Universität Regensburg
Fakultät für Wirtschaftswissenschaften
Lehrstuhl für Wirtschaftsinformatik I - Informationssysteme

Design und Implementierung eines Identity and Access Management Reporting Moduls basierend auf Metriken

Masterarbeit

Zur Erlangung des akademischen Grades „Master of Science (M.Sc.)“ im Studiengang Wirtschaftsinformatik an der Fakultät für Wirtschaftswissenschaften der Universität Regensburg

Eingereicht bei: Prof. Dr. Günther Pernul
Betreuung: Thomas Baumer

Eingereicht am 12. April 2023

Eingereicht von:

Julian Bauer
Marienweg 21
94086 Bad Griesbach i. Rottal
Matrikelnummer: 2259323
Fachsemester: 5

Abstract

In den letzten Jahren stieg die Anzahl an erfolgreichen Cyberangriffen gegen Organisationen kontinuierlich an [BS21]. Gleichzeitig haben sich die Ausgaben zur Abwehr dieser stetig erhöht [Bit22]. Die Organisationen stehen hier vor der Herausforderung, die Performance der implementierten Informationstechnologie-(IT-)Sicherheitsmaßnahmen zu quantifizieren, um fundierte Entscheidungen treffen zu können. Durch Bildung von Metriken, die auf Messwerten aus verschiedenen Systemen und Datenquellen aufbauen, werden die Performance und der Reifegrad von der IT-Sicherheit in Organisationen bestimmt. Um diese Metriken nachhaltig zu ermitteln und bereitzustellen, ist ein adäquates IT-System einzusetzen. Im Rahmen der vorliegenden Arbeit wird untersucht, wie im Identity and Access Management (IAM) – ein Teilgebiet von IT-Sicherheit – die Performance mithilfe eines Tools – namens IAM Reporting Modul – basierend auf Metriken ermittelt und berichtet werden kann. Das Vorgehen ist dabei angelehnt an die Design Science Research (DSR) Methodologie nach Hevner [HMPR04]. Ziel ist es, die Anforderungen an ein IAM Reporting Modul zu untersuchen, ein Konzept für ein solches Tool zu entwerfen und auf Grundlage des Konzepts einen Prototyp des IAM Reporting Moduls zu implementieren. Abschließend sind aus den gewonnenen Erkenntnissen Gestaltungsprinzipien abzuleiten. Bei den abgeleiteten Gestaltungsprinzipien handelt es sich um die regelmäßige Berechnung und Ermittlung der Metriken und der Messwerte, um den Prozess zur Konfiguration von den Metriken und den Messwerten und um die zusätzlich benötigten Informationen über den Messwert je nach Typ der Datenquelle sowie um das konzipierte Datenmodell für Reporting Module.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Abkürzungsverzeichnis

ABAC	Attribute-Based Access Control
BI	Business Intelligence
BSR	Behavioral Science Research
CCPA	California Consumer Privacy Act
CSS	Cascading Style Sheets
CSV	Comma-separated values
DSGVO	Datenschutz-Grundverordnung
DAC	Discretionary Access Control
DBMS	Datenbankmanagementsystem
DSR	Design Science Research
EE	Enterprise Edition
FICAM	Federal Identity, Credential, and Access Management
GLBA	Gramm-Leach-Bliley Act
HIPAA	Health Insurance Portability and Accountability Act
HTML	Hypertext Markup Language
IAM	Identity and Access Management
IEC	Internationale Elektrotechnische Kommission
IEEE	Institute of Electrical and Electronics Engineers
ISO	Internationale Organisation für Normung
IT	Informationstechnologie
JAR	Java Archive
JPA	Java Persistence API
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
LGPLv3	GNU Lesser General Public License Version 3
LTS	Long Term Support
MAC	Mandatory Access Control
MIME	Multipurpose Internet Mail Extensions
MVC	Model-View-Controller
NoSQL	Not only Structured Query Language (**SQL**)
OLAP	Online Analytical Processing
OpenJDK	Open Java Development Kit
ORM	Object-Relational Mapping

RBAC	Role-Based Access Control
SOX	Sarbanes-Oxley Act
SSO	Single Sign-on
SQL	Structured Query Language
UML	Unified Modeling Language
USA	Vereinigten Staaten von Amerika
UUID	Universally Unique Identifier
UXD	User Experience Design
XML	eXtensible Markup Language

Kapitel 1

Einleitung

„Was man nicht messen kann, kann man nicht managen.“ – Dieses Zitat wird unter anderem Peter F. Drucker und W. Edwards Deming zugeschrieben [RS21] und gilt ebenso für den Bereich der Informationstechnologie-(IT-)Sicherheit. In 86 Prozent der befragten Unternehmen in Deutschland haben Cyberangriffe im Jahr 2021 einen Schaden verursacht, das ist eine Steigerung von 16 Prozent im Vergleich zu 2019 [BS21]. Gleichzeitig sind die Ausgaben für die IT-Sicherheit in Deutschland in den vergangenen Jahren stetig angestiegen und lagen in 2021 bei 6,9 Mrd. Euro [Bit22]. In den kommenden Jahren wird sich dieser Trend voraussichtlich fortsetzen: Die Ausgaben für die IT-Sicherheit sollen in Deutschland in 2025 auf über 10 Mrd. Euro ansteigen [SM22]. Fraglich ist, ob die zusätzlichen Ausgaben für die IT-Sicherheit tatsächlich den gewünschten Effekt erzielen. Dieser kann nur ermittelt werden, indem die Performance von der IT-Sicherheit quantifiziert wird.

Das Erfassen, die Analyse und das Berichten der Performance von Maßnahmen im Bereich der IT-Sicherheit ist essentiell, um Entscheidungsfindungen zu erleichtern und die Leistung sowie die Zuständigkeiten zu erhöhen [CSS+08]. Die Performance der Maßnahmen und der Reifegrad der IT-Sicherheit kann durch den Einsatz von Metriken evaluiert werden. Um fundierte Aussagen treffen zu können, haben Heinrich et al. [HHK+18] Anforderungen an Datenqualitätsmetriken und Yasain und Schryen [YS15] Anforderungen an Metriken im Bereich der IT-Sicherheit definiert. Den Einsatz von Metriken im Bereich der IT-Sicherheit haben Chew et al. [CSS+08] und Black et al. [BSS09] beschrieben. Wobei Chew et al. ihren Schwerpunkt auf die Implementierung von Metriken und Black et al. auf die Herausforderungen beim Einsatz von Metriken gelegt haben.

In vorliegende Arbeit beschränkt sich auf den Teilbereich Identity and Access Management (IAM) von der IT-Sicherheit. IAM identifiziert und authentisiert Identitäten und autorisiert den Zugriff auf geschützte Ressourcen in Organisationen [Bun22]. Während es bereits mehrere Ansätze und Leitlinien für IAM gibt [FP07; Roy08; IAB18], ist ein noch nicht vollständig untersuchtes Gebiet die Ermittlung der Performance von IAM. Hummer et al. [HGK+18] haben Ziele für IAM definiert, um das bestehende IAM in Organisationen zu analysieren. Noch nicht betrachtet wurde jedoch, wie der aktuelle

Zustand von IAM und die Zielerreichung von IAM in Organisationen langfristig ermittelt und berichtet werden kann.

Dieser Forschungslücke wird in dieser Arbeit nachgegangen. Dabei wurden drei Forschungsfragen definiert. Der Aufbau der Forschungsfragen orientiert sich an den Leitlinien zur Formulierung von Design Science Research-(DSR-)Forschungsfragen von Thuan et al. [TDA19]. Im Rahmen dieser Arbeit wird ein Tool konzipiert und implementiert, mithilfe dessen IAM Metriken erhoben und dadurch der aktuelle Zustand von IAM in Organisationen berichtet werden kann. Dieses Tool wird in dieser Arbeit IAM Reporting Modul genannt. Um ein nachhaltiges und sinnvolles Tool für diesen Anwendungsfall zu erstellen, bedarf es der Erhebung der Anforderungen an ein solches Tool. Deshalb lautet die erste Forschungsfrage RQ1:

> **RQ1:** Was sind die wesentlichen Anforderungen an ein IAM Reporting Modul basierend auf Metriken?

Auf Basis der definierten Anforderungen können Lösungsvorschläge für ein IAM Reporting Modul erarbeitet, abgewogen, ausgewählt und implementiert werden. Die zweite Forschungsfrage RQ2 bezieht sich daher auf die Konzeption und Implementierung des IAM Reporting Moduls:

> **RQ2:** Wie kann ein IAM Reporting Modul basierend auf Metriken konzipiert und implementiert werden, um den aktuellen Zustand von IAM in Organisationen zu berichten?

Ist das IAM Reporting Modul konzipiert und implementiert, kann es evaluiert und Erkenntnisse für ähnliche Anwendungsfälle und Anforderungen daraus gezogen werden. Diese Erkenntnisse sollen im Rahmen der dritten Forschungsfrage RQ3 dargestellt werden:

> **RQ3:** Welche Gestaltungsprinzipien lassen sich von einem IAM Reporting Modul basierend auf Metriken ableiten?

Folglich ist das Ziel dieser Arbeit, die Anforderungen an ein IAM Reporting Modul zu definieren, das IAM Reporting Modul zu konzipieren sowie zu implementieren und schließlich daraus Gestaltungsprinzipien abzuleiten. Dabei wird das Artefakt, das IAM Reporting Modul, unter Einsatz der DSR Methodologie nach Hevner et al. [HMPR04] instantiiert: Das Artefakt wird in einem iterativen Prozess erstellt und evaluiert. Dazu wird sich einer bestehenden Wissensbasis aus wissenschaftlichen Veröffentlichungen über IAM, Metriken, Reporting und Softwareentwicklung bedient. Bei der Erstellung und Evaluierung findet zudem die Umgebung des Artefakts Berücksichtigung: Organisationen, welche die Performance von eingesetzten IAM-Systemen evaluieren möchten.

Der Aufbau dieser Arbeit orientiert sich an dem Publizierungsschema für DSR von Gregor und Hevner [GH13]. Der weitere Verlauf der Arbeit ist wie folgt strukturiert: Im zweiten Kapitel wird der theoretische Hintergrund zu IAM sowie Reporting und Metriken erläutert. Im darauffolgenden Kapitel wird die eingesetzte Methodik vorgestellt.

Im vierten Kapitel, der Anforderungsanalyse, werden zunächst das Ziel, der Systemkontext, die Use Cases und das Domänenmodell bestimmt und danach die wesentlichen Anforderungen an das IAM Reporting Modul definiert. Daraufhin erfolgt auf Basis dieser Anforderungen im fünften Kapitel das Design des IAM Reporting Moduls. Dabei finden die Aspekte Architektur, Datenmodell, Prozesse und Benutzeroberfläche Berücksichtigung. Die Beschreibung der prototypischen Implementierung des zuvor konzipierten IAM Reporting Moduls erfolgt im sechsten Kapitel. In Kapitel Evaluation wird das IAM Reporting Modul mithilfe der Anforderungen evaluiert. Die Ergebnisse der Arbeit werden im achten Kapitel diskutiert sowie abgeleitete Gestaltungsprinzipien, Limitierungen der Arbeit und Vorschläge für zukünftige Arbeiten vorgestellt. Zuletzt wird im neunten Kapitel ein Fazit gezogen.

Kapitel 2

Theoretischer Hintergrund

Dieses Kapitel beschäftigt sich damit, die Arbeit in bestehende Literatur einzuordnen und eine einheitliche Begriffsgrundlage zu schaffen. Zuerst wird in das IAM eingeführt und Reporting definiert. Daraufhin wird definiert, worum es sich bei Metriken handelt und wie diese zu Messwerten abzugrenzen sind. Abschließend wird der Einsatz von Metriken im IT-Sicherheitskontext erläutert.

2.1 Identity and Access Management

IAM. Organisationen nutzen IAM-Systeme, um Identitäten und Zugriffsberechtigungen zu verwalten. IAM setzt sich aus Identity Management und Access Management zusammen [Bun22]. Das Identity Management beschäftigt sich damit, Identitäten von Benutzern als auch Systemen zu verwalten und zu authentifizieren [Bun22; Int19b]. Das Access Management autorisiert den Zugriff von Benutzern und Systemen auf Ressourcen, wie beispielsweise Informationen oder Netzwerke. [Bun22; Int16a].

Die Motivation von IAM ist, Benutzern und Systemen den benötigten Zugriff auf Ressourcen bereitzustellen und unautorisierten Zugriff zu unterbinden [Bun22]. IAM-Systeme bauen gemäß Fuchs et al. [FP07] auf drei Säulen auf: Prozesse, Richtlinien und Technologien. Systeme, Applikationen und Infrastruktur (Technologie) realisieren definierte Geschäftsprozesse, Benutzerverwaltung und Speicherung von Identitäten (Prozesse) sowie Sicherheitsrichtlinien und interne und externe Vorschriften (Richtlinien).

Ziele von IAM. Hummer et al. [HGK+18] haben die Ziele von IAM herausgearbeitet: Risikoreduktion, IT-Kostenreduktion, Verbesserung der Prozess- und Datenqualität, Einhaltung gesetzlicher Vorschriften und Geschäftserleichterung. Diese werden im folgenden kurz erläutert.

IAM unterstützt die IT-Sicherheit und trägt zur allgemeinen Risikoreduktion bei, indem es das Risiko von unbefugtem Zugriff auf Ressourcen verringert. Eine zentrale Datenhaltung von Identitäten, Zugriffsrechten und Zugriffsprotokollen ermöglicht eine Übersicht über alle Daten. Somit können falsche Berechtigungen entdeckt und korrigiert werden.

IAM-Technologien wie beispielsweise ein unternehmensweites Single Sign-on (SSO) oder ein Selbsthilfeportal für Benutzer, können zur IT-Kostenreduktion beitragen. Bei der Verwendung von SSO entfällt die Notwendigkeit die Bereitstellung von mehreren Accounts für einen Benutzer. Benutzer können sich fortan mit einem Account an allen bereitgestellten Systemen anmelden und müssen sich nicht mehr mehrere Zugriffskennungen merken. Vergessene Passwörter können vom Benutzer im Selbsthilfeportal selbstständig zurückgesetzt werden. Auch häufig gestellte Fragen können im Selbsthilfeportal beantwortet werden. Dadurch werden die IT-Support-Mitarbeiter entlastet und Kosten gespart.

Ein zentrales IAM ermöglicht einheitliche und verbesserte Prozesse und steigert die Datenqualität durch eine zentralisierte Datenhaltung. Denn ein zentrales IAM reduziert die Anzahl an benötigten Schnittstellen zwischen den unterschiedlichen Systemen der Organisation und vermeidet Medienbrüche. Infolgedessen werden Fehler durch manuelle Arbeiten oder inkonsistente Prozesse bei der Benutzer- und Rechteverwaltung verhindert.

Für Organisationen gibt es zunehmend mehr nationale und internationale regulatorische Vorgaben und Gesetze, die eingehalten werden müssen. IAM kann bei der Einhaltung der Vorgaben und Gesetze unterstützen. In der Europäische Union gibt es mit der Datenschutz-Grundverordnung (DSGVO) und in Kalifornien mit dem California Consumer Privacy Act (CCPA) Datenschutzgesetze. Auch im Gesundheitswesen gibt es beispielsweise in den Vereinigten Staaten von Amerika (USA) im Health Insurance Portability and Accountability Act (HIPAA) vorgegebene Sicherheitsanforderungen für den Umgang mit Gesundheitsinformationen. Weiter sind in Gramm-Leach-Bliley Act (GLBA) Datenschutzanforderungen für Finanzinstitute der USA festgehalten. Für in den USA ansässige oder tätige Aktiengesellschaften sind im Sarbanes-Oxley Act (SOX) Complianceanforderungen verankert. Im Bankensektor sind die Standards des Basler Rahmenwerks [Bas22] verpflichtend. Zudem besteht für Organisationen die Möglichkeit einer Zertifizierung der Informationssicherheit nach der Internationale Organisation für Normung (ISO)/Internationale Elektrotechnische Kommission (IEC) 27000 Reihe [Int18a].

Durch den Einsatz von IAM gibt es eine zentrale Stelle und standardisierte Prozesse für die Anforderung von Identitäten und Zugriffen. Dadurch wird das Anlegen, Ändern und Löschen von Identitäten und Zugriffen beschleunigt, was die Mitarbeiter schneller arbeitsfähig macht. Auch die Benutzerzufriedenheit steigt durch schnelle und korrekt durchgeführte Prozesse. Insgesamt führt dies zu Geschäftserleichterungen.

In einer durch Hummer et al. [HGK+18] ausgeführten Umfrage von IAM-Experten bezüglich der aufgeführten IAM-Ziele wurde festgestellt, dass alle Ziele bis auf IT-Kostenreduktion für die Experten relevant sind. Die geringe Relevanz der IT-Kostenreduktion wird dadurch hergeleitet, dass die Einführung einer IAM-Lösung zuerst IT-Kosten verursacht und erst mit längerer Laufzeit und Automatisierung Kosten einspart.

Identity Mangement. Im Rahmen des Identity Managements werden Identitäten von Benutzern und Systemen verwaltet und verifiziert. In den Prozessen des Identity Manage-

ments spielen mehrere Begriffe eine Rolle: Entitäten, Identitäten, Attribute, Identifikatoren und Beweise.

Entitäten sind in der realen Welt vorhandene Dinge. Diese können beispielsweise Personen, Organisationen oder Geräte sein. Eine Identität ist ein virtuelles Konstrukt und repräsentiert eine Entität in einem System. Attribute können sowohl mit Entitäten als auch mit Identitäten verknüpft sein. Die Attribute beschreiben deren Charakteristika und Eigenschaften. Beispiele für Attribute sind der Name oder die Adresse. [Cla94; Cla10; GGF17]

In einem spezifischen System oder Kontext sind die Identitäten eindeutig und werden mithilfe von Identifikatoren identifiziert. Die Identifikatoren können einzelne Attribute sein oder sich aus mehreren Attributen zusammensetzen. Typische Identifikatoren sind der Benutzername oder die E-Mail-Adresse. [Cla94]

Um sich als eine Identität zu authentisieren, erbringt eine Entität dafür Beweise. Die Beweise können bilateral vereinbarte Zugangsdaten, wie beispielsweise ein Benutzername und ein Passwort, oder durch eine dritte Partei ausgegebene Nachweise sein, wie z. B. ein Personalausweis. Das eingesetzte Identity Management System oder der Identity Provider prüft daraufhin den erbrachten Beweis und authentifiziert die Anfrage. Bei den Beweisen wird zwischen drei Arten unterschieden:

- Etwas, was man weiß (z. B. ein Passwort)
- Etwas, was man besitzt (z. B. einen Personalausweis)
- Etwas, was man ist (z. B. einen Fingerabdruck)

Um mehr Sicherheit zu erlangen, können mehrere Beweise zur Authentisierung und Authentifizierung kombiniert werden. [GGF17]

Access Management. Im Access Management werden die Berechtigungen von Identitäten verwaltet und erfolgreich authentifizierten Identitäten Zugang zu den geschützten Ressourcen erteilt. Es gibt unterschiedliche Mechanismen und Strategien, um Zugriffsberechtigungen zu steuern. Die Zugriffsberechtigungen bestimmen, welche Aktionen Identitäten ausführen dürfen. [SS94; IAB18]

Discretionary Access Control (DAC) war eine der ersten Mechanismen, um Zugriffe auf Ressourcen zu erteilen. Der Besitzer der Ressource legt fest, welchen anderen Identitäten Zugriffsberechtigungen eingeräumt werden. Bei Zugriff auf eine Ressource wird anhand der Identität die Zugriffsberechtigung überprüft. [GD71; Lam74; Ahn09]

Mandatory Access Control (MAC) ist ein weiteres Verfahren. Die Zugriffsberechtigung erfolgt auf Basis von allgemeinen Regeln. Dabei wird ein hierarchisches Vorgehen verfolgt: Den Identitäten und Ressourcen werden Sicherheitslevel zugewiesen. Greift eine Identität auf eine Ressource zu, wird auf Basis der Sicherheitslevel evaluiert, ob der Zugriff gestattet wird. [Den76; San93]

Role-Based Access Control (RBAC) ist ein weit verbreitetes Verfahren zur Erteilung von Zugängen. Die Identitäten werden anhand von Rollen gruppiert und die Zugriffsbe-

rechtigungen auf der Ebene der Rollen vergeben. Eine Identität kann mehreren Rollen zugewiesen sein. In der Praxis bietet es sich an, die Rollen auf Basis der Geschäftsrollen in Organisationen zu bilden. [FK92; SCFY96]

Attribute-Based Access Control (ABAC) ist ein weiterer Mechanismus, um Zugriffsberechtigungen zuzuweisen. Es werden Regeln für Zugriffsberechtigungen basierend auf den Attributen der Identitäten, den Attributen der Ressourcen und der Umgebungsbedingungen erstellt. Bei Zugriff wird anhand der Regeln ausgewertet, ob der Zugriff gestattet wird. Dies ermöglicht eine dynamische und granulare Zuweisung von Zugriffsrechten. [HFK$^+$14] Sowohl RBAC als auch ABAC lassen sich so anpassen, dass die zuvor beschrieben Verfahren DAC und MAC abgebildet werden können [NO96; San96; HFK$^+$14].

Mithilfe dieser Mechanismen lassen bestimmte Sicherheitsanforderungen und -prinzipien umsetzen: Das Principle of Least Privilege besagt, dass die Zugriffsberechtigungen von Identitäten auf das Minimum eingeschränkt werden sollen. Die Identitäten sollen nur auf die Ressourcen Zugriff besitzen, die für Ausführung ihrer Tätigkeit unabdingbar sind. Dieses Prinzip minimiert das Risiko von ungewollter oder unangemessener Nutzung von Zugriffsberechtigungen und kann den Schaden bei Fehlern oder Sicherheitsvorfällen reduzieren. [SS75; SCFY96]

Ein weiteres Prinzip ist die Separation of Duties: Keine Identität soll die vollständigen Zugriffsberechtigungen besitzen, die es ermöglichen würden, ein System für eigene Zwecke auszunutzen. Für ausgewählte Aufgaben soll sichergestellt werden, dass mehr als eine Identität benötigt wird, um diese abzuschließen. Dieses Prinzip dient dazu, Betrug und Fehler zu verhindern. Zugriffsberechtigungen müssen beispielsweise so verteilt sein, dass ein Antragssteller eines Antrags nicht gleichzeitig den Antrag genehmigen kann. [SS75; SCFY96]

Identity Lifecycle. Identitäten durchlaufen über die Zeit hinweg einem Lebenszyklus. Die ISO [Int19b] hat verschiedene Zustände, in denen sich Identitäten befinden können, und Zustandsübergänge definiert. Eine Identität ist anfangs unbekannt und wird in einem ersten Schritt erstellt und verifiziert. Bevor eine Identität auf Ressourcen zugreifen kann, muss diese aktiviert werden. Im aktiven Zustand können Identitäten editiert, suspendiert oder archiviert werden.

Cameron und Grewe [CG22] unterscheidet zwischen verschiedenen Lebenszyklen von Identitäten je nach Art des Anwendungsfalls: Mitarbeiter, Kunden oder Geräte einer Organisation. Im folgenden wird der Lebenszyklus für Identitäten von Mitarbeitern vorgestellt. Dieser unterteilt sich in drei Phasen. In der Joiner-Phase werden die Identitäten von neu eingetretenen Mitarbeitern angelegt und Zugriffsrechte vergeben. Wechselt ein Mitarbeiter die Abteilung oder das Projekt, befindet sich die Identität in der Mover-Phase. Die Zugriffsrechte werden überprüft und angepasst: Nicht mehr benötigte Zugriffsberechtigungen werden entzogen und zukünftig benötigte Zugriffsberechtigungen werden

vergeben. In der Leaver-Phase werden alle Zugriffsberechtigungen der Identitäten von Mitarbeitern entfernt, welche die Organisation verlassen haben.

2.2 Reporting und Metriken

Reporting. Das betriebliche Berichtswesen bzw. Reporting ist in den Bereich des Controllings einzuordnen und ist dort ein Instrument zur Koordination und Kommunikation [GHM08, 17]. Gleich et al. [GHM08, 17 ff.] haben unterschiedlich eng gefasste Definitionen von Reporting in der Literatur gegenübergestellt: Das Reporting nach Blohm [Blo75] umfasst sämtliche Aktivitäten von der Ermittlung des Informationsbedarfs über die Informationsbeschaffung, Informationserzeugung, Informationsbereitstellung und -übermittlung bis hin zur Informationsnutzung. Göpfert [Gö06, 694] grenzt das Reporting von der Informationserzeugung bis zur Informationsnutzung ein. Bei Horváth [Hor08, 540], Koch [Koc94, 53 ff.] und Gleich et al. [GHM08, 19] konzentriert sich das Reporting auf die Informationsbereitstellung und -übermittlung sowie auf die Informationsnutzung. In dieser Arbeit findet die letzte, enger gefasste Definition Anwendung. Ein Bericht bzw. Report sind für eine übergeordnete Zielsetzung und einen Berichtszweck zusammengefasste Informationen [Blo74, 15].

Reportingsysteme. Reportingsysteme sind den Business Intelligence (BI)-Systemen zuzuordnen [CG16, 5]. BI-Systeme dienen der „Informationsversorgung und funktionale[n] Unterstützung betrieblicher Fach- und Führungskräfte zu Analysezwecken“ und bilden inhaltlich das „logische Komplement zu den operativen Informationssystemen“ [CG16, 7 f.].

Reportingsysteme stellen ihren Benutzern Informationen in aufbereiteter Form zur Verfügung. Die Aufbereitung erfolgt dabei i. d. R. durch Visualisierung von Sachzusammenhängen in Diagrammen. Zu unterscheiden sind aktive Reportingsysteme, die Reports anhand von Zeitplänen oder nach Überschreitung von Grenzwerten erstellen, und passive Reportingsysteme, die auf konkrete Anforderung eines Benutzers einen individuellen und bedarfsspezifischen Report generieren. [KBM10, 124 ff.]

Metriken. Um festzustellen, ob die IAM-Ziele in Organisationen effektiv und effizient umgesetzt werden, können diese mit geeigneten Metriken ausgewertet werden. Es gibt eine Vielzahl an unterschiedlichen Definitionen für Metriken [YS15]. In dieser Arbeit wird die Definition von Chew et al. [CSS+08] verwendet: Metriken sind Hilfsmittel, welche die Entscheidungsfindung erleichtern und die Leistung und Nachvollziehbarkeit verbessern. Hierzu werden die relevanten leistungsbezogenen Daten gesammelt, analysiert und berichtet.

Messwerte. Die Begriffe Metriken und Messwerte werden in der Literatur oft als Synonyme eingesetzt (z.B. in [CSS+08] und [BKY11]). In dieser Arbeit wird an Anlehnung an die Definition von Black et al. [BSS09] jedoch zwischen den beiden Begriffen unter-

schieden: Ein Messwert ist ein konkretes, objektives Attribut. Dagegen ist eine Metrik ein abstraktes, subjektives Attribut. Auch die ISO unterscheidet in der ISO 15939 [Int17] zwischen Messwerten und Metriken: Ein Messwert ist eine Variable mit bestimmten Attributen und einer definierten Messmethode, deren Wert durch eine Messung quantifiziert wird. Eine Metrik wird anhand einer Formel aus mehreren Messwerten oder Metriken abgeleitet. Ein Beispiel für einen Messwert ist die Länge eines Passworts. Eine Metrik ist beispielsweise der Anteil an Accounts mit schwachen Passwörtern.

Einsatz von Metriken. Sowohl Black et al. [BSS09] als auch Chew et al. [CSS+08] beschreiben den Einsatz und die empfohlene Verwendung von Messwerten und Metriken im IT-Sicherheitskontext in Organisationen. Im Nachfolgenden werden ausgewählte, für diese Arbeit relevante Aspekte vorgestellt.

Die Anwendungsfälle von Metriken in Organisationen im Bereich IT-Sicherheit können unterschiedlich sein: Die Compliance von Sicherheitsmaßnahmen mit Richtlinien, Prozessen und Verfahren kann mithilfe von Metriken überprüft werden. Ferner können Stärken und Schwächen im Bereich IT-Sicherheit und Sicherheitstrends innerhalb als auch außerhalb der Kontrolle der Organisation identifiziert werden. [BSS09]

Chew et al. [CSS+08] führen neben dem Nachweis von Compliance noch weitere Vorteile beim Einsatz von Metriken an: Durch das Berichten von Metriken wird die Verantwortlichkeit erhöht und die Effektivität der IT-Sicherheit gesteigert. Metriken stellen außerdem quantifizierbare Daten für die Ressourcenzuweisung bereit.

Metriken sollten in mehrere Ebenen eingeteilt werden, um unterschiedliche Zielgruppen zu adressieren. Taktische Entscheidungen können durch Metriken auf niedriger Ebene erleichtert werden, während strategische Entscheidungen durch Metriken auf höheren Ebene unterstützten werden können. Weiter können Metriken von niedrigeren Ebenen zu Metriken auf höheren Ebenen kombiniert werden. [BSS09]

Organisatorisch sollen für die Entwicklung von Metriken alle relevanten Stakeholder identifiziert und eingebunden werden. Falls bereits organisatorische Vorgaben für den Umgang mit Metriken bestehen, soll sich an diese angelehnt werden. [CSS+08]

Bevor Messwerte und Metriken analysiert und berichtet werden können, müssen Messwerte gesammelt werden. Dies kann einerseits manuell als auch automatisch geschehen. Ein wesentlicher Erfolgsfaktor für Metriken ist die Qualität und Validität der Messwerte und Metriken. Aus Gründen von potentiell höherer Datenqualität, Wiederholbarkeit und Verhinderung menschlicher Fehler, sollten Messwerte automatisch gesammelt werden. [BSS09; CSS+08]

Durch das Sammeln der Messwerte fallen große Datenmengen an. Ein Konzept zur Verwaltung der Messwerte und Metriken ist aus diesem Grund essentiell. Aus diesem Grund wird empfohlen, die Messverfahren, die Datenhaltung und das Berichtswesen zu standardisieren und klar zu definieren. Da speziell auch im IAM personenbezogene Daten für Metriken Verwendung finden können, sind Datenschutzgrundlagen und -gesetze zu

beachten und die Daten gegebenenfalls zu pseudonymisieren oder zu anonymisieren. [CSS+08]

Nachdem die Messwerte gesammelt sind, können diese für weitere Analysen und Auswertungen nach verschiedenen Kategorien gruppiert werden. Für die Darstellung der Metriken werden diese anhand von Formeln aus den Messwerten berechnet. Die Metriken können daraufhin in Dashboards angezeigt werden. Spezielle BI-Software ermöglicht es dessen Benutzern Online Analytical Processing (OLAP) Operationen auf verschiedenen Dimensionen der Messwerten und Metriken auszuführen, um diese zu analysieren und zu visualisieren. [BSS09]

Herausforderungen. Bei dem Einsatz von Metriken haben Black et al. [BSS09] verschiedene Herausforderungen erarbeitet, die sich in drei Teilgebiete gliedern lassen: Genauigkeit, Auswahl und Verwendung der Messwerte.

Die Genauigkeit der Metriken ist per se abhängig von der Genauigkeit der Messwerte, da Metriken sich aus diesen zusammensetzen. Probleme mit der Genauigkeit treten auf, wenn diese oder die Terminologie selbst unpräzise definiert sind. Auch inkonsistente Messmethoden, qualitative Messwerte oder absolute Werte ohne Kontext, Norm oder Ziel sind zu vermeiden. Allgemein befindet sich IT im ständigen Wandel, weshalb sich die Bedeutung von Messwerten und Metriken über die Zeit auch verändern können.

Die Herausforderung bei der Auswahl der Messwerte ist die Vielzahl an zur Verfügung stehenden Quellen und Messwerte. Ohne vorab die benötigten Metriken und Messwerte zu planen bzw. deren Nützlichkeit zu evaluieren, werden von der Organisation gegebenenfalls nicht benötigte Messwerte gesammelt. Daraus resultiert ein unnötiger Zeit- und Ressourcenaufwand zum Sammeln, Analysieren und Berichten der Messwerte und Metriken. Ebenso können die Abhängigkeiten zwischen Messwerten und Metriken unklar oder nicht genau repräsentierend sein. Dieser Umstand kann bei der darauffolgenden Analyse der Metriken zu irreführenden Ergebnissen führen. Wenn die Verwendung der Messwerte unklar ist, kann das Sammeln der Messwerte den Mitarbeitern als eine Zeitverschwendung vorkommen. Ferner ist es möglich, dass Mitarbeiter vor allem solche Messwerte und Metriken auswählen und mit anderen teilen, die überwiegend positive Ergebnisse darstellen.

Wurden ausgewählte Messwerte mit ausreichender Güte gesammelt, müssen diese korrekt verwendet und zu Metriken kombiniert werden. Die Messwerte können in verschiedenen Maßeinheiten, Skalen und unterschiedlicher Präzision vorliegen und müssen mithilfe von geeigneten Formeln kombiniert werden. Sind einige Messwerte für eine Metrik von höherer Bedeutung als andere, so können diese gewichtet werden. Herausfordernd ist dabei, eine passende Gewichtung auszuwählen. Wie auch schon bei der Genauigkeit der Messwerte und Metriken angemerkt, ist bei der Verwendung ebenso darauf zu achten, dass sich Metriken und Messwerte über die Zeit verändern können und somit anzupassen sind.

Zusätzlich zu den Herausforderungen von Black et al. [BSS09] haben Chew et

al. [CSS+08] die Herausforderung der Verwaltbarkeit angeführt. Grundsätzlich können viele Aktivitäten im IT-Sicherheitskontext quantifiziert werden. Aufgrund meist begrenzter Ressourcen, ist eine Priorisierung erstrebenswert. Ziel sollte es sein, Performancedefizite mit wenigen geeigneten Metriken zu analysieren und zu schließen. Dabei wird empfohlen, nur zwei bis drei Metriken in die Verantwortung eines Stakeholders zu legen. Dies soll sicherstellen, dass die gesammelten Messwerte und Metriken aussagekräftig sind, zu relevanten Ergebnissen führen und die Stakeholder genügend Zeit haben, um Verbesserungen anzuwenden. Wenn sich das IT-Sicherheitsprogramm weiterentwickelt und die Zielvorgaben der Metriken erreicht wurden, können obsolete Metriken aussortiert werden und neue Metriken identifiziert werden. Auch bei Änderung der Mission der Organisation oder der Vorgaben im IT-Sicherheitsprogramm sind die Metriken daran anzupassen.

Anforderungen an Metriken. Um mithilfe von Metriken fundierte Aussagen generieren zu können, werden viele Messwerte benötigt. Für die Messwerte gelten dabei die allgemein bekannten Big Data Merkmale Volume, Velocity, Variety, Veracity und Value [DLSA15]. Die Messwerte liegen in großen Mengen vor, werden mit großer Geschwindigkeit generiert, sind heterogen und durch Extraktion von Wissen kann ein Mehrwert generiert werden. Daher muss sichergestellt sein, dass die Messwerte in ausreichender Güte vorliegen. Es gibt verschiedene Veröffentlichungen, die Anforderungen an die Messwerte und Metriken stellen. Heinrich et al. [HHK+18] haben beispielsweise fünf Anforderungen an Datenqualitätsmetriken festgelegt. Zunächst sollen Messwerte und Metriken nach unten und oben hin begrenzt sein, sodass es genau einen kleinstmöglichen und genau einen größtmöglichen Wert gibt. Die Grenzen bilden dabei die schlechteste und die beste mögliche Datenqualität ab. Die Verwendung einer Intervallskala durch Messwerte und Metriken ist die zweite Anforderung. Die dritte Anforderung verlangt bei der Bestimmung der Metriken und deren Konfigurationsparametern die Berücksichtigung der Gütekriterien Objektivität, Reliabilität und Validität. Gemäß der vierten Anforderung sollen die Messwerte und Metriken konsistent aggregierbar sein und auf unterschiedliche Ebenen der Datensichten anwendbar sein. Zuletzt soll der erwartete Nutzen der Messwerte und Metriken größer als die erwarteten Kosten für die Bestimmung der Metriken und deren Konfigurationsparameter sein.

Yasain und Schryen [YS15] haben für IT-Sicherheitsmetriken ebenfalls fünf Anforderungen formuliert. Einige der Anforderungen an IT-Sicherheitsmetriken decken sich dabei mit den Anforderungen an Datenqualitätsmetriken von Heinrich et al. [HHK+18]. Erstens sollen IT-Sicherheitsmetriken genauso wie Datenqualitätsmetriken Werte besitzen, welche die untere und obere Grenze festlegen. Im Gegensatz zu Datenqualitätsmetriken wird in der zweiten Anforderung an IT-Sicherheitsmetriken eine metrische Skalierung gefordert. Die metrische Skalierung ermöglicht es Abstände zwischen den Metriken zu messen. Als drittes werden für die IT-Sicherheitsmetriken ebenso die Gütekriterien Objektivität, Reliabilität und Validität gefordert. Laut der vierten Anforderung sollen

IT-Sicherheitsmetriken kontextspezifisch sein. Somit sollen die Metriken für den jeweiligen Anwendungsfall wertvoll sein. Ebenso sind bei dieser Anforderung Parallelen zu der vierten Anforderung an Datenqualitätsmetriken erkennbar. Als Letztes wird gefordert, die IT-Sicherheitsmetriken automatisch zu berechnen, um Kosten zu sparen und Fehler zu verhindern. Auch wenn diese Anforderung von der letzten Anforderung an Datenqualitätsmetriken abweicht, verfolgen beide das gleiche Ziel: Metriken wirtschaftlich rentabel einzusetzen.

Aufbau von Metriken. Um Metriken und Messwerte wiederkehrend zu sammeln, zu analysieren und zu berichten, bedarf es einer einheitlichen Dokumentation der Metriken. Chew et al. [CSS+08] und die ISO in ISO/IEC 27004:2016 [Int16b] haben relevante Attribute für die Dokumentation von Metriken und Messwerte vorgeschlagen. Auf die einzelnen Attribute wird nachfolgend eingegangen.

Um eine Metrik oder einen Messwert zu identifizieren, wird ein im Kontext eindeutiger Identifikator vergeben. Mithilfe des Identifikators können Metriken oder Messwerte beispielsweise nachverfolgt und sortiert werden. [CSS+08; Int16b]

Die Erfassung eines Messwerts oder einer Metrik ist nur sinnvoll, wenn damit benötigte Informationen für einen Informationsbedarf geliefert werden. Daher ist zu beschreiben, welche Informationen ein Messwert oder eine Metrik für einen Informationsbedarf beitragen. Informationsbedarfe können beispielsweise die Evaluation von strategischen oder IT-Sicherheitszielen sein. [CSS+08; Int16b]

Neben der Beschreibung, wozu der Messwert oder die Metrik beiträgt, soll der Messwert oder die Metrik selbst beschrieben werden. Für die Beschreibung von Metriken und Messwerten bietet sich eine numerische Aussage beginnend mit den Wörtern Prozent, Nummer, Frequenz oder Durchschnitt an. [CSS+08; Int16b]

Um den Wert der beschriebenen Metrik oder des Messwerts zu berechnen, ist eine Formel zu definieren. Die Formel legt fest, wie der Wert der Metrik oder des Messwerts bewertet, berechnet oder benotet wird. Das Ergebnis sollte eine numerische Aussage sein. Da Metriken oder Messwerte auf anderen Messwerten und Metriken aufbauen können, kann das Ergebnis einer Metrik oder eines Messwerts als Eingabeparameter für die Formel einer Metrik oder eines Messwerts eingesetzt werden. [CSS+08; Int16b]

Metriken und Messwerte messen, zu welchem Grad das gewünschte Ergebnis oder Ziel erreicht ist. Aus diesem Grund bedarf es der Definition einer Zielsetzung. Die Zielsetzung können Meilensteine, statistische Maße oder eine Gruppe von Schwellenwerten sein. Die Zielsetzung kann in unterschiedlichen Einheiten definiert werden: z. B. als Prozentsatz, Zeit, Geld oder in anderen passenden Maßeinheiten. Durch regelmäßige Ermittlung der Metriken oder Messwerte kann der Zielerreichungsgrad im Zeitverlauf dargestellt werden. [CSS+08; Int16b]

Die Abstände zwischen den einzelnen Messungen als auch Analysen von Metriken oder Messwerten werden als Frequenz hinterlegt. Die Frequenz sollte passend zu der Änderungsrate der evaluierten Daten gewählt werden. Auch die Frequenz, in der die

Metriken und Messwerte berichtet werden, sollte festgelegt werden. Diese Frequenz sollte ausgelegt an die Bedürfnisse der Empfänger sein. [CSS+08; Int16b]

In einem weiteren Punkt wird vorgeschlagen, Belege für die Durchführung der Messung zu definieren. Diese Belege sollen dabei unterstützen, die Ursachen für mangelhafte Ergebnisse zu identifizieren, die Durchführung der Aktivitäten zu validieren und die Daten als Eingabeparameter für die Formeln zu liefern. Für die Datenerhebung sind die benötigten Daten zu identifizieren, die Akzeptanzkriterien der Daten und die Strategien zur Validierung der Daten festzulegen. Zusätzlich dazu sind bei der manuellen Datenerhebung Fragebögen vorzubereiten. [CSS+08; Int16b]

Für Metriken und Messwerte sind die wesentlichen Stakeholder zu identifizieren. Zum einen gibt es einen Besitzer der Metrik und des Messwerts. Weiter gibt es Verantwortliche für das Sammeln der Daten. Zuletzt gibt es die Nutzer oder die Zielgruppen der Metrik oder des Messwerts, welche diese empfangen. [CSS+08; Int16b]

Die benötigten Datenquellen zur der Metriken und Messwerte sind ebenso festzuhalten. Diese umfassen Datenbanken, Trackingtools, Organisationen, spezifische Rollen in Organisationen oder externe Organisationen, die Informationen bereitstellen können. [CSS+08; Int16b]

Zuletzt ist das Format, in dem die Metriken und die Messwerte gesammelt und berichtet werden, zu definieren. Sowohl textuelle, nummerisch oder grafische Berichtsformate als Kuchen-, Linien- oder Balkendiagramm sind möglich. Die einzelnen Metriken können Teil