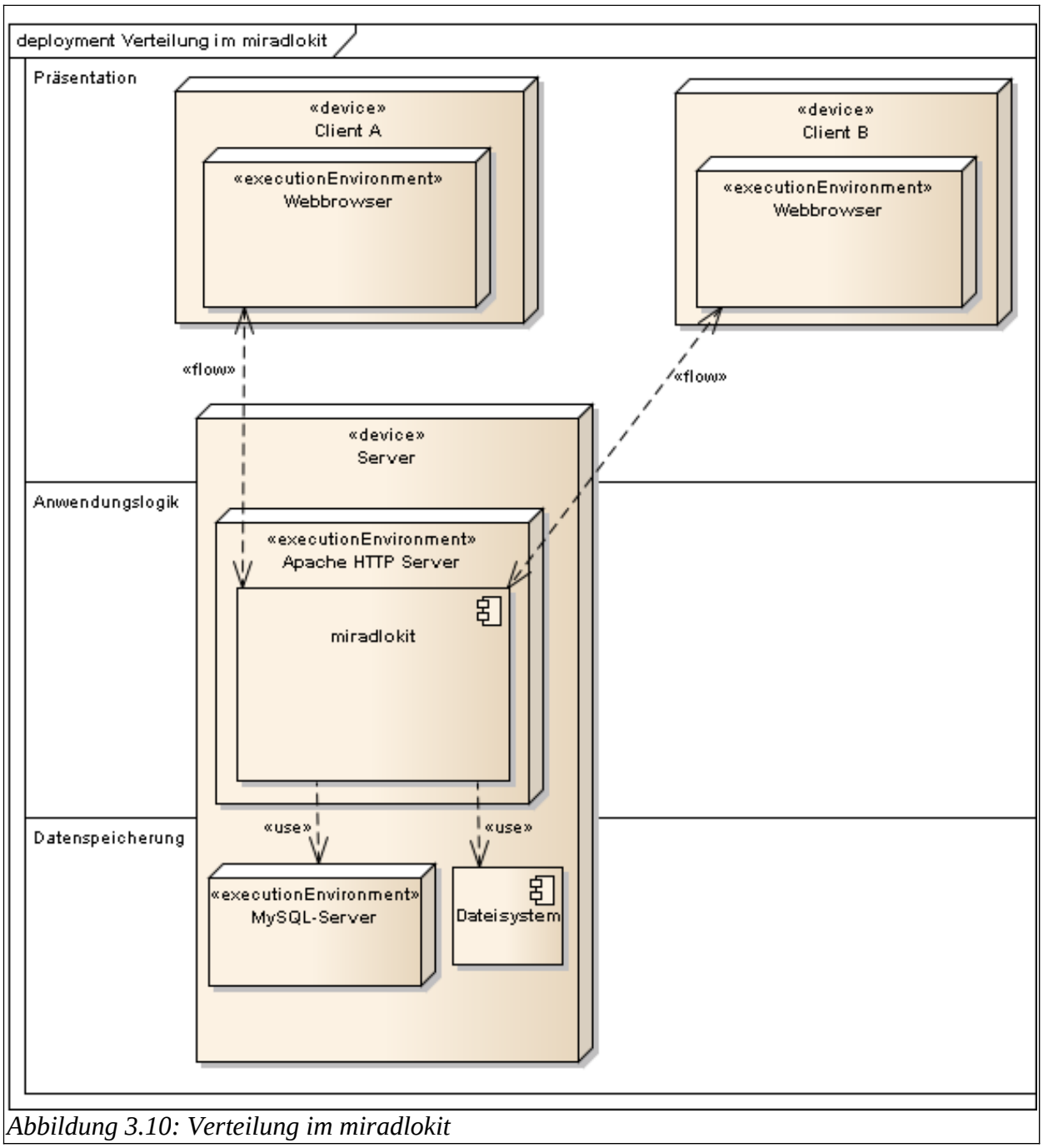

Abbildung 3.10: Verteilung im miradlokit

Ein Benutzer, der beispielsweise den Webbrowser von Client A bedient, ruft eine miradlokit gestützte Anwendung auf. Der Webserver baut hierfür eine Verbindung zum Webserver auf und dieser leitet die Anfrage an die jeweilige miradlokit-Installation weiter. miradlokit benutzt bei der Bearbeitung einen MySQL-Server und das Dateisystem, um alle relevanten Daten zu laden bzw. zu speichern. Sobald miradlokit die Verarbeitung der Anfrage abgeschlossen hat, übergibt es das Ergebnis dem Webserver, welcher dieses an den Webbrowser des Clients ausliefert.

3.5.2 Client-Server

Wenn man die Architektur von miradlokit als Vorbild nimmt, ist es möglich, die gleiche Verteilungsstrategie bei der miradlo-Galerie einzusetzen.

Bei diesem Ansatz gibt es insgesamt zwei Arten von miradlo-Galerien. Zum einen gibt es den miradlo-Galerie-Client, welcher neue Bilder von Benutzern entgegen nimmt und bei Bedarf mit der Serverkomponente abgleicht. Zum anderen gibt es den miradlo-Galerie-Server, welcher die Bilder von miradlo-Galerie-Clients empfängt und präsentiert.

Da der miradlo-Galerie-Server für die Präsentation der Bilder zuständig ist, bietet es sich an, ihn als miradlokit-basierte Anwendung umzusetzen. Die Möglichkeit, mittels Webbrowser auf die Inhalte der miradlo-Galerie zuzugreifen ermöglicht es vielen Clients, gleichzeitig die Inhalte anzusehen.

Vorteile

Die Teilung in Client und Server hat den Vorteil, dass im Prinzip zwei von einander unabhängige Anwendungen entwickelt werden, welche miteinander kommunizieren. Da jede der beiden Anwendungen einen Teil der Gesamtfunktionalität umsetzt, kann die Entwicklung schneller von statten gehen. Lediglich das Kommunikationsprotokoll zwischen Client und Server muss dabei feststehen.

Durch den vergleichbar geringen Funktionsumfang des Clients, kann dieser sehr ressourcenschonend umgesetzt werden.

Die Realisierung von Clients mit einer anderen Softwarearchitektur ist aufgrund der geringeren Komplexität verhältnismäßig einfach möglich. So kann es z.B. einen Client als native Anwendung für ein beliebiges Betriebssystem und einen Client auf Basis von miradlokit geben. Denkbar sind auch Clients für Mobiltelefone und Tablet-Computer [TABL].

Bei diesem zentralisierten Ansatz muss den Clients lediglich die Adresse des Servers bekannt sein, um den Betrieb aufnehmen zu können.

Die Wartung der durch den Server realisierten Funktionen kann zentral erfolgen.

Nachteile

Der miradlo-Galerie-Server ist allein für die Präsentation zuständig und stellt einen Single Point of Failure dar. Sobald die Last auf diesem System zu groß wird, kann kein miradlo-Galerie-Client mehr dessen Dienste nutzen.

Da die Clients nur vergleichsweise geringe Funktionalität bieten müssen, haben sie potenziell viele ungenutzte Ressourcen, welche der miradlo-Galerie nicht zugute kommen.

Fazit

Diese Variante stellt den konventionellen Ansatz der Umsetzung dar. Durch die klare Trennung zwischen Client und Server wird die stattfindende Kommunikation vereinfacht. Auf der anderen Seite ist der einzelne Server eine Schwachstelle im Sinne der Skalierbarkeit. Mittels Replikation

des Servers könnte dieser Effekt vermindert werden, jedoch würden nach wie vor ungenutzte Ressourcen brach liegen. Das folgende Diagramm illustriert den Aufbau der soeben beschriebenen Client-Server-Verteilung.

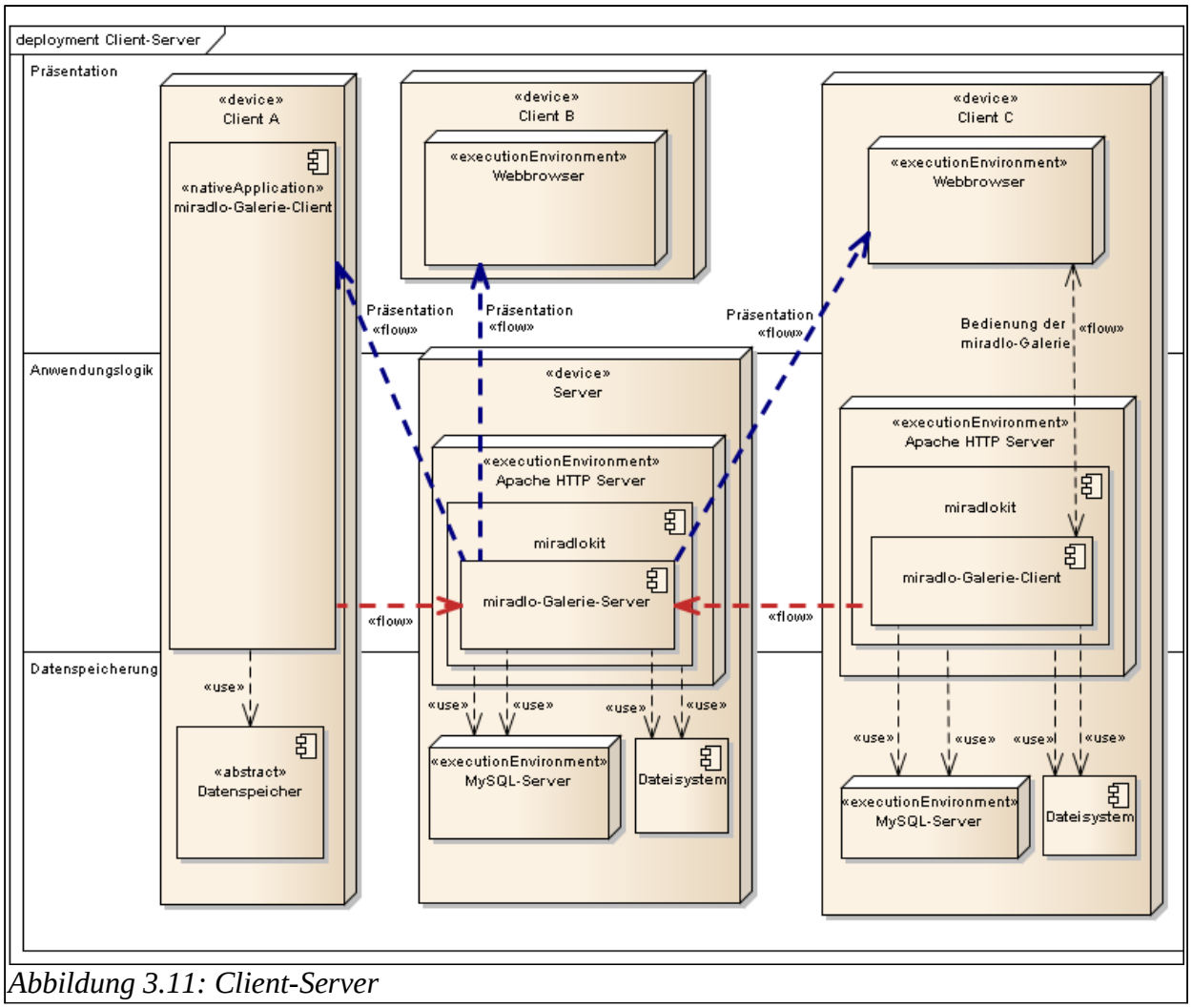

Abbildung 3.11: Client-Server

Die Nutzer an Client A und Client C haben jeweils einen miradlo-Galerie-Client auf dem Computer, welchem sie Bilder übergeben können.

Sobald diese Clients eine Verbindung zum Server aufbauen können, werden diese neuen Bilder auf den Server übertragen. Im Diagramm wird diese Verbindung durch die roten Pfeile dargestellt. Die Pfeile sind gerichtet, da Bilder auf der Ebene der miradlo-Galerie in diesem Ansatz grundsätzlich von den Clients auf den Server übertragen werden.

Mithilfe des Servers werden die Bilder präsentiert. Dies wird durch die blauen Pfeile dargestellt.

Alle Clients greifen für die Präsentation auf den miradlo-Galerie-Server zu.

Es ist zu sehen, dass Client A und Client C eine unterschiedliche Implementierung des miradlo-Galerie-Clients benutzen. Es findet auf der Ebene der miradlo-Galerien jegliche Kommunikation über den zentral angeordneten Server statt, was ihn zum Hauptknotenpunkt der gesamten

miradlo-Galerie macht.

Jede Instanz mit einer miradlo-Galerie besitzt im unteren Drittel des Diagramms eine Datenspeicherungsschicht. Dort werden alle für den Betrieb notwendigen Konfigurationen, Galeriebilder und sonstige Inhalte gespeichert. Der Client A nutzt lediglich einen abstrakten Datenspeicher, weil die Form der Speicherung bei einem nativen miradlo-Galerie-Client nicht der Gegenstand dieser Arbeit ist.

3.5.3 Peer-To-Peer

Eine weitere Variante ist es, jeder miradlo-Galerie im Vergleich zum vorherigen Ansatz einen Client- und Serverteil mitzugeben. Diese Kombination aus Client und Server in einer Anwendung wird als Peer ("gleichberechtigte[r] Partner" [22]) bezeichnet. Ein jeder Peer ist somit in der Lage, neue Bilder von Nutzern entgegen zunehmen und zu präsentieren.

"*Die Mehrzahl verteilter Anwendungen kommuniziert nach dem Client-Server-Prinzip. Im Gegensatz dazu gibt es einige wenige verteilte Anwendungen des Typs Peer-to-Peer[...]. Bei Peer-to-Peer-Anwendungen spielen alle beteiligten Instanzen sowohl die Rolle des Clients als auch des Servers.*" [22]

Folglich kann ein Benutzer neue Bilder seiner lokal installierten miradlo-Galerie hinzufügen, welche bei Bedarf mit anderen miradlo-Galerien abgeglichen werden. Nach einem erfolgten Abgleich sind die Bilder mehrfach innerhalb der vernetzten miradlo-Galerien vorhanden.

Vorteile

Jede miradlo-Galerie stellt ein autonomes System dar, mit welchem Bilder verwaltet und präsentiert werden können. Die durch den Abgleich entstehende Redundanz erhöht die potenzielle Verfügbarkeit der Bilder. Selbst wenn ein im Netzwerk aus miradlo-Galerien wohl bekannter und oft genutzter Peer ausfällt, existieren noch andere, die dessen Daten vorhalten. Jeder Peer ist in der Lage, die ihm von außen zugeführten Bilder selbst zu verwenden. Benutzer sind somit in der Lage, Bilder über mehrere miradlo-Galerien hinweg gemeinsam zu nutzen und zu bearbeiten.

Nachteile

Diese Variante hat den Nachteil, dass die Komplexität eines Peers höher ist, als ein dedizierter miradlo-Galerie-Server und -Client zusammen. Das für einen erfolgreichen Abgleich notwendige Kommunikationsprotokoll ist komplexer und benötigt dadurch mehr Entwicklungszeit. Da jeder Peer in der Regel eine andere Leistungsfähigkeit hat, müssen Wege gefunden werden, jeden Peer nur begrenzt zu belasten. Ein Peer mit beispielsweise nur einem Gigabyte freiem Speicher kann somit potenziell nur einen kleinen Teil an Daten von anderen Peers übernehmen. In einem Netzwerk aus miradlo-Galerien mit n Peers, hat jeder einzelne Peer n - 1 potenzielle Kommunikationspartner. Ein dem Netzwerk aus miradlo-Galerien frisch beitretender Peer besitzt somit das Problem des initialen Kontakts und es muss definiert werden, wie ein Peer andere Peers "findet".

Fazit

Bei dem Peer-To-Peer-Ansatz ist gewährleistet, dass die Last vieler Benutzer auf mehrere Peers aufgeteilt werden kann. Das Problem der aktiven Lastverteilung wird bei diesem Ansatz noch nicht gelöst. Wenn beispielsweise ein Peer zu viele Anfragen erhält stellt dieser aufgrund einer Überlastung seinen Betrieb ein. Die Benutzer sind jedoch in der Lage, durch manuelles Wechseln auf einen anderen Peer, ihre Arbeit dort fortzusetzen. Wenn sich alle Benutzer so verhalten, ist damit zu rechnen, dass alle bekannten Peers größtenteils gleichmäßig ausgelastet werden. Das folgende Diagramm illustriert den Aufbau der soeben beschriebenen Peer-To-Peer-Verteilung.

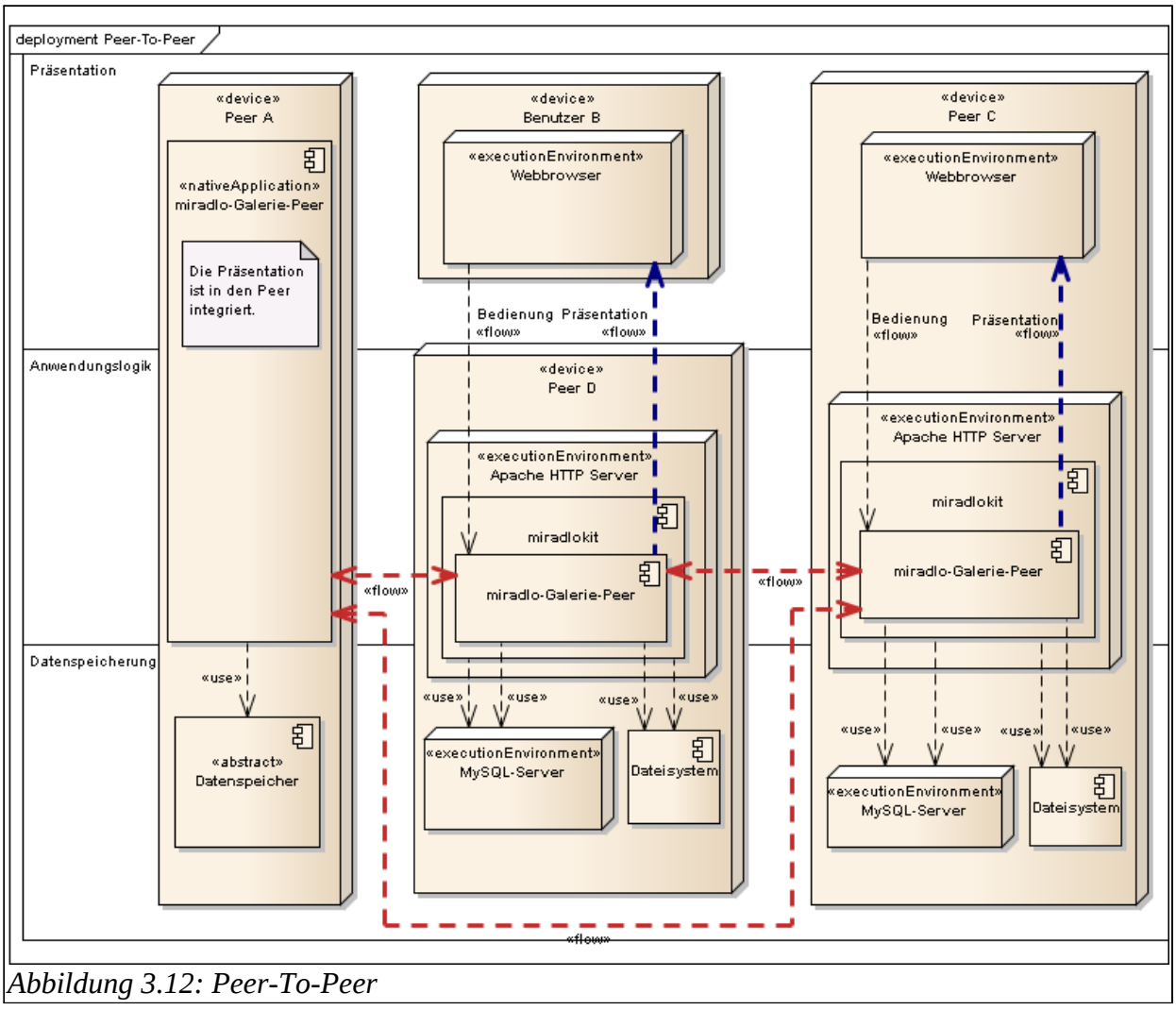

Abbildung 3.12: Peer-To-Peer

Der Peer A, Peer C und Peer D haben jeweils die miradlo-Galerie installiert, welche es ermöglicht Bilder gemeinsam zu verwalten. Die Peers gleichen ihre Datenstände untereinander ab. Der Umstand der gemeinsamen Verwaltung versetzt prinzipiell einen Benutzer in die Lage, seine Bilder auf jedem Peer verwalten zu können.

Es ist zu sehen, dass Peer A und Peer C eine unterschiedliche Implementierung des miradlo-Galerie-Peers benutzen. Die Kommunikation findet zwischen den Peers untereinander statt. Dies wird im Diagramm durch die roten Pfeile dargestellt. Da der Transfer von Bildern zwischen

miradlo-Galerien in diesem Ansatz in beide Richtungen geht, sind die Pfeile bidirektional.

Ein miradlo-Galerie-Peer, der auf Basis von miradlokit realisiert wurde, ist potenziell ein Mehrbenutzersystem, wo viele registrierte Benutzer und auch Besucher die Inhalte der miradlo-Galerie nutzen können. Ein miradlo-Galerie-Peer, der auf Basis einer nativen Anwendung realisiert wurde, ist ebenfalls potenziell mehrbenutzerfähig. Dies bedeutet, dass die Anwendung so realisiert werden kann, dass mehrere von einander unabhängige Benutzer die Anwendung abwechselnd benutzen können. Besucher, die von außen die Inhalte dieses miradlo-Galerie-Peers betrachten wollen, werden in dieser Ausprägung jedoch nicht unterstützt. Die native Anwendung ist primär für dem Einsatzbereich auf Heimcomputern ausgelegt, wo eine permanente Zugriffsmöglichkeit von außen nicht erwünscht ist.

Falls der miradlo-Galerie-Peer auf Basis von miradlokit realisiert wurde, kann ein Benutzer mithilfe eines Webbrowsers das System nutzen. In diesem Diagramm werden beide Fälle illustriert.

- Webbrowser und miradlo-Galerie-Peer befinden sich auf unterschiedlichen Rechnern. (Benutzer B und Peer D)
- Webbrowser und miradlo-Galerie-Peer befinden sich auf demselben Rechner. (Peer C)

Wie im vorhergehenden Diagramm *Client-Server* wird die Präsentation durch die blauen Pfeile illustriert. Peer A ist ein miradlo-Galerie-Peer, welcher auf einer nativen Anwendung basiert. Hier ist die Präsentation in die Anwendung integriert.

3.5.4 Bewertung der Verteilungsansätze

Anhand der vorhergehenden Betrachtungen wurde erarbeitet, dass die Client-Server-Infrastruktur aus Gründen der Skalierbarkeit nicht zur Kommunikation zwischen miradlo-Galerien eingesetzt werden kann. Eine Replikation des Servers würde dieses Problem zwar eingrenzen, aber nicht vollends lösen. Alle Replikate müssten permanent miteinander synchronisiert werden, um die Konsistenz der Daten zu gewährleisten.

Der Peer-To-Peer-Ansatz ermöglicht es hingegen, die Daten zwischen miradlo-Galerien direkt auszutauschen. Dabei kann für jeden Peer individuell festgelegt werden, was synchronisiert werden soll und was nicht.

Die Synchronisation zwischen miradlo-Galerien wird auf Basis des Peer-To-Peer-Ansatzes realisiert. Die Schwierigkeiten, die bei der Unabhängigkeit von einer zentralen Instanz gegeben sind, werden in den folgenden Kapiteln analysiert.

3.6 Adress- und Inhaltsverwaltung

Der Peer-To-Peer-Ansatz geht davon aus, dass alle Instanzen der miradlo-Galerien miteinander kommunizieren. Das Problem dabei ist jedoch, dass den einzelnen miradlo-Galerien bekannt sein muss, unter welchen Adressen sie die anderen miradlo-Galerien erreichen können.

Bei der Diskussion dieser Problemstellung in der Firma, ist die Anforderung *REQ1009 - Löschen von Bildern, Seite 103*, entstanden, welche umfasst, dass Bilder rückstandslos aus einem Verbund von miradlo-Galerien gelöscht und bei Bedarf wieder zugelassen werden können. So soll gewährleistet werden, dass rechtliche Verstöße von Benutzern nicht die komplette Abschaltung aller Teilsysteme nach sich ziehen.

Um dies zu realisieren muss eine Löschliste eingeführt werden. Jede miradlo-Galerie in einem Netzwerk aus miradlo-Galerien, muss während der Verarbeitung von Bildern mithilfe der Löschliste prüfen, ob verbotene Bilder vorhanden sind. Sollte dies der Fall sein, wird das jeweilige Bild entfernt. Die Verteilung und Verwaltung dieser Löschliste ist somit in den Prozess der Adress- und Inhaltsverwaltung einzubeziehen.

Damit die Löschliste praktikabel ist, muss die Prüfung auf verbotene Bilder schnell von statten gehen. Der triviale Ansatz für die Löschliste wäre, eine 1:1-Kopie eines jeden verbotenen Bildes auf die miradlo-Galerien zu verteilen. Bei einer Prüfung könnte dadurch ein zu prüfendes Bild mit jedem der verbotenen Bilder verglichen werden. Die vollständige Übertragung der verbotenen Bilder belastet das Netzwerk aus miradlo-Galerien stark und verbraucht bei einer hohen Anzahl verbotener Bilder viel Speicherplatz.

Aus diesem Grund wird bei der Realisierung der miradlo-Galerie auf eine Hashfunktion zurückgegriffen. Eine solche Funktion erzeugt aus einem zu verbietenden Bild eine Zahl, die dieses Bild praktisch eindeutig identifiziert. Diese Zahl wird im folgenden auch als Hashwert und Hashsumme bezeichnet. Bei dieser Herangehensweise braucht die Löschliste nur aus Hashsummen verbotener Bilder bestehen. Jede miradlo-Galerie erzeugt bei der Verarbeitung von Bildern einfach die Hashsumme des Bildes mithilfe der gleichen Hashfunktion. Anschließend vergleicht sie die daraus entstandene Hashsumme mit jeder Hashsumme aus der Löschliste. Eine Hashsumme hat unabhängig vom Bild datentechnisch immer die gleiche Länge. Eine Hashsumme hat abhängig von der gewählten Hashfunktion in der Regel eine Größe von wenigen hundert Byte. Wenn man annimmt, dass ein Bild mit zwei Megabyte durch eine Hashfunktion in eine Hashsumme mit einer Größe von 200 Byte überführt wird, verringert man die zu übertragende Datenmenge um das 10.485 fache im Gegensatz zum trivialen Ansatz. Die Hashfunktion erzeugt auch bei einem 200 Megabyte großen Bild lediglich eine Hashsumme von 200 Bytes.

In den folgenden Kapiteln werden verschiedene Varianten für das gegenseitige Finden der miradlo-Galerien und deren Indizierung von Inhalten beschrieben und bewertet. Weiter wird deren Vereinbarkeit mit dem Mechanismus der Löschliste analysiert.

3.6.1 Manuelle Konfiguration

Die manuelle Verwaltung der miteinander verbundenen miradlo-Galerien verlagert die Verantwortung dieser Problematik aus dem technischen Kontext der miradlo-Galerie.

Beim Verbinden von einer bestimmten Anzahl an miradlo-Galerien, wird bei jeder einzeln eine Liste mit entfernten miradlo-Galerien eingespielt. Nach dieser initialen Prozedur, ist jede miradlo-Galerie in der Lage, die entfernten zu kontaktieren und deren freigegebene Inhalte abzufragen.

Vorteil

Bei dieser Variante wird auf ein Kommunikationsprotokoll zum Austausch von Adressen der verfügbaren miradlo-Galerien vollständig verzichtet. Die Verlagerung der Adressverwaltung in den administrativen Kontext ist bei kleinen relativ statischen miradlo-Galerie-Netzwerken sinnvoll.

Nachteil

Das Hinzufügen einer miradlo-Galerie zu einem bestehenden Netzwerk aus miradlo-Galerien ist aufwändig. In allen miradlo-Galerien muss die neue miradlo-Galerie jeweils eingetragen werden. Das kurzfristige Beitreten einer miradlo-Galerie zu einem miradlo-Galerie-Netzwerk ist ein hoher administrativer Aufwand. Die Durchführbarkeit der manuellen Verwaltung begrenzt die Anzahl der miteinander verbundenen miradlo-Galerien.

Die Adressverwaltung kann nur gemeinsam erfolgen. Ein Einsatz über administrative Domänen hinweg, stellt ein Problem dar. Eine ständige Absprache aller Administratoren wäre von Nöten und der Ausfall eines Administrators könnte zu einer Inkonsistenz des Netzwerks aus miradlo-Galerien führen.

Um bei Suchanfragen nach Bildergalerien den Benutzern schnell Ergebnisse liefern zu können, muss eine Indizierung der Inhalte entfernter Galerien stattfinden.

Fazit

Obwohl dieses Verfahren aus technischer Sicht dezentral ist, besteht aus administrativer Sicht eine Zentralisierung.

Die manuelle Konfiguration ist stark von den Administratoren abhängig und nicht sehr flexibel.

Bei dieser Variante muss eine Löschliste auf jeder miradlo-Galerie verfügbar sein. Eine Aktualisierung der Löschliste muss auf jeder betroffenen miradlo-Galerie einzeln durchgeführt werden, was wie bei der Adressliste einen hohen administrativen Aufwand nach sich zieht. Das folgende Diagramm illustriert diesen Aufbau.

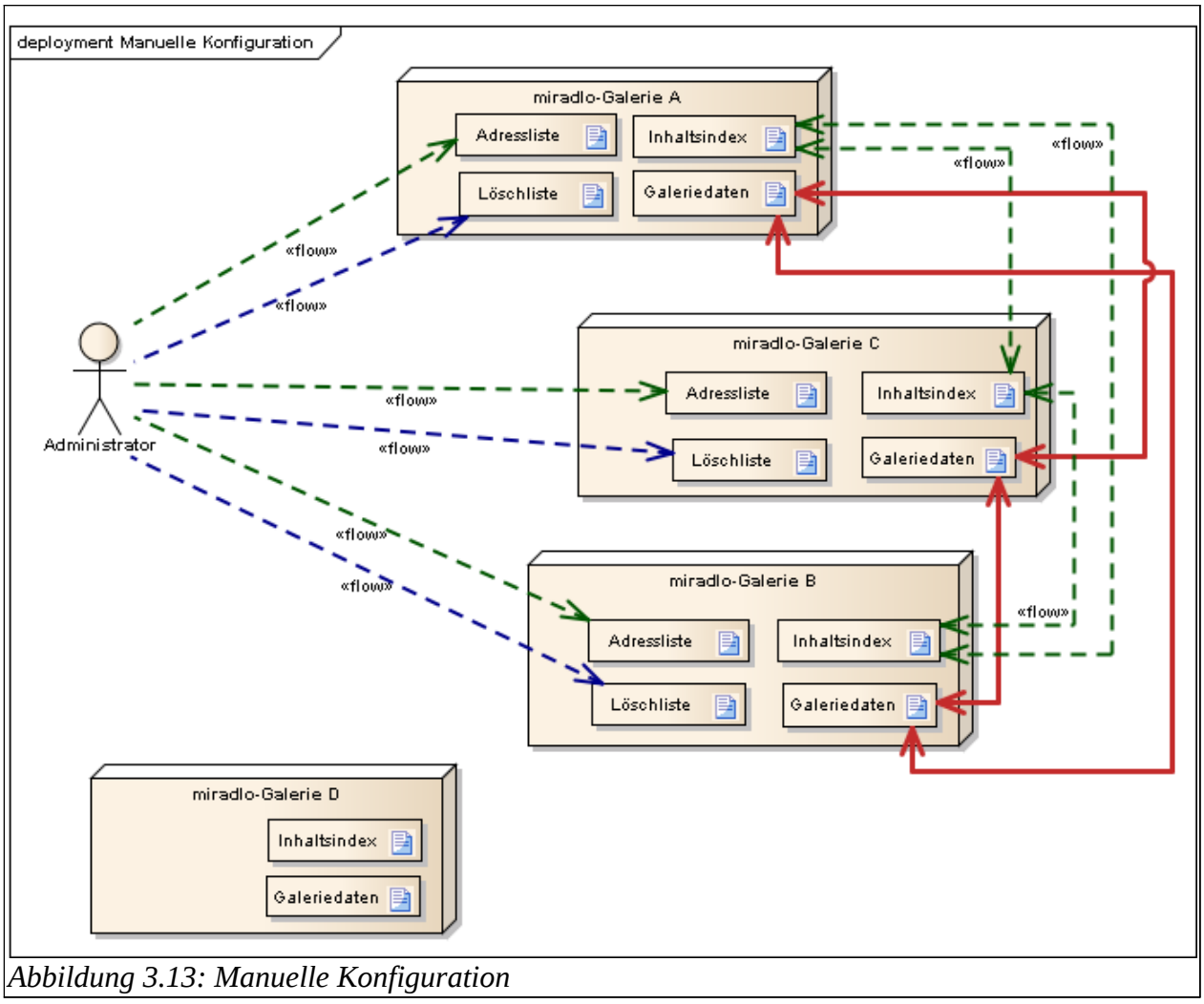

Abbildung 3.13: Manuelle Konfiguration

Es sind drei miradlo-Galerien A,B und C abgebildet, welche miteinander in Verbindung stehen und Galeriedaten austauschen. Dies wird durch die roten Pfeile illustriert. Jede miradlo-Galerie hat durch periodisches Abfragen der anderen Galerien einen Inhaltsindex des gesamten Netzwerks aus miradlo-Galerien. Ein Administrator ist dafür zuständig, alle Adresslisten zu verwalten. Diese beiden Umstände werden durch die grünen Pfeile illustriert. Die Löschliste wird ebenfalls durch den Administrator verwaltet. Dies wird durch die blauen Pfeile illustriert.

Eine hinzukommende miradlo-Galerie D erfordert es, die Konfiguration von miradlo-Galerie A, B und C zu aktualisieren.

3.6.2 Selbstkonfiguration

Bei dieser Variante hält jede miradlo-Galerie all ihr bekannten Adressen der entfernten miradlo-Galerien selbst.

Eine zu einem miradlo-Galerie-Netzwerk hinzukommende miradlo-Galerie übermittelt einer existierenden ihre Adresse.

Die angesprochene miradlo-Galerie antwortet daraufhin mit ihrer gesamten Adressliste. Mithilfe dieser Liste kann die neue miradlo-Galerie jeder entfernten miradlo-Galerie ihre eigene Adresse

mitteilen. Weiter kann über diesen Weg des direkten Ansprechens jede miradlo-Galerie in Erfahrung bringen, welche Inhalte die entfernten miradlo-Galerien bereitstellen.

Vorteile

Dieses Verfahren benötigt keine zentrale Verwaltungsinstanz und erlaubt es miradlo-Galerien mit minimalem initialen Konfigurationsaufwand miteinander zu verbinden. Die miradlo-Galerie-Software kann mit einer Startliste ausgeliefert werden, welche Adressen hochverfügbarer miradlo-Galerien enthält. Dadurch kann eine Konfiguration der Verbindung zu anderen miradlo-Galerien vollständig entfallen.

Nachteile

Es entsteht ein großer Aufwand für die einzelnen miradlo-Galerien, ihre Adresslisten synchron zu halten. Bei einer hohen Anzahl von Galerien, die nur zeitweise erreichbar sind, muss jede miradlo-Galerie individuell entscheiden, ob sie die Verbindung zu einer nicht antwortenden miradlo-Galerie verwirft oder nicht. Jede miradlo-Galerie muss einen Teil ihrer Ressourcen für diesen Verwaltungsprozess abstellen.

Weiter ist die Suche in einem auf diese Weise verknüpften Verteilten System ein Problem. Um Benutzern bei Suchanfragen schnell Ergebnisse liefern zu können, muss theoretisch jede miradlo-Galerie die Inhalte des gesamten Netzwerks aus miradlo-Galerien indiziert vorliegen haben.

Bei einer steigenden Anzahl an miradlo-Galerien im Netzwerk aus miradlo-Galerien, steigt somit auch der individuelle Verwaltungsaufwand für jede miradlo-Galerie. Ab einer bestimmten Anzahl an miradlo-Galerien wird das Netzwerk aus miradlo-Galerien nur noch mit der Selbstorganisation belastet sein und keine Benutzeranfragen mehr entgegennehmen können.

Fazit

Dieses Vorgehen ist technisch und administrativ vollständig dezentral. Die Selbstorganisation verbraucht bei einem größeren Netzwerk aus miradlo-Galerien einen erheblichen Teil der verfügbaren Ressourcen.

In diesem Ansatz müssten die Löschlisten genauso wie die Adresslisten zwischen den Galerien ausgetauscht werden. Dabei ist die Gefahr eines Missbrauchs der Löschfunktion sehr hoch. Ein bösartiger Administrator einer miradlo-Galerie könnte alle verfügbaren Bilder im Netzwerk aus miradlo-Galerien auf die Löschliste setzen. Eine damit verbundene Verteilung der Liste hätte zur Folge, dass jede miradlo-Galerie ihre freigegebenen Inhalte löscht. Das folgende Diagramm illustriert diesen Aufbau.

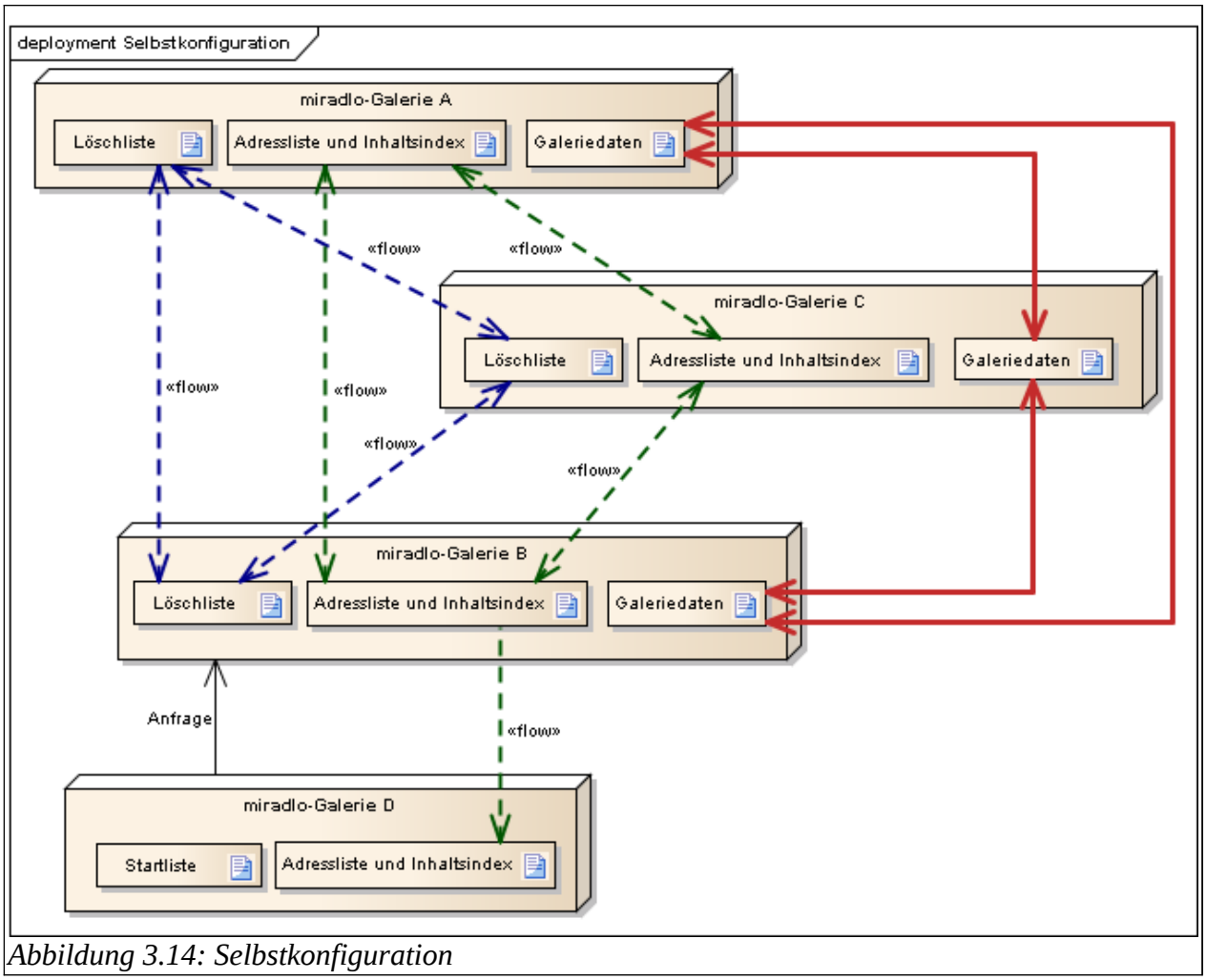

Abbildung 3.14: Selbstkonfiguration

Es sind drei miradlo-Galerien A,B und C abgebildet, welche miteinander in Verbindung stehen und Galeriedaten austauschen. Dies wird durch die roten Pfeile visualisiert. Jede miradlo-Galerie hat sich durch periodisches Abfragen der entfernten miradlo-Galerien eine Adressliste und einen Inhaltsindex des gesamten Netzwerks aus miradlo-Galerien erstellt. Dies wird durch die grünen Pfeile visualisiert. Die Inhalte der Löschlisten werden ebenfalls zwischen den miradlo-Galerien ausgetauscht. Dies wird durch die blauen Pfeile visualisiert.

Eine hinzukommende miradlo-Galerie D fragt mithilfe der in der Startliste eingetragenen Adresse von miradlo-Galerie B deren aktuelle Adressliste ab.

3.6.3 Service-Locator

Mithilfe eines zentralen Service-Locators wird die Verantwortlichkeit der Adress- und Inhaltsverwaltung in einem eigens dafür konzipierten Dienst konzentriert.

Es gibt für ein Netzwerk aus miradlo-Galerien exakt einen Service-Locator, welcher Listen über die Adressen aller miradlo-Galerien und deren freigegebene Bildergalerien führt. Eine miradlo-Galerie, die einem Netzwerk aus miradlo-Galerien beitreten möchte, registriert sich bei dem Service-Locator. Dieser verteilt auf Anfrage die Adressen von miradlo-Galerien, die eine bestimmte freigegebene Bildergalerie halten. Wenn eine miradlo-Galerie nach einer freigegebenen Bildergalerie sucht oder eine Synchronisation durchführen möchte, greift sie auf den Service-Locator zu. Mithilfe dieser Information kann die direkte Kommunikation zwischen den miradlo-Galerien beginnen.

Vorteile

Damit eine miradlo-Galerie einem bestehenden Netzwerk aus miradlo-Galerien beitreten kann, benötigt sie lediglich die Adresse des zum Netzwerk aus miradlo-Galerien zugehörigen Service-Locators. Suchanfragen nach freigegebenen Bildergalerien werden direkt vom Service-Locator bearbeitet.

Da der Service-Locator ausschließlich als Indexdienst eingesetzt wird und somit keine eigentlichen Bilddateien überträgt, kann er auf diese eine Aufgabe spezialisiert umgesetzt werden.

Nachteile

Der Service-Locator stellt einen Single-Point-of-Failure dar. Im Falle eines Ausfalls kann keine Kommunikation zwischen beteiligten miradlo-Galerien stattfinden.

Fazit

Die einzelnen miradlo-Galerien können im Falle eines Ausfalls des Service-Locators autonom weiterarbeiten, da sie dank des Peer-To-Peer-Ansatzes alle Funktionalitäten bereitstellen, Bilder zu verwalten und zu präsentieren. Lediglich eine Suche von entfernten Bildergalerien und der Abgleich mit entfernten miradlo-Galerien ist in diesem Zustand nicht möglich.

Dieser Ansatz ist aus technischer und administrativer Sicht zentralisiert und mit dem Mechanismus der Löschliste kompatibel.

Dabei wird von Service-Locator eine netzwerkweite Löschliste geführt und bei Bedarf an alle miradlo-Galerien verteilt. Jede miradlo-Galerie hält die aktuelle Version der Löschliste, um Bilder, die sich in der Verarbeitung befinden, schnell prüfen zu können. Das folgende Diagramm illustriert diesen Aufbau.

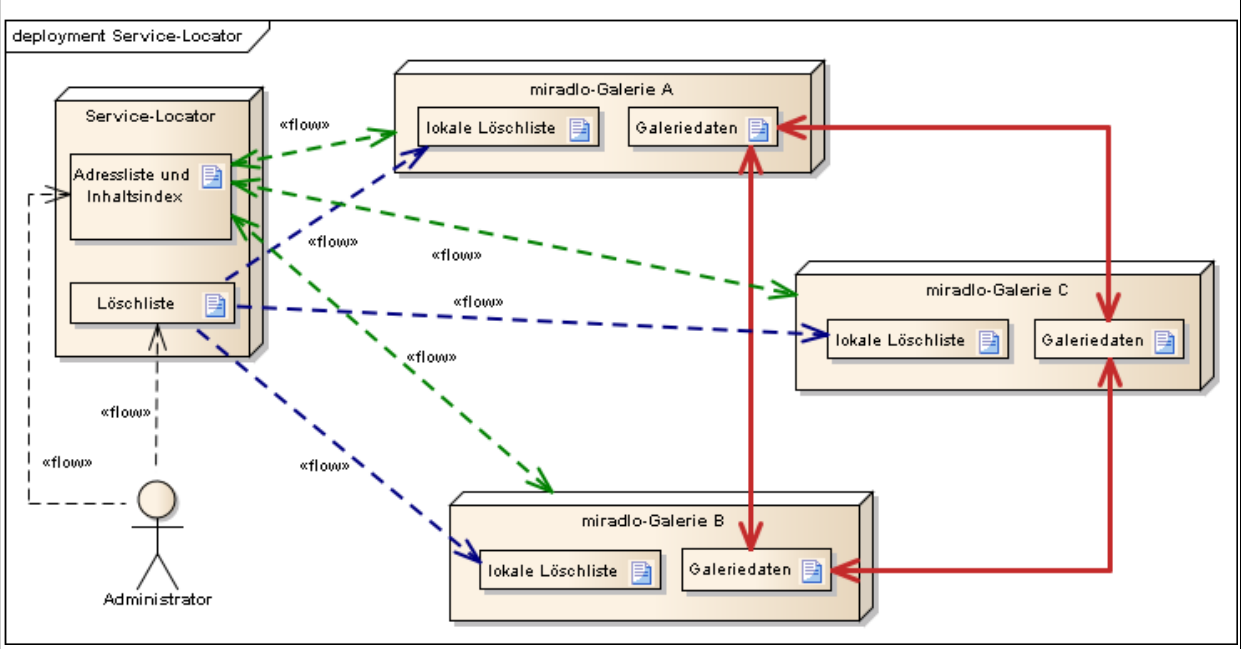

Abbildung 3.15: Service-Locator

Es sind drei miradlo-Galerien A, B und C abgebildet, welche miteinander in Verbindung stehen und Galeriedaten austauschen. Dies wird im Diagramm durch die roten Pfeile dargestellt. Jede miradlo-Galerie erhält auf Anfrage beim Service-Locator die Adressen der entfernten miradlo-Galerien, welche eine bestimmte Bildergalerie führen.

Auch Suchanfragen werden mithilfe des Inhaltsindexes auf dem Service-Locator ausgeführt. Dies wird durch die grünen Pfeile im Diagramm illustriert.

In der Adressliste ist gespeichert, welche miradlo-Galerie, welche Bildergalerie besitzt. Jede dieser Bildergalerien hat eine eindeutige, vom Service-Locator vergebene ID.

Der Inhaltsindex wird vom Administrator des Service-Locators betreut. Durch Modifikation dieses Indexes werden die Suchergebnisse optimiert.

Die Löschliste wird auf dem Service-Locator von dessen Administrator betreut. Der Service-Locator liefert diese Liste auf eine periodisch erfolgende Anfrage hin den miradlo-Galerien aus. Dies wird im Diagramm durch die blauen Pfeile dargestellt.

3.6.4 Bewertung der Adress- und Inhaltsverwaltungsansätze

Die beschriebenen Verfahren unterscheiden sich lediglich darin, wo die Verwaltung stattfindet. Es wurde festgestellt, dass bei der manuellen Konfiguration und der Selbstkonfiguration Probleme im Bereich der Skalierbarkeit existieren. Der Service-Locator stellt einen Single-Point-of-Failure dar.

Es wurde festgestellt dass die Anwendbarkeit der Löschliste bei den Ansätzen *Manuelle Konfiguration* und *Selbstkonfiguration* nicht gegeben ist. Diese Ansätze können somit nicht zur Umsetzung der miradlo-Galerie in Betracht gezogen werden.

Der Ansatz *Service-Locator* ist stattdessen zur Löschliste kompatibel. Sollte der Service-Locator eines umfangreichen Netzwerks aus miradlo-Galerien an seine Belastbarkeitsgrenzen stoßen, besteht die Möglichkeit, ihn mittels Replikation zu skalieren.

Nach den vorhergehenden Betrachtungen ist der Ansatz des Service-Locators der einzige, der die gestellten Anforderungen erfüllt. Die miradlo-Galerie wird mit einem Service-Locator realisiert.

3.7 Datenaustausch

Im Pflichtenheft der miradlo-Galerie ist als Rahmenbedingung definiert, dass eine Synchronisation einen "ressourcensparenden Abgleichsmechanismus" [0] bereitstellen muss. Dieser Punkt wurde in der Anforderung *REQ0018 - Effizienz der Synchronisation bei bereits abgeglichenen Datenbeständen, Seite 111*, festgehalten.

Im folgenden wird als erstes ein Algorithmus entwickelt, der für die Durchführung des Abgleichs zuständig ist. Danach werden Methodiken und Technologien für dessen Umsetzung analysiert und bewertet.

3.7.1 Strategie der Synchronisation

Im *Kapitel 3.6, Adress- und Inhaltsverwaltung, Seite 48*, wurde beschrieben, dass der Service-Locator bei Bedarf von den miradlo-Galerien für die Durchführung von Suchanfragen und die Findung von anderen miradlo-Galerien benutzt werden kann. Der Ablauf einer Synchronisation zwischen mehreren miradlo-Galerien wurde jedoch noch nicht beschrieben.

Eine Synchronisation findet grundsätzlich zwischen zwei miradlo-Galerien statt. Die erneute Ausführung der Synchronisation auf andere miradlo-Galerien gewährleistet einen Datenabgleich im gesamten Netzwerk aus miradlo-Galerien.

Im folgenden wird der Begriff einer relevanten miradlo-Galerie verwendet. Eine miradlo-Galerie wird bei der Synchronisation als relevant bezeichnet, wenn sie eine freigegebene Bildergalerie besitzt. Diese freigegebene Bildergalerie wird dabei von beiden Synchronisationspartnern geführt.

In diesem Kapitel werden zwei verschiedene Synchronisationsstrategien beschrieben und bewertet.

3.7.1.1 Unidirektionale Synchronisation

Bei einer unidirektionalen Synchronisation findet die Aktualisierung der Datenbestände nur bei der miradlo-Galerie statt, von der die Synchronisation ausgeht. Dabei aktualisiert diese miradlo-Galerie ihren eigenen Datenbestand mit den Daten aus allen anderen relevanten miradlo-Galerien.

Die bedeutet, dass die miradlo-Galerie, von der die Synchronisation ausgeht, nur sich selbst aktualisiert. Jede miradlo-Galerie muss somit periodisch Synchronisationen über alle relevanten miradlo-Galerien durchführen, um aktuell zu bleiben.

Vorteil

Der Administrator einer miradlo-Galerie kann selbst entscheiden, wann eine Synchronisation durchgeführt werden soll. Durch den direkten Einfluss des Administrators kann die Synchronisation beispielsweise zu Tageszeiten durchgeführt werden, wo nur geringe Last auf der miradlo-Galerie herrscht.

Nachteil

Bis sich ein neues Bild im gesamten Netzwerk aus miradlo-Galerien ausgebreitet hat, kann je nach Synchronisationsintervall einer jeden miradlo-Galerie einige Zeit vergehen. Die zeitlich gesehen kurze Verfügbarkeit einer miradlo-Galerie im Netzwerk aus miradlo-Galerien, würde nicht zwangsläufig dazu führen, dass ihre Inhalte ins Netzwerk aus miradlo-Galerien gelangen. In der Tat muss bei diesem Ansatz erst eine andere miradlo-Galerie die Inhalte abfragen. Dies ist durch die oben beschriebene Strategie nicht vorhersagbar.

Fazit

Dieses Verfahren erlaubt dem Administrator einer miradlo-Galerie genau zu steuern, wann neue Inhalte aus entfernten miradlo-Galerien bezogen werden. Da eine miradlo-Galerie ihre Inhalte nicht aktiv verteilen kann, besteht die Möglichkeit, dass eine Verbreitung der Inhalte nur langsam oder gar nicht stattfindet.

3.7.1.2 Bidirektionale Synchronisation

Bei der bidirektionalen Synchronisation werden zwischen zwei synchronisierenden miradlo-Galerien solange Daten ausgetauscht, bis beide ihre Datenstände der gemeinsamen freigegebenen Bildergalerien vollständig abgeglichen haben.

Das bedeutet, dass eine miradlo-Galerie eine Synchronisation mit einer anderen initiiert und daraufhin durch bidirektionale Kommunikation die Unterschiede zwischen den Datenbeständen beider miradlo-Galerien gefunden werden. Anschließend übermittelt jede miradlo-Galerie die Inhalte, die der jeweils anderen miradlo-Galerie zur Vollständigkeit fehlen.

Vorteil

Der Administrator einer miradlo-Galerie kann selbst entscheiden, wann Inhalte aktiv im Netzwerk aus miradlo-Galerien verteilt werden sollen.

Nachteil

Ein Administrator einer miradlo-Galerie kann nicht bestimmen, wann neue Inhalte von außen durch andere miradlo-Galerien zugeführt werden.

Fazit

Durch das Initiieren von Synchronisationsvorgängen kann eine miradlo-Galerie ihre Inhalte schnell im gesamten Netzwerk aus miradlo-Galerien verbreiten.

3.7.1.3 Bewertung der Strategiekonzepte

Der unidirektionale Ansatz ist für die Realisierung der miradlo-Galerie nicht praktikabel, da eine nur kurzzeitig verfügbare miradlo-Galerie ihre Inhalte nicht aktiv verteilen kann. Ein Benutzer, der seinen Computer mit der miradlo-Galerie zur Synchronisation mit dem Netzwerk aus miradlo-Galerien verbindet, könnte theoretisch sehr lange warten müssen, bis die Inhalte von einer anderen miradlo-Galerie abgefragt werden.

Die bidirektionale Synchronisation bietet hingegen die Möglichkeit, Inhalte aktiv zu verteilen. Dies ermöglicht es einem Benutzer, seine miradlo-Galerie kurz mit dem Netzwerk zu verbinden, um einen Abgleich durchführen zu können. Es ist möglich die miradlo-Galerie wieder vom Netzwerk zu trennen, wenn sie sich mit mindestens einer anderen miradlo-Galerie synchronisiert hat.

Der Benutzer kann seine miradlo-Galerie noch länger am Netzwerk belassen, um die Synchronisation mit vielen weiteren entfernten miradlo-Galerien durchzuführen und die Verbreitung

seiner Inhalte zu vergrößern. Die bidirektionale Synchronisationsstrategie wird zur Realisierung der miradlo-Galerie eingesetzt.

3.7.2 Methodik der Synchronisation

Der Abgleich zweier miradlo-Galerien besteht rein prinzipiell aus den folgenden Schritten.

- Prüfen, ob sich eine freigegebene Bildergalerie geändert hat.
- Prüfen, ob neue Bilder hinzugekommen sind und diese bei Bedarf austauschen.
- Prüfen, ob bei den vorhandenen Bildern neue Transformationsbeschreibungen vorhanden sind und diese bei Bedarf austauschen.

Im folgenden werden Verfahren entwickelt, analysiert und bewertet, welche diese Problemstellung lösen.

Im folgenden Diagramm werden die in diesem Kapitel verwendeten Begriffe näher erläutert. Eine miradlo-Galerie besteht aus mehreren Bildergalerien. In dieser werden thematisch zusammenhängende Bilder gruppiert. Jedes Bild besteht wiederum aus mindestens einem Originalbild und evtl. Transformationsbeschreibungen.

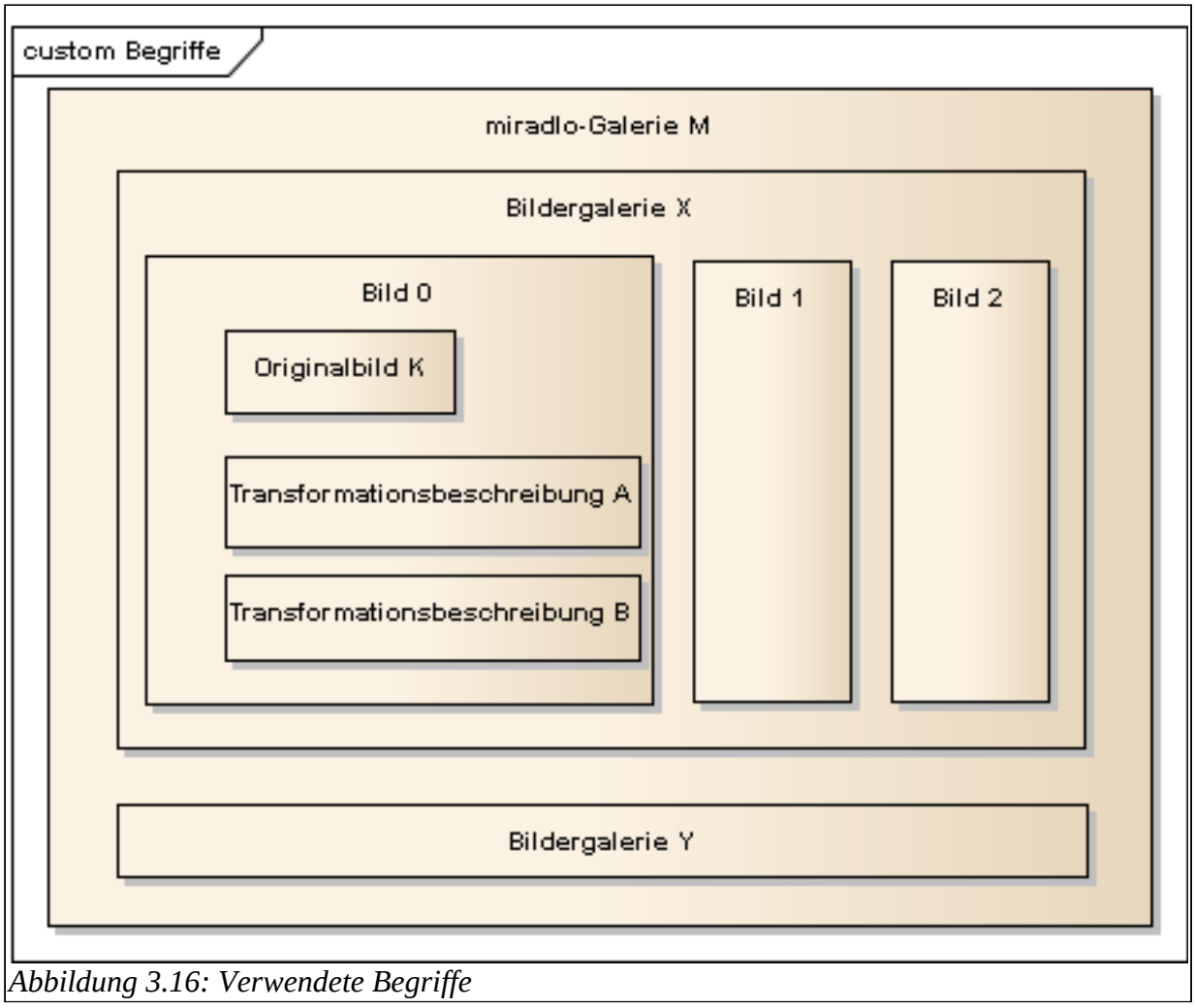

Abbildung 3.16: Verwendete Begriffe

Originalbilder bestehen aus dem Bild, welches ein Benutzer einst hochgeladen hat und diversen

Metadaten. Diese Daten ändern sich nach ihrer Erstellung nicht mehr.

Transformationsbeschreibungen umfassen Informationen, wie das zugehörige Originalbild bearbeitet werden muss, um eine bestimmte Version zu erhalten.

Die einem Bild bzw. einer Bildergalerie untergeordneten Entitäten werden je nach Kontext als Elemente bezeichnet.

3.7.2.1 Zeitstempel

Zu jeder freigegebenen Bildergalerie und jeweils jedem darin enthaltenen Bild wird ein Zeitstempel der letzten Änderung abgespeichert. Ändert sich ein Bild in einer solchen freigegebenen Bildergalerie, so wird der Zeitstempel des Bildes und das der Bildergalerie aktualisiert.

Bei einem Abgleich werden als erstes die Zeitstempel der freigegebenen Bildergalerien verglichen. Falls diese voneinander abweichen, werden als nächstes die Zeitstempel der einzelnen Bilder verglichen. Beim Vorliegen eines neuen Bildes oder eines neueren Standes eines Bildes wird ein Abgleich vorgenommen.

Dieses Verfahren basiert auf der Annahme, dass alle beteiligten Rechner eine vollständig synchrone Systemuhr besitzen. In der Tat kann bei dieser Art Vergleich von Zeitstempeln nie eine verlässliche Aussage getroffen werden, da die Auflösung des Zeitstempels eine fundamentale Rolle spielt. Findet z.B. ein Vergleich auf Sekundenebene statt, können Änderungen, die zufällig in der gleichen Sekunde erfolgt sind, nicht erkannt werden. Im Millisekundenbereich tritt dieses Problem ebenfalls auf, obwohl die Wahrscheinlichkeit eines Fehlverhaltens sinkt. Je höher die Auflösung des Vergleichs gewählt wird, desto wichtiger wird es, exakt synchronisierte Systemuhren zu haben.

Da dieser Umstand selbst bei einer häufigen Synchronisation mit einem zentralen Zeitserver aufgrund von unterschiedlichen Signallaufzeiten nicht mit Sicherheit gewährleistet ist, können Zeitstempel auf der Ebene der Synchronisation nicht eingesetzt werden.

In folgendem Diagramm ist eine miradlo-Galerie abgebildet, welche die beiden freigegebenen Bildergalerien X und Y beinhaltet. Diese besitzen jeweils einen Zeitstempel, welcher beschreibt, wann die letzte Änderung an einem der enthaltenen Bilder erfolgt ist.

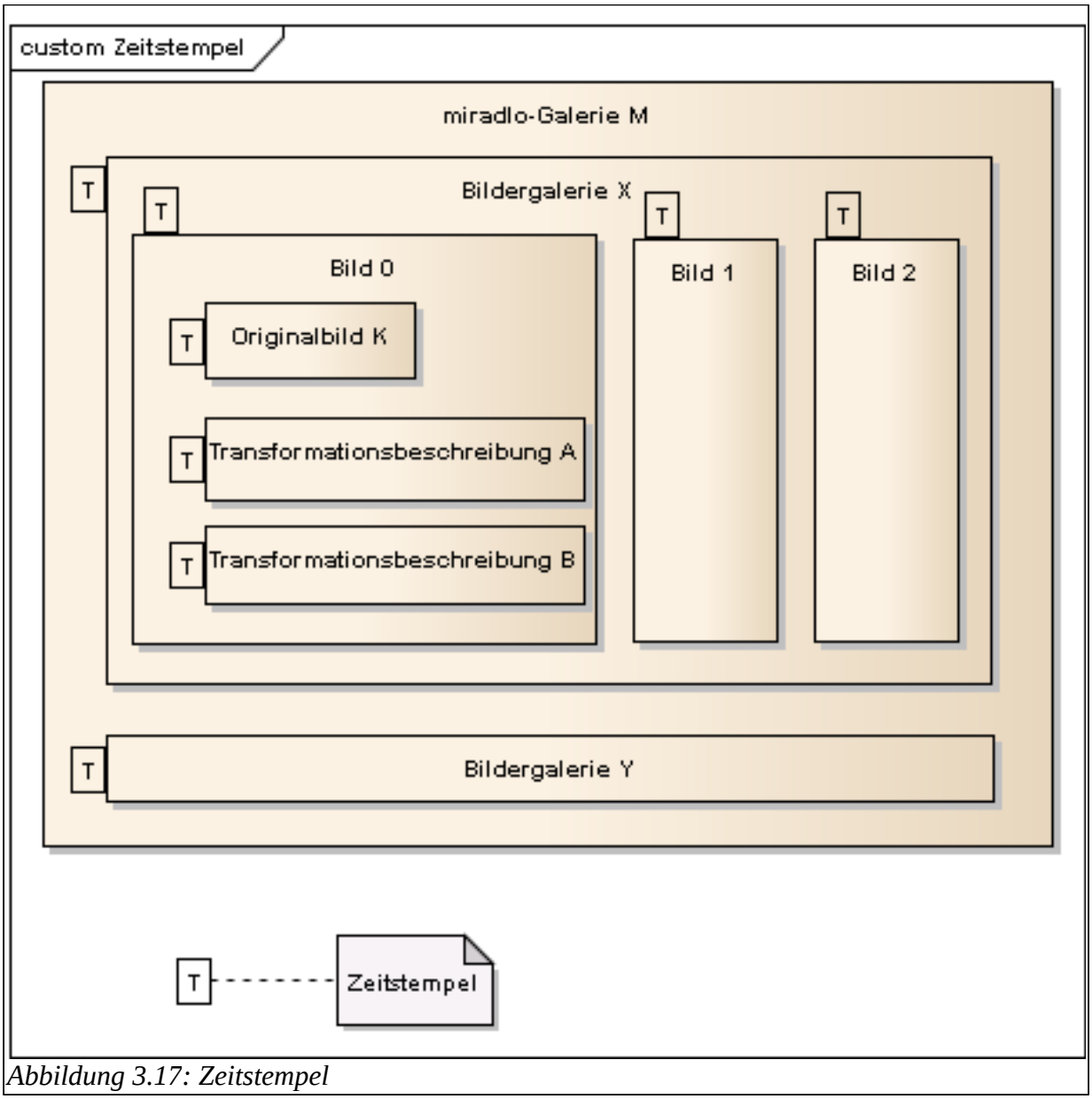

Abbildung 3.17: Zeitstempel

Jedes Bild besitzt wiederum jeweils einen Zeitstempel, welcher die letzte Änderung an den enthaltenen Elementen beschreibt.

Das Originalbild ändert sich nach dessen Erstellung nicht mehr. Es besitzt somit lediglich den Zeitstempel der Erstellung.

3.7.2.2 Hashsummen

Bei der im folgenden beschriebenen Methodik wird vollständig auf die Verwendung von Zeitstempeln verzichtet. Stattdessen wird auf eine Hashfunktion zurückgegriffen, mit welcher der Inhalt einer Bildergalerie, eines Bildes und anderen Elementen beschrieben werden kann.

Über jeder freigegebenen Bildergalerie wird eine Hashsumme berechnet. Weiter wird über jedem Bild und jeder Transformationsbeschreibung eine Hashsumme berechnet.

Bei einem Abgleich werden als erstes die Hashsummen der freigegebenen Bildergalerien verglichen. Bei einer Abweichung werden nun alle Hashsummen der einzelnen Bilder verglichen. Neue und bisher existierende Bilder werden mithilfe der Hashsumme der Originalbilder unterschieden.

Wenn sich ein Bild verändert hat, werden die Hashsummen der Transformationsbeschreibungen verglichen. Erst nachdem die zu übertragenden Daten identifiziert wurden, findet der eigentliche Transfer derer statt.

Die Berechnung von Hashsummen über große Datenmengen benötigt Zeit. Bei dem bisher beschriebenen Modell müsste bei jeder Änderung in einer Bildergalerie eine Neuberechnung der Hashsumme der gesamten Bildergalerie erfolgen. Weiter muss bei einer Änderung eines Bildes ebenfalls eine Neuberechnung dessen Hashwerts erfolgen.

Die Berechnung der Hashsumme eines Bildes lässt sich nicht umgehen, wenn jederzeit ein konsistenter Stand zwischen Hashsummen und Inhalten vorliegen soll. Die Hashsumme einer Bildergalerie braucht jedoch nur bei Bedarf berechnet werden, da sie die hierarchisch gesehen oberste Instanz ist und sonst keine andere Funktion erfüllt, als die ihr untergeordneten Bilder an sich zu binden. Durch Einführung eines DirtyFlags für die Hashsummen der Bildergalerien, kann die Neuberechnung bei unverändertem Inhalt gespart werden. Nur wenn dieses Flag gesetzt ist, findet eine Neuberechnung statt. Das Flag wird immer dann gesetzt, wenn eine Änderung des Inhalts einer Bildergalerie stattfindet.

Es ist bei diesen Überlegungen zu beachten, dass mit steigendem Inhalt einer Bildergalerie die benötigte Zeit für die Berechnung der Hashsummen für Bilder und die Bildergalerie selbst linear steigt.

Um die Berechnungszeit der Hashsummen für Bilder und Bildergalerien klein zu halten, werden nicht sämtliche Inhalte bei der Berechnung herangezogen. Es werden lediglich die Hashsummen der jeweils untergeordneten Ebene verwendet. Dies bedeutet, dass bei der Berechnung der Hashsumme einer Bildergalerie lediglich die Hashsummen aller Bilder beachtet werden.

Bei der Nutzung von Hashfunktionen in diesem Kontext muss beachtet werden, dass durch eine ungünstige Hashsummen-Kollision die Synchronisation fehlschlagen kann. Es ist zwar definiert, dass jedes Originalbild nur einmal zu einer Bildergalerie hinzugefügt werden kann, jedoch besteht eine sehr kleine Wahrscheinlichkeit, dass trotz einer Änderung dieselbe Hashsumme berechnet wird. Dieses Problem kann nicht gelöst werden, jedoch kann es durch die Verwendung von kollisionsresistenten Hashfunktionen stark minimiert werden.

Fakt ist, dass diese Methodik nur funktioniert, wenn alle beteiligten miradlo-Galerien die gleiche Hashfunktion einsetzen. Da Hashfunktionen ständig weiterentwickelt werden, müssen diese vollständig austauschbar sein. Im Falle der miradlo-Galerien müsste jede miradlo-Galerie nach erfolgter Installation der Hashfunktion sämtliche Hashsummen einmalig erneut berechnen. Theoretisch ist es im Falle einer längerfristigen Umstellung auch möglich, für eine gewisse Zeit zwei oder mehr Hashfunktionen gleichzeitig zu unterstützen. Es müssten lediglich zu jedem Element zwei Hashsummen hinterlegt werden. Das Kommunikationsprotokoll zwischen den miradlo-Galerien muss folglich eine so hohe Flexibilität aufweisen, dass eine Aushandlung der verwendeten Hashfunktion stattfinden kann.

Ein weiteres Problem ist, dass es für einzelne Bilder bzw. Bildergalerien eine Rolle spielt, in welcher Reihenfolge die einzelnen Elemente in die Berechnung der Hashsumme einfließen. Dieses Problem lässt sich lösen, indem die einzelnen Elemente in einer fest vorgegebenen Reihenfolge verarbeitet werden. Dies erreicht man, indem die zu verarbeitenden Hashsummen vor der erneuten Verarbeitung sortiert werden.

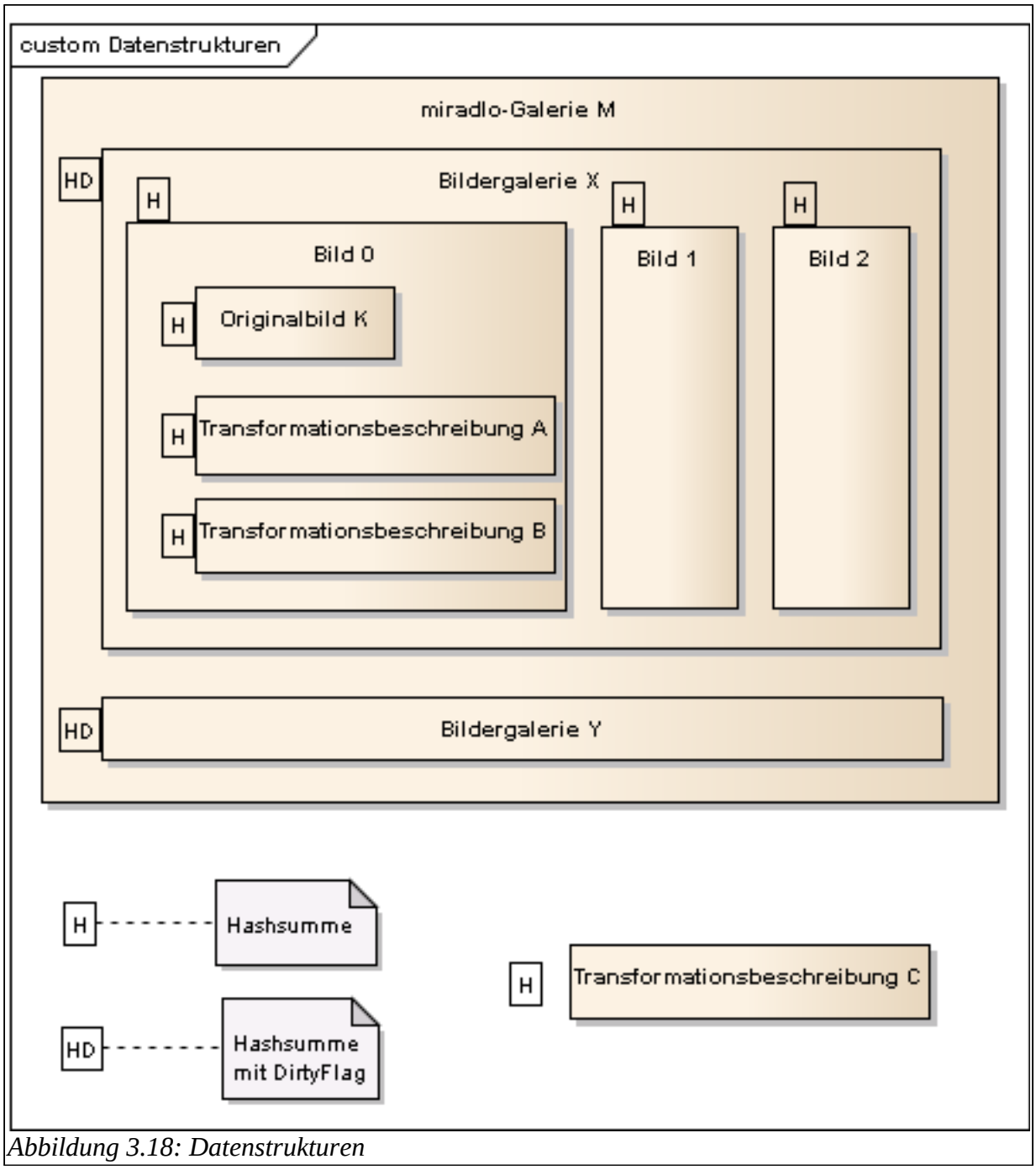

Abbildung 3.18: Datenstrukturen

Im Diagramm zu Datenstrukturen ist die Verteilung der Hashsummen dieses Ansatzes illustriert. Bis auf die Hashsummen der Bildergalerien werden sämtliche Hashsummen stets aktuell gehalten. Dies bedeutet, dass bei jeglichen Änderungen eine Neuberechnung stattfindet. Wenn

beispielsweise zum *Bild 0* der *Bildergalerie X* eine *Transformationsbeschreibung C* hinzugefügt wird, berechnet die *miradlo-Galerie M* umgehend die Hashsumme der neuen Transformationsbeschreibung, aktualisiert die Hashsumme von *Bild 0* und setzt das *DirtyFlag* von *Bildergalerie X.* Die folgenden vier Diagramme illustrieren den Ablauf einer vollständigen Synchronisation einer miradlo-Galerie.

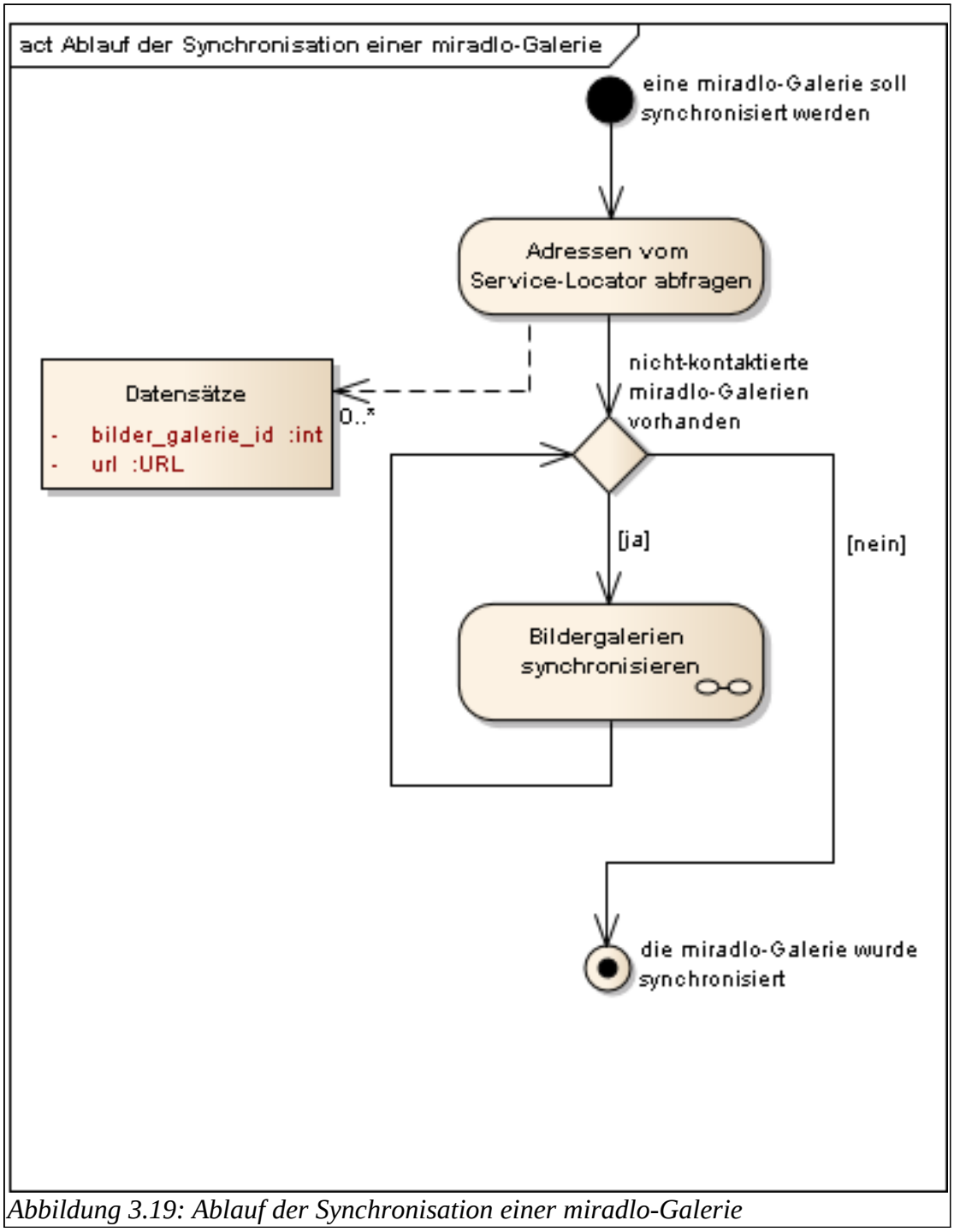

Abbildung 3.19: Ablauf der Synchronisation einer miradlo-Galerie

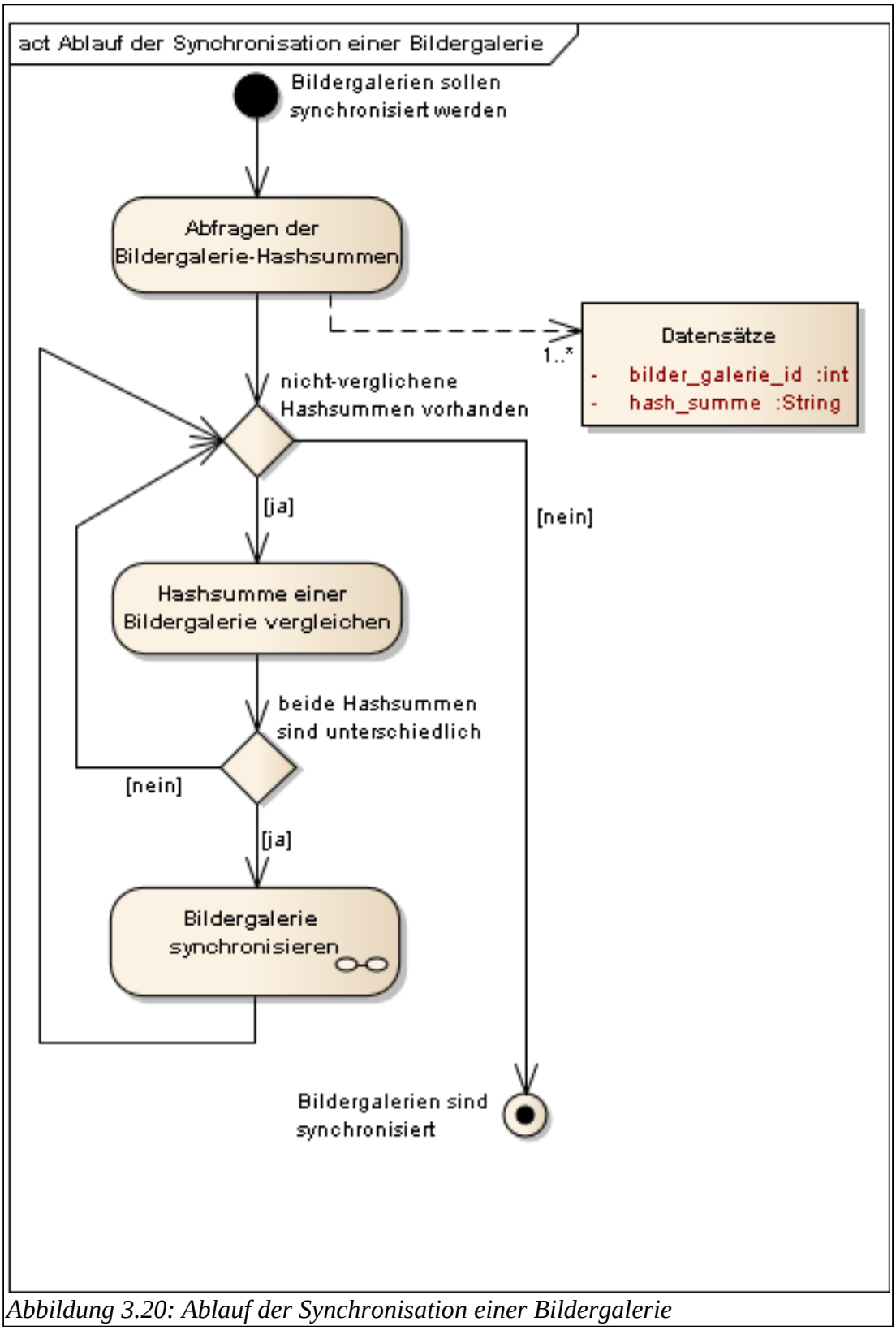

Abbildung 3.20: Ablauf der Synchronisation einer Bildergalerie

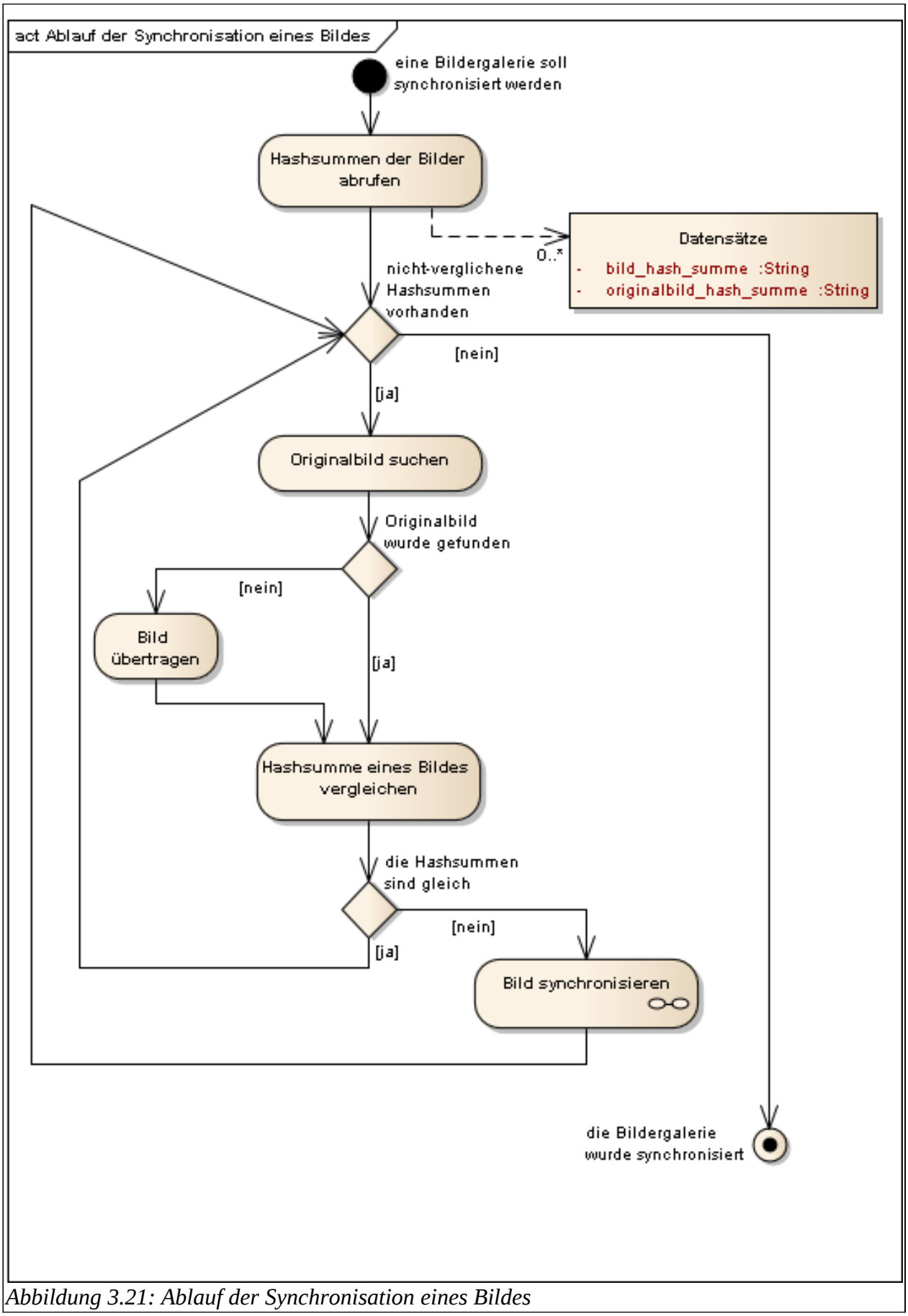

Abbildung 3.21: Ablauf der Synchronisation eines Bildes

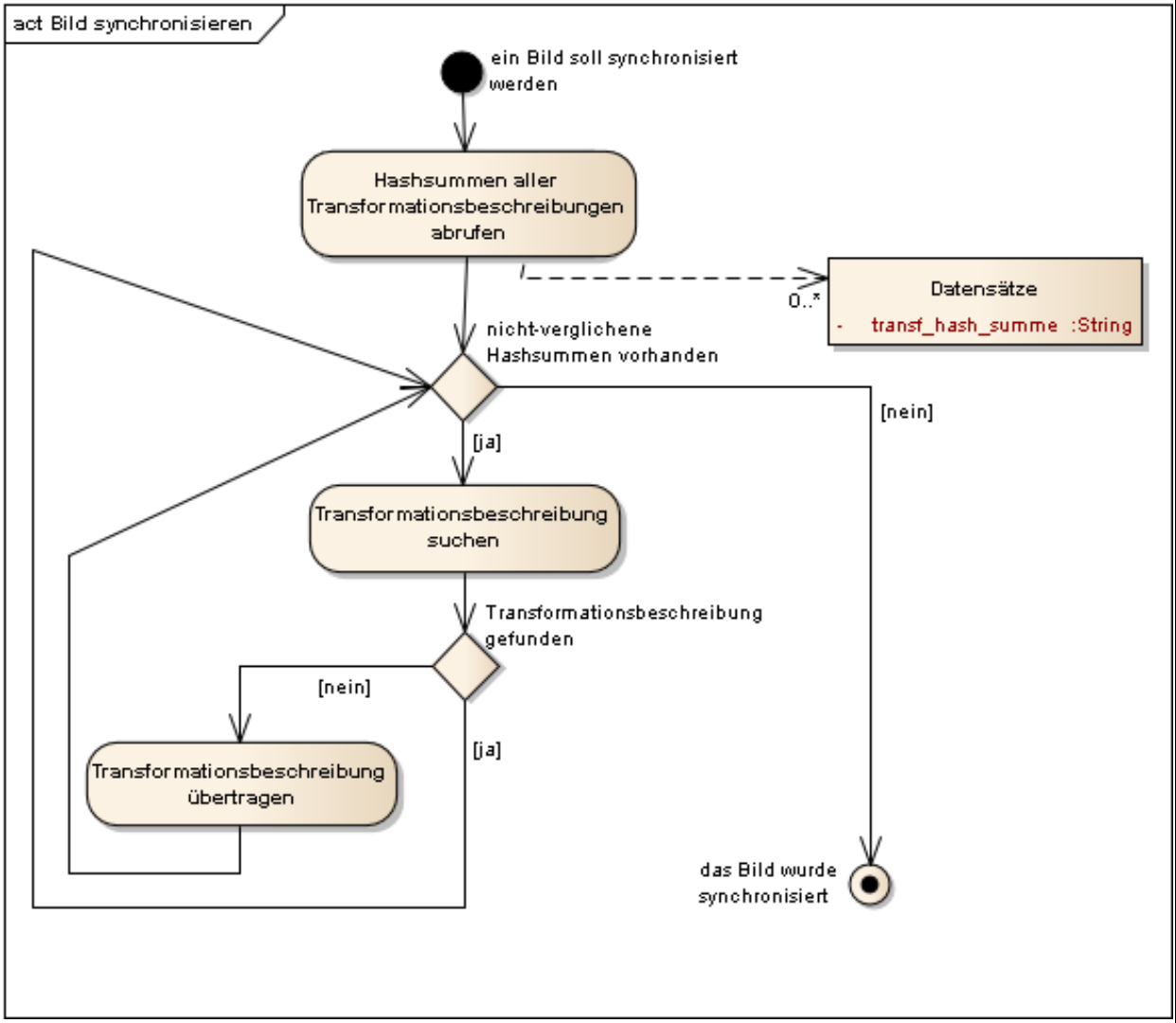

Abbildung 3.22: Bild synchronisieren falls Hashsummen ungleich waren

3.7.2.3 Bewertung der Methodikkonzepte

Aus den Betrachtungen ist hervorgegangen, dass die Nutzung von Zeitstempeln zur Identifizierung von zu übertragenden Inhalten problematisch ist.

Die Nutzung von Hashsummen erfordert zwar zusätzlichen Rechenaufwand für deren Erstellung, jedoch wird keine zentrale Zeitsynchronisierung benötigt. Lediglich die verwendete Hashfunktion muss auf jeder miradlo-Galerie gleich sein.

Bei der Realisierung der miradlo-Galerie wird auf Hashsummen zur Identifizierung von zu übertragenden Inhalten zurückgegriffen.

3.7.3 Initiierung der Synchronisation

In den *Kapiteln 3.7.1, Strategie der Synchronisation, Seite 56* und *3.7.2, Methodik der Synchronisation, Seite 58,* wurde erarbeitet, dass eine bidirektionale und hashsummengestützte Synchronisation zur Realisierung der miradlo-Galerie eingesetzt wird.

In diesem Kapitel wird analysiert, in welchen Situationen eine Synchronisation gestartet wird.

3.7.3.1 Zeitbasierte Auslösung

Bei einer zeitbasierten Auslösung wird nach einem definierten Zeitintervall eine Synchronisation angestoßen.

Bei diesem Ansatz kann der Administrator einer jeden miradlo-Galerie das Zeitintervall einstellen, nachdem eine Synchronisation angestoßen wird.

Vorteil

Es ist sichergestellt, dass nach dem Ablauf der eingestellten Zeit eine Synchronisation durchgeführt wird. Bei dieser Variante ist die Wahrscheinlichkeit gering, dass eine miradlo-Galerie veraltete freigegebene Bildergalerien vorhält. Dies ist nur der Fall, wenn entfernte relevante miradlo-Galerien stets zum Zeitpunkt der Synchronisation verfügbar sind.

Nachteil

Es ist möglich, dass miradlo-Galerien ausschließlich freigegebene Bildergalerien vorhalten, die nur selten oder nicht mehr mit neuen Inhalten versorgt werden. Eine periodische Synchronisation würde bei diesen Systemen langfristig gesehen eine Menge Prozessorzeit [CPUT] verbrauchen. Weiter würde der Service-Locator und somit alle anderen miradlo-Galerien durch diese Vorgehensweise unnötig belastet werden.

Fazit

Diese Variante stellt sicher, dass freigegebene Bildergalerien in einem festgelegten Zeitintervall synchronisiert werden. Bei seltenen Änderungen an den freigegebenen Bildergalerien wird langfristig eine erhebliche Last erzeugt.

3.7.3.2 Ereignisbasierte Auslösung

Beim ereignisbasierten Ansatz wird eine Synchronisation nur beim Eintreten bestimmter Zustände ausgelöst.

Solche Zustandsübergänge können beispielsweise wie folgt sein.

- Wiederherstellung der Netzwerkverbindung nach einer längeren Zeit ohne Netzwerkzugriff
- Verfügbarkeit eines neuen Bildes oder einer neuen Version

Vorteil

Der Administrator einer miradlo-Galerie kann festlegen, ob beim Eintreten eines bestimmten Ereignisses, eine Synchronisation initiiert werden soll. Beispielsweise kann eine miradlo-Galerie mit potenziell stabiler Netzwerkverbindung bei der Verfügbarkeit eines neuen Bildes eine Synchronisation beginnen. Auf eine Synchronisation bei einer Wiederherstellung der Netzwerkverbindung kann in diesem Fall verzichtet werden. Anders ist es bei einer miradlo-Galerie, die nur selten eine Netzwerkverbindung besitzt. In diesem Fall würde es ausreichen, wenn eine Synchronisation bei einer erfolgreichen Wiederherstellung der Netzwerkverbindung stattfindet.

Nachteil

Wenn zu viele mögliche Ereignistypen bei der Konfiguration zur Verfügung stehen, kann es passieren, dass die Einstellungen nicht mehr überschaubar sind.

Fazit

Dieser Ansatz erlaubt es, zeitunabhängig eine Synchronisation zu initiieren. Dabei wird potenziell erheblich Prozessorzeit [CPUT] gespart, da unnötige Synchronisationsvorgänge vermieden werden können. Die Konfiguration der Ereignisse muss gut dokumentiert sein, damit Fehleingaben eines Administrators minimiert werden.

3.7.3.3 Bewertung der Initiierungskonzepte

Alles in allem ist zu sagen, dass beide Varianten ihre Daseinsberechtigung haben.

Bei miradlo-Galerien mit häufig aktualisierten Bildergalerien kann auf eine zeitbasierte Synchronisation zurückgegriffen werden. Besonders, wenn die Synchronisation nicht der Hauptanwendungsbereich der miradlo-Galerie ist, kann durch dieses Vorgehen eine Menge Prozessorzeit [CPUT] gespart werden. So könnte sich beispielsweise eine miradlo-Galerie jeden morgen um 03:00 Uhr mit den relevanten miradlo-Galerien synchronisieren. Diese Vorgehensweise hätte den Charakter einer täglich durchgeführten Datensicherung [DATEN].

Für miradlo-Galerien, die häufige Änderungen erfahren und für die die Synchronisation ein elementarer Teil des Geschäftsmodells ist, bietet sich eine ereignisbasierte Synchronisation an. Besonders wenn Inhalte möglichst ohne zeitlichen Verzug über viele miradlo-Galerien verbreitet werden sollen, erzielt man so eine rasche Verteilung.

Das Problem bei beiden Ansätzen ist, dass eine Überlastung auftreten kann, wenn mehrere Synchronisationsvorgänge gleichzeitig ablaufen. Beispielsweise kann sich die Synchronisation wie eine Lawine im Netzwerk aus miradlo-Galerien verhalten. Jede miradlo-Galerie hat das Ziel, sich mit jeder relevanten miradlo-Galerie zu synchronisieren. Dieser Prozess fängt bei einer miradlo-Galerie an, welcher ein neues Bild hinzugefügt wurde. Durch die Ereignissteuerung beginnt sie, sich mit einer ersten relevanten miradlo-Galerie zu synchronisieren. Nach diesem Vorgang existieren zwei miradlo-Galerien, die sich synchronisieren möchten. Sobald sich diese nun mit jeweils einer weiteren miradlo-Galerie synchronisiert haben, existieren vier miradlo-Galerien dieser Art. Es ist zu sehen, dass sich dieses ereignisbasierte Verhalten aufschaukelt und

nach kurzer Zeit sehr viele miradlo-Galerien erreicht. Es muss verhindert werden, dass miradlo-Galerien durch exzessive Synchronisationsvorgänge zu stark belastet werden.

Aus diesem Grund wird die Begrenzung eingeführt, auf einer miradlo-Galerie zu jeder Zeit höchstens eine Synchronisation durchzuführen. Synchronisationsanfragen werden während einer bereits laufenden Synchronisation abgelehnt. Anfragende miradlo-Galerien, die eine solche Ablehnung erhalten, brechen die Synchronisation mit dieser miradlo-Galerie sofort ab.

Das nachfolgende Sequenzdiagramm [23] besteht aus den miradlo-Galerien A, B und C sowie einem Service-Locator und einem Redakteur. Es wird davon ausgegangen, dass alle drei miradlo-Galerien die freigegebene Bildergalerie *bildergalerie2* besitzen. Weiter sind alle drei miradlo-Galerien zum Beginn des Prozesses in synchronisiertem Zustand.

Der Redakteur fügt der miradlo-Galerie A ein Bild der *bildergalerie2* hinzu. Die miradlo-Galerie A fragt daraufhin beim Service-Locator an, welche miradlo-Galerien die *bildergalerie2* besitzen. Die miradlo-Galerie A fängt daraufhin an, allen miradlo-Galerien dieser Liste außer sich selbst eine Synchronisationsanfrage zu senden. Im Diagramm werden diese Anfragen mit *sync(bildergalerie2)* ausgezeichnet. Auf diese Anfrage antwortet jede miradlo-Galerie mit einem *msg(ok)* oder einem *msg(rejected)*. Ein *ok* bedeutet, dass eine Synchronisation stattfindet. Ein *rejected* bedeutet wiederum, dass eine Synchronisation aufgrund einer bereits laufenden Synchronisation nicht stattfinden kann.

Alles in allem ist zu sehen, dass das neue Bild, welches vom Redakteur eingespielt wurde, zu sämtlichen miradlo-Galerien übertragen wird. Selbst bei einer anderen Abfolge der einzelnen Kommunikationspfade, wird am Ende sichergestellt sein, dass alle erreichbaren miradlo-Galerien auf dem neusten Stand sind.

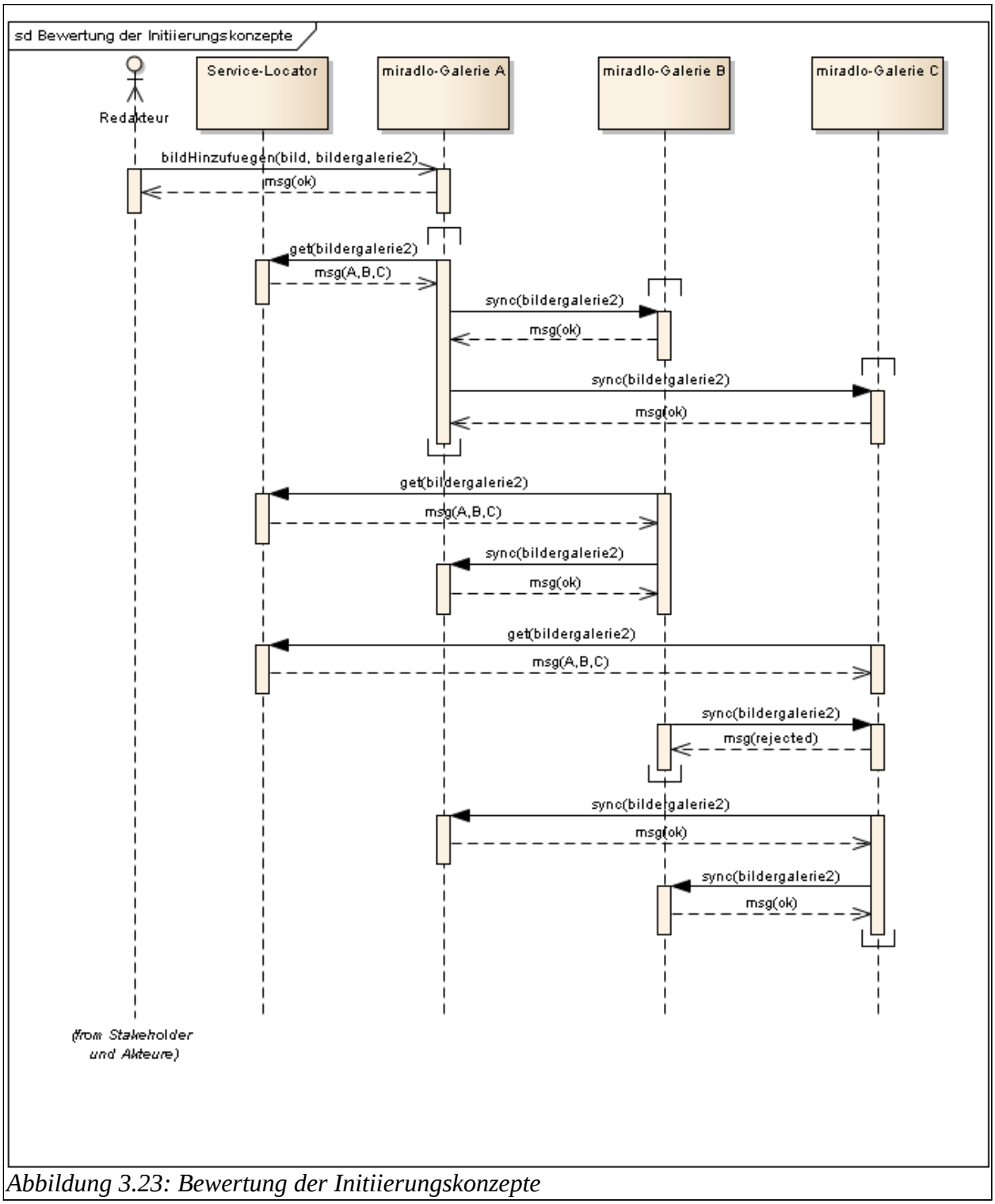

Abbildung 3.23: Bewertung der Initiierungskonzepte

Diagrammelement	Beschreibung
Service-Locator	Der Service-Locator hält die Informationen, welche freigegebene Bildergalerie von welcher miradlo-Galerie geführt wird. In diesem Beispiel hat der Service-Locator lediglich die Information, dass die Bildergalerie *bildergalerie2* von den miradlo-Galerien A, B und C geführt wird. Der Service-Locator liefert diese Information auf Anfrage hin aus.
miradlo-Galerie A	Dieser miradlo-Galerie wird vom Redakteur ein Bild hinzugefügt. Daraufhin beginnt sie, sich mit allen miradlo-Galerien zu synchronisieren, welche die *bildergalerie2* besitzen. Die miradlo-Galerie A synchronisiert sich erfolgreich mit den miradlo-Galerien B und C. Bei den miradlo-Galerien B und C stößt sie aufgrund der Übertragung neuer Inhalte ebenfalls eine Synchronisation an.
miradlo-Galerie B	Die miradlo-Galerie B wird von miradlo-Galerie A um neue Inhalte erweitert. Daraufhin beginnt die miradlo-Galerie B ebenfalls, sich mit relevanten miradlo-Galerien zu synchronisieren. Eine erneute Synchronisationsanfrage bei miradlo-Galerie A verläuft erfolgreich. Weil bei miradlo-Galerie A keine neuen Inhalte hinzugefügt werden, führt miradlo-Galerie A keine erneute Synchronisation mit allen relevanten miradlo-Galerien durch. Die darauf folgenden Synchronisation von miradlo-Galerie B und miradlo-Galerie C schlägt fehl, da bei miradlo-Galerie C bereits eine Synchronisation im Gange ist.
miradlo-Galerie C	Die miradlo-Galerie C wird ebenfalls von miradlo-Galerie A um neue Inhalte erweitert. Daraufhin beginnt die miradlo-Galerie C, sich mit relevanten miradlo-Galerien zu synchronisieren. Ihre Synchronisationsvorgänge mit miradlo-Galerie A und B verlaufen erfolgreich. Diese Vorgänge stoßen bei den miradlo-Galerien A und B keine erneute Synchronisation an, weil der Datenbestand von *bildergalerie2* bereits abgeglichen ist.

3.7.4 Prüfungen während der Synchronisation

Im *Kapitel 3.7.2, Methodik der Synchronisation, Seite 58,* wurde beschlossen, Hashsummen zur Identifizierung von Inhalten zu verwenden. Es wurde beschrieben, dass bis auf die Hashsummen der Bildergalerien sämtliche Hashsummen der miradlo-Galerie aktuell gehalten werden. Es wurde jedoch noch nicht detailliert beschrieben, inwiefern die Hashsummen Einfluss auf den Synchronisationsvorgang haben.

Im folgenden werden Varianten entwickelt, analysiert und bewertet, welche diese Problemstellung lösen.

3.7.4.1 Hashing-Bedarfsauswertung

Die Bezeichnung der im folgenden beschriebenen Variante lautet *Hashing-Bedarfsauswertung.* Der Begriff ist dabei an Lazy Evaluation [24] angelehnt.

Der Kern dieses Ansatzes ist es, dass sämtliche Hashsummen auf dem System berechnet werden, auf dem die Änderung stattfindet.

Beispielsweise wird beim Anlegen einer neuen Transformationsbeschreibung von der miradlo-Galerie direkt eine Hashsumme über die Transformationsbeschreibung berechnet. Diese Änderung hat zur Folge, dass die Hashsumme des Bildes, zu dem die Hashsumme gehört, ebenfalls neu berechnet wird. Weiter wird das DirtyFlag der Bildergalerie, die das soeben geänderte Bild enthält, gesetzt.

Wird nun eine Synchronisation durchgeführt, werden die Hashsumme und die neue Transformationsbeschreibung übertragen. Die miradlo-Galerie, die diese für sie neue Transformationsbeschreibung empfängt, führt keine erneute Berechnung durch. Es findet lediglich die direkte Speicherung der Transformationsbeschreibung und der Hashsumme statt.

Es wird folglich immer auf der miradlo-Galerie die Hashsumme einer Transformationsbeschreibung bzw. eines Originalbildes berechnet, in der die Erstellung stattfand.

Vorteil

Da die Berechnung nur einmalig stattfindet, sparen die miradlo-Galerien bei der Synchronisation Prozessorzeit [CPUT]. Die bereits vorliegenden Daten werden übernommen, abgespeichert und bei folgenden Synchronisationen weiterverbreitet.

Nachteil

Eventuell auftretende Fehler bei der Übertragung werden nicht erkannt. Eine einmalig auftretende fehlerhafte Übertragung kann sich auf das gesamte Netzwerk aus miradlo-Galerien ausbreiten. Neben Übertragungsfehlern können sonstige Fehler ebenfalls nicht erkannt werden. Das Dateisystem, in dem Bilder abgelegt sind, kann theoretisch aufgrund vielfältiger Ursachen fehlerhafte Inhalte liefern, die erst von Benutzern während des Betrachtens erkannt werden.

Fazit

Dieser Ansatz ermöglicht durch die einmalige Berechnung von Hashsummen Prozessorzeit und somit Energie zu sparen. Falls keine Fehlersicherung außerhalb der miradlo-Galerie stattfindet, können sich korrumpierte Daten auf viele miradlo-Galerien ausbreiten.

3.7.4.2 Hashing mit strikter Auswertung

Die Bezeichnung der im folgenden beschriebenen Variante lautet *Hashing mit strikter Auswertung.* Der Begriff ist dabei an Eager Evaluation [24] angelehnt.

Bei diesem Ansatz werden Hashsummen während der Synchronisation auf jeder miradlo-Galerie stets neu berechnet. Wenn beispielsweise eine miradlo-Galerie eine neue Transformationsbe-

schreibung eines Bildes durch die Synchronisation mit einer anderen miradlo-Galerie erhält, liegen die Hashsumme und die Transformationsbeschreibung bereits durch die Übertragung vor. Die empfangende miradlo-Galerie führt eine erneute Berechnung der Hashsumme durch, um zu überprüfen, ob die empfangenen Daten korrekt sind.

Vorteil

Eine miradlo-Galerie erkennt vor dem Abspeichern neuer Daten, ob Übertragungsfehler aufgetreten sind.

Nachteil

Bei einer Synchronisation muss jede miradlo-Galerie die gleiche Berechnung durchführen. Je mehr miradlo-Galerien eine freigegebene Bildergalerie besitzen, desto öfter wird die gleiche Berechnung durchgeführt.

Fazit

Die Fehlererkennung bei der Übertragung zieht Rechenaufwand nach sich, welcher jede miradlo-Galerie belastet.

3.7.4.3 Bewertung der Prüfungsansätze

Da jede miradlo-Galerie Hashsummen über ihre Bildergalerien abspeichert, kann bei Bedarf eine Überprüfung des gesamten Datenbestands automatisiert durchgeführt werden. Dabei werden die Hashsummen sämtlicher Inhalte erneut berechnet und mit dem gespeicherten Wert verglichen. Falls eine Abweichung festgestellt wird, kann zumindest ein Fehlerprotokoll angelegt werden.

Die nicht vorhandene Neuberechnung und die damit verbundene Fehlererkennung beim Ansatz *Hashing-Bedarfsauswertung* hat zur Folge, dass die Ursache von Datenkorrumpierung innerhalb eines Netzwerks aus miradlo-Galerien nicht erkannt werden kann.

Die Variante *Hashing mit strikter Auswertung* ermöglicht es, während der Synchronisation evtl. auftretende Fehler zu erkennen und eine erneute Übertragung der fehlerhaften Daten zu initiieren. Sollte der Fehler bestehen bleiben, kann die miradlo-Galerie, die den Fehler bemerkt, die beteiligte miradlo-Galerie über die Dateninkonsistenz informieren.

Bei der Realisierung der miradlo-Galerie wird auf die Variante *Hashing mit strikter Auswertung* zurückgegriffen.

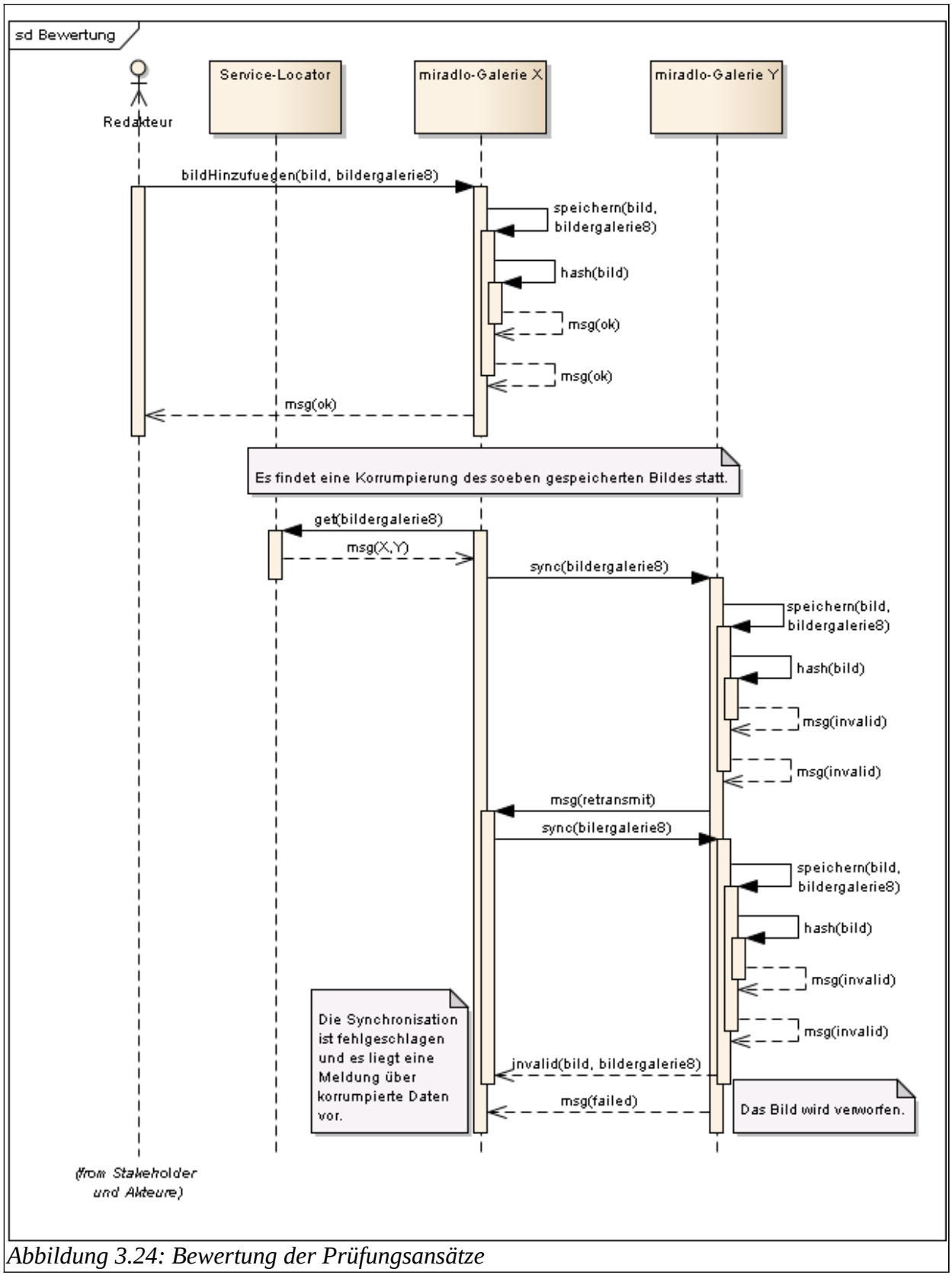

Abbildung 3.24: Bewertung der Prüfungsansätze

Das Sequenzdiagramm [23] B*ewertung der Prüfungsansätze* besteht aus den miradlo-Galerien X und Y sowie einem Service-Locator und einem Redakteur. Es wird davon ausgegangen, dass beide miradlo-Galerien die freigegebene Bildergalerie *bildergalerie8* besitzen.

Der Redakteur fügt der miradlo-Galerie X ein Bild der *bildergalerie8* hinzu. Nach diesem Vorgang korrumpieren die Daten des soeben abgespeicherten Bildes aufgrund eines Festplattenfehlers. Durch die Änderung des Redakteurs, beginnt die miradlo-Galerie X, die *bildergalerie8* zu synchronisieren. Die miradlo-Galerie X fragt beim Service-Locator an, welche miradlo-Galerien die *bildergalerie8* besitzen. Dieser antwortet, dass lediglich die miradlo-Galerie X und Y die *bildergalerie8* besitzen. Die miradlo-Galerie X sendet daraufhin eine Synchronisationsanfrage an die miradlo-Galerie Y. Im Diagramm wird diese Anfragen mit *sync(bildergalerie8)* ausgezeichnet. Die miradlo-Galerie X übermittelt bei diesem Vorgang, welche Hashsumme das neue Bild haben wird. Die miradlo-Galerie Y stellt bei der Verarbeitung des neuen Bildes eine Abweichung der Hashsumme fest und fordert die miradlo-Galerie X auf, die Synchronisation erneut zu starten. Da dieser Fehler erneut auftritt, verwirft die miradlo-Galerie Y sämtliche empfangene Daten und informiert die miradlo-Galerie X über das Vorliegen der Inkonsistenz.

Diagrammelement	**Beschreibung**
Service-Locator	Der Service-Locator hält die Informationen, welche freigegebene Bildergalerie von welcher miradlo-Galerie geführt wird. In diesem Beispiel hat der Service-Locator lediglich die Information, dass die Bildergalerie bildergalerie8 von den miradlo-Galerien X und Y geführt wird. Der Service-Locator liefert diese Information auf Anfrage hin aus.
miradlo-Galerie X	Der miradlo-Galerie X wird von einem Redakteur um ein neues Bild erweitert. Da nach dem Speichern des Bildes eine Korrumpierung dessen stattfindet, schlägt die darauf folgende Synchronisation mit miradlo-Galerie Y fehl.
miradlo-Galerie Y	Die miradlo-Galerie Y erkennt eine Dateninkonsistenz und bricht letztendlich die Synchronisation ab.

4 Technische Konzepte

In den bisherigen Analysen lag der Fokus primär auf den Punkten Persistenz, Ablaufsteuerung, Replikation, Verteilung, Kommunikation und Integration. Dies ist darin begründet, dass die extrahierten Geschäftsprozesse *Kapitel 2, Geschäftssicht, Seite 7,* im Großen und Ganzen in diesen Bereichen angesiedelt sind.

Die bisherigen Betrachtungen reichen jedoch nicht aus, um das System der miradlo-Galerie vollends zu beschreiben. Es existieren noch relevante Aspekte, die es zu beleuchten gilt, bevor eine Umsetzung der miradlo-Galerie in Betracht gezogen werden kann. Das folgende Diagramm visualisiert, auf welche Aspekte bisher besonders Wert gelegt wurde.

Im folgenden Diagramm werden die Schwerpunkte dieser Bachelorthesis in einer Wolke visualisiert. Die verschiedenen Aspekte sind dabei unterschiedlich groß dargestellt. Es gilt, je größer ein Aspekt ist, desto ausführlicher wurde er bereits betrachtet.

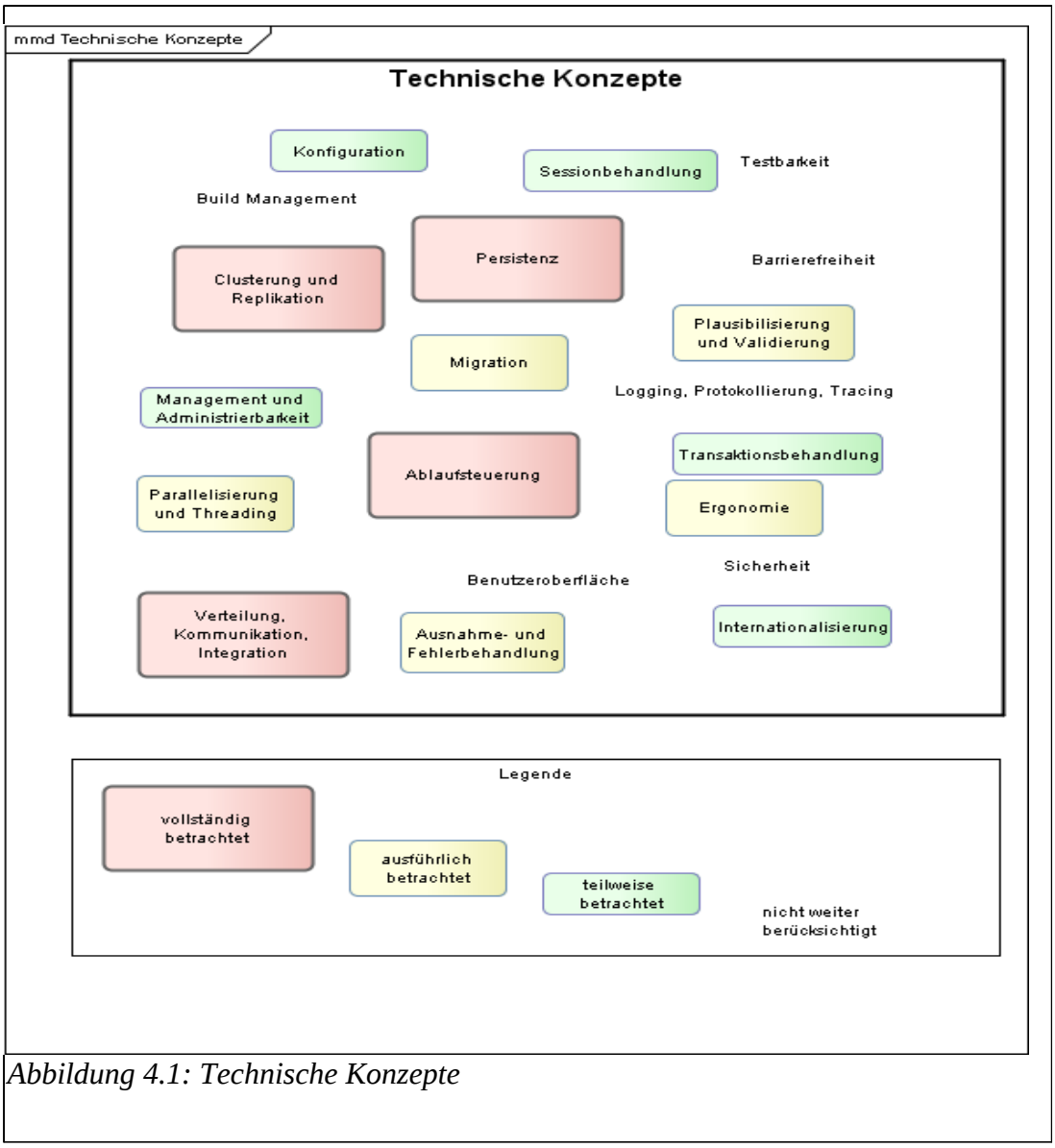

Abbildung 4.1: Technische Konzepte

4.1 Persistenz

Im *Kapitel 3, Problemstellung und Analyse, Seite 20,* wurden in den Unterkapiteln *Kapitel 3.3, Datenverwaltung, Seite 31,* und *Kapitel 35, Datenspeicherung, Seite 35,* diverse Varianten analysiert. Dabei wurde beschlossen, auf Bildergalerien zurückzugreifen. Die in einer Bildergalerie befindlichen Bilder können obendrein mit Tags versehen werden, um eine weitere Strukturierung zu erreichen. Bei der Datenspeicherung wird das Dateisystem für sämtliche Binärdaten und die Datenbank für Transformationsbeschreibungen und Metadaten eingesetzt. In der *Kapitel 3.6, Adress- und Inhaltsverwaltung, Seite 48,* wurde beschlossen einen Service-Locator mit Löschliste, siehe auch *Kapitel 3.6.3, Service-Locator, Seite 53,* zu nutzen.

Das nachfolgende Diagramm ist in zwei Hauptbereiche unterteilt. Im oberen Teil wird der Service-Locator und im unteren Teil wird die miradlo-Galerie visualisiert. Zwischen beiden Teilbereichen befinden sich Kommunikationsschnittstellen, welche die gemeinsame Basis der Kommunikation festlegen.

Innerhalb der miradlo-Galerie wurden an vielen Stellen Schnittstellen (Interfaces) eingeführt, welche die Abhängigkeiten zwischen den einzelnen Entitäten abschwächen. Die Interfaces garantieren obendrein eine konsistente Architektur, welche Änderungen in den konkreten Implementierungen zulässt. Durch allgemein gehaltene Entitäten, wie *Picture* wird erreicht, dass beispielsweise jede Version und jedes Originalbild ein skaliertes Bild und ein Miniaturbild besitzt.

SLGPGID (Service-Locator-Global-Picture-Gallery-ID) in *IMiradloGallery* wird vom Service-Locator erzeugt. Eine neue freigegebene Bildergalerie bekommt diese beim Registrieren am Service-Locator zugewiesen.

SLGMGID (Service-Locator-Global-Miradlo-Gallery-ID) wird vom Service-Locator erzeugt. Eine miradlo-Galerie bekommt diese beim Registrieren am Service-Locator zugewiesen.

Die Assoziation zwischen SLMiradloGallery und PeerMiradloGalllery existiert nur, wenn sich die miradlo-Galerie bei einem Service-Locator registriert hat. Wenn sich eine miradlo-Galerie nicht an einem Service-Locator registriert hat, kann sie keinerlei Synchronisationen durchführen.

Eine miradlo-Galerie überprüft periodisch, ob der Hash der Löschliste (SLDeletionlist) abweicht. Sollte dies der Fall sein, werden sämtliche Einträge (SLDeletedPicture) dieser Löschliste heruntergeladen und alle Bilder (Picture) überprüft, ob ein zu löschendes Originalbild existiert. Sollte dies der Fall sein, wird das gesamte Bild (Picture) aus der miradlo-Galerie entfernt.

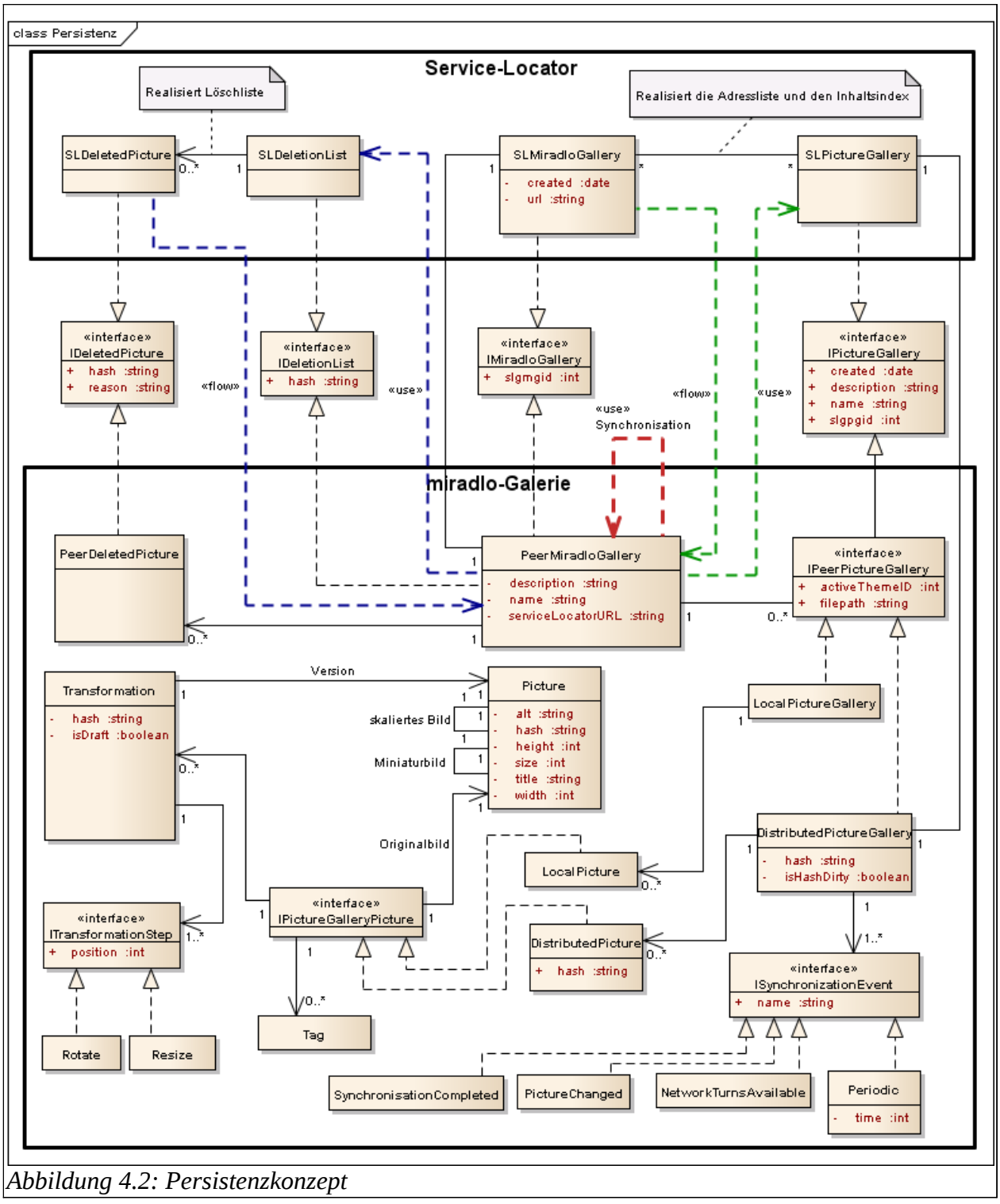

Abbildung 4.2: Persistenzkonzept

Diagrammelement	Beschreibung
SynchronisationCompleted	Dieses Synchronisationsereignis löst eine Synchronisation aus, sobald die Synchronisation dieser freigegebenen Bildergalerie (DistributedPictureGallery) abgeschlossen wurde und dabei in dem Vorgang ein Bild (DistributedPicture) auf der eigenen miradlo-Galerie (PeerMiradloGallery) geändert wurde.

Diagrammelement	Beschreibung
Tag	Mithilfe dieser Entität werden Tags, *Kapitel 3.3.2, Tags, Seite 32*, abgebildet.
PictureChanged	Dieses Synchronisationsereignis löst eine Synchronisation aus, sobald ein Bild (DistributedPicture) einer freigegebenen Bildergalerie (DistributedPictureGallery) in irgendeiner Form geändert wurde. Eine solche Änderung kann durch einen Benutzer erfolgen.
LocalPictureGallery	Diese Bildergalerie ist nicht im Netzwerk aus miradlo-Galerien freigegeben und ausschließlich auf der miradlo-Galerie (PeerMiradloGalllery) verfügbar, wo sie erstellt wurde. Das Attribut slgpgid (IPictureGallery) wird nicht verwendet.
NetworkTurnsAvailable	Dieses Synchronisationsereignis löst eine Synchronisation aus, sobald eine Netzwerkverbindung wieder zur Verfügung steht.
Periodic	Dieses Synchronisationsereignis tritt zeitlich periodisch auf. Immer wenn das Zeitintervall (time) verstrichen ist, wird eine Synchronisation der freigegebenen Bildergalerie (DistributedPictureGallery) angestoßen.
Resize	Dieser Transformationsschritt wird verwendet, um eine Skalierung des Bildes durchzuführen.
Rotate	Dieser Transformationsschritt wird verwendet, um eine Rotation des Bildes durchzuführen.
DistributedPictureGallery	Dies ist eine freigegebene Bildergalerie. Sie wird innerhalb des Netzwerks aus miradlo-Galerien zwischen verschiedenen miradlo-Galerien (PeerMiradloGallery) synchronisiert. Die Synchronisation findet zu bestimmten Synchronisationsereignissen (ISynchronizationEvent) statt. Bei der Erstellung einer freigegebenen Bildergalerie wird sie beim Service-Locator registriert.
PeerDeletedPicture	Diese Entität enthält Informationen über ein zu löschendes Originalbild. Diese Entität existiert nur auf den miradlo-Galerien (PeerMiradloGallery).
PeerMiradloGallery	Diese Entität speichert Basisinformationen, um eine miradlo-Galerie betreiben zu können. Damit die Synchronisation der miradlo-Galerie verwendet werden kann, muss eine Registrierung an einem Service-Locator durchgeführt werden. Mithilfe der Adressliste und des Inhaltsindex des Service-Locators, können freigegebene Bildergalerien (DistributedPictureGallery) gesucht werden. Wenn eine miradlo-Galerie eine freigegebene Bildergalerie einer entfernten miradlo-Galerie nutzen möchte, so muss sie dies beim Service-Locator anmelden. Danach kann sie sich eine leere freigegebene miradlo-Galerie (DistributedPictureGallery) erstellen und eine Synchronisation durchführen.
SLDeletedPicture	Diese Entität enthält Informationen über ein zu löschendes

Diagrammelement	Beschreibung
	Originalbild. Diese Entität existiert nur auf dem Service-Locator (PeerMiradloGallery).
SLDeletionList	Diese Entität verwaltet alle gelöschten Originalbilder auf dem Service-Locator. Sie wird von dem Administrator des Service-Locators mit Informationen über zu löschende Originalbilder versorgt.
SLMiradloGallery	Diese Entität hält alle Informationen, die ein Service-Locator von einer miradlo-Galerie benötigt, um sie mit anderen miradlo-Galerien im Netzwerk aus miradlo-Galerien zu verwalten. Sie wird besonders für die Adressliste und den Inhaltsindex verwendet.
SLPictureGallery	Diese Entität hält alle Informationen, die ein Service-Locator von einer freigegebenen Bildergalerie (DistributedPictureGallery) benötigt, um sie im Netzwerk aus mirado-Galerien zu verwalten. Sie wird besonders für den Inhaltsindex und die Adressliste verwendet. Das Finden von relevanten miradlo-Galerien wird durch sie ermöglicht.
LocalPicture	Dieses Bild wird ausschließlich in lokalen Bildergalerien (LocalPictureGallery) eingesetzt.
DistributedPicture	Dieses Bild wird ausschließlich in freigegebenen Bildergalerien (DistributedPictureGallery) eingesetzt.
Picture	Dieses Bild (Picture) bildet die grundlegende Entität eines Bildes in der miradlo-Galerie. Mit ihr als Basis werden Originalbild und Version in *IPictureGalleryPicture* bzw. *Transformation* umgesetzt.
Transformation	Eine Transformationsbeschreibung (Transformation) besteht aus Transformationsschritten (ITransformationStep) und einer Version. Die Version ist das resultierende Bild (Picture), wenn alle Transformationsschritte auf dem Originalbild der Bildergalerie (IPictureGalleryPicture) der Reihe nach durchgeführt werden.
ITransformationStep	Die einzelnen Transformationsschritte (ITransformationStep) werden in beschreibender Form zwischen den miradlo-Galerien (PeerMiradloGalerie) ausgetauscht. Die hier dargestellten konkreten Transformationsschritte sind Beispiele. Im Laufe der Entwicklung werden weitere hinzukommen.
IPictureGalleryPicture	Dieses Bild einer Bildergalerie (IPictureGalleryPicture) bildet die Verbindung zwischen Originalbildern und ihren zugehörigen Transformationsbeschreibungen (Transformation).
IDeletedPicture	Dieses zu löschende Bild (IDeletedPicture) bildet die Basis der Kommunikation zwischen Service-Locator und miradlo-Galerien (PeerMiradloGallery). Es stellt die Basis für Datenobjekte dar, welche zwischen Service-Locator (SLDeletedPicture) und miradlo-Galerien (PeerMiradloGallery) ausgetauscht werden.
IDeletionList	Diese Löschliste (IDeletionList) bildet die Basis für ihre Implementierung auf der Seite des Service-Locators

Diagrammelement	Beschreibung
	(SLDeletionList) und der miradlo-Galerie (PeerMiradloGallery).
IMiradloGallery	Diese miradlo-Galerie (IMiradloGallery) bildet die Basis der Kommunikation zwischen Service-Locator (SLMiradloGallery) und miradlo-Galerien (PeerMiradloGallery) dar. Es stellt die Basis für Datenobjekte dar, welche zwischen Service-Locator (SLMiradloGallery) und miradlo-Galerien (PeerMiradloGallery) ausgetauscht werden.
IPictureGallery	Diese Bildergalerie (IPictureGallery) bildet die Basis der Kommunikation zwischen Service-Locator (SLPictureGallery) und miradlo-Galerien (PeerMiradloGallery) dar.
IPeerPictureGallery	Diese Bildergalerie einer miradlo-Galerie (IPeerPictureGallery) erweitert die allgemeine Bildergalerie (IPictureGallery). Sie ist für die Verwaltung durch eine miradlo-Galerie (PeerMiradloGallery) zugeschnitten. Sie bildet die Basis für die Implementierung von konkreten Bildergalerien.
ISynchronizationEvent	Dieses Synchronisationsereignis (ISynchronizationEvent) bildet die Basis für die Implementierung konkreter Synchronisationsereignisse.

4.2 Benutzeroberfläche

In dieser Arbeit liegt der Schwerpunkt auf der Synchronisation mehrerer miradlo-Galerien. Konzepte für Benutzeroberflächen wurden nicht erarbeitet. Ein Folgeprojekt kann an dieser Arbeit anknüpfen und Konzepte für die Benutzeroberflächen der miradlo-Galerie erstellen und umsetzen.

4.3 Barrierefreiheit

In miradlokit werden bereits durchgängig Maßnahmen ergriffen, um ein barrierearmes Arbeiten zu ermöglichen. Die Benutzeroberflächen sind dabei grundsätzlich so optimiert, dass eine ausschließliche Bedienung mit der Tastatur komfortabel möglich ist. Weiter werden die Inhalte so aufbereitet, dass zu Elementen, wie Bildern Alternativtexte ausgegeben werden. Bei der Umsetzung der miradlo-Galerie in einem Folgeprojekt ist stets darauf zu achten, dass barrierearme Konzepte, wo es möglich und sinnvoll ist, eingesetzt werden.

4.4 Ergonomie

Die Ergonomie (Benutzbarkeit) wird mithilfe der bisher erarbeiteten Lösungen an vielen Stellen gewährleistet. Beispielsweise wird es dem Benutzer mit einer automatischen Synchronisation *Abbildung 3.23: Bewertung der Initiierungskonzepte, Seite 70,* ermöglicht, sich ausschließlich auf die Arbeit mit der miradlo-Galerie zu konzentrieren. Er braucht sich nicht um die Belange der Synchronisation zu kümmern.

Mithilfe von Bildergalerien und Tags *Abbildung 3.8: Bildergalerien mit Tags, Seite 35,* ist der Benutzer in der Lage, sich anhand mehrerer Mechanismen eine für ihn angenehme Struktur aufzubauen.

miradlokit bietet durch sein klar aufgebautes Bedienkonzept die Möglichkeit, sich in jedem Bedienschritt durch eine Hilfefunktion über die vorzunehmenden Eingaben zu informieren. Besonders für unerfahrene Benutzer erhöht dies die Benutzbarkeit. Weiter beinhaltet miradlokit eine größtenteils automatische und umfangreich beschriebene Installationsfunktion. Mit deren Hilfe ist ein jeder Benutzer in der Lage, eine miradlo-Galerie mit wenigen Eingaben in Betrieb zu nehmen.

Weitere tiefgreifende Ergonomiekonzepte wurden bisher nicht realisiert. Diese sollten in einem Folgeprojekt in Verbindung mit dem Entwurf der Benutzeroberflächen und deren Ablaufsteuerung angegangen werden.

4.5 Ablaufsteuerung

Die in der miradlo-Galerie stattfindenden Abläufe wurden bisher besonders im Bezug auf die Synchronisation beschrieben. Die Sequenzdiagramme für stattfindende Prüfungen, *Abbildung 3.24: Bewertung der Prüfungsansätze, Seite 74* und die Initiierung der Synchronisation *Abbildung 3.23: Bewertung der Initiierungskonzepte, Seite 70,* bilden den Kern des Synchronisationskonzeptes.

In einem Folgeprojekt, kann diese Grundlage in Kombination mit dem Prinzip der Hashsummen, *Abbildung 3.19: Ablauf der Synchronisation einer miradlo-Galerie Seite 63, Abbildung 3.20: Ablauf der Synchronisation einer Bildergalerie Seite 64 und Abbildung 3.21: Ablauf der Synchronisation eines Bildes, Seite 65,* erfasst und umgesetzt werden. Weitere Abläufe finden sich in der Versionsverwaltung.

In einem Folgeprojekt kann diese auf Basis der Transformationsbeschreibungen, *Abbildung 3.5: Originalbild mit Transformationsbeschreibungen Seite 28,* umgesetzt werden.

Die eigentliche Umsetzung der miradlo-Galerie, welche in einem Folgeprojekt stattfinden kann, sollte stets die Betrachtungen zum Thema Datenspeicherung *Kapitel: Datenspeicherung, Seite 35,* beachten. Besonders bei der Verwendung von miradlokit als Basis der Umsetzung können dadurch erhebliche Leistungsbeschränkungen vermieden werden.

Die in einem Folgeprojekt zu erstellenden Benutzeroberflächen müssen den grundlegenden Bedienungsabläufen von miradlokit genügen. Beispielsweise ist es Vorgabe, nicht mehr als sie-

ben Menü bzw. Eingabeelemente dem Benutzer auf einer Seite zu präsentieren. Die Aufteilung in mehrere Eingabeseiten kann nur erfolgen, wenn klare Bedienungsabläufe entworfen und beschrieben werden.

4.6 Transaktionsbehandlung

Die miradlo-Galerie besitzt Aktionen, die stets vollständig durchlaufen werden müssen, um die Konsistenz des Systems zu gewährleisten. Beispielsweise darf die Berechnung einer Hashsumme nur stattfinden, wenn kein anderer Prozess schreibenden Zugriff auf die zu hashenden Daten besitzt. Sollte dies trotzdem erfolgen, ist damit zu rechnen, dass die Hashsumme nicht dem verarbeiteten Inhalt entspricht. Die daraus entstehende Inkonsistenz kann die Funktionalität der miradlo-Galerie erheblich stören. Weiter dürfen übertragene Daten nur in den Datenbestand einer miradlo-Galerie aufgenommen werden, wenn deren Konsistenz, *Abbildung 3.24: Bewertung der Prüfungsansätze, Seite 74*, überprüft wurde.

Bei der Umsetzung der Synchronisation in einem Folgeprojekt sollte stets darauf geachtet werden, dass Transaktionen so klein wie möglich sind. Beispielsweise könnte eine Synchronisation nur als erfolgreich betrachtet werden, wenn sämtliche Daten einwandfrei übertragen wurden. Bei zeitlich gesehen kurzen Synchronisationsprozessen stellt dieses Vorgehen kein Problem dar. Wenn jedoch eine 1.000 Bilder umfassende Synchronisation bei dem 999. Bild wegen eines Übertragungsfehlers abbricht, kann die Akzeptanz des Systems durch den Benutzer erheblich sinken. Wenn die Transaktion der Synchronisation auf die einzelnen Bilder aufgeteilt wird, ist der entstehende Nachteil für den Benutzer nur noch sehr gering.

4.7 Sessionbehandlung

Die Sessionbehandlung (Sitzungsbehandlung) wird vollständig von miradlokit übernommen. Plugins von miradlokit, *Abbildung 3.2: Plugin miradlo-Galerie in Abhängigkeit von miradlokit Seite 21*, wie der miradlo-Galerie wird dabei eine umfangreiche Benutzer und Rechteverwaltung bereitgestellt. Jedes Plugin ist dadurch selbst in der Lage, mithilfe von Konfigurationsoptionen zu bestimmen, welche Benutzergruppen bei der Einrichtung des Plugins in miradlokit angelegt werden sollen. In einem Folgeprojekt, welches die Umsetzung der miradlo-Galerie übernimmt, muss festgelegt werden, welche Benutzergruppen mit welchen konkreten Rechten ausgestattet werden. Die Basis für diese konzeptionellen Entscheidungen wurde bereits in der Geschäftssicht, *Abbildung 2.8: Stakeholder und Akteure, Seite 18*, erbracht.

4.8 Clusterung und Replikation

Die miradlo-Galerie ist in der bisher beschriebenen Form in der Lage, sich selbst zu replizieren. Mithilfe des Konzeptes des freigegebenen Bildergalerie, *Abbildung 3.8: Bildergalerien mit Tags , Seite 35*, ist es möglich Inhalte auf eine sehr große Anzahl an Rechnern zu verteilen und synchron zu halten.

Dieses Vorgehen macht Sicherheitskopien (Backups) von freigegebenen Bildergalerien überflüssig, falls sichergestellt ist, dass genügend miradlo-Galerien existieren, welche die entsprechenden Bildergalerien besitzen. Über den Service-Locator kann jede miradlo-Galerie in

periodischen Abständen überprüfen, wie viele miradlo-Galerien für eine bestimmte freigegebene Bildergalerie registriert sind. Der Administrator einer jeden miradlo-Galerie kann selbst entscheiden, wie viele relevante miradlo-Galerien mindestens notwendig sind, bis die lokale Datensicherung über Backups zu deaktivieren ist.

In einem Folgeprojekt oder direkt bei der Umsetzung der miradlo-Galerie kann eine simple Schnittstelle definiert werden. Deren Aufgabe ist es, eine Informationsquelle für evtl. existierende Backupprogramme eines Rechners zu Verfügung zu stellen. Über einen solchen Mechanismus kann die Notwendigkeit der Erstellung von Backups für bestimmte freigegebene Bildergalerien an Umsysteme weitergeleitet werden. Lokale Bildergalerien sollten grundsätzlich durch Backups gesichert werden.

Eine Erhöhung der Verfügbarkeit erreicht die miradlo-Galerie dadurch, dass freigegebene Bildergalerien auf mehreren miradlo-Galerien bereitgestellt werden können. Dieser Umstand ermöglicht es Benutzern, dass sie beim Ausfall einer miradlo-Galerie einfach eine andere miradlo-Galerie weiterverwenden können, ohne ihre Arbeit zu verlieren. Das Finden von relevanten miradlo-Galerien kann für die, von einem Ausfall betroffenen, Benutzer jedoch zu einem schwierigen Unterfangen werden. Über eine Fehlerseite könnte die vom Ausfall betroffene miradlo-Galerie auf andere verfügbare relevante miradlo-Galerien verweisen. Weiter könnte der Service-Locator von Benutzern direkt nach relevanten miradlo-Galerien befragt werden. Ein Folgeprojekt könnte diese und weitere Möglichkeiten evaluieren und eine allgemeine Empfehlung zur Lösung dieses Problems aussprechen.

In einem weiteren Folgeprojekt könnte der Service-Locator der miradlo-Galerie um einen Mechanismus erweitert werden, welcher ihn befähigt, mit einem anderen Service-Locator zu kommunizieren. Auf diese Weise könnte ein Netzwerk aus Netzwerken von miradlo-Galerien erstellt werden. Dieser Schritt würde die miradlo-Galerie für den Einsatz im globalen Rahmen befähigen.

4.9 Sicherheit

Die Sicherheit der miradlo-Galerie beruht teilweise auf der Sessionbehandlung von miradlokit. Dies hat den Vorteil, dass neue Sicherheitsmechanismen zentral in miradlokit eingepflegt werden können, ohne Plugins, wie die miradlo-Galerie überarbeiten zu müssen.

Umfangreiche Administrationsrechte für den Rechner, auf dem die miradlo-Galerie läuft, sind lediglich bei der Installation von Nöten. Jede weitere Konfiguration und die eigentliche Benutzung findet ausschließlich im Kontext von miradlokit statt. Da durch dieses Konzept ein regelmäßiger privilegierter Zugriff auf den Rechner vermieden wird, sinkt die Wahrscheinlichkeit einer Fehlbedienung, welche die Schaffung einer Sicherheitslücke zur Folge haben könnte.

Es wird in dieser Bachelorthesis auf die Analyse von Sicherheitsmechanismen für die Synchronisation von miradlo-Galerien verzichtet. Dies ist darin begründet, dass es sich bei der miradlo-Galerie um eine Software handelt, die das Ziel hat, anderen Leuten den Zugang zu Bildern zu erleichtern. Es wurde dadurch anstatt einer verschlüsselten Übertragung eine Konsistenzprüfung, *Abbildung 3.24: Bewertung der Prüfungsansätze, Seite 74*, entwickelt.

Nichtsdestotrotz sollten in einem Folgeprojekt die möglichen Sicherheitsrisiken für die miradlo-Galerie analysiert und ein Konzept erstellt werden. Es sei dabei besonders auf das Szenario von böswilligen Angriffen verwiesen, in denen miradlo-Galerien über die Synchronisationsschnittstellen absichtlich mit fehlerhaften Daten versorgt werden. Die dabei entstehenden Missstände könnten die Akzeptanz und den Ruf des Systems massiv schädigen. Die Einführung einer gesicherten Authentisierung zwischen miradlo-Galerie und Service-Locator würde einen erheblichen Gewinn an Sicherheit mit sich bringen. Nur wenn bei der Kommunikation zwischen miradlo-Galerie und Service-Locator die Identität und damit die Vertrauenswürdigkeit zu jeder Zeit sichergestellt ist, kann von einem grundlegenden Sicherheitsmechanismus gesprochen werden. In einem nächsten Schritt kann dann die Kommunikation zwischen den miradlo-Galerien selbst abgesichert werden.

4.10 Verteilung, Kommunikation, Integration

Die Themen Verteilung, *Kapitel 3.5, Verteilung, Seite 40,* Kommunikation *Kapitel 3.7, Datenaustausch, Seite 55* und Integration *Kapitel 3.1, Grundlagen, Seite 20,* wurden in dieser Arbeit bereits ausführlich beschrieben. Für die Umsetzung der miradlo-Galerie und deren Synchronisationsmechanismus sei an dieser Stelle auf JSON [30] verwiesen. JSON ist ein im Internet eingesetztes Datenaustauschformat. Die Skriptsprache PHP, in welcher miradlokit implementiert ist, bietet für JSON native En- und Dekodierungsfunktionen an. Weiter wird JSON von einer Vielzahl anderer Programmiersprachen ebenfalls unterstützt. Dieser Verweis hat nur Empfehlungscharakter. Es können selbstverständlich auch andere Datenaustauschformate in Betracht gezogen werden.

4.11 Ausnahme- und Fehlerbehandlung

In dieser Arbeit wurde die Ausnahme- und Fehlerbehandlung bereits im Bereich der Synchronisation, *Kapitel 3.7.4, Prüfungen während der Synchronisation, Seite 71,* angesprochen. Hierbei können sich miradlo-Galerien während der Synchronisation gegenseitig über Fehlerzustände informieren. Beim Auftreten einer Inkonsistenz während der Übertragung, kann jede miradlo-Galerie die betroffenen Datenobjekte selbst überprüfen. Durch eine erneute Berechnung und Prüfung von Hashsummen kann somit eine verlässliche Aussage gemacht werden, ob der eigene Datenbestand in Ordnung ist oder ob er einer Wartung bedarf. Sobald beispielsweise eine solche Inkonsistenz erkannt wurde, kann eine miradlo-Galerie Synchronisationsanfragen für die betroffene freigegebene Bildergalerie ablehnen. Dies verhindert ein erneutes Fehlschlagen von Synchronisationsvorgängen. Nachdem ein Administrator die Fehlerursache beseitigt hat, kann er die Synchronisation manuell wieder aktivieren.

Die miradlo-Galerie sollte bei der Umsetzung die Möglichkeit bieten, einen Administrator mit einer automatisierten Fehlermeldung per E-Mail zu benachrichtigen. So können Ausfallzeiten minimiert und die Akzeptanz des Systems erhöht werden. Weiter ist darauf zu achten, dass Benutzer im Falle eines Fehler stets mit einer hilfreichen Fehlermeldung informiert werden. miradlokit bietet hierbei die Möglichkeit, einheitliche Fehlermeldungen für alle Bereiche des Systems zu erzeugen.

4.12 Management und Administrierbarkeit

Die Administration und das Management werden über die von miradlokit bereitgestellten Administrationswerkzeuge abgewickelt. Mithilfe dieser Werkzeuge kann jede miradlo-Galerie für sich selbst administriert werden.

In einem Folgeprojekt könnte eine Management-Lösung entwickelt werden, um ein gesamtes Netzwerk aus miradlo-Galerien gleichzeitig zu administrieren. Dies hätte den Vorteil evtl. auftretende Fehler früh zu erkennen und administrativ eingreifen zu können.

4.13 Logging, Protokollierung, Tracing

In miradlokit wird bereits eine detaillierte Protokollierung realisiert. Diese beschränkt sich allerdings nur auf das eigene System. Entfernte miradlo-Galerien werden dabei jedoch nicht berücksichtigt. Es ist mit aktuellen Mitteln somit nicht möglich, von einem zentralen Punkt aus, alle Protokolle gleichzeitig einzusehen.

In einem Folgeprojekt sollte eine netzwerkweite Administrationsoberfläche für miradlokit entwickelt werden. Die Administratoren der miradlo-Galerien wären mithilfe dieser zentralen Verwaltungseinheit in der Lage, jederzeit eine Aussage über den Zustand eines Netzwerks aus miradlo-Galerien zu machen.

Der Umfang der Protokollierung sollte grundsätzlich immer nur so hoch sein, um eine ausreichende Überwachung sicherzustellen. Eine permanente exzessive Protokollierung einer jeden Systemaktivität hätte lediglich zur Folge, dass die Leistungsfähigkeit des Systems eingeschränkt wird.

4.14 Konfiguration

Die Konfiguration der miradlo-Galerie findet über die von miradlokit vorgegebenen Konfigurationseinrichtungen statt. Beispielsweise kann über die Einrichtungsdialoge von miradlokit festgelegt werden, in welcher Bildgröße die Darstellung von Vorschaubildern erfolgen soll.

4.15 Parallelisierung und Threading

Die einzelnen miradlo-Galerien operieren außer während einer Synchronisation vollständig unabhängig voneinander. Mehrere miradlo-Galerien in einem Netzwerk aus miradlo-Galerien können somit parallel Benutzeranfragen bearbeiten. Selbst eine miradlo-Galerie allein ist durch die Verwendung heutiger Betriebssysteme und Hardware in der Lage, mehrere Anfragen gleichzeitig zu beantworten. Der Kontext des Webservers *Abbildung 3.12: Peer-To-Peer, Seite 46,* in dem miradlokit läuft, besitzt grundsätzlich die Möglichkeit für jede Anfrage einen eigenen Bearbeitungsprozess, *Abbildung 3.9: Bewertung der Datenspeicherungsansätze, Seite 39,* zu erzeugen. Diese Prozesse können dann parallel vom Betriebssystem des Computers bearbeitet werden.

4.16 Internationalisierung

In miradlokit existieren Mechanismen, um eine vollständige Mehrsprachigkeit der Benutzeroberfläche der miradlo-Galerie zu realisieren.

In einem Folgeprojekt kann geprüft werden, wie eine Internationalisierung der Inhalte einer miradlo-Galerie stattfinden könnte. Dabei wäre grundsätzlich die Idee, dass Benutzer, die mehrere Sprachen beherrschen, beispielsweise Titel und Beschreibung eines Bildes auch in einer weiteren Sprache angeben können. Damit könnte ein Bild z.B. mit einer deutschen, spanischen und englischen Beschreibung ausgestattet werden, (*siehe REQ1004 - miradlo-Galerie Name und Beschreibung mehrsprachig, Seite 97*)

4.17 Migration

Es wurde bereits beschrieben, wie Hashfunktionen *Kapitel 3.7.2, Methodik der Synchronisation: Datenaustausch-Methodik, Seite 58,* während des Betriebs der miradlo-Galerie über einen längeren Zeitraum ausgetauscht werden können. Erweiterbare Kommunikationsprotokolle spielen dabei eine wichtige Rolle. Durch die Möglichkeit, die unterstützten Versionsnummern einer bestimmten Funktion abzufragen, können unterschiedliche miradlo-Galerien verschiedener Versionsstände in einem Netzwerk aus miradlo-Galerien genutzt werden. Es sollte in diesem Sinn stets sichergestellt sein, dass ältere Versionen auch nach einer Aktualisierung des Service-Locators oder einer anderen miradlo-Galerie funktionstüchtig bleiben. Dies ist darin begründet, dass es nicht möglich ist, eine große Anzahl an Systemen gleichzeitig auf eine neue Softwareversion zu aktualisieren. Selbst über eine zentrale Administrationsinstanz ist dies nicht praktikabel. Zweckmäßig ist es stattdessen eine kleine Anzahl an miradlo-Galerien zu aktualisieren, um zu sehen, ob die neue Softwareversion den Anforderungen entspricht. Erst nachdem dies hinreichend sichergestellt ist, werden nach und nach ein großer Teil der miradlo-Galerien aktualisiert. Es muss stets damit gerechnet werden, dass manche miradlo-Galerien aus verschiedenen Gründen nie oder nur äußerst selten aktualisiert werden.

4.18 Testbarkeit

Bei der Umsetzung der miradlo-Galerie sollte permanentes, automatisiertes Testen im Mittelpunkt stehen. Die miradlo-Galerie besteht aus einer Vielzahl einzelner Komponenten. Diese können in der Regel allein automatisierten Regressionstests unterzogen werden. Die Durchführung eines Regressionstests bedeutet, dass alle existierenden Testfälle nacheinander auf die zu testende Komponente angewendet werden. Beispielsweise bietet sich dieses Vorgehen für den Service-Locator an. Da er eine zentrale Rolle in einem Netzwerk aus miradlo-Galerien einnimmt, ist seine verlässliche Funktion unabdingbar. Die Testfälle für diese Komponente sind dabei vor ihrer eigentlichen Umsetzung zu erstellen. Während der Erstellung des Service-Locator kann der Fortschritt seiner Implementierung dadurch stets überprüft werden. Bei eventuell auftretenden Fehlern, die bisher nicht durch einen Testfall abgedeckt werden, sind vor deren Beseitigung einer oder mehrere neue Testfälle zu erstellen, welches dieses Verhalten abdecken. Bei der Durchführung von Regressionstests ist darauf zu achten, dass in der zu testenden Komponente Zustände aus vorherigen Testfällen verbleiben können. Im schlimmsten

Fall kann dies dazu führen, dass die Reihenfolge der Ausführung der Testfälle entscheidet, ob die Komponente den Test besteht oder nicht. Um dies zu vermeiden, sollte jeder Testfall nach seiner Ausführung keinerlei Rückstände innerhalb der Komponente hinterlassen. Sollte dies nicht möglich sein, ist die Komponente so zu überarbeiten, dass ein Zurücksetzen sämtlicher interner Zustände möglich ist.

Diese allgemeinen Konzepte sind bei der Umsetzung der miradlo-Galerie aufzugreifen und mit passenden Mitteln zu realisieren.

4.19 Plausibilisierung und Validierung

Die Plausibilisierung der miradlo-Galerie überprüft den Aufbau der Hashwerte. Weiter werden die übertragenen Daten auf Plausibilität geprüft (z.B. negative Dateigrößen).

Für die Validierung sind folgende Regeln aufgestellt worden:

Die miradlo-Galerie ist in der Lage, sämtliche übertragenen Daten einer Synchronisation *Abbildung 3.24: Bewertung der Prüfungsansätze, Seite 74*, zu überprüfen.

Durch die exzessive Verwendung von Hashsummen in einer Vielzahl an Datenobjekten, bietet es sich an, die miradlo-Galerie grundsätzlich mit einer Selbsttestfunktion auszustatten. Diese Funktion soll die miradlo-Galerie in die Lage versetzen, ihren eigenen Datenbestand anhand der Hashsummen vollständig zu überprüfen und auftretende Fehler zu protokollieren. Da diese Funktion potenziell eine hohe Last erzeugt, sollte sie nur eingesetzt werden, wenn es der Administrator der miradlo-Galerie für notwendig hält.

Es bietet sich außerdem an, eine zeitlich planbare Selbsttestfunktion zu integrieren. Der Gedanke dabei ist, dass eine miradlo-Galerie zu einer Tageszeit mit wenig Benutzeraufkommen kleine Teile ihres Datenbestandes selbsttätig überprüft und erkannte Fehler protokolliert.

Die Berechnung der Hashsumme eines neu hinzugefügten Originalbildes erfolgt laut der bisherigen Beschreibung erstmalig in der miradlo-Galerie, *Abbildung 3.24: Bewertung der Prüfungsansätze, Seite 74*, Dieses Vorgehen hat das Problem, dass Übertragungsfehler beim Hinzufügen in die miradlo-Galerie nicht entdeckt werden können. Um dies zu vermeiden, ist dem Benutzer die Möglichkeit zu bieten, mithilfe eines kleinen Programms die Hashsumme einer zu übertragende Datei vor der eigentlichen Übertragung zu berechnen. Dies kann konkret mithilfe eines Java-Applets [31] geschehen, welches die berechnete Hashsumme, während der Übertragung, der miradlo-Galerie mitteilt. Sollte bei einer Übertragung ein Fehler auftreten, kann die miradlo-Galerie den Benutzer bitten, den Vorgang bei den betroffenen Bildern zu wiederholen.

4.20 Build Management

Mithilfe des Build Managements werden Konzepte definiert, wie die Kompatibilität zwischen verschiedenen Softwareversionen der miradlo-Galerie sichergestellt werden. Dieser Punkt wurde bereits im Konzept Migration beschrieben. Darüber hinaus ist das Build Management für die Kompatibilität zwischen verschiedenen Komponenten verantwortlich. Beispielsweise muss sichergestellt sein, welche Versionen der miradlo-Galerie mit welchen Versionen von miradlokit funktionieren.

Diesbezüglich muss vor der Umsetzung der miradlo-Galerie ein Konzept entwickelt werden, wie Versionsnummern zu gestalten und Entwicklungsprozesse gesteuert werden müssen. Damit soll sichergestellt werden, dass von vornherein klar erkennbar ist, zwischen welchen Versionen Inkompatibilitäten vorliegen.

5 Fazit

Diese Arbeit hat ergeben, dass eine Peer-To-Peer-Architektur mit einer zentralen Verwaltungsinstanz die wichtigsten Vorteile für den Betrieb des Bilderverwaltungssystems aufweist. Für die Datenspeicherung hat sich gezeigt, dass eine kombinierte Verwendung von Datenbank und Dateisystem für die Produktionsumgebung der Firma miradlo am günstigsten ist. Die weiterführende Betrachtung von ursprünglich unbekannten Problemstellungen, wie Versions- und Datenverwaltung ebnete den Weg für die Entwicklung eines effizienten Synchronisationsalgorithmuses. Der Verzicht auf die Übertragung von Binärdaten bei jeder Bildbearbeitung und die exzessive Verwendung von Hashsummen zur Identifikation von Inhalten bilden dabei die eigentliche Innovation.

Die Realisierung der miradlo-Galerie kann in mehreren Folgeprojekten stattfinden. Diese Arbeit dient dabei als Basis, da das Fundament mit der Beschreibung von Synchronisation und Persistenz gelegt wurde. Der Firma miradlo steht es vollkommen frei, die Prioritäten bei der weiteren Entwicklung zu setzen. Eine netzwerkweite Wartung vieler miradlo-Galerien, die Umsetzung der Synchronisation oder die ledigliche Realisierung der miradlo-Galerie ohne Synchronisation sind nur ein Teil der möglichen Optionen. Es sind auch Folgeprojekte auf theoretischer Basis möglich. Die Analyse der möglichen Belastungsgrenzen eines Service-Locators mit wachsender Anzahl an Anfragen bei bestimmten Implementierungsformen ist dafür ein Beispiel.

Aus meiner Sicht wäre es lohnend, das Bilderverwaltungssystem weiter zu verfolgen. Ich würde mich freuen, wenn ich Gelegenheit bekäme, dieses System umzusetzen bzw. es in Folgeprojekten zu konkretisieren.

Ich, Thomas Heidrich habe durch die intensive Arbeit an dieser Bachelorthesis gelernt, was es bedeutet, Software zu entwickeln. In früheren Semestern meines Studiums wurden zwar ebenfalls Projekte durchgeführt, jedoch war die Vielzahl der zu beachtenden Aspekte nicht einmal annähernd mit dieser Aufgabe vergleichbar. Ein solcher Qualitätsanspruch war für mich bisher unbekannt. Ohne die Hilfe der im Vorwort benannten Personen hätte ich für die Erstellung mindestens die dreifache Zeit benötigt, um dieses qualitative Niveau zu erreichen. Eine Veranstaltung im Studium, welche auf die Erstellung von wissenschaftlichen Arbeiten abzielt, hätte mir den ersten Monat der Bearbeitung erheblich erleichtert.

Ich bin nun in der Lage, eine professionelle Analyse von bestehenden Problemstellungen durchzuführen und daraus Lösungskonzepte abzuleiten. Die zu beachtenden Aspekte bei der tatsächlichen Umsetzung werde ich nun durch meine weitere Arbeit bei der Firma miradlo erlernen. Ich bin der Meinung, dass ich mich nur durch die weiterführende Arbeit bei miradlo, zu einem guten Software-Ingenieur entwickeln kann. Ich werde in einem oder in zwei Jahren evaluieren, ob der Beginn eines Masterstudiums der Informatik für mich rentabel ist. Die von Masterstudenten durchgeführten Vorlesungen während meines Studiums haben mich diesbezüglich skeptisch gemacht.

Alles in allem bin ich der Meinung, dass die Entscheidung, an der HTWG Konstanz im Studiengang Software Engineering zu studieren, eine gute Entscheidung war. Ich habe dadurch das Rüstzeug erhalten, um ein guter und erfolgreicher Ingenieur zu werden.

6 Anhang

In diesem Kapitel befinden sich alle für die Arbeit verwendeten, aber nicht primär relevanten Informationen. Im folgenden Kapitel Anforderungen stehen sämtliche, während der Entwicklung des Bilderverwaltungssystems, erfassten Anforderungen. Im *Kapitel 6.2.6, Lizenz, Seite 117,* sind die Nutzungs- und Weitergabebedingungen festgehalten. Im *Kapitel 6.2, Quellen, Seite 112,* wird auf alle verwendeten externen Werke verwiesen.

6.1 Anforderungen

Aus dem Pflichtenheft [0] und mehreren firmeninternen Besprechungen wurden Anforderungen an das Bilderverwaltungssystem extrahiert. "Anforderungen (Requirements) legen Eigenschaften, Qualitätsmerkmale des zukünftigen Systems fest. In der Regel werden die Anforderungen in zwei Hauptgruppen unterteilt: funktionale und nicht-funktionale Anforderungen" [32]

Die in diesem Dokument enthaltenen Diagramme orientieren sich an der Modellierungssprache UML. Durch "UML werden Begrifflichkeiten und Diagrammdarstellungen der objektorientierten Analyse und des Designs vereinheitlicht". [9] Um die Verständlichkeit der Diagramme zu erhöhen wurden teilweise verschiedene Diagrammtypen miteinander kombiniert und nicht-standardkonforme Elemente eingesetzt.

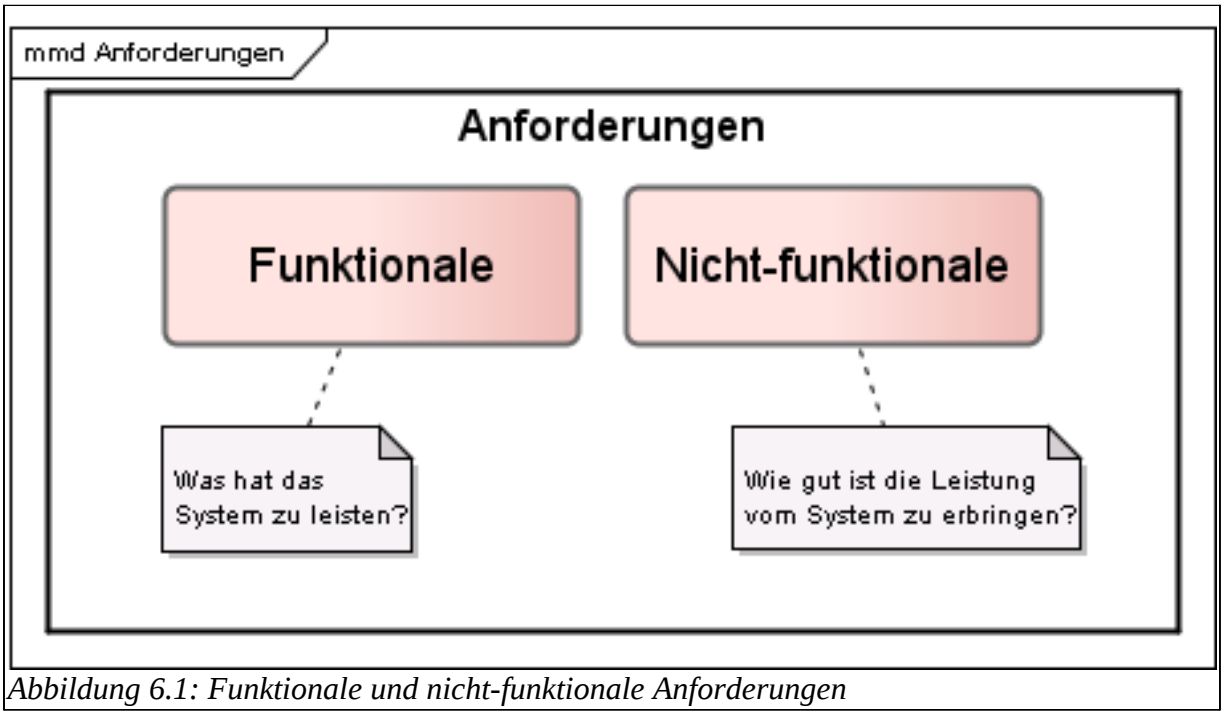

Abbildung 6.1: Funktionale und nicht-funktionale Anforderungen

Im Diagramm ist visualisiert, dass funktionale Anforderungen grundsätzlich beschreiben, was ein System zu leisten hat. In der Regel umfasst dies, was mit welchen Daten geschehen soll. Meist wird dabei noch angegeben, welche Arten von Benutzern dabei involviert sind.

Nicht-funktionale Anforderungen beschreiben hingegen, wie gut das System arbeiten muss, um

grundsätzlich einsetzbar zu sein. An dieser Stelle werden meist zu erreichende Mindestkriterien angegeben.

Bei der Formulierung der Anforderungen wurde stets darauf geachtet, dass neben den Anforderungen klare Abnahmekriterien vorhanden sind. Dies ist darin begründet, dass nur eindeutig nachprüfbare Anforderungen bei der Umsetzung korrekt berücksichtigt werden können. Weiter kann dadurch die Erfüllung dieser Kriterien dem Auftraggeber bei der Übergabe des Systems nachgewiesen werden. Das folgende Diagramm visualisiert, welche Anforderungen extrahiert wurden.

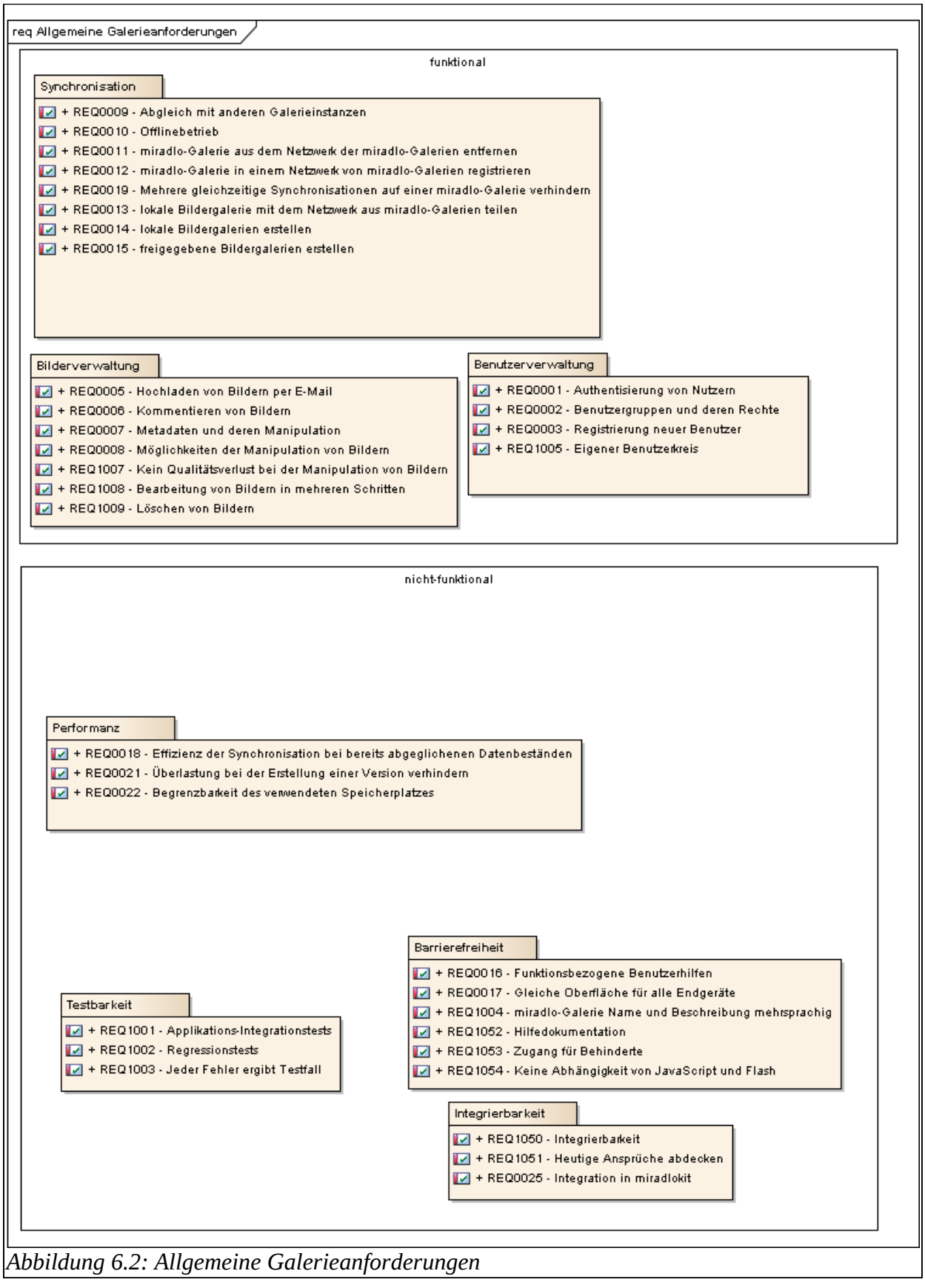

Abbildung 6.2: Allgemeine Galerieanforderungen

6.1.1 Barrierefreiheit

Die Firma miradlo erstellt Applikationen die auch von Menschen mit Behinderungen eingesetzt werden können. Für die miradlo-Galerie ergeben sich diesbezüglich diverse Anforderungen:

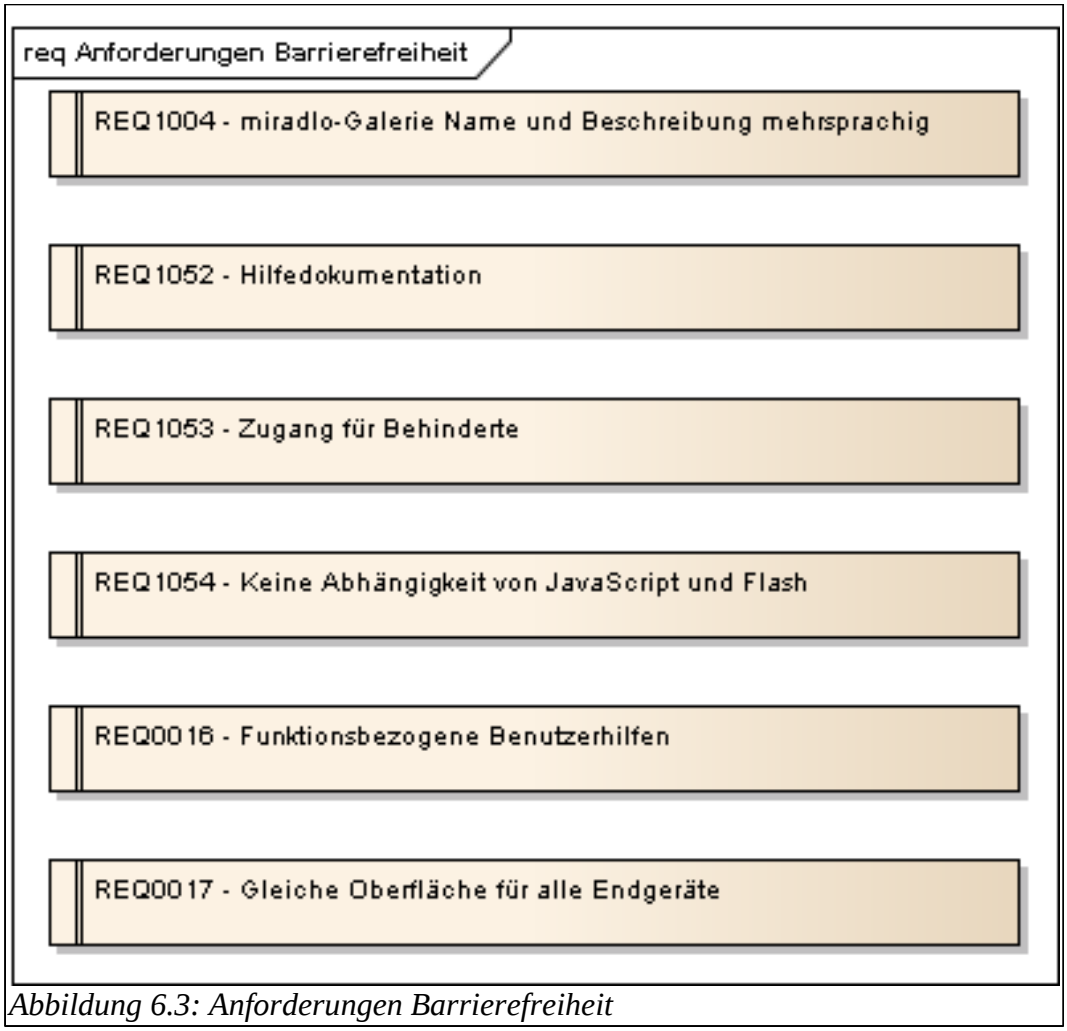

Abbildung 6.3: Anforderungen Barrierefreiheit

Diagrammelement	Beschreibung
REQ0016 - Funktionsbezogene Benutzerhilfen	Der Benutzer muss für jede Funktion eine selbsterklärende Hilfedokumentation einsehen können, damit die Verwendung des Systems selbst erlernt werden kann. **Problem** Besonders Benutzer, die das System erst kurz benutzen, benötigen gelegentlich Hilfestellung bei der Bedienung **Quelle** miradlo **Abnahmekriterium** Jede Funktion des Systems ist mit einer Hilfedokumentation versehen und kann ohne den Bedienungskontext zu verlassen, konsultiert werden.

Diagrammelement	Beschreibung
REQ0017 - Gleiche Oberfläche für alle Endgeräte	Auf allen relevanten Endgeräten ist eine Benutzeroberfläche anzubieten, die das gleiche Look-And-Feel aufweist, damit die Benutzer die miradlo-Galerie immer wiedererkennen können. Zusatz: Falls technisch möglich soll überall die gleiche Funktionalität verfügbar sein, damit sich Benutzer nicht umstellen müssen. **Problem** Benutzer können durch unterschiedlich geartete Benutzeroberflächen auf verschiedenen Endgeräten verwirrt werden. **Quelle** miradlo **Abnahmekriterium** Auf sämtlichen Endgeräten werden nahezu identische Benutzeroberflächen angeboten. Zusatz: Es werden auf allen Endgeräten die gleichen Funktionalitäten angeboten.
REQ1004 - miradlo-Galerie Name und Beschreibung mehrsprachig	Texte, wie Name und Beschreibung einer miradlo-Galerie müssen von einem Benutzer in mehreren Sprachen eingegeben werden können, damit die miradlo-Galerie in einem internationalen Umfeld eingesetzt werden kann und keine Ausgrenzung von Menschen stattfindet. **Problem** Durch das Internet ist es möglich, miradlo-Galerien von jedem Punkt der Erde zu bedienen. **Quelle** miradlo **Abnahmekriterium** Sämtliche textuellen Inhalte können in verschiedenen Sprachen abgerufen werden.
REQ1052 - Hilfedokumentation	Der Benutzer soll einfach auf Hilfedokumentation zugreifen können, damit eine schnelle Einarbeitung ins System ermöglicht wird. **Problem** Umfangreiche Benutzerdokumentationen ohne direkten Kontextbezug sind aufwändig zu durchsuchen. **Quelle** miradlo **Abnahmekriterium** Benutzer sind in der Lage mit weniger als zwei Eingaben eine kontextbezogene Hilfe zu erhalten.
REQ1053 - Zugang für Behinderte	Menschen mit Sehschwäche oder motorischen Schwierigkeiten sollen die miradlo-Galerie bedienen können, damit diese Personengruppe nicht ausgegrenzt wird. Weiter hilft diese Anforderung, mobile Endgeräte einfacher an die Galerie anzubinden. **Problem**

Diagrammelement	Beschreibung
	Es existieren Menschen mit eingeschränkten körperlichen Fähigkeiten. **Quelle** miradlo **Abnahmekriterium** Es existieren einfach zu bedienende Möglichkeiten, des Bildschirminhalt stufenlos zu vergrößern. Weiter wird die Möglichkeit angeboten, die miradlo-Galerie ausschließlich über die Tastatur zu bedienen.
REQ1054 - Keine Abhängigkeit von JavaScript und Flash	Der Benutzer muss die miradlo-Galerie ohne JavaScript und Flash [33] verwenden können, damit spezielle Eingabegeräte, wie Tastaturen für Blinde verwendet werden können. Weiterhin hilft diese Anforderung, die miradlo-Galerie auch in Firmen einzusetzen, bei denen JavaScript aus Sicherheitsgründen nicht erlaubt ist. **Problem** Nicht jeder Benutzer ist in der Lage Systeme zu verwenden, die auf JavaScript und Flash zurückgreifen. **Quelle** miradlo **Abnahmekriterium** Die Benutzung der miradlo-Galerie kann problemlos ohne die Unterstützung von JavaScript oder Flash erfolgen.

6.1.2 Benutzerverwaltung

Um ein von Menschen gesteuertes Verteiltes Softwaresystem außerhalb einer Testumgebung einsetzen zu können, ist eine Benutzerverwaltung zwingend von Nöten. Nur so kann gewährleistet werden, dass ausschließlich berechtigte Zugriffe stattfinden.

Das konkrete Design und die Umsetzung dieser Anforderungen liegt nicht im Rahmen dieser Bachelorarbeit. Diese Anforderungen wurden aus dem Pflichtenheft [34] abgeleitet, siehe folgende Abbildung.

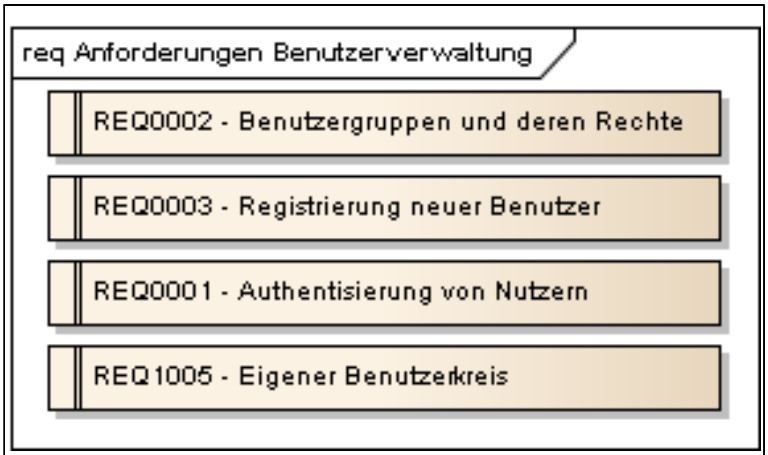

Abbildung 6.4: Anforderungen Benutzerverwaltung

Diagrammelement	Beschreibung
REQ0001 - Authentisierung von Nutzern	An der miradlo-Galerie registrierte Benutzer müssen sich mit ihrer Benutzerkennung und ihrem Passwort an der miradlo-Galerie anmelden können. Damit kann sichergestellt werden, dass kein unbefugter Zugriff auf die Daten erfolgt. **Problem** Es muss vom System unterschieden werden können, ob es gerade von einem bekannten Benutzer oder einem Besucher bedient wird. Besucher dürfen nur in klar definierten Grenzen auf das System zugreifen. **Quelle** miradlo **Abnahmekriterium** Es müssen sich unterschiedliche Benutzer am System an- und abmelden können. Im angemeldeten Zustand wird der jeweilige Benutzer mit definierten Rechte ausgestattet, die er im abgemeldeten Zustand nicht besitzt.
REQ0002 - Benutzergruppen und deren Rechte	Im Umfeld der miradlo-Galerie sollen die Benutzer unterschiedliche Berechtigungen erhalten, damit jeder Benutzer nur die Funktionen verwenden kann, die für die jeweilige Arbeit notwendig sind. **Problem** Nicht jeder am System registrierte Benutzer darf vollständigen Zugriff auf alle Funktionen erhalten, da sonst Fehleingaben und Vandalismus verstärkt auftreten können. **Quelle** miradlo **Abnahmekriterium** Es müssen unterschiedliche Benutzer angelegt werden können, die Mitglied verschiedener Benutzergruppen sind. Jede Benutzergruppe muss die definierten Rechte besitzen und eine Mitgliedschaft in mehreren Gruppen muss möglich sein. Die Rechte müssen sich in einer solchen Situation entsprechend erweitern. Die folgenden Benutzergruppen sind mindestens zu unterstützen. •Administrator - Installieren und Einrichten der miradlo-Galerie, Verwalten von Bildergalerien, deren Metadaten und Benutzer •Redakteur - Hochladen und Bearbeiten von Bildern und deren Metadaten •Besucher - Betrachten und Kommentieren von Bildern, gesamten Bildergalerien und Einsehen der Metadaten
REQ0003 - Registrierung neuer Benutzer	Der Administrator einer miradlo-Galerie soll festlegen können ob sich weitere Benutzer an dieser miradlo-Galerie registrieren dürfen. Damit wird sichergestellt, dass der Administrator die Entscheidung hat, wer mit dem System arbeitet. **Problem** Es muss möglich sein, dass sich keine weiteren Benutzer am System registrieren dürfen.

Diagrammelement	Beschreibung
	Quelle miradlo **Abnahmekriterium** Die geschilderte Funktionalität muss mit fünf Benutzern im System und einem, der sich neu registriert, demonstriert werden.
REQ1005 - Eigener Benutzerkreis	Da die miradlo-Galerie ein Plugin für miradlokit ist, sollen eigene Benutzergruppen für die miradlo-Galerie vergeben werden. **Problem** Die Verwendung von Standardbenutzergruppen in miradlokit durch Plugins kann unvorhergesehene Abhängigkeiten zwischen einzelnen Plugins erzeugen. **Quelle** miradlo **Abnahmekriterium** Die miradlo-Galerie verwendet ausschließlich eigene Benutzergruppen.

6.1.3 Bilderverwaltung

Die Bilderverwaltung stellt den offensichtlichsten Teil der miradlo-Galerie dar. Grundlegende Funktionen, wie das Hochladen und Betrachten von Bildern, sind der eigentliche Zweck des Systems und somit unabdingbar. Das konkrete Design und die Umsetzung dieser Anforderungen liegt nicht im Rahmen dieser Bachelorarbeit, die bereits bekannten Anforderungen zeigt die folgende Abbildung.

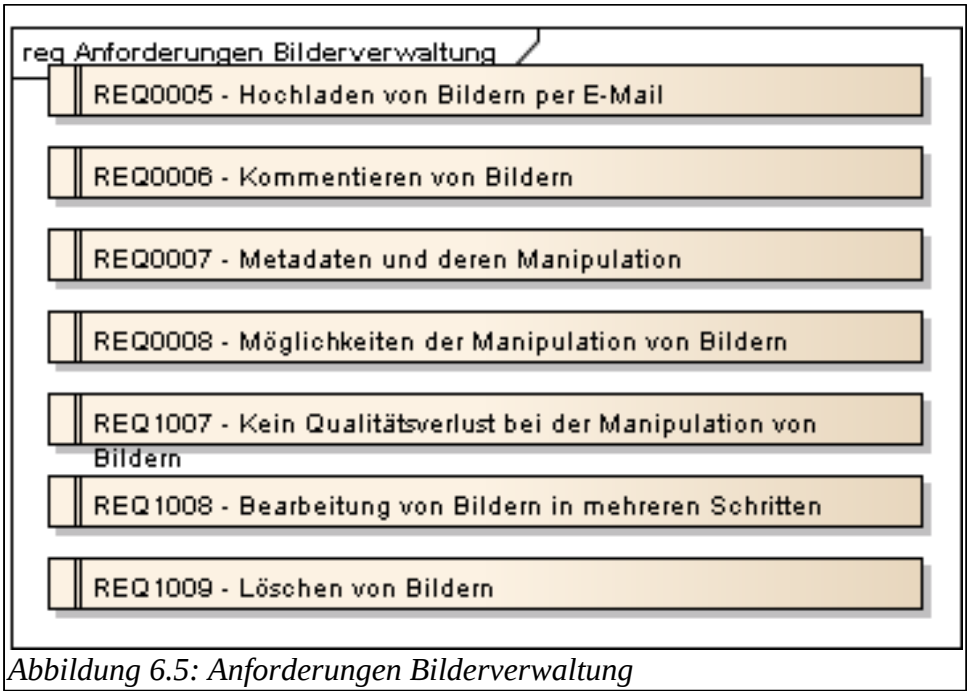

Abbildung 6.5: Anforderungen Bilderverwaltung

Diagrammelement	Beschreibung
REQ0005 - Hochladen von Bildern per E-Mail	Benutzer sollen per E-Mail Bilder in die miradlo-Galerie einstellen können, damit z.B. Schnappschüsse auf einem iPhone sofort in der miradlo-Galerie mitverwendet werden können. **Problem** Bei Endgeräten mit kleinen Bildschirmen stellt die Arbeit mit umfangreichen Eingabemasken ein Hindernis dar. **Quelle** miradlo **Abnahmekriterium** Die Form, in der die E-Mail an das System gesendet werden muss, ist selbsterklärend. Das System muss per E-Mail eine Rückmeldung geben, ob die Eingabe korrekt bearbeitet wurde. Weiter muss das System mit dem Text- und HTML-Format von E-Mails zurecht kommen. Bei dieser Funktion müssen neben dem Bild selbst mindestens die folgenden Metadaten gesetzt werden können. •Name der Bildergalerie, in die das Bild eingefügt werden soll •Titel des Bildes •Beschreibung des Bildes Es muss eine Möglichkeit geschaffen werden, mit der sich ein Benutzer per E-Mail authentisieren kann. Die Galerie muss mit mindestens 500 Spammails pro Stunde zurecht kommen. Die Galerie darf davon nicht beeinträchtigt werden.

Diagrammelement	Beschreibung
REQ0006 - Kommentieren von Bildern	Benutzer sollen einzelne Bilder kommentieren können, damit die Standardfunktionalität von modernen Webseiten abgedeckt werden kann. **Problem** - **Quelle** miradlo **Abnahmekriterium** Es muss möglich sein, dass verschiedene Benutzer, einem Bild Kommentare geben. Der Ersteller und der Erstellungszeitpunkt des Kommentars muss dabei ausgezeichnet werden. Die einzelnen Kommentare dürfen nicht veränderbar sein. Einzelne Kommentare müssen gelöscht werden können.
REQ0007 - Metadaten und deren Manipulation	Benutzer müssen den Bildern und den Bildergalerien beschreibende Metadaten geben können, damit eine sinnvolle Suchanfrage abgesetzt werden kann. **Problem** Ein Bild einer Bildergalerie besteht nicht nur aus dem Bild selbst, sondern beinhaltet auch noch zusätzliche Informationen, um eine textbasierte Suche zu ermöglichen. Bilder müssen die folgenden Felder für Metadaten besitzen. •Name (muss) •Beschreibung (kann) •Breite (muss) •Höhe (muss) Bildergalerien besitzen die Metadaten •Name •Beschreibung Diese müssen auch nach der Erstellung bearbeitet werden können. **Quelle** miradlo **Abnahmekriterium** Alle spezifizierten Datenfelder müssen bearbeitet werden können.
REQ0008 - Möglichkeiten der Manipulation von Bildern	Hochgeladene Bilder müssen vom Benutzer verändert werden können, damit die Bilder für die Bedürfnisse der Benutzer entsprechend angepasst werden können. **Problem** Nicht jeder Benutzer ist in der Lage, Bildbearbeitungsprogramme auf seinem Endgerät auszuführen. Die Galerie muss es ermöglichen, Bilder einer Bildergalerie zu bearbeiten. Funktionen wie •rotieren (90° links, 90° rechts, 180°) •zuschneiden •skalieren müssen mindestens unterstützt werden. Es ist darauf zu achten, dass neue Bearbeitungsfunktionen ohne fundamentale Inkompatibilitäten zu bisherigen Versionen der miradlo-Galerie hinzugefügt werden

Diagrammelement	Beschreibung
	können. **Quelle** miradlo **Abnahmekriterium** Es muss möglich sein Bilder zu rotieren, zuzuschneiden und zu skalieren.
REQ1007 - Kein Qualitätsverlust bei der Manipulation von Bildern	Ein Benutzer muss Bilder mehrfach manipulieren können, ohne einen Qualitätsverlust der Bilder zu erhalten. Damit soll sichergestellt werden, dass der Benutzer mehrere Versuche durchführen kann bevor ein Bild seinen Bedürfnissen entspricht. **Problem** Die Kompressionsverfahren heutiger Grafikformate können sowohl verlustfrei als auch verlustbehaftet sein. Eine stetige Verschlechterung der Bildqualität bei mehrfachen Bearbeitungsvorgängen würde die Bilder in der miradlo-Galerie langfristig gesehen unbenutzbar machen. **Quelle** miradlo **Abnahmekriterium** Bei einer verlustbehafteten Kompression darf der Komprimierungsalgorithmus innerhalb der miradlo-Galerie höchstens einmal auf ein Bild angewendet werden.
REQ1008 - Bearbeitung von Bildern in mehreren Schritten	Ein Benutzer muss Bilder in mehreren Teilschritten bearbeiten können. Beispielsweise könnte der Benutzer ein Bild ausschneiden, skalieren und drehen wollen. Damit muss ein Benutzer die Bilder seinen Bedürfnissen entsprechend anpassen können. **Problem** Meist ist es nicht möglich, das gewünschte Bearbeitungsresultat in einem einzigen Bearbeitungsschritt zu erreichen. Beispielsweise könnte es verlangt sein, ein Bild um 90° im Uhrzeigersinn zu rotieren, es auf den wesentlichen Bildinhalt zuzuschneiden und es dann auf 50% zu skalieren. **Quelle** miradlo **Abnahmekriterium** Die miradlo-Galerie ist in der Lage, eine zu definierende Anzahl an Bearbeitungsvorgängen auf ein Bild anzuwenden. Dies muss exemplarisch mit fünf Bearbeitungsvorgängen demonstriert werden.
REQ1009 - Löschen von Bildern	Ein Bild muss mindestens vom Administrator aus dem Gesamtsystem heraus gelöscht werden können, damit sichergestellt werden kann, dass keine rechtlich bedenklichen Bilder auf irgendeiner miradlo-Galerie weiter existieren. **Problem** Sollte ein Bild gegen geltendes Recht verstoßen, muss es aus dem System entfernt werden können. Änderungen können im rechtlichen Rahmen laufend auftreten. **Quelle**

Diagrammelement	Beschreibung
	miradlo **Abnahmekriterium** In einem Netzwerk aus miradlo-Galerien ist es möglich, ein Bild einer freigegebenen Bildergalerie aus allen relevanten miradlo-Galerien zu entfernen. Ein erneutes Hinzufügen des Bildes muss ebenfalls möglich sein.

6.1.4 Integrierbarkeit

Im Pflichtenheft [34] werden Anforderungen an die Integrierbarkeit der Galerie in bestehende Webauftritte definiert, siehe Abbildung:

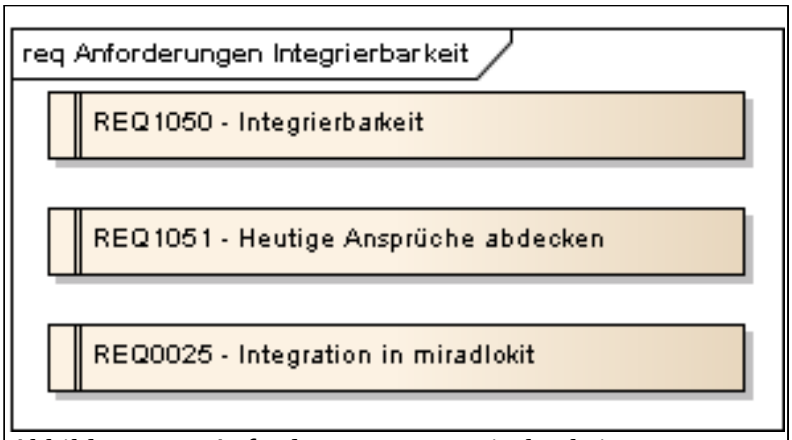

Abbildung 6.6: Anforderungen Integrierbarkeit

Diagrammelement	Beschreibung
REQ0025 - Integration in miradlokit	Die miradlo-Galerie muss als Plugin für miradlokit umgesetzt werden, damit eine einfache Integration in Kundenprojekten der Firma miradlo möglich ist. **Problem** - **Quelle** miradlo **Abnahmekriterium** Ein Kundenprojekt, welches auf der aktuellen Version von miradlokit basiert, wird um die miradlo-Galerie erweitert.
REQ1050 - Integrierbarkeit	Ein Mitarbeiter der Firma miradlo soll die miradlo-Galerie einfach in bestehende Webauftritte integrieren können, damit diese dort verwendet werden kann. **Problem** Anforderungen von Kunden können sich laufend ändern. Eine schnelle Anpassungsfähigkeit von Webauftritten ist wichtig. **Quelle** miradlo **Abnahmekriterium** Ein bestehender Webauftritt, der die aktuelle Softwareversion von miradlokit einsetzt, kann innerhalb von 30 Minuten von einem geschulten Mitarbeiter von miradlo erweitert werden.
REQ1051 - Heutige Ansprüche abdecken	Die miradlo-Galerie muss den heutigen Standard-Ansprüchen an Webseiten genügen, damit das System von den Benutzern akzeptiert werden kann. **Problem** Ohne die Erfüllung aktueller, technischer Voraussetzungen und Funktionalitäten kann die Akzeptanz des Systems in Gefahr sein. **Quelle** miradlo **Abnahmekriterium** Das System ist flexibel erweiterbar und ermöglicht einen einfachen, standardisierten Austausch zwischen weiteren Systemen, darüberhinaus unterstützt es verschiedene Darstellungsmöglichkeiten. Beispiel für eine aktuelle Anforderung an zeitgemäße Funktionalität ist eine einfache Bildersuche.

6.1.5 Synchronisation

Die Synchronisation zwischen verschiedenen Instanzen der miradlo-Galerie ist der Fokus dieser Bachelorarbeit und stellt die eigentliche Innovation dar. Die Anforderungen an die Synchronisation zeigt die folgende Abbildung.

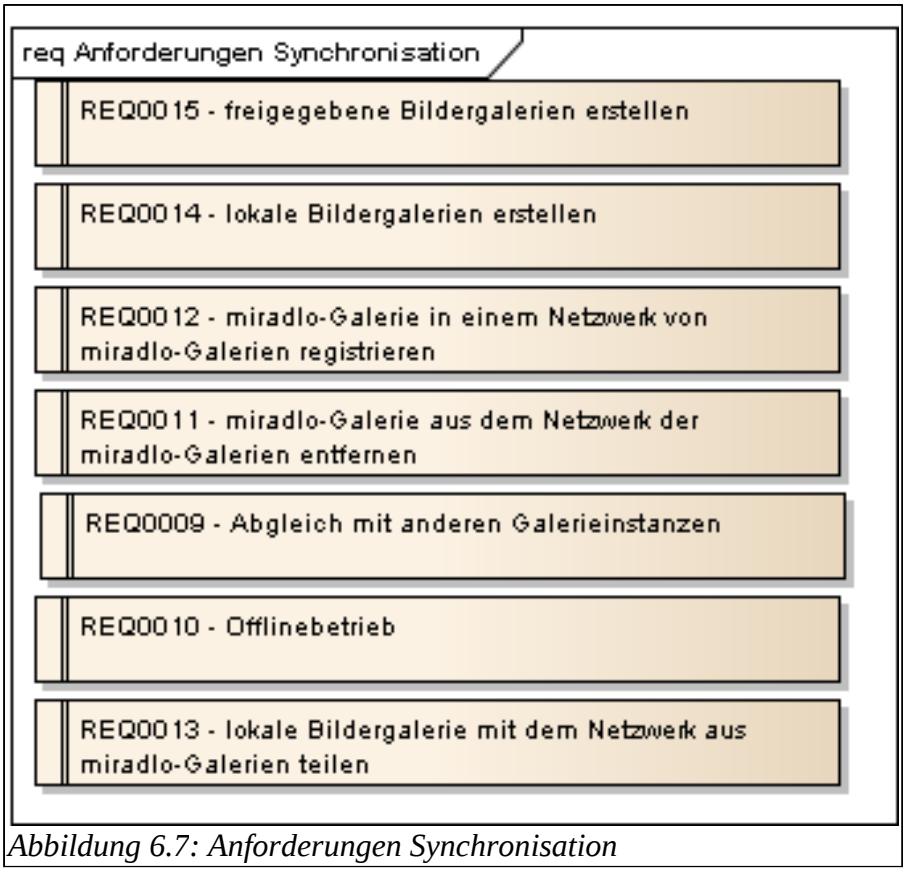

Abbildung 6.7: Anforderungen Synchronisation

Diagrammelement	Beschreibung
REQ0009 - Abgleich mit anderen Galerieinstanzen	Die miradlo-Galerie muss in der Lage sein, ihren Datenbestand mit anderen miradlo-Galerien abzugleichen. Dies umfasst freigegebene Bildergalerien inklusive aller Bilder und Metadaten. Dabei ist zu beachten, dass verschiedene Versionen für Benutzer klar ausgezeichnet sind. Dieser Abgleich ist notwendig damit Bilder und Metadaten von allen in einem Netzwerk befindlichen miradlo-Galerien mitverwendet werden können. Benutzer müssen in der Lage sein, die Verbreitung von Bildern in einer freigegebenen Bildergalerie zu unterbinden, bis diese explizit freigegeben wurden. **Problem** Verschiedene miradlo-Galerien müssen sich automatisiert abgleichen können. **Quelle** Thomas Heidrich **Abnahmekriterium** Mindestens fünf miradlo-Galerien müssen sich automatisiert

Diagrammelement	Beschreibung
	abgleichen können. Dabei muss jede miradlo-Galerie mindestens zehn Bildergalerien mit jeweils 100 Bildern besitzen. Jede miradlo-Galerie muss mindestens zwei Bildergalerien zweier anderer miradlo-Galerien verwenden. Eine Änderung an einem Bild muss bei allen anderen miradlo-Galerien ankommen.
REQ0010 - Offlinebetrieb	Ein Benutzer soll ohne Internetzugang die Funktionalität der miradlo-Galerie nutzen können, damit beispielsweise auf Reisen neue Bilder in das System eingestellt werden können. Alle weiteren Funktionen außer dem dem Abgleich mit anderen Galerien und Applikationen müssen auch offline nutzbar sein. **Problem** Es besteht die Möglichkeit, dass miradlo-Galerien ihre Kommunikationsverbindung zum Netzwerk aus miradlo-Galerien für gewisse Zeit verlieren. **Quelle** miradlo **Abnahmebedingung** Es muss möglich sein, dass eine miradlo-Galerie, nachdem ihre zuvor verlorene Kommunikationsverbindung wiederhergestellt wurde, sich automatisch mithilfe der entfernten miradlo-Galerien auf den aktuellen Stand bringt.
REQ0011 - miradlo-Galerie aus dem Netzwerk der miradlo-Galerien entfernen	Ein Administrator muss eine miradlo-Galerie aus einem bestehenden Netzwerk aus miradlo-Galerien dauerhaft entfernen können. **Problem** - **Quelle** Thomas Heidrich **Abnahmekriterium** Die freigegebenen Bildergalerien der relevanten miradlo-Galerien sind im Netzwerk aus miradlo-Galerien nicht mehr verfügbar.
REQ0012 - miradlo-Galerie in einem Netzwerk von miradlo-Galerien registrieren	Ein Administrator muss in der Lage sein, eine miradlo-Galerie in einem Netzwerk aus miradlo-Galerien anzumelden. Die Funktionalität des Netzwerks aus miradlo-Galerien darf dabei nicht beeinträchtigt werden. **Problem** - **Quelle** Thomas Heidrich **Abnahmekriterium** Einem bestehenden Netzwerk aus miradlo-Galerien müssen neue miradlo-Galerien hinzugefügt werden können.
REQ0019 - Mehrere gleichzeitige Synchronisationen auf einer miradlo-Galerie verhindern	Eine Synchronisation darf vom System nicht angestoßen werden, wenn bereits eine im Gange ist. **Problem** Mehrere parallele Synchronisationsprozesse belasten die Netzwerkverbindung erheblich.

Diagrammelement	Beschreibung
	Quelle miradlo **Abnahmebedingung** Zu jedem Zeitpunkt findet auf einer miradlo-Galerie ausschließlich eine Synchronisation statt.
REQ0013 - lokale Bildergalerie mit dem Netzwerk aus miradlo-Galerien teilen	Ein Administrator muss eine lokale Bildergalerie in eine freigegebene Bildergalerie umwandeln können, damit die Bilder von anderen Personen mitverwendet werden können. **Problem** Es besteht die Möglichkeit, dass Benutzer eine Bildergalerie erst komplett aufbauen wollen, bevor sie deren Inhalte mit anderen teilen. **Quelle** Thomas Heidrich **Abnahmekriterium** Eine lokale Bildergalerie wird in eine freigegebene Bildergalerie umgewandelt und von einer anderen miradlo-Galerie benutzt. Alle Bilder und Metadaten sind daraufhin auf beiden miradlo-Galerien verfügbar.
REQ0014 - lokale Bildergalerien erstellen	Ein Administrator muss Bildergalerien erstellen können, die nur auf der aktuell verwendeten miradlo-Galerie vorkommen. **Problem** Es gibt Benutzer, die an den Synchronisationsfähigkeiten der miradlo-Galerie nicht interessiert sind. **Quelle** Thomas Heidrich **Abnahmekriterium** Eine lokale Bildergalerie muss für die entfernten miradlo-Galerien eines Netzwerks aus miradlo-Galerien vollständig verborgen sein.
REQ0015 - freigegebene Bildergalerien erstellen	Ein Administrator muss Bildergalerien so anlegen können, damit diese mit anderen miradlo-Galerien im Netzwerk aus miradlo-Galerien synchronisiert werden können. **Problem** Benutzer möchten Bilder mit anderen miradlo-Galerien teilen. **Quelle** miradlo **Abnahmekriterium** Auf einer miradlo-Galerie wird eine freigegebene Bildergalerie erstellt, die von anderen miradlo-Galerien im Netzwerk aus miradlo-Galerien genutzt wird. Nach der Synchronisation ist die Bildergalerie auf der weiteren miradlo-Galerie vorhanden.

6.1.6 Testbarkeit

Bei der miradlo-Galerie handelt es sich um ein Verteiltes System. Die einzelnen Komponenten, sowie das Zusammenspiel zwischen den miradlo-Galerien müssen entsprechend automatisch getestet werden können, siehe folgende Anforderungen:

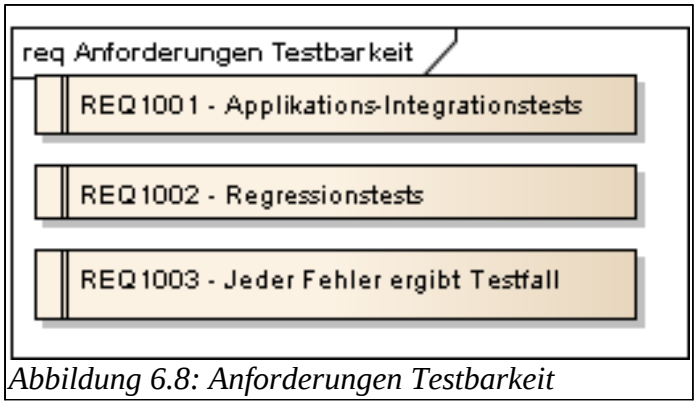

Abbildung 6.8: Anforderungen Testbarkeit

Diagrammelement	Beschreibung
REQ1001 - Applikations-Integrationstests	Die von einem Tester durchlaufenen Abläufe innerhalb der miradlo-Galerie müssen über ein Testwerkzeug wiederholbar durchlaufen werden können. Damit soll erreicht werden, dass Fehler vor der Inproduktionsnahme erkannt werden. Für die Testautomatisierung ist bei miradlo Selenium [SELE] einzusetzen. **Problem** Tests müssen reproduzierbar sein. **Quelle** miradlo **Abnahmekriterien** Die Testabläufe sind verfügbar und mit dem Quellcode der miradlo-Galerie versioniert. Die Tests sind lauffähig.
REQ1002 - Regressionstests	Jeder Entwicklungsschritt muss mit den zuvor erstellten Testabläufen von einem Administrator erfolgreich durchlaufen werden. **Problem** Damit soll erreicht werden, dass schon erkanntes Fehlverhalten alter Versionen nicht wieder auftritt. **Quelle** miradlo **Abnahmekriterium** Entwicklungsschritte werden durch Regressionstests geprüft.

Diagrammelement	Beschreibung
REQ1003 - Jeder Fehler ergibt Testfall	Jeder erkannte Fehler muss von einem Entwickler mit einem Testfall nachgestellt werden. Ist die Korrektur erfolgt, muss der Testfall überprüfen, ob der Fehler tatsächlich nicht mehr auftritt. Der neu generierte Testfall muss den Regressionstests hinzugefügt werden. **Problem** Die Softwarequalität muss mit repräsentativen Tests jederzeit überprüfbar sein. **Quelle** miradlo **Abnahmekriterium** Es soll geprüft werden können, ob alte Fehler wieder auftreten.

6.1.7 Performanz

Die Leistungsfähigkeit der miradlo-Galerie ist für die Akzeptanz der Benutzer von entscheidender Bedeutung, daraus ergeben sich folgende Anforderungen:

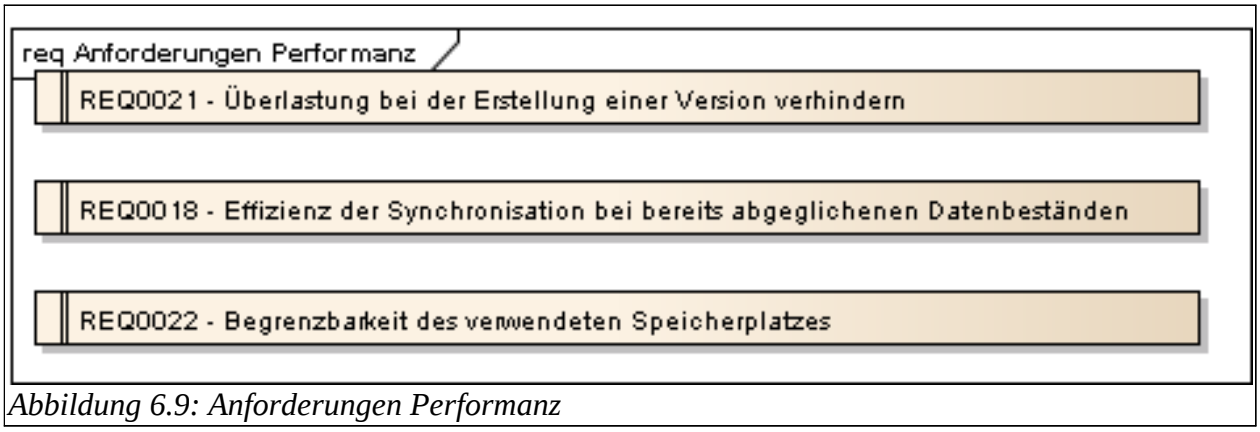

Abbildung 6.9: Anforderungen Performanz

Diagrammelement	Beschreibung
REQ0018 - Effizienz der Synchronisation bei bereits abgeglichenen Datenbeständen	Es muss ein ressourcenschonender Synchronisationsalgorithmus eingesetzt werden, welcher die Belastung des Netzwerks gering hält. **Problem** Es sind nur begrenzte Netzwerkressourcen verfügbar. **Quelle** miradlo **Abnahmekriterium** Bei einer Synchronisation zweier bereits abgeglichener Bildergalerien, finden keine unnötigen Übertragungen von Daten statt.
REQ0021 - Überlastung bei der Erstellung einer Version verhindern	Ein Benutzer soll ungehindert arbeiten können, auch wenn neue Versionen eines Bildes erstellt werden müssen. Damit soll erreicht werden, dass der Benutzer ohne Leistungseinbußen arbeiten kann. **Problem** Besonders große Bilder erzeugen bei der Erstellung eine hohe Systembelastung. Es muss verhindert werden, dass die Erstellung einer Version die vorhandenen Systemressourcen völlig ausschöpft. Es muss möglich sein, auch bei leistungsschwachen Computern wie z.B. einem Asus Eee 900 [EEE9] Bildversionen zu erstellen, ohne dabei das Antwortverhalten des Systems erheblich zu beeinträchtigen. **Quelle** miradlo **Abnahmekriterium** Es muss einstellbar sein, wie viele Versionen einer miradlo-Galerie gleichzeitig erstellt werden dürfen.
REQ0022 - Begrenzbarkeit des verwendeten Speicherplatzes	Ein Administrator soll die Größe der Datenbank- und der Dateisystemverwendung bei jeder miradlo-Galerie individuell einstellen können. **Problem** Prinzipiell wird die miradlo-Galerie auf verschiedenartigen Rechnern eingesetzt. **Quelle** miradlo **Abnahmekriterium** Eine miradlo-Galerie kann z.B. 12 GByte Speicherplatz für ihre Galeriedaten reservieren. Eine andere miradlo-Galerie kann 1.000 GByte reservieren. Das System muss dafür Sorge tragen, dass die eingestellte Grenze nicht überschritten wird. Es muss nach vernünftigen Abgleichsmethoden gesucht werden, die es erlauben, auch bei einem 12 GByte Rechner die notwendigen Daten vorzuhalten.

6.2 Quellen

Zur Erstellung dieses Dokuments wurden unterschiedliche externe Quellen verwendet. In diesem Kapitel wird auf die genaue Herkunft und die Schöpfer dieser externen Werke verwiesen.

6.2.1 arc42

Im *Kapitel 4, Technische Konzepte, Seite 76,* wurde das arc42-Template 4.0 [ARC42] zurückgegriffen. Das Template (Vorlage) ist ein Hilfsmittel für Softwarearchitekten, um bei der Beschreibung von Systemen alle relevanten Aspekte zu beleuchten und keinen unbeabsichtigt zu vernachlässigen. Es findet eine direkte Anlehnung an *Technische Konzepte* [A42TK] aus statt.

6.2.2 Literaturverzeichnis

In dieser Arbeit wurden gelegentlich Literaturverweise vorgenommen. Die im folgenden aufgezählten Dokumente und Werke wurden hierbei zurate gezogen.

- [0] Baldenhofer, Roland und Hauth, Ute: Pflichtenheft Bachelorthesis - Galerie Synchronisation, miradlo, PDF-Dokument, 03.09.2010
- [1] verschiedene Autoren: Geschäftsprozess - Artikel aus Wikipedia, der freien Enzyklopädie, Webseite in der Version vom 09.03.2011, http://de.wikipedia.org/w/index.php?title=Gesch%C3%A4ftsprozess&oldid=86231001
- [2] Victor, Frank, Prof. Dr.-Ing. habil. Dr. rer. nat.: Taschenbuch der Informatik [TDI], Seite 219 Kapitel 8, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Abkürzung "PHP"; [PHP]
- [3] Müller, Heinz, Dr.: Taschenbuch der Informatik [TDI], Seite 609 Kapitel 20, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Blog";
- [4] Victor, Frank, Prof. Dr.-Ing. habil. Dr. rer. nat.: Taschenbuch der Informatik [TDI], Seite 199 Kapitel 8, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Interpreter";
- [5] Disterer, Georg, Prof. Dr., Fels, Friedrich, Prof. Dr., Hausotter, Andreas, Prof. Dr.: Taschenbuch der Informatik [TDI], Seite 696 Kapitel 24, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Anwendungssystem";
- [6] Krauß, Ludwig, Prof. Dr.-Ing.: Taschenbuch der Informatik [TDI], Seite 149 Kapitel 5, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Begriffe "Flash-Speicherkarten und Memory-Sticks";
- [7] Müller, Heinz, Dr.: Taschenbuch der Informatik [TDI], Seite 614 Kapitel 20, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "E-Mail";
- [8] Henning, Peter A., Prof. Dr. rer. nat. habil.: Taschenbuch der Informatik [TDI], Seite 391 Kapitel 12, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Web-Browser";
- [9] Spielmann, Heinz-Jürgen, Prof. Dr. agr.: Taschenbuch der Informatik [TDI], Seite 239 Kapitel 9, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Abkürzung "UML";
- [10] Henning, Peter A., Prof. Dr. rer. nat. habil.: Taschenbuch der Informatik [TDI], Seite 390 Kapitel 12, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Web-Server";
- [11] Spielmann, Heinz-Jürgen, Prof. Dr. agr.: Taschenbuch der Informatik [TDI], Seite 235 Kapitel 9, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Datenflussdiagramm";
- [12] Oestereich, Bernd, Dipl.-Ing., Bremer, Stefan, Dipl. Wirt.-Inf.: Analyse und Design mit UML 2.3: Objektorientierte Softwareentwicklung, Seite 67 Kapitel 2, 9. Auflage, 2009, Oldenburg Wissenschaftsverlag GmbH, München. Erklärung des Begriffs "Persistenz";
- [13] Staas, Dieter, Dr. phil.: Taschenbuch der Informatik [TDI], Seite 693 Kapitel 23, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Abkürzung "JPEG";
- [14] Victor, Frank, Prof. Dr.-Ing. habil. Dr. rer. nat.: Taschenbuch der Informatik [TDI], Seite 216 Kapitel 8, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Abkürzung "SQL";

- [15] Schneider, Uwe, Prof. Dr.-Ing.: Taschenbuch der Informatik [TDI], Seite 288 Kapitel 10, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Dateisystem";
- [16] Schneider, Uwe, Prof. Dr.-Ing.: Taschenbuch der Informatik [TDI], Seite 289 Kapitel 10, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Dateiverzeichnis";
- [17] Schneider, Uwe, Prof. Dr.-Ing.: Taschenbuch der Informatik [TDI], Seite 287 Kapitel 10, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Datei";
- [18] Henning, Peter A., Prof. Dr. rer. nat. habil.: Taschenbuch der Informatik [TDI], Seite 403 Kapitel 12, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Apache"; [APAC]
- [19] Oechsle, Rainer, Prof. Dr.: Taschenbuch der Informatik [TDI], Seite 421 Kapitel 13, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Thin Client";
- [20] Oechsle, Rainer, Prof. Dr.: Taschenbuch der Informatik [TDI], Seite 407 Kapitel 13, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Server";
- [21] Oechsle, Rainer, Prof. Dr.: Taschenbuch der Informatik [TDI], Seite 407 Kapitel 13, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Client";
- [22] Oechsle, Rainer, Prof. Dr.: Taschenbuch der Informatik [TDI], Seite 407 Kapitel 13, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Peer";
- [23] Spielmann, Heinz-Jürgen, Prof. Dr. agr.: Taschenbuch der Informatik [TDI], Seite 241 Kapitel 9, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung des Begriffs "Sequenzdiagramm";
- [24] Zhang, Ben: A unification of functional and imperative languages [UFIL], Seite 27, 2007, University of Arkansas. Erklärung der Begriffe "Lazy evaluation" und "Eager evaluation";
- [25] Mansmann, Urs: c't Magazin - Breiband rasant [CTNET], KW25/2010, Heise Zeitschriften Verlag.
- [26] Allweyer, Thomas, Prof. Dr.: BPMN 2.0 - Business Process Model and Notation [BPMN], 2009, Books on Demand GmbH.
- [27] Figge,Frank, Schaltegger, Stefan: Was ist Stakeholder Value? - Vom Schlagwort zur Messung
- [STAK], Seite 11 Absatz 3, 1999, Universität Lüneburg. Erklärung des Begriffs "Stakeholder";
- [28] Baldenhofer, Roland und Hauth, Ute: Pflichtenheft Bachelorthesis - Galerie Synchronisation, Seite 2 Abbildung 1, miradlo, PDF-Dokument, 03.09.2010
- [29] Baldenhofer, Roland und Hauth, Ute: Pflichtenheft Bachelorthesis - Galerie Synchronisation, Seite 6 Kapitel 5, miradlo, PDF-Dokument, 03.09.2010
- [30] Crockford, D.: Request for Comments: 4627 - The application/json Media Type for JavaScript Object Notation (JSON), IETF - Network Working Group, 2006, The Internet Society. Erklärung der Abkürzung JSON
- [31] Victor, Frank, Prof. Dr.-Ing. habil. Dr. rer. nat.: Taschenbuch der Informatik [TDI], Seite 205 Kapitel 8, 6. Auflage, 2007, Carl Hanser Verlag, München. Erklärung der Abkürzung "Java-Applet";
- [32] Böhm, Rolf, Fuchs, Emmerich: System-Entwicklung in der Wirtschaftsinformatik, Seite 139 Kapitel 3.2, Auflage 5, vdf Hochschulverlag AG an der ETH Zürich. Erklärung des Begriffs "Anforderung";
- [33] Wenz, Christian: JavaScript - Das umfassende Handbuch, 6. Auflage, 2005, Galileo Press. Nachschlagewerk zu den Begriffen "JavaScript" und "Flash";
- [34] Baldenhofer, Roland und Hauth, Ute: Pflichtenheft Bachelorthesis - Galerie Synchronisation, Seite 3 Kapitel 2, miradlo, PDF-Dokument, 03.09.2010

6.2.3 Internetverweise

Auf die folgenden Webseiten wird in dieser Bachelorarbeit verwiesen. Der Abruf der Seiten fand hierbei im ersten Quartal des Jahres 2011 statt.

- [TWIT] Twitter, Twitter Inc., https://twitter.com/about
- [WP] WordPress - Blog Tool and Publishing Platform, Matt Mullenweg und weitere Autoren, http://wordpress.org/
- [BPMN] BPMN Information Home, Object Management Group, Inc., http://www.bpmn.org/
- [MIRAD] miradlo - Informatikdienstleistungen und Webdesign, Ute Hauth, Roland Baldenhofer, http://miradlo.com/
- [CTNET] c't Magazin - Breitband rasant, Heise Zeitschriften Verlag, http://www.heise.de/ct/artikel/Breitband-rasant-1138076.html
- [CAKE] CakePHP - The rapid development php framework., Cake Software Foundation Inc., http://cakephp.org/
- [PHP] PHP - PHP: Hypertext Preprocessor, The PHP Group, http://php.net/
- [ARC42] arc42 - Das Template, Dr. Peter Hruschka, Dr. Gernot Starke, Version 4.0, http://www.arc42.de/template/template.html
- [EA8] Enterprise Architect - UML Design Tools and UML CASE tools for software development, Sparx Systems Pty Ltd. Australia, Version 8.0, http://www.sparxsystems.com/products/ea/index.html
- [PAINT] Microsoft Paint - Windows XP Professional Product Documentation - Microsoft Paint overview, Microsoft Corporation, http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/mspaint_overview.mspx
- [GIMP] GIMP - The GNU Image Manipulation Program, The GIMP Team, http://www.gimp.org/
- [EEE9] ASUS Eee PC™ Comparison List, ASUSTeK Computer Inc., http://event.asus.com/eeepc/comparison/eeepc_comparison.htm
- [TAG] Tag (Informatik) - Wikipedia, die freie Enzyklopädie, verschiedene Autoren, Artikel vom 12. März 2011, http://de.wikipedia.org/w/index.php?title=Tag_%28Informatik%29&oldid=86342374
- [APAC] The Apache HTTP Server Project, The Apache Software Foundation, http://httpd.apache.org/
- [MYSQL] MySQL - Die populärste Open-Source-Datenbank der Welt, Oracle Corporation, http://mysql.de
- [TABL] Tablet-Computer - Wikipedia, die freie Enzyklopädie, verschiedene Autoren, Artikel vom 25. Februar 2011, http://de.wikipedia.org/w/index.php?title=Tablet-Computer&oldid=85735455
- [CPUT] Prozessorzeit - Wikipedia, die freie Enzyklopädie, verschiedene Autoren, Artikel vom 05. Dezember 2010, http://de.wikipedia.org/w/index.php?title=Prozessorzeit&oldid=82281726
- [DATEN] Datensicherung - Wikipedia, die freie Enzyklopädie, verschiedene Autoren, Artikel vom 07. März 2011, http://de.wikipedia.org/w/index.php?title=Datensicherung&oldid=86145117
- [TDI] Taschenbuch der Informatik, Google Books, http://books.google.de/books?id=nlPKHk5XzXAC
- [IH] Informatik-Handbuch, Google Books, http://books.google.de/books?id=N4V2q941AD8C
- [UFIL] A unification of functional and imperative languages, Google Books, http://books.google.de/books?id=iFUhx_dD3h0C
- [STAK] Was ist Stakeholder Value? Vom Schlagwort zur Messung, http://www2.leuphana.de/umanagement/csm/content/nama/downloads/download_publikationen/02-3downloadversion.pdf
- [A42TK] arc42 - 9. Technische Konzepte, Übergreifende Themen und technische Konzepte, http://www.arc42.de/template/template/8-concepts.html
- [JSON] RFC 4627 - The application/json Media Type for JavaScript Object Notation (JSON), http://tools.ietf.org/html/rfc4627
- [ZUG] File:Fotothek df roe-neg 0006232 033 Porträt eines Mannes im oberen Abteil eines Dopp.jpg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:Fotothek_df_roe-neg_0006232_033_Portr%C3%A4t_eines_Mannes_im_oberen_Abteil_eines_Dopp.jpg

- [OFFL] File:Network-offline.svg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:Network-offline.svg
- [UPDT] File:System-software-update.svg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:System-software-update.svg
- [KC85] File:KC85-1_Arbeitsplatz_2.jpg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:KC85-1_Arbeitsplatz_2.jpg
- [TWIC] Twitter.svg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:Twitter.svg
- [FB] Facebook icon.svg - Wikimedia Commons, http://commons.wikimedia.org/wiki/File:Facebook_icon.svg
- [CC] Creative Commons - Attribution-ShareAlike 3.0 Germany, CC BY-SA 3.0 - http://creativecommons.org/licenses/by-sa/3.0/de/deed.de
- [SELE] Selenium web application testing system, http://seleniumhq.org/

6.2.4 Bildquellen

Die in diesem Kapitel aufgeführten Bilder stammen zumindest teilweise von externen Quellen. Sämtliche anderen Bilder dieser Arbeit wurden selbst erstellt.

- [Bild 0] Heidrich, Thomas: Online und Offline arbeiten, Kombination aus [Bild 1] [Bild 2] [Bild 3] und [Bild 4], 08.04.2011, CC BY-SA 3.0.
- [Bild 1] Rössing, Roger, Rössing, Renate: Porträt eines Mannes im oberen Abteil eines Doppelstockwagens [ZUG],1952, Deutsche Fotothek, CC BY-SA 3.0.
- [Bild 2] The Tango! Desktop Project: Network-offline [OFFL], 2007, Gemeinfrei.
- [Bild 3] The Tango! Desktop Project: System-software-update [UPDT], 2007, Gemeinfrei.
- [Bild 4] Wollny, Hans: KC85-1 Arbeitsplatz 2 [KC85], 19.03.2007, Copyright-Halter-Erlaubnis.
- [Bild 5] Heidrich, Thomas: miradlo-Galerien mit externen Anwendungen, Kombination aus [Bild 6], [Bild 7] und eigenen Bildern, 08.04.2011, CC BY-SA 3.0
- [Bild 6] ZyMOS: Twitter [TWIC], Januar 2010, Public Domain
- [Bild 7] ZyMOS: Twitter [FB], Januar 2010, Public Domain

6.2.5 Abbildungsverzeichnis

6.2.6 Lizenz

Diese Bachelorthesis ist unter der Creative Commons-Lizenz Namensnennung-Weitergabe unter gleichen Bedingungen 3.0 Deutschland (CC BY-SA 3.0) [CC] lizenziert. Es folgt der Lizenztext.

Creative Commons

Namensnennung - Weitergabe unter gleichen Bedingungen 3.0 Deutschland

CREATIVE COMMONS IST KEINE RECHTSANWALTSKANZLEI UND LEISTET KEINE RECHTSBERATUNG. DIE BEREITSTELLUNG DIESER LIZENZ FÜHRT ZU KEINEM MANDATSVERHÄLTNIS. CREATIVE COMMONS STELLT DIESE INFORMATIONEN OHNE GEWÄHR ZUR VERFÜGUNG. CREATIVE COMMONS ÜBERNIMMT KEINE GEWÄHRLEISTUNG FÜR DIE GELIEFERTEN INFORMATIONEN UND SCHLIEßT DIE HAFTUNG FÜR SCHÄDEN AUS, DIE SICH AUS DEREN GEBRAUCH ERGEBEN.
Lizenz
DER GEGENSTAND DIESER LIZENZ (WIE UNTER "SCHUTZGEGENSTAND" DEFINIERT) WIRD UNTER DEN BEDINGUNGEN DIESER CREATIVE COMMONS PUBLIC LICENSE ("CCPL", "LIZENZ" ODER "LIZENZVERTRAG") ZUR VERFÜGUNG GESTELLT. DER SCHUTZGEGENSTAND IST DURCH DAS URHEBERRECHT UND/ODER ANDERE GESETZE GESCHÜTZT. JEDE FORM DER NUTZUNG DES SCHUTZGEGENSTANDES, DIE NICHT AUFGRUND DIESER LIZENZ ODER DURCH GESETZE GESTATTET IST, IST UNZULÄSSIG.
DURCH DIE AUSÜBUNG EINES DURCH DIESE LIZENZ GEWÄHRTEN RECHTS AN DEM SCHUTZGEGENSTAND ERKLÄREN SIE SICH MIT DEN LIZENZBEDINGUNGEN RECHTSVERBINDLICH EINVERSTANDEN. SOWEIT DIESE LIZENZ ALS LIZENZVERTRAG ANZUSEHEN IST, GEWÄHRT IHNEN DER LIZENZGEBER DIE IN DER LIZENZ GENANNTEN RECHTE UNENTGELTLICH UND IM AUSTAUSCH DAFÜR, DASS SIE DAS GEBUNDENSEIN AN DIE LIZENZBEDINGUNGEN AKZEPTIEREN.
1. Definitionen
a. Der Begriff "Abwandlung" im Sinne dieser Lizenz bezeichnet das Ergebnis jeglicher Art von Veränderung des Schutzgegenstandes, solange die eigenpersönlichen Züge des Schutzgegenstandes darin nicht verblassen und daran eigene Schutzrechte entstehen. Das kann insbesondere eine Bearbeitung, Umgestaltung, Änderung, Anpassung, Übersetzung oder Heranziehung des Schutzgegenstandes zur Vertonung von Laufbildern sein. Nicht als Abwandlung des Schutzgegenstandes gelten seine Aufnahme in eine Sammlung oder ein Sammelwerk und die freie Benutzung des Schutzgegenstandes.
b. Der Begriff "Sammelwerk" im Sinne dieser Lizenz meint eine Zusammenstellung von literarischen, künstlerischen oder wissenschaftlichen Inhalten, sofern diese Zusammenstellung aufgrund von Auswahl und Anordnung der darin enthaltenen selbständigen Elemente eine geistige Schöpfung darstellt, unabhängig davon, ob die Elemente systematisch oder methodisch angelegt und dadurch einzeln zugänglich sind oder nicht.
c. "Verbreiten" im Sinne dieser Lizenz bedeutet, den Schutzgegenstand oder Abwandlungen im Original oder in Form von Vervielfältigungsstücken, mithin in körperlich fixierter Form der Öffentlichkeit anzubieten oder in Verkehr zu bringen.
d. Unter "Lizenzelementen" werden im Sinne dieser Lizenz die folgenden übergeordneten Lizenzcharakteristika verstanden, die vom Lizenzgeber ausgewählt wurden und in der Bezeichnung der Lizenz zum Ausdruck kommen: "Namensnennung", "Weitergabe unter gleichen Bedingungen".
e. Der "Lizenzgeber" im Sinne dieser Lizenz ist diejenige natürliche oder juristische Person oder Gruppe, die den Schutzgegenstand unter den Bedingungen dieser Lizenz anbietet und insoweit als Rechteinhaberin auftritt.
f. "Rechteinhaber" im Sinne dieser Lizenz ist der Urheber des Schutzgegenstandes oder jede andere natürliche oder juristische Person oder Gruppe von Personen, die am Schutzgegenstand ein Immaterialgüterrecht erlangt hat, welches die in Abschnitt 3 genannten Handlungen erfasst und bei dem eine Einräumung von Nutzungsrechten oder eine Weiterübertragung an Dritte möglich ist.
g. Der Begriff "Schutzgegenstand" bezeichnet in dieser Lizenz den literarischen, künstlerischen oder wissenschaftlichen Inhalt, der unter den Bedingungen dieser Lizenz angeboten wird. Das kann insbesondere eine persönliche geistige Schöpfung jeglicher Art, ein Werk der kleinen Münze, ein nachgelassenes Werk oder auch ein Lichtbild oder anderes Objekt eines verwandten Schutzrechts sein, unabhängig von der Art seiner Fixierung und unabhängig davon, auf welche Weise jeweils eine Wahrnehmung erfolgen kann, gleichviel ob in analoger oder digitaler Form. Soweit Datenbanken oder Zusammenstellungen von Daten einen immaterialgüterrechtlichen Schutz eigener Art genießen, unterfallen auch sie dem Begriff "Schutzgegenstand" im Sinne dieser Lizenz.
h. Mit "Sie" bzw. "Ihnen" ist die natürliche oder juristische Person gemeint, die in dieser Lizenz im Abschnitt 3 genannte Nutzungen des Schutzgegenstandes vornimmt und zuvor in Hinblick auf den Schutzgegenstand nicht gegen Bedingungen dieser Lizenz verstoßen oder aber die ausdrückliche Erlaubnis des Lizenzgebers erhalten hat, die durch diese Lizenz gewährten Nutzungsrechte trotz eines vorherigen Verstoßes auszuüben.
i. Unter "Öffentlich Zeigen" im Sinne dieser Lizenz sind Veröffentlichungen und Präsentationen des Schutzgegenstandes zu verstehen, die für eine Mehrzahl von Mitgliedern der Öffentlichkeit bestimmt sind und in unkörperlicher Form mittels öffentlicher Wiedergabe in Form von Vortrag, Aufführung, Vorführung, Darbietung, Sendung, Weitersendung, zeit- und ortsunabhängiger Zugänglichmachung oder in körperlicher Form mittels Ausstellung erfolgen, unabhängig von bestimmten Veranstaltungen und unabhängig von den zum Einsatz

kommenden Techniken und Verfahren, einschließlich drahtgebundener oder drahtloser Mittel und Einstellen in das Internet.
j. "Vervielfältigen" im Sinne dieser Lizenz bedeutet, mittels beliebiger Verfahren Vervielfältigungsstücke des Schutzgegenstandes herzustellen, insbesondere durch Ton- oder Bildaufzeichnungen, und umfasst auch den Vorgang, erstmals körperliche Fixierungen des Schutzgegenstandes sowie Vervielfältigungsstücke dieser Fixierungen anzufertigen, sowie die Übertragung des Schutzgegenstandes auf einen Bild- oder Tonträger oder auf ein anderes elektronisches Medium, gleichviel ob in digitaler oder analoger Form.
k. "Mit Creative Commons kompatible Lizenz" bezeichnet eine Lizenz, die unter http://creativecommons.org/compatiblelicenses aufgelistet ist und die durch Creative Commons als grundsätzlich zur vorliegenden Lizenz äquivalent akzeptiert wurde, da zumindest folgende Voraussetzungen erfüllt sind:
Diese mit Creative Commons kompatible Lizenz
i. enthält Bestimmungen, welche die gleichen Ziele verfolgen, die gleiche Bedeutung haben und die gleichen Wirkungen erzeugen wie die Lizenzelemente der vorliegenden Lizenz; und
ii. erlaubt ausdrücklich das Lizenzieren von ihr unterstellten Abwandlungen unter vorliegender Lizenz, unter einer anderen rechtsordnungsspezifisch angepassten Creative-Commons-Lizenz mit denselben Lizenzelementen, wie sie die vorliegende Lizenz aufweist, oder unter der entsprechenden Creative-Commons-Unported-Lizenz.
2. Schranken des Immaterialgüterrechts
Diese Lizenz ist in keiner Weise darauf gerichtet, Befugnisse zur Nutzung des Schutzgegenstandes zu vermindern, zu beschränken oder zu vereiteln, die Ihnen aufgrund der Schranken des Urheberrechts oder anderer Rechtsnormen bereits ohne Weiteres zustehen oder sich aus dem Fehlen eines immaterialgüterrechtlichen Schutzes ergeben.
3. Einräumung von Nutzungsrechten
Unter den Bedingungen dieser Lizenz räumt Ihnen der Lizenzgeber - unbeschadet unverzichtbarer Rechte und vorbehaltlich des Abschnitts 3.e) - das vergütungsfreie, räumlich und zeitlich (für die Dauer des Schutzrechts am Schutzgegenstand) unbeschränkte einfache Recht ein, den Schutzgegenstand auf die folgenden Arten und Weisen zu nutzen ("unentgeltlich eingeräumtes einfaches Nutzungsrecht für jedermann"):
a. Den Schutzgegenstand in beliebiger Form und Menge zu vervielfältigen, ihn in Sammelwerke zu integrieren und ihn als Teil solcher Sammelwerke zu vervielfältigen;
b. Abwandlungen des Schutzgegenstandes anzufertigen, einschließlich Übersetzungen unter Nutzung jedweder Medien, sofern deutlich erkennbar gemacht wird, dass es sich um Abwandlungen handelt;
c. den Schutzgegenstand, allein oder in Sammelwerke aufgenommen, öffentlich zu zeigen und zu verbreiten;
d. Abwandlungen des Schutzgegenstandes zu veröffentlichen, öffentlich zu zeigen und zu verbreiten.
e. Bezüglich Vergütung für die Nutzung des Schutzgegenstandes gilt Folgendes:
i. Unverzichtbare gesetzliche Vergütungsansprüche: Soweit unverzichtbare Vergütungsansprüche im Gegenzug für gesetzliche Lizenzen vorgesehen oder Pauschalabgabensysteme (zum Beispiel für Leermedien) vorhanden sind, behält sich der Lizenzgeber das ausschließliche Recht vor, die entsprechende Vergütung einzuziehen für jede Ausübung eines Rechts aus dieser Lizenz durch Sie.
ii. Vergütung bei Zwangslizenzen: Sofern Zwangslizenzen außerhalb dieser Lizenz vorgesehen sind und zustande kommen, verzichtet der Lizenzgeber für alle Fälle einer lizenzgerechten Nutzung des Schutzgegenstandes durch Sie auf jegliche Vergütung.
iii. Vergütung in sonstigen Fällen: Bezüglich lizenzgerechter Nutzung des Schutzgegenstandes durch Sie, die nicht unter die beiden vorherigen Abschnitte (i) und (ii) fällt, verzichtet der Lizenzgeber auf jegliche Vergütung, unabhängig davon, ob eine Einziehung der Vergütung durch ihn selbst oder nur durch eine Verwertungsgesellschaft möglich wäre.
Das vorgenannte Nutzungsrecht wird für alle bekannten sowie für alle noch nicht bekannten Nutzungsarten eingeräumt. Es beinhaltet auch das Recht, solche Änderungen am Schutzgegenstand vorzunehmen, die für bestimmte nach dieser Lizenz zulässige Nutzungen technisch erforderlich sind. Alle sonstigen Rechte, die über diesen Abschnitt hinaus nicht ausdrücklich durch den Lizenzgeber eingeräumt werden, bleiben diesem allein vorbehalten. Soweit Datenbanken oder Zusammenstellungen von Daten Schutzgegenstand dieser Lizenz oder Teil dessen sind und einen immaterialgüterrechtlichen Schutz eigener Art genießen, verzichtet der Lizenzgeber auf sämtliche aus diesem Schutz resultierenden Rechte.
4. Bedingungen
Die Einräumung des Nutzungsrechts gemäß Abschnitt 3 dieser Lizenz erfolgt ausdrücklich nur unter den folgenden Bedingungen:
a. Sie dürfen den Schutzgegenstand ausschließlich unter den Bedingungen dieser Lizenz verbreiten oder öffentlich zeigen. Sie müssen dabei stets eine Kopie dieser Lizenz oder deren vollständige Internetadresse in Form des Uniform-Resource-Identifier (URI) beifügen. Sie dürfen keine Vertrags- oder Nutzungsbedingungen anbieten oder fordern, die die Bedingungen dieser Lizenz oder die durch diese Lizenz gewährten Rechte beschränken. Sie dürfen den Schutzgegenstand nicht unterlizenzieren. Bei jeder Kopie des Schutzgegenstandes, die Sie verbreiten oder öffentlich zeigen, müssen Sie alle Hinweise unverändert lassen, die auf diese Lizenz und den Haftungsausschluss hinweisen. Wenn Sie den Schutzgegenstand verbreiten oder öffentlich zeigen, dürfen Sie (in Bezug auf den Schutzgegenstand) keine technischen Maßnahmen ergreifen, die den Nutzer des Schutzgegenstandes in der Ausübung der ihm durch diese Lizenz gewährten Rechte behindern können. Dieser Abschnitt 4.a) gilt auch für den Fall, dass der Schutzgegenstand einen Bestandteil eines Sammelwerkes bildet, was jedoch nicht bedeutet, dass das Sammelwerk insgesamt dieser Lizenz unterstellt werden muss. Sofern Sie ein Sammelwerk erstellen, müssen Sie auf die Mitteilung eines Lizenzgebers hin aus dem Sammelwerk die in Abschnitt 4.c) aufgezählten Hinweise

entfernen. Wenn Sie eine Abwandlung vornehmen, müssen Sie auf die Mitteilung eines Lizenzgebers hin von der Abwandlung die in Abschnitt 4.c) aufgezählten Hinweise entfernen.
b. Sie dürfen eine Abwandlung ausschließlich unter den Bedingungen
i. dieser Lizenz,
ii. einer späteren Version dieser Lizenz mit denselben Lizenzelementen,
iii. einer rechtsordnungsspezifischen Creative-Commons-Lizenz mit denselben Lizenzelementen ab Version 3.0 aufwärts (z.B. Namensnennung - Weitergabe unter gleichen Bedingungen 3.0 US),
iv. der Creative-Commons-Unported-Lizenz mit denselben Lizenzelementen ab Version 3.0 aufwärts, oder
v. einer mit Creative Commons kompatiblen Lizenz
verbreiten oder öffentlich zeigen.
Falls Sie die Abwandlung gemäß Abschnitt (v) unter einer mit Creative Commons kompatiblen Lizenz lizenzieren, müssen Sie deren Lizenzbestimmungen Folge leisten.
Falls Sie die Abwandlungen unter einer der unter (i)-(iv) genannten Lizenzen ("Verwendbare Lizenzen") lizenzieren, müssen Sie deren Lizenzbestimmungen sowie folgenden Bestimmungen Folge leisten: Sie müssen stets eine Kopie der verwendbaren Lizenz oder deren vollständige Internetadresse in Form des Uniform-Resource-Identifier (URI) beifügen, wenn Sie die Abwandlung verbreiten oder öffentlich zeigen. Sie dürfen keine Vertrags- oder Nutzungsbedingungen anbieten oder fordern, die die Bedingungen der verwendbaren Lizenz oder die durch sie gewährten Rechte beschränken. Bei jeder Abwandlung, die Sie verbreiten oder öffentlich zeigen, müssen Sie alle Hinweise auf die verwendbare Lizenz und den Haftungsausschluss unverändert lassen. Wenn Sie die Abwandlung verbreiten oder öffentlich zeigen, dürfen Sie (in Bezug auf die Abwandlung) keine technischen Maßnahmen ergreifen, die den Nutzer der Abwandlung in der Ausübung der ihm durch die verwendbare Lizenz gewährten Rechte behindern können. Dieser Abschnitt 4.b) gilt auch für den Fall, dass die Abwandlung einen Bestandteil eines Sammelwerkes bildet, was jedoch nicht bedeutet, dass das Sammelwerk insgesamt der verwendbaren Lizenz unterstellt werden muss.
c. Die Verbreitung und das öffentliche Zeigen des Schutzgegenstandes oder auf ihm aufbauender Abwandlungen oder ihn enthaltender Sammelwerke ist Ihnen nur unter der Bedingung gestattet, dass Sie, vorbehaltlich etwaiger Mitteilungen im Sinne von Abschnitt 4.a), alle dazu gehörenden Rechtevermerke unberührt lassen. Sie sind verpflichtet, die Rechteinhaberschaft in einer der Nutzung entsprechenden, angemessenen Form anzuerkennen, indem Sie - soweit bekannt - Folgendes angeben:
i. Den Namen (oder das Pseudonym, falls ein solches verwendet wird) des Rechteinhabers und / oder, falls der Lizenzgeber im Rechtevermerk, in den Nutzungsbedingungen oder auf andere angemessene Weise eine Zuschreibung an Dritte vorgenommen hat (z.B. an eine Stiftung, ein Verlagshaus oder eine Zeitung) ("Zuschreibungsempfänger"), Namen bzw. Bezeichnung dieses oder dieser Dritten;
ii. den Titel des Inhaltes;
iii. in einer praktikablen Form den Uniform-Resource-Identifier (URI, z.B. Internetadresse), den der Lizenzgeber zum Schutzgegenstand angegeben hat, es sei denn, dieser URI verweist nicht auf den Rechtevermerk oder die Lizenzinformationen zum Schutzgegenstand;
iv. und im Falle einer Abwandlung des Schutzgegenstandes in Übereinstimmung mit Abschnitt 3.b) einen Hinweis darauf, dass es sich um eine Abwandlung handelt.
Die nach diesem Abschnitt 4.c) erforderlichen Angaben können in jeder angemessenen Form gemacht werden; im Falle einer Abwandlung des Schutzgegenstandes oder eines Sammelwerkes müssen diese Angaben das Minimum darstellen und bei gemeinsamer Nennung mehrerer Rechteinhaber dergestalt erfolgen, dass sie zumindest ebenso hervorgehoben sind wie die Hinweise auf die übrigen Rechteinhaber. Die Angaben nach diesem Abschnitt dürfen Sie ausschließlich zur Angabe der Rechteinhaberschaft in der oben bezeichneten Weise verwenden. Durch die Ausübung Ihrer Rechte aus dieser Lizenz dürfen Sie ohne eine vorherige, separat und schriftlich vorliegende Zustimmung des Lizenzgebers und / oder des Zuschreibungsempfängers weder explizit noch implizit irgendeine Verbindung zum Lizenzgeber oder Zuschreibungsempfänger und ebenso wenig eine Unterstützung oder Billigung durch ihn andeuten.
d. Die oben unter 4.a) bis c) genannten Einschränkungen gelten nicht für solche Teile des Schutzgegenstandes, die allein deshalb unter den Schutzgegenstandsbegriff fallen, weil sie als Datenbanken oder Zusammenstellungen von Daten einen immaterialgüterrechtlichen Schutz eigener Art genießen.
e. Persönlichkeitsrechte bleiben - soweit sie bestehen - von dieser Lizenz unberührt.
5. Gewährleistung
SOFERN KEINE ANDERS LAUTENDE, SCHRIFTLICHE VEREINBARUNG ZWISCHEN DEM LIZENZGEBER UND IHNEN GESCHLOSSEN WURDE UND SOWEIT MÄNGEL NICHT ARGLISTIG VERSCHWIEGEN WURDEN, BIETET DER LIZENZGEBER DEN SCHUTZGEGENSTAND UND DIE EINRÄUMUNG VON RECHTEN UNTER AUSSCHLUSS JEGLICHER GEWÄHRLEISTUNG AN UND ÜBERNIMMT WEDER AUSDRÜCKLICH NOCH KONKLUDENT GARANTIEN IRGENDEINER ART. DIES UMFASST INSBESONDERE DAS FREISEIN VON SACH- UND RECHTSMÄNGELN, UNABHÄNGIG VON DEREN ERKENNBARKEIT FÜR DEN LIZENZGEBER, DIE VERKEHRSFÄHIGKEIT DES SCHUTZGEGENSTANDES, SEINE VERWENDBARKEIT FÜR EINEN BESTIMMTEN ZWECK SOWIE DIE KORREKTHEIT VON BESCHREIBUNGEN. DIESE GEWÄHRLEISTUNGSBESCHRÄNKUNG GILT NICHT, SOWEIT MÄNGEL ZU SCHÄDEN DER IN ABSCHNITT 6 BEZEICHNETEN ART FÜHREN UND AUF SEITEN DES LIZENZGEBERS DAS JEWEILS GENANNTE VERSCHULDEN BZW. VERTRETENMÜSSEN EBENFALLS VORLIEGT.
6. Haftungsbeschränkung

DER LIZENZGEBER HAFTET IHNEN GEGENÜBER IN BEZUG AUF SCHÄDEN AUS DER VERLETZUNG DES LEBENS, DES KÖRPERS ODER DER GESUNDHEIT NUR, SOFERN IHM WENIGSTENS FAHRLÄSSIGKEIT VORZUWERFEN IST, FÜR SONSTIGE SCHÄDEN NUR BEI GROBER FAHRLÄSSIGKEIT ODER VORSATZ, UND ÜBERNIMMT DARÜBER HINAUS KEINERLEI FREIWILLIGE HAFTUNG.

7. Erlöschen

a. Diese Lizenz und die durch sie eingeräumten Nutzungsrechte erlöschen mit Wirkung für die Zukunft im Falle eines Verstoßes gegen die Lizenzbedingungen durch Sie, ohne dass es dazu der Kenntnis des Lizenzgebers vom Verstoß oder einer weiteren Handlung einer der Vertragsparteien bedarf. Mit natürlichen oder juristischen Personen, die Abwandlungen des Schutzgegenstandes oder diesen enthaltende Sammelwerke unter den Bedingungen dieser Lizenz von Ihnen erhalten haben, bestehen nachträglich entstandene Lizenzbeziehungen jedoch solange weiter, wie die genannten Personen sich ihrerseits an sämtliche Lizenzbedingungen halten. Darüber hinaus gelten die Ziffern 1, 2, 5, 6, 7, und 8 auch nach einem Erlöschen dieser Lizenz fort.

b. Vorbehaltlich der oben genannten Bedingungen gilt diese Lizenz unbefristet bis der rechtliche Schutz für den Schutzgegenstand ausläuft. Davon abgesehen behält der Lizenzgeber das Recht, den Schutzgegenstand unter anderen Lizenzbedingungen anzubieten oder die eigene Weitergabe des Schutzgegenstandes jederzeit einzustellen, solange die Ausübung dieses Rechts nicht einer Kündigung oder einem Widerruf dieser Lizenz (oder irgendeiner Weiterlizenzierung, die auf Grundlage dieser Lizenz bereits erfolgt ist bzw. zukünftig noch erfolgen muss) dient und diese Lizenz unter Berücksichtigung der oben zum Erlöschen genannten Bedingungen vollumfänglich wirksam bleibt.

8. Sonstige Bestimmungen

a. Jedes Mal wenn Sie den Schutzgegenstand für sich genommen oder als Teil eines Sammelwerkes verbreiten oder öffentlich zeigen, bietet der Lizenzgeber dem Empfänger eine Lizenz zu den gleichen Bedingungen und im gleichen Umfang an, wie Ihnen in Form dieser Lizenz.

b. Jedes Mal wenn Sie eine Abwandlung des Schutzgegenstandes verbreiten oder öffentlich zeigen, bietet der Lizenzgeber dem Empfänger eine Lizenz am ursprünglichen Schutzgegenstand zu den gleichen Bedingungen und im gleichen Umfang an, wie Ihnen in Form dieser Lizenz.

c. Sollte eine Bestimmung dieser Lizenz unwirksam sein, so bleibt davon die Wirksamkeit der Lizenz im Übrigen unberührt.

d. Keine Bestimmung dieser Lizenz soll als abbedungen und kein Verstoß gegen sie als zulässig gelten, solange die von dem Verzicht oder von dem Verstoß betroffene Seite nicht schriftlich zugestimmt hat.

e. Diese Lizenz (zusammen mit in ihr ausdrücklich vorgesehenen Erlaubnissen, Mitteilungen und Zustimmungen, soweit diese tatsächlich vorliegen) stellt die vollständige Vereinbarung zwischen dem Lizenzgeber und Ihnen in Bezug auf den Schutzgegenstand dar. Es bestehen keine Abreden, Vereinbarungen oder Erklärungen in Bezug auf den Schutzgegenstand, die in dieser Lizenz nicht genannt sind. Rechtsgeschäftliche Änderungen des Verhältnisses zwischen dem Lizenzgeber und Ihnen sind nur über Modifikationen dieser Lizenz möglich. Der Lizenzgeber ist an etwaige zusätzliche, einseitig durch Sie übermittelte Bestimmungen nicht gebunden. Diese Lizenz kann nur durch schriftliche Vereinbarung zwischen Ihnen und dem Lizenzgeber modifiziert werden. Derlei Modifikationen wirken ausschließlich zwischen dem Lizenzgeber und Ihnen und wirken sich nicht auf die Dritten gemäß Ziffern 8.a) und b) angeboteten Lizenzen aus.

f. Sofern zwischen Ihnen und dem Lizenzgeber keine anderweitige Vereinbarung getroffen wurde und soweit Wahlfreiheit besteht, findet auf diesen Lizenzvertrag das Recht der Bundesrepublik Deutschland Anwendung.

Creative Commons Notice

Creative Commons ist nicht Partei dieser Lizenz und übernimmt keinerlei Gewähr oder dergleichen in Bezug auf den Schutzgegenstand. Creative Commons haftet Ihnen oder einer anderen Partei unter keinem rechtlichen Gesichtspunkt für irgendwelche Schäden, die - abstrakt oder konkret, zufällig oder vorhersehbar - im Zusammenhang mit dieser Lizenz entstehen. Unbeschadet der vorangegangen beiden Sätze, hat Creative Commons alle Rechte und Pflichten eines Lizenzgebers, wenn es sich ausdrücklich als Lizenzgeber im Sinne dieser Lizenz bezeichnet.

Creative Commons gewährt den Parteien nur insoweit das Recht, das Logo und die Marke "Creative Commons" zu nutzen, als dies notwendig ist, um der Öffentlichkeit gegenüber kenntlich zu machen, dass der Schutzgegenstand unter einer CCPL steht. Ein darüber hinaus gehender Gebrauch der Marke "Creative Commons" oder einer verwandten Marke oder eines verwandten Logos bedarf der vorherigen schriftlichen Zustimmung von Creative Commons. Jeder erlaubte Gebrauch richtet sich nach der Creative Commons Marken-Nutzungs-Richtlinie in der jeweils aktuellen Fassung, die von Zeit zu Zeit auf der Website veröffentlicht oder auf andere Weise auf Anfrage zugänglich gemacht wird. Zur Klarstellung: Die genannten Einschränkungen der Markennutzung sind nicht Bestandteil dieser Lizenz.

Creative Commons kann kontaktiert werden über http://creativecommons.org/.

6.3 Glossar

Begriff	Beschreibung
Bilder verwalten	Der Prozess *Bilder verwalten* deckt das Anlegen, Verändern, Löschen und Synchronisieren der einzelnen Bilder ab.
Bildergalerie	Eine installierte Instanz einer miradlo-Galerie wird in diesem Dokument durchgängig als miradlo-Galerie bezeichnet. Eine miradlo-Galerie ist eine Bildverwaltungsinstanz, davon können mehrere existieren, z.B. eine miradlo-Galerie mit Unternehmensbildern und eine mit Privatbildern. Innerhalb einer miradlo-Galerie können mehrere Bildergalerien existieren, z.B. Urlaubsbilder, Zeichnungen, usw. Der Begriff Bildergalerie bezeichnet in diesem Dokument also nicht ein e Instanz sondern eine"Teilgalerie". Eine miradlo-Galerie dient dazu, große Mengen von Bildern zu speichern und überschauen zu können. Die Bilder werden hierbei in Bildergalerien abgespeichert. Jedes Originalbild kann nur in einer Bildergalerie abgelegt werden. Die Bildergalerie erlaubt es, innerhalb von einer miradlo-Galerie mehrere "Datentöpfe" für Bilder zu erstellen und so die Bilder einer ersten Sortierung zu unterziehen. Eine Bildergalerie hat einen Namen und diverse Metadaten. Sie dient als Sammlung von Originalbildern, welche einen bestimmten Zusammenhang haben. Ein Bild muss zu jedem Zeitpunkt exakt einer Bildergalerie zugeordnet sein.
Blogsystem	Ein Blogsystem kann z.B. eine WordPress-Installation [WP] darstellen. Ein Blog ist ein System, in dem eine "Ansammlung periodisch neuer Einträge auf einer Webseite der chronologische Fortschritt einer individuellen Lern- und Mitteilungsbereitschaft dokumentiert" [3] wird. Innerhalb des Blogs sollen Bilder, die in der Online-Galerie vorhanden sind, dargestellt werden. Auf diese Weise kann die Verwaltung der Bilder in der Online-Galerie erfolgen und die Darstellung im Kontext des Blogsystems geschehen.
DirtyFlag	Ein DirtyFlag gibt die Änderung in einer freigegebenen Bildergalerie an. Damit kann nur anhand des Flags klar gestellt werden, dass eine Synchronisation nötig wäre.
E-Mail	Bilder können der Online-Galerie per E-Mail [7] hinzugefügt werden. So kann z.B. ein Foto mit dem Mobiltelefon aufgenommen und an die Online-Galerie gesendet werden. Die E-Mail beinhaltet dabei das eigentliche Bild und Zusatzinformationen, welche von der Online-Galerie zur Einordnung verwendet werden. "Die verbreitetste elektronische Kommunikationsform ist die *Electronic Mail*, kurz E-Mail [...]. Die Einschränkung auf textuelle Mitteilungen kann man überwinden, indem man beliebige Multimediadateien als Anlagen mitverschickt." [7]
Enterprise Architect	Der Enterprise Architect ist ein auf UML-basierendes Softwaremodellierungswerkzeug. Der Enterprise Architect wurde zur Erstellung dieser Bachelorarbeit verwendet. [EA8]
freigegebene Bildergalerie	Eine freigegebene Bildergalerie wird mit anderen miradlo-Galerien im Netzwerk synchronisiert. Sie ist am Service-Locator zu registrieren, wo sie eine eineindeutige ID zugewiesen bekommt. Auf diesem Weg kann auf ein eigenes Backup verzichtet werden, falls die freigegebene Bildergalerie von mehreren vollständig genutzt wird.
Galeriedaten	Die Galeriedaten beschreiben die persistente Speicherung der Bilder und deren Metadaten. Die Synchronisation beinhaltet die Betrachtung

Begriff	Beschreibung
	der zu speichernden Daten.
GUI	Der Begriff GUI steht für Graphical User Interface und beschreibt die Benutzeroberfläche, die einem Benutzer präsentiert wird.
Hashfunktion / Hashsumme	Eine Hashfunktion nimmt als Eingabewert eine große Quelldatenmenge, wie beispielsweise ein digitales Bild, und berechnet daraus eine kleinere - im Idealfall eineindeutige Zahl - die Hashsumme genannt wird. Diese Hashsumme kann für einfachere Vergleiche als die zwischen den Originalbildern herangezogen werden. Jede Hashsumme, die innerhalb dieser Arbeit genannt wird, hat eine konkrete Länge.
Kamera	Digitale Bilder werden im Normalfall von Digitalkameras erzeugt. Die Kameras können ihre Bilder der Online-Galerie zur Verfügung stellen. Dies kann durch den Zugriff über einen Browser erfolgen.
lokale Bildergalerie	Eine lokale Bildergalerie existiert nur auf dem Peer, auf dem sie erstellt wurde. Sie wird nicht mit dem Netzwerk aus miradlo-Galerien geteilt. Das bedeutet auch, dass selbst für ein etwaiges Backup gesorgt werden muss.
Lokaler Rechner (offline / online)	Der lokale Rechner erlaubt es dem Benutzer, die Bilder zuerst von einer Online-Galerie herunterzuladen und dann ohne Netzwerkverbindung z.B. im Zug weiter zu bearbeiten. Der lokale Rechner ist somit nicht permanent online. Der Abgleich jeglicher Änderungen soll automatisch erfolgen, sobald der Rechner wieder Netzwerkverbindung besitzt.
Metadaten	Metadaten umfassen unter anderem eine Bildbeschreibung, die Dateigröße, den Speicherort, einen Titel und weitere beschreibende Informationen.
miradlo-Galerie	Der Begriff miradlo-Galerie ist der Arbeitsname der installierten Instanz. Eine installierte Instanz einer miradlo-Galerie wird in diesem Dokument durchgängig als miradlo-Galerie bezeichnet. Eine miradlo-Galerie ist eine Bildverwaltungsinstanz, davon können mehrere existieren, z.B. eine miradlo-Galerie mit Unternehmensbildern und eine mit Privatbildern.
miradlokit	Das von miradlo entwickelte Framework miradlokit ist ein Baukasten für Webapplikationen, basierend auf CakePHP [CAKE] mit dem Basisanforderungen von Kunden abgedeckt werden.
Originalbild	Die ursprüngliche Version eines Bildes, welches in die miradlo-Galerie geladen wurde, wird als Originalbild bezeichnet. Auf das Originalbild ist nur lesender Zugriff erlaubt, solange es nicht explizit gelöscht wird.
Peer	Ein Peer entspricht genau einer physischen Instanz im Netzwerk aus Instanzen. Ein Peer enthält meist eine miradlo-Galerie, kann jedoch auch mehrere enthalten. Ein Peer ist in der Lage, mit anderen Peers zu kommunizieren, um Daten abzugleichen.
Redakteur	Dieser Akteur meldet sich an der miradlo-Galerie an, bevor er mit seiner Arbeit beginnt. Seine Arbeit besteht hauptsächlich aus dem Hinzufügen und Bearbeiten von Bildern.
Service-Locator	Der Service-Locator ist ein Informationsdienst für miradlo-Galerien. Er hilft den von ihm verwalteten miradlo-Galerien, sich untereinander zu finden. In der Suche nach freigegebene Bildergalerien kann nach Namen gesucht werden, oder es können alle freigegebenen Bildergalerien angezeigt werden.
SLGMGID	Die Service-Locator-Global-Miradlo-Gallery-ID wird vom Service-Locator generiert. Jede miradlo-Galerie, die sich bei einem Service-Locator registriert, bekommt eine solche Identifikationsnummer. Auf dem Service-Locator ist diese ID eineindeutig. Sie dient zur eindeutigen Identifikation einer miradlo-Galerie in einem Netzwerk aus miradlo-Galerien.
SLGPGID	Die Service-Locator-Global-Picture-Gallery-ID wird vom Service-Locator generiert. Jede freigegebene Bildergalerie, die sich bei einem

Begriff	Beschreibung
	Service-Locator registriert ist, besitzt eine solche Identifikationsnummer. Auf dem Service-Locator ist diese ID eineindeutig. Sie wird bei der Erstellung einer neuen freigegebenen Bildergalerie vom Service-Locator zugewiesen. Sie dient zur eindeutigen Identifikation einer freigegebenen Bildergalerie in einem Netzwerk aus miradlo-Galerien.
Stakeholder	Die Definition des Begriffs findet sich im *Kapitel 2.7, Stakeholder und Akteure, Seite 17.*
Transformationsbeschreibung	Die Definition des Begriffs findet sich im *Kapitel 3.2.3, Originalbild mit Transformationsbeschreibungen, Seite 27.*
USB-Stick	Der USB-Stick [6] steht allgemein für portable Massenspeichermedien. Hierunter fallen auch externe Festplatten, Kamera-Speicherkarten und viele mehr.