Modellierung und Quantifizierung der Zuverlässigkeit in Architekturen zur Bit Preservation

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Danah Tonne

aus Minden

Tag der mündlichen Prüfung:	08.02.2016
Erster Gutachter:	Prof. Dr. Bernhard Neumair
Zweiter Gutachter:	Prof. Dr. Michael Beigl
Dritter Gutachter:	Prof. Dr. Marc Weber

Zusammenfassung

Die langfristige Erhaltung von Informationen ist eine essentielle Voraussetzung für den wissenschaftlichen Fortschritt. Wurde früher zu Stein und Meissel gegriffen, um das kulturelle Erbe für die Ewigkeit zu bewahren, so ist die Problematik in der heutigen digitalen Welt um ein Vielfaches komplexer. „Bit Preservation“ ist der Grundbaustein einer nachhaltigen Langzeitarchivierung und bezeichnet Maßnahmen, die eine möglichst hohe Integrität von Daten über einen langen Zeitraum - geisteswissenschaftliche Forscher sprechen von Jahrhunderten - garantieren sollen.

Eine besondere Bedrohung stellen Fehler dar, die von der verwendeten Maßnahme zur Sicherstellung der Datenintegrität nicht entdeckbar sind und zu möglicherweise erst nach Jahren erkennbaren Datenverlusten führen. Heterogene Forschungsdaten sind unterschiedlich anfällig für diese Fehlerart, welche sich von einer leichten Farbveränderung eines Pixels in einer Bilddatei bis zur Unlesbarkeit einer Archivdatei auswirken können. Durch steigende Datenmengen wird das Auftreten mindestens eines unentdeckten Fehlers immer wahrscheinlicher, so dass die Entwicklung einer effektiven Strategie notwendig wird.

Gegenstand der vorliegenden Dissertation ist die Erforschung und Entwicklung eines neuen Modells zur Quantifizierung der Zuverlässigkeit in Architekturen zur Bit Preservation. Basierend auf der Analyse der Anforderun-

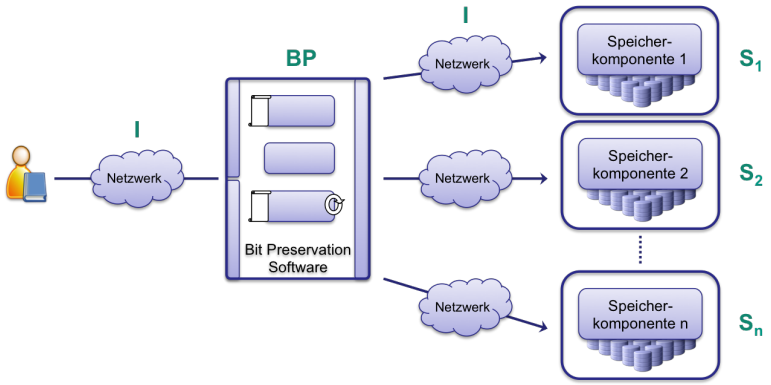

Abbildung 0.1: Schematische Darstellung einer synchronen Replikation mit Fehlerwahrscheinlichkeit S_i (BP, I, P) der Speicherkomponente i (der Software zur Bit Preservation, des Netzwerkes, der Prüfsumme).

gen geisteswissenschaftlicher Forscher sowie der Untersuchung der führenden Forschungsprojekte und verbreiteten Repositorien im Bereich Preservation wurde eine Standardarchitektur, die Architektur zur archivalischen Speicherung heterogener Forschungsdaten, definiert. Anhand einer Referenzimplementierung konnte die Architektur mit den inkludierten, disziplinübergreifend einsatzbaren Konzepten zur Langzeitarchivierung evaluiert und die Umsetzbarkeit nachgewiesen werden. Nach Definition der Wahrscheinlichkeit unentdeckter Fehler als Metrik wird eine Abschätzung der Zuverlässigkeit in Abhängigkeit der verwendeten Architekturkomponenten und Maßnahmen zur Bit Preservation getroffen. Bei einer gleichzeitigen Speicherung in allen genutzten Speicherkomponenten wie in Abbildung 0.1 gilt beispielsweise für die Wahrscheinlichkeit keines unentdeckten Fehlers

$$\overline{P}_{uFehler} = \frac{1}{n} \times BP^2 \times I^4 \times \left(\sum_{i=0}^{m} (P \times BP \times I)^i \times (\sum_{j=1}^{n} S_j^2 \times \prod_{k=1}^{i} \overline{S_{j-k}}) \right),$$

wobei $\overline{*}$ die Gegenwahrscheinlichkeit und n (m) die maximale Anzahl an Repliken (Ausweichschritten) beschreibt. Mit Hilfe des Modells sowie den abgeleiteten geschlossenen Ausdrücken wird erstmals eine Analyse unterschiedlicher Replikationsszenarien oder der Auswirkungen steigender Datenmengen in Bezug auf unentdeckte Fehler durchgeführt. Unter anderem wird deutlich, dass der aus dem Umgang mit entdeckten Fehlern bekannte Zusammenhang „je mehr Repliken, desto sicherer die Daten“ in diesem Kontext keine allgemeine Gültigkeit mehr besitzt. Basierend auf den Erkenntnissen der Analyse können Empfehlungen für Maßnahmen zur Bit Preservation abgeleitet werden, die auf unterschiedliche Anforderungen an die Zuverlässigkeit einer langfristigen Speicherung abgestimmt sind. Auf diese Weise sind Speicheranbieter in der Lage, passgenaue Architekturen bereitzustellen und eine langfristige und nachhaltige Erhaltung wertvoller Forschungsdaten maßgeblich zu unterstützen.

Abstract

Long-term preservation of data is essential for scientific progress. While in former times stone and chisel were the tools of choice to conserve the cultural heritage, the digital world complicates the task significantly. Bit preservation as the basic module for a sustainable long-term archiving denotes mechanisms to ensure data integrity for a long period of time. Humanities scholars in particular request reliable storage for hundreds of years.

Faults which cannot be detected by the selected mechanism for data integrity are of crucial importance as they lead to data loss, often recognizable years later. Heterogeneous research data varies in vulnerability to these faults, for instance a slight color change in image data or the unreadability of archive data can occur. Due to ever increasing amounts of data an undetected fault becomes more likely, thus the development of an effective strategy is absolutely essential.

The main result of this thesis is the development and analysis of a new model to quantify reliability in bit preservation architectures. Based on humanities scholars' requirements and the exploiration of leading research projects as well as repositories in the context of preservation a standard architecture, the architecture for archival storage of heterogeneous research data, is defined. With the help of a reference implementation the architecture including interdisciplinary concepts for long-term archiving is evaluated

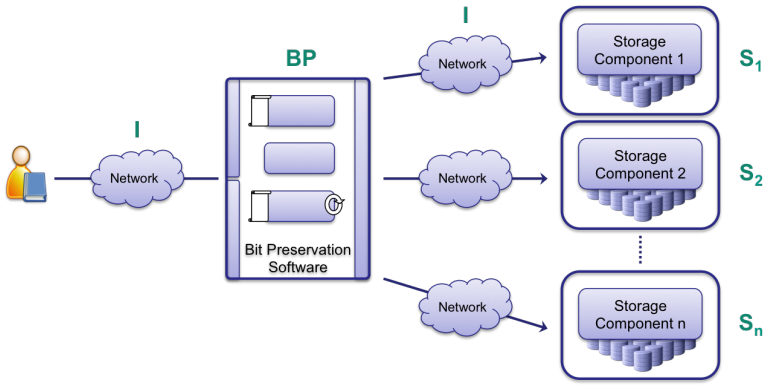

Figure 0.2: Schematic representation of a synchronous replication with fault probability S_i (BP, I, P) of the storage component i (the bit preservation software, the network, the checksum).

and the feasability is proven. After defining the probability of an undetected fault as a metric a reliability estimation depending on utilized architecture components and bit preservation mechanisms is given. In case of a simultaneous storage in all storage components (figure 0.2) the probability of no undetected fault is derived by

$$\overline{P}_{uFault} = \frac{1}{n} \times BP^2 \times I^4 \times \left(\sum_{i=0}^{m} (P \times BP \times I)^i \times (\sum_{j=1}^{n} S_j^2 \times \prod_{k=1}^{i} \overline{S_{j-k}}) \right),$$

whereat $\overline{*}$ denotes the counter probability and n (m) denotes the maximum number of replicas (evading steps). The model and the derived formulas allow a first-time analysis of replication scenarios as well as the effects of increasing data amounts focused on undetected faults. Among other findings it becomes apparent, that the general approach to deal with faults "the more replicas, the safer the data" is not valid in every scenario. Based on the analysis recommendations for bit preservation mechanisms adapted to different reliability requirements can be deduced. Thus storage providers are in a position to offer custom-fit architectures and support the long-term and sustainable preservation of valuable research data significantly.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Der neue Forschungszweig der Digital Humanities 1
1.2 Steigende Datenmengen und Big Data 4
1.3 Langzeitarchivierung von Forschungsdaten 5
1.4 Erkenntnisse und Forschungsfragen 8
1.5 Methodisches Vorgehen und Ergebnisse der Arbeit 9

2 Stand der Technik **12**
2.1 Forschungsprojekte im Bereich Preservation 14
2.2 Archivierungskonzepte und Repositorien 18
2.3 Methodiken der Bit Preservation 21
2.4 Erkenntnisse 27

3 Architektur zur archivalischen Speicherung heterogener Forschungsdaten **28**
3.1 Anforderungen und Struktur 28
3.2 Standardisierte Schnittstellen 31
3.3 Kernkonzepte zur Langzeitarchivierung 33
3.4 Anwendungsfall Virtuelles Skriptorium 39
3.5 Möglichkeiten der Datenmigration 43
3.6 Erkenntnisse 52

4 Modellierung der Zuverlässigkeit einer Architektur zur Bit Preservation **54**
4.1 Annahmen und Notationen 55
4.2 Szenario 0: Ohne Replikation 56
4.3 Verwendung von Prüfsummen 57
4.4 Replikationsszenarien aus Systemsicht 62
4.4.1 Szenario 1: Synchrone Replikation 63
4.4.2 Szenario 2: Asynchrone Replikation mit Masterdatei . . 66
4.4.3 Szenario 3: Asynchrone Replikation ohne Masterdatei . 68
4.5 Replikationsszenarien aus Nutzersicht 71
4.5.1 Szenario 4: Synchrone Replikation 71
4.5.2 Szenario 5: Asynchrone Replikation mit Masterdatei . . 73
4.5.3 Szenario 6: Asynchrone Replikation ohne Masterdatei . 75
4.6 Erkenntnisse 78

5 Quantifizierung der Wahrscheinlichkeit unentdeckter Fehler 79
5.1 Szenario 1: Synchrone Replikation aus Systemsicht 80
5.2 Vergleich der Replikationsszenarien aus Systemsicht 83
5.3 Szenario 5+6: Asynchrone Replikation aus Nutzersicht 85
5.4 Vergleich der Replikationsszenarien aus Nutzersicht 96
5.5 Auswirkungen steigender Datenmengen 99
5.6 Erkenntnisse 102

6 Diskussion 103
6.1 Forschung und Entwicklung im Projekt DARIAH 103
6.2 Entwicklungen im Bereich der Preservation 104
6.3 Diskussion der Modellannahmen 105
6.4 Empfehlungen für den Umgang mit unentdeckten Fehlern . . . 107
6.5 Auswirkungen auf die Langzeitarchivierung 111

7 Schlussfolgerungen 114

A Anhang 130
A.1 Speicherschnittstelle „DARIAH Storage API“ 130
A.2 Preservationsschnittstelle „DARIAH Admin API“ 132
A.3 Authentifizierungs- und Autorisierungsinfrastruktur (AAI) . . 133
A.4 Referenzimplementierung 135
A.5 Szenario 1: Synchrone Replikation aus Systemsicht 141
A.6 Vergleich der Replikationsszenarien aus Systemsicht 147
A.7 Szenario 5+6: Asynchrone Replikation aus Nutzersicht 150
A.8 Vergleich der Replikationsszenarien aus Nutzersicht 160
A.9 Auswirkungen steigender Datenmengen 166

1 Einleitung

1.1 Der neue Forschungszweig der Digital Humanities

In den letzten Jahren sind Computer ein immer wichtigerer Teil des wissenschaftlichen Arbeitens geworden. Während die klassischen Naturwissenschaften jedoch schon lange von den Vorteilen profitieren, ist die Entwicklung in den Geistes- und Kulturwissenschaften bei weitem noch nicht abgeschlossen. Obwohl die erste geisteswissenschaftliche Computeranwendung bereits im Jahr 1949 entwickelt wurde [43], steigen die Ansprüche an die Technologie noch immer. Die Hauptschwierigkeit stellt die Übertragung der geisteswissenschaftlichen Forschungsfrage in die digitale Welt dar. Diese Aufgabe beinhaltet neue Herausforderungen, die nur in enger Zusammenarbeit von Geisteswissenschaften und Informatik gelöst werden können. Die „Digital Humanities“ verbinden diese zwei so unterschiedlichen Welten.

Die Geistes- und Kulturwissenschaften sind in verschiedene Disziplinen mit jeweils eigenen Fragestellungen, Anliegen und Methodiken unterteilt. Aus diesem Grund ist eine exakte Definition des intuitiv verständlichen Begriffes Digital Humanities nicht möglich, da es zu viele und sehr unterschiedliche Ansichten gibt [88]. Die Schärfung des Begriffes ist Teil eines aktuellen Diskurses zwischen den Disziplinen und verschiedenen Experten.

Aus der Sicht eines Geisteswissenschaftlers können ein Computer oder computergestützte Methoden als Werkzeug für die Verbesserung der Effizienz von wissenschaftlichen Arbeiten genutzt werden. Üblicherweise handelt es sich dabei um die Automatisierung sich wiederholender Prozesse wie beispielsweise das Suchen oder Zählen von Wortformen. Fortgeschrittenere Nutzungsszenarien beinhalten die Kombination von Methoden aus den Geisteswissenschaften und der Informatik, um neue Methoden aber auch neue Forschungsfragen zu entwickeln. Forscherinnen und Forscher der Digital Humanities können daher als Nutzer computergestützter Methoden in den Geisteswissenschaften verstanden werden, wobei diese Methoden die Forschung unterstützen und durch neue Fragen und Möglichkeiten bereichern. Für eine optimale Ausschöpfung des Potentials interdisziplinärer Zusammenarbeit muss eine sichere und zuverlässige Forschungsinfrastruktur entwickelt werden, die kollaboratives Arbeiten erleichtert und den Zugang zu digitalen Technologien ermöglicht. Insbesondere muss eine solche Infrastruktur Komponenten einer zuverlässigen Datenspeicherung zur langfristigen Nachnutzung von Daten und Werkzeugen bereitstellen.

Die europäische Kommission fördert den Aufbau von Forschungsinfrastrukturen, um Anreize zur Verbesserung guter Forschung zu schaffen. Alle damit in Zusammenhang stehenden Aufgaben werden vom European Strategy Forum on Research Infrastructure (ESFRI) [28] wahrgenommen und koordiniert. Charakteristisch für ESFRI-Projekte ist eine mehrjährige Betriebsphase, die nachhaltige Infrastrukturen für oft mehr als zwanzig Jahre erforderlich macht. Im Gegensatz zur großen Vielfalt naturwissenschaftlicher Projekte, sind die Geistes- und Sozialwissenschaften mit lediglich fünf Beiträgen unterrepräsentiert. DARIAH [14], kurz für „Digital Research Infrastructure for the Arts and Humanities“, und CLARIN [95], kurz für „Common Language Resources and Technology Infrastructure“, sind zwei der Projekte, die als nachhaltig förderungswürdig identifiziert wurden und die digitale Forschung in den Geistes- und Kulturwissenschaften unterstützen. CLARIN setzt den Fokus im Bereich der Linguistik, während DARIAH eine Infrastruktur für alle geisteswissenschaftlichen Disziplinen auf internationaler Ebene aufbaut und betreut. Beide Projekte basieren auf nationalen Beiträgen der einzelnen teilnehmenden Länder.

Der deutsche Beitrag DARIAH-DE [64] verfolgt vier Zielrichtungen:

- den Aufbau einer nachhaltigen digitalen Forschungsinfrastruktur in enger internationaler Kooperation,
- die Unterstützung von digitalen Methoden in der geisteswissenschaftlichen Forschung,
- die Ermöglichung des Zugriffs auf europäische Forschungsdaten
- und die direkte Unterstützung der Forscherinnen und Forscher.

Um die Projektziele zu erreichen, müssen verschiedene Herangehensweisen für die jeweiligen Zielgruppen gewählt werden:

Wissenschaftler profitieren von einer breiten Menge an Empfehlungen für (Meta-) Datenformate, Werkzeuge oder Prozesse, um das höchstmögliche Maß an Interoperabilität mit anderen Disziplinen zu gewährleisten. Zusätzlich unterstützt DARIAH den Aufbau von virtuellen Forschungsumgebungen, die durch die Forschungsinfrastruktur die Nutzung einer großen Anzahl von Datenquellen ermöglichen und die Kollaboration der Wissenschaftler unterstützen.

Forschungsgemeinschaften und Institutionen werden in den Bereichen Forschung, Lehre, Rechte und Lizenzmanagement unterstützt.

Datenzentren nutzen die Infrastruktur, um ihren Nutzern den Zugriff auf inhaltsreiche Archive zu ermöglichen.

Eine große Herausforderung für jede Forschungsinfrastruktur ist eine langfristige und nachhaltige Erhaltung von Forschungsdaten aus einer Vielzahl von Disziplinen mit ihren speziellen Anforderungen. Bis heute konnte jedoch kein allgemein anerkannter Konsens über eine einheitliche Vorgehensweise zur Erhaltung digitaler Informationen für die Zukunft erzielt werden. Auch DARIAH arbeitet an Strategien für die erneute Nutzung und Erhaltung von digitalen Objekten, da die Diversität der geistes- und kulturwissenschaftlichen Disziplinen auf andere Forschungszweige übertragbare Erkenntnisse verspricht. Institutionen und Forschungsgemeinschaften nutzen auf Grund ihrer Heterogenität unterschiedliche Standards, Protokolle und Methoden, die kontinuierlich weiterentwickelt werden. Zusätzlich werden in den nächsten Jahren neue Standards und Methoden entstehen, deren Integration möglich sein muss. Datenzentren müssen mit den Auswirkungen von wechselnder Hardware und Software umgehen können, insbesondere muss die langfristige Unterstützung veralteter Geräte, Medien, Datenformate, disziplinspezifischer Standards und Zugriffsprotokolle gewährleistet werden.

Die nachfolgenden Beispiele illustrieren die Anfragen, mit denen DARIAH täglich in Berührung kommt:

- Ein musikalisches Projekt bietet einen kompletten Überblick über die Arbeit eines Komponisten, bestehend unter anderem aus Partituren, Briefen und Aufnahmen von Orchestern.

- Ein Forscher der Judaistik analysiert einen alten jüdischen Friedhof. Zu diesem Zweck muss er Inschriften übersetzen und auf Karten und Chroniken verschiedener Jahrzehnte zugreifen.

- Ein Digitalisierungsprojekt arbeitet an der virtuellen Rekonstruktion einer mittelalterlichen Bibliothek, deren Bestand auf der ganzen Welt verstreut ist.

- Ein Archäologe rekonstruiert Gebäude virtuell aus ihren Überresten. Die gewonnenen Erkenntnisse und Daten aus Ausgrabungen werden zur Erstellung von 3D-Modellen der Landschaft und Gebäude genutzt.

Die Gemeinsamkeit aller vorgestellten Forschungsvorhaben ist die Notwendigkeit eines zugreifbaren, zuverlässigen Langzeitspeichers für ihre Daten. Die konkreten Anforderungen können allerdings deutlich voneinander abweichen. Die Daten der Projekte und Forscher unterscheiden sich in ihrer Größe (von einigen wenigen Kilobytes für einen Brief in einer Textdatei bis zu vielen Gigabytes für eine Filmaufnahme einer Oper), Menge (von einigen Bilddateien eines seltenen, wertvollen Manuskripts bis zu mehreren Millionen Bilddateien einer gesamten Bibliothek) und Typen mit einer Vielzahl unterschiedlicher Formate für Text, Bild, Audio und Video. Diese Aufzählung gibt keinen kompletten Überblick, illustriert aber anhand der ausgewählten Beispiele die Heterogenität der geisteswissenschaftlichen Forschungsdaten in DARIAH und allgemein den Digital Humanities. Zusätzlich erzeugen und analysieren geisteswissenschaftliche Disziplinen heutzutage stetig wachsende Datenströme. Zumindest Teile ihres Forschungsprozesses werden immer datenintensiver und müssen durch die entstehenden Forschungsinfrastrukturen unterstützt werden.

1.2 Steigende Datenmengen und Big Data

Die Problematik einer langfristigen Speicherung von Forschungsdaten wird im Kontext einer stetig steigenden Datenmenge wieder in den Fokus gerückt und verstärkt. In den Naturwissenschaften werden durch stark verbesserte Methoden der Datenakquisition, -prozessierung und -speicherung immer größere Datenraten ermöglicht. Auch in den Geistes- und Kulturwissenschaften steigen die verwendeten Datenmengen beispielsweise durch groß angelegte Digitalisierungsprojekte, den heutzutage ermöglichten Zugriff auf einen großen Bestand an Forschungsdaten oder neue Datenquellen wie die sozialen Medien.

Giaretta et al. [38] identifizieren gleichzeitig einen Mangel an skalierbaren Infrastrukturen für die effiziente Planung und Anwendung von Strategien zur Preservation für große und heterogene Datensammlungen. Eine Kommission der führenden Datenexperten Europas [100] bezeichnet die langfristige Erhaltung von Daten ebenfalls als eine der wichtigsten Herausforderungen für die nahe Zukunft. Zum einen stellt sich die Frage, wie Daten bei ständigem Wandel der Medien zur Datenspeicherung erhalten werden sollen. Zusätzlich ist der Sinn einer heutigen Speicherung zu beleuchten, wenn die Daten in einem Jahrhundert zerstört, beschädigt oder zu schwierig zu benutzen sind. Zum anderen muss das Thema des Schutzes der Datenintegrität aufgegriffen

werden, da das absichtliche oder unabsichtliche Ändern oder Beschädigen von digitalen Dateien einfach möglich ist.

Der häufig in diesem Zusammenhang verwendete Begriff „Big Data“ beschränkt sich jedoch nicht allein auf die Datenmenge. Zwar ist Big Data nicht eindeutig definiert, die verschiedenen Aspekte des Themas werden jedoch meist mit Hilfe von vier Schlagworten beschrieben [91]:

Volume beschreibt die Menge der Daten, üblicherweise werden hier Terabyte, Petabyte oder auch Exabyte angenommen.

Variety beschreibt die mögliche Vielfalt der Form der Daten, beispielsweise Text-, Bild- und Videodateien oder auch strukturierte und unstrukturierte Daten.

Velocity beschreibt die nötige unterschiedliche Verarbeitungsgeschwindigkeit der Daten, beispielsweise das Auslesen der Daten einer Hochgeschwindigkeitskamera in Echtzeit bei wissenschaftlichen Experimenten im Gegensatz zur asynchronen Kalibrierung digitalisierter Objekte.

Veracity beschreibt die Wahrhaftigkeit der Daten oder die Unsicherheit, welcher Anteil der verwendeten Daten korrekt ist.

Mit **Value** wird oft noch ein fünfter Aspekt hinzugefügt, der sich mit dem Wert der Daten beschäftigt. Die Bedeutung eines Digitalisats eines zerstörten Originals oder von Daten, die in jahrelanger Projektarbeit gewonnen wurden, ist sicherlich höher einzuschätzen als ein Entwurf eines Textes. Der Wert von Daten wird ebenfalls in Kapitel 6.4 zur Empfehlung von Maßnahmen zur langfristigen Erhaltung der Daten herangezogen. Die verschiedenen Wissenschaftsdisziplinen priorisieren die Schlagworte allerdings unterschiedlich. Historisch gesehen ist der Fokus im Falle der Geistes- und Kulturwissenschaften bei Variety und Value zu sehen, aber auch der Bereich Volume gewinnt immer mehr an Bedeutung.

1.3 Langzeitarchivierung von Forschungsdaten

Durch die entstehenden Forschungsinfrastrukturen bieten sich für geisteswissenschaftliche Forscher neue Möglichkeiten, kollaborativ in einem großen Verbund gemeinsame Fragestellungen zu bearbeiten. Editoren ermöglichen die

zeitgleiche Erstellung und Bearbeitung von Texten, vielfältige Annotationswerkzeuge erlauben das Annotieren von Texten, Bildern oder Videos. Die auf diese Weise entstehenden Forschungsdaten sollten für eine sichere Speicherung ebenfalls in einer Forschungsinfrastruktur abgelegt werden. Die Nutzung in einer kollaborativen Arbeitsumgebung stellt jedoch auch neue Herausforderungen an eine Speicherinfrastruktur. Forscherinnen und Forscher unterschiedlicher Herkunft arbeiten zusammen und benötigen verteilte Speicherpunkte, die einen performanten Zugriff auf hochvolatile Daten erlauben. Ebenso muss der Zugriff weltweit möglich sein und das System in einem Dauerbetrieb stabil zur Verfügung stehen, um unterschiedliche Zeitzonen zu berücksichtigen.

Auch innerhalb einer Forschungsinfrastruktur ist die langfristige Speicherung von Forschungsdaten ein komplexer Ablauf, der an vielen Stellen gefährdet und fehleranfällig ist. Für analoge Medien haben vor allem Archive und Bibliotheken belastbare Methoden und Verfahren entwickelt, um eine langfristige Erhaltung zu gewährleisten. Die Erhaltung digitaler Medien ist jedoch ein aktuelles Forschungsthema. Viele bekannte Methoden lassen sich nur bedingt übertragen und zusätzlich müssen komplett neue Herausforderungen gelöst werden. Eine Übersicht und Kategorisierung möglicher Schwachstellen und Gefahren für die digitale Langzeiterhaltung ist in Tabelle 1.1 dargestellt. Eine gesamtheitliche Betrachtung ist schwierig, für einzelne Teilaspekte wurden jedoch bereits Fortschritte erzielt. Beispielsweise wird daran gearbeitet, die Haltbarkeit moderner Speichermedien bei mindestens gleichbleibender Informationsdichte zu erhöhen, um einer drohenden Hardwareobsoleszenz zu begegnen. Ein weiteres Beispiel ist die Nutzung ausgefeilter Replikationsmethoden zur Vermeidung von Datenverlusten bei lokalen Naturkatastrophen oder Hardwarefehlern. Eine 100%-ige Sicherheit der Forschungsdaten kann jedoch in keinem Fall gewährleistet werden. Durch den Einsatz geeigneter Infrastrukturen lässt sich die Wahrscheinlichkeit für Datenverlust jedoch im Vergleich zu manuellen Lösungen deutlich verringern.

In Anlehnung an die Beschreibung von Big Data (vgl. Kapitel 1.2) kann das Problem der Langzeitarchivierung in den Geistes- und Kulturwissenschaften folgendermaßen beschrieben werden:

Variety Forschungsdaten jeglicher Herkunft sind charakterisiert durch eine starke Disziplinabhängigkeit und eine große Vielfalt. Die Geistes- und Kulturwissenschaften im Speziellen decken eine besonders große Bandbreite wissenschaftlicher Disziplinen ab. Aus diesem Grund variieren

Schwachstellen	Prozess	Softwarefehler Softwareobsoleszenz
	Daten	Medienfehler Medienobsoleszenz
	Infrastruktur	Hardwarefehler Hardwareobsoleszenz Kommunikationsfehler Netzwerkfehler
Gefahren	Katastrophen	Naturkatastrophen Menschliche Bedienungsfehler
	Angriffe	Interne Angriffe Externe Angriffe
	Management	Wirtschaftliche Ausfälle Organisatorische Ausfälle
	Gesetzgebung	Rechtliche Veränderungen Rechtliche Voraussetzungen

Tabelle 1.1: Übersicht verschiedener Schwachstellen und Gefahren der digitalen Langzeiterhaltung [10].

die entstehenden Forschungsdaten deutlich in Form, Inhalt und Größe. Eine Infrastruktur zur Langzeitarchivierung muss in der Lage sein, die Heterogenität der Daten und Anforderungen abzudecken.

Value Geisteswissenschaftlichen Forschungsdaten sind äußerst wertvoll. Eine Infrastruktur zur Langzeitarchivierung muss das Risiko eines Datenverlustes soweit wie möglich minimieren, um das kulturelle Erbe für zukünftige Generationen zu bewahren.

Volume Auch in den Geistes- und Kulturwissenschaften entstehen durch neue Technologien und Methoden stetig steigende Datenmengen. Aus diesem Grund muss eine Infrastruktur zur Langzeitarchivierung skalieren, um auf neue Entwicklungen reagieren zu können.

Den maßgeblichen Aspekt stellt die gewünschte langfristige Speicher- und Nutzungsdauer dar, die von einer Infrastruktur ermöglicht werden muss.

Nachhaltigkeit Eine Infrastruktur für die Geistes- und Kulturwissenschaften muss eine nachhaltige und langfristige Speicherung der Forschungsdaten unabhängig von Technologiewechseln ermöglichen. Entsprechend der Archivierung analoger Forschungsobjekte sollen auch ihre digitalen Gegenstücke noch in Jahrzehnten und Jahrhunderten verwendbar sein, insbesondere wenn das Original zwischenzeitlich verloren gegangen ist oder zerstört wurde.

1.4 Erkenntnisse und Forschungsfragen

Mit den Geisteswissenschaften und der Informatik wurden zwei der unterschiedlichsten Disziplinen und ihre spezifischen Denkweisen zu einem neuen Bereich interdisziplinärer Forschung zusammengeführt. Durch geschickte Nutzung und Kombination der Stärken beider Disziplinen entstehen zum einen neue, innovative Werkzeuge und Methoden zur Beantwortung von bestehenden Forschungsfragen, zum anderen werden gänzlich neue Forschungsthemen aufgeworfen. DARIAH, „Digital Research Infrastructure for the Arts and Humanities“, etabliert eine europäische Forschungsinfrastruktur zur optimalen Unterstützung der geistes- und kulturwissenschaftlichen Forscher. Bedingt durch die Forschungstradition und die Vielzahl der betrachteten Disziplinen ergeben sich nach Analyse des geisteswissenschaftlichen Forschungsprozesses spezifische Anforderungen an eine Datenspeicherung und die zugehörige Infrastruktur:

- Sehr heterogene Daten müssen langfristig gespeichert werden.
- Die Integrität der Daten (Bitstreamebene) muss sichergestellt werden.
- Besonderer Fokus liegt auf der (technologischen) Nachhaltigkeit.
- Die Kompatibilität mit anderen Komponenten und die Integration in weitere Infrastrukturen muss gewährleistet werden.

Im Spannungsfeld der aufgeworfenen Anforderungen lassen sich zwei konsekutive Forschungsbereiche identifizieren, die in der vorliegenden Dissertation betrachtet werden.

Wie sieht eine Architektur zur archivalischen Speicherung heterogener Forschungsdaten aus?
Um die Langfristigkeit und Nachhaltigkeit einer Speicherung zu ermöglichen,

müssen Technologiewechsel in der Speicherinfrastruktur berücksichtigt und eine hohe Flexibilität gewährleistet werden. In der Architektur sollte zusätzlich eine generische Migration unabhängig von der verwendeten Speicherinfrastruktur vorgesehen sein. Zu diesem Zweck müssen die Speicherkomponenten abstrahiert und Interoperabilität auf Speicherebene definiert werden.

Wie lässt sich Zuverlässigkeit in Systemen zur Bit Preservation quantifizieren?
Für eine langfristige nachhaltige Speicherung in einer modularen Infrastruktur muss die Zuverlässigkeit zur Abschätzung der Risiken einer Speicherung quantifiziert werden. Wichtig ist die Wahl einer geeigneten Metrik, die Zuverlässigkeit im Kontext der Langzeitarchivierung adäquat beschreibt. Auf diese Weise ist es möglich, verschiedene Speicherstrategien zu vergleichen, etablierte Maßnahmen zur Bit Preservation zu untersuchen und bei der Wahl einer passenden Strategie zu unterstützen. Konkret lassen sich die folgenden Teilfragen formulieren:

- Welche Auswirkung hat mehrfache Replikation auf die Zuverlässigkeit?
- Welche Auswirkung hat der Einsatz von Prüfsummen an unterschiedlichen Stellen auf die Zuverlässigkeit?
- Welche Auswirkung haben steigende Datenmengen auf die Zuverlässigkeit?

1.5 Methodisches Vorgehen und Ergebnisse der Arbeit

Zur Beantwortung der formulierten Forschungsfragen wurde die folgende Vorgehensweise gewählt:

1. Anforderungsanalyse geisteswissenschaftlicher Forscherinnen und Forscher in Bezug auf langfristige Speicherung und Nachhaltigkeit (repräsentative Abdeckung der Disziplinen durch einen großen, heterogenen Forschungsverbund)
2. Untersuchung der führenden europäischen Forschungsprojekte und weit verbreiteten Repositorien im Bereich Preservation mit Schwerpunkt Konzepte und Funktionalitäten zur Bit Preservation

3. Definition einer Standardarchitektur zur Bit Preservation, der Architektur zur archivalischen Speicherung heterogener Forschungsdaten (Identifikation erforderlicher Schnittstellen und Funktionsbereiche zur Modularisierung, Generalisierung benötigter Funktionalitäten)

4. Nachweis der Umsetzbarkeit durch Erstellung einer Referenzimplementierung und Evaluierung der entwickelten Architektur sowie der inkludierten generischen Konzepte mit Hilfe eines repräsentativen Anwendungsfalls

5. Definition einer Metrik (Wahrscheinlichkeit unentdeckter Fehler) zur Bestimmung der Zuverlässigkeit von Architekturen zur Bit Preservation und Modellierung der Zuverlässigkeit verschiedener Replikations- und Prüfsummenszenarien

6. Quantifizierung der Zuverlässigkeit der Standardarchitektur für verschiedene Anwendungsfälle

7. Ableitung von Empfehlungen und pragmatisch verwendbarer Heuristiken für eine auf Nutzeranforderungen abgestimmte Zuverlässigkeit einer langfristigen Datenspeicherung durch Verschränkung von Architektur und Analytik

Der Hauptbeitrag dieser Arbeit ist die Entwicklung und Erforschung eines neuen Modells zur Quantifizierung der Zuverlässigkeit in Architekturen zur Bit Preservation. Insbesondere wird mit Hilfe des Modells eine methodische Auseinandersetzung mit unentdeckten Fehlern in der Bit Preservation durchgeführt.

Basierend auf der Analyse der heterogenen Anforderungen der geistes- und kulturwissenschaftlichen Forscher wird eine Architektur zur archivalischen Speicherung heterogener Forschungsdaten entwickelt, die generische Speicher-, Preservations- und Migrationskonzepte vereint. Mit der Wahl der Wahrscheinlichkeit unentdeckter Fehler als Metrik wird die Modellierung und Quantifizierung der Zuverlässigkeit im Kontext der Langzeitarchivierung innerhalb der Architektur erlaubt. Für verschiedene Szenarien der Bit Preservation werden geschlossene Ausdrücke für die Wahrscheinlichkeit unentdeckter Fehler in Abhängigkeit der verwendeten Architekturkomponenten und Maßnahmen zur Bit Preservation entwickelt. Auf diese Weise werden unterschiedliche Replikationsszenarien in Bezug auf unentdeckte Fehler verglichen

und die Auswirkungen steigender Datenmengen oder der Verwendung unterschiedlicher Prüfsummen untersucht. Als Mehrwert können basierend auf der Analyse Maßnahmen zur Bit Preservation kategorisiert und Empfehlungen abgeleitet werden, die auf die unterschiedlichen Anforderungen der Daten und der Nutzer an die Zuverlässigkeit einer langfristigen Speicherung abgestimmt sind.

Als Ergebnis wird durch das vorgestellte Modell die Grundlage für eine Quantifizierung und Vergleichbarkeit der Zuverlässigkeit von Architekturen zur Bit Preservation geschaffen sowie ein essentieller Baustein für eine langfristige und nachhaltige Erhaltung heterogener Forschungsdaten entwickelt.

2 Stand der Technik

Der Ursprung der Begriffe Kuration, (Langzeit-) Archivierung und Preservation liegt in der Arbeit der Bibliotheken und Archive, deren Sicht- und Herangehensweisen auch die vorliegende Übersicht prägt. Für die Arbeit im digitalen Kontext mussten die Definitionen an die neuen Anforderungen angepasst werden. Der Hauptunterschied der Definition verschiedener Autoren ist lediglich im Fokus der Ausführungen zu sehen. Digitale Preservation nach ursprünglich Beagrie und Jones bezeichnet eine Reihe von gezielten Aktivitäten, die einen ununterbrochenen Zugriff auf digitale Materialien so lange wie nötig sicherstellen [25]. Digitale Preservation nach der American Library Association kombiniert Policies, Strategien und Aktionen, um den Zugriff auf digitalen Inhalt (sowohl reformatiert als auch ursprünglich digital) unabhängig von Herausforderungen bei Medienfehlern oder Technologiewechseln sicherzustellen. Das Ziel von digitaler Preservation ist das korrekte Rendering von authentisiertem Inhalt im Laufe der Zeit [58]. Im Rahmen der Research Data Alliance Working Group „Data Foundation and Terminology“ wurde die folgende Unterscheidung zwischen den verschiedenen Begrifflichkeiten getroffen [101]:

Kuration Die Aktivität, Daten zu verwalten und ihre Nutzung zu fördern, beginnend beim Zeitpunkt ihrer Erstellung. Kuration stellt sicher, dass Daten passend für heutige Verwendungszwecke sowie auffindbar und wiederverwendbar sind.

Archivierung Eine Kurationsaktivität zur Sicherstellung, dass Daten angemessen ausgewählt, gespeichert und zugreifbar sind und dass die logische und physische Integrität gewährleistet bleibt; beinhaltet sind daher Aspekte der Sicherheit und Authentizität.

Preservation Eine Aktivität innerhalb der Archivierung zur Pflege spezifischer Teile von Daten oder Sammlungen, so dass sie auch bei Technologiewechseln zugreifbar sind und verstanden werden können.

In der Literatur wird häufig zwischen Bit(stream) und funktionaler Preservation oder auch zwischen physischer und logischer Preservation unterschieden ([83], [98], [104]). In der Architektur [5] des Projektes WissGrid [56] werden drei Ebenen unterschieden, die grundsätzlich aufeinander aufbauen, aber auch als unabhängige Einheiten genutzt werden können.

Bitstream Preservation soll sicherstellen, „dass jedes Bit eines Datenobjekts ohne unbeabsichtigte Veränderungen verfügbar ist“ [5]. Als wesentliche Bestandteile einer Bit Preservation werden die Anzahl der Kopien, die Verteilung und die Unabhängigkeit der Kopien (geographisch, organisatorisch, finanziell, technologisch, politisch), die Haltbarkeit der Speichermedien und regelmäßige Integritätstests genannt.

Content Preservation soll sicherstellen, dass „das Objekt auch inhaltlich im Wesentlichen identisch [wiedergegeben werden kann], selbst wenn die ursprüngliche Technik (Formate, Software, Speicherort, etc.) veraltet ist“ [5], häufig zusammengefasst als „Sicherstellung langfristiger Lesbarkeit“. Als wesentliche Bestandteile einer Content Preservation werden persistente Identifikation der Objekte, kontinuierliche Beobachtung der Technologieentwicklung, technische Qualitätskontrollen sowie Erhaltungsmaßnahmen (Formatkonvertierungen / Migration, Bereitstellung von Emulatoren) genannt.

Data Curation „betrachtet die gesamte Lebensdauer und ist die langfristige Erhaltung der intellektuellen Nutzbarkeit und Nachnutzbarkeit“ [5], häufig zusammengefasst als „Sicherstellung langfristiger Interpretierbarkeit“. Als wesentliche Bestandteile einer Data Curation werden Konzeption von Daten und Metadaten, Integration von Funktionalitäten in virtuelle Forschungsumgebungen, Versionierung von Objekten, Pflege der Zugriffsberechtigungen, Bewertung der Aufbewahrungswürdigkeit, Sammlungsbildung sowie inhaltliche Anreicherungen und Verknüpfungen genannt.

Anhand der genannten wesentlichen Bestandteile werden in den folgenden Kapiteln 2.1 und 2.2 verschiedene Forschungsprojekte und -initiativen sowie Repositorien untersucht und den entsprechenden Ebenen „Bit Preservation“, „Content Preservation“ und „Data Curation“ zugeordnet. Die Kategorisierung in drei Ebenen ist ein verbreiteter Ansatz im Bereich Preservation und erlaubt mittels der Ausdifferenzierung eine detaillierte Analyse der verschiedenen Projekte und Systeme.

Das Open Archival Information System (OAIS) Referenzmodell [85] ist ein internationaler Standard für die Nomenklatur von Archivsystemen. In der vereinfachten Darstellung werden sechs funktionalen Einheiten betrachtet (vgl. Abbildung 2.1). Insbesondere werden je nach Interaktion mit dem

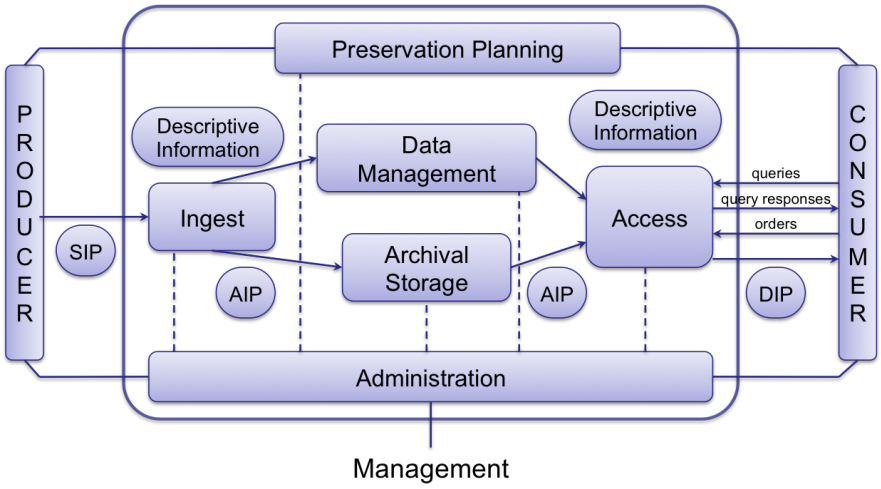

Abbildung 2.1: Open Archival Information System Referenzmodell [85] (SIP: Submission Information Package, AIP: Archival Information Package, DIP: Dissemination Information Package).

System verschiedene Informationspakete, Container aus Daten und Metadaten, unterschieden. Das „Submission Information Package“ (SIP) wird vom Produzenten der Daten an das System gesendet. Aus einem oder mehreren SIP(s) wird das „Archival Information Package“ (AIP) gebildet, das innerhalb des Systems verwaltet wird. Als Antwort auf eine Anfrage werden ein oder mehrere AIP(s) zu einem „Dissemination Information Package“ (DIP) zusammengefügt und dem Konsumenten zur Verfügung gestellt.

2.1 Forschungsprojekte im Bereich Preservation

Im europäischen Umfeld gibt es eine Vielzahl von Projekten, die im Bereich Preservation angesiedelt sind. Im Folgenden werden die wichtigsten Projekte aufgeführt und an Hand der beschriebenen Ebenen „Bit Preservation“, „Content Preservation“ und „Data Curation“ mit ihren wesentlichen Bestandteilen analysiert. Ein besonderer Fokus liegt auf Grund der formulierten Forschungsfragen auf dem Bereich Bit Preservation. Tabelle 2.1 zeigt eine Übersicht der Forschungsprojekte inklusive einer detaillierten Einordnung ihrer Aktivitäten.

		APARSEN	CASPAR	KEEP	PLANETS	Preserv2	SCAPE	SHAMAN
Data Curation	Konzeption Daten / Metadaten	o		x			x	x
	Integration in virtuelle Forschungsumgebung							
	Versionierung							(x)
	Pflege Zugriffsberechtigungen	x	x				x	x
	Bewertung Aufbewahrungswürdigkeit	o	(x)					
	Sammlungsbildung		(x)					x
	inhaltliche Anreicherung / Verknüpfung		x					
Content Preservation	persistente Identifikation	x	x				x	x
	Beobachtung Technologieentwicklung	(x)		x	x	x	x	x
	technische Qualitätskontrollen	x / o			x			x
	Formatkonvertierung / Migration	o	x	x	x	x	x	x
	Bereitstellung von Emulatoren	o		x	x	x		
Bitstream Preservation	Anzahl der Kopien		o					o
	Verteilung / Unabhängigkeit der Kopien		o					o
	Haltbarkeit der Speichermedien		o					
	regelmäßige Integritätstests		o		(x)			o

Tabelle 2.1: Übersicht der betrachteten Forschungsprojekte mit Einordnung der abgedeckten Bereiche der Langzeitarchivierung (x: wird behandelt; (x): wird teilweise behandelt; o: wird von anderen Projekten oder Komponenten übernommen; kein Eintrag: wird nicht behandelt).

APARSEN [4], kurz für „Alliance Permanent Access to the Records of Science in Europe Network“, ist ein Projekt aus dem siebten Forschungsrahmenprogramm der Europäischen Union. Besonderer Fokus liegt auf der Koordinierung von Preservationsaktivitäten auf europäischer Ebene, um die langfristige Zugreifbarkeit und Nutzbarkeit von digitalen Informationen und Daten zu ermöglichen. Die Bündelung verschiedener Aspekte und Ansätze in die Oberthemen Vertrauen, Nachhaltigkeit, Nutzbarkeit und Zugriff soll zu einer gemeinsamen Vorgehensweise und Vision führen. APARSEN lässt sich den Bereichen Data Curation und Content Preservation zuordnen. Die gesammelten Informationen stützen sich größtenteils auf Ergebnisse vorausgegangener Projekte.

CASPAR [37], kurz für „Cultural, Artistic and Scientific Knowledge for Preservation, Access and Retrieval“, ist ein Projekt aus dem sechsten Forschungsrahmenprogramm der Europäischen Union. Als Hauptziele werden die Entwicklung von Infrastrukturkomponenten zur Unterstützung der digitalen Preservation auf Basis des OAIS Konzeptes und die Validierung der Infrastruktur genannt. Ein besonderer Fokus liegt auf der Betrachtung von gemäß OAIS Nomenklatur „Representation Information“ zur Erhaltung der Verständlichkeit und Nutzbarkeit von digitalen Objekten. CASPAR deckt Teile der Data Curation und Content Preservation ab. Es wird vorausgesetzt, dass Funktionalitäten der Bit Preservation von der Speicherkomponente übernommen werden. Insbesondere geht CASPAR davon aus, dass Objekte oder gemäß OAIS Nomenklatur Archival Information Packages (AIPs) verwaltet werden. Datenmigrationen werden als gegeben vorausgesetzt und eher Transformationen betrachtet.

KEEP [70], kurz für „Keeping Emulation Environments Portable“, ist ein Projekt aus dem siebten Forschungsrahmenprogramm der Europäischen Union. Ziel des Projekts ist die Entwicklung von Emulationsdiensten zur Ermöglichung des Renderings von statischen und dynamischen digitalen Objekten. Es werden Dienste für den Zugriff, die Manipulation und die Speicherung von digitalen Objekten entwickelt. Mit Hilfe der Emulationsdienste kann entweder die originale Umgebung der Objekte reproduziert oder eine Migration in eine neue Umgebung ermöglicht werden. Der Fokus von KEEP liegt auf der Emulation, die der Content Preservation zuzuordnen ist. Das entwickelte „Transfer Tools Framework“ erlaubt das Einbinden externer Speicherpunkte. Über Funktionalitäten der Bit Preservation werden keine weiteren Aussagen getroffen.

PLANETS [79], kurz für „Preservation and Long-term Access through

Network Services“, ist ein Projekt aus dem sechsten Forschungsrahmenprogramm der Europäischen Union. Ziel des Projekts ist die Bereitstellung einer dienstebasierten Forschungsumgebung, die sich den Herausforderungen der Preservation für digitale Bibliotheken und Archive stellt. Ein besonderer Fokus liegt auf der Erweiterung des OAIS „Preservation Planning“ Moduls [80], insgesamt werden hauptsächlich Aspekte der Content Preservation betrachtet. Analog zu CASPAR werden in PLANETS Objekte oder nach OAIS Nomenklatur Archival Information Packages (AIPs) verwaltet. Im Kontext der Migrationen werden lediglich Formatmigrationen betrachtet. Dateien werden an unterschiedlichen Stellen wie beispielsweise Repositoriums- oder Dateisystemen gespeichert und ihre Prüfsummen berechnet. Weitere Aussagen zur Bit Preservation werden nicht getroffen.

Preserv2 [42], kurz für „Preservation Eprint Services“, ist ein nationales Projekt aus dem Vereinigten Königreich. Es wird die Erhaltung von Daten in institutionellen Repositorien untersucht und der Fokus auf die Verwaltung von Speicher, Daten und Dateiformaten gelegt. Preserv2 fokussiert sich auf Teile der Content Preservation, insbesondere wurde die Verwaltung von Dateiformaten mit automatischer Speicherverwaltung verbunden. Der Austausch von digitalen Objekten wird mittels OAI-ORE [52] ermöglicht und so eine Interoperabilität zwischen Repositorien erreicht. Verschiedene Speicherpunkte sind per Kontroller ansprechbar, dieser ist allerdings nicht generisch verwendbar sondern auf die verwendete Software zugeschnitten. Auf dieser Ebene werden keine weiteren Aussagen über Funktionalitäten der Bit Preservation und Migration getroffen.

SCAPE [59], kurz für „Scalable Preservation Environments“, ist ein Projekt aus dem siebten Forschungsrahmenprogramm der Europäischen Union. Ziel des Projekts ist die Entwicklung von skalierbaren Werkzeugen, Diensten und Infrastrukturen für die effiziente Planung und Ausführung von Preservationstrategien für großskalige, heterogene Sammlungen komplexer digitaler Objekte. SCAPE ist dem Bereich Data Curation und Content Preservation zuzuordnen, ein Fokus liegt analog zu PLANETS im Bereich „Preservation Planning“, so dass Komponenten nachgenutzt werden. Zur Bit Preservation werden keine weiteren Aussagen getroffen.

SHAMAN [44], kurz für „Sustaining Heritage Access through Multivalent Archiving“, ist ein Projekt aus dem siebten Forschungsrahmenprogramm der Europäischen Union. Es wurde die Langzeiterhaltung großer Mengen digitaler Objekte in einer verteilten Umgebung untersucht. Zu diesem Zweck wurden Werkzeuge sowie ein verifizierbares, offenes und erweiterbares „Preservation

Framework“ entwickelt. SHAMAN betrachtet Teile der Data Curation und Content Preservation. Es wird vorausgesetzt, dass Funktionalitäten der Bit Preservation von der Speicherkomponente, Grid oder Cloud, übernommen werden. Die Konzentration liegt auf konkreten Anwendungsfällen und Änderungen von Hardware, Applikationen und Formaten. Als Strategie werden spezifische Komponenten genutzt, auf Speicherebene wird beispielsweise das Datenmanagementsystem iRODS [73] verwendet.

Insgesamt stellen die Forschungsprojekte meist Werkzeuge zur Bearbeitung spezieller Preservationsaufgaben zur Verfügung, die Teil von Preservationsumgebungen sind oder über verschiedene Schnittstellen eingebunden werden können. Funktionalitäten im Bereich der Bit Preservation werden entweder nicht behandelt oder auf spezifische Speichersysteme ausgelagert. Auf Grund der Konzeption der Forschungsprojekte ist daher auf dieser Ebene keine allgemeine Aussage über die Sicherstellung der Datenintegrität leistbar.

2.2 Archivierungskonzepte und Repositorien

Analog zu der Betrachtung der Forschungsprojekte werden im Folgenden weit verbreitete Repositorien an Hand der beschriebenen Ebenen „Bit Preservation“, „Content Preservation“ und „Data Curation“ mit ihren wesentlichen Bestandteilen untersucht. Auf Grund der formulierten Forschungsfragen wird ebenfalls der Fokus auf den Bereich Bit Preservation gelegt. Tabelle 2.2 zeigt eine Übersicht der Repositorien inklusive einer detaillierten Einordnung ihrer Aktivitäten.

Das aDORe Repositorium [13] aus dem Jahr 2005 ist eines der früh entwickelten Repositorien. Es werden nur wenige Bereiche der Langzeitarchivierung abgedeckt und der Fokus auf die Interoperabilität und Föderation von Repositorien gelegt.

Archivematica [94] ist ein OAIS-kompatibles Repositorium, das komplexe Abläufe mit Hilfe von sogenannten Mikroservices abbilden kann. Viel Wert wird auf die Integration von Preservationsstandards und Best Practices gelegt. Funktionalitäten zur Bit Preservation werden nur eingeschränkt zur Verfügung gestellt. Prüfsummen werden beim Ingest berechnet, die regelmäßige Überprüfung sollte soweit möglich von der Speicherkomponente ansonsten von einem Daemon übernommen werden.

		aDORe	Archivematica	dSpace	DuraCloud	Fedora	Preservica	Rosetta
Data Curation	Konzeption Daten / Metadaten	x	(x)	x		x	x	x
	Integration in virtuelle Forschungsumgebung							
	Versionierung	x				x		x
	Pflege Zugriffsberechtigungen			x		x	x	(x)
	Bewertung Aufbewahrungswürdigkeit							
	Sammlungsbildung	x	(x)	x	(x)	x	x	x
	inhaltliche Anreicherung / Verknüpfung		x					x
Content Preservation	persistente Identifikation		(x)	x		x		x
	Beobachtung Technologieentwicklung			(x)				
	technische Qualitätskontrollen		x	(x)		x	x	(x)
	Formatkonvertierung / Migration		x	(x)		(x)	x	x
	Bereitstellung von Emulatoren							
Bitstream Preservation	Anzahl der Kopien		(x)		x	x	x	x
	Verteilung / Unabhängigkeit der Kopien				x / (x)		x	
	Haltbarkeit der Speichermedien							
	regelmäßige Integritätstests		(x)	x	x	x	x	x

Tabelle 2.2: Übersicht der betrachteten Repositorien mit Einordnung der abgedeckten Bereiche der Langzeitarchivierung (x: wird behandelt; (x): wird teilweise behandelt; o: wird von anderen Projekten oder Komponenten übernommen; kein Eintrag: wird nicht behandelt).

Das dSpace Repositorium [83] zielt auf die Unterstützung von Erfassung, Ingest, einfachem Zugriff und langfristiger Erhaltung von Daten. Funktionalitäten zur Bit Preservation werden durch Metadatenerfassung unterstützt.

DuraCloud [61] basiert auf Cloud Infrastruktur und kann mehrere Anbieter ansprechen. Die Funktionalitäten konzentrieren sich auf den Bereich Bit Preservation, die verwendeten Verfahren und Methodiken sind jedoch auf die Cloud eingeschränkt.

Fedora [86], [51], kurz für „Flexible Extensible Digital Object Repository Architecture“, ist ein erweiterbares Framework zur Speicherung, zum Management und zur Weitergabe von komplexen Objekten und ihren Beziehungen. Migrationen werden jedoch nur zwischen verschiedenen Versionen des Repositoriums unterstützt. Auf der Fedora Technologie basieren eine Reihe anwendungsspezifischer Entwicklungen wie beispielsweise eSciDoc [74], Hydra [6] oder RODA [11]. Funktionalitäten der Bit Preservation werden nur teilweise abgedeckt.

Preservica [90] ist eine kommerzielle Plattform für digitale Preservation und Zugriff, die OAIS kompatible Workflows zur Verfügung stellt. Es werden Teile aus allen drei Bereichen Data Curation, Content Preservation und Bit Preservation abgedeckt, die Funktionalitäten der Bit Preservation werden hauptsächlich durch den verwendeten Cloudspeicher bereitgestellt. Die Premium Variante erlaubt zusätzlich die Einbindung weiterer Systeme durch anpassbare Adapter.

Rosetta [48], [29] ist ein kommerzielles Preservationssystem basierend auf OAIS und weiteren Standards. Ein Fokus liegt auf der langfristigen Speicherung von digitalen Objekten verschiedener Formate und Strukturen. Über eine Speicherabstraktionsschicht kann unterschiedliche Hardware angesprochen werden, es werden jedoch meist Festplatten verwendet. Funktionalitäten der Bit Preservation werden nur teilweise angeboten.

Insgesamt wird auch bei den Repositorien der Bereich Bit Preservation nur unzureichend abgedeckt, zusätzlich meist durch sehr spezifische Ansätze, die nur schwer übertragbar sind. Aussagen zur Datenintegrität sind auf dieser Ebene beispielsweise durch Vereinbarungen zur Dienstgüte (Service Level Agreements - SLAs) mit den Speicheranbietern nur eingeschränkt leistbar.

2.3 Methodiken der Bit Preservation

Rosenthal [76] fasst Erkenntnisse zum Umgang mit Daten im Kontext der Bit Preservation in drei Punkten zusammen, die die Grundlage vieler Preservationsaktivitäten bilden und in der Diskussion der Ergebnisse der Arbeit erneut aufgegriffen werden:

- Je mehr Kopien erzeugt werden, desto sicherer die Daten.
- Je unabhängiger die Kopien sind, desto sicherer die Daten.
- Je häufiger die Kopien überprüft werden, desto sicherer die Daten.

Grundsätzlich lassen sich Methodiken und Verfahren der Bit Preservation wie in der WissGrid-Einordnung in mindestens vier Teilbereiche einteilen.

Haltbarkeit der Speichermedien
Analoge Medien wie Steintafeln oder mittelalterliche Manuskripte haben teilweise die letzten Jahrtausende bzw. Jahrhunderte ihre Integrität bewahrt, der verloren gegangene Anteil lässt sich jedoch nur schwer abschätzen. Digitale Medien als alternative Informationsspeicher besitzen im Vergleich zwar eine deutlich höhere Informationsdichte, die Haltbarkeit hat sich im Laufe der Zeit jedoch deutlich verringert [19]. Hersteller von beispielsweise Festplatten propagieren eine Mindestlebensdauer von fünf Jahren, die tatsächliche Nutzungsdauer bei täglichen Gebrauch liegt auf Grund verschiedener Umwelteinflüsse jedoch häufig darunter. Eine Untersuchung des CERN [66] zeigt zusätzlich eine im Vergleich zu Herstellerangaben erhöhte Fehlerrate im Bereich von 10^{-7}. Aktuelle Entwicklungen zeigen erste Erfolge für längerlebige Medien, doch unter anderem auf Grund der Kosten werden auch in den nächsten Jahre weiterhin Festplatten und Bandsysteme verwendet werden.

Architekturen zur Bit Preservation müssen daher wechselnde Speichermedien berücksichtigen und verwerten können.

Anzahl der Kopien + Verteilung / Unabhängigkeit der Kopien
Die am weitesten verbreitete Form einer Kopie einer Datei ist das Backup. Bei diesem Verfahren werden zusätzliche Dateien vorzugsweise in Bandsystemen abgelegt, um bei Datenverlusten das Original wiederherstellen zu können. Backups sind für eine langfristige und zuverlässige Bit Preservation jedoch nicht ausreichend, beispielsweise fehlen regelmäßige Integritätsüberprüfungen

zur Entdeckung von Datenkorruption. Auch eine Replikation von Dateien, eine exakte Kopie von Dateien in anderer Umgebung (andere Software, andere Hardware, andere geographische Lokation), ist nur mit regelmäßigen Integritätsüberprüfungen sinnvoll. An dieser Stelle entstehen als zusätzliche Probleme, die Konsistenz der replizierten Dateien zu gewährleisten und einen performanten Zugriff auf die Repliken zu ermöglichen. Insgesamt wird meist davon ausgegangen, dass eine höhere Anzahl an Kopien und eine weite Verteilung bzw. Unabhängigkeit (Hardware, Software, etc.) der Kopien mehr Sicherheit verspricht. Allerdings führen mehr Kopien automatisch auch zu erhöhtem Speicherplatzbedarf und daraus resultierend erhöhten Kosten.

Architekturen zur Bit Preservation müssen mehrere voneinander unabhängige Kopien der einzelnen Dateien erlauben.

Regelmäßige Integritätsüberprüfungen

Auf Hardwareebene wird standardmäßig ein Cyclic Redundancy Check [68] durchgeführt, um eine Prüfsumme zur Erkennung von Fehlern bei der Datenübertragung und -speicherung zu bestimmen. Bedingt durch die Einfachheit lässt sich diese Methode jedoch leicht manipulieren und Fehler bleiben unerkannt. Durch bekannte Algorithmen wie MD5 [75] oder SHA1 [63] lassen sich fortschrittlichere Prüfsummen berechnen, die jedoch immer noch anfällig für Fehler und Manipulation sind. Eine Übersicht der Fehlerwahrscheinlichkeiten sowie der Länge der einzelnen Prüfsummen ist in Tabelle 4.1 aufgeführt. Algorithmen, die eine bessere Fehlererkennung aufweisen, benötigen deutlich mehr Rechenleistung und -zeit. Gerade bei steigenden Datenmengen wird eine Prüfsummenberechnung auf diese Weise immer aufwändiger und belastet die verwendete Hardware [76]. Zur Zeit am weitesten verbreitet ist auf Grund der moderaten Rechenzeit und der akzeptablen Fehlererkennung eine Prüfsummenberechnung mittels MD5 Algorithmus. Sollen Fehler nicht nur erkannt, sondern auch behoben werden, können fehlerkorrigierende Codes verwendet werden. Auch bei diesem Verfahren steigt die benötigte Rechenleistung und -zeit an, je mehr Fehler man ausgleichen können möchte. Als Beispiel seien Erasure Codes genannt, die eine höhere Zuverlässigkeit als Replikation aufweisen können [67]. Eine ausführlichere Betrachtung der Zuverlässigkeit von Prüfsummen findet sich in Kapitel 4.3.

Architekturen zur Bit Preservation müssen Funktionalitäten zur regelmäßigen Überprüfung der Datenintegrität bereitstellen, die idealerweise verschiedene existierende und zukünftige Algorithmen integrieren.

Es gibt eine Vielzahl bestehender Systeme, die benötigte Funktionalitäten zur Bit Preservation abdecken können. iRODS [73], kurz für „integrated Rule-Oriented Data System“, bildet eine Middleware Schicht zwischen dem Nutzer und dem verwendeten Speichersystem. Unterstützt werden heterogene Umgebungen und jedes als Dateisystem ansprechbare Archiv. dCache [27] bietet ähnliche Funktionalitäten und ist als ein Caching System für häufig genutzte Daten konzipiert, um einen performanten Zugriff auf Bandsysteme zu ermöglichen. Merritt [93] basiert auf modularen Mikroservices, die unabhängig voneinander ein flexibles System passend für verschiedene Anwendergruppen bilden können.

Bei den betrachteten Systemen können Replikation und Datenintegritätsüberprüfungen durch bereitgestellte und konfigurierbare Methoden durchgeführt werden. Bei der Verwendung als monolithisches System ist allerdings die Nachhaltigkeit als besonders kritisch zu bewerten. APARSEN [38] identifiziert zusätzlich die Isolierung der vielen unterschiedlichen Programme und Hilfsmittel der einzelnen Disziplinen als ein Hindernis für inter-institutionelle Preservation und Interoperabilität.

Für Architekturen zur Bit Preservation können die vorgestellten Systeme daher lediglich als Komponenten in Betracht gezogen werden.

Im Allgemeinen wurde oftmals zwischen den zwei folgenden Vorgehensweisen im Bereich der Preservation unterschieden:

Verteilte Replikation Implementierungen der verteilten Replikation basieren meist auf LOCKSS [57], kurz für „Lots Of Copies Keep Stuff Safe“. Es werden mehrere identische, geographisch verteilte Kopien der zu speichernden Dateien erstellt, die Integrität der Dateien regelmäßig überprüft und Fehler behoben. Eine solche verteilte Replikation ist jedoch nicht für den regelmäßigen Zugriff ausgelegt und soll lediglich die Sicherheit bei Datenverlusten im Katastrophenfall gewährleisten. Strategien zur Preservation und Metadaten werden nur eingeschränkt unterstützt.

Repositoriumsmodell Beim Repositoriumsmodell werden lokale Systeme für die Preservation und / oder den Zugriff bereitgestellt. Die Systeme sind fest mit dem Speicher gekoppelt und können durch externe Werkzeuge zur Implementierung von Preservationsstrategien ergänzt

werden. Bei diesem Modell ist die bereitstellende Institution zusätzlich für Backups und Replikation sowie für die Implementierung passender Sicherheits- und Auditmaßstäbe verantwortlich.

Beide Vorgehensweisen zeigen eine starke Festlegung auf eine Technologie, die insbesondere beim Repositoriumsmodell große Kenntnisse der Mitarbeiter voraussetzt. In der Praxis wird die strikte Trennung meist nicht mehr eingehalten, sondern versucht, die Vorteile beider Ansätze zu nutzen. Viele Institutionen haben bereits unterschiedlich umfassende Angebote und Ansätze, die als Mischform zu betrachten sind und oftmals Ergebnisse der Forschungsprojekte im Bereich Preservation nachnutzen:

- Hoppla [87] bietet ein System zur Preservation für kleine Institutionen und Privatnutzer. Es werden eingeschränkte, nach OAIS modellierte Funktionalitäten zur Preservation geboten und mehrere Speichermedien sind nach Vorbild eines LOCKSS-Netzwerkes und des CASPAR Projektes zur Replikation ansprechbar.

- Das Florida Digital Archive [17] ist ein Archiv zur Erhöhung der Sicherheit bei Datenverlusten basierend auf DAITSS [16] und modelliert nach OAIS. Ein zentrales System übernimmt ebenfalls periodische Aktualisierungen, Integritätsüberprüfungen und Medienmigrationen.

- Der Preservation Aware Storage [30] führt eine Kombination von Speicher und Preservation ein. Es werden zusätzliche Informationen zur Provenance und Authentizität im Speicher abgelegt. Der Preservation DataStore bildet basierend auf CASPAR das logische Konzept eines Preservationsobjektes als physikalisches Objekt im Speicher ab.

- Das Danish Bit Repository [47], [62] verspricht eine nachhaltige Bit Preservation unabhängig vom Speicher. In einer geschichteten Architektur werden autonome Speichersäulen, die unabhängig von externen Informationen sind, angeboten. Eine Koordinationsschicht koordiniert den Austausch zwischen Klienten und Speichersäulen. Migrationen werden in diesem Kontext nicht betrachtet. Zusätzlich trägt der Klient die Verantwortlichkeit für die Bit Preservation und kann direkt auf die Speichersäulen zugreifen.

- Die Library of Congress [46] bietet Werkzeuge und Dienste für den digitalen Lebenszyklus von Dateien und Verzeichnissen mittels eines zentralen Systems, das auf Bibliotheksdaten zugeschnitten ist.

Zur Kategorisierung von Preservationsanstrengungen werden verschiedene Konzepte vorgeschlagen. Die NSDA Level of Digital Preservation [69] sollen eine einfache Menge von Richtlinien zum Aufbau sowie zur Verbesserung der Preservation darstellen. Sie definieren vier allgemeine Level basierend auf den Bewertungskriterien „Lokation“, „Beständigkeit der Daten“, „Informationssicherheit“, „Metadaten“ und „Dateiformate“. Eine andere Form der Bewertung ist TAP [72], kurz für „Tiered Preservation Model for Digital Resources“. Digitale Ressourcen werden basierend auf ihrem Ressourcentyp, archivalischer Verantwortlichkeit und erwartetem Preservierbarkeitslevel bewertet und eine passende Strategie zur Preservation zugeordnet.

Einer der wichtigsten Aspekte der Langzeitarchivierung und damit auch der Bit Preservation ist das Vertrauen der Nutzer. Nur wenn die Nutzer von der Zuverlässigkeit beispielsweise eines Repositoriums überzeugt sind, lassen sich Dateien langfristig erhalten. Verschiedene Organisationen haben daher Zertifizierungsverfahren wie beispielsweise das Data Seal of Approval (DSA) [22] oder das Trustworthy Repositories Audit & Certification (TRAC) [3] entwickelt, um die Qualität eines Angebots beurteilen zu können.

Soll die Ausfallwahrscheinlichkeit einer Architektur zur Bit Preservation bestimmt werden, sind verschiedene Vorgehensweisen möglich. Rosenthal [76] sieht Bit Preservation als Blackbox an. Ein Bitstring $S(0)$ wird zum Zeitpunkt $T(0)$ gespeichert, $S(i)$ wird zum Zeitpunkt $T(i)$ gelesen. Die Bit Preservation gilt als erfolgreich, falls $S(0) = S(i)\ \forall i$.

Weit verbreitet ist der Einsatz von Markovketten, um eine Abschätzung der erwarteten Zeit bis zu einem Datenverlust zu erhalten. Markovketten sind gedächtnislos, das heißt ein reparierter Fehler wird äquivalent zu keinem Fehler behandelt und Prüfsummen können nur implizit über die Reparaturrate modelliert werden. Projekte wie PrestoPrime [1] nutzen Markovketten, um Strategien zur Preservation zu vergleichen. Parameter wie Fehlerraten müssen jedoch gewählt werden und bestimmen maßgeblich die Qualität der Quantifizierung.

Alternativ kann die Ausfallwahrscheinlichkeit auch mittels einer Verteilung beschrieben werden. Die Wahl einer passenden Verteilung ist Gegenstand der Diskussion, häufig werden Exponential-, Poisson-, Weibull- oder Paretoverteilungen angenommen. Bei einer ebenfalls möglichen Annäherung der Ausfallwahrscheinlichkeit durch Simulation müssen die benötigten Parameter passend bestimmt werden.

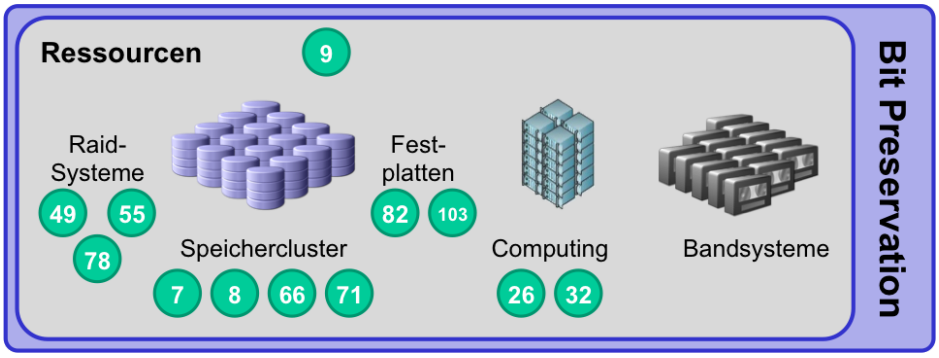

Abbildung 2.2: Übersicht der Ansätze und Untersuchungen im Kontext unentdeckter Fehler in der Bit Preservation mit Angabe der Referenzen.

Im Modell von Baker et al. [9] wird zwischen entdeckten und latenten Fehlern unterschieden, um eine Aussage über die erwartete Zeit bis zu einem Datenverlust zu treffen. Latente Fehler werden nicht direkt nach dem Auftreten erkannt und können beispielsweise durch fehlgeleitete Schreiboperationen, unlesbare Sektoren oder veraltete Datenformate auftreten. Das Modell bezieht zeitliche und räumliche Zusammenhänge der Fehler mit ein und diskutiert Strategien, um einzelne Faktoren zu beeinflussen.

Eine Übersicht weiterer Arbeiten im Bereich von unentdeckten Fehlern ist in Abbildung 2.2 dargestellt. Neben Studien konkreter, großer Speichersysteme ([8], [7], [71], [66]) stehen vor allem RAID-Systeme im Fokus der Untersuchung. Es wird modelliert, welche Auswirkungen unentdeckte Fehler haben [78], wie sie sich bei Datenrekonstruktion verstärken können [55] und welche Effekte verschiedene existierende Schutzmaßnahmen auf die Datenintegrität haben [49]. Beispiele von Entwicklungen, die auf Basis der Untersuchungen bereits explizit Preservationsfunktionalitäten bereitstellen, sind das ZFS-Dateisystem [103] oder auch Speicherkomponenten mit erweiterten Datenintegritätsfunktionalitäten [82]. Im Bereich Computing wurden unter anderem eine Methode zur Erkennung und Korrektur unentdeckter Fehler auf Protokollebene [32] und Algorithmen, die unentdeckte Fehler tolerieren können [26], vorgeschlagen. In den meisten der vorgestellten Ansätze ist die Detektion der unentdeckten Fehler im Fokus, die Fehlerwahrscheinlichkeit der Detektion selber wird jedoch nicht berücksichtigt.

Bei der Quantifizierung von Architekturen zur Bit Preservation mit Hilfe der beschriebenen Modelle ist die Bestimmung von Parametern von zentraler

Bedeutung. Nur durch passend gewählte Werte kann eine möglichst realistische Abschätzung erreicht werden, in vielen Fällen ist jedoch die Messbarkeit der Parameter problematisch. Laut Rosenthal [77] dauern Experimente im Bereich Bit Preservation zu lange und sind zu teuer, um belastbare Aussagen zu erhalten. Besonders kritisch sind in diesem Kontext die latenten oder unentdeckten Fehler, da ihr Auftreten nicht direkt überprüft werden kann.

2.4 Erkenntnisse

Die Vielzahl an (europäischen) Projekten unterstreicht die Bedeutung und Aktualität des Forschungsthemas Langzeitarchivierung bzw. Preservation. Die verschiedenen Teilaspekte werden jedoch unterschiedlich detailliert betrachtet, Bit Preservation wird meist als selbstverständlich vorausgesetzt. Datenmigrationen werden als gegeben betrachtet oder für konkrete Speichersysteme erstellt, Maßnahmen zur Bit Preservation müssen stets neu konzipiert werden. Insgesamt fehlen in diesem Kontext Konzepte, die eine langfristige und disziplinübergreifende Speicher- und Nutzungsdauer möglich machen.

Für eine erfolgreiche Langzeitarchivierung ist die Bestimmung der Zuverlässigkeit eine Grundvoraussetzung zur Auswahl der passenden Strategie. Zu diesem Zweck wird ein generisches Modell zur Bit Preservation benötigt, das eine Quantifizierung der Zuverlässigkeit erlaubt. Von besonderer Bedeutung ist dabei die Betrachtung unentdeckter Fehler, weil sie dem System zur Bit Preservation zunächst verborgen bleiben und Datenverluste erst Jahre später erkannt werden können. Bei einer steigenden Menge an Forschungsdaten wird selbst bei geringen Fehlerwahrscheinlichkeiten das Auftreten von mindestens einem unentdeckten Fehler immer wahrscheinlicher, so dass die Entwicklung einer effektiven Strategie für den Umgang mit dieser Fehlerart notwendig wird. Trotz verschiedener Vorarbeiten im Kontext unentdeckter Fehler fehlt es jedoch insbesondere an einer ganzheitlichen Betrachtung der Auswirkungen von Preservationsmaßnahmen wie verteilter Replikation und Datenintegritätsüberprüfungen.

3 Architektur zur archivalischen Speicherung heterogener Forschungsdaten

Zur Quantifizierung der Zuverlässigkeit wird zunächst eine generische Architektur zur Bit Preservation entwickelt, die als zusätzliche Anforderung generische Speicher-, Preservations- und Migrationskonzepte vereint. Die Beschreibungen und Erkenntnisse der Kapitel 3.1 und 3.2 beruhen auf [92]. Als Nachweis der Umsetzbarkeit und zur Quantifizierung der Architektur wurde eine Referenzimplementierung erstellt, die bereits im Kontext von DARIAH-DE und DARIAH-EU zur Anwendung kommt. Eine detaillierte Beschreibung befindet sich im Anhang. Mit Hilfe des Anwendungsfalls „Virtuelles Skriptorium St. Matthias“ werden Möglichkeiten der Datenmigration exemplarisch evaluiert.

3.1 Anforderungen und Struktur

Geisteswissenschaftliche Forscher stellen vielfältige Anforderungen an ein System zur langfristigen und nachhaltigen Speicherung ihrer Forschungsdaten. Eine Architektur zur archivalischen Speicherung heterogener Forschungsdaten sollte daher generisch ausgelegt sein, um so viele Anwendungsfälle wie möglich erfüllen zu können. Abbildung 3.1 illustriert die grundlegenden Anforderungen, die sich aus den beschriebenen Anwendungsfällen in Kapitel 1.1 sowie den Bedarfen der Fachwissenschaftler ableiten und an eine solche Architektur gestellt werden:

- Für die geisteswissenschaftlichen Forscher ist die langfristige Erhaltung einzelner Dateien ausreichend. Lediglich administrative Metadaten müssen auf der Ebene der Bit Preservation verwaltet werden.
- Es werden heterogene Daten mit unterschiedlichen Größen, Formaten oder Inhalten gespeichert.
- Die geisteswissenschaftlichen Forscher führen hauptsächlich CREATE und READ Operationen durch. UPDATE und DELETE sind möglich, werden aber selten genutzt.
- Mechanismen zur Überprüfung und Sicherstellung der Datenintegrität sind notwendig.

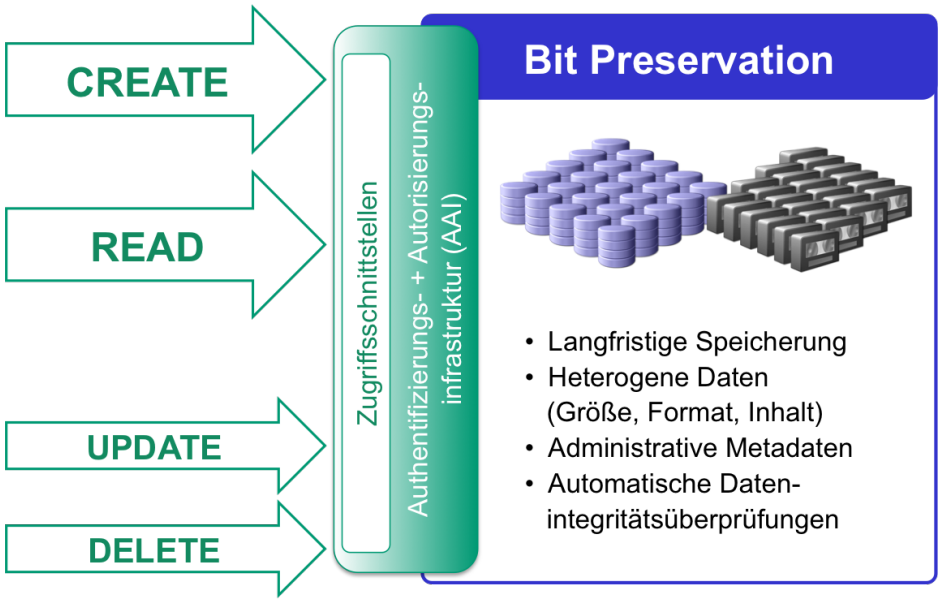

Abbildung 3.1: Anforderungen an eine Architektur zur archivalischen Speicherung heterogener Forschungsdaten.

- Wünschenswert ist ein verteiltes System, das sowohl eine Schnittstelle für die geisteswissenschaftlichen Forscher, als auch eine Schnittstelle für den automatischen hochperformanten Zugriff bietet.
- Unerlaubte Zugriffe werden durch eine Authentifizierungs- und Autorisierungsinfrastruktur (AAI) verhindert.

Abbildung 3.2 zeigt eine Übersicht der Architektur zur archivalischen Speicherung heterogener Forschungsdaten. Basierend auf den beschriebenen geisteswissenschaftlichen Anforderungen sowie den technischen Rahmenbedingungen wie einer leichte Austauschbarkeit von Technologien ist eine modulare Struktur gewählt. Zusätzlich unterscheidet die Architektur verschiedene Schichten und Komponenten, um eine klare Trennung von Funktionalitäten zu gewährleisten. Die unterste Schicht wird von Ressourcen wie Festplattenspeicher, Bandspeicher und Datenbanken gebildet. Speicherressourcenföderationen werden auf dieser Ebene verwaltet, um die Komplexität vor dem Nutzer zu verbergen. Über den Ressourcen liegen die generischen Datendienste, die Speichervirtualisierung und der Metadatendienst, die die existierenden Ressourcen zur Speicherung der Daten und Metadaten

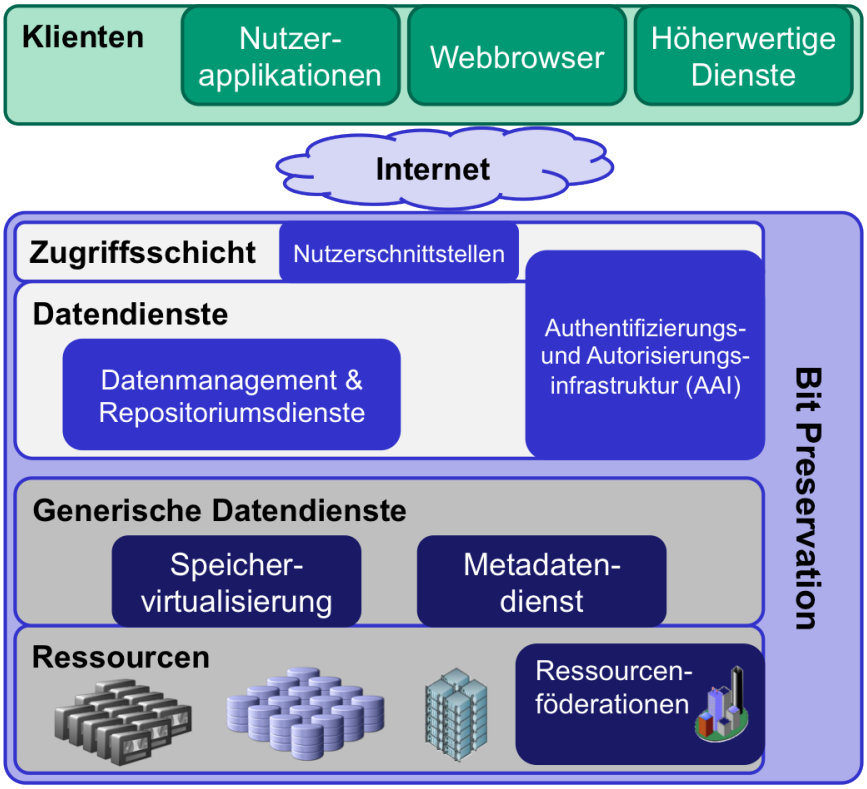

Abbildung 3.2: Architektur zur archivalischen Speicherung heterogener Forschungsdaten.

nutzen. Die Speichervirtualisierung abstrahiert von spezifischen Ressourcen durch einen logischen Namensraum und bietet Mechanismen, die Datenintegrität zu gewährleisten. Der Metadatendienst speichert die administrativen Metadaten für jede Datei unabhängig von der darunter liegenden Datenbank. Die Datenmanagement- und Repositoriumsdienste in der darüber liegenden Schicht kombinieren die generischen Datendienste und beinhalteten Teile der Authentifizierungs- und Autorisierungsinfrastruktur (AAI). Die Zugriffsschicht bietet Schnittstellen für den Zugriff auf den Speicher und ebenfalls Teile der AAI Funktionalitäten. Klienten repräsentieren die oberste Schicht der Architektur. Nutzerapplikationen, ein Webbrowser oder höherwertige Dienste bieten spezialisiertere Funktionalitäten und nutzen die Architektur zur archivalischen Speicherung heterogener Forschungsdaten als zuverlässigen Speicher.

3.2 Standardisierte Schnittstellen

Wichtiger Teil einer nachhaltigen und zuverlässigen Speicherinfrastruktur ist die Bereitstellung einer standardisierten und konsistenten Schnittstelle für den Nutzer. Veränderungen führen zu notwendigen Anpassungen auf der Klientenseite und sollten daher möglichst vermieden werden. In der Nomenklatur der Architektur zur archivalischen Speicherung heterogener Forschungsdaten ist die Schnittstelle Teil der Zugriffsschicht. Durch die Nutzung in den Geistes- und Kulturwissenschaften ergeben sich weitere Voraussetzungen für die Schnittstelle:

- Die Schnittstelle bietet lediglich Basisfunktionalitäten für den Umgang mit Dateien.
- Die Schnittstelle ist leicht zu benutzen.
- Die Schnittstelle agiert als Abstraktionsschicht und verbirgt die Komplexität des darunterliegenden Speichersystems.

Die DARIAH Storage API

Grundsätzlich können existierende, weit verbreitete Schnittstellen für den Datenzugriff verwendet werden. Beispiele sind der Amazon Simple Storage Service (S3) [2], das Cloud Data Management Interface (CDMI) [18], das Merritt Storage Service Interface [93] oder die DataOne API [21]. Alle bieten die benötigen Methoden für Speicher-, Aktualisierungs-, Lese- und Löschoperationen von Dateien. Zusätzlich erlauben sie dem Nutzer, direkt auf die Speicherknoten des unterliegenden Systems zuzugreifen und Funktionalitäten zur Replikation anzusprechen. Ebenfalls werden durch die genannten Schnittstellen Funktionalitäten wie Versionierung, Suchen, Auflisten von Dateien, Rechtemanagement, Verwaltung von persistenten Identifiern (PIDs) und Formatidentifikation bereitgestellt, die nicht Teil der Architektur sind. Wie beschrieben sollen die Komplexität des Speichersystems und die Verantwortlichkeit für die Replikation vor dem Nutzer verborgen werden, um eine möglichst einfach zu nutzende Schnittstelle zu schaffen. Die Zusatzfunktionalitäten widersprechen der Anforderung der Beschränkung auf Basisfunktionen für den Umgang mit Dateien und werden von höherwertigen Diensten abgedeckt. Konzeptionell beschränkt sich die Architektur zur archivalischen Speicherung heterogener Forschungsdaten auf generische Basisfunktionalitäten, auf denen höherwertige Dienste aufbauen können.

Die für die aufgeführten Anforderungen spezifizierte DARIAH Storage API ist eine REST-Schnittstelle basierend auf dem HTTP-Protokoll, die die grundlegenden Methoden zur Datenspeicherung abdeckt. Speziell stimmt sie in Bezug auf Fehlermeldungen und Semantik mit RFC 2616 [33] überein. Eine detaillierte Beschreibung bereitgestellter Funktionalitäten sowie Verwendungsmöglichkeiten befindet sich im Anhang.

Die DARIAH Admin API

Die Bit Preservation Admin API ist eine administrative Schnittstelle zur Konfiguration ausgewählter Funktionalitäten der Bit Preservation und den Abruf bestimmter Informationen über die gespeicherten Daten. Zusätzlich zur Storage API mit ihren nach Spezifikation grundlegenden Speicherfunktionalitäten wird eine weitere Schnittstelle zum Ansprechen der notwendigen Funktionalitäten zur Bit Preservation benötigt. Auch bei dieser Schnittstelle liegt der Fokus auf einer einfachen Benutzbarkeit und auf möglichst generischen Funktionalitäten als Abstraktionsschicht zur Unterstützung einer langfristigen Nutzbarkeit. Eine detailliertere Beschreibung inklusive Verwendungsmöglichkeiten befindet sich im Anhang.

Die Konfiguration von Funktionalitäten zur Bit Preservation bietet drei Möglichkeiten zur Veränderung von Parametern. Ein Nutzer hat die Möglichkeit, sich das Bit Preservation Level der gespeicherten Daten anzeigen zu lassen. Das Bit Preservation Level wird durch die Anzahl und die Lokation der gespeicherten Repliken, unterschiedliche Prüfsummenalgorithmen und das Intervall zwischen zwei Integritätsüberprüfungen bestimmt. Die Level und die zugehörigen Werte der drei Parameter werden von den Institutionen, die eine Instanz der Architektur zur archivalischen Speicherung heterogener Forschungsdaten zur Verfügung stellen, bestimmt. Zusätzlich kann der Nutzer seine Daten als archivierbar markieren und damit längere Zugriffszeiten akzeptieren, da die Daten in diesem Fall auf Nearline- oder Offlinespeicher verschoben werden können. Ein Administrator hat die Möglichkeit, zusätzliche Datenintegritätsüberprüfungen auszulösen.

Ein Nutzer oder ein Administrator kann unterschiedliche Informationen über die Dateien oder die Mechanismen zur Bit Preservation abrufen: das der Datei zugewiesene Bit Preservation Level, die Anzahl und Lokation der Repliken, die Archivierbarkeit der Dateien, die Prüfsumme der Dateien sowie den genutzten Prüfsummenalgorithmus, das Intervall der Datenintegritätsüberprüfungen und den Zeitpunkt der letzten Überprüfung.

3.3 Kernkonzepte zur Langzeitarchivierung

Basierend auf der in Kapitel 2.3 durchgeführten Einteilung der Methodiken und Verfahren der Bit Preservation lassen sich die folgenden typischen Anwendungsfälle identifizieren, die durch generische Konzepte innerhalb der Architektur zur archivalischen Speicherung heterogener Forschungsdaten abgedeckt werden müssen:

- ***Haltbarkeit der Speichermedien:*** Die derzeitig verwendete Speicherkomponente muss auf Grund von Softwareobsoleszenz ausgetauscht werden. Als Konsequenz müssen existierende Dateien in eine alternative Speicherkomponente migriert werden.
- ***Anzahl der Kopien + Verteilung / Unabhängigkeit der Kopien:*** Für eine Datei soll eine zusätzliche Replik in einer alternativen Speicherkomponente erstellt werden. Eine weitere Datei wird inklusive aller Repliken gelöscht.
- ***Regelmäßige Integritätsüberprüfungen*** Vor der Bereitstellung für den Nutzer muss die Integrität einer Datei überprüft werden.

Speicherkonzept

Bei einer Langzeitarchivierung von Daten sollten zum einem möglichst unabhängige Speicherkomponenten verwendet werden und zum anderen eine bei Obsoleszenz notwendige Austauschbarkeit der Speicherkomponenten erleichtert werden. Das in der Architektur zur archivalischen Speicherung heterogener Forschungsdaten verwendete Speicherkonzept ist zu diesem Zweck konzipiert und definiert zentrale Funktionalitäten, die von einer Speicherkomponente erwartet werden. Basierend auf einer Analyse existierender Speichervirtualisierungskomponenten konnten vier Methoden extrahiert und generalisiert werden, die für den Umgang mit heterogenen Forschungsdaten erforderlich sind:

save Die Speicherkomponente muss in der Lage sein, eine Datei zu speichern. Zur möglichen Nutzung bei großskaligen Daten wird die Verwendung von Streams empfohlen.

get Die Speicherkomponente muss in der Lage sein, eine gespeicherte Datei zur Verfügung zu stellen. Zur möglichen Nutzung bei großskaligen Daten wird die Verwendung von Streams empfohlen.

delete Die Speicherkomponente muss in der Lage sein, eine gespeicherte Datei zu löschen.

exists Die Speicherkomponente muss in der Lage sein, die Existenz einer gespeicherten Datei zu verifizieren.

Durch die Beschränkung auf diese vier Basisfunktionalitäten wird eine leichte Austauschbarkeit der Speichervirtualisierungskomponente unterstützt. Auf diese Weise ist es ebenfalls möglich, unterschiedliche Speichervirtualisierungskomponenten parallel anzusprechen. Als weitere administrative Funktionalität ist ein **close** zur ordnungsgemäßen Schließung möglicher Verbindungen zur Speichervirtualisierungskomponente vorgesehen.

Der ISO-Standard 2382:2015 [45] beschreibt Interoperabilität als die Fähigkeit, zwischen verschiedenen funktionalen Einheiten zu kommunizieren, Programme auszuführen oder Daten zu transferieren, so dass der Nutzer kaum oder keine Kenntnisse der einzigartigen Charakteristiken der Einheiten haben muss.[1] Aufbauend auf der abstrakten Definition muss Interoperabilität stets für den speziellen Fall definiert werden. Grundsätzlich ist im Kontext einer Langzeitarchivierung die Definition von Interoperabilität auf verschiedenen Ebenen wie beispielsweise Daten, Metadaten oder Repositorien denkbar. Die vier vorgestellten Grundfunktionalitäten stellen eine Definition von Interoperabilität für Speichervirtualisierungskomponenten dar. Sobald Speicherkomponenten die beschriebenen Funktionalitäten erfüllen, können sie auch interoperabel verwendet werden. Beispiele von aktuellen Systemen, die die Anforderungen erfüllen, finden sich in Tabelle 3.1. Ebenfalls aufgeführt sind die jeweils verwendeten Methoden bzw. Konstrukte, die die Grundfunktionalitäten bereitstellen können. Durch die Beschränkung auf die Grundfunktionalitäten lässt sich ein breites Spektrum von Speicherkomponenten integrieren. Auch an zukünftige Speicherkomponenten werden auf diese Weise Anforderungen gestellt, die eine nahtlose Integration und damit die Zukunftsfähigkeit der Architektur zur archivalischen Speicherung heterogener Forschungsdaten ermöglichen.

[1] „Interoperability: capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units.“ [45]

	save	get	delete	exists
Festplatte	createNewFile() + IOUtils.copy()	InputStream	delete()	exists()
iRODS	createNewFile() + IOUtils.copy()	InputStream	delete()	exists()
dCache	srmcp	srmcp	srmrm	srmls

Tabelle 3.1: Beispiele von Speicherkomponenten, die die vier Grundfunktionalitäten und somit die Anforderungen an Interoperabilität von Speicherkomponenten erfüllen.

Preservationskonzept

Für eine effektive Unterstützung einer Langzeitarchivierung von Daten müssen grundlegende Preservationsfunktionalitäten bereitgestellt werden. Im idealen Fall wird der Themenbereich (Bit) Preservation bereits architekturell berücksichtigt und generisch ausgelegt, so dass flexibel auf Veränderungen und neue Entwicklungen reagiert werden kann. Die Architektur zur archivalischen Speicherung heterogener Forschungsdaten integriert zu diesem Zweck ein Preservationskonzept, das mit Datenintegritätsüberprüfungen und Replikation die beiden wichtigsten Funktionalitäten zur Bit Preservation beinhaltet. Die Berechnung der Prüfsummen wird in die Speicherkomponente ausgelagert und erweitert auf diese Weise die Anforderungen aus Kapitel 3.3.

checksum Die Speicherkomponente muss in der Lage sein, eine Prüfsumme für eine gespeicherte Datei zu bestimmen und zurückzugeben.

Die berechnete Prüfsumme wird zentral gespeichert und kann generisch, automatisch und periodisch überprüft werden. Auf diese Weise werden Datenintegritätsüberprüfungen von der spezifischen Speicherkomponente entkoppelt und können generisch (beispielsweise per Admin API) angesprochen werden. Der verwendete Prüfsummenalgorithmus kann an die jeweils aktuellen Anforderungen der Nutzer und der anbietenden Institution angepasst werden und wird nicht im Vorfeld festgelegt. Auf diese Weise lassen sich neue Entwicklungen flexibel integrieren oder wechselnde Anforderungen an die Datensicherheit abdecken. Die Replikation kann wie in Abbildung 3.3 dargestellt

Abbildung 3.3: Replikation - Aneinanderreihung der Grundfunktionalitäten.

durch Aneinanderreihung der atomaren Grundfunktionalitäten erreicht werden. Soll eine Datei von Speichervirtualisierungskomponente A in Speichervirtualisierungskomponente B repliziert werden, sind die folgenden Schritte durchzuführen:

1. Es wird mittels *exists* festgestellt, ob die zu replizierende Datei in der Speichervirtualisierungskomponente A vorhanden ist.
2. Die zu replizierende Datei wird mittels *get* aus Speichervirtualisierungskomponente A geholt.
3. Die zu replizierende Datei wird mittels *save* in Speichervirtualsierungskomponente B abgelegt.
4. Es wird mittels *checksum* überprüft, ob die Prüfsumme der Datei in Speichervirtualisierungskomponente B mit der gespeicherten Prüfsumme der Datei in Speichervirtualisierungskomponente A übereinstimmt.

Dieses Verfahren kann mehrmals durchgeführt werden, um die Anzahl der Repliken durch die Nutzung verschiedener Speicherkomponenten zu erhöhen. Im Falle eines Fehlers kann der Ablauf ebenfalls erneut angestoßen werden. Innerhalb der Architektur zur archivalischen Speicherung heterogener Forschungsdaten liegt ein konsistenter Zustand vor, wenn alle Repliken einer Datei identisch sind. Es wird die Annahme getroffen, dass die originale Datei fehlerfrei vorliegt und daher durch die vorgesehene Prüfsummenberechnung ein konsistenter Zustand erreicht wird. Ist die Annahme nicht haltbar, können sowohl nach Schritt 1 als auch nach Schritt 2 zusätzliche Prüfsummenberechnungen durchgeführt werden. Soll die Anzahl der Repliken verringert werden, genügt es, die Replik aus der entsprechenden Speicherkomponente zu löschen:

1. Die Datei wird mittels *delete* in der Speichervirtualisierungskomponente gelöscht.

Die bereits in der Speichervirtualisierungskomponente vorgesehenen und integrierten Replikationsfunktionalitäten können zur Erhöhung der Sicherheit zusätzlich ebenfalls genutzt werden, werden allerdings nicht zentral verwaltet und überprüft. Die entsprechende Komponente steht in der Verantwortung, diese zusätzlichen Maßnahmen zu koordinieren, zu verifizieren und die Konsistenz zu gewährleisten.

Sowohl die Funktionalitäten zur Prüfsummenberechnung als auch zur Replikation werden nicht direkt gesteuert, stattdessen wird das Konzept von so genannten Bit Preservation Level verwendet. Ein Bit Preservation Level beschreibt die Intensität von Verfahren zur Sicherung der Datenintegrität. Die unterschiedlichen Level können von der anbietenden Institution definiert werden und bieten Standardwerte für die Parameter „Anzahl der Repliken“, „Lokation der Repliken“, „Intervalle der Prüfsummenberechnungen“ und „Prüfsummenalgorithmen“. Die Admin API stellt eine mögliche Schnittstelle zur Modifikation der Bit Preservation Level dar. Durch dieses Konzept wird von den konkreten Maßnahmen abstrahiert, um eine generische und zukunftsfähige Bit Preservation zu ermöglichen.

Für eine nachhaltige Speicherung müssen Speicherkomponente und Bit Preservation in hohem Maße interagieren. Durch das vorgestellte Preservationskonzept konnte in der Architektur zur archivalischen Speicherung heterogener Forschungsdaten eine generische Verzahnung der beiden Komponenten erreicht werden.

Migrationskonzept

Bei einer Langzeitarchivierung von Daten treten in regelmäßigen Intervallen beispielsweise Softwareobsoleszenzen auf, die wiederholte Datenmigrationen zur Sicherstellung der kontinuierlichen Verfügbarkeit notwendig machen. Insbesondere müssen wie beim entwickelten Speicherkonzept neue Technologien und Entwicklungen berücksichtigt werden. Basierend auf den beschriebenen Funktionalitäten der Speicherkomponente beinhaltet die Architektur zur archivalischen Speicherung heterogener Forschungsdaten zu diesem Zweck ein Datenmigrationskonzept. Die Datenmigration einer Datei aus Speichervirtualisierungskomponente A nach Speichervirtualisierungskomponente B kann wie in Abbildung 3.4 dargestellt durch Aneinanderreihung der atomaren Grundfunktionalitäten realisiert werden:

Abbildung 3.4: Migration - Aneinanderreihung der Grundfunktionalitäten.

1. Es wird mittels *exists* festgestellt, ob die zu migrierende Datei in Speichervirtualisierungskomponente A vorhanden ist.

2. Die zu migrierende Datei wird mittels *get* aus Speichervirtualisierungskomponente A geholt. Optional kann hier mittels *checksum* eine Überprüfung der Datenintegrität veranlasst werden.

3. Die zu migrierende Datei wird mittels *save* in Speichervirtualisierungskomponente B abgelegt.

4. Es wird mittels *checksum* überprüft, ob die Prüfsumme der Datei in Speichervirtualisierungskomponente B mit der gespeicherten Prüfsumme der Datei in Speichervirtualisierungskomponente A übereinstimmt.

5. Die zu migrierende Datei wird mittels *delete* in Speichervirtualisierungskomponente A gelöscht.

Während der Durchführung der Schritte 1 - 4 kann der Nutzer stets auf die Datei in Speichervirtualisierungskomponente A zugreifen. Verläuft der Prozess fehlerfrei, greift der Nutzer ab Schritt 5 automatisch auf die migrierte Version in Speichervirtualisierungskomponente B zu. Für den Nutzer sind keinerlei Anpassungen vorzunehmen. Tritt zwischenzeitlich ein Fehler auf, wird der Prozess abgebrochen und kann zur Sicherstellung eines konsistenten Zustandes ohne Beeinträchtigung neu gestartet werden.

Insgesamt kann auf diese Weise eine Datenmigration ohne zusätzlichen Aufwand für neue Speicherkomponenten realisiert werden. Erfüllt eine Speicherkomponente die beschriebenen Grundfunktionalitäten *save*, *get*, *delete*, *exists* und *checksum*, lässt sie sich als Speicherkomponente integrieren und eine Datenmigration per vorgestelltem Ablauf ist möglich. Einzig die Integration und Bereitstellung der Grundfunktionalitäten ist für jede Speicherkomponente zu leisten. Eine beispielhafte Quantifizierung und Evaluation der Datenmigration findet sich in Kapitel 3.5.

Zusammenfassung

Speicher-, Preservations- und Migrationskonzept lassen sich direkt auf die formulierten Anwendungsfälle übertragen. Auf diese Weise wird auf Grund der Abdeckung der repräsentativen Anwendungsfälle die Vollständigkeit der Grundfunktionalitäten belegt.

Die derzeitig verwendete Speicherkomponente muss auf Grund von Softwareobsoleszenz ausgetauscht werden. Als Konsequenz müssen existierende Dateien in eine alternative Speicherkomponente migriert werden.
Für die Integration einer neuen Speicherkomponente müssen die Grundfunktionalitäten *save*, *get*, *delete*, *exists* und *checksum* implementiert werden. Der beschriebene Ablauf des Migrationsprozesses wird anschließend durchgeführt und die veraltete Speicherkomponente kann entfernt werden.

Für eine Datei soll eine zusätzliche Replik in einer alternativen Speicherkomponente erstellt werden. Eine weitere Datei wird inklusive aller Repliken gelöscht.
Zur Erstellung einer zusätzlichen Replik wird der beschriebene Ablauf des Replikationsprozesses mit einer vorhandenen oder neuen Speichervirtualisierungskomponente durchgeführt. Soll eine Datei gelöscht werden, wird *delete* sowohl auf die Repliken als auch auf das Original angewendet.

Vor der Bereitstellung für den Nutzer muss die Integrität einer Datei überprüft werden.
Mit Hilfe von *checksum* kann die Integrität beliebiger gespeicherten Dateien inklusive der Repliken überprüft werden.

3.4 Anwendungsfall Virtuelles Skriptorium

Zur Quantifizierung und Validierung der konzipierten Architektur wird ein typischer Anwendungsfall der Geistes- und Kulturwissenschaften herangezogen. Das „Virtuelle Skriptorium St. Matthias“ ist eine Onlineedition mit Bildern von mehr als 440 mittelalterlichen Handschriften, beispielhaft in Abbildung 3.5 dargestellt und im Folgenden Kodizes genannt, die zwischen dem achten und sechszehnten Jahrhundert entstanden sind, und einer Datenbank, gefüllt mit Informationen mehrerer Manuskriptkataloge. Obwohl der Bestand während der Säkularisierung verstreut wurde, sind heutzuta-

Abbildung 3.5: Kodex des St. Mattheiser Bestandes (Stadbibliothek und Stadtarchiv Trier, Hs 1108/55 4° 7v und 8r, segmentiert).

ge die meisten Kodizes in der Stadtbibliothek Trier und der Bibliothek des Bischöflichen Priesterseminars zu finden. Zusätzlich befinden sich mehrere Kodizes im Bistumsarchiv Trier und in der Bibliothek der Abtei St. Matthias. Ziel des Projektes ist die digitale Rekonstruktion der Bibliothek mit ihrem breiten Spektrum von Traditionen und Themen. Die aktuelle und der Digitalisierung zugrunde liegende Übersicht der einst vorhandenen Kodizes kann in [12] eingesehen werden. Die verfügbaren Manuskriptbeschreibungen werden in einer SQL-Datenbank zur Verfügung gestellt. Die Informationen werden zum einen mit den digitalisierten Bildern, zum anderen mit den Daten aus den Personennormdaten (PND) der Deutschen Nationalbibliothek verglichen.

Die Digitalisierung des Bestandes ist ein aufwändiger Prozess, der mehr als vier Jahre in Anspruch genommen hat. Jede Seite wurde einzeln aufgenommen und gespeichert, die entstandenen TIFF-Bilddateien variieren in ihrer Größe im Bereich 1-100 MB und besitzen eine Auflösung von 300 dpi. Im

Abbildung 3.6: Stadbibliothek und Stadtarchiv Trier, Hs 31 8° 2v und 3r, dargestellt im DFG-Viewer.

Laufe der Bearbeitungszeit wurden zu diesem Zweck unterschiedliche Scanner mit verschiedenen Eigenschaften verwendet, die eine Kalibrierung und Vorprozessierung der Bilddaten für eine wissenschaftliche Auswertung zwingend erforderlich machen. In einem weiteren Verarbeitungsschritt werden für jedes hochauflösende Masterdigitalisat fünf zusätzliche Derivate im JPEG- sowie PDF-Format mit geringerer Auflösung für eine komfortable Webpräsentation erstellt.

Die tägliche Arbeit mit dem Virtuellen Skriptorium folgt meist einem standardisierten Ablauf. Die Datenbank wird mit speziellen PHP-Skripten von der Projekthomepage abgefragt. Der Nutzer kann dabei nach bestimmten Texten, Autoren, Daten, Schreibern, Neumen oder Glossen suchen. Das Ergebnis beinhaltet neben zusätzlichen Informationen zur Geschichte des Kodex, Editionen und Sekundärliteratur auch eine URL zu einer XML-Datei mit beschreibenden Metadaten des Kodex und spezifischen Metadaten zur Anzeige des Kodex mit dem DFG-Viewer [24]. Der DFG-Viewer, in Abbildung 3.6

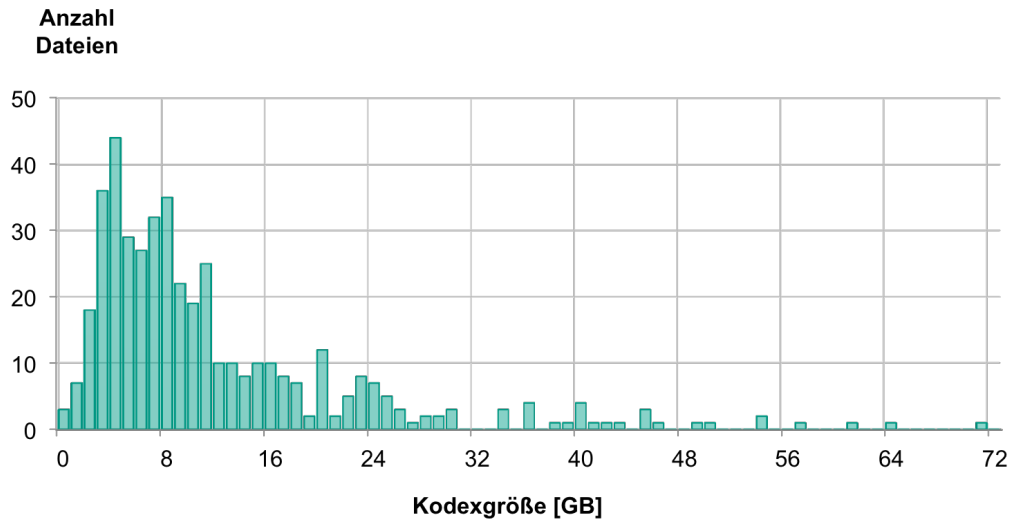

Abbildung 3.7: Histogramm des Virtuellen Skriptoriums St. Matthias (x-Achse: Größe der Kodizes, y-Achse: Häufigkeit der Kodexgrößen).

dargestellt, ist eine Applikation zur Darstellung von Kodizes, die bei Digitalisierungsprojekten in Deutschland weit verbreitet ist. Um diese Applikation nutzen zu können, müssen die digitalisierten Bilder sowie ihre Metadaten öffentlich auf einem Webserver zur Verfügung stehen. Die Applikation arrangiert die Bilder und die Metadaten zu einem virtuellen Kodex und erlaubt dem Nutzer, zu scrollen, zu zoomen oder Teile des Kodex herunterzuladen. Um eine nahtlose Integration mit dem DFG-Viewer zu ermöglichen, wurde die unterliegende Struktur der Daten angepasst. Die digitalisierten Bilder werden in verschiedenen Auflösungen für die Zoomfunktion und als PDF für den Download zur Verfügung gestellt. Die Metadaten der XML-Datei sind im METS-Format [60] strukturiert und beinhalten ebenfalls einen TEI-Header [89], da dieser Standard umfangreichere Manuskriptbeschreibungen erlaubt und von der TEI-Version des DFG-Viewers unterstützt wird.

Zur quantitativen Betrachtung der Migrationsfunktionalitäten der Architektur zur archivalischen Speicherung heterogener Forschungsdaten wurden die TIFF-Masterdateien des Virtuellen Skriptoriums verwendet. Insgesamt handelt es sich um 170.000 Bilddateien in 440 Verzeichnissen (Stand Au-

gust 2013). Diese Datensammlung beinhaltet fast den gesamten Bestand und weist eine Größe von 4,7 TB auf. Es wurde jeweils ein Kodex, der einem der Verzeichnisse entspricht, zu einem Tarball zusammengefasst. Die Verteilung der entstandenen Dateigrößen ist in Abbildung 3.7 dargestellt. Durch die große Varianz der möglichen Dateigrößen stellt das Virtuelle Skriptorium St. Matthias eine realistische Sammlung dar, die in geisteswissenschaftlichen Forschungsprojekten entstehen kann.

3.5 Möglichkeiten der Datenmigration

Bei der Untersuchung der Datenmigration in der Architektur zur archivalischen Speicherung heterogener Forschungsdaten werden zwei Ansätze betrachtet und zu diesem Zweck Instanzen mit verschiedenen Speicheradaptern - Festplatte, iRODS oder dCache - genutzt. Beim *Ingest* werden die Daten mit einer POST-Operation in einer Instanz gespeichert, die eine der möglichen Speicherkomponenten verwendet. Im Falle der *Migration* werden die PUT-Operation sowie eine Instanz, die alle drei Speicherkomponenten ansprechen kann, genutzt. Die Festplatte und die iRODS-Instanz sind in der verwendeten Testmaschine lokalisiert und wurden dezidiert für die Migrationsuntersuchungen verwendet. Die dCache-Instanz ist Teil einer Testumgebung der GridKa-Infrastruktur und kann von mehreren Nutzern angesprochen werden. Zur Bestimmung des Datendurchsatzes wurde die gesamte Zeit der jeweiligen Operation inklusive Transfer, Metadatenverwaltung und Prüfsummenberechnung gemessen und mit der Datengröße in Beziehung gesetzt.

Ansatz 1 - Ingest

Die Abbildungen 3.8 und 3.9 zeigen die Ergebnisse von sechs Ingestoperationen des gesamten Bestandes. Für jede der drei Speicherkomponenten wurden zwei Durchläufe durchgeführt, so dass 880 Dateien von jeder Speicherkomponente verarbeitet werden mussten.

Abbildung 3.8 zeigt ein Histogramm des erreichten Datendurchsatzes in Abhängigkeit der verwendeten Speicherkomponente. Der höchste Durchsatz wird bei Nutzung einer Festplatte erreicht, hier ist ebenfalls die geringste Varianz zu beobachten. Bei Nutzung von iRODS oder dCache als Speicherkomponente ist der Durchsatz durch die von der Software bedingten zusätzlichen Operationen und nötiger Netzwerkübertragungen geringer. Auffällig ist im Vergleich der Komponenten eine sehr hohe Varianz bei Speicherung der Da-

ten in dCache, einzelne Dateien wurden mit weniger als 6 MB/s übertragen, sowie eine leichte Schieflage der Verteilung. In diesem Fall ist von wechselnder Auslastung der verwendeten Infrastruktur auszugehen.

Abbildung 3.9 zeigt den Datendurchsatz in chronologischer Abfolge in Abhängigkeit der verwendeten Speicherkomponente. Auch in dieser Darstellung zeigt sich die beschriebene Einordnung des Datendurchsatzes. Zusätzlich lässt sich insgesamt eine hohe Stabilität der Speicherkomponenten über die gesamte Zeitspanne des Ingests beobachten, lediglich dCache zeigt einige Ausreißer in Bereiche geringeren Datendurchsatzes, die durch konkurrierende Zugriffe auf das System begründet werden. Bei Verwendung von iRODS als Speicherkomponente brach der Ingestvorgang einige Male ab und erforderte auf Grund der systemspezifischen Ablehnung weiterer Anfragen einen Neustart des Systems, erkennbar an Sprüngen der Datendurchsatzkurve (beispielhaft nach Datei 25 und nach Datei 81 des zweiten iRODS-Durchlaufes). Für einen stabilen Betrieb ist daher die Stabilität und Skalierbarkeit der gewählten Speicherkomponente sicher zu stellen.

Insgesamt ist der Ingestansatz praktikabel und erlaubt eine stabile Migration der Daten des gesamten Bestandes unabhängig von der Dateigröße. Die verwendete Methode ist bereits in der Referenzimplementierung realisiert, es ist lediglich die Bereitstellung einer zusätzlichen Instanz mit der gewünschten Speicherkomponente nötig. Während des Migrationsprozesses werden allerdings alternative Identifier von der Instanz vergeben, die Änderungen in höherwertigen Diensten oder die zusätzliche Verwendung von persistenten Identifiern notwendig machen. Falls kein direkter Zugriff auf die Daten möglich ist, ist zusätzlich jeweils der Aufwand einer GET-Operation für eine Migration miteinzubeziehen.

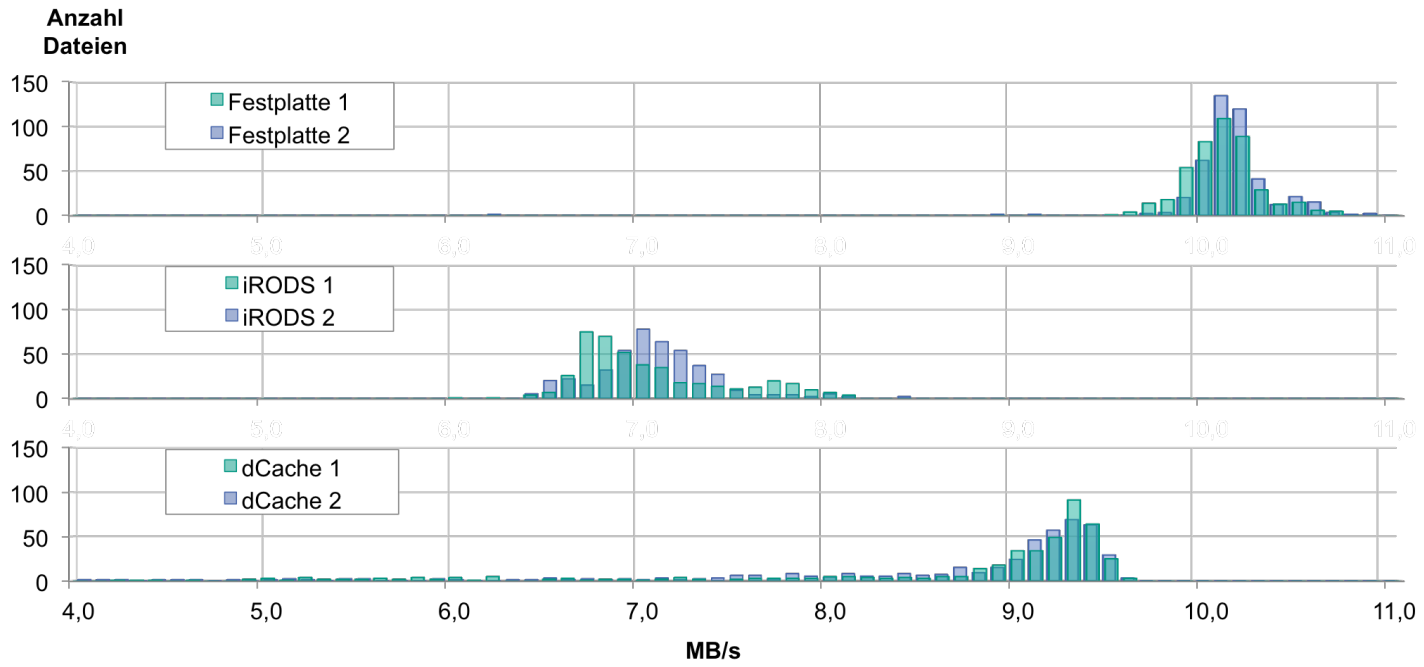

Abbildung 3.8: Vergleich des Datendurchsatzes des Ingestansatzes in Abhängigkeit der verwendeten Speicherkomponente (x-Achse: Geschwindigkeit des Datendurchsatzes, y-Achse: Häufigkeit der Geschwindigkeit).

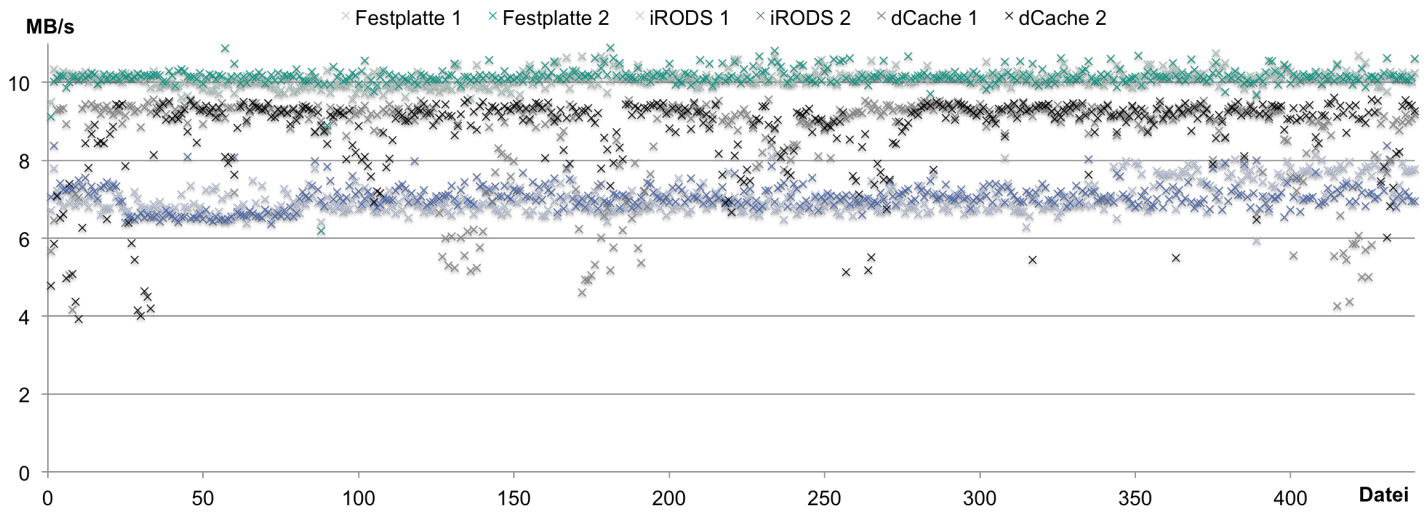

Abbildung 3.9: Datendurchsatz des Ingestansatzes in zeitlicher Abfolge in Abhängigkeit der Speicherkomponente (x-Achse: chronologischer Dateiname, y-Achse: Geschwindigkeit des Datendurchsatzes).

Ansatz 2 - Migration

Die Abbildungen 3.10, 3.11 und 3.12 zeigen die Ergebnisse von 12 Migrationsoperationen des gesamten Bestandes, es wurden jeweils zwei Durchläufe pro Speicherkomponentenkombination durchgeführt. Jede Speicherkomponente musste daher 1760 Dateien sowohl speichern als auch bereitstellen.

In Abbildung 3.10 ist die Kombination der Speicherkomponenten Festplatte – iRODS dargestellt. Die Migration von der Festplatte ins iRODS-System zeigt eine höhere Varianz als von iRODS zur Festplatte, die durch systemspezifische Zusatzoperationen bei der Speicherung erklärbar ist. Die weite Streuung im ersten Durchlauf ist auf eine hohe Systemauslastung zurückzuführen. Bei der Rückrichtung ist eine geringe Varianz zu beobachten, beim Lesen einer Datei aus iRODS sind weniger zusätzliche Operationen, die die gesamte Migration verzögern können, als beim Speichern durchzuführen.

Die Kombination Festplatte – dCache in Abbildung 3.11 zeigt einen höheren Datendurchsatz, jedoch auch eine deutlich höhere Varianz. Aus diesem Grund sind auch geringere maximale Anzahlen von Dateien mit gleichem Durchsatz zu erkennen. Bei der Migration von der Festplatte ins dCache-System kann auf Grund der Möglichkeit eines parallelen Datentransfers bei der Speicherung ein höherer Durchsatz als bei der Migration von dCache zur Festplatte erreicht werden. Die Ausreißer in Bereiche eines niedrigen Datendurchsatzes können durch erhöhte Netzwerk- und Systemauslastung der dCache-Infrastruktur begründet werden.

Abbildung 3.12 stellt die Kombination der Speicherkomponenten iRODS – dCache dar. Auch in diesem Fall ist bei der Migration iRODS – dCache auf Grund der Möglichkeit des parallelen Datentransfers ein höherer Durchsatz als bei der Migration vom dCache- ins iRODS-System zu beobachten. Geringe Datendurchsätze können durch verzögerte Operationen sowohl im iRODS- als auch im dCache-System begründet sein.

Abbildung 3.13 zeigt einen Vergleich des Datendurchsatzes der verschiedenen Kombinationsmöglichkeiten beim Migrationsansatz. Die beiden Durchläufe wurden jeweils zusammengefasst. Auch in dieser Darstellung lässt sich die deutlich erhöhte Varianz der Kombinationen Festplatte – dCache und dCache – Festplatte erkennen, die jedoch auch den höchsten Durchsatz aufweist. Auf Grund der Möglichkeit eines parallelen Datentransfers bei der Speicherung im dCache-System werden bei den Kombinationen Festplatte – dCache und iRODS – dCache höhere Datendurchsätze als bei den Migrationen dCache – Festplatte und dCache – iRODS erreicht.

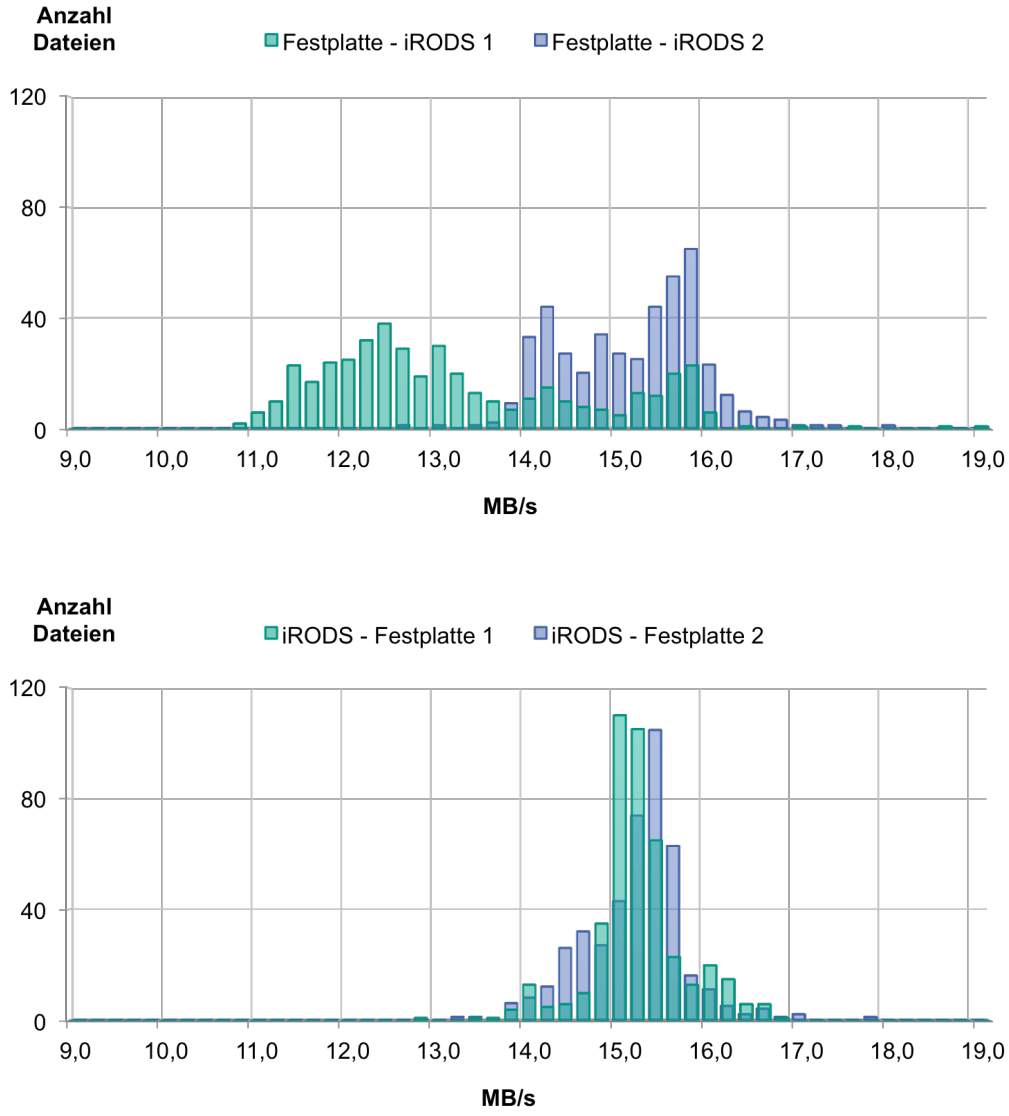

Abbildung 3.10: Datendurchsatz des Migrationsansatzes bei der Kombination Festplatte – iRODS (x-Achse: Geschwindigkeit des Datendurchsatzes, y-Achse: Häufigkeit der Geschwindigkeit).

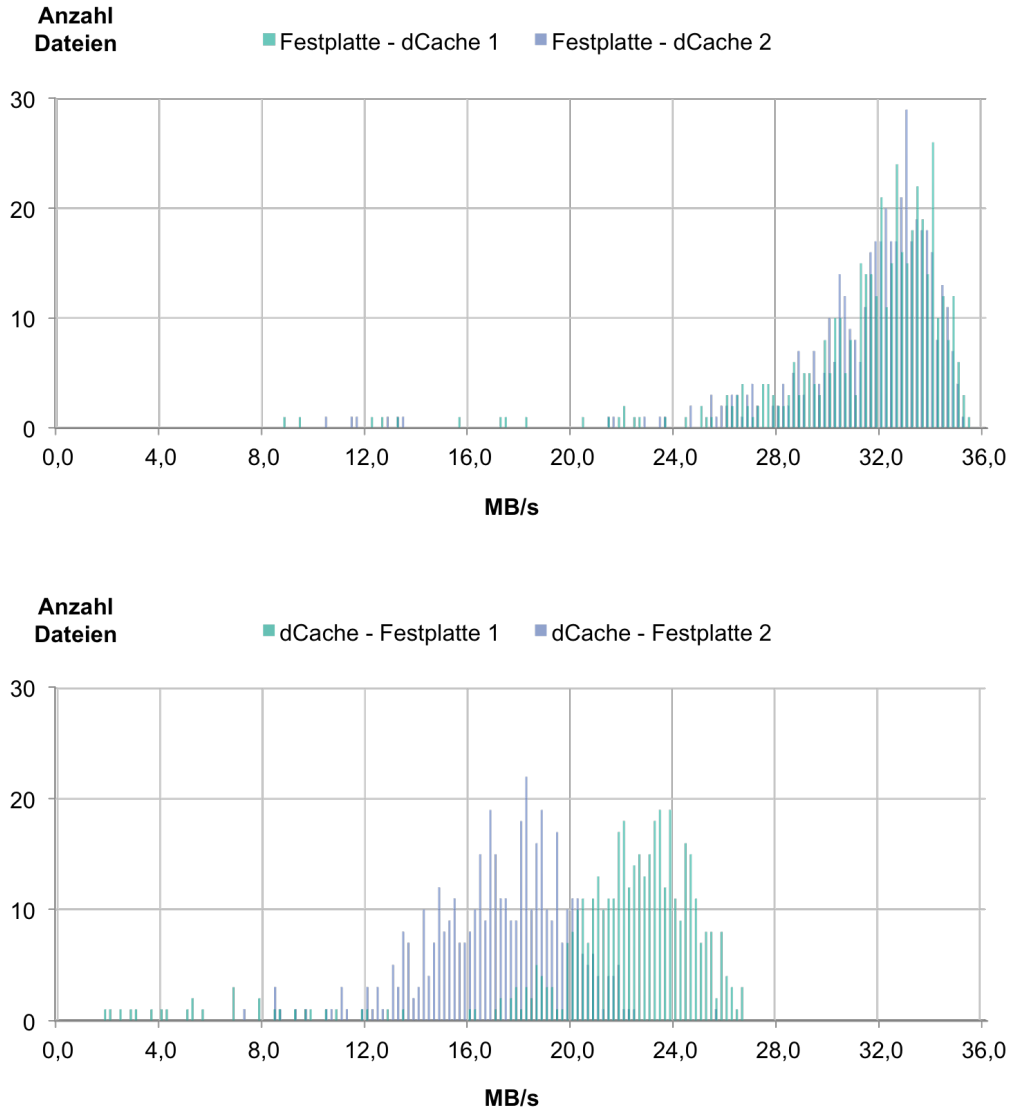

Abbildung 3.11: Datendurchsatz des Migrationsansatzes bei der Kombination Festplatte – dCache (x-Achse: Geschwindigkeit des Datendurchsatzes, y-Achse: Häufigkeit der Geschwindigkeit).

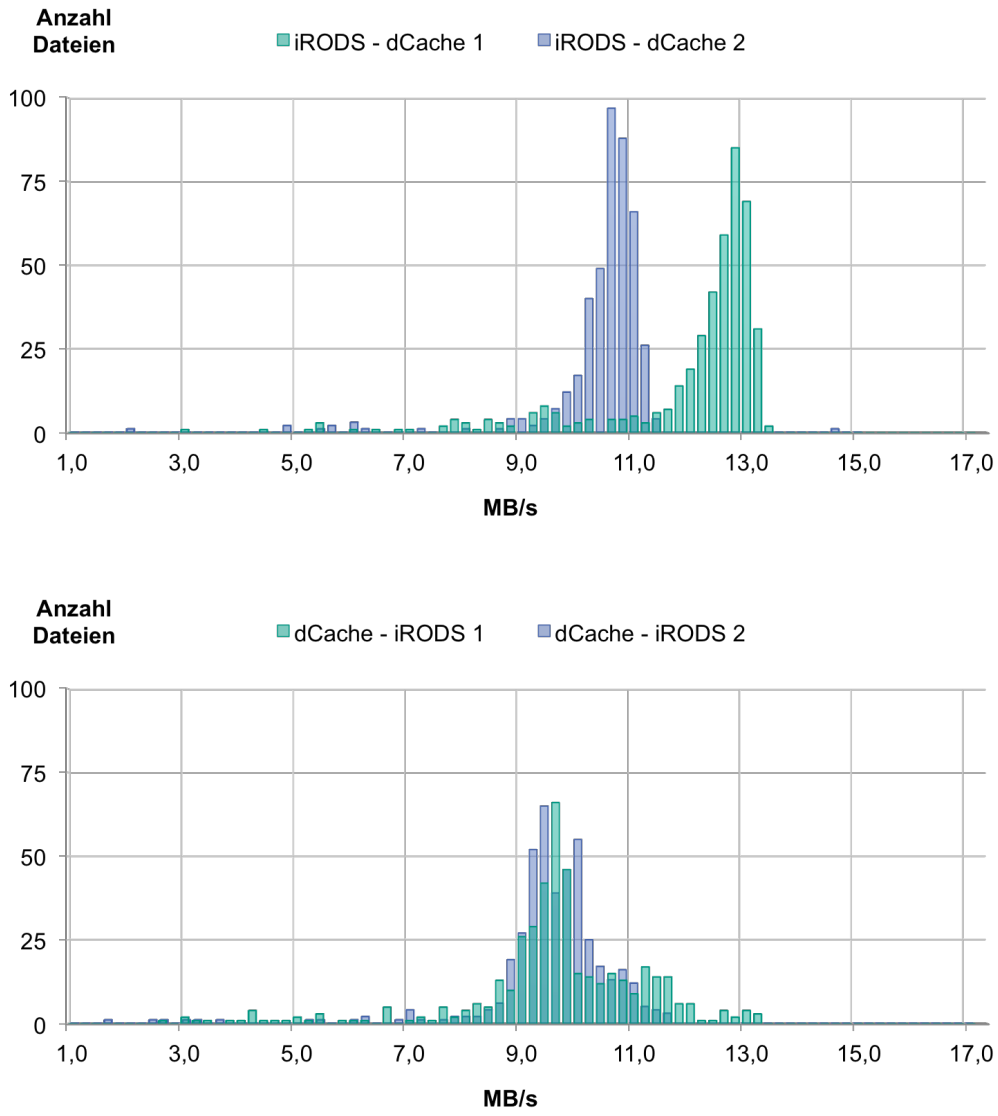

Abbildung 3.12: Datendurchsatz des Migrationsansatzes bei der Kombination iRODS – dCache (x-Achse: Geschwindigkeit des Datendurchsatzes, y-Achse: Häufigkeit der Geschwindigkeit).

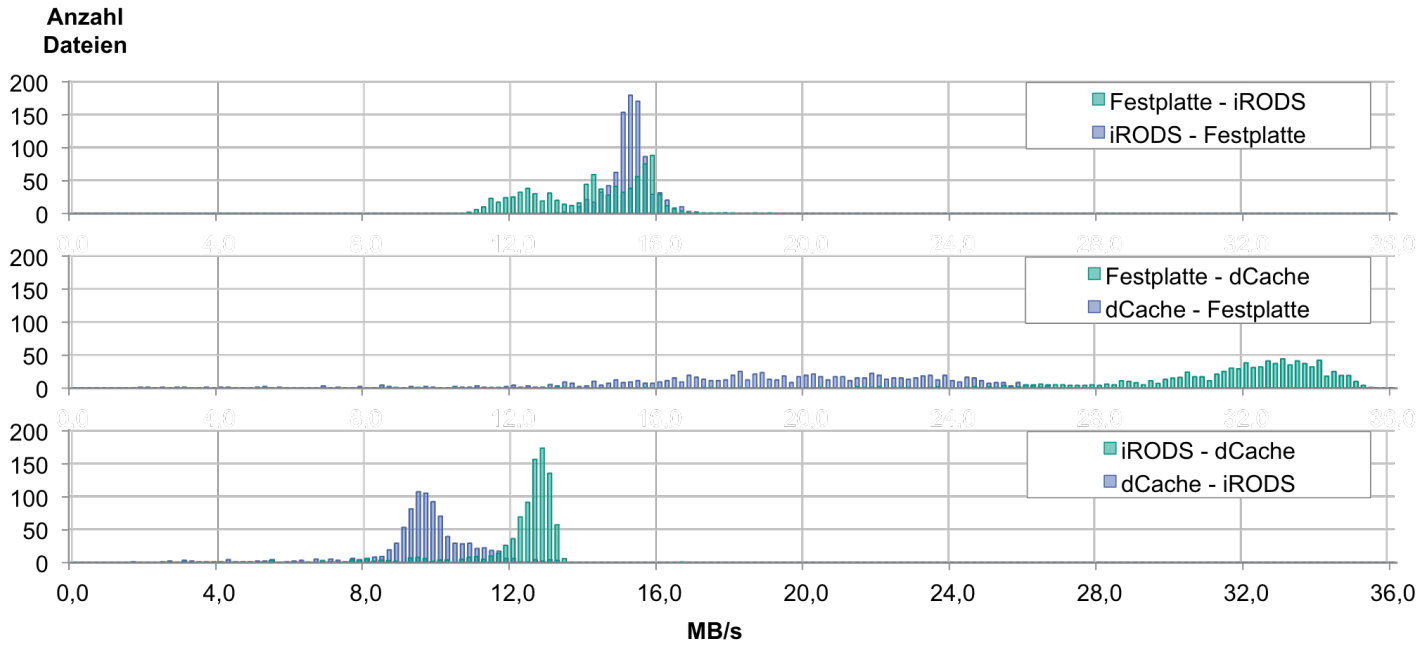

Abbildung 3.13: Vergleich des Datendurchsatzes des Migrationsansatzes in Abhängigkeit der Speicherkomponente (x-Achse: Geschwindigkeit des Datendurchsatzes, y-Achse: Häufigkeit der Geschwindigkeit).

Insgesamt ist der Migrationsansatz praktikabel und erlaubt eine stabile Migration des gesamten Bestandes unabhängig von der Dateigröße. Bei diesem Ansatz musste eine zusätzliche PUT-Methode implementiert werden, die in der Referenzimplementierung nicht vorgesehen war. Die Methode erlaubt eine generische Migration, indem zwei beliebige der bereitgestellten Speicherkomponenten angesprochen werden können. Der Konzeptions- und Implementierungsaufwand ist gering, für zusätzliche Speicherkomponenten müssen lediglich die entsprechenden Grundfunktionalitäten neu implementiert werden. Während des Migrationsprozesses bleibt der Identifier erhalten, so dass auf Nutzerseite keine Anpassungen vorgenommen werden müssen. Im Vergleich zum Ingestansatz kann beim Migrationsansatz ein höherer Datendurchsatz erreicht werden.

3.6 Erkenntnisse

Die vorgestellte Architektur zur archivalischen Speicherung heterogener Forschungsdaten ermöglicht eine klare Trennung von Funktionalitäten und Verantwortlichkeiten durch die Nutzung von Schichten und dedizierter Komponenten. Sowohl für die Zugriffsschnittstellen als auch die Authentifizierungs- und Autorisierungsinfrastruktur werden zur Sicherung der notwendigen Kompatibilität mit anderen Komponenten Standards (HTTP, REST, SAML) genutzt. Durch den gezielten Einsatz von Abstraktionsschichten wird die Interoperabilität und zukünftige Austauschbarkeit der Speicherkomponente

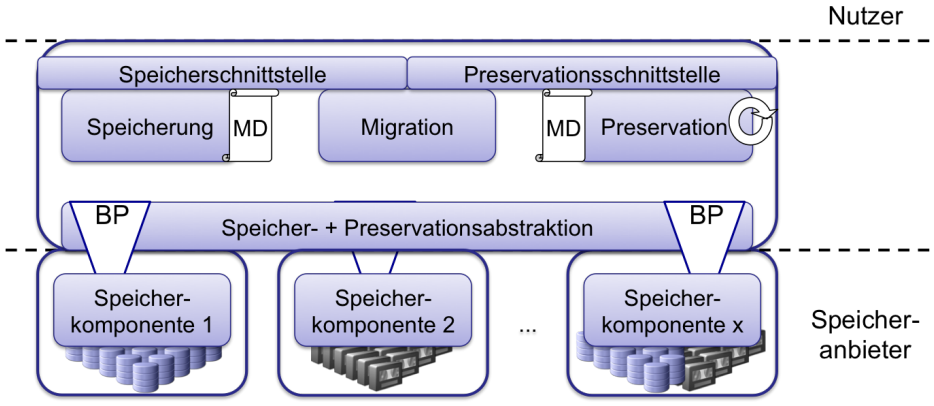

Abbildung 3.14: Zusammenfassende Darstellung der Kernkonzepte in der Architektur zur archivalischen Speicherung heterogener Forschungsdaten.

gewährleistet. Die Architektur ermöglicht auf dieser Basis eine generische Datenmigration ohne zusätzlichen Aufwand und einen generischen Aufruf von Funktionalitäten zur Bit Preservation. Insgesamt werden auf diese Weise, wie in Abbildung 3.14 dargestellt, Speicher-, Preservations- und Migrationskonzepte innerhalb der Architektur vereint.

Der Anwendungsfall „Virtuelles Skriptorium St. Matthias“ bietet eine realistische geisteswissenschaftliche Sammlung zur quantitativen Auswertung der entwickelten Konzepte. Anhand einer Referenzimplementierung konnte die praktische Umsetzbarkeit der Architektur zur archivalischen Speicherung heterogener Forschungsdaten nachgewiesen werden. Die betrachteten Ansätze der Datenmigration sind beide praktikabel, Ansatz 2 ist trotz einmaligem Implementierungsaufwandes auf Grund des höheren Durchsatzes und des gleichbleibenden Identifiers vorzuziehen. Insgesamt konnten die Stabilität des Systems nachgewiesen und eine qualitative Einschätzung der verwendeten Speicherkomponenten vorgenommen werden.

4 Modellierung der Zuverlässigkeit einer Architektur zur Bit Preservation

Grundlage einer Langzeitarchivierung ist eine Einschätzung der Zuverlässigkeit der verwendeten Komponenten. Auf diese Weise können Gefahrenquellen bewertet, eine Risikoabschätzung durchgeführt und passende Strategien für die Datenspeicherung ausgewählt werden. In diesem Kapitel werden basierend auf der Architektur zur archivalischen Speicherung heterogener Forschungsdaten ein Modell zur Quantifizierung der Zuverlässigkeit sowie geschlossene Ausdrücke der Wahrscheinlichkeit unentdeckter Fehler für verschiedene Szenarien entwickelt.

Die Zuverlässigkeit eines System als Funktion der Zeit, $R(t)$, ist die bedingte Wahrscheinlichkeit, dass das System das Intervall $[0, t]$ unbeschadet übersteht. Vorausgesetzt wird, dass das System zum Zeitpunkt $t = 0$ funktionsfähig war [81]. Greenan et al. [41] ergänzen, dass die Formulierung „unbeschadet überstehen“ implizit beinhaltet, dass das System seine beabsichtigte Funktion in einem gewissen Einsatzbereich ausführt. Formal lässt sich daher für den Kontext der Bit Preservation definieren:

Zuverlässigkeit der Bit Preservation Wahrscheinlichkeit, dass ein abgespeicherter Bitstream korrekt zurückgeliefert wird.

In der Literatur werden verschiedene Metriken zur Bestimmung der Zuverlässigkeit einzelner Komponenten und ganzer Systeme verwendet:

Mean Time To Failure (MTTF) bezeichnet die durchschnittliche Zeit, bis ein Fehler in einem Speichersystem auftritt.

Mean Time To Data Loss (MTTDL) bezeichnet die durchschnittliche Zeit, bis ein Fehler in einem Speichersystem auftritt, der zu Datenverlust führt.

Unrecoverable Bit Error Rate (UBER) beschreibt die Wahrscheinlichkeit, dass ein Bit unabhängig von der Verweildauer auf der Festplatte inkorrekt gelesen wird [77].

Bit Half Life Die Bit-Halbwertszeit gibt die Zeit an, nach der ein Bit mit 50%-iger Wahrscheinlichkeit umgekippt ist [76]. Allgemein wird hier und im Folgenden angenommen, dass die Übergänge $0 \to 1$ und $1 \to 0$ mit gleicher Wahrscheinlichkeit auftreten.

Mean Latent Error Time (MLET) bezeichnet den prozentualen Zeitanteil, in der eine Festplatte auf Grund von verborgenen Sektorenfehler anfällig für Datenverlust ist [65].

Normalized Magnitude of Data Loss (NOMDL) kennzeichnet die erwartete Menge verlorener Daten pro nutzbarem Terabyte während der Einsatzzeit t [41].

4.1 Annahmen und Notationen

Zur Bestimmung der Zuverlässigkeit einer Bit Preservation Architektur wird die Wahrscheinlichkeit $P_{uFehler}$, dass ein Bitstream trotz Maßnahmen zur Sicherstellung der Datenintegrität unerkannt fehlerhaft ist, als Metrik verwendet. Für die Gegenwahrscheinlichkeit bzw. die Wahrscheinlichkeit keines unerkannten Fehlers $\overline{P}_{uFehler}$ gilt

$$\overline{P}_{uFehler} = 1 - P_{uFehler}.$$

Generell wird für die folgenden Betrachtungen die (stochastische) Unabhängigkeit der Ereignisse vorausgesetzt. Die Wahrscheinlichkeit des gleichzeitigen Eintretens unabhängiger Ereignisse $A_1, ..., A_n$ lässt sich als Multiplikation der Einzelwahrscheinlichkeiten bestimmen.

$$P(A_1 \cap ... \cap A_n) = \prod_{i=1}^{n} P(A_i)$$

Allgemein bezeichnet $\overline{P}_{uFehler}(x)$ die Wahrscheinlichkeit, dass in einer Komponente x keine unentdeckten Fehler auftreten. Wird keine Komponente angegeben, bezieht sich die Wahrscheinlichkeit auf das gesamte System. Für die einzelnen Bestandteile einer Bit Preservation Architektur lassen sich die folgenden Variablen definieren:

$BP := \overline{P}_{uFehler}(BP)$	Wahrscheinlichkeit, dass in der Bit Preservation Software kein unentdeckter Fehler auftritt
$I := \overline{P}_{uFehler}(I)$	Wahrscheinlichkeit, dass bei der Übertragung per Netzwerk/Internet kein unentdeckter Fehler auftritt
$P := \overline{P}_{uFehler}(P)$	Wahrscheinlichkeit, dass bei der Berechnung der Prüfsumme kein unentdeckter Fehler auftritt
$S_i := \overline{P}_{uFehler}(S_i)$	Wahrscheinlichkeit, dass in Speicherkomponente S_i kein unentdeckter Fehler auftritt

Durch Verwendung der Gegenwahrscheinlichkeiten und Ausnutzung der stochastischen Unabhängigkeit der Ereignisse können die Einzelwahrscheinlichkeiten der Komponenten zur Bestimmung der Gesamtwahrscheinlichkeit je nach Kombination multipliziert werden.

Zusätzlich wird davon ausgegangen, dass die Wahrscheinlichkeit eines unentdeckten Fehlers bei Schreib- und Leseoperationen annähernd gleich,

$$S_i|\text{Schreiben} \approx S_i|\text{Lesen}, \quad BP|\text{Schreiben} \approx BP|\text{Lesen},$$

und in den Komponenten zeitlich konstant ist

$$S_i = const, \quad BP = const, \quad I = const, \quad P = const.$$

4.2 Szenario 0: Ohne Replikation

Grundlage der Modellierung ist die entwickelte Architektur zur archivalischen Speicherung heterogener Forschungsdaten in Abbildung 3.14. Zu diesem Zweck wird die Architektur zweckdienlich abstrahiert, die maßgeblichen Komponenten Netzwerk, Bit Preservation Software und verschiedene Speicherkomponenten identifiziert und als unabhängig voneinander betrachtet. Insbesondere sind auf Grund der modellierten möglichen Distribution von Bit Preservation Software und Speicherkomponenten zwei Netzwerkkomponenten mit gleicher Fehlerwahrscheinlichkeit vorgesehen. Die Zuverlässigkeit alternativer Architekturen lässt sich nach der Identifikation der maßgeblichen Komponenten mit Hilfe der angewendeten Methodik dieses Kapitels ebenfalls quantifizieren. Bei der Speicherung wird eine Datei vom Nutzer per

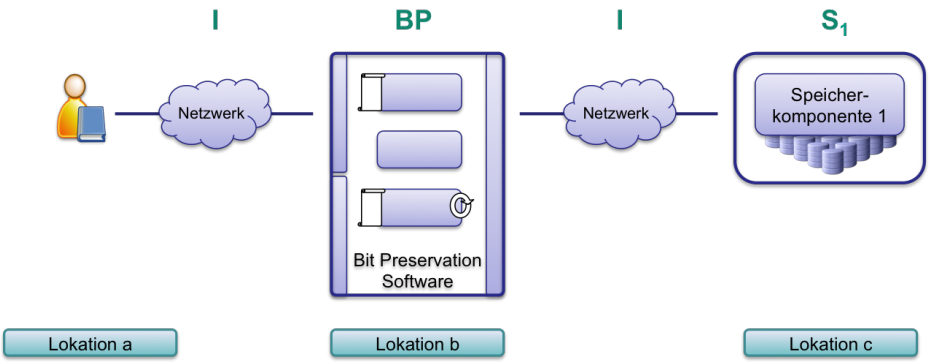

Abbildung 4.1: Szenario 0 - Ohne Replikation (schematische Darstellung).

Netzwerk an die Bit Preservation Software übertragen, die die beschriebenen Funktionalitäten wie beispielsweise Datenverwaltung oder Metadatenverwaltung übernimmt. Anschließend wird die Datei je nach Replikationsszenario in einer oder mehreren Speicherkomponenten abgelegt. Bei einer Leseoperation wird die Datei aus der Speicherkomponente per Netzwerk an die Bit Preservation Software übertragen, die die Datei per Netzwerk dem Nutzer zur Verfügung stellt.

Der Basisfall der Betrachtungen ist das Speichern einer Datei ohne Repliken, dargestellt in Abbildung 4.1. Die Datei wird zweimal durch das Netzwerk (I) und Bit Preservation Software (BP) übertragen und in einer Speicherkomponente (S_i) abgelegt. Für die Wahrscheinlichkeit unentdeckter Fehler bei der Schreiboperation gilt

$$\overline{P}_{uFehler}\{\text{Schreiben}\} = S_1 \times BP \times I^2.$$

Das Szenario ist abgeschlossen, wenn auch die Leseoperation durchgeführt ist. Es gilt $\overline{P}_{uFehler}\{\text{Schreiben}\} = \overline{P}_{uFehler}\{\text{Lesen}\}$ und damit für die Gesamtwahrscheinlichkeit unentdeckter Fehler

$$\overline{P}_{uFehler} = \overline{P}_{uFehler}\{\text{Schreiben}\} \times \overline{P}_{uFehler}\{\text{Lesen}\} = S_1^2 \times BP^2 \times I^4.$$

4.3 Verwendung von Prüfsummen

Eine der bekanntesten Methoden zur Erhöhung der Zuverlässigkeit einer Architektur zur Bit Preservation ist die Berechnung einer Prüfsumme. Auf diese Weise sollen Veränderungen der zu Grunde liegenden Datei festgestellt werden, um auf eine Alternative ausweichen zu können, oder eine Auswahl der

korrekten Datei aus mehreren Repliken ermöglicht werden. Sollen die Fehler nicht nur mit Hilfe der Prüfsumme erkannt werden, ist der Einsatz von fehlerkorrigierenden Codes möglich. Problematisch ist ein Szenario, wenn eine Prüfsumme den korrekten Wert angibt, die entsprechende Datei jedoch fehlerhaft ist, der Fehler folglich nicht erkannt wurde. In diesem Kapitel werden drei in Hard- und Software weit verbreitete Verfahren zur Berechnung von Prüfsummen mit ihren jeweiligen Fehlerwahrscheinlichkeiten vorgestellt.

In einer Bit Preservation Architektur wie beispielsweise in Abbildung 4.1 sind drei Lokationen der Verwendung vorstellbar, wobei die Speicherung der Prüfsumme vernachlässigt wird:

Lokation a: Der Nutzer berechnet lokal eine Prüfsumme nach Erhalt der Datei aus dem Bit Preservation System.

Lokation b: Die Prüfsumme wird von der Bit Preservation Software verwendet, um im Fehlerfall auf eine andere Speicherkomponente ausweichen zu können.

Lokation c: Innerhalb der einzelnen Speicherkomponenten wird eine Prüfsumme berechnet, die Ergebnisse jedoch nicht zwischen den Speicherkomponenten ausgetauscht.

Bei einer Prüfsummenberechnung in Lokation a ist für den Nutzer kein Ausweichen auf eine alternative Speicherkomponente möglich, da diese für ihn nicht zugreifbar sind. Im Fehlerfall kann die entsprechende Datei lediglich neu angefordert werden. Für die Fehlerwahrscheinlichkeit inklusive Prüfsummenberechnung gilt

$$\overline{P}^{*}_{uFehler} = \overline{P}_{uFehler} + P_{uFehler} \times P,$$

wobei $\overline{P}_{uFehler}$ abhängig vom den innerhalb des Systems verwendeten Replikations- und Prüfsummenszenarien ist.

Die Betrachtung einer Prüfsummenberechnung in Lokation b ist durch die Möglichkeit des Ausweichens von dem verwendeten Replikationsszenario abhängig und wird in den Kapiteln 4.4 und 4.5 durchgeführt.

Eine Prüfsummenberechnung in Lokation c hat auf Grund der fehlenden direkten Kommunikation zwischen den Speicherkomponenten lediglich Auswirkungen auf die Wahrscheinlichkeit S_i, die im Allgemeinen durch verbesserte Prüfsummenverfahren steigen wird. Der Wert S_i wird jedoch von den Speicheranbietern festgelegt, daher muss eine Prüfsummenberechnung in Lokation c in die weiteren Betrachtungen nicht einbezogen werden.

Paritätsbits

Die einfachste Form der Fehlererkennung wird durch das Anhängen eines Paritätsbits an die zu sendende Nachricht erreicht. Innerhalb der Nachricht wird jedes von Null verschiedene Bit aufsummiert und das Paritätsbit so gewählt, dass diese Summe in jedem Fall entweder gerade oder ungerade ist [84].

Bei der Nutzung von Paritätsbits können Einzelfehler erkannt werden, da sich in diesem Fall die Parität der Summe ändert. Eine gerade Anzahl von Fehlern ändert jedoch die Parität der Summe nicht, so dass sich die Wahrscheinlichkeit unentdeckter Fehler bei der Nutzung von Paritätsbits als Summe der Wahrscheinlichkeiten, dass genau zwei, vier, usw. Fehler auftreten, berechnen lässt.

$$\begin{aligned}\overline{P} &= \binom{n}{2} \times p^2 \times (1-p)^{n-2} + \binom{n}{4} \times p^4 \times (1-p)^{n-4} + ... \\ &= \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{2i} \times p^{2i} \times (1-p)^{n-2i},\end{aligned}$$

wobei p die Wahrscheinlichkeit bezeichnet, dass ein einzelnes Bit falsch übertragen wird und für die untere Gaußklammer

$$\lfloor x \rfloor = \max\{y \in \mathbb{Z} \mid y \leq x\}$$

gilt.

Wird die zu sendende Nachricht als Matrix notiert, können sowohl für Zeilen als auch für Spalten Paritätsbits bestimmt und übertragen werden. Auf diese Weise können Einzelbitfehler nicht nur erkannt sondern auch ohne weiteren Informationsaustausch korrigiert werden. Grundsätzlich erlaubt das Verfahren ebenfalls das Erkennen von mehrfachen Bitfehlern, jedoch nicht für alle Fehlermuster, so dass eine Restwahrscheinlichkeit für unentdeckte Fehler verbleibt.

Zyklische Redundanzüberprüfungen

Ziel der zyklischen Redundanzüberprüfung [84] (Cyclic Redundancy Check, CRC) ist die fehlerfreie Übertragung von k Bits, die Informationen enthalten. Zur Redundanzerzeugung und Erkennung bzw. Korrektur von Fehlern

werden $n - k$ Bits angefügt, so dass insgesamt eine Übertragung der Länge n entsteht.

Zu diesem Zweck werden die Informationsbits als Koeffizienten eines Polynoms $P(x)$ mit Grad $k-1$ und Koeffizienten aus $\mathbb{F}_2$ aufgefasst. Das Polynom wird durch ein Generatorpolynom $G(x)$ geteilt und die Koeffizienten des resultierenden Rests $R(x)$ als Redundanzbits an die Nachrichtenbits angefügt. Der Empfänger führt eine erneute Polynomdivision mit Hilfe des Generatorpolynoms durch. Gilt $R(x) = 0$, gilt die Übertragung als fehlerfrei.

Entscheidend für die Eigenschaften und Zuverlässigkeit der zyklischen Redundanzüberprüfung ist die Wahl des Generatorpolynoms $G(x)$. Enthält $G(x)$ beispielsweise mehr als einen Term, so können alle Einzelfehler erkannt werden. Enthält $G(x)$ einen Faktor $1+x^c$, ist eine gerade Anzahl von Fehlern erkennbar. Zusätzliche Aussagen können für die Erkennung von Zweifach-, Dreifach- und Bitbündelfehlern getroffen werden [68].

Für allgemeine Blockcodes wird meist eine Wahrscheinlichkeit für unentdeckte Fehler von

$$\overline{P} \leq 2^{-(n-k)}$$

angenommen. Die obere Schranke wird jedoch von verschiedenen Codes gebrochen [54]. Für die zyklische Redundanzüberprüfung konnte gezeigt werden, dass sich die Fehlerwahrscheinlichkeit für steigende k dem Wert $2^{-(n-k)}$ annähert [99].

In Ethernetnetzwerken wird ein CRC-32, ein Code zur zyklischen Redundanzüberprüfung mit Generatorpolynom $G(x)$ von Grad 32 genutzt, wodurch sich eine Fehlerwahrscheinlichkeit von $\overline{P} \leq 2^{-32}$ ergibt. Durch die Nutzung des Generatorpolynoms $G(x) = (x + 1)$ erhält man als Spezialfall ein Paritätsbit mit gerader Parität.

Kryptographische Hashfunktionen

Im Gegensatz zu klassischen Prüfsummen werden kryptographische Hashfunktionen zum Schutz vor absichtlichen Angriffen auf die zu sendenden Nachrichten verwendet. Im Allgemeinen werden von einer solchen Funktion drei Eigenschaften erwartet, die abhängig von den spezifischen Anforderungen der kryptographischer Anwendung betrachtet werden [20]:

Kollisionsresistenz beschreibt die Eigenschaft, dass es rechnerisch unmöglich ist, zwei unterschiedliche Nachrichten mit identischem Hashwert zu finden. Für eine Hashfunktion, die einen Hashwert der Länge x erzeugt,

wird ein Aufwand von $2^{x/2}$ für eine wahrscheinliche Erzeugung einer Kollision erwartet.

Urbildresistenz beschreibt die Eigenschaft, dass es rechnerisch unmöglich ist, zu einem gegebenen Hashwert die zugehörige Nachricht mit diesem Hashwert zu finden. Für eine Hashfunktion, die einen Hashwert der Länge x erzeugt, wird ein Aufwand von 2^x für das Auffinden eines Urbildes erwartet.

Zweite Urbildresistenz beschreibt die Eigenschaft, dass es rechnerisch unmöglich ist, eine zweite Nachricht zu einer gegeben Nachricht zu finden, so dass die Hashwerte übereinstimmen. Für eine Hashfunktion, die einen Hashwert der Länge x erzeugt, wird ein Aufwand von 2^x für das sehr wahrscheinliche Auffinden eines zweiten Urbildes erwartet.

Zu den bekanntesten Vertretern gehören Hashfunktionen der Message Digest (MD) Gruppe, beispielsweise MD5 [75], und der Secure Hash Algorithm (SHA) Gruppe, beispielsweise SHA1 [63]. In einem iterativen Prozess mit verschiedenen Funktionen und Operationen wird aus einer Nachricht (MD5 + SHA1: $< 2^{64}$ Bits) ein Hashwert bestimmter Länge (MD5: 128 Bits, SHA1: 160 Bits) erzeugt und zur Fehlererkennung verwendet.

Der MD5-Algorithmus gilt heutzutage als gebrochen, das heißt seine kryptographische Sicherheit ist nicht mehr gewährleistet. Es können Kollisionen ohne den Aufwand einer Brute-Force-Attacke erzeugt werden [97]. Auch der SHA1-Algorithmus gilt nicht mehr als sicher, da Kollisionen mit weniger als 2^{69} Operationen erzeugt werden können [96]. Es wird empfohlen, neue SHA Algorithmen zu verwenden, die längere Nachrichten und Hashwerte erlauben.

Die Verwendung von MD5 und SHA1 zur Erkennung von unabsichtlichen Fehlern ist nach wie vor möglich, auch wenn im Vergleich zu den einfacheren Prüfsummen der Aufwand erhöht ist. Trotz der komplexen Bildung ist auch bei diesen Algorithmen eine Wahrscheinlichkeit von unentdeckten Fehlern anzunehmen, eine Quantifizierung ist bis jetzt nicht erfolgt. Als Abschätzung wird im Folgenden ein Wert von $\overline{P} \leq 2^{-64}$ angenommen.

Zusammenfassung

Die Verwendung von Prüfsummen ist eine weit verbreitete Maßnahme zur Überprüfung der Datenintegrität. Sowohl Fehler in den Daten als auch in der verwendeten Prüfsumme sollten erkannt werden können. In Tabelle 4.1

	Länge	$\overline{P}$	Verwendung
Paritätsbit	1	$\sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} (\binom{n}{2i}) p^{2i}(1-p)^{n-2i}$	Speichermodule, Netzwerktechnik
CRC	variabel	$\leq 2^{-(n-k)}$ für große k	Telekommunikation, Bluetooth, USB
CRC-32	32	$\leq 2^{-32} \approx 2,33 \times 10^{-10}$	Ethernet, PNG, MPEG-2, SATA
Fletcher [34] / Adler [23]	16, 32, 64 / 32	$\leq 2^{-(n-k)}$	zlib
MD5 / SHA-256	128 / 256	$\leq 2^{-64}$	Datentransfer, SSH, Git

Tabelle 4.1: Übersicht der Wahrscheinlichkeiten unentdeckter Fehler der verbreitetsten Prüfsummenalgorithmen inklusive beispielhafter Anwendungsgebiete (p: Wahrscheinlichkeit eines einzelnen Bitfehlers, n: Anzahl der Nachrichtenbits, k: Anzahl der Informationsbits).

sind die häufigsten Prüfsummenalgorithmen mit ihren jeweiligen möglichen Längen und Wahrscheinlichkeiten unentdeckter Fehler zusammengefasst. Im Allgemeinen reduziert eine größere Anzahl von Redundanzbits erwartungsgemäß die Wahrscheinlichkeit unentdeckter Fehler, führt jedoch ebenfalls zu erhöhtem Aufwand und Speicherplatzbedarf. Auch einfache Prüfsummenverfahren mit relativ großer Fehlerwahrscheinlichkeit werden nach wie vor in vielen aktuellen Technologien verwendet und lassen sich beispielsweise aus Kompatibilitätsgründen oft nur schwer ersetzen.

4.4 Replikationsszenarien aus Systemsicht

Zunächst werden die verschiedenen Replikationsszenarien aus Systemsicht betrachtet. In diesen Szenarien werden Aussagen für die Wahrscheinlichkeit von unentdeckten Fehlern im gesamten System hergeleitet, die während der Schreib- sowie Leseoperation auftreten können. Die Herleitungen der Wahrscheinlichkeit, dass der Nutzer in den verschiedenen Replikationsszenarien eine fehlerhafte Datei zurückerhält, wird in Kapitel 4.5 beschrieben.

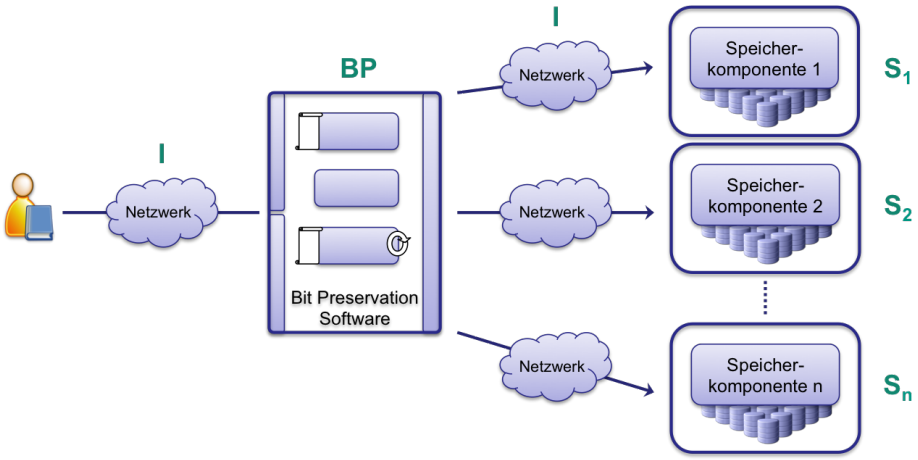

Abbildung 4.2: Szenario 1 - Synchrone Replikation (schematische Darstellung).

4.4.1 Szenario 1: Synchrone Replikation

Das erste betrachtete Replikationsszenario ist die synchrone Replikation wie in Abbildung 4.2 dargestellt. Eine zu speichernde Datei wird unmittelbar in allen verfügbaren Speicherkomponenten abgelegt. Bei einer Leseoperation wird zufällig eine Speicherkomponente ausgewählt.

Szenario 1a: Keine Prüfsummenberechnung
Das einfachste Szenario bei synchroner Replikation ist die Replikation in zwei Speicherkomponenten ohne Berechnung einer Prüfsumme. Zur Verdeutlichung werden zunächst Schreib- und Leseoperation getrennt betrachtet.

$$\text{Schreiben:} \quad \overline{P}_{uFehler} = S_1 \times S_2 \times BP \times I^3$$
$$\text{Lesen:} \quad \overline{P}_{uFehler} = \frac{1}{2} \times (S_1 + S_2) \times BP \times I^2$$

Beim Schreiben wird die Datei auf beide Speicherkomponenten verteilt, so dass die jeweiligen Fehlerwahrscheinlichkeiten berücksichtigt werden müssen. Beim Lesen wird von einer gleichmäßigen Verteilung der Zugriffe auf beide Speicherkomponenten ausgegangen. Insgesamt ergibt sich auf diese Weise für die Wahrscheinlichkeit unentdeckter Fehler durch Multiplikation der Fehlerwahrscheinlichkeiten der beiden Teiloperationen

$$\overline{P}_{uFehler} = \frac{1}{2} \times S_1 \times S_2 \times (S_1 + S_2) \times BP^2 \times I^5.$$

Das Vorgehen lässt sich zur Betrachtung von n Repliken verallgemeinern und die Wahrscheinlichkeit unentdeckter Fehler ergibt sich in diesem Szenario zu

$$\overline{P}_{uFehler} = \frac{1}{n} \times \prod_{i=1}^{n} S_i \times (\sum_{i=1}^{n} S_i) \times BP^2 \times I^{n+3}. \tag{4.1}$$

Die Hinzunahme einer weiteren Speicherkomponente lässt sich mit Hilfe von Gleichung 4.1 als Faktor beschreiben:

$$\overline{P}_{uFehler}(n+1 \text{ Repliken}) = \frac{I \times n \times (\sum_{i=1}^{n+1} S_i)}{(n+1) \times (\sum_{i=1}^{n} S_i)} \times \overline{P}_{uFehler}(n \text{ Repliken}).$$

Szenario 1b: Einmalige Prüfsummenberechnung
Wird in der Softwarekomponente (Lokation b) beim Lesen eine Prüfsumme berechnet, ist im Fehlerfall ein Ausweichen auf eine alternative Speicherkomponente möglich. Allgemein wird von einer zyklischen Anordnung der Speicherkomponenten ausgegangen. Beinhaltet ein Replikationsszenario n Speicherkomponenten, entspricht Speicherkomponente $n+1$ der Speicherkomponente 1, usw. Im einfachsten Szenario werden zwei Speicherkomponenten betrachtet, so dass lediglich ein einmaliges Ausweichen möglich ist. Zur Verdeutlichung werden erneut Schreib- und Leseoperation getrennt betrachtet.

$$\begin{aligned} \text{Schreiben: } \overline{P}_{uFehler} =& S_1 \times S_2 \times BP \times I^3 \\ \text{Lesen: } \overline{P}_{uFehler} =& \frac{1}{2} \times BP \times I^2 \times (S_1 + S_2 \\ & + P \times BP \times I \times (\overline{S_1} \times S_2 + S_1 \times \overline{S_2})) \end{aligned}$$

Die Prüfsumme wird mit der Fehlerwahrscheinlichkeit P bezeichnet und lediglich beim Lesen berechnet, daher ergeben sich keine Änderungen der Fehlerwahrscheinlichkeit bei der Schreiboperation. Bei der Leseoperation ergeben sich zwei zusätzliche Summanden, die das Ausweichen auf die alternative Speicherkomponente im Fehlerfall beschreiben. Die in den Summanden auftretenden Fehlerwahrscheinlichkeiten $\overline{S_1}$ und $\overline{S_2}$ kennzeichnen die Gegenwahrscheinlichkeit von S_1 und S_2. Durch Multiplikation kann die Gesamtfehlerwahrscheinlichkeit ermittelt werden:

$$\overline{P}_{uFehler} = \frac{1}{2} \times S_1 \times S_2 \times BP^2 \times I^5 \times (S_1 + S_2 + P \times BP \times I \times (\overline{S_1} \times S_2 + S_1 \times \overline{S_2})).$$

Durch Verallgemeinerung auf n Repliken ergibt sich für die Wahrscheinlichkeit unentdeckter Fehler bei einmaligem Ausweichen für $n \geq 2$

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times \prod_{i=1}^{n} S_i \times BP^2 \times I^{n+3} \times (\sum_{i=1}^{n} S_i + P \times BP \times I \\ & \times (\sum_{i=1}^{n} \overline{S_i} \times S_{i+1})),\end{aligned} \tag{4.2}$$

wobei stets ein Ausweichen von Speicherkomponente i auf Speicherkomponente $i+1$ angenommen wird.

Erlaubt man zweifaches Ausweichen, besteht der Minimalfall aus drei Speicherkomponenten. Von Speicherkomponente i wird dabei stets auf die Speicherkomponente $i+1$ und ggf. Speicherkomponente $i+2$ ausgewichen. Auch in diesem Szenario ergeben sich keine Änderungen bei der Schreiboperation, lediglich die Leseoperation ist betroffen. Für die Gesamtwahrscheinlichkeit unentdeckter Fehler mit $n \geq 3$ gilt

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times \prod_{i=1}^{n} S_i \times BP^2 \times I^{n+3} \times (\sum_{i=1}^{n} S_i + P \times BP \times I \\ & \times (\sum_{i=1}^{n} \overline{S_i} \times S_{i+1}) + (P \times BP \times I)^2 \times \sum_{i=1}^{n} \overline{S_i} \times \overline{S_{i+1}} \times S_{i+2}).\end{aligned}$$

In diesem Szenario bleiben kein und einfaches Ausweichen möglich und werden zur Bestimmung der Fehlerwahrscheinlichkeit herangezogen. Zusätzlich ergibt sich der letzte Summand, der zweimaliges Berechnen der Prüfsumme und zweimaliges Ausweichen beinhaltet. Generalisierung auf m-faches Ausweichen mit $1 \leq m \leq n-1$ ergibt eine Wahrscheinlichkeit für unentdeckte Fehler von

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times \prod_{i=1}^{n} S_i \times BP^2 \times I^{n+3} \times (\sum_{i=1}^{m+1} (P \times BP \times I)^{i-1} \\ & \times (\sum_{j=1}^{n} S_{i+j-1} \times \prod_{k=j}^{i+j-2} \overline{S_k})).\end{aligned}$$

Die erste Summe beschreibt dabei alle Terme, die durch kein, einfaches, zweifaches, ..., m-faches Ausweichen zusätzlich entstehen. Die zweite Summe

beschreibt alle Kombinationsmöglichkeiten von Speicherkomponenten, die vom jeweiligen Ausweichen berührt werden.

Hinzufügen einer n-ten Speicherkomponente
Beispielsweise im Szenario der synchronen Replikation mit einfachem Ausweichen lässt sich eine Aussage über die Zuverlässigkeit einer n-ten Speicherkomponente bei einer existierenden Menge von Speicherkomponenten $1, ..., n-1$ ableiten. Wird die Zuverlässigkeit der ersten $n-1$ Speicherkomponenten als konstant angenommen, ergibt sich durch Umformen und Differenzieren von Formel 4.2 nach S_n

$$\begin{aligned}\partial S_n :&\frac{1}{n} \times BP^2 \times I^{n+3} \times \prod_{i=1}^{n-1} S_i \times (\sum_{i=1}^{n} S_i + P \times BP \times I \times \sum_{i=1}^{n} \overline{S_i} \times S_{i+1}) \\ &+ \frac{1}{n} \times BP^2 \times I^{n+3} \times \prod_{i=1}^{n} S_i \times (1 + P \times BP \times I \times (\overline{S_{n-1}} - S_1)) \stackrel{!}{=} 0 \\ \Leftrightarrow S_n &= \frac{-\sum_{i=1}^{n-1} S_i - P \times BP \times I \times \sum_{i=1}^{n-2} \overline{S_i} S_{i+1} - P \times BP \times I \times S_1}{2(1 + P \times BP \times I \times \overline{S_{n-1}} - P \times BP \times I \times S_1)}.\end{aligned}$$

Als hinreichendes Kriterium eines Maximums, was in diesem Szenario die maximale Zuverlässigkeit des Gesamtsystems beschreibt, muss die zweite Ableitung einen Wert kleiner Null annehmen. Die erhaltenen Faktoren sind nach Definition größer Null, es muss daher lediglich der geklammerte Ausdruck betrachtet werden.

$$\begin{aligned}\partial^2 S_n :&\frac{2}{n} \times BP^2 \times I^{n+3} \times \prod_{i=1}^{n-1} S_i \times (1 + P \times BP \times I \times (\overline{S_{n-1}} - S_1)) \stackrel{!}{<} 0 \\ \Leftrightarrow S_1 &> \frac{1}{P \times BP \times I} + \overline{S_{n-1}}\end{aligned}$$

Die Existenz eines Maximums hängt folglich nur von den direkt „benachbarten“ Speicherkomponenten ab. Das entspricht den Speicherkomponenten, die bei einmaligem Ausweichen mit der zusätzlichen Speicherkomponente n in Kontakt stehen.

4.4.2 Szenario 2: Asynchrone Replikation mit Masterdatei

Bei der asynchronen Replikation mit Masterdatei wird die zu speichernde Datei in einer Speicherkomponente abgelegt, o.B.d.A. Speicherkomponente 1 wie

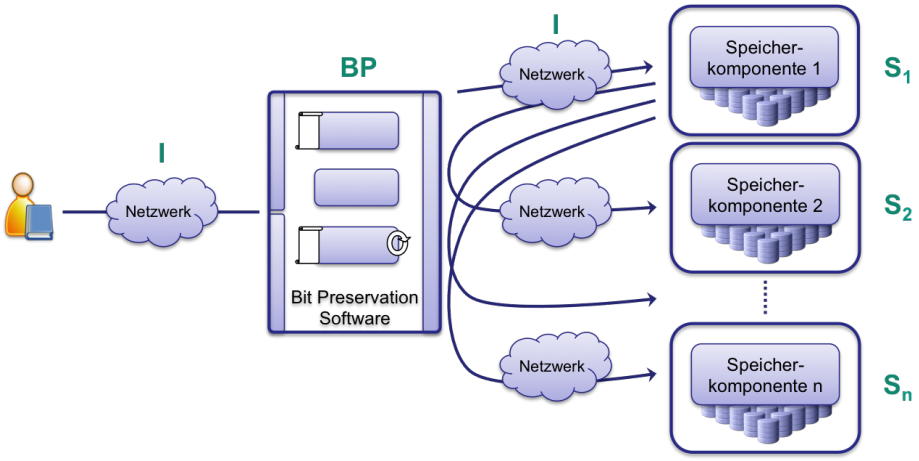

Abbildung 4.3: Szenario 2 - Asynchrone Replikation mit Masterdatei (schematische Darstellung).

in Abbildung 4.3 dargestellt, und auf diese Weise eine Masterdatei erstellt, die von dieser Stelle in die anderen Speicherkomponenten repliziert wird. Für die Leseoperation wird eine zufällige Speicherkomponente ausgewählt. Die Speicherkomponenten sind beim Ausweichen auf eine alternative Replik zyklisch angeordnet.

Szenario 2a: Keine Prüfsummenberechnung
Für die Untersuchung der asynchronen Replikation mit Masterdatei wird ein System mit zwei Speicherkomponenten betrachtet. Bei der Schreiboperation muss sowohl die Speicherung in Speicherkomponente 1 als auch die folgende Replikation von Speicherkomponente 1 in Speicherkomponente 2 berücksichtigt werden.

$$\text{Schreiben: } \overline{P}_{uFehler} = (S_1 \times BP \times I^2) \times (S_1 \times S_2 \times BP \times I^2)$$
$$\text{Lesen: } \overline{P}_{uFehler} = \frac{1}{2} \times (S_1 + S_2) \times BP \times I^2$$

Die Leseoperation entspricht der Leseoperation bei synchroner Replikation, so dass sich insgesamt eine Wahrscheinlichkeit unentdeckter Fehler von

$$\overline{P}_{uFehler} = \frac{1}{2} \times S_1^2 \times S_2 \times (S_1 + S_2) \times BP^3 \times I^6$$

ergibt. Nimmt man eine weitere Speicherkomponente i hinzu, ergibt sich für die Schreiboperation stets ein zusätzlicher Faktor von $S_1 \times S_i \times BP \times I^2$. Verallgemeinert auf n Repliken ergibt sich die Wahrscheinlichkeit unentdeckter Fehler zu

$$\overline{P}_{uFehler} = \frac{1}{n} \times S_1^n \times \prod_{i=2}^{n} S_i \times (\sum_{i=1}^{n} S_i) \times BP^{n+1} \times I^{2(n+1)}.$$

Szenario 2b: Einmalige Prüfsummenberechnung
Basierend auf den Erkenntnissen von Kapitel 4.4.1 lassen sich die Wahrscheinlichkeiten eines unentdeckten Fehlers für die Lese- sowie die Schreiboperation bei n Repliken und m-maligen Ausweichen mit $0 \leq m \leq n-1$ direkt angeben.

$$\begin{aligned}
\text{Schreiben:} \qquad & \overline{P}_{uFehler} = S_1^n \times \prod_{i=2}^{n} S_i \times BP^n \times I^{2n} \\
\text{Lesen:} \qquad & \overline{P}_{uFehler} = \frac{1}{n} \times BP \times I^2 \times (\sum_{i=1}^{m+1} (P \times BP \times I)^{i-1} \\
& \qquad \times (\sum_{j=1}^{n} S_{i+j-1} \times \prod_{k=j}^{i+j-2} \overline{S_k}))
\end{aligned}$$

Die Wahrscheinlichkeiten für die Leseoperation entsprechen dem Szenario der synchronen Replikation, lediglich die Schreiboperation muss an das aktuelle Replikationsszenario angepasst werden. Durch Multiplikation der Einzelwahrscheinlichkeiten für die Schreib- und Leseoperation ergibt sich insgesamt für die Wahrscheinlichkeit eines unentdeckten Fehlers

$$\begin{aligned}
\overline{P}_{uFehler} = & \frac{1}{n} \times S_1^n \times \prod_{i=2}^{n} S_i \times BP^{n+1} \times I^{2(n+1)} \times \\
& (\sum_{i=1}^{m+1} (P \times BP \times I)^{i-1} \times (\sum_{j=1}^{n} S_{i+j-1} \times \prod_{k=j}^{i+j-2} \overline{S_k})).
\end{aligned}$$

4.4.3 Szenario 3: Asynchrone Replikation ohne Masterdatei

Bei der asynchronen Replikation ohne Masterdatei wird die Datei zunächst in einer Speicherkomponente, o.B.d.A. Speicherkomponente 1, abgelegt, jedoch

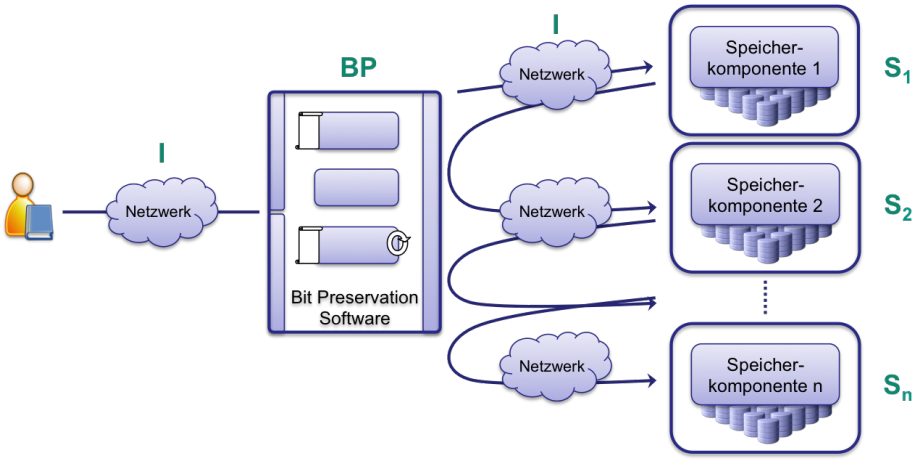

Abbildung 4.4: Szenario 3 - Asynchrone Replikation ohne Masterdatei (schematische Darstellung).

nicht wie im vorherigen Szenario nur aus dieser Speicherkomponente repliziert. Die weiteren Replikationen erfolgen sukzessive wie in Abbildung 4.4 dargestellt, eine Datei in Speicherkomponente 2 wurde von Speicherkomponente 1 nach 2 repliziert, eine Datei in Speicherkomponente 3 wurde von Speicherkomponente 1 nach 2 nach 3 repliziert, usw. Für die Leseoperation wird eine zufällige Speicherkomponente ausgewählt. Die Speicherkomponenten sind beim Ausweichen auf eine alternative Replik zyklisch angeordnet.

Szenario 3a: Keine Prüfsummenberechnung
Im Szenario mit lediglich zwei Speicherkomponenten existiert kein Unterschied zwischen asynchroner Replikation mit oder ohne Masterdatei und die Wahrscheinlichkeit unentdeckter Fehler ergibt sich ebenfalls zu

$$\overline{P}_{uFehler} = \frac{1}{2} \times S_1^2 \times S_2 \times (S_1 + S_2) \times BP^3 \times I^6.$$

Bei der Hinzunahme einer weiteren Speicherkomponente i muss in diesem Replikationsszenario ein Faktor $S_{i-1} \times S_i \times BP \times I^2$ hinzugenommen werden. Verallgemeinert auf n Repliken erhält man auf diese Weise für die Wahrscheinlichkeit eines unentdeckten Fehlers im System

$$\overline{P}_{uFehler} = \frac{1}{n} \times S_n \times \prod_{i=2}^{n} S_{i-1}^2 \times (\sum_{i=1}^{n} S_i) \times BP^{n+1} \times I^{2(n+1)}.$$

Szenario 3b: Einmalige Prüfsummenberechnung
Analog zur asynchronen Replikation mit Masterdatei lässt sich die Wahrscheinlichkeit eines unentdeckten Fehlers direkt aus den Erkenntnissen aus Kapitel 4.4.1 ableiten. Lediglich die Fehlerwahrscheinlichkeit der Schreiboperation unterscheidet sich in den verschiedenen Szenarien, so dass sich mit

$$\text{Schreiben: } \overline{P}_{uFehler} = S_n \times \prod_{i=2}^{n} S_{i-1}^2 \times BP^n \times I^{2n}$$

die Wahrscheinlichkeit eines unentdeckten Fehlers bei n Repliken und m-maligen Ausweichen mit $0 \leq m \leq n-1$ zu

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_n \times \prod_{i=2}^{n} S_{i-1}^2 \times BP^{n+1} \times I^{2(n+1)} \times \\ & (\sum_{i=1}^{m+1} (P \times BP \times I)^{i-1} \times (\sum_{j=1}^{n} S_{i+j-1} \times \prod_{k=j}^{i+j-2} \overline{S_k}))\end{aligned}$$

ergibt.

Beispielhaft stehen dem Nutzer zwei Speicherkomponenten mit den Fehlerwahrscheinlichkeiten $S_1 = 1 - 2^{-24}$ und $S_2 = 1 - 2^{-48}$ zur Verfügung. Für das Netzwerk wird die Fehlerwahrscheinlichkeit $I = 1 - 2^{-32}$ und für die Bit Preservation Software $BP = 1 - 2^{-48}$ angenommen. Entscheidet der Nutzer sich für eine synchrone Replikation, lässt sich eine Wahrscheinlichkeit für mindestens einen unentdeckten Fehler von $9,057 \times 10^{-8}$ ableiten. In diesem Szenario ist eine von 11.041.046 Operationen während des Schreib- und Leseprozesses fehlerhaft. Wird zusätzlich eine einfache Prüfsumme mit der Fehlerwahrscheinlichkeit $P = 1 - 2^{-16}$ in Lokation b berechnet, verringert sich die Wahrscheinlichkeit für mindestens einen unentdeckten Fehler auf $6,077 \times 10^{-8}$ und eine von 16.455.687 Operationen ist fehlerhaft. Im Szenario der asynchronen Replikation ergibt sich eine Wahrscheinlichkeit von $1,504 \times 10^{-7}$ bzw. $1,206 \times 10^{-7}$ inklusive Berechnung einer Prüfsumme. Entsprechend ist eine von 6.648.556 bzw. 8.291.411 Operationen fehlerhaft.

Große Dateien werden für den Netzwerktransfer in kleinere Datenpakete aufgeteilt. Bei einer Übertragung einer Datei der Größe vier Gigabyte kann man auf Grund der Beschränkung auf 1452 Datenbytes bei TCP/IP ca.

3.000.000 Millionen nötige Operationen annehmen. Für die oben bestimmten Fehlerwahrscheinlichkeiten lässt sich unter der Annahme dieser Menge unabhängiger Operationen daher vereinfacht ableiten, dass im Schnitt eine von vier (synchrone Replikation ohne Prüfsumme), sechs (synchrone Replikation mit Prüfsumme) bzw. drei (asynchrone Replikation mit und ohne Prüfsumme) Dateien im System fehlerhaft ist.

4.5 Replikationsszenarien aus Nutzersicht

Zusätzlich zu den drei bereits beschriebenen Replikationsszenarien aus Systemsicht, können die Szenarien aus Nutzersicht untersucht werden. Im Gegensatz zur Systemsicht werden nicht die Schreib- und Leseoperation für das gesamte System betrachtet, sondern die Wahrscheinlichkeiten unentdeckter Fehler für jede Speicherkomponente bestimmt und gemäß der Annahme einer zufälligen Auswahl der Speicherkomponente gemittelt. Insbesondere können mit dieser Herangehensweise Abhängigkeiten der Speicherkomponenten in den verschiedenen Replikationsszenarien modelliert werden. Die ermittelte Fehlerwahrscheinlichkeit entspricht der Wahrscheinlichkeit, dass der Nutzer eine fehlerhafte Datei zurückerhält.

4.5.1 Szenario 4: Synchrone Replikation

Den Startpunkt stellt erneut die synchrone Replikation nach Abbildung 4.2 dar.

Szenario 4a: Keine Prüfsummenberechnung
Der Minimalfall bei synchroner Replikation weist zwei Speicherkomponenten auf, die getrennt während der Schreib- und Leseoperation betrachtet werden.

$$\begin{aligned}
\text{Schreiben } S_1\text{:}\quad \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Lesen } S_1\text{:}\quad \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Schreiben } S_2\text{:}\quad \overline{P}_{uFehler} &= S_2 \times BP \times I^2 \\
\text{Lesen } S_2\text{:}\quad \overline{P}_{uFehler} &= S_2 \times BP \times I^2
\end{aligned}$$

Auf Grund des Replikationsschemas wird die Fehlerwahrscheinlichkeit in der jeweiligen Speicherkomponente nicht von der Fehlerwahrscheinlichkeit der anderen Speicherkomponenten beeinflusst. Komponentenweise Multiplikation der Wahrscheinlichkeit der Schreib- und Leseoperation und anschließendes

Aufsummieren ergibt bei einer gleichmäßigen Verteilung der Zugriffe auf die Speicherkomponenten

$$\overline{P}_{uFehler} = \frac{1}{2} \times (S_1^2 + S_2^2) \times BP^2 \times I^4.$$

Wird eine weitere Speicherkomponente S_i hinzugefügt, ergibt sich jeweils der zusätzliche Summand $S_i^2 \times BP^2 \times I^4$, so dass allgemein für die Wahrscheinlichkeit unentdeckter Fehler bei n Repliken gilt

$$\overline{P}_{uFehler} = \frac{1}{n} \times \sum_{i=1}^{n} S_i^2 \times BP^2 \times I^4.$$

Szenario 4b: Einmalige Prüfsummenberechnung
Die Berechnung einer Prüfsumme während der Leseoperation ermöglicht ein Ausweichen auf alternative Speicherkomponenten. Der betrachtete Minimalfall weist zwei Speicherkomponenten auf und erlaubt daher maximal einmaliges Ausweichen. Die Wahrscheinlichkeit eines unentdeckten Fehlers wird für die Speicherkomponenten getrennt betrachtet.

$$\begin{aligned} S_1\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times BP^2 \times I^4 + \overline{S_1} \times S_2^2 \times P \times BP^3 \times I^5 \\ S_2\text{:}\ \overline{P}_{uFehler} &= S_2^2 \times BP^2 \times I^4 + \overline{S_2} \times S_1^2 \times P \times BP^3 \times I^5. \end{aligned}$$

Wie im Szenario ohne Prüfsummenberechnung beeinflussen sich die Fehlerwahrscheinlichkeiten der einzelnen Speicherkomponenten nicht. Die Berechnung einer Prüfsumme erzeugt jeweils einen zusätzlichen Faktor von $P \times BP \times I$. Aufsummieren ergibt eine Gesamtfehlerwahrscheinlichkeit von

$$\overline{P}_{uFehler} = \frac{1}{2} \times BP^2 \times I^4 \times (S_1^2 + S_2^2 + P \times BP \times I \times (\overline{S_1} \times S_2^2 + \overline{S_2} \times S_1^2)).$$

Verallgemeinerung auf n Repliken liefert für eine zyklische Anordnung der Speicherkomponenten

$$\overline{P}_{uFehler} = \frac{1}{n} \times BP^2 \times I^4 \times (\sum_{i=1}^{n} S_i^2 + P \times BP \times I \times \sum_{i=1}^{n} \overline{S_{i-1}} S_i^2).$$

Wird mehrmaliges Ausweichen erlaubt, werden für jeden Ausweichschritt zusätzliche Summanden in der Klammer aufgenommen.

$$\text{Zweimaliges Ausweichen: } (P \times BP \times I)^2 \times \sum_{i=1}^{n} \overline{S_{i-2}S_{i-1}} S_i^2$$

$$\text{Dreimaliges Ausweichen: } (P \times BP \times I)^3 \times \sum_{i=1}^{n} \overline{S_{i-3}S_{i-2}S_{i-1}} S_i^2$$

Verallgemeinert auf m-maliges Ausweichen mit $0 \leq m \leq n-1$ erhält man für die Wahrscheinlichkeit eines unentdeckten Fehlers

$$\overline{P}_{uFehler} = \frac{1}{n} \times BP^2 \times I^4 \times (\sum_{i=0}^{m} (P \times BP \times I)^i \times (\sum_{j=1}^{n} S_j^2 \times \prod_{k=1}^{i} \overline{S_{j-k}})).$$

4.5.2 Szenario 5: Asynchrone Replikation mit Masterdatei

Das zweite betrachtete Replikationsszenario ist die asynchrone Replikation mit Masterdatei wie in Abbildung 4.3 dargestellt. Es wird die Wahrscheinlichkeit hergeleitet, dass ein Nutzer eine fehlerhafte Datei in diesem Szenario zurückerhält.

Szenario 5a: Keine Prüfsummenberechnung
Der Minimalfall bei asynchroner Replikation beinhaltet zwei Speicherkomponenten. Bei der Betrachtung von Schreib- und Leseoperation müssen die Speicherkomponenten in diesem Szenario getrennt behandelt werden.

$$\begin{aligned}
\text{Schreiben } S_1\text{: } \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Lesen } S_1\text{: } \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Schreiben } S_2\text{: } \overline{P}_{uFehler} &= S_1^2 \times S_2 \times BP^2 \times I^4 \\
\text{Lesen } S_2\text{: } \overline{P}_{uFehler} &= S_2 \times BP \times I^2
\end{aligned}$$

Die Gesamtwahrscheinlichkeit ergibt sich durch komponentenweise Multiplikation der Wahrscheinlichkeiten für Schreib- und Leseoperation und anschließendes Aufsummieren. Es wird erneut von einer gleichmäßigen Verteilung der Zugriffe auf die Speicherkomponenten ausgegangen.

$$\overline{P}_{uFehler} = \frac{1}{2} \times S_1^2 \times BP^2 \times I^4 \times (1 + S_2^2 \times BP \times I^2)$$

Allgemein für n Speicherkomponenten ergibt sich

$$\overline{P}_{uFehler} = \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (1 + BP \times I^2 \times (\sum_{i=2}^{n} S_i^2)). \tag{4.3}$$

Die Hinzunahme einer weiteren Speicherkomponente lässt sich mit Hilfe von Gleichung 4.3 als Faktor beschreiben.

$$\begin{aligned}\overline{P}_{uFehler}(n+1 \text{ Repliken}) =& \frac{n \times (1 + BP \times I^2 \times (\sum_{i=2}^{n+1} S_i^2))}{(n+1) \times (1 + BP \times I^2 \times (\sum_{i=2}^{n} S_i^2))} \\ & \times \overline{P}_{uFehler}(n \text{ Repliken})\end{aligned}$$

Szenario 5b: Einmalige Prüfsummenberechnung
Wird beim Lesen eine Prüfsumme berechnet und einfaches Ausweichen erlaubt, beinhaltet der Minimalfall bei der asynchronen Replikation ebenfalls zwei Speicherkomponenten. Zur Verdeutlichung der Vorgänge werden die Speicherkomponenten separat betrachtet.

$$\begin{aligned}S_1\text{: } \overline{P}_{uFehler} =& S_1^2 \times BP^2 \times I^4 + \overline{S_1} \times S_1^2 \times S_2^2 \times P \times BP^4 \times I^7 \\ S_2\text{: } \overline{P}_{uFehler} =& S_1^2 \times S_2^2 \times BP^3 \times I^6 + \overline{S_2} \times S_1^2 \times P \times BP^3 \times I^5\end{aligned}$$

Der jeweils erste Summand beschreibt die Fehlerwahrscheinlichkeit ohne Ausweichen. Tritt ein Fehler auf, wird auf die jeweils andere Speicherkomponente ausgewichen, was durch den zweiten Summanden repräsentiert wird. In diesem Szenario erhält man als zusätzlichen Faktor den Einfluss der Prüfsummenberechnung $P \times BP \times I$. Nimmt man eine gleichmäßige Verteilung der Zugriffe auf die beiden Speicherkomponenten an, ergibt sich für die Gesamtfehlerwahrscheinlichkeit

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{2} \times S_1^2 \times BP^2 \times I^4 \times (1 + BP \times I^2 \times S_2^2 \\ & + P \times BP^2 \times I^3 \times \overline{S_1} \times S_2^2 + \overline{S_2} \times P \times BP \times I).\end{aligned}$$

Verallgemeinerung auf n Speicherkomponenten mit $n \geq 2$ liefert

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (1 + BP \times I^2 \times (\sum_{i=2}^{n} S_i^2) \\ & + P \times BP^2 \times I^3 \times (\sum_{i=2}^{n} \overline{S_{i-1}} \times S_i^2) + \overline{S_n} \times P \times BP \times I).\end{aligned}$$

Wird zweifaches Ausweichen erlaubt, ergibt sich die Fehlerwahrscheinlichkeit für $n \geq 3$ zu

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (1 + BP \times I^2 \times (\sum_{i=2}^{n} S_i^2) + P \times BP^2 \times I^3 \\ & \times (\sum_{i=2}^{n} \overline{S_{i-1}} \times S_i^2) + P^2 \times BP^3 \times I^4 \times (\sum_{i=2}^{n} \overline{S_{i-2}} \times \overline{S_{i-1}} \times S_i^2) \\ & + \overline{S_{n-1}} \times \overline{S_n} \times P^2 \times BP^2 \times I^2 + \overline{S_n} \times P \times BP \times I).\end{aligned}$$

Es ist zu beachten, dass $S_0 = S_n$ gilt.

Generalisierung auf m-faches Ausweichen mit $1 \leq m \leq n-1$ ergibt eine Wahrscheinlichkeit für unentdeckte Fehler von

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (\sum_{i=0}^{m} P^i \times BP^{i+1} \times I^{i+2} \\ & \times (\sum_{j=2}^{n} S_j^2 \times \prod_{k=1}^{i} \overline{S_{j-k}}) + (P \times BP \times I)^i \times \prod_{l=1}^{i} \overline{S_{n-l+1}}).\end{aligned}$$

4.5.3 Szenario 6: Asynchrone Replikation ohne Masterdatei

Das letzte betrachtete Replikationsszenario ist die asynchrone Replikation ohne Masterdatei wie in Abbildung 4.4 dargestellt. Es wird die Wahrscheinlichkeit hergeleitet, dass ein Nutzer eine fehlerhafte Datei in diesem Szenario zurückerhält.

Szenario 6a: Keine Prüfsummenberechnung
Der Minimalfall bei asynchroner Replikation entspricht dem Minimalfall in Kapitel 4.5.2. Zur Verdeutlichung des Unterschiedes der asynchronen Replikationsszenarien wird an dieser Stelle die Replikation mit drei Speicherkomponenten betrachtet.

$$
\begin{aligned}
\text{Schreiben } S_1\text{:}\ \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Lesen } S_1\text{:}\ \overline{P}_{uFehler} &= S_1 \times BP \times I^2 \\
\text{Schreiben } S_2\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times S_2 \times BP^2 \times I^4 \\
\text{Lesen } S_2\text{:}\ \overline{P}_{uFehler} &= S_2 \times BP \times I^2 \\
\text{Schreiben } S_3\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times S_2^2 \times S_3 \times BP^3 \times I^6 \\
\text{Lesen } S_3\text{:}\ \overline{P}_{uFehler} &= S_3 \times BP \times I^2
\end{aligned}
$$

Durch komponentenweise Multiplikation der Wahrscheinlichkeiten und anschließendes Aufsummieren ergibt sich bei gleichmäßiger Verteilung der Zugriffe auf die Speicherkomponenten

$$\overline{P}_{uFehler} = \frac{1}{3} \times S_1^2 \times BP^2 \times I^4 \times (1 + S_2^2 \times BP \times I^2 + S_2^2 \times S_3^2 \times BP^2 \times I^4).$$

Allgemein gilt in einem System mit n Speicherkomponenten für die Wahrscheinlichkeit unentdeckter Fehler

$$\overline{P}_{uFehler} = \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (\sum_{i=1}^{n} BP^{i-1} \times I^{2(i-1)} \times \prod_{j=2}^{i} S_j^2).$$

Szenario 6b: Einmalige Prüfsummenberechnung
Betrachtet wird erneut das Szenario mit drei Speicherkomponenten. Wird beim Lesen eine Prüfsumme in Lokation b berechnet und im Fehlerfall einfaches Ausweichen erlaubt, ergeben sich für die einzelnen Speicherkomponenten die folgenden Fehlerwahrscheinlichkeiten:

$$
\begin{aligned}
S_1\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times BP^2 \times I^4 + \overline{S_1} \times S_1^2 \times S_2^2 \times P \times BP^4 \times I^7 \\
S_2\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times S_2^2 \times BP^3 \times I^6 + \overline{S_2} \times S_1^2 \times S_2^2 \times S_3^2 \times P \times BP^5 \times I^9 \\
S_3\text{:}\ \overline{P}_{uFehler} &= S_1^2 \times S_2^2 \times S_3^2 \times BP^4 \times I^8 + \overline{S_3} \times S_1^2 \times P \times BP^3 \times I^5
\end{aligned}
$$

Der erste Summand beschreibt die Fehlerwahrscheinlichkeit ohne Ausweichen, der zweite Summand beschreibt die Fehlerwahrscheinlichkeit bei Ausweichen von Speichkomponente i auf Speicherkomponente i+1. Bei gleichmässiger Verteilung der Zugriffe auf die Speicherkomponenten ergibt sich für die Gesamtfehlerwahrscheinlichkeit

$$\begin{aligned}\overline{P}_{uFehler} = \frac{1}{3} \times S_1^2 \times BP^2 \times I^4 \times (1 + S_2^2 \times BP \times I^2 + S_2^2 \times S_3^2 \times BP^2 \times I^4 \\ + \overline{S_1} \times S_2^2 \times P \times BP^2 \times I^3 + \overline{S_2} \times S_2^2 \times S_3^2 \times P \times BP^3 \times I^5 \\ + \overline{S_3} \times P \times BP \times I + \overline{S_1} \times \overline{S_2} \times S_2^2 \times S_3^2 \times P^2 \times BP^4 \times I^6 \\ + \overline{S_2} \times \overline{S_3} \times P^2 \times BP^2 \times I^2 + \overline{S_3} \times \overline{S_1} \times S_2^2 \times P^2 \times BP^3 \times I^4)\end{aligned}$$

Verallgemeinerung auf n Speicherkomponenten mit $n \geq 2$ und $S_0 = S_n$ liefert

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (1 + \sum_{i=2}^{n} BP^{i-1} \times I^{2(i-1)} \times \prod_{j=2}^{i} S_j^2 \\ &+ \sum_{i=2}^{n} \overline{S_{i-1}} \times P \times BP^i \times I^{2i-1} \times \prod_{j=2}^{i} S_j^2 + \overline{S_n} \times P \times BP \times I) \\ =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (\sum_{i=1}^{n} BP^{i-1} \times I^{2(i-1)} \times \prod_{j=2}^{i} S_j^2 \\ &+ \sum_{i=1}^{n} \overline{S_{i-1}} \times P \times BP^i \times I^{2i-1} \times \prod_{j=2}^{i} S_j^2)\end{aligned}$$

Wird zweifaches Ausweichen erlaubt, ergibt sich die Fehlerwahrscheinlichkeit mit $n \geq 3$ zu

$$\begin{aligned}\overline{P}_{uFehler} =& \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (\sum_{i=1}^{n} BP^{i-1} \times I^{2(i-1)} \times \prod_{j=2}^{i} S_j^2 \\ &+ \sum_{i=1}^{n} \overline{S_{i-1}} \times P \times BP^i \times I^{2i-1} \times \prod_{j=2}^{i} S_j^2 \\ &+ \sum_{i=1}^{n} \overline{S_{i-1}} \times \overline{S_{i-2}} \times P^2 \times BP^{i+1} \times I^{2i} \times \prod_{j=2}^{i} S_j^2)\end{aligned}$$

Auch hier ist zu beachten, dass $S_0 = S_n$ und $S_{-1} = S_{n-1}$ gilt.

Generalisierung auf m-faches Ausweichen bei n Repliken mit $0 \leq m \leq n-1$ ergibt sich eine Wahrscheinlichkeit für unentdeckte Fehler von

$$\overline{P}_{uFehler} = \frac{1}{n} \times S_1^2 \times BP^2 \times I^4 \times (\sum_{i=0}^{m} P^i \times \sum_{j=1}^{n} \prod_{k=1}^{i} \overline{S_{j-k}} \times \prod_{l=2}^{j} S_l^2 \times BP^{i+j-1} \times I^{i+2j-2}).$$

Betrachtet wird erneut das Beispiel des Nutzers mit zwei Speicherkomponenten mit den Fehlerwahrscheinlichkeiten $S_1 = 1 - 2^{-24}$ und $S_2 = 1 - 2^{-48}$ sowie $I = 1 - 2^{-32}$ für das Netzwerk, $BP = 1 - 2^{-48}$ für die Bit Preservation Software und $P = 1 - 2^{-16}$ für eine mögliche Prüfsumme in Lokation b. Im Szenario der synchronen Replikation werden die Wahrscheinlichkeiten mindestens eines unentdeckten Fehlers von $6,054 \times 10^{-8}$ ohne Prüfsumme bzw. $3,073 \times 10^{-8}$ mit Prüfsumme abgeleitet, entsprechend ist eine von 16.519.103 bzw. 32.537.142 Operationen fehlerhaft. Bei einer asynchronen Replikation werden die Wahrscheinlichkeiten eines unentdeckten Fehlers mit $1,204 \times 10^{-7}$ ohne Prüfsumme bzw. $9,057 \times 10^{-8}$ abgeschätzt, entsprechend ist eine von 8.307.480 bzw. 11.040.990 Operationen fehlerhaft. Werden auch in diesen Szenarien Dateien der Größe vier Gigabyte übertragen, ist im Schnitt eine von sechs (synchrone Replikation ohne Prüfsumme), elf (synchrone Replikation mit Prüfsumme), drei (asynchrone Replikation ohne Prüfsumme) bzw. vier (asynchrone Replikation mit Prüfsumme) dem Nutzer zur Verfügung gestellten Dateien fehlerhaft.

4.6 Erkenntnisse

Durch die Wahl der Wahrscheinlichkeit unentdeckter Fehler als Metrik für die Zuverlässigkeit von Architekturen zur Bit Preservation ist eine Modellierung der Architekturen möglich. Für die Formulierung von geschlossenen Ausdrücken für die Wahrscheinlichkeit unentdeckter Fehler muss zum einen zwischen System- und Nutzersicht und zum anderen zwischen synchroner Replikation, asynchroner Replikation mit Masterdatei sowie asynchroner Replikation ohne Masterdatei unterschieden werden, so dass insgesamt sechs verschiedene Replikationsszenarien untersucht werden. Die Betrachtung einer Prüfsumme in der Bit Preservation Software oder direkt beim Nutzer erweitert die entwickelten Ausdrücke.

5 Quantifizierung der Wahrscheinlichkeit unentdeckter Fehler

Auf Basis der in Kapitel 4 entwickelten Modelle wird in den folgenden Abschnitten die Wahrscheinlichkeit für unentdeckte Fehler mit konkreten Parametern für verschiedene Anwendungsfälle berechnet. Der Ausdruck „Wahrscheinlichkeit eines unentdeckten Fehlers“ steht in den Berechnungen stellvertretend für die Wahrscheinlichkeit, dass mindestens ein unentdeckter Fehler auftritt. Insbesondere wird, wie in den Modellannahmen formuliert, eine zeitliche Unabhängigkeit der Fehlerwahrscheinlichkeiten vorausgesetzt. Zusätzlich sind die folgenden Berechnungen zunächst unabhängig von der Größe der einzelnen Dateien. Die den Tabellen und Grafiken zu Grunde liegenden Zahlen können dem Anhang entnommen werden.

Bei der quantitativen Untersuchung der verschiedenen Replikationsszenarien werden die folgenden Grundannahmen getroffen:

- Die Fehlerwahrscheinlichkeiten der Speicherkomponenten liegen im Bereich $[2^{-16}, 2^{-64}]$.
- Die Fehlerwahrscheinlichkeit der Bit Preservation Software wird mit 2^{-64} abgeschätzt. Für die Netzwerkverbindungen gilt die Fehlerwahrscheinlichkeit des CRC-32 von 2^{-32}.
- Die maximale Anzahl der Replikationen ist in den meisten Szenarien auf fünf beschränkt. In realen Situationen werden auf Grund der steigenden Kosten oftmals nur wenige Repliken erzeugt.

In realen Systemen können die Fehlerwahrscheinlichkeiten einzelner Komponenten von den gewählten Werten abweichen. Je nach Komplexität des als Bit Preservation Software eingesetzten Systems wird die Wahrscheinlichkeit eines unentdeckten Fehlers deutlich variieren. Bei der vorliegenden Untersuchung erlaubt die gewählte Fehlerwahrscheinlichkeit auf Grund der geringen Auswirkung auf die Gesamtfehlerwahrscheinlichkeit eine Fokussierung auf den Einfluss der verwendeten Speicherkomponente innerhalb der Replikationsszenarien. Auch die Ausgestaltung der Netzwerkkomponente wie beispielsweise die Wahl des Transferprotokolls beeinflusst die Wahrscheinlichkeit eines unentdeckten Fehlers. Die zusätzlichen Algorithmen weisen jedoch meist eine deutlich geringere Fehlerwahrscheinlichkeit als der verwendete CRC-32 auf, so dass die Abschätzung eine realistische Näherung bietet. Im Allgemeinen

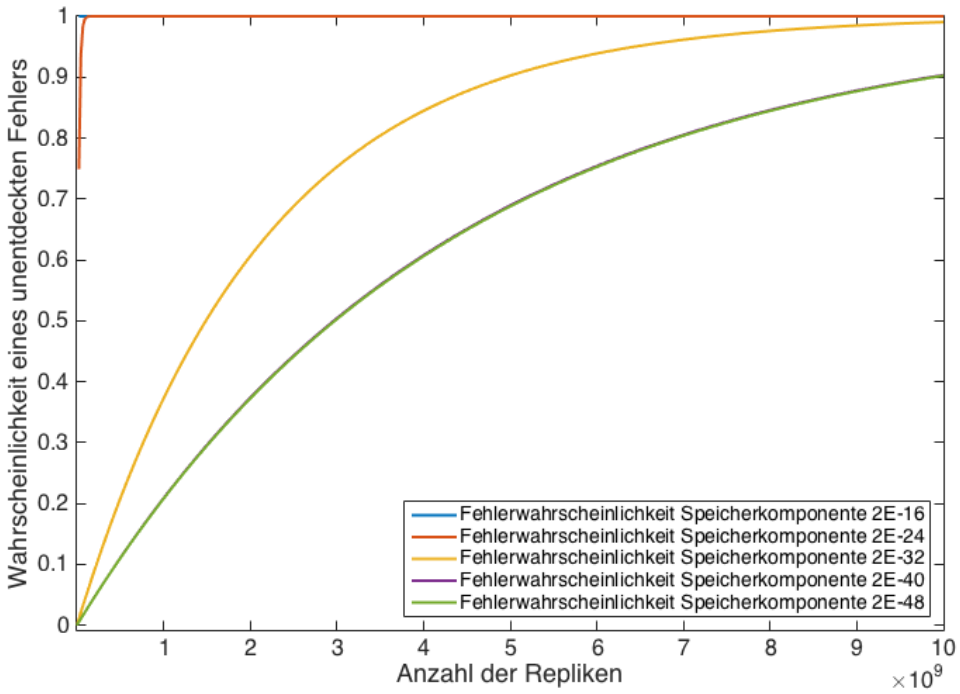

Abbildung 5.1: Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation ohne Prüfsumme in Abhängigkeit der Fehlerwahrscheinlichkeit der Speicherkomponenten und der Anzahl Repliken.

können alternative Fehlerwahrscheinlichkeiten, die beispielsweise durch eine vorangegangene Quantifizierung bestimmt wurden und von den getroffenen Annahmen abweichen, mit Hilfe der gleichen Methodik untersucht werden.

5.1 Szenario 1: Synchrone Replikation aus Systemsicht

Betrachtet wird zunächst das Szenario der synchronen Replikation aus Systemsicht ohne Prüfsumme. Abbildung 5.1 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers in Abhängigkeit von der Fehlerwahrscheinlichkeit der verwendeten Speicherkomponenten und der Anzahl der Repliken. Zur Untersuchung der Plausibilität der entwickelten Ausdrücke wurde eine sehr große Anzahl an Repliken gewählt, die in der Realität nicht auftreten wird. Mögliche Fehlerwahrscheinlichkeiten werden aus der Menge

$$\{2^{-16}, 2^{-24}, 2^{-32}, 2^{-40}, 2^{-48}, 2^{-56}, 2^{-64}\}$$

gewählt, wobei innerhalb der Fälle für alle Speicherkomponenten die gleiche Fehlerwahrscheinlichkeit angenommen wird. Die folgenden Erkenntnisse sind beobachtbar:

- Die Wahrscheinlichkeit eines unentdeckten Fehlers nähert sich in allen Szenarien für eine steigende Anzahl an Repliken eins an. Durch eine steigende Anzahl von Repliken wird das System komplexer und es kommen zusätzliche mögliche Fehlerquellen hinzu.
- Die Fehlerwahrscheinlichkeit der Speicherkomponenten beeinflusst die Steigung der zugehörigen Kurve: je größer die Fehlerwahrscheinlichkeit, desto schneller erfolgt die Annäherung.
- Für Fehlerwahrscheinlichkeiten von $2^{-40}, 2^{-48}, 2^{-56}$ und 2^{-64} der Speicherkomponenten unterscheidet sich die Gesamtfehlerwahrscheinlichkeit auch für eine hohe Anzahl von Repliken nur minimal. In diesen Szenarien ist die höhere Fehlerwahrscheinlichkeit des Netzwerks der bestimmende Faktor.

Die Verwendung einer Prüfsumme in Lokation b verspricht eine Verbesserung der Gesamtfehlerwahrscheinlichkeit durch Einführung einer Ausweichmöglichkeit auf alternative Speicherkomponenten. Die Tabellen 5.1, 5.2 und 5.3 zeigen die Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit der verwendeten Prüfsumme von 2^{-16} und der Speicherkomponenten von 2^{-16}, 2^{-32} bzw. 2^{-48}. Betrachtet werden maximal fünf Repliken und resultierend vier Ausweichschritte. Als Beobachtungen lassen sich zusammenfassen:

- Die Wahrscheinlichkeit eines unentdeckten Fehlers steigt in diesen Szenarien erwartungsgemäß mit einer steigenden Anzahl von Repliken.
- Wird die maximale Anzahl an Ausweichschritten in den verschiedenen Szenarien erhöht, verringert sich die Fehlerwahrscheinlichkeit.
- Die geringste Fehlerwahrscheinlichkeit wird durch Ausnutzung der maximalen Anzahl an Ausweichschritten ermöglicht.
- Die größte Verbesserung wird durch den ersten Ausweichschritt erreicht, weitere Ausweichschritte weisen immer geringere Verbesserungen auf. Auffällig ist jedoch die gleiche Größenordnung der Verbesserung der Fehlerwahrscheinlichkeit bei jedem zusätzlichen Ausweichschritt unabhängig von der Anzahl der Repliken.

	A0 [10^{-5}]	A1 [10^{-5}]	A2 [10^{-10}]	A3 [10^{-15}]	A4 [10^{-20}]
R1	3.051828	-	-	-	-
R2	4.577683	-1.525785774	-	-	-
R3	6.103516	-1.525762492	-2.328093278	-	-
R4	7.629325	-1.525739210	-2.328057753	-3.552279998	-
R5	9.155110	-1.525715929	-2.328022229	-3.552225788	-5.421010862

Tabelle 5.1: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation, einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328270905	-	-	-
R3	2.328306	-2.328270903	-5.420845412	-	-
R4	2.793968	-2.328270902	-5.420845410	-1.262119667	-
R5	3.259629	-2.328270901	-5.420845407	-1.262119666	-2.938556503

Tabelle 5.2: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-15}]	A2 [10^{-30}]	A3 [10^{-44}]	A4 [10^{-58}]
R1	0.931330	-	-	-	-
R2	1.164164	-3.552659464	-	-	-
R3	1.396998	-3.552659463	-1.262138928	-	-
R4	1.629832	-3.552659462	-1.262138927	-4.483949810	-
R5	1.862666	-3.552659461	-1.262138927	-4.483949809	-1.592994674

Tabelle 5.3: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation, einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

- Bei zusätzlichen Repliken sinkt die Verbesserung bei einer gleichbleibenden Anzahl der Ausweichschritte nur leicht. Maßgeblicher Faktor der Wahrscheinlichkeit eines unentdeckten Fehlers ist in allen Szenarien die genutzte Anzahl der Repliken bei gleichbleibender Fehlerwahrscheinlichkeit der Speicherkomponenten.

Die Tabellen 5.4 und 5.5 zeigen die Auswirkungen der Verwendung einer besseren Prüfsumme für die betrachteten Fälle aus Tabelle 5.2. Allgemein bleiben die zuvor beobachteten Zusammenhänge bei Variation der Fehlerwahrscheinlichkeit der Prüfsumme erhalten. Erwartungsgemäß kann durch eine Prüfsumme mit geringerer Fehlerwahrscheinlichkeit eine größere Verbesserung der Gesamtfehlerwahrscheinlichkeit erreicht werden.

Zusätzlich ist es möglich, für alle untersuchten Fälle aus Tabelle 5.2 eine zusätzliche Prüfsumme in Lokation a einzuführen. Die beobachteten Zusammenhänge lassen sich direkt auf das erweiterte Szenario übertragen, da die Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers auf Grund der Konstruktion einer Prüfsummenberechnung in Lokation a unabhängig von der Wahrscheinlichkeit innerhalb des Replikationsszenarios ist. Die Auswirkungen lassen sich daher durch einen konstanten Faktor beschreiben, der genau der Fehlerwahrscheinlichkeit der verwendeten Prüfsumme entspricht.

5.2 Vergleich der Replikationsszenarien aus Systemsicht

Betrachtet werden die drei in den Kapiteln 4.4.1, 4.4.2 und 4.4.3 vorgestellten Replikationsszenarien aus Systemsicht. Wird keine Prüfsumme innerhalb des Systems berechnet und die Fehlerwahrscheinlichkeit der Speicherkomponenten als konstant angenommen, entsprechen sich die asynchronen Replikationsszenarien. Es genügt daher die Betrachtung der asynchronen Replikation o.B.d.A. mit Masterdatei.

Tabelle 5.6 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation und asynchroner Replikation mit Masterdatei jeweils ohne Prüfsumme bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-16} für maximal fünf Repliken. Die aufgeführten Werte sind zur besseren Vergleichbarkeit mittels der Wahrscheinlichkeit bei einer Replik normiert. Die folgenden Erkenntnisse sind beobachtbar:

- Erwartungsgemäß steigt in beiden Szenarien die Wahrscheinlichkeit eines unentdeckten Fehlers bei einer steigenden Anzahl von Repliken.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306431	-	-	-
R3	2.328306	-2.328306430	-5.421010844	-	-
R4	2.793968	-2.328306429	-5.421010842	-1.262177443	-
R5	3.259629	-2.328306428	-5.421010840	-1.262177442	-2.938735862

Tabelle 5.4: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ...,R5 und maximalen **A**usweichschritte A0, ...,A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306432	-	-	-
R3	2.328306	-2.328306431	-5.421010847	-	-
R4	2.793968	-2.328306429	-5.421010845	-1.262177444	-
R5	3.259629	-2.328306428	-5.421010842	-1.262177443	-2.938735865

Tabelle 5.5: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner Replikation, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ...,R5 und maximalen **A**usweichschritte A0, ...,A4.

# Repliken	Szenario 1	Szenario 2
1	1	1
2	1.499981	1.999954
3	1.999954	2.999878
4	2.499920	3.999771
5	2.999878	4.999634

Tabelle 5.6: Normierter Vergleich der synchronen Replikation sowie der asynchronen Replikation aus Systemsicht in Abhängigkeit der Anzahl Repliken bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-16}.

- Die Fehlerwahrscheinlichkeit bei synchroner Replikation steigt im betrachteten Intervall jedoch langsamer als die Fehlerwahrscheinlichkeit der asynchronen Replikation. Grund ist der geringere Komplexitätszuwachses eines Systems mit synchroner Replikation bei Hinzunahme einer weiteren Speicherkomponente im Vergleich zu einem System mit asynchroner Replikation.

Das gleiche Verhalten zeigt sich, wie aus Tabelle 5.2 ersichtlich, für alternative Fehlerwahrscheinlichkeiten der Speicherkomponente. Auffällig ist, dass die Wahrscheinlichkeit eines unentdeckten Fehlers für n Repliken in Szenario 2 in der Größenordnung der Wahrscheinlichkeit für $2n-1$ Repliken in Szenario 1 entsprechen. Die exakten Werte unterscheiden sich jedoch leicht.

Erweitert man das Intervall der betrachteten Repliken auf $[0, 50000]$ wie in Abbildung 5.3, lassen sich die folgenden Beobachtungen formulieren:

- Sowohl die Wahrscheinlichkeit bei synchroner Replikation als auch bei asynchroner Replikation nähern sich erwartungsgemäß 1 an.
- Der beobachtete langsamere Anstieg bei synchroner Replikation gilt ebenfalls bei der erhöhten Anzahl der Repliken.

Allgemein lässt sich für die Wahrscheinlichkeit eines unentdeckten Fehlers der folgende Zusammenhang zwischen synchroner und asynchroner Replikation aus Systemsicht herstellen:

1. Asynchrone Replikation mit Masterdatei:

$$\overline{P}_{uFehler} = \overline{P}_{uFehler}\{\text{synchron}\} \times S_1^{n-1} \times BP^{n-1} \times I^{n-1}$$

2. Asynchrone Replikation ohne Masterdatei:

$$\overline{P}_{uFehler} = \overline{P}_{uFehler}\{\text{synchron}\} \times \prod_{i=2}^{n} S_{i-1} \times BP^{n-1} \times I^{n-1}$$

5.3 Szenario 5+6: Asynchrone Replikation aus Nutzersicht

Die Wahrscheinlichkeit unentdeckter Fehler bei synchroner Replikation aus Nutzersicht ist bei einer konstanten Fehlerwahrscheinlichkeit der Speicherkomponenten von der Anzahl der Repliken unabhängig. Untersucht werden daher die Fälle der asynchronen Replikation aus Kapitel 4.5.2 und 4.5.3.

# Repl.	Szenario 1	Szenario 2
1	1	1
2	1.333333	1.666667
3	1.666667	2.333333
4	2.000000	3.000000
5	2.333333	3.666667

# Repl.	Szenario 1	Szenario 2
1	1	1
2	1.250002	1.500004
3	1.500004	2.000008
4	1.750006	2.500011
5	2.000008	3.000015

Abbildung 5.2: Normierter Vergleich der synchronen Replikation sowie der asynchronen Replikation aus Systemsicht in Abhängigkeit der Anzahl Repliken bei einer konstanten Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-32} (links) bzw. 2^{-48} (rechts).

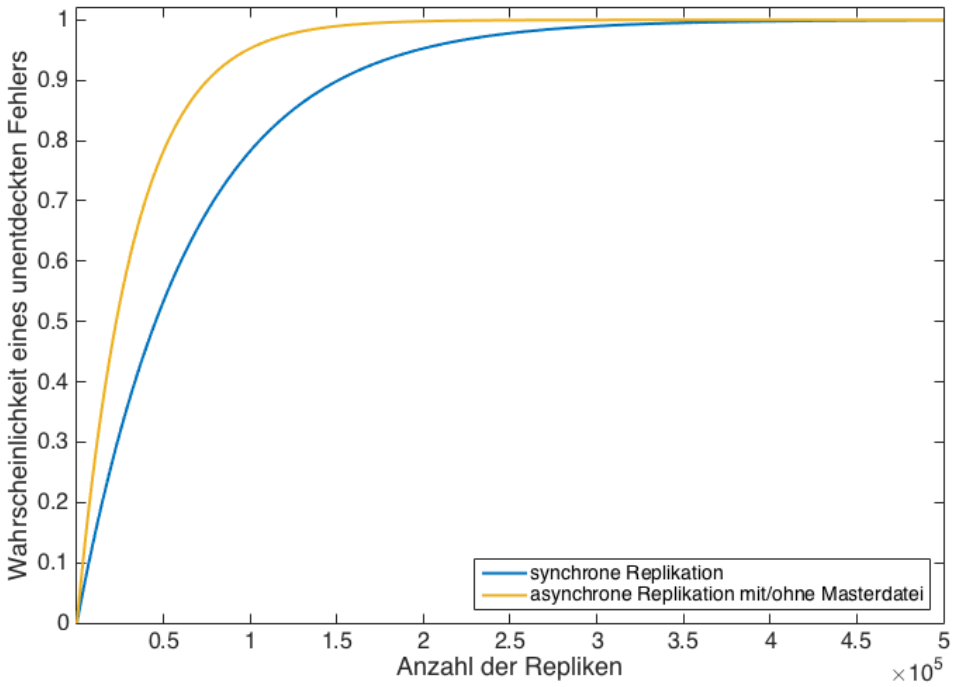

Abbildung 5.3: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit der Speicherkomponente von 2^{-16} in Abhängigkeit der Wahl des Replikationsszenarios und der Anzahl Repliken.

Abbildung 5.4 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation in Abhängigkeit der Anzahl der Repliken und der Fehlerwahrscheinlichkeit der verwendeten Speicherkomponenten.

Die folgenden Zusammenhänge sind im Szenario der Replikation ohne Masterdatei beobachtbar:

- Die Fehlerwahrscheinlichkeit nähert sich 1 an. Jede zusätzliche Replik weist auf Grund der Konstruktion des Replikationsszenarios eine höhere Fehlerwahrscheinlichkeit als die vorherigen auf und erhöht auf diese Weise die Gesamtfehlerwahrscheinlichkeit bei einer gleichmäßigen Verteilung der Leseoperationen auf die Speicherkomponenten.

- Eine geringere Fehlerwahrscheinlichkeit der Speicherkomponenten führt zu einer geringeren Steigerung der Fehlerwahrscheinlichkeiten.

- Analog zu den Untersuchungen in Kapitel 5.1 führt eine sehr geringe Fehlerwahrscheinlichkeit der Speicherkomponenten, beispielsweise 2^{-56} oder 2^{-64}, nur zu sehr geringen Verbesserungen der Gesamtfehlerwahrscheinlichkeit.

Die asynchrone Replikation mit Masterdatei weist einen deutlich anderen Verlauf auf. Die Gesamtwahrscheinlichkeit eines unentdeckten Fehlers nähert sich für Fehlerwahrscheinlichkeiten der Speicherkomponente aus der Menge $\{2^{-40}, 2^{-48}, 2^{-56}, 2^{-64}\}$ wie abgebildet einem Wert von $\sim 1,4 \times 10^{-9}$ an. Bei höheren Fehlerwahrscheinlichkeiten der Speicherkomponenten liegen die Werte ebenfalls höher. Grund ist, dass bei gleicher Fehlerwahrscheinlichkeit der Speicherkomponenten die Wahrscheinlichkeit eines unentdeckten Fehlers bei Lese- und Schreiboperation für alle Speicherkomponenten mit Ausnahme von Speicherkomponente 1 identisch ist. Bei einer Vielzahl von Repliken wird es immer unwahrscheinlicher, dass aus Speicherkomponente 1 gelesen wird und die Gesamtfehlerwahrscheinlichkeit konvergiert gegen $S^4 \times BP^3 \times I^6$. Diese Wahrscheinlichkeit entspricht einer Schreib- und Leseoperation in einer beliebigen Speicherkomponente $\neq 1$. Tabelle 5.7 zeigt eine Übersicht der maximalen Wahrscheinlichkeiten für einen unentdeckten bei asynchroner Replikation mit Masterdatei für die unterschiedlichen Fehlerwahrscheinlichkeiten der Speicherkomponenten.

Die Einführung einer Prüfsumme in Lokation b verspricht auch für die Betrachtungen aus Nutzersicht eine Verringerung der Wahrscheinlichkeit eines unentdeckten Fehlers. In den Tabellen 5.8, 5.9 und 5.10 sind die Veränderungen der Gesamtfehlerwahrscheinlichkeiten bei asynchroner Replikation mit

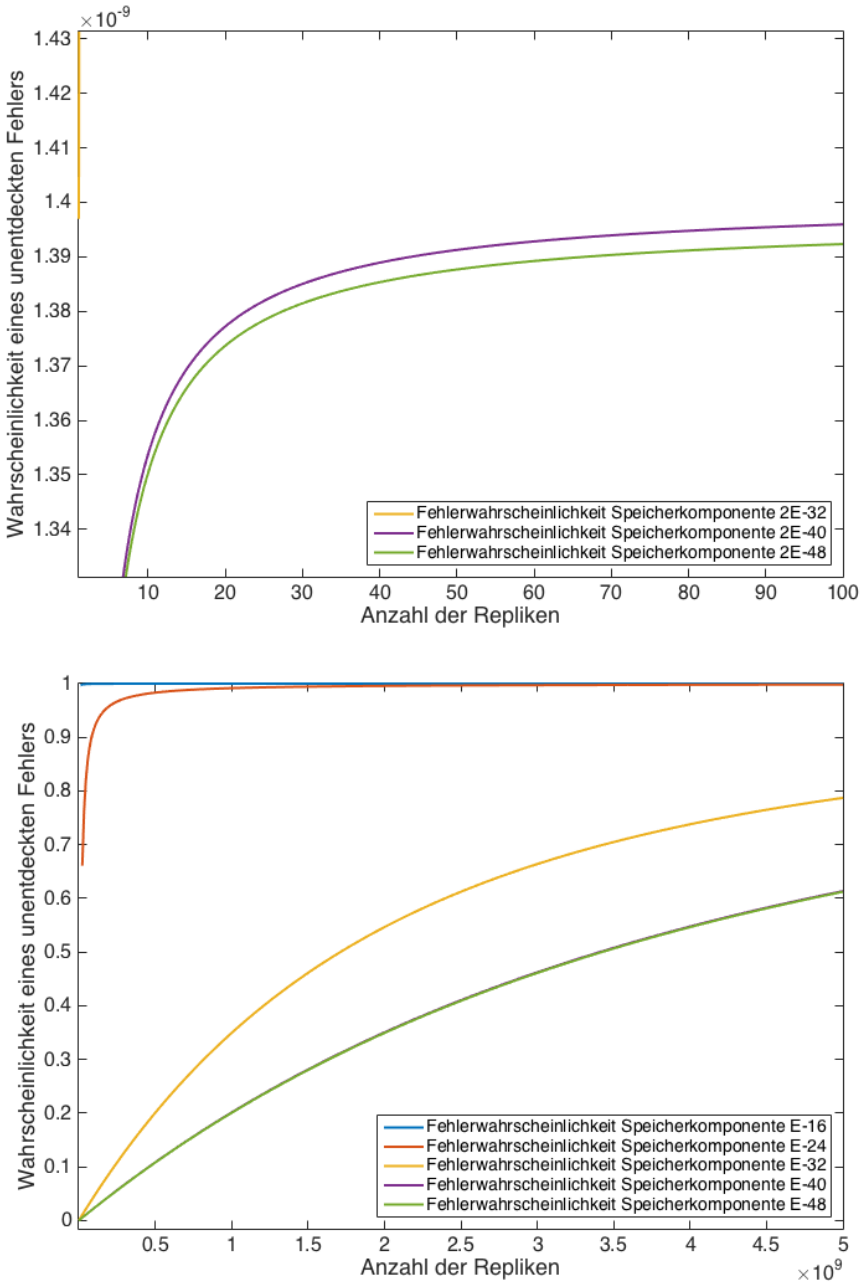

Abbildung 5.4: Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei (oben) und ohne Masterdatei (unten) ohne Prüfsumme aus Nutzersicht in Abhängigkeit der Anzahl Repliken sowie der Fehlerwahrscheinlichkeit der Speicherkomponenten (man beachte die unterschiedlichen Skalen der Darstellung).

S_i	$\overline{P}_{uFehler}$
2^{-16}	6.103516×10^{-5}
2^{-24}	2.398155×10^{-7}
2^{-32}	2.328306×10^{-9}
2^{-40}	1.400621×10^{-9}
2^{-48}	1.396998×10^{-9}
2^{-56}	1.396984×10^{-9}
2^{-64}	1.396984×10^{-9}

Tabelle 5.7: Maximale Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei in Abhängigkeit der Fehlerwahrscheinlichkeit der Speicherkomponenten.

	A0 [10^{-5}]	A1 [10^{-5}]	A2 [10^{-10}]	A3 [10^{-15}]	A4 [10^{-20}]
R1	3.051828	-	-	-	-
R2	4.577672	-1.525785774	-	-	-
R3	5.086286	-1.525778013	-2.328116961	-	-
R4	5.340594	-1.525774133	-2.328111040	-3.552361313	-
R5	5.493178	-1.525771805	-2.328107488	-3.552354537	-5.421010862

Tabelle 5.8: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328270905	-	-	-
R3	2.017866	-2.328270904	-5.420845414	-	-
R4	2.095476	-2.328270904	-5.420845414	-1.262119668	-
R5	2.142042	-2.328270904	-5.420845413	-1.262119668	-2.938556506

Tabelle 5.9: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-15}]	A2 [10^{-29}]	A3 [10^{-44}]	A4 [10^{-58}]
R1	0.931330	-	-	-	-
R2	1.164164	-3.552659464	-	-	-
R3	1.241775	-3.552659463	-1.262138928	-	-
R4	1.280581	-3.552659463	-1.262138928	-4.483949812	-
R5	1.303864	-3.552659463	-1.262138928	-4.483949812	-1.592994675

Tabelle 5.10: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

Masterdatei für Fehlerwahrscheinlichkeiten der Speicherkomponenten von 2^{-16}, 2^{-32} bzw. 2^{-48} und Verwendung einer Prüfsumme mit Fehlerwahrscheinlichkeit 2^{-16} aufgeführt.

Die bereits in Kapitel 5.1 beschriebenen Zusammenhänge lassen sich ebenfalls in diesem Szenario beobachten:

- Unabhängig von der Fehlerwahrscheinlichkeit der Speicherkomponenten erhöht sich die Wahrscheinlichkeit eines unentdeckten Fehlers bei einer steigenden Anzahl von Repliken.
- Jeder Ausweichschritt verringert die Fehlerwahrscheinlichkeit, wobei mit dem ersten Ausweichschritt die größte Verbesserung erzielt werden kann.
- Die beste Fehlerwahrscheinlichkeit für eine feste Anzahl an Repliken ist mit der maximalen Anzahl an Ausweichschritten zu erreichen.
- Für die betrachtete geringe Anzahl an Repliken ist bei asynchroner Replikation mit Masterdatei die Wahrscheinlichkeit eines unentdeckten Fehlers noch beeinflusst von der Fehlerwahrscheinlichkeit der ersten Speicherkomponente und der Verlauf ähnlich zur synchronen Replikation. Auch die absoluten Werte der Verbesserung entsprechen sich.

Die Tabellen 5.11 und 5.12 zeigen die Auswirkungen der Verwendung einer besseren Prüfsumme mit Fehlerwahrscheinlichkeit 2^{-32} bzw. 2^{-48} für die untersuchten Fälle aus Tabelle 5.9. Insgesamt ist der Zusammenhang erkennbar, dass eine geringere Fehlerwahrscheinlichkeit der Prüfsumme auch eine geringere Wahrscheinlichkeit eines unentdeckten Fehlers für den Nutzer zur Folge hat. Der Unterschied der Verwendung einer Prüfsumme mit Fehlerwahrscheinlichkeit 2^{-32} oder 2^{-48} ist nur minimal.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306431	-	-	-
R3	2.017866	-2.328306431	-5.421010846	-	-
R4	2.095476	-2.328306431	-5.421010846	-1.262177444	-
R5	2.142042	-2.328306430	-5.421010846	-1.262177444	-2.938735865

Tabelle 5.11: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306432	-	-	-
R3	2.017866	-2.328306431	-5.421010849	-	-
R4	2.095476	-2.328306431	-5.421010849	-1.262177445	-
R5	2.142042	-2.328306431	-5.421010848	-1.262177445	-2.938735868

Tabelle 5.12: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation mit Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

Im Folgenden wird das Szenario der asynchronen Replikation ohne Masterdatei betrachtet. Die Tabellen 5.13, 5.14 und 5.15 zeigen die Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers für Fehlerwahrscheinlichkeiten der Speicherkomponenten von 2^{-16}, 2^{-32} bzw. 2^{-64}. In diesem Szenario lässt sich beobachten:

- Wie bei der asynchronen Replikation mit Masterdatei gelten die Zusammenhänge aus Kapitel 5.1.
- Im Vergleich zur asynchronen Replikation mit Masterdatei nehmen die Verbesserungen des gleichen Ausweichschrittes bei einer steigenden Anzahl von Repliken deutlicher ab.
- Auch bei Verwendung einer besseren Prüfsumme wie in Tabelle 5.16 oder 5.17 lässt sich wie bei der synchronen Replikation und asynchronen Replikation mit Masterdatei eine Verringerung der Wahrscheinlichkeit eines unentdeckten Fehlers erreichen.

	A0 [10^{-5}]	A1 [10^{-5}]	A2 [10^{-10}]	A3 [10^{-15}]	A4 [10^{-20}]
R1	3.051828	-	-	-	-
R2	4.577672	-1.525785774	-	-	-
R3	6.103485	-1.525762492	-2.328093278	-	-
R4	7.629266	-1.525739211	-2.328057754	-3.552280014	-
R5	9.155017	-1.525715930	-2.328022231	-3.552225812	-5.420183729

Tabelle 5.13: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation ohne Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328270904	-	-	-
R3	2.328306	-2.328270903	-5.420845412	-	-
R4	2.793968	-2.328270902	-5.420845410	-1.262119667	-
R5	3.259629	-2.328270901	-5.420845407	-1.262119666	-2.938556503

Tabelle 5.14: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation ohne Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-15}]	A2 [10^{-29}]	A3 [10^{-44}]	A4 [10^{-58}]
R1	0.931330	-	-	-	-
R2	1.164164	-3.552659464	-	-	-
R3	1.396998	-3.552659463	-1.262138928	-	-
R4	1.629832	-3.552659462	-1.262138927	-4.483949810	-
R5	1.862666	-3.552659461	-1.262138927	-4.483949809	-1.592994674

Tabelle 5.15: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation ohne Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306431	-	-	-
R3	2.328306	-2.328306430	-5.421010845	-	-
R4	2.793968	-2.328306429	-5.421010842	-1.262177443	-
R5	3.259629	-2.328306428	-5.421010840	-1.262177442	-2.938735862

Tabelle 5.16: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation ohne Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

	A0 [10^{-9}]	A1 [10^{-10}]	A2 [10^{-20}]	A3 [10^{-29}]	A4 [10^{-39}]
R1	1.396984	-	-	-	-
R2	1.862645	-2.328306432	-	-	-
R3	2.328306	-2.328306431	-5.421010847	-	-
R4	2.793968	-2.328306429	-5.421010845	-1.262177444	-
R5	3.259629	-2.328306428	-5.421010842	-1.262177443	-2.938735865

Tabelle 5.17: Veränderung der Wahrscheinlichkeit eines unentdeckten Fehlers bei asynchroner Replikation ohne Masterdatei, einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der Prüfsummen in Lokation B in Abhängigkeit der Anzahl **R**epliken R1, ..., R5 und maximalen **A**usweichschritte A0, ..., A4.

5.4 Vergleich der Replikationsszenarien aus Nutzersicht

Betrachtet werden die drei in den Kapiteln 4.5.1, 4.5.2 und 4.5.3 vorgestellten Replikationsszenarien aus Nutzersicht. Tabelle 5.18 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten 2^{-16} und ohne Prüfsumme für maximal fünf Repliken, zur besseren Vergleichbarkeit normiert mit Hilfe der Fehlerwahrscheinlichkeit bei einer Replik.

Die folgenden Zusammenhänge sind beobachtbar:

- Bei synchroner Replikation ändert sich die Wahrscheinlichkeit eines unentdeckten Fehlers für unterschiedliche Anzahlen von Repliken nicht.
- In den Szenarien asynchroner Replikation erhöht sich die Fehlerwahrscheinlichkeit, wenn die Anzahl der Repliken erhöht wird.
- Während der Zuwachs bei asynchroner Replikation mit Masterdatei mit jeder zusätzlichen Replik geringer wird, kann bei der asynchronen Replikation ohne Masterdatei ein annähernd konstanter Zuwachs beobachtet werden.

Die gleichen Zusammenhänge gelten auch bei einer geringeren Fehlerwahrscheinlichkeit der Speicherkomponenten wie in Tabelle 5.19 aufgeführt. In diesen Szenarien ist jedoch der geringere prozentuale Zuwachs der Wahrscheinlichkeit eines unentdeckten Fehlers bei zuverlässigeren Speicherkomponenten auffällig.

Tabelle 5.20 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers in den Replikationsszenarien aus Nutzersicht bei maximal fünf Repliken. Eine Prüfsumme wird zusätzlich in Lokation b berechnet. Sowohl die verwendeten Speicherkomponenten als auch die Prüfsummen weisen eine Fehlerwahrscheinlichkeit von 2^{-16} auf. Es wird jeweils von der maximalen Anzahl an Ausweichschritten ausgegangen, da so die geringste Fehlerwahrscheinlichkeit erreicht werden kann. Zur besseren Vergleichbarkeit sind die Fehlerwahrscheinlichkeiten mit Hilfe der Fehlerwahrscheinlichkeit bei einer Replik normiert. Die folgenden Zusammenhänge sind beobachtbar:

- Im Szenario der synchronen Replikation kann durch Einführung der Möglichkeit des Ausweichens mit Berechnung einer Prüfsumme erstmals eine Verringerung der Wahrscheinlichkeit eines unentdeckten Fehlers erreicht werden.

# Repliken	Szenario 4	Szenario 5	Szenario 6
1	1	1	1
2	1	1.499977	1.499977
3	1	1.666636	1.999944
4	1	1.749966	2.499901
5	1	1.799963	2.999847

Tabelle 5.18: Normierter Vergleich der synchronen Replikation sowie der asynchronen Replikation mit und ohne Masterdatei aus Nutzersicht in Abhängigkeit der Anzahl Repliken bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-16}.

# Repl.	Szenario 5	Szenario 6
1	1	1
2	1.333333	1.333333
3	1.444444	1.666667
4	1.500000	2.000000
5	1.533333	2.333333

# Repl.	Szenario 5	Szenario 6
1	1	1
2	1.250002	1.250002
3	1.333336	1.500004
4	1.375003	1.750006
5	1.400003	2.000008

Tabelle 5.19: Normierter Vergleich der asynchronen Replikation mit und ohne Masterdatei aus Nutzersicht in Abhängigkeit der Anzahl Repliken bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-32} (links) bzw. 2^{-48} (rechts).

# Repliken	Szenario 4	Szenario 5	Szenario 6
1	1	1	1
2	0.500034331	1.000019072	1.000019072
3	0.500026703	1.166673024	1.499986013
4	0.500026702	1.250003814	1.999950409
5	0.500026702	1.300002288	2.499904634

Tabelle 5.20: Normierter Vergleich der synchronen Replikation sowie der asynchronen Replikation mit und ohne Masterdatei aus Nutzersicht in Abhängigkeit der Anzahl Repliken bei einer Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-16} und der Prüfsummen von 2^{-16}.

- Durch die Hinzunahme einer zweiten Replik kann die Fehlerwahrscheinlichkeit um fast 50% gesenkt werden. Die Entwicklung der Kosten, beispielsweise linear oder exponentiell, ist abhängig von der Infrastruktur der Speicheranbieter und kann durch eine Berechnung der Gesamtbetriebskosten des spezifischen Anbieters (Total Cost of Ownership [31]) abgeschätzt werden.

- Zusätzliche Repliken führen in diesem Szenario zu einer weiteren Verbesserung, die im Vergleich zum Übergang von einer zu zwei Repliken jedoch deutlich geringer ausfällt.

- In den Szenarien der asynchronen Replikation verbessert das mögliche Ausweichen die Fehlerwahrscheinlichkeit im Vergleich zu den Szenarien ohne Prüfsumme, kann jedoch die Auswirkungen der zusätzlichen Repliken nicht kompensieren.

- Mit Ausnahme des Szenarios mit zwei Repliken, wo sich die Fehlerwahrscheinlichkeit nur sehr gering vom Vergleichsfall unterscheidet, steigt insgesamt in beiden Szenarien die Wahrscheinlichkeit eines unentdeckten Fehlers bei einer steigenden Anzahl von Repliken.

- Wie bereits vorher beobachtet, erhöht sich die Fehlerwahrscheinlichkeit bei der synchronen Replikation mit Masterdatei auch in diesen Szenarien langsamer als die Fehlerwahrscheinlichkeit bei synchroner Replikation ohne Masterdatei.

Tabelle 5.21 zeigt die Wahrscheinlichkeit eines unentdeckten Fehlers für zuverlässigere Speicherkomponenten mit einer Fehlerwahrscheinlichkeit von 2^{-32} bzw. 2^{-48}. In diesem Szenario lassen sich die folgenden Beobachtungen zusammenfassen:

- Prozentual kann bei der synchronen Replikation eine geringere Verbesserung erzielt werden.

- Bei der asynchronen Replikation mit Masterdatei kann man höhere Fehlerwahrscheinlichkeiten als in Tabelle 5.20 beobachten. Zum einen

# Repliken	Szenario 4	Szenario 5	Szenario 6
1	1	1	1
2	0.833336	1.166669	1.166669
3	0.833336	1.277780	1.500003
4	0.833336	1.333336	1.833336
5	0.833336	1.366669	2.166669

# Repliken	Szenario 4	Szenario 5	Szenario 6
1	1	1	1
2	0.999996	1.249998	1.249998
3	0.999996	1.333332	1.500000
4	0.999996	1.374999	1.750002
5	0.999996	1.399999	2.000004

Tabelle 5.21: Normierter Vergleich der synchronen Replikation sowie der asynchronen Replikation mit und ohne Masterdatei aus Nutzersicht in Abhängigkeit der Anzahl Repliken bei maximalem Ausweichen mit einer Fehlerwahrscheinlichkeit der Prüfsummen von 2^{-16} und der Speicherkomponenten von 2^{-32} (oben) bzw. 2^{-48} (unten) und der Prüfsummen von 2^{-16}.

können diese Zusammenhänge durch geringere Fehlerwahrscheinlichkeiten im Basisfall und zum anderen durch die im Vergleich zu den Speicherkomponenten geringere Zuverlässigkeit der Prüfsumme begründet werden.

- Im Szenario der asynchronen Replikation ohne Masterdatei steigt die Wahrscheinlichkeit eines unentdeckten Fehlers langsamer bei der Wahl von zuverlässigeren Speicherkomponenten auf Grund der Konstruktion des Replikationsszenarios.

5.5 Auswirkungen steigender Datenmengen

Sind die entsprechenden Fehlerwahrscheinlichkeiten pro Lese- und Schreiboperation bekannt, lässt sich die Wahrscheinlichkeit für mindestens einen Fehler bei einer Vielzahl von Operationen bestimmen. $P_{uFehler}$ bezeichne die

# Dateien	1 [10^{-9}]	10 [10^{-8}]	100 [10^{-7}]	1000 [10^{-6}]	10000 [10^{-5}]
S4 - R1	1.396984	1.396984	1.396984	1.396983	1.396974
S4 - R*	1.164157	1.164157	1.164157	1.164156	1.164150
S5 - R1	1.396984	1.396984	1.396984	1.396983	1.396974
S5 - R2	1.629818	1.629818	1.629818	1.629817	1.629805
S5 - R3	1.785038	1.785038	1.785038	1.785037	1.785023
S5 - R4	1.862649	1.862649	1.862649	1.862647	1.862631
S5 - R5	1.909215	1.909215	1.909215	1.909213	1.909197
S6 - R1	1.396984	1.396984	1.396984	1.396983	1.396974
S6 - R2	1.629818	1.629818	1.629818	1.629817	1.629805
S6 - R3	2.095479	2.095479	2.095479	2.095477	2.095457
S6 - R4	2.561141	2.561141	2.561140	2.561137	2.561108
S6 - R5	3.026802	3.026802	3.026801	3.026797	3.026756

Tabelle 5.22: Wahrscheinlichkeit eines unentdeckten Fehlers in den verschiedenen Szenarien beschrieben durch das Replikations**S**zenario S4 , S5 , S6 und die Anzahl **R**epliken R1 , ... , R5 in Abhängigkeit der Anzahl der durchgeführten Operationen (für eine erleichterte Lesbarkeit und Vergleichbarkeit sind die jeweiligen Einheiten in der Kopfzeile aufgeführt).

Wahrscheinlichkeit eines Fehlers pro Lese- und Schreiboperation in den verschiedenen Szenarien. Die Wahrscheinlichkeit P_f von genau f Operationen mit unentdeckten Fehlern bei d Gesamtoperationen lässt sich durch den Zusammenhang

$$P_f = \binom{d}{f} \times P_{uFehler}^{f} \times (1 - P_{uFehler})^{d-f}$$

beschreiben. Summation von P_f über f ergibt die Wahrscheinlichkeit von mindestens einer fehlerhaften Operation. Vereinfachend kann auch hier die Gegenwahrscheinlichkeit genutzt werden, da

$$\sum_{f=1}^{d} P_f = 1 - (1 - P_{uFehler})^{d}.$$

Dateien werden bei der Übertragung je nach Größe in unterschiedlich viele Fragmente aufgespalten, für die jeweils eine Lese- und Schreiboperation

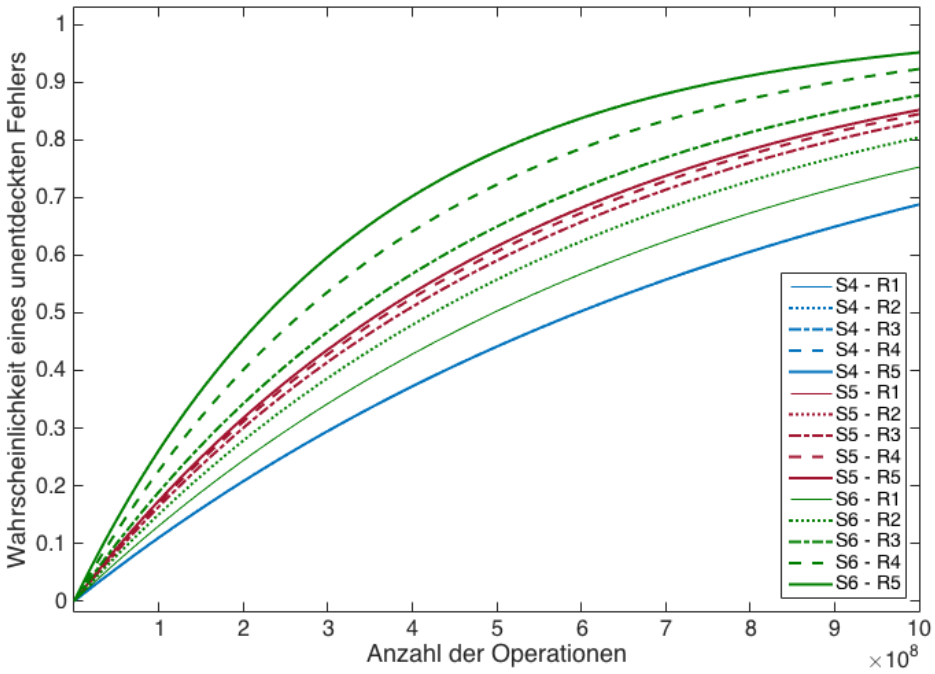

Abbildung 5.5: Wahrscheinlichkeit eines unentdeckten Fehlers bei synchroner und asynchroner Replikation aus Nutzersicht in Abhängigkeit der Anzahl der Operationen.

durchgeführt werden muss. Wie in den Beispielen in Kapitel 4.4 und Kapitel 4.5 genutzt, werden bei Übertragung per TCP/IP 1252 Datenbytes pro Operation übertragen. Die beobachteten Zusammenhänge lassen sich daher direkt auf Fälle mit einer großen Anzahl von Dateien und / oder Dateien mit hoher Dateigröße übertragen.

Beispielhaft werden in Tabelle 5.22 die Szenarien 4, 5 und 6 mit einer konstanten Fehlerwahrscheinlichkeit der Speicherkomponenten von 2^{-32} verglichen. Auch bei dieser Untersuchung sind die Szenarien auf eine realistische Anzahl von Repliken beschränkt und die maximale Anzahl an Ausweichschritten wird ausgenutzt. Auffällig ist die identische Größenordnung der Wahrscheinlichkeit eines unentdeckten Fehlers bei einer festen Anzahl Operationen für alle betrachteten Szenarien.

Abbildung 5.5 zeigt die Szenarien aus Tabelle 5.22 in Abhängigkeit der

Anzahl von Operationen, wobei das Intervall $[1, 10^9]$ betrachtet wird. Erwartungsgemäß nähert sich die Gesamtwahrscheinlichkeit bei höherer Operationsanzahl 1 an. Auf Grund der unterschiedlichen Ausgangswahrscheinlichkeiten $P_{uFehler}$ weisen die Graphen unterschiedliche Steigungen auf, zeigen jedoch einen qualitativ gleichen Verlauf.

5.6 Erkenntnisse

Das vorgestellte Modell zur Quantifizierung von Architekturen zur Bit Preservation erlaubt eine Abschätzung der Wahrscheinlichkeit unentdeckter Fehler in Abhängigkeit der verwendeten Architekturkomponenten und Maßnahmen zur Bit Preservation. Mit Hilfe des Modells ist es erstmals möglich, unterschiedliche Replikationsszenarien in Bezug auf unentdeckte Fehler zu vergleichen und die Auswirkungen steigender Datenmengen (vgl. Kapitel 1.2) oder der Verwendung unterschiedlicher Prüfsummen zu untersuchen. Die Erkenntnisse der Analyse bilden die Grundlage, basierend auf den spezifischen Eigenschaften der Daten, Empfehlungen für Maßnahmen zur Bit Preservation abzuleiten, die auf unterschiedliche Anforderungen an die Zuverlässigkeit einer langfristigen Speicherung abgestimmt sind.

6 Diskussion

Die vorliegende Dissertation ist während der Laufzeit des Projektes DARIAH entstanden und wurde durch das Umfeld eines großen, kollaborativen und europäischen Verbundes geprägt. Seit Beginn des Projektes im Jahr 2011 hat sich der Bereich (Bit) Preservation gewandelt, so dass die Möglichkeit einer dynamischen Einbindung dieser Neuentwicklungen gegeben sein sollte. Die Einbeziehung und Untersuchung der Auswirkungen von unentdeckten Fehlern erlauben nach einer Diskussion der getroffenen Modellannahmen eine Empfehlung von Strategien mit dieser Fehlerart sowie eine Auseinandersetzung mit den Auswirkungen auf die Langzeitarchivierung von Daten.

6.1 Forschung und Entwicklung im Projekt DARIAH

Sämtliche Forschungsaktivitäten innerhalb des Projektes DARIAH mussten im Spannungsfeld zwischen Geisteswissenschaft und Informatik bestehen. Interdisziplinäre Forschung eröffnet einerseits eine Vielzahl neuer und spannender Fragestellungen, muss sich aber andererseits verstärkt Rechtfertigungsfragen der Fachdisziplinen stellen und Methodiken begründen.

Bei der Spezifikation der DARIAH Storage API und der DARIAH Admin API konnte ein Konsens aller vier im deutschen Teilprojekt DARIAH-DE beteiligten Rechenzentren sowie zwei weiteren Institutionen erreicht werden. Durch die Einbeziehung aller Infrastrukturanbieter und ihrer technischen Experten wurde ein Großteil der spezifischen Anforderungen berücksichtigt und eine nachhaltige Bereitstellung unterstützt. Auf der geisteswissenschaftlichen Seite konnten durch die enge Kooperation der verschiedenen Disziplinen mit der Informatik zum einen die fachwissenschaftliche Anforderungen umgesetzt und zum anderen die Akzeptanz in den jeweiligen Fachgesellschaften verstärkt werden. Die breite Masse an Stakeholdern mit ihren unterschiedlichen Anforderungen erforderte jedoch die Einführung äußerst zeit- und arbeitsintensiver Kommunikationsprozesse, die konträr zu zügigem Konsens und der Umsetzung von Ergebnissen zu sehen sind. Die Etablierung einer gemeinsamen Verständigungsbasis war daher ebenfalls ein wichtiger Bestandteil der vorliegenden Arbeit.

DARIAH-DE ist ein wichtiger Bestandteil des europäischen Projektes DARIAH-EU und im Bereich der technischen Grundlagen im Forschungsverbund führend. Durch die zusätzliche Betrachtung von 14 europäischen Ländern mit ihren jeweiligen Voraussetzungen und Anforderungen wird der

Aufbau einer verteilten Speicherinfrastruktur erheblich komplexer und aufwändiger. Insbesondere sind Länder mit nur geringer finanzieller Förderung der Forschung im Bereich Digital Humanities nicht in der Lage, eigene Ressourcen zur Verfügung zu stellen und müssen auf Infrastrukturangebote in anderen Ländern zurückgreifen. Rechtliche Fragestellungen wie beispielsweise Lizenzierung von Daten, Möglichkeiten und Beschränkungen einer europaweiten Authentifizierung und Autorisierung oder der Zugriff auf Ressourcen sind jedoch für länderübergreifende Kollaborationen noch nicht abschließend geklärt. Im Falle einer Speicherinfrastruktur müssen neue Entwicklungen und Entscheidungen in diesen Bereichen flexibel berücksichtigt werden können. Durch Einführung von generischen und anerkannten Schnittstellen können notwendige Adaptionen mit nur minimalen Beeinträchtigungen der Nutzer durchgeführt und Ressourcen als „Data As A Service“ bereitgestellt werden.

Im Kontext von DARIAH-DE und DARIAH-EU existieren bereits mehrere Entwicklungen, die auf den Ergebnissen der Arbeit sowie den verfügbaren Schnittstellen aufbauen. Einerseits müssen diese prototypischen Anwendungen in einen nachhaltigen Betrieb überführt und andererseits die Dissemination der entwickelten Konzepte aktiv weiterbetrieben werden, um eine breite Akzeptanz in den Fachdisziplinen und eine langfristige Speicherung der Daten zu ermöglichen.

6.2 Entwicklungen im Bereich der Preservation

Seit Projektbeginn im Jahr 2011 sind zahlreiche neue Ansätze und Initiativen zur langfristigen Erhaltung von Daten entstanden. Neue Technologien wie ObjectStores [39], Schnittstellenspezifikationen wie S3 [2] oder CDMI [18] und neue Konzepte sind zu wichtigen Bestandteilen im Datenbereich geworden. Jegliche Preservationsaktivität setzt daher eine intensive Beobachtung der Technologie- und Konzeptentwicklungen und eine stetige Bewertung der Sinnhaftigkeit voraus. Auf diese Weise können Anpassungen der eigenen Preservationsstrategie vorgenommen werden, sobald sie nötig werden.

Zahlreiche europäische Infrastrukturprojekte stellen ebenfalls verschiedene Angebote im Bereich Preservation zur Verfügung. Als Beispiel einer weiteren geisteswissenschaftlichen Infrastruktur übernehmen in CLARIN [95] beteiligte CLARIN-Zentren die Aufgabe einer sichereren Datenspeicherung und unterstützen die Nutzer insbesondere im Bereich der Konvertierung linguistischer Daten. Zur Harmonisierung der verwendeten Konzepte kooperieren die beiden Projekte DARIAH und CLARIN bereits auf technischer

Ebene, um Parallelentwicklungen zu vermeiden und Möglichkeiten der Interoperabilität zu erhalten. EUDAT [53] als generische Infrastruktur bietet mit B2SAFE einen Speicherdienst, der eine langfristige Erhaltung von Dateien verspricht. Unter anderem sind persistente Identifier, Datenintegritätsüberprüfungen und eine auf iRODS [73] basierende „sichere Replikation“ Teil des Dienstes [36]. Erst in der aktuellen Förderphase werden dedizierte Dienste zur Bit Preservation inklusive der entsprechenden Schnittstellen als Arbeitsgebiet definiert und Konzepte angedacht, die Überschneidungen zu den vorgestellten Modellen dieser Dissertation aufweisen. Diese und weitere Beispiele werfen allgemein die Frage nach Interoperabilität von Forschungs- und insbesondere Speicherinfrastrukturen auf. Die langfristige Erhaltung von Forschungsdaten ist eine zentrale Herausforderung, die nicht durch Einzelentwicklungen sondern nur durch globale Strategien bewältigt werden kann.

Im Rahmen der Research Data Alliance wurde die Interessengruppe „Preservation e-Infrastructure“ mit dem Ziel gebildet, eine weltweite Einigung in Bezug auf Aktivitäten und Dienste zur Preservation zu erreichen. Die Interessengruppe ist als Kommunikationsplattform ein wichtiger erster Schritt für die Akzeptanz zukünftiger Ergebnisse in den einzelnen Fachdisziplinen, konkrete Aktivitäten müssen jedoch noch verstärkt durchgeführt werden. (Bit) Preservation ist eins der Themengebiete, das durch Erkenntnisse und eine enge Verschränkung verschiedenster Fachdisziplinen enorm profitieren kann. Synergetische Effekte und die Übertragung von Konzepten unterstützen maßgeblich die Bereitstellung von Lösungen zur langfristigen Erhaltung von Forschungsdaten.

6.3 Diskussion der Modellannahmen

Zur Modellierung von Architekturen zur Bit Preservation und zur Bestimmung der Wahrscheinlichkeit unentdeckter Fehler wurden vereinfachende Annahmen getroffen, die die Wirklichkeit nicht exakt abbilden. Durch Erweiterungen des Modells kann eine bessere Abbildung realer Zustände erreicht werden.

Stochastische Unabhängigkeit der Ereignisse

Eine der zentralen Voraussetzungen zur vorgestellten Bestimmung der Wahrscheinlichkeit unentdeckter Fehler ist die stochastische Unabhängigkeit der Fehlerereignisse, bei der allgemeinen Fehlerbetrachtung ist Unabhängigkeit jedoch nicht gegeben. Fällt beispielsweise eine Speicherkomponente aus, muss

von einer erhöhten Ausfallwahrscheinlichkeit der Speicherkomponenten vom identischen Hersteller und/oder mit identischer Software ausgegangen werden. Der beschriebene Zusammenhang lässt sich direkt auf die Untersuchung unentdeckter Fehler übertragen, die stochastische Unabhängigkeit ist daher nicht unbedingt gegeben. Wird zusätzlich das gleiche Verfahren zur Prüfsummenberechnung in den verschiedenen Architekturkomponenten verwendet, widerspricht das Vorgehen ebenfalls einer stochastischen Unabhängigkeit. Durch die Einführung von bedingten Wahrscheinlichkeiten lässt sich das vorgestellte Modell erweitern. Zu diesem Zweck muss zunächst ein Vorgehen für die Quantifizierung der bedingten Wahrscheinlichkeiten entwickelt und durchgeführt werden. Insgesamt wird die berechnete Wahrscheinlichkeit eines unentdeckten Fehlers etwas niedriger als der reale Wert sein, es sind jedoch vergleichbare Trends zu erwarten.

Wahrscheinlichkeit unentdeckter Fehler bei Schreib- und Leseoperation annähernd identisch
Bei Speicherkomponenten stellt die Schreiboperation eine beachtliche Fehlerquelle dar. Dateien können beispielsweise unvollständig oder in falschen Sektoren gespeichert werden und auf diese Weise bereits gesicherte Dateien überschreiben oder beschädigen. Insbesondere die letztgenannte Fehlerquelle führt zu einer erhöhten Wahrscheinlichkeit unentdeckter Fehler bei der Schreiboperation im Vergleich zur Leseoperation, im Allgemeinen kann daher nicht von annähernd identischen Fehlerwahrscheinlichkeiten ausgegangen werden. Durch Erweiterung des Modells und Einführung unterschiedlicher Wahrscheinlichkeiten für die Schreib- und Leseoperation der Speicherkomponenten, $S_i|$Schreiben bzw. $S_i|$Lesen, sowie der Bit Preservation Software, $BP|$Schreiben bzw. $BP|$Lesen, lässt sich der Zusammenhang abbilden.

Zeitliche Konstanz der Fehlerwahrscheinlichkeit
In der Realität lässt sich bei den identifizierten Komponenten einer Architektur zur Bit Preservation eine zeitliche Abhängigkeit bei der Wahrscheinlichkeit eines unentdeckten Fehlers beobachten. Festplatten weisen beispielsweise zu Beginn und Ende ihrer Nutzungszeit eine erhöhte Fehlerwahrscheinlichkeit auf, die auch für die Wahrscheinlichkeit von unentdeckten Fehlern anzunehmen ist und auf die gesamte Speicherkomponente übertragen werden kann. Der Effekt lässt sich durch die Einführung einer variablen Wahrscheinlichkeit in Abhängigkeit der Zeit, hier $S_i(t)$, abbilden. Auch für die Bit Preservation Software $BP(t)$, die Netzwerkverbindung $I(t)$ und die Prüfsummenberech-

nung $P(t)$ müssen auf diese Weise zeitliche Veränderungen modelliert werden. Die Verwendung von konstanten Fehlerwahrscheinlichkeiten für die einzelnen Komponenten ist eine Vereinfachung, erlaubt jedoch durch Wahl einer geeigneten Fehlerwahrscheinlichkeit eine Abschätzung zu treffen.

6.4 Empfehlungen für den Umgang mit unentdeckten Fehlern

Zur Entwicklung einer effektiven Strategie für den Umgang mit unentdeckten Fehlern müssen zunächst die Daten und sowie die vorhandenen Möglichkeiten zur Bit Preservation mit Hilfe verschiedener Kategorien bewertet werden.

Anfälligkeit des Datenformats
Datenformate zeigen eine unterschiedliche Anfälligkeit gegenüber (unentdeckten) Fehlern. Während ein Fehler in einer unkomprimierten Bilddatei möglicherweise nur eine leichte Farbveränderung eines Pixels zur Folge hat, kann ein Fehler in einem entropiekodierten Bild zu großen Verfälschungen führen. In einer Archivdatei kann ein einzelner Fehler bereits eine vollständige Unlesbarkeit zur Folge haben. In einer Untersuchung des CERN [66] wurde ermittelt, dass bei gepackten Dateien 99,8% aller einzelnen Bitfehler bereits zur Nichtlesbarkeit der Datei führen. Je anfälliger die zu speichernden Dateien sind, desto mehr Aufwand sollte auch zur Vermeidung unentdeckter Fehler eingesetzt werden.

Wert der Daten
Der Wert der zu speichernden Daten ist in vielen Fällen subjektiv und nur schwer zu bestimmen. Anhaltspunkte können der Aufwand, der für die Datenerzeugung eingesetzt wurde, oder die Anzahl der Wissenschaftler, die die Daten nutzen, sein. Einzigartigen Daten beispielsweise aus nicht reproduzierbaren Experimenten oder Digitalisate zerstörter Handschriften wird ein hoher Wert zugeordnet. Je wertvoller die zu speichernden Daten sind, desto mehr Aufwand sollte auch zur Vermeidung unentdeckter Fehler eingesetzt werden.

Verfügbare Ressourcen
Die maximale Anzahl von Repliken und damit verbunden die maximale Anzahl von Ausweichschritten, die Häufigkeit der durchgeführten Prüfsummen-

überprüfungen und die Komplexität der Prüfsumme sind von den verfügbaren Ressourcen beeinflusst. Jegliche Nutzung von sowohl Speicher- als auch Rechenressourcen sind mit Kosten verbunden, die für die Wahl einer geeigneten Strategie berücksichtigt werden müssen. Je mehr Ressourcen verfügbar sind, desto mehr Aufwand kann für die Vermeidung unentdeckter Fehler eingesetzt werden.

Anzahl der Dateien
Wie in Abbildung 5.5 zu erkennen, hat die Anzahl der Operationen bzw. der Dateien einen erheblichen Einfluss auf die Wahrscheinlichkeit eines unentdeckten Fehlers. Auch kleine Unterschiede der Fehlerwahrscheinlichkeit bei wenigen Dateien in unterschiedlichen Szenarien können für eine große Anzahl von Dateien zu deutlich unterschiedlichen Fehlerwahrscheinlichkeiten führen. Je mehr Dateien zu speichern sind, desto mehr Aufwand sollte zur Vermeidung unentdeckter Fehler eingesetzt werden.

Dauer der Aufbewahrung
Im Zuge von Maßnahmen zur Langzeitarchivierung werden Dateien beispielsweise bei einem Ausfall von Speicherkomponenten migriert, so dass der gleiche Zusammenhang wie bei einer großen Anzahl von Dateien gilt. Je länger die Dateien zu speichern sind, desto mehr Aufwand sollte zur Vermeidung unentdeckter Fehler eingesetzt werden.

Die Bewertung der einzelnen Kategorien kann durch ein dreistufiges Verfahren erfolgen: gering (+), mittel (++) und hoch (+++). In enger Kooperation mit den Fachwissenschaftlern lässt sich auf diese Weise eine realistische Abschätzung der Bedürfnisse erstellen. Im Falle einer erhöhten Bedeutung einzelner Kategorien ist ebenfalls die Einführung einer Gewichtung der Kategorien möglich und sollte vor allem bei der Bewertung des Wertes der Daten in Betracht gezogen werden.

Wichtig im Kontext einer Langzeitarchivierung ist die Unterscheidung zwischen entdeckten und unentdeckten Fehlern und die daraus resultierende Berücksichtigung der unentdeckten Fehler. Abbildung 6.1 zeigt eine Einordnung der Fehler. In Abhängigkeit der Wichtigkeit der Daten, die durch die oben beschriebenen Ausprägungen definiert wird, können Maßnahmen zur Bit Preservation durchführen. Je wichtiger die Daten, desto mehr Anstrengungen sollten zur Reduzierung der Wahrscheinlichkeit unentdeckter Fehler unternommen werden. Zur Vereinfachung lassen sich die möglichen Szenarien

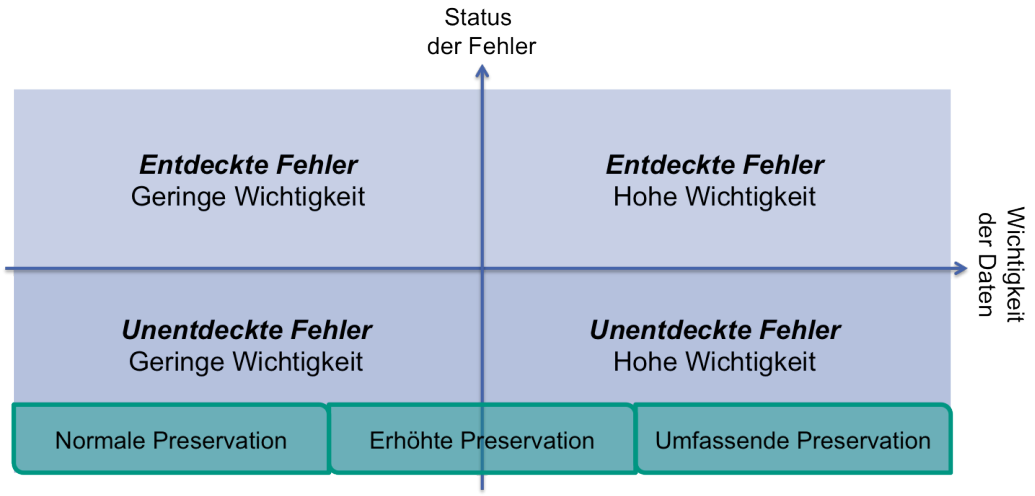

Abbildung 6.1: Kategorisierung von Maßnahmen zur Bit Preservation.

zu Bit Preservation Level zusammenfassen. Die Bewertungen der einzelnen Kategorien werden hierzu zu einem Wert in Abhängigkeit der Gewichtung der Kategorien gemittelt. Bei entdeckten Fehlern hat sich eine ebenfalls dreistufige Einteilung als praktikabel erwiesen und lässt sich auf den Umgang mit unentdeckten Fehlern übertragen.

normale Preservation ist der Basisfall beim Umgang mit unentdeckten Fehlern (gemittelte Bewertung liegt im Intervall $[0, 1]$). Anwendungsfall sind beispielsweise wenige Bilddateien, die für einen begrenzten Zeitraum gespeichert werden sollen.

erhöhte Preservation sollte bei einer gemittelten Bewertung im Intervall $(1, 2]$ gewählt werden. Es kann bei vertretbarem Aufwand eine geringere Wahrscheinlichkeit für unentdeckte Fehler als bei der normalen Preservation erreicht werden.

umfassende Preservation bietet maximale Zuverlässigkeit beim Umgang mit unentdeckten Fehlern (gemittelte Bewertung im Intervall $(2, 3]$), erfordert aber auch den größten Aufwand. Dieses Level wird vor allem bei der Speicherung einzigartiger, nicht reproduzierbarer Daten gewählt oder falls beispielsweise bereits ein einzelner Fehler zu Datenverlusten führt.

Die architekturelle Ausgestaltung der einzelnen Bit Preservation Level ist von den vorhandenen Ressourcen abhängig und kann je nach Langzeitarchivierungskomponente variieren. Der Anbieter kann die vorgestellten Formeln nach einer Quantifizierung seiner Architekturkomponenten nutzen, um eine Abschätzung der verschiedenen Fälle vorzunehmen. Gleichzeitig bietet die Modellierung die Möglichkeit, alternative Architekturen zu untersuchen.

Eine mögliche Strategie für den Umgang mit (unentdeckten) Fehlern ist das Lesen der gewünschten Datei aus mehreren Speicherkomponenten mit anschließendem bitweisen Vergleich. Auf diese Weise lässt sich die Fehlerwahrscheinlichkeit deutlich reduzieren. Es bleibt jedoch stets eine Restfehlerwahrscheinlichkeit vorhanden und im Falle großer Dateien ist dieses Vorgehen auf Grund hoher Ressourcenauslastung und Kosten nicht praktibel. Aus diesem Grund empfiehlt sich bereits bei der Planung einer Architektur zur Bit Preservation unentdeckte Fehler miteinzubeziehen. Im Falle von gleichwertigen Speicherressourcen können die folgenden Zusammenhänge für den Umgang mit unentdeckten Fehlern abgeleitet werden:

- Die synchrone Replikation ist gegenüber der asynchronen Replikation zu bevorzugen.
- Mit Ausnahme der synchronen Replikation aus Nutzersicht vergrößert sich die Wahrscheinlichkeit unentdeckter Fehler bei einer höheren Anzahl von Repliken. In diesen Szenarien gilt daher der Zusammenhang „je mehr Repliken, desto sicherer die Daten“ nicht.
- Besteht durch mehrere Repliken die Möglichkeit eines Ausweichschrittes, führt der erste Ausweichschritt zur größten Verbesserung der Fehlerwahrscheinlichkeit. Weitere Ausweichschritte bei einer festen Anzahl von Repliken sollten nur für ein hohes Bit Preservation Level durchgeführt werden.
- Berechnungen von Prüfsummen sind außerhalb des Systems zur Bit Preservation effektiver als innerhalb des Systems. Für eine hohe Zuverlässigkeit sollten verschiedene Lokationen der Prüfsummenberechnung genutzt werden. Für Daten des höchsten Preservation Levels sollten bitweise Vergleiche in Betracht gezogen werden, da jede Prüfsumme eine Wahrscheinlichkeit für unentdeckte Fehler aufweist. Allerdings haben auch Prüfsummen mit relativ hoher Fehlerwahrscheinlichkeit eine merkliche Verringerung der Gesamtfehlerwahrscheinlichkeit zur Folge.

Für das Beispiel aus Kapitel 4.5 ergibt sich beim Übergang von der asynchronen zur synchronen Replikation eine Verbesserung der Fehlerwahrscheinlichkeit von 49,72% bzw. 66,07% ohne bzw. mit Berechnung einer Prüfsumme. Mit einem Ausweichschritt lässt sich eine Verbesserung der Fehlerwahrscheinlichkeit von 49,24% bzw. 24,78% im Szenario der synchronen bzw. asynchronen Replikation erreichen.

Bei der Wahl einer Architektur zur Bit Preservation müssen allerdings neben den beschriebenen Zusammenhängen auch stets die entdeckten Fehler in Betracht gezogen werden. Insbesondere sind für den Katastrophenfall eine ausreichende Anzahl unabhängiger Repliken vorzuhalten. Eine passende Architektur ist daher stets ein Kompromiss zwischen den spezifischen Anforderungen von entdeckten und unentdeckten Fehlern.

6.5 Auswirkungen auf die Langzeitarchivierung

Die Langzeitarchivierung von Daten ist ein komplexes Themengebiet, das sich durch neue Erkenntnisse in einem ständigen Wandel befindet. Zusammenfassend lässt sich für die in Kapitel 1.4 aufgeworfenen Forschungsfragen formulieren:

Wie sieht eine Architektur zur archivalischen Speicherung heterogener Forschungsdaten aus?
Die vorgestellte Architektur zur archivalischen Speicherung heterogener Forschungsdaten verwendet anerkannte Standards zur Unterstützung der Nachhaltigkeit der Datenspeicherung. Durch eine strikte Trennung in funktionale Einheiten und die Spezifikation von Schnittstellen können Technologiewechsel durchgeführt und eine hohe Flexibilität gewährleistet werden. Interoperabilität auf Speicherebene kann an Hand der fünf Grundfunktionalitäten in Kapitel 3.3 beschrieben werden. Auf diese Weise wird von der konkreten Speicherinfrastruktur abstrahiert und eine generische Migration bereitgestellt. Alle jetzigen und zukünftigen Speicherkomponenten, die die definierten Grundfunktionalitäten erfüllen, können innerhalb der Architektur zur archivalischen Speicherung heterogener Forschungsdaten verwendet werden und Daten austauschen.

Wie lässt sich Zuverlässigkeit in Systemen zur Bit Preservation quantifizieren?
Zur Abschätzung der Risiken einer langfristigen Datenspeicherung ist auf

Grund der Bedeutung für die Langzeitarchivierung die Wahrscheinlichkeit unentdeckter Fehler die wichtigste Metrik zur Bestimmung der Zuverlässigkeit. Unentdeckte Fehler sind als besonders kritisch zu bewerten und müssen daher in den Fokus der Betrachtungen gerückt werden. Auf diese Weise wurden verschiedene Speicherstrategien inklusive sechs Replikationsszenarien sowie Prüfsummen an zwei Lokationen untersucht. Unter der Annahme gleichwertiger Speicherkomponenten gilt:

- *Welche Auswirkung hat mehrfache Replikation auf die Zuverlässigkeit?* - Mit Ausnahme der synchronen Replikation aus Nutzersicht (vergleiche Kapitel 4.5.1) erhöht sich die Wahrscheinlichkeit unentdeckter Fehler bei einer steigenden Anzahl von Repliken. Im Szenario der synchronen Replikation ist die Wahrscheinlichkeit ohne Ausweichen gleichbleibend bzw. verringert sich bei einer steigenden Anzahl von Repliken, falls Ausweichen erlaubt wird.

- *Welche Auswirkung hat der Einsatz von Prüfsummen an unterschiedlichen Stellen auf die Zuverlässigkeit?* - Der Einsatz von Prüfsummen verringert durch die Möglichkeit des Ausweichens auf alternative Speicherkomponenten die Wahrscheinlichkeit unentdeckter Fehler. Die Berechnung einer Prüfsumme auf Nutzerseite verspricht eine deutlichere Verbesserung als die Berechnung innerhalb der Architektur zur Bit Preservation. Zur Verringerung der Wahrscheinlichkeit unentdeckter Fehler kann eine Kombination verschiedener Lokationen der Prüfsumme gewählt werden.

- *Welche Auswirkung haben steigende Datenmengen auf die Zuverlässigkeit* - Bei einer steigenden Datenmenge erhöht sich die Wahrscheinlichkeit für unentdeckte Fehler.

Die vorliegende Arbeit liefert die notwendige Methodik, Zuverlässigkeit in Architekturen zur Bit Preservation zu quantifizieren und darauf aufbauend die zu speichernden Daten zur Wahl einer passenden Strategie zu bewerten. Exemplarisch wurde dieses Verfahren für gleichwertige Speicherkomponenten durchgeführt. Durch die Wahl passender Parameter ist eine leichte Übertragbarkeit auf alternative Szenarien wie ebenfalls in der Realität vorkommende heterogene Speicherkomponenten gegeben, die eine Reihe weiterer Fragestellungen eröffnen.

Eine verstärkte Berücksichtigung der Wahrscheinlichkeit von unentdeckten Fehlern wird die Herausforderung einer Langzeitarchivierung von Daten nachhaltig verändern. Etablierte Methoden zur Sicherung der Datenintegrität müssen in Bezug auf unentdeckte Fehler neu bewertet werden, um eine optimale Strategie für den Umgang mit sowohl entdeckten als auch unentdeckten Fehlern zu entwickeln. Empfehlenswert ist in diesem Kontext ebenfalls die Erforschung und Entwicklung neuer Technologien, die beide Fehlerarten kombiniert abdecken können. Infrastrukturanbieter stehen vor der Herausforderung, die Zuverlässigkeit ihrer zur Verfügung gestellten Komponenten möglichst exakt zu quantifizieren, um optimale Architekturen entwerfen zu können. Basierend auf den Ergebnissen müssen unter Umständen große existierende Speicherinfrastrukturen zur Unterstützung einer erfolgreichen Langzeitarchivierung angepasst werden.

7 Schlussfolgerungen

Die langfristige Erhaltung von Forschungsdaten ist eine seit Jahren bekannte Herausforderung, die auf Grund ihrer Komplexität bis zum heutigen Tag nicht gelöst werden konnte. Durch weltweit steigende Datenmengen und eine immer datenintensivere Forschung ist das Thema Langzeitarchivierung jedoch aktueller denn je. Punktuell konnten bereits Erfolge erzielt werden, die jedoch kontinuierlich an neue Entwicklungen angepasst werden müssen. Die Langzeitarchivierung kann daher nicht als statischer Prozess, sondern muss als dynamische Aktivität verstanden werden.

Das in dieser Dissertation vorgestellte Modell zur Quantifizierung der Zuverlässigkeit in Architekturen zur Bit Preservation stellt unentdeckte Fehler in den Mittelpunkt der Analyse und bietet ein methodisches Vorgehen mit dieser Fehlerart. Unentdeckte Fehler stellen für eine nachhaltige Langzeitarchivierung auf Grund der potentiellen Verstärkung im Laufe der Jahre eine besonders kritische Bedrohung dar, so dass sie bei jeglichen Preservationsaktivitäten berücksichtigt werden sollten. An Hand einer Referenzimplementierung können verschiedene Strategien untersucht und die Grundlage für die Abschätzung der Risiken realer Verfahren und Konfigurationen gelegt werden. Auf diese Weise werden Infrastrukturanbietern die Möglichkeiten und Werkzeuge geboten, ihre bestehenden Architekturen zu quantifizieren und im Falle von erkanntem Handlungsbedarf anzupassen.

Die Betrachtung unentdeckter Fehler im Bereich von Architekturen zur Bit Preservation bietet ebenfalls einen Ansatzpunkt für weitere Forschungen. Auf Grund der stetig steigenden Menge an Forschungsdaten gewinnen die vorher als zu gering erachteten Wahrscheinlichkeiten unentdeckter Fehler immer mehr an Bedeutung und erfordern Lösungen. Beispielhaft lassen sich die folgenden Themenbereiche identifizieren:

Verfahren zur Messbarkeit von unentdeckten Fehlern
Für eine realistische Quantifizierung der Wahrscheinlichkeit unentdeckter Fehler in Architekturen zur Bit Preservation müssen die Fehlerwahrscheinlichkeiten der verwendeten Komponenten bestimmt werden. Die Wahrscheinlichkeit unentdeckter Fehler ist jedoch im Gegensatz zum Umgang mit entdeckten Fehlern in vielen Fällen nicht direkt bestimmbar und hängt im Algorithmenbereich von der verwendeten Methode ab. Die Entwicklung eines präzisen Verfahrens zur (indirekten) Messbarkeit von unentdeckten Fehlern ist daher zwingend notwendig.

Modellierung komplexerer Szenarien
In der vorliegenden Dissertation wurden einfache Replikations- und Prüfsummenszenarien analysiert und quantifiziert. Mit Hilfe der vorgestellten Methodik lassen sich auch komplexere Szenarien wie beispielsweise mehrfache Prüfsummenberechnung beim Lesen und Schreiben, variierende Ausweichschemata, mehrfache Bit Preservation Softwarekomponenten oder kombinierte Architekturen untersuchen.

Modellierung zeitlicher Abhängigkeiten
Im vorliegenden Modell wurden die Fehlerwahrscheinlichkeiten der einzelnen Komponenten als zeitlich konstant angenommen. Wie bereits in Kapitel 6.3 ausgeführt, lässt sich auf diese Weise eine Quantifizierung durchführen. Die Auswirkungen einer Einbeziehung zeitlicher Abhängigkeiten in das Modell bleiben jedoch zu untersuchen. Insbesondere stellt sich die Frage, ob durch diese Erweiterung eine realistischere Abbildung des Sachverhalts erreicht werden kann.

Ausarbeitung Preservationskategorien und Adaption für verschiedene Anwendungsfälle
Die Formulierung der Preservationskategorien in Kapitel 6.4 bleibt auf einem abstrakten Niveau. Für unterschiedliche Speicheranbieter und Anwendungsfälle kann die Ausgestaltung der Kategorien untersucht und der entstehende Nutzen bewertet werden. Durch die Durchführung der Kategorisierung können zukünftigen Nutzern Best Practices zur Verfügung gestellt werden. Wird bei der Verteilung der Forschungsdaten eine Kategorie besonders häufig angewählt, kann der entsprechende Speicheranbieter sein Infrastrukturangebot an die entsprechenden Bedürfnisse anpassen.

Erweiterung der Preservationsstrategien zur Behandlung von entdeckten und unentdeckten Fehlern
Im Bereich der Langzeitarchivierung sind Strategien zur Zeit auf den Umgang mit entdeckten Fehler ausgelegt. Durch die zunehmende Wichtigkeit müssen allerdings auch unentdeckte Fehler berücksichtigt und in die Strategien integriert werden. Kapitel 5 zeigt, dass sich entdeckte und unentdeckte Fehler nicht gleich verhalten müssen. Folglich muss eine Untersuchung der Preservationsstrategien in Bezug auf beide Fehlerarten erfolgen. Auf diese Weise kann abgewägt werden, welche Strategie den größten Nutzen für den

entsprechenden Anwendungsfall verspricht.

Verbesserung der Wahrscheinlichkeit unentdeckter Fehler im Algorithmenbereich
Algorithmen, die im Kontext der Langzeitarchivierung Verwendung finden, sollten eine möglichst geringe Wahrscheinlichkeit unentdeckter Fehler aufweisen. Der im Ethernet verwendete CRC-32 versprach bei der Einführung eine ausreichende Fehlererkennung, bei den heutzutage übertragenen Datenmengen werden zu viele Fehler nicht mehr erkannt. Sofern nicht vorhanden, können die verwendeten Algorithmen in Bezug auf die Wahrscheinlichkeit unentdeckter Fehler quantifiziert und anschließend verglichen bzw. optimiert werden.

Auswirkungen der Data Curation auf entdeckte und unentdeckte Fehler
Im Bereich der Bit Preservation erlaubt das vorgestellte Modell eine Quantifizierung der unentdeckten Fehler. Welche Auswirkungen notwendige, verschiedene Kurationstätigkeiten sowohl auf entdeckte als auch auf unentdeckte Fehler haben, bleibt jedoch zu untersuchen. Zu diesem Zweck müssen Verfahren der Data Curation in Bezug auf (un)entdeckte Fehler beschrieben und anschließend analysiert werden, um eine Quantifizierung der Auswirkungen zu ermöglichen.

Erweiterung der Architektur zur archivalischen Speicherung heterogener Forschungsdaten
Im Zusammenhang mit der Untersuchung der Auswirkungen von Kurationstätigkeiten kann die vorgestellte Architektur zur archivalischen Speicherung heterogener Forschungsdaten um Funktionalitäten der Content Preservation und Data Curation ergänzt werden. Es stellen sich beispielsweise die Fragen, welche Schnittstellen für diese höherwertigen Dienste benötigt werden oder welche Funktionalitäten technisch wie unterstützt werden können.

Der Themenbereich Langzeitarchivierung ist so umfangreich, dass belastbare Ergebnisse nur in Kollaborationen zu erwarten sind. Insbesondere Teile der Content Preservation und der Data Curation werden nur durch gemeinsame Forschungsaktivitäten in einer engen Kooperation zwischen Informatik und den jeweiligen Fachwissenschaften möglich sein, da diese Bereiche speziell auf die Forschungsdaten zugeschnitten sind und nur zum Teil generisch

konzipiert werden können. Gerade die Bibliotheken sowie Forscherinnen und Forscher der Geistes- und Kulturwissenschaften haben eine lange Tradition einer langfristigen Erhaltung analoger Objekte und können durch ihre etablierten Methoden und Verfahren die Forschung im Bereich der Langzeitarchivierung entscheidend voranbringen. Zusätzlich müssen für die Diskussion und Schaffung von Standards und Best Practices weltweit Experten und Wissenschaftler eingebunden werden, um eine kritische Masse zur Umsetzung zu erreichen und Insellösungen auf den verschiedenen Kontinenten zu vermeiden.

Eine fehlerlose Langzeitarchivierung inklusive einer kompletten Elimination des Risikos unentdeckter Fehler wird jedoch auch in Zukunft nicht möglich sein. Selbst sehr aufwändige Integritätsüberprüfungen wie ein bitweiser Vergleich des gesamten Datenbestandes entdecken nicht alle Fehler. Leistungsstärkere bzw. effizientere Hard- und Software erlauben zwar auch bei stetig steigender Datenmenge immer komplexere und aufwändigere Verfahren zur weiteren Reduzierung der Wahrscheinlichkeit unentdeckter Fehler, ein Restrisiko kann jedoch in keinem Fall ausgeschlossen werden. Für eine optimale Zuteilung der stets beschränkten Ressourcen, die zur Langzeitarchivierung der Daten eingesetzt werden können, ist daher eine Kategorisierung des Datenbestandes wie in Kapitel 6.4 und darauf basierende Fokussierung der Maßnahmen zwingend notwendig.

Infrastrukturprojekte wie beispielsweise DARIAH müssen im Kontext der steigenden Datenmengen und einer sehr langfristigen Aufbewahrungsdauer Strategien für den Umgang mit unentdeckten Fehler bereitstellen. Bei bereits existierenden großen Speicherinfrastrukturen werden Anpassungen nur sehr aufwändig umsetzbar sein. Aus diesem Grund sollten Anstrengungen zur Sensibilisierung in Bezug auf unentdeckte Fehler verstärkt werden, so dass bei aus Technologiewechseln resultierenden, notwendigen Umbaumaßnahmen oder Neukonzeptionen Erkenntnisse aus der Forschung im Bereich Preservation berücksichtigt werden können.

Insgesamt wird durch die vorliegende Dissertation die Grundlage für eine Quantifizierung und Vergleichbarkeit der Zuverlässigkeit von Architekturen zur Bit Preservation entwickelt. Im Zusammenspiel mit vielfältigen weiteren Forschungsaktivitäten ist das vorgestellte Modell ein wichtiger Schritt auf dem Weg zur langfristigen und nachhaltigen Erhaltung heterogener Forschungsdaten.

Literatur

[1] Addis, Matthew ; Jacyno, Mariusz: Tools for Modelling and Simulating Migrationbased Preservation. Version: Dezember 2010. {http://eprints.soton.ac.uk/339147/}. 2010. – Forschungsbericht

[2] Amazon Web Services Inc.: *Amazon Simple Storage Service (S3)*. http://awsdocs.s3.amazonaws.com/S3/latest/s3-api.pdf, März 2006. – abgerufen 23.10.2015

[3] Ambacher, Bruce ; Ashley, Kevin ; Berry, John ; Brooks, Connie ; Dale, Robin L. ; Flecker, Dale ; Giaretta, David ; Hamidzadeh, Babak ; Johnson, Keith ; Jones, Maggie ; McGovern, Nancy ; McHugh, Andrew ; Sawyer, Don ; Steenbakkers, Johan: *Trustworthy Repositories Audit & Certification (TRAC)*. http://www.crl.edu/sites/default/files/attachments/pages/trac_0.pdf, Februar 2007. – abgerufen 23.10.2015

[4] *APARSEN - Alliance Permanent Access to the Records of Science in Europe Network*. http://www.alliancepermanentaccess.org/, . – abgerufen 23.10.2015

[5] Aschenbrenner, Andreas ; Dickmann, Frank ; Enke, Harry ; Fritzsch, Bernadette ; Lautenschlager, Michael ; Löhnhardt, Benjamin ; Ludwig, Jens ; Rathmann, Torsten ; Reiser, Angelika ; Schintke, Florian ; Stegmann, Jens ; Strathmann, Stefan: Generische Langzeitarchivierungsarchitektur für D-Grid. Version: Januar 2010. http://www.wissgrid.de/publikationen/deliverables/wp3/WissGrid-D3.1-LZA-Architektur-v1.1.pdf. 2010. – Forschungsbericht

[6] Awre, Chris ; Cramer, Tom ; Green, Richard ; McRae, Lynn ; Sadler, Bess ; Sigmon, Tim ; Staples, Thornton ; Wayland, Ross: Project Hydra: Designing & Building a Reusable Framework for Multi-purpose, Multifunction, Multi-institutional Repository-Powered Solutions. In: *4th International Conference on Open Repositories*, 2009

[7] Bairavasundaram, Lakshmi N. ; Arpaci-Dusseau, Andrea C. ; Arpaci-Dusseau, Remzi H. ; Goodson, Garth R. ; Schroeder,

Bianca: An Analysis of Data Corruption in the Storage Stack. In: *ACM Transactions on Storage (TOS)* 4 (2008), Nr. 3, S. 8

[8] Bairavasundaram, Lakshmi N. ; Goodson, Garth R. ; Pasupathy, Shankar ; Schindler, Jiri: An Analysis of Latent Sector Errors in Disk Drives. In: *ACM SIGMETRICS Performance Evaluation Review* 35 (2007), Nr. 1, S. 289–300

[9] Baker, Mary ; Shah, Mehul ; Rosenthal, David S. H. ; Roussopoulos, Mema ; Maniatis, Petros ; Giuli, Thomas J. ; Bungale, Prashanth: A Fresh Look at the Reliability of Long-term Digital Storage. In: *ACM SIGOPS Operating Systems Review* Bd. 40, 2006, S. 221–234

[10] Barateiro, José ; Antunes, Gonçalo ; Freitas, Filipe ; Borbinha, José: Designing Digital Preservation Solutions: A Risk Management-based Approach. In: *International Journal of Digital Curation* 5 (2010), Nr. 1, S. 4–17

[11] Barbedo, Francisco ; Castro, Rui ; Corujo, Luís ; Faria, Luís ; Ferreira, Miguel ; Henriques, Cecília ; Ramalho, José C.: RODA-A Service-Oriented Repository to Preserve Authentic Digital Objects. In: *4th International Conference on Open Repositories*, 2009

[12] Becker, Petrus: *Die Benediktinerabtei St.Eucharius-St.Matthias vor Trier*. Berlin : Walter de Gruyter, 1996. – (Germania sacra, Neue Folge 34; Die Bistümer der Kirchprovinz Trier, Das Erzbistum Trier 8)

[13] Bekaert, Jeroen ; Liu, Xiaoming ; Sompel, Herbert Van d.: aDORe: A Modular and Standards-based Digital Object Repository at the Los Alamos National Laboratory. In: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2005, S. 367–367

[14] Blanke, Tobias ; Bryant, Michael ; Hedges, Mark ; Aschenbrenner, Andreas ; Priddy, Michael: Preparing DARIAH. In: *IEEE 7th International Conference on E-Science (e-Science)*, 2011, S. 158–165

[15] Cantor, Scott ; Kemp, John ; Philpott, Rob ; Maler, Eve: Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. Version: März 2005. `http://docs.oasis-open.org/`

security/saml/v2.0/saml-core-2.0-os.pdf. 2005. – Forschungsbericht

[16] Caplan, Priscilla: DAITSS, an OAIS-based Preservation Repository. In: *Proceedings of the 2010 Roadmap for Digital Preservation Interoperability Framework Workshop*, 2010, S. 17

[17] Caplan, Priscilla: The Florida Digital Archive and DAITSS: A Model for Digital Preservation. In: *Library Hi Tech* 28 (2010), Nr. 2, S. 224–234

[18] CDMI Technical Working Group: *Cloud Data Management Interface (CDMI)*. http://www.snia.org/sites/default/files/CDMI_Spec_v1.1.1.pdf, März 2015. – abgerufen 23.10.2015

[19] Conway, Paul: *Preservation in the Digital World*. Commission on Preservation and Access, 1996 http://www.clir.org/pubs/reports/conway2/index.html

[20] Dang, Quynh: *Recommendation for Applications Using Approved Hash Algorithms*. http://csrc.nist.gov/publications/nistpubs/800-107-rev1/sp800-107-rev1.pdf, August 2012. – abgerufen 23.10.2015

[21] Data Observation Network for Earth (DataONE): *DataONE Architecture*. https://releases.dataone.org/online/api-documentation-v1.2.0/, Oktober 2013. – abgerufen 23.10.2015

[22] *Data Seal of Approval (DSA)*. http://www.datasealofapproval.org/, . – abgerufen 23.10.2015

[23] Deutsch, Peter ; Gailly, Jean-Loup: Zlib Compressed Data Format Specification Version 3.3 / RFC 1950. Version: Mai 1996. http://www.hjp.at/doc/rfc/rfc1950.html. 1996. – Forschungsbericht

[24] *DFG-Viewer*. http://dfg-viewer.de, . – abgerufen 23.10.2015

[25] Digital Preservation Coalition: *Preservation Management of Digital Materials: The Handbook*. http://www.dpconline.org/advice/preservationhandbook/introduction/definitions-and-concepts/, 2015. – abgerufen 23.10.2015

[26] Elliott, James ; Hoemmen, Mark ; Mueller, Frank: A Numerical Soft Fault Model for Iterative Linear Solvers. In: *Proceedings of the ACM Symposium on High-Performance Parallel and Distributed Computing*, 2015

[27] Ernst, Michael ; Fuhrmann, Patrick ; Gasthuber, Martin ; Mkrtchyan, Tigran ; Waldman, Charles: dCache, a Distributed Data Storage Caching System. In: *Computing in High Energy and Nuclear Physics (CHEP 2001)*, 2001

[28] *ESFRI - European Strategy Forum on Research Infrastructures*. `http://ec.europa.eu/research/infrastructures`,

[29] Ex Libris Ltd.: The Ability to Preserve a Large Volume of Digital Assets. Version: 2010. `http://www.exlibrisgroup.com/files/Products/Preservation/RosettaScalingProofofConcept.pdf`. 2010. – Forschungsbericht

[30] Factor, Michael ; Henis, Ealan ; Naor, Dalit ; Rabinovici-Cohen, Simona ; Reshef, Petra ; Ronen, Shahar ; Michetti, Giovanni ; Guercio, Maria: Authenticity and Provenance in Long Term Digital Preservation: Modeling and Implementation in Preservation Aware Storage. In: *Workshop on the Theory and Practice of Provenance*, 2009

[31] Ferrin, Bruce G. ; Plank, Richard E.: Total Cost of Ownership Models: An Exploratory Study. In: *Journal of Supply chain management* 38 (2002), Nr. 2, S. 18–29

[32] Fiala, David ; Mueller, Frank ; Engelmann, Christian ; Riesen, Rolf ; Ferreira, Kurt ; Brightwell, Ron: Detection and Correction of Silent Data Corruption for Large-scale High-performance Computing. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012, S. 78

[33] Fielding, R. ; Mogul, J. C. ; Frystyk, H. ; Masinter, L. ; Leach, P. ; Lee, Berners T.: *RFC 2616, Hypertext Transfer Protocol – HTTP/1.1*. Version: Juni 1999. `http://www.ietf.org/rfc/rfc2616.txt`. – abgerufen 23.10.2015

[34] Fletcher, John: An Arithmetic Checksum for Serial Transmissions. In: *IEEE Transactions on Communications* (1982), S. 247–252

[35] Foster, Ian ; Kesselman, Carl ; Tuecke, Steven: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In: *International Journal of High Performance Computing Applications* 15 (2001), August, Nr. 3, 200–222. http://www.globus.org/alliance/publications/papers/anatomy.pdf. – ISSN 1094–3420

[36] Gentzsch, Wolfgang ; Lecarpentier, Damien ; Wittenburg, Peter: Big Data in Science and the EUDAT Project. In: *Annual SRII Global Conference*, 2014, S. 191–194

[37] Giaretta, David: The CASPAR Approach to Digital Preservation. In: *International Journal of Digital Curation* 2 (2008), Nr. 1, S. 112–121

[38] Giaretta, David et al.: Report on a Common Vision of Digital Preservation: Progess to Year 3. Version: Januar 2014. http://www.alliancepermanentaccess.org/wp-content/uploads/downloads/2014/06/APARSEN-REP-D11_3-01-1_1_inclURN.pdf. 2014. – Forschungsbericht

[39] Gibson, Garth A. ; Nagle, David F. ; Amiri, Khalil ; Chang, Fay W. ; Feinberg, Eugene: A Case for Network-attached Secure Disks / DTIC Dokument. Version: 1996. http://repository.cmu.edu/cgi/viewcontent.cgi?article=1154&context=pdl. 1996. – Forschungsbericht

[40] Gietz, Peter ; Haase, Martin: DARIAH Authorization and Authentication Infrastructure. 2011. – Forschungsbericht

[41] Greenan, Kevin M. ; Plank, James S. ; Wylie, Jay J.: Mean Time to Meaningless: MTTDL, Markov Models, and Storage System Reliability. In: *Proceedings of the 2nd USENIX Conference on Hot Topics in Storage and File Systems*, 2010

[42] Hitchcock, Steve ; Tarrant, David ; Carr, Les: Preserv2 Final Report. Version: Juli 2009. {http://eprints.soton.ac.uk/268148/}. 2009. – Forschungsbericht

[43] Hockey, Susan: The History of Humanities Computing. In: *A Companion to Digital Humanities* (2004), S. 3–19

[44] Innocenti, Perla ; Ross, Seamus ; Maceviciute, Elena ; Wilson, Tom ; Ludwig, Jens ; Pempe, Wolfgang: Assessing Digital Preservation Frameworks: The Approach of the SHAMAN Project. In: *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, 2009, S. 61

[45] International Organization for Standardization: Information Technology - Vocabulary (ISO/IEC 2382:2015). – Forschungsbericht

[46] Johnston, Leslie: Developing Bit Preservation Services at the Library of Congress. In: *Proceedings of the 2010 Roadmap for Digital Preservation Interoperability Framework Workshop*, 2010, S. 12

[47] Jurik, Bolette A. ; Nielsen, Anders B. ; Zierau, Eld M. O.: Flexible Bit Preservation on a National Basis. In: *Archiving Conference* Bd. 2012, 2012, S. 2–7

[48] Kaschte, Axel: *Digital Preservation System - ExLibris Rosetta*. `http://indico.cern.ch/event/48321/contribution/39/material/slides/1.pdf`, Juni 2009. – abgerufen 23.10.2015

[49] Krioukov, Andrew ; Bairavasundaram, Lakshmi N. ; Goodson, Garth R. ; Srinivasan, Kiran ; Thelen, Randy ; Arpaci-Dusseau, Andrea C. ; Arpaci-Dusseau, Remzi H.: Parity Lost and Parity Regained. In: *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, 2008, S. 127–141

[50] Kunze, John A. ; Haye, Martin ; Hetzner, Erik ; Reyes, Mark ; Snavely, Cory: *Pairtrees for Collection Storage (V0.1)*. `https://wiki.ucop.edu/display/Curation/PairTree?preview=/14254128/16973838/PairtreeSpec.pdf`, Dezember 2008. – abgerufen 23.10.2015

[51] Lagoze, Carl ; Payette, Sandy ; Shin, Edwin ; Wilper, Chris: Fedora: An Architecture for Complex Objects and Their Relationships. In: *International Journal on Digital Libraries* 6 (2006), Nr. 2, S. 124–138

[52] Lagoze, Carl ; Sompel, Herbert Van d. ; Johnston, Pete ; Nelson, Michael ; Sanderson, Robert ; Warner, Simeon: *Open Archives Initiative Object Reuse and Exchange (OAI-ORE) User Guide - Primer*.

https://www.openarchives.org/ore/1.0/primer, Oktober 2008. – abgerufen 23.10.2015

[53] Lecarpentier, Damien ; Wittenburg, Peter ; Elbers, Willem ; Michelini, Alberto ; Kanso, Riam ; Coveney, Peter ; Baxter, Rob: EUDAT: A New Cross-Disciplinary Data Infrastructure for Science. In: *International Journal of Digital Curation* 8 (2013), Nr. 1, S. 279–287

[54] Leung-Yan-Cheong, Sik K. ; Hellman, Martin E.: Concerning a Bound on Undetected Error Probability (Corresp.). In: *IEEE Transactions on Information Theory* 22 (1976), Nr. 2, S. 235–237

[55] Li, Mingqiang ; Shu, Jiwu: Preventing Silent Data Corruptions from Propagating during Data Reconstruction. In: *IEEE Transactions on Computers* 59 (2010), Nr. 12, S. 1611–1624

[56] Ludwig, Jens ; Enke, Harry: *Leitfaden zum Forschungsdaten-Management*. vwh, 2013. – ISBN 978–3–86488–032–2

[57] Maniatis, Petros ; Roussopoulos, Mema ; Giuli, Thomas J. ; Rosenthal, David S. H. ; Baker, Mary: The LOCKSS Peer-to-peer Digital Preservation System. In: *ACM Transactions on Computer Systems (TOCS)* 23 (2005), Nr. 1, S. 2–50

[58] Martyniak, Cathy ; Nadal, Jake ; Ryder, Becky ; Frangakis, Evelyn ; Blood, George ; Brown, Karen ; Byrnes, Margaret ; Meikle, Sian: *Association for Library Collections and Technical Services - Definitions of Digital Preservation*. http://www.ala.org/alcts/sites/ala.org.alcts/files/content/resources/preserv/defdigpres0408.pdf, Juni 2007. – abgerufen 23.10.2015

[59] May, Peter ; Wilson, Carl: Technical Architecture Report Version 2. Version: März 2014. http://www.scape-project.eu/wp-content/uploads/2014/04/SCAPE_D2.3_BL_V1.0.pdf. 2014. – Forschungsbericht

[60] McDonough, Jerome P.: METS: Standardized Encoding for Digital Library Objects. In: *International Journal on Digital Libraries* 6 (2006), Nr. 2, S. 148–158

[61] McLean, Bradley: *DuraCloud: Federated Repositories and Cyberinfrastructure Open Technologies and Services for Managing Durable Data in the Cloud*. http://hdl.handle.net/2077/21366, Oktober 2009. – abgerufen 13.11.2015

[62] Mydtskov, Lars ; Hofman Hansen, Jens: Cultural Heritage at the Bit Level–A Danish Bit Repository Initiative. In: *Microform & Digitization Review* 41 (2012), Nr. 3-4, S. 140–146

[63] National Institute of Standards and Technology (NIST): *Secure Hash Standard (SHS)*. http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf, März 2012. – abgerufen 23.10.2015

[64] Neuroth, Heike: DARIAH-DE - Forschungsinfrastrukturen für die eHumanities. In: *BIS - Das Magazin der Bibliotheken in Sachsen* 5 (2012), Nr. 3, S. 156–158. – http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-96560

[65] Oprea, Alina ; Juels, Ari: A Clean-Slate Look at Disk Scrubbing. In: *Proceedings of the 8th USENIX Conference on File and Storage Technologies*, 2010, S. 57–70

[66] Panzer-Steindel, Bernd: Data Integrity. Version: April 2007. http://indico.cern.ch/event/13797/session/0/material-old/paper/1?contribId=3. 2007. – Forschungsbericht

[67] Peter, Kathrin: Reliability Study of Coding Schemes for Wide-area Distributed Storage Systems. In: *19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 2011, S. 19–23

[68] Peterson, William W. ; Brown, Daniel T.: Cyclic Codes for Error Detection. In: *Proceedings of the IRE* 49 (1961), Nr. 1, S. 228–235

[69] Phillips, Megan ; Bailey, Jefferson ; Goethals, Andrea ; Owens, Trevor: The NDSA Levels of Digital Preservation: An Explanation and Uses. In: *IS&T Archiving, Washington, USA* (2013)

[70] Pinchbeck, Dan ; Anderson, David ; Delve, Janet ; Alemu, Getaneh ; Ciuffreda, Antonio ; Lange, Andreas: Emulation As a Strategy for the Preservation of Games: The KEEP Project. In: *DiGRA 2009-Breaking New Ground: Innovation in Games, Play, Practice and Theory*, 2009

[71] Pinheiro, Eduardo ; Weber, Wolf-Dietrich ; Barroso, Luiz A.: Failure Trends in a Large Disk Drive Population. In: *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, 2007, S. 17–23

[72] Qasim, Umar ; Farnel, Sharon ; Huck, John: TAP: A Tiered Preservation Model for Digital Resources. In: *iPRES 2013 - Proceedings of the 10th International Conference on Preservation of Digital Objects*, 2013

[73] Rajasekar, Arcot ; Moore, Reagan ; Hou, Chien-yi ; Lee, Christopher A. ; Marciano, Richard ; Torcy, Antoine de ; Wan, Michael ; Schroeder, Wayne ; Chen, Sheau-Yen ; Gilbert, Lucas u. a.: iRODS Primer: Integrated Rule-oriented Data System. In: *Synthesis Lectures on Information Concepts, Retrieval, and Services* 2 (2010), Nr. 1, S. 1–143

[74] Razum, Matthias ; Schwichtenberg, Frank ; Wagner, Steffen ; Hoppe, Michael: eSciDoc Infrastructure: a Fedora-based e-Research Framework. In: *Research and Advanced Technology for Digital Libraries*. Springer, 2009, S. 227–238

[75] Rivest, Ronald L.: *The MD5 Message-Digest Algorithm*. `https://tools.ietf.org/html/rfc1321`, April 1992. – abgerufen 23.10.2015

[76] Rosenthal, David S. H.: Bit preservation: A Solved Problem? In: *International Journal of Digital Curation* 5 (2010), Nr. 1, S. 134–148

[77] Rosenthal, David S. H.: Keeping Bits Safe: How Hard Can It Be? In: *Communications of the ACM* 53 (2010), Nr. 11, S. 47–55

[78] Rozier, Eric W. D. ; Belluomini, Wendy ; Deenadhayalan, Veera ; Hafner, Jim ; Rao, K. K. ; Zhou, Pin: Evaluating the Impact of Undetected Disk Errors in Raid Systems. In: *IEEE/IFIP International Conference on Dependable Systems & Networks*, 2009, S. 83–92

[79] Schmidt, Rainer ; Lindley, Andrew ; King, Ross ; Jackson, Andrew ; Wilson, Carl ; Steeg, Fabian: The Planets IF - A Framework for Integrated Access to Preservation Tools. In: *Proceedings of the 1st International Digital Preservation Interoperability Framework Symposium*, 2010, S. 10:1–10:8

[80] Sierman, Barbara: Report on Comparison of Planets with OAIS. Version: Juni 2007. `http://www.planets-project.eu/docs/reports/Planets_PP7-D1_ReportOnComparisonOfPlanetsWithOais.pdf`. 2007. – Forschungsbericht

[81] Siewiorek, Daniel ; Swarz, Robert: *Reliable Computer Systems: Design and Evaluatuion*. Digital Press, 2014. – ISBN 1–55558–075–0

[82] *Silent Data Corruption in Disk Arrays: A Solution*. `https://www.necam.com/Docs/?id=54157ff5-5de8-4966-a99d-341cf2cb27d3`, 2009. – abgerufen 23.10.2015

[83] Smith, MacKenzie ; Barton, Mary ; Bass, Mick ; Branschofsky, Margret ; McClellan, Greg ; Stuve, Dave ; Tansley, Robert ; Walker, Julie H.: dSpace: An Open Source Dynamic Digital Repository. In: *D-Lib Magazine* 9 (2003), Januar, Nr. 1

[84] Sobolewski, John S.: Cyclic Redundancy Check. In: *Encyclopedia of Computer Science*. 2003, S. 476–479

[85] Space Data Systems (CCSDS), Consultative C.: *Reference Model for an Open Archival Information System (OAIS), CCSDS 650.0-M-2, Magenta Book*. `http://public.ccsds.org/publications/archive/650x0m2.pdf`, Juni 2012. – abgerufen 23.10.2015

[86] Staples, Thornton ; Wayland, Ross ; Payette, Sandra: The Fedora Project - An Open-source Digital Object Repository Management System. In: *D-Lib Magazine* 9 (2003), Nr. 4

[87] Strodl, Stephan ; Motlik, Florian ; Rauber, Andreas: Hoppla-Digital Preservation Support for Small Institutions. In: *Research and Advanced Technology for Digital Libraries*, 2009, S. 493–494

[88] Svensson, Patrik: The Landscape of Digital Humanities. In: *Digital Hummanities Quarterly* 4 (2010), Nr. 1. – http://digitalhumanities.org/dhq/vol/4/1/000080/000080.html

[89] *TEI*. http://www.tei-c.org/index.xml, . – abgerufen 23.10.2015

[90] Tessella Group: Preservica Cloud Edition V5.0 Service Description. Version: März 2014. {http://preservica.com/files/2014/04/Preservica-Cloud-Edition-V-5.0-Service-Description-2014.pdf}. 2014. – Forschungsbericht

[91] *The four V's of Big Data*. http://www-01.ibm.com/software/data/bigdata/, September 2015. – abgerufen 23.10.2015

[92] Tonne, Danah ; Rybicki, Jedrzej ; Funk, Stefan E. ; Gietz, Peter: Access to the DARIAH Bit Preservation Service for Humanities Research Data. In: *21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 2013, S. 9–15

[93] University of California Curation Center: *Merritt Storage Service*. https://confluence.ucop.edu/download/attachments/16744547/Merritt-storage-service-latest.pdf?version=7&modificationDate=1320361737000, Juli 2011. – abgerufen 23.10.2015

[94] Van Garderen, Peter: ARCHIVEMATICA: Using Micro-Services and Open-Source Software to Deliver a Comprehensive Digital Curation Solution. In: *iPRES 2013 - Proceedings of the 7th International Conference on Preservation of Digital Objects*, 2010, S. 145

[95] Váradi, Tamás ; Krauwer, Steven ; Wittenburg, Peter ; Wynne, Martin ; Koskenniemi, Kimmo: CLARIN: Common Language Resources and Technology Infrastructure. In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, 2008, S. 1244–1248

[96] Wang, Xiaoyun ; Yin, Yiqun L. ; Yu, Hongbo: Finding Collisions in the Full SHA-1. In: *Advances in Cryptology*, 2005, S. 17–36

[97] Wang, Xiaoyun ; Yu, Hongbo: How to Break MD5 and Other Hash Functions. In: *Advances in Cryptology*, 2005, S. 19–35

[98] Waugh, Andrew ; Wilkinson, Ross ; Hills, Brendan ; Dell'Oro, Jon: Preserving Digital Information Forever. In: *Proceedings of the Fifth ACM Conference on Digital Libraries*, 2000, S. 175–184

[99] Witzke, K. ; Leung, C.: A Comparison of Some Error Detecting CRC Code Standards. In: *IEEE Transactions on Communications* 33 (1985), Nr. 9, S. 996–998

[100] Wood, John ; Andersson, Thomas ; Bachem, Achim ; Best, Christoph ; Genova, Françoise ; Lopez, Diego R. ; Los, Wouter ; Marinucci, Monica ; Romary, Laurent ; Sompel, Herbert Van d. ; Vigen, Jens ; Wittenburg, Peter ; Giaretta, David ; Hudson, Richard L.: *Riding the Wave - How Europe Can Gain From the Rising Tide of Scientific Data*. `http://cordis.europa.eu/fp7/ict/e-infrastructure/docs/hlg-sdi-report.pdf`, Oktober 2010. – abgerufen 23.10.2015

[101] Working Group Data Foundation and Terminology: *RDA DFT Model Overview & Term Definitions*. `https://rd-alliance.org/groups/data-foundation-and-terminology-wg.html`, März 2014. – abgerufen 13.11.2015

[102] Zeilenga, Kurt: *Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map - IETF RFC 4510*. Version: Juni 2006. `http://www.ietf.org/rfc/rfc4510.txt`. – abgerufen 23.10.2015

[103] Zhang, Yupu ; Rajimwale, Abhishek ; Arpaci-Dusseau, Andrea C. ; Arpaci-Dusseau, Remzi H.: End-to-end Data Integrity for File Systems: A ZFS Case Study. In: *Proceedings of the 8th USENIX Conference on File and Storage Technologies*, 2010, S. 29–42

[104] Zierau, Eld Maj-Britt O.: *A Holistic Approach to Bit Preservation*, University of Copenhagen, Dissertation, 2011

A Anhang

Die Kapitel A.1 und A.2 bieten einen detaillierten Überblick über die spezifizierten Schnittstellen der Architektur zur archivalischen Speicherung heterogener Forschungsdaten, zusätzlich wird in A.3 das im Projektkontext verwendete Authentifizierungs- und Autorisierungskonzept beschrieben. Kapitel A.4 illustriert die für Quantifizierungszwecke erstelle Referenzimplementierung, für die nach Spezifikation und ausführlicher Diskussion mit Stakeholdern ein pragmatischer Implementierungsansatz gewählt wurde.

Die Kapitel A.5, A.6, A.7, A.8 und A.9 führen die Datenquellen auf, die die Grundlage der Untersuchungen in Kapitel 5 bilden. Für eine Quantifizierung der Zuverlässigkeit mit Hilfe der Wahrscheinlichkeit unentdeckter Fehler müssen sehr geringe Fehlerwahrscheinlichkeiten betrachtet werden. Trotz der unübersichtlichen Darstellung wurde daher eine Genauigkeit von 50 Nachkommastellen gewählt. Auf diese Weise ist es möglich, selbst kleinste Veränderungen nachzuvollziehen.

A.1 Speicherschnittstelle „DARIAH Storage API“

Die DARIAH Storage API ist eine REST-Schnittstelle basierend auf dem HTTP-Protokoll, die die grundlegenden Methoden zur Datenspeicherung abdeckt. Speziell stimmt sie in Bezug auf Fehlermeldungen und Semantik mit RFC 2616 [33] überein. Im Folgenden werden die bereitgestellten Funktionalitäten sowie Verwendungsmöglichkeiten dargestellt.

Auf Grund der Nutzung der Begriffe Ressourcen und Repräsentationen von Ressourcen war die Entscheidung für einen ressourcenorientierten Architekturstil die logische Konsequenz. Gründe für die Nutzung des HTTP-Protokolls waren die weite Verbreitung, die Nutzbarkeit von beliebigen Orten und die einfache Implementierung mit vielfältigen Techniken.

Durch die Verwendung von HTTP ist die Schnittstelle an eine Menge von Methoden mit spezifischer Semantik gebunden. Innerhalb der Architektur zur archivalischen Speicherung heterogener Forschungsdaten werden die Methoden POST, PUT, GET, HEAD, DELETE und OPTIONS genutzt. Zur effizienten Verwaltung der heterogenen Daten werden generische „Binary Large Objects“ (BLOBs) verwendet. Die Storage API bietet eine Menge von Methoden, um eine flache Sammlung solcher BLOBs zu verwalten. Hierarchische Strukturen werden von höherwertigen Diensten (vergleiche Abbildung 3.2) angeboten.

Jeder BLOB besitzt einen einzigartigen Identifier, der in der URL zur Feststellung der zu bearbeitenden Ressource genutzt wird. Entsprechend besitzt eine URL für die Schnittstelle die Form

`http(s)://<Speicherinfrastruktur>/<ObjektID>`.

Die ObjektID muss innerhalb der Speicherinfrastruktur zur Vermeidung von Konflikten zwischen den Nutzern eindeutig sein. Die einfachste Möglichkeit zur Sicherstellung dieser Bedingung ist die Delegation der Zuweisung der ObjektID an die Infrastruktur anstelle des Nutzers. Innerhalb der Architektur zur archivalischen Speicherung heterogener Forschungsdaten wird die ObjektID daher während der ersten Speicheroperation beispielsweise von der genutzten Datenbank vergeben.

Eine neue Datei wird durch eine POST-Anfrage an die Basis-URL gespeichert. Der eigentliche Inhalt der Anfrage wird dabei im Body der Nachricht übertragen. Ist der Nutzer nicht authentifiziert oder autorisiert, wird die Anfrage zurückgewiesen und die Fehlermeldung „401 - Unauthorized“ versendet. Der Authentifizierungs- und Autorisierungsprozess wird in Kapitel A.3 beschrieben. Für die folgenden Ausführungen wird angenommen, dass der Nutzer zu den gewünschten Aktionen berechtigt ist. In diesem Fall wird die Anfrage akzeptiert, ein eindeutiger Identifer erzeugt und mit „201 - Created“ geantwortet. Der Header der Antwort enthält ein Feld Location mit der URL inklusive der ObjektID, wo der gespeicherte Inhalt mittels nachfolgender GET-Anfragen abrufbar ist.

Zur Modifizierung des Inhalts existierender Objekte in der Speicherinfrastruktur muss der Nutzer dem REST-Ansatz folgen. Zuerst wird der Inhalt mittels einer GET-Anfrage heruntergeladen, danach die Repräsenation lokal bearbeitet und schließlich die neue Version mittels einer PUT-Anfrage an die gleiche Lokation gespeichert. Entsprechend der HTTP-Spezifikation ist PUT eine idempotente Methode. Eine Reihe von PUT-Anfragen auf eine Ressource haben dasselbe Ergebnis wie eine einzelne Anfrage zur Folge. Diese Funktionalität wird häufig zur Sicherstellung der Zuverlässigkeit einer REST-Schnittstelle genutzt. Erhält ein Nutzer im Fehlerfall nicht die Antwort „201 - Created“, kann er dieselbe Anfrage erneut stellen und sicherstellen, dass der Inhalt korrekt übertragen wurde.

Das Löschen eines Objektes verläuft analog zur Aktualisierung. Der Nutzer stellt eine DELETE-Anfrage an die URL inklusive ObjektID und erhält bei Erfolg die Antwort „204 - No Content“. DELETE kann ebenfalls als idempotente Methode implementiert werden. In diesem Fall haben mehrere Anfragen unter Benutzung der gleichen ObjektID stets ein „204 - No Content“ zur

Folge, auch wenn das Objekt bereits bei einem früheren Versuch erfolgreich gelöscht wurde.

Die beiden letzten verwendeten HTTP-Methoden sind OPTIONS und HEAD. HEAD-Anfragen sind für die Verwaltung großer Datenobjekte nützlich, weil die Methode Informationen über ein Objekt liefert, ohne den Inhalt herunterzuladen. Die Antwort einer HEAD-Anfrage entspricht der Antwort einer GET-Anfrage mit dem entscheidenen Unterschied, dass die Antwort keinen Body sondern nur den Header enthält. Ein Klient kann so beispielsweise zuerst den Header einer Datei anfordern, um das letzte Änderungsdatum oder die Größe einer Datei zu überprüfen. Mit diesen Informationen kann er die Entscheidung treffen, ob die Datei heruntergeladen werden soll oder nicht. Als zusätzliche Nutzungsmöglichkeit der HEAD-Methode kann eine Anfrage gestellt werden, um die Existenz einer Datei zu einer speziellen ObjektID zu überprüfen. Für diese Schnittstelle existieren eine sehr große Anzahl von validen URLs, für die der Nutzer stets eine wohldefinierte Antwort erhalten wird. Eine GET- oder HEAD-Anfrage an eine nicht existierende Datei hat in jedem Fall die Antwort „404 - Not Found“ zur Folge. Die OPTIONS-Methode bietet eine einfache Möglichkeit zur Überprüfung der implementierten Methoden für eine spezifische Ressource. Für jede semantisch korrekte URL kann eine OPTIONS-Anfrage gestellt werden und der Nutzer erhält eine Liste von aufrufbaren Methoden. Beispielsweise liefert eine OPTIONS-Anfrage auf die Lokation eines existierenden Objekts die Liste OPTIONS, GET, HEAD, PUT und DELETE. POST ist nicht möglich und nur für die Basis URL `http(s)://<Speicherinfrastruktur>/` erlaubt.

A.2 Preservationsschnittstelle „DARIAH Admin API“

Die Admin API basiert ebenfalls auf dem REST-Architekturstil und dem HTTP-Protokoll zur Sicherung der Konformität zur Storage API. Im Folgenden werden die bereitgestellten Funktionalitäten dargestellt. Die generelle Form einer Anfrage lautet

`http(s)://<Speicherinfrastruktur>/admin/<ObjektID>`,

wobei *admin* den Einstiegspunkt zur Admin API spezifiziert. Bei dieser Schnittstelle sind die HTTP-Methoden PUT und GET zur Interaktion mit dem Bit Preservation Teil der Architektur zur archivalischen Speicherung heterogener Forschungsdaten ausreichend.

Für die Konfiguration von Funktionalitäten zur Bit Preservation wird eine PUT-Anfrage genutzt. Alle Werte werden während einer Speicheroperation

mittels der Storage API auf einen Standardwert gesetzt. Die PUT-Anfrage muss die adressierte Speicherinfrastruktur, die ObjektID der Datei, eine Konfigurationsdatei entsprechend des XML-Schemas der Spezifikation und den Content Type der Konfigurationsdatei enthalten.

Für die Informationen über die Bit Preservation wird eine GET-Anfrage genutzt. Die Anfrage muss die adressierte Speicherinfrastruktur und die ObjektID der Datei, für die Informationen abgerufen werden sollen, enthalten. Optional ist die Angabe, welcher Content Type der angefragten Informationen akzeptiert wird. Die Anfrage

`http(s)://<Speicherinfrastruktur>/admin/<ObjektID>`

liefert alle über die Datei gespeicherten Informationen. Individuelle Elemente können per Erweiterung der URL angefordert werden. Eine Gesamtliste der Erweiterungen und ein XML-Schema für die gelieferten Informationen sind in der Spezifikation hinterlegt.

Die Status- und Fehlermeldungen ähneln sich bei beiden Anfragetypen. Im Falle einer erfolreichen Aktualisierung der Metadaten einer Datei mittels eines PUT wird „200 - OK“ zurückgegeben. Im Falle eines GET wird die Statusmeldung zusammen mit einer Datei mit den gewünschten Informationen und dem Content Type der Datei gesendet. Hat der Nutzer nicht die Berechtigung für die durchgeführte Anfrage, wird „401 - Unauthorized“ zurückgegeben, „404 - Not Found“, wenn die Datei nicht im Speicher vorhanden ist. Zusätzlich kann ein PUT zu einem „400 - Bad Request“ führen, falls die Anfrage nicht dem spezifizierten Schema entspricht. GET liefert ein „406 - Not Acceptable“, falls der gewünschte Content Type nicht geliefert werden kann.

A.3 Authentifizierungs- und Autorisierungsinfrastruktur (AAI)

Die Architektur zur archivalischen Speicherung heterogener Forschungsdaten ist in die im DARIAH Projekt entstandene Authentifizierungs- und Autorisierungsinfrastruktur (AAI) [40] eingebunden, die auf dem SAML Standard [15] basiert und föderiertes Identitätsmanagement erlaubt. Nicht nur registrierte DARIAH Nutzer können auf den Speicher zugreifen, sondern auch jeder Nutzer, der sich mit Hilfe seines Campusaccounts in einer Föderation authentifizieren kann und der für den Dienst autorisiert wurde. Allerdings werden innerhalb der Campusverwaltung keine Information wie beispielsweise

dienstspezifische Rollen gespeichert, die für eine Autorisierungsentscheidung herangezogen werden können. Aus diesem Grund wurde das Konzept einer Virtuellen Organisation [35] durch die Nutzung von SAML 2.0 Attributabfragen und Attributaggregation des Shibboleth Service Provider (SP) für die SAML-basierte Infrastruktur angepasst.

Die SPs sammeln zunächst Authentifizierungsinformationen und eine persistente ID vom Campus Identity Proviver (IdP). Danach werden zusätzliche Autorisierungsattribute vom DARIAH IdP abgefragt. Auf der IdP Seite werden Rechte mit Hilfe von Gruppenmitgliedschaften im LDAP-basierten [102] Backend verwaltet und zur Verfügung gestellt. Auf Basis aller Informationen, bereitgestellt durch die Aggregation der verschiedenen Attribute, kann schließlich über die Autorisierung der Nutzer für verschiedene Operationen entschieden werden. Die aktuelle Gruppenverwaltung vergibt globale Lese- und/oder Schreibrechte. Wenn einzelne Datenobjekte von der AAI abgesichert werden sollen, kann ein externer Policy Decision Point (PDP) verwendet werden. Einzelne Dienste agieren in diesem Fall als Policy Enforcement Point (PEP) und können beim PDP abfragen, ob ein bestimmter Nutzer für eine bestimmte Operation auf einem bestimmten Objekt berechtigt ist. Abgeleitet aus diesem Ablauf werden in der Architektur zur archivalischen Speicherung aktuell die zwei folgenden Autorisierungsmodelle unterstützt.

OpenStorage beschreibt das Modell, dass ein Nutzer seine Forschungsdaten ablegt und alle weiteren authentifizierten Nutzer auf diese Daten zugreifen können.

OwnStorage beschreibt das Modell, dass ein Nutzer seine Forschungsdaten ablegt und nur er selbst auf diese Daten zugreifen kann.

Innerhalb der Architektur werden für die Interaktionen mit der Bit Preservation mit dem Nutzer und dem Administrator zwei unterschiedliche Rollen verwendet. Die Rollen bauen hierarchisch aufeinander auf, das heißt jede Operation, die der Nutzer ausführen darf, ist auch dem Administrator erlaubt.

Nutzer Der Nutzer verwendet die Architektur zur archivalischen Speicherung heterogener Forschungsdaten, um seine Daten langfristig und sicher abzulegen. Zu diesem Zweck sollen Einstellungen geändert werden und Informationen über die Daten abgerufen werden können.

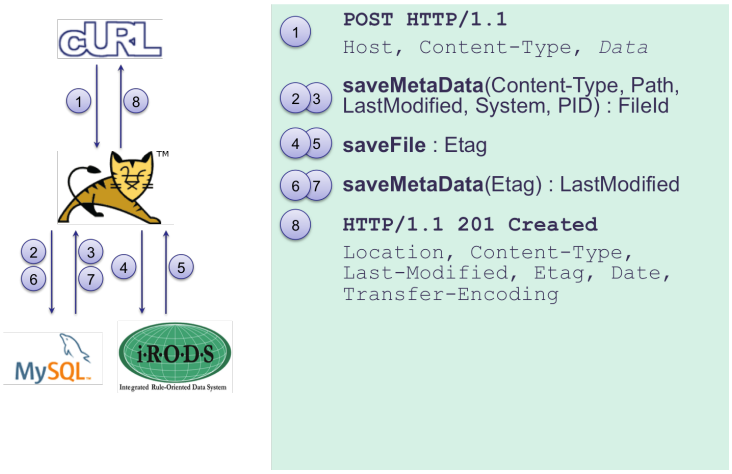

Abbildung A.1: Implementierung von POST.

Administrator Der Administrator ist für die Daten in der Architektur zur archivalischen Speicherung heterogener Forschungsdaten verantwortlich. Zu diesem Zweck ist er in der Lage, zusätzliche Datenintegritätsüberprüfungen auszulösen und sich Informationen über die Mechanismen zur Bit Preservation anzeigen zu lassen.

A.4 Referenzimplementierung

Entsprechend der sechs in Kapitel A.1 beschriebenen HTTP-Methoden stellt die Implementierung sechs Methoden mit den Annotationen @POST, @PUT, @GET, @HEAD, @DELETE und @OPTIONS bereit, um die Funktionalitäten abzudecken. Die Annotation @PATH spezifiert den relativen Pfad für jede Methode und legt auf diese Weise fest, für welche URLs sie bereitgestellt wird.

Die POST-Methode (Abbildung A.1) erzeugt eine neue Datei auf dem Server. Administrative Metadaten werden in der Datenbank gespeichert:

- der mit der Anfrage mitgelieferte Content Type der Datei,
- das genutzte System zur Speichervirtualisierung,
- der logische Dateipfad basierend auf dem für die Datei genutzten System zur Speichervirtualisierung,

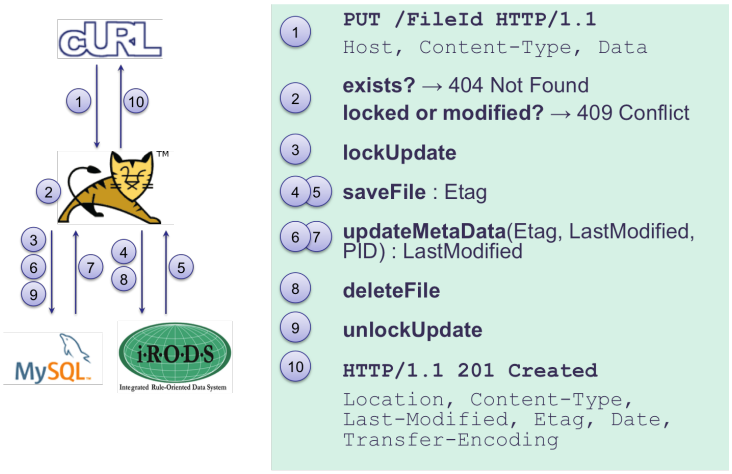

Abbildung A.2: Implementierung von PUT.

- ein Suffix für das Aktualisieren der Datei,
- die aktuelle Systemzeit als Zeitpunkt der letzten Änderung der Datei,
- der persistente Identifier, falls vorhanden, und
- eine Markierung zum Blockieren der Datei im Falle einer Aktualisierung (wird als Standardwert auf falsch gesetzt).

Die Datenbank erzeugt einen einzigartigen Identifier, der als Dateiname für die zu speichernde Datei genutzt wird. Die Datei wird im System zur Speichervirtualisierung abgelegt. An dieser Stelle wird eine Ordnerstruktur bestehend aus Jahr, Monat und Tag genutzt, um eine hierarchische Struktur zur Verbesserung der Performanz des Speicherzugriffs zu schaffen. Grundsätzlich wären auch Alternativen wie die Verwendung des Pairtree-Verfahrens [50] denkbar. Die Speichervirtualisierung berechnet eine Prüfsumme zur Zeit basierend auf dem MD5-Algorithmus. Die Prüfsumme wird zusätzlich als ETag zur Feststellung von Veränderungen der Datei verwendet. Am Schluss wird „201 - Created“ inklusive der spezifizierten Header-Felder an den Nutzer zurückgegeben.

Die PUT-Methode (Abbildung A.2) aktualisiert eine Datei auf dem Server. Existiert die Datei auf dem Server nicht, wird „404 - Not Found“ zurückgegeben. Wurde die Datei bereits vor der Anfrage blockiert oder geändert,

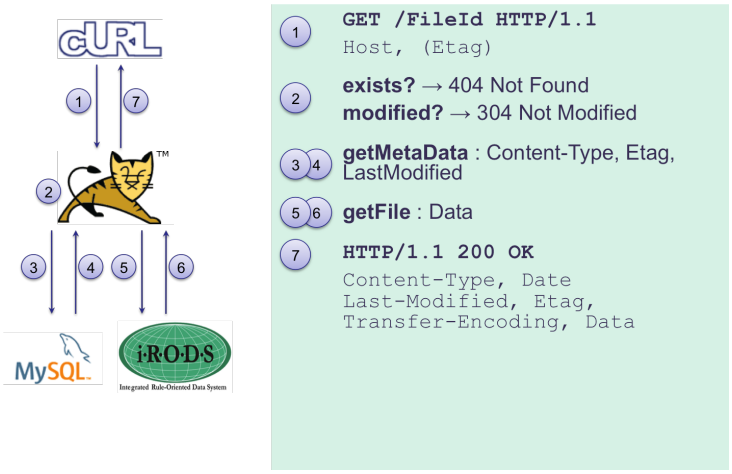

Abbildung A.3: Implementierung von GET.

wird „409 - Conflict“ an den Nutzer gesendet. Nach den Überprüfungen wird die Datei mit Hilfe der Markierung in der Datenbank blockiert, so dass keine konkurrente Aktualisierung oder ein konkurrentes Löschen stattfinden kann. Die Datei wird mit einem geänderten Suffix im System zur Speichervirtualisierung abgelegt. Ist der Transfer erfolgreich, werden die relevanten administrativen Metadaten (Suffix, Zeit der letzten Änderung, ETag und, falls vorhanden, persistenter Identifier) aktualisiert. Eine GET-Anfrage gibt ab diesem Zeitpunkt die neue Datei zurück. Die alte Version wird aus dem Speicher gelöscht und die Blockierung aufgehoben. Schließlich wird „201 - Created“ inklusive der spezifierten Header-Felder an den Nutzer zurückgegeben. Im Falle eines Fehlers während der Aktualisierung wird die Blockierung der Datei automatisch aufgehoben und dem Nutzer so die Möglichkeit gegeben, die Datei erneut zu aktualisieren.

Die GET-Methode (Abbildung A.3) lädt eine Datei vom Server herunter. Zuerst wird die Existenz der Datei auf dem Server überprüft. Ist sie nicht vorhanden, wird „404 - Not Found“ zurückgegeben. Eine weitere Überprüfung stellt mit Hilfe des ETags oder des Zeitpunkts der letzten Änderung fest, ob die Datei seit dem letzten Herunterladen verändert wurde. Falls keine Aktualisierung stattgefunden hat, wird „304 - Not Modified“ gesendet und die Datei wird nicht übertragen. Ansonsten werden die Daten und die Metadaten aus dem System zur Speichervirtualisierung bzw. der Datenbank dem Nutzer mit „200 - OK“ zur Verfügung gestellt.

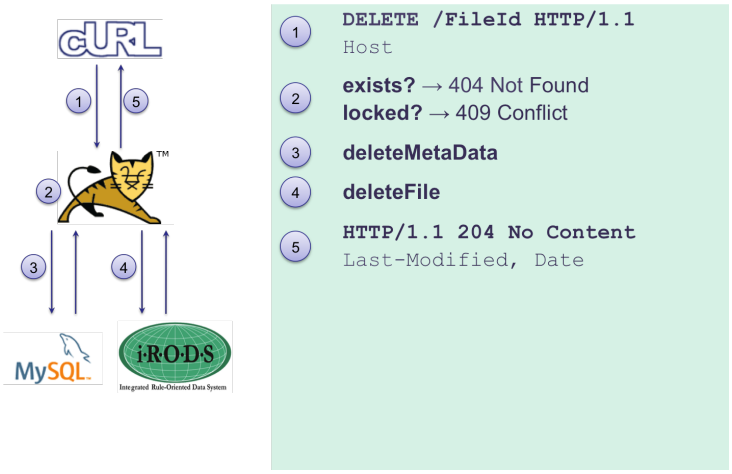

Abbildung A.4: Implementierung von DELETE.

Die HEAD-Methode lädt die Metadaten einer Datei vom Server herunter. Die Implementierung entspricht der GET-Methode mit dem einzigen Unterschied, dass die Datei nicht aus dem System zur Speichervirtualisierung geholt und dem Nutzer nicht zur Verfügung gestellt wird.

Die DELETE-Methode (Abbildung A.4) löscht eine Datei auf dem Server. Zunächst wird überprüft, ob die Datei auf dem Server existiert, falls nicht wird „404 - Not Found“ zurückgegeben. Durch diese Abfrage wird auf die Idempotenz von DELETE verzichtet. Wird die Datei zur Zeit aktualisiert, wird die Meldung „409 - Conflict“ gesendet. Ansonsten werden die administrativen Metadaten in der Datenbank und die Datei im System zur Speichervirtualisierung gelöscht und „204 - No Content“ wird mit den spezifierten Header-Feldern an den Nutzer gesendet.

Die OPTIONS-Methode liefert einen Überblick über die implementierten Methoden in der DARIAH Bit Preservation. Auf Grund der standardmäßigen Bereitstellung der Methode in der genutzten Javabibliothek ist keine weitere Implementierung nötig.

Entsprechend der in Kapitel A.2 beschriebenen HTTP-Methoden der Admin API stellt die Implementierung zwei Methoden mit den Annotationen @GET und @PUT bereit, um die HTTP-Funktionalitäten abzudecken. Durch die Annotation @PATH werden die beiden Methoden entsprechend der Spezifikation unter dem relativen Pfad `/admin` zur Verfügung gestellt.

Die GET-Methode bietet dem Nutzer Informationen über die Datei auf

dem Server. Zunächst werden alle in der Datenbank gespeicherten Informationen zur Bit Preservation abgerufen. Danach wird der Pfad der aufgerufenen Methode überprüft. Besitzt der Pfad keinerlei Erweiterung, werden alle Informationen in der resultierenden XML-Datei verwendet. Stimmt der Pfad mit einer der spezifizierten Erweiterung überein, werden nur diese Informationen in der resultierenden XML-Datei verwendet, alle anderen werden ausgelassen. Stimmt der Pfad mit keiner Erweiterung überein, wird der Fehler „400 - Bad Request“ ausgelöst. Abschließend wird „200 - OK“ inklusive der angefragten Informationen per XML-Datei an den Nutzer gesendet.

Die PUT-Methode bietet Funktionalitäten, um Parameter der Bit Preservation zu ändern. Die nach Spezifikation mitzuliefernde XML-Datei wird geparst und die Informationen ausgelesen. Soll das Bit Preservation Level geändert werden, werden die von der anbietenden Institution definierten Werte für die Anzahl der Repliken, die Lokation der Repliken, das Intervall zwischen zwei Datenintegritätsüberprüfungen und den verwendeten Algorithmus in der Datenbank angepasst. Zusätzlich wird in diesem Schritt die Datei entweder entsprechend der geforderten Anzahl der Repliken repliziert (bzw. gelöscht) falls das Bit Preservation Level erhöht (bzw. verringert) wird. Soll die Zeit bis zu einer möglichen Verschiebung auf Offline- oder Nearlinespeicher verändert werden, werden ebenfalls die Werte in der Datenbank verändert. Ein unabhängiger Prozess kann auf diese Weise nach verschiebbaren Dateien suchen und eine Migration, wie in Kapitel 3.3 beschrieben, durchführen. Soll eine zusätzliche Integritätsüberprüfung durchgeführt werden, wird die entsprechende Datei in der Datenbank markiert. Ein Cron Job überprüft mit Hilfe der Markierung täglich, ob Dateien kontrolliert werden sollen, und führt ggf. die Neuberechnung der Prüfsumme durch. Entspricht die Anfrage keiner der spezifizierten Fälle, so wird der Fehler „400 - Bad Request“ ausgelöst. Sonst wird „200 - OK“ an den Nutzer gesendet.

Eine weitere Möglichkeit für eine zusätzliche Sicherungsebene ist die Verwendung der persistenten Identifier für die Kennzeichnung der einzelnen Dateien. Sollte die Speichervirtualisierungskomponente ausfallen, können die Daten mit Hilfe der gespeicherten administrativen Metadaten und des Identifiers wiederhergestellt werden.

Neben den beschriebenen HTTP-Methoden zur Bereitstellung der grundlegenden Speicherfunktionalitäten stellt die Referenzimplementierung zusätzliche Funktionalitäten und Methodiken zur Bit Preservation bereit. Basierend auf der Einteilung in Kapitel 2 kann für die Referenzimplementierung zusammengefasst werden:

Haltbarkeit der Speichermedien Die DARIAH Bit Preservation ist in der Lage, durch die Nutzung von Speicheradaptern unterschiedliche Medien mit variierender Haltbarkeit unterstützen.

Anzahl der Kopien In der DARIAH Bit Preservation wird die Anzahl der Kopien durch das von der anbietenden Institution spezifizierte Bit Preservation festgelegt.

Verteilung / Unabhängigkeit der Kopien Die innerhalb der DARIAH Bit Preservation verwendeten Speicherkomponenten können an verschiedenen Lokationen angeboten werden, die Verantwortlichkeit für die Verteilung liegt bei der anbietenden Institution. Zusätzlich kann durch die Verwendung verschiedener Speicherkomponenten eine gewisse Unabhängigkeit der Kopien gewährleistet werden.

Regelmäßige Integritätsüberprüfungen Alle Dateien werden entsprechend des per Bit Preservation Level spezifizierten Intervalls überprüft. Ebenfalls können zusätzliche Datenintegritätsüberprüfungen jederzeit ausgelöst werden, die von einem Cron Job täglich ausgeführt werden.

A.5 Szenario 1: Synchrone Replikation aus Systemsicht

# Repliken	Ausweichen 0
1	0.000030518276588509252185100055242924143993363400 9
2	0.000045776832799049695722527298133033909927963783 0
3	0.000061035156178946919069539964829534813903465593 4
4	0.000076293246731753690107602479299928218534873863 9
5	0.000091551104461022722506416393610439461716060852 7
# Repliken	Ausweichen 1
2	0.000030518975059123227017895894627581309772306006 3
3	0.000045777531259005692167445140232952515974400952 2
4	0.000061035854628245099756905329062235471903289343 2
5	0.000076293945170394217665259305385919448136835457 0
# Repliken	Ausweichen 2
3	0.000045777298449677940876698834842856165291006607 8
4	0.000061035621822469791094416920706584195090965655 0
5	0.000076293712368171297424228967961491334840329841 1
# Repliken	Ausweichen 3
4	0.000061035621818917511081595860063200342028209040 28
5	0.000076293712364619071615726380911057508020617815 75
# Repliken	Ausweichen 4
5	0.000076293712364619017413889146242808921134778409 37

Tabelle A.1: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514782149420752371521508767787614390
3	0.00000000232830643420766161948084291488200832567970
4	0.00000000279396772037698859739525360887954419998814
5	0.00000000325962900632947514136792149272523108371780
# Repliken	Ausweichen 1
2	0.00000000162981805736918691169484638654292331811247
3	0.00000000209547934386377288629948936798275787938312
4	0.00000000256114063014151842681092901618117850607897
5	0.00000000302680191620242353333013952680664031107515
# Repliken	Ausweichen 2
3	0.00000000209547934380956443217527414927783529711821
4	0.00000000256114063008730997271195657600376186367410
5	0.00000000302680191614821507925640986514497502378735
# Repliken	Ausweichen 3
4	0.00000000256114063008730997269933537933447184841707
5	0.00000000302680191614821507924378866848156221121802
# Repliken	Ausweichen 4
5	0.00000000302680191614821507924378866847862365471537

Tabelle A.2: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.000000000931329679825989064621750068587203476167 93
2	0.000000001164163875976691269292787296712390828353 09
3	0.000000001396998072073181711016191831666029057742 37
4	0.000000001629832268115460389804586025707698216037 56
5	0.000000001862666464103527305670592231094039439695 82
# Repliken	Ausweichen 1
2	0.000000001164160323317227540461940268594318005779 54
3	0.000000001396994519413718809365956008412915944012 13
4	0.000000001629828715455998315334961214723611950449 06
5	0.000000001862662911444066058381578239783047206390 41
# Repliken	Ausweichen 2
3	0.000000001396994519413718809353334619135939245597 07
4	0.000000001629828715455998315322339825449573943063 35
5	0.000000001862662911444066058368956850511947890033 36
# Repliken	Ausweichen 3
4	0.000000001629828715455998315322339825449573898223 85
5	0.000000001862662911444066058368956850511947845193 86
# Repliken	Ausweichen 4
5	0.000000001862662911444066058368956850511947845193 86

Tabelle A.3: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514782149420752371521508767787614390
3	0.00000000232830643420766161948084291488200832567970
4	0.00000000279396772037698859739525360887954419998814
5	0.00000000325962900632947514136792149272523108371780
# Repliken	Ausweichen 1
2	0.00000000162981450470972566433010290404737057222383
3	0.00000000209547579120431329327072378336442750751962
4	0.00000000256113707748206048811814055907984949048916
5	0.00000000302679836354296724897332742686209199273450
# Repliken	Ausweichen 2
3	0.00000000209547579115010318482315340417789149079525
4	0.00000000256113707742785037969581372877516251123197
5	0.00000000302679836348875714057624414542749910747735
# Repliken	Ausweichen 3
4	0.00000000256113707742785037968319195434452356803112
5	0.00000000302679836348875714056362237100273763600529
# Repliken	Ausweichen 4
5	0.00000000302679836348875714056362237099979890014329

Tabelle A.4: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514782149420752371521508767787614390
3	0.00000000232830643420766161948084291488200832567970
4	0.00000000279396772037698859739525360887954419998814
5	0.00000000325962900632947514136792149272523108371780
# Repliken	Ausweichen 1
2	0.00000000162981450465551638300003447229257175092339
3	0.00000000209547579115010401196589851533487705692191
4	0.00000000256113707742785120683855845476379264647952
5	0.00000000302679836348875796771898848624777399120377
# Repliken	Ausweichen 2
3	0.00000000209547579109589390349308497244072486050696
4	0.00000000256113707737364109839098846076324425163821
5	0.00000000302679836343454785929666204112907445446345
# Repliken	Ausweichen 3
4	0.00000000256113707737364109837836668632378923536368
5	0.00000000302679836343454785928404026669549690992183
# Repliken	Ausweichen 4
5	0.00000000302679836343454785928404026669255817405709

Tabelle A.5: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139696254493642431176528182869237733977946
2	0.00000000186261672611208531021219904511674453144974
3	0.00000000232827090707090918323786987955921044347258
4	0.00000000279392508781289593094326698696778761514622
5	0.00000000325957926833804555342936302224346952928843
# Repliken	Ausweichen 1
2	0.00000000162979318831923926174807369906754271930696
3	0.00000000209544736938648004308528191727829036903950
4	0.00000000256110155023688369905173085137075093682686
5	0.00000000302675573087045022974839315626942707498642
# Repliken	Ausweichen 2
3	0.00000000209544736933227241611643358418156335749270
4	0.00000000256110155018267607210812491162729696252261
5	0.00000000302675573081624260283002960986749173254248
# Repliken	Ausweichen 3
4	0.00000000256110155018267607209550390754218464028950
5	0.00000000302675573081624260281740860578825652331807
# Repliken	Ausweichen 4
5	0.00000000302675573081624260281740860578531801165424

Tabelle A.6: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation a+b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

A.6 Vergleich der Replikationsszenarien aus Systemsicht

# Repliken	Szenario 1
1	0.000030518276588509252185100055242924143993363400 94
2	0.000045776832799049695722527298133033909927963783 01
3	0.000061035156178946919069539964829534813903465593 45
4	0.000076293246731753690107602479299928218534873863 93
5	0.000091551104461022722506416393610439461716060852 78
# Repliken	Szenario 2
1	0.000030518276588509252185100055242924143993363400 94
2	0.000061035156178946973276339866658396198811949659 03
3	0.000091551104461022830916707651525874040299182511 04
4	0.000122066121463158317621727788459885546092714354 03
5	0.000152580207213774058545189393951585166503078033 84

Tabelle A.7: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 1
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514782149420752371521508767787614390
3	0.00000000232830643420766161948084291488200832567970
4	0.00000000279396772037698859739525360887954419998814
5	0.00000000325962900632947514136792149272523108371780
# Repliken	Szenario 2
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000232830643426187172797890039187338471950229
3	0.00000000325962900643789535826306225110711932476898
4	0.00000000419095157774655725308317545661833621180561
5	0.00000000512227414818785741324703357276726390123136

Tabelle A.8: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 1
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416387597669126929278729671239082835309
3	0.00000000139699807207318171101619183166602905774237
4	0.00000000162983226811546038980458602570769821603756
5	0.00000000186266646410352730567059223109403943969582
# Repliken	Szenario 2
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000139699807212739181956473563613141145216957
3	0.00000000186266646421194752271719197179215226740437
4	0.00000000232833485607965617418009789364606525923194
5	0.00000000279400324773051777405443221972276712109787

Tabelle A.9: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

A.7 Szenario 5+6: Asynchrone Replikation aus Nutzersicht

# Repliken	Ausweichen 0
1	0.000030518276588509252185100055242924143993363400 94
2	0.000045776716383728112730719960950660171402656529 98
3	0.000050862862982134399579259929519905513872420906 33
4	0.000053405936281337543003529913804528185107303094 50
5	0.000054931780260859429058091904375301787848232407 41
# Repliken	Ausweichen 1
2	0.000030518858642025314295878421204206274776845675 77
3	0.000035605082847685458322276333991494970960036291 67
4	0.000038148194950515530335475290385139319051631599 63
5	0.000039674062212213573543394664221325927906588784 40
# Repliken	Ausweichen 2
3	0.000035604850035989357738122068192777104110761003 75
4	0.000038147962139411517074673017843731567779317345 68
5	0.000039673829401464812676603587634304245980451150 85
# Repliken	Ausweichen 3
4	0.000038147962135859155753513219807840538267061436 56
5	0.000039673829397912456776082418857742561291114166 72
# Repliken	Ausweichen 4
5	0.000039673829397912402572260167809111178259332152 34

Tabelle A.10: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 5 bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000201786557658074327246023236585045231059715
4	0.00000000209547579100102538633989937235618541282343
5	0.00000000214204191965319465466769957625962527415920
# Repliken	Ausweichen 1
2	0.00000000162981805728787174885309715148927517872468
3	0.00000000178503848616457549750545424327030025982104
4	0.00000000186264870060292737183163278916081280036921
5	0.00000000190921482926593849642733991669512032469812
# Repliken	Ausweichen 2
3	0.00000000178503848611036704336441050553724489702344
4	0.00000000186264870054871891769479618118552529425140
5	0.00000000190921482921173004229302758657449353258817
# Repliken	Ausweichen 3
4	0.00000000186264870054871891768217498450741947495717
5	0.00000000190921482921173004228040638989697543356343
# Repliken	Ausweichen 4
5	0.00000000190921482921173004228040638989403687705750

Tabelle A.11: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 5 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.000000000931329679825989064621750068587203476167 93
2	0.000000001164163875976690442093242852359307464168 75
3	0.000000001241775274693590901250407113616675460169 02
4	0.000000001280580974052041130828989244245359458169 16
5	0.000000001303864393667111268576138522622569856969 24
# Repliken	Ausweichen 1
2	0.000000001164160323317226713262395824238295883301 82
3	0.000000001241771722034127448146430551314981831333 57
4	0.000000001280577421392577815588447914853324805349 44
5	0.000000001303860841007648036053658332976330589758 97
# Repliken	Ausweichen 2
3	0.000000001241771722034127448133809162036046005565 38
4	0.000000001280577421392577815575826525574878761419 71
5	0.000000001303860841007648036041036943698178414932 31
# Repliken	Ausweichen 3
4	0.000000001280577421392577815575826525574878716580 21
5	0.000000001303860841007648036041036943698178370092 81
# Repliken	Ausweichen 4
5	0.000000001303860841007648036041036943698178370092 81

Tabelle A.12: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 5 bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000201786557658074327246023236585045231059715
4	0.00000000209547579100102538633989937235618541282343
5	0.00000000214204191965319465466769957625962527415920
# Repliken	Ausweichen 1
2	0.00000000162981450462841050148835338010863934939535
3	0.00000000178503493350511480158603659834045131951265
4	0.00000000186264514794346695163487820745635730457129
5	0.00000000190921127660647824166418317292590089560648
# Repliken	Ausweichen 2
3	0.00000000178503493345090469312163718655622015824732
4	0.00000000186264514788925684317468605382228461340808
5	0.00000000190921127655226813320651537418192328650453
# Repliken	Ausweichen 3
4	0.00000000186264514788925684316206427938282946260839
5	0.00000000190921127655226813319389359974305588287844
# Repliken	Ausweichen 4
5	0.00000000190921127655226813319389359974011714701316

Tabelle A.13: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 5 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000201786557658074327246023236585045231059715
4	0.00000000209547579100102538633989937235618541282343
5	0.00000000214204191965319465466769957625962527415920
# Repliken	Ausweichen 1
2	0.00000000162981450457420122015828494834943249154883
3	0.00000000178503493345090552026438255449210949442506
4	0.00000000186264514788925767031743135756344799586317
5	0.00000000190921127655226896034926063940625109672604
# Repliken	Ausweichen 2
3	0.00000000178503493339669541177473997899242564406031
4	0.00000000186264514783504756183199604021588174295308
5	0.00000000190921127649805885186634967694995540228875
# Repliken	Ausweichen 3
4	0.00000000186264514783504756181937426576761051907356
5	0.00000000190921127649805885185372790250227192558324
# Repliken	Ausweichen 4
5	0.00000000190921127649805885185372790249933318971521

Tabelle A.14: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 5 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.000030518276588509252185100055242924143993363400 94
2	0.000045776716383728112730719960950660171402656529 98
3	0.000061034845742826352126049191142398127701498523 67
4	0.000076292664672909343499968840471769982299302481 26
5	0.000091550173181082286509012951167733019140057591 77
# Repliken	Ausweichen 1
2	0.000030518858642025314295878421204206274776845675 77
3	0.000045777220818148318222987196744884086180529649 12
4	0.000061035272560519375544627514228286128099104488 92
5	0.000076293013876243794332421888432094989527675724 71
# Repliken	Ausweichen 2
3	0.000045776988008820494655404924474671844732230183 96
4	0.000061035039754743931365139540612365425674152201 25
5	0.000076292781074020657267500789805086412170260957 99
# Repliken	Ausweichen 3
4	0.000061035039751191651350250686211224042529823784 46
5	0.000076292781070468431455689783225833199532261145 86
# Repliken	Ausweichen 4
5	0.000076292781070468377253852498075879205337833895 30

Tabelle A.15: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 6 bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000232830643397275114922161965226169717901938
4	0.00000000279396771991620267518700860335085693721593
5	0.00000000325962900557053362279901359723413833001902
# Repliken	Ausweichen 1
2	0.00000000162981805728787174885309715148927517872468
3	0.00000000209547934362886241598557258395333389299182
4	0.00000000256114062968073250449540056486353136049209
5	0.00000000302680191544348201458452948539455958692562
# Repliken	Ausweichen 2
3	0.00000000209547934357465396186135736523567717724691
4	0.00000000256114062962652405039642812466113622550395
5	0.00000000302680191538927356051079982369175325304972
# Repliken	Ausweichen 3
4	0.00000000256114062962652405038380692799184621024110
5	0.00000000302680191538927356049817862702834044047081
# Repliken	Ausweichen 4
5	0.00000000302680191538927356049817862702540188396817

Tabelle A.16: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 6 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416387597669044209324285235930746416875
3	0.00000000139699807205510946896789255883692239858062
4	0.00000000162983226806124614527094389253920811374345
5	0.00000000186266646399510047102764155797591991521434
# Repliken	Ausweichen 1
2	0.00000000116416032331722671326239582423829588330182
3	0.00000000139699451939564656731765667137928746641728
4	0.00000000162982871540178407080131888895037271257871
5	0.00000000186266291133563922373862718146130701702903
# Repliken	Ausweichen 2
3	0.00000000139699451939564656730503528210231076800200
4	0.00000000162982871540178407078869749967633470519231
5	0.00000000186266291133563922372600579219020770067061
# Repliken	Ausweichen 3
4	0.00000000162982871540178407078869749967633466035281
5	0.00000000186266291133563922372600579219020765583111
# Repliken	Ausweichen 4
5	0.00000000186266291133563922372600579219020765583111

Tabelle A.17: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 6 bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponenten und 2^{-16} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000232830643397275114922161965226169717901938
4	0.00000000279396771991620267518700860335085693721593
5	0.00000000325962900557053362279901359723413833001902
# Repliken	Ausweichen 1
2	0.00000000162981450462841050148835338010863934939535
3	0.00000000209547579096940282295680616477809725414773
4	0.00000000256113707702127456580261047074673315300953
5	0.00000000302679836278402573022771468918923976913474
# Repliken	Ausweichen 2
3	0.00000000209547579091519271450923578557882671532548
4	0.00000000256113707696706445738028364041706691888017
5	0.00000000302679836272981562183063140771350458175481
# Repliken	Ausweichen 3
4	0.00000000256113707696706445736766186598642797567350
5	0.00000000302679836272981562181800963328874311027317
# Repliken	Ausweichen 4
5	0.00000000302679836272981562181800963328580437441117

Tabelle A.18: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 6 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-32} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

# Repliken	Ausweichen 0
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000232830643397275114922161965226169717901938
4	0.00000000279396771991620267518700860335085693721593
5	0.00000000325962900557053362279901359723413833001902
# Repliken	Ausweichen 1
2	0.00000000162981450457420122015828494834943249154883
3	0.00000000209547579091519354165198089673581247575664
4	0.00000000256113707696706528452302836640569743526344
5	0.00000000302679836272981644897337574853378009323420
# Repliken	Ausweichen 2
3	0.00000000209547579086098343317916735382892575723789
4	0.00000000256113707691285517607545837238016978553832
5	0.00000000302679836267560634055104930337393825435195
# Repliken	Ausweichen 3
4	0.00000000256113707691285517606283659794071476925798
5	0.00000000302679836267560634053842752894036070980075
# Repliken	Ausweichen 4
5	0.00000000302679836267560634053842752893742197393601

Tabelle A.19: Wahrscheinlichkeit eines unentdeckten Fehlers in Szenario 6 bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponenten und 2^{-48} der verwendeten Prüfsumme (Lokation b) in Abhängigkeit der Anzahl der Repliken und der Anzahl der Ausweichschritte.

A.8 Vergleich der Replikationsszenarien aus Nutzersicht

# Repliken	Szenario 4
1	0.000030518276588509252185100055242924143993363400 94
2	0.000030518276588509252185100055242924143993363400 94
3	0.000030518276588509252185100055242924143993363400 94
4	0.000030518276588509252185100055242924143993363400 94
5	0.000030518276588509252185100055242924143993363400 94
# Repliken	Szenario 5
1	0.000030518276588509252185100055242924143993363400 94
2	0.000045776716383728112730719960950660171402656529 98
3	0.000050862862982134399579259929519905513872420906 33
4	0.000053405936281337543003529913804528185107303094 50
5	0.000054931780260859429058091904375301787848232407 41
# Repliken	Szenario 6
1	0.000030518276588509252185100055242924143993363400 94
2	0.000045776716383728112730719960950660171402656529 98
3	0.000061034845742826352126049191142398127701498523 67
4	0.000076292664672909343499968840471769982299302481 26
5	0.000091550173181082286509012951167733019140057591 77

Tabelle A.20: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 4
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000139698386121848636142289631380458749278686
3	0.00000000139698386121848636142289631380458749278686
4	0.00000000139698386121848636142289631380458749278686
5	0.00000000139698386121848636142289631380458749278686
# Repliken	Szenario 5
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000201786557658074327246023236585045231059715
4	0.00000000209547579100102538633989937235618541282343
5	0.00000000214204191965319465466769957625962527415920
# Repliken	Szenario 6
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000186264514774017904470089835283898610614457
3	0.00000000232830643397275114922161965226169717901938
4	0.00000000279396771991620267518700860335085693721593
5	0.00000000325962900557053362279901359723413833001902

Tabelle A.21: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 4
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000093132967982598906462175006858720347616793
3	0.00000000093132967982598906462175006858720347616793
4	0.00000000093132967982598906462175006858720347616793
5	0.00000000093132967982598906462175006858720347616793
# Repliken	Szenario 5
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416387597669044209324285235930746416875
3	0.00000000124177527469359090125040711361667546016902
4	0.00000000128058097405204113082898924424535945816916
5	0.00000000130386439366711126857613852262256985696924
# Repliken	Szenario 6
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416387597669044209324285235930746416875
3	0.00000000139699807205510946896789255883692239858062
4	0.00000000162983226806124614527094389253920811374345
5	0.00000000186266646399510047102764155797591991521434

Tabelle A.22: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponente ohne Prüfsummenberechnung in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 4
1	0.000030518276588509252185100055242924143993363400094
2	0.000015260186025044882216684682842340186227273828 04
3	0.000015259953208612083045714470985141394762310268 23
4	0.000015259953205059640414975234059310192906270484 07
5	0.000015259953205059586209829605101592913899343568 84
# Repliken	Szenario 5
1	0.000030518276588509252185100055242924143993363400094
2	0.000030518858642025314295878421204206274776845675 77
3	0.000035604850035989357738122068192777104110761003 75
4	0.000038147962135859155753513219807840538267061436 56
5	0.000039673829397912402572260167809111178259332152 34
# Repliken	Szenario 6
1	0.000030518276588509252185100055242924143993363400094
2	0.000030518858642025314295878421204206274776845675 77
3	0.000045776988008820494655404924474671844732230183 96
4	0.000061035039751191651350250686211224042529823784 46
5	0.000076292781070468377253852498075879205337833895 30

Tabelle A.23: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-16} der Speicherkomponente und 2^{-16} der Prüfsumme (Lokation b) bei maximaler Anzahl Ausweichschritte in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 4
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000116415677065776050289602587614619993543562
3	0.00000000116415677060355204872132510035100171919981
4	0.00000000116415677060355204870870390366408009586325
5	0.00000000116415677060355204870870390366114153935514
# Repliken	Szenario 5
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000162981805728787174885309715148927517872468
3	0.00000000178503848611036704336441050553724489702344
4	0.00000000186264870054871891768217498450741947495717
5	0.00000000190921482921173004228040638989403687705750
# Repliken	Szenario 6
1	0.00000000139698386121848636142289631380458749278686
2	0.00000000162981805728787174885309715148927517872468
3	0.00000000209547934357465396186135736523567717724691
4	0.00000000256114062962652405038380692799184621024110
5	0.00000000302680191538927356049817862702540188396817

Tabelle A.24: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-32} der Speicherkomponente und 2^{-16} der Prüfsumme (Lokation b) bei maximaler Anzahl Ausweichschritte in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

# Repliken	Szenario 4
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000093132612716652450861029164300823803920657
3	0.00000000093132612716652450859767025372538395873077
4	0.00000000093132612716652450859767025372538391389127
5	0.00000000093132612716652450859767025372538391389127
# Repliken	Szenario 5
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416032331722671326239582423829588330182
3	0.00000000124177172203412744813380916203604600556538
4	0.00000000128057742139257781557582652557487871658021
5	0.00000000130386084100764803604103694369817837009281
# Repliken	Szenario 6
1	0.00000000093132967982598906462175006858720347616793
2	0.00000000116416032331722671326239582423829588330182
3	0.00000000139699451939564656730503528210231076800200
4	0.00000000162982871540178407078869749967633466035281
5	0.00000000186266291133563922372600579219020765583111

Tabelle A.25: Wahrscheinlichkeit eines unentdeckten Fehlers bei einer Fehlerwahrscheinlichkeit von 2^{-48} der Speicherkomponente und 2^{-16} der Prüfsumme (Lokation b) bei maximaler Anzahl Ausweichschritte in Abhängigkeit der Anzahl der Repliken und des Replikationsszenarios.

A.9 Auswirkungen steigender Datenmengen

# Dateien	S4 - R1
1	0.00000000139698386121848636142289631380458749278686
10	0.00000001396983852436448805765461219475628184168896
100	0.00000013969837646160772987610809832097398034727719
1000	0.00000139698288641276708459890051357550599904356394
10000	0.00001396974104420150389886779821370335306261178866
# Dateien	S4 - R2
1	0.00000000116415677065776050289602587614619993543562
10	0.00000001164156764559086081947700023678259924738296
100	0.00000011641567035723442141405637728499707941037484
1000	0.00000116415609370515982982758624681069386962270665
10000	0.00001164149995056745294415985754960470677836983473
# Dateien	S4 - R3
1	0.00000000116415677060355204872132510035100171919981
10	0.00000001164156764504877628252614156890207813031467
100	0.00000011641567035181357661251200673379225866627262
1000	0.00000116415609365095143860853142470097761392929809
10000	0.00001164149995002537471157546843791329718529673610

Tabelle A.26: Wahrscheinlichkeit eines unentdeckten Fehlers in Abhängigkeit des Replikations**S**zenarios, der Anzahl der **R**epliken und der Anzahl der gespeicherten Dateien.

# Dateien	S4 - R4
1	0.00000000116415677060355204870870390366408009586325
10	0.0000000116415676450487762825261415689020781303 1467
100	0.000000116415670351813576612512006733792258666272 62
1000	0.00000116415609365095143860853142470097761392929809
10000	0.0000116414999500253747115754684379132971852967361 0
# Dateien	S4 - R5
1	0.00000000116415677060355204870870390366114153935514
10	0.0000000116415676450487762825261415689020781303 1467
100	0.000000116415670351813576612512006733792258666272 62
1000	0.00000116415609365095143860853142470097761392929809
10000	0.0000116414999500253747115754684379132971852967361 0
# Dateien	S5 - R1
1	0.00000000139698386121848636142289631380458749278686
10	0.0000000139698385243644880576546121947562818416889 6
100	0.000000139698376461607729876108098320973980347277 19
1000	0.00000139698288641276708459890051357550599904356394
10000	0.0000139697410442015038988677982137033530626117886 6
# Dateien	S5 - R2
1	0.00000000162981805728787174885309715148927517872468
10	0.0000000162981804533449075124438225423714789770929 0
100	0.000000162981792580068720599315434991670646422700 80
1000	0.00000162981673046329465378891684895356347730668551
10000	0.0000162980477715365890086084505545759630437200814 9

Tabelle A.27: Wahrscheinlichkeit eines unentdeckten Fehlers in Abhängigkeit des Replikations**S**zenarios, der Anzahl der **R**epliken und der Anzahl der gespeicherten Dateien.

# Dateien	S5 - R3
1	0.00000000178503848611036704336441050553724489702344
10	0.00000001785038471771736256249580504693824932819131
100	0.00000017850383283854375248517093097869086320626562
1000	0.00000178503689452329484571397972202540078003422458
10000	0.00001785022555986331161886728600056776057442563885
# Dateien	S5 - R4
1	0.00000000186264870054871891768217498450741947495717
10	0.00000001862648684936148177328327505470674210656301
100	0.00000018626485288104503749441795763662946188300907
1000	0.00000186264696755443201485594224152925802289534610
10000	0.00001862631355090214342867149410975397247626225930
# Dateien	S5 - R5
1	0.00000000190921482921173004228040638989403687705750
10	0.00000001909214812808774436938211999654645168307265
100	0.00000019092146487792287228954199478770720952778917
1000	0.00000190921300848480503520895187555106558803433555
10000	0.00001909196605643912939713790649411347267120286458
# Dateien	S6 - R1
1	0.00000000139698386121848636142289631380458749278686
10	0.00000001396983852436448805765461219475628184168896
100	0.00000013969837646160772987610809832097398034727719
1000	0.00000139698288641276708459890051357550599904356394
10000	0.00001396974104420150389886779821370335306261178866

Tabelle A.28: Wahrscheinlichkeit eines unentdeckten Fehlers in Abhängigkeit des Replikations**S**zenarios, der Anzahl der **R**epliken und der Anzahl der gespeicherten Dateien.

# Dateien	S6 - R2
1	0.00000000162981805728787174885309715148927517872468
10	0.0000000162981804533449075124438225423714789770929 0
100	0.000000162981792580068720599315434991670646422700 80
1000	0.00000162981673046329465378891684895356347730668551
10000	0.0000162980477715365890086084505545759630437200 8149
# Dateien	S6 - R3
1	0.00000000209547934357465396186135736523567717724691
10	0.0000000209547932381500237621152071002113961544413 1
100	0.000000209547912621850157366716855351414062634681 74
1000	0.00000209547715025485994350863315962504970547911658
10000	0.0000209545739075508245532734892056962558362766523 1
# Dateien	S6 - R4
1	0.00000000256114062962652405038380692799184621024110
10	0.0000000256114060010903828805324353962151576838783 8
100	0.000000256114030493420563892384945948247094345133 94
1000	0.00000256113735318837389414156842524505825211615387
10000	0.0000256110783597952950174086604116592591660188632 1
# Dateien	S6 - R5
1	0.00000000302680191538927356049817862702540188396817
10	0.0000000302680187416238963061656430320329955772099 4
100	0.000000302680146189359156229375909099088698716560 98
1000	0.00000302679733920972880082939960989618817157307628
10000	0.0000302675611278289024965604846668417199457628524 2

Tabelle A.29: Wahrscheinlichkeit eines unentdeckten Fehlers in Abhängigkeit des Replikations**S**zenarios, der Anzahl der **R**epliken und der Anzahl der gespeicherten Dateien.

Eigene Publikationen

Konferenzbeiträge

- D. Tonne, R. Stotzka, T. Jejkal, V. Hartmann, H. Pasic, A. Rapp, P. Vanscheidt, B. Neumair, A. Streit, A. Garcia, D. Kurzawe, T. Kalman, J. Rybicki, B. Sanchez Bribian, „ Federated Data Zone for the Arts and Humanities“, Seiten 198-205, 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 2012

- A. Rapp, D. Tonne, P. Vanscheidt, „Storage Infrastructure of the Virtual Scriptorium St. Matthias“, Seiten 529-532, Digital Humanities, 2012

- D. Tonne, J. Rybicki, S. E. Funk, P. Gietz, „Access to the DARIAH Bit Preservation Service for Humanities Research Data“, Seiten 9-15, 21th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 2013

- M. Embach, C. Krause, C. Moulin, A. Rapp, F. Rindone, D. Tonne, P. Vanscheidt, „eCodicology - Algorithms for the Automatic Tagging of Medieval Manuscripts“, Seiten 172-178, The Linked TEI: Text Encoding in the Web, 2013

- S. Chandna, D. Tonne, T. Jejkal, R. Stotzka, C. Krause, P. Vanscheidt, H. Busch, A. Prabhune, „Software Workflow for the Automatic Tagging of Medieval Manuscript Images (SWATI)“, SPIE Electronic Imaging 2015

- T. Kalman, D. Tonne, O. Schmitt, „Towards a Sustainable European Preservation Infrastructure for the Arts and Humanities“, 1st Annual Conference on Digital Preservation for the Arts, Humanities, and Social Sciences (DPASSH2015) in New Review of Information Networking (NRIN)

- S. Chandna, D. Tonne, R. Stotzka, P. Vanscheidt, C. Krause, H. Busch, „An Effektive Visualization Technique for Determining Co-relations in High-dimensional Medieval Manuscripts Data“, Visualization and Data Analysis (VDA 2016)

Buchbeiträge

- F. Enders, C. Krause, R. Stotzka, D. Tonne, P. Vanscheidt, „Nach der Digitalisierung - Zur computergestützten Erschließung mittelalterlicher Handschriften“, Digitale Rekonstruktionen mittelalterlicher Bibliotheken (Trierer Beiträge zu den historischen Kulturwissenschaften 12), 2014

- H. Hahn, T. Kalman, W. Kolbmann, T. Kollatz, M. Neuschäfer, S. Pielström, J. Puhl, J. Stiller, D. Tonne, „Handbuch Digital Humanities“, DARIAH-DE Working Papers, `https://osl.tib.eu/w/DH-Handbuch`, im Druck

Projektberichte

- S. E. Funk, P. Gietz, M. Haase, P. Harms, A. Aschenbrenner, D. Tonne, J. Rybicki, „DARIAH Storage API - A Basic Storage Service API on Bit Preservation Level“, Februar 2012

- D. Tonne, S. E. Funk, „DARIAH Bit Preservation Admin API“

- J. Puhl, S. E. Funk, D. Tonne, M. Haase, T. Kalman, „Auswahl und Beschreibung der initialen technischen Workflows und Policies für den Data Lifecycle“, Mai 2014

Eingeladene Vorträge

- Danah Tonne, „Integration von eCodicology in die DARIAH Dienstewelt“, Digitale Rekonstruktionen mittelalterlicher Bibliotheken, Trier, 18.01.2013

- Danah Tonne, „eCodicology - Algorithms for the Automatic Tagging of Medieval Manuscripts“, Dynamics of Change: eHumanities and Medieval Studies, Mannheim, 25.04.2014

- Danah Tonne, „Data Preservation“, GridKa Summer School, Karlsruhe, 08.09.2015

Danksagung

Diese Arbeit wäre ohne vielfältigen Beistand nicht möglich gewesen, mein Dank gilt daher:

- Prof. Dr. Bernhard Neumair, Prof. Dr. Michael Beigl und Prof. Dr. Marc Weber für die Betreuung inklusive zahlreicher Diskussionen und hilfreicher Anregungen,
- Dr. Rainer Stotzka für die kontinuierliche Unterstützung, das Vertrauen, die Möglichkeit zur Promotion für eine adoptierte Mathematikerin sowie unzählige Bahnkilometer,
- den Mitarbeiterinnen und Mitarbeitern des Instituts für Prozessdatenverarbeitung und Elektronik und insbesondere der Fachgruppe Softwaremethoden, die mich freundlich aufgenommen, trotz meines seltsamen Akzents (meistens) verstanden und außerdem in den fünf Jahren trotz Gefährdung des Idealgewichts stets mit unermüdlicher Begeisterung bei kulinarischen Testreihen unterstützt haben,
- den DARIAHnerinnen und DARIAHnern, die mich immer wieder mit Besonderheiten aus der Welt der Geisteswissenschaften überrascht und meinen Wortschatz erweitert haben,
- den eCodicologinnen und eCodicologen für die familiäre Atmosphäre und zahlreiche interdisziplinäre Momente,
- meinen Freunden und meiner Familie, die meine Eigenheiten ertragen, gerüchteweise sogar mögen, und durch dezente Fragen nach dem Stand der Arbeit die Motivation hochgehalten haben.

Mein größter Dank gilt allen, die fest an meiner Seite stehen, egal wie kompliziert der Weg auch sein mag oder wie viele Kilometer zwischen uns liegen.

Erklärung

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der Fassung vom 27.11.2014 beachtet zu haben.

Karlsruhe, den 17.12.2015

.....
(Danah Tonne)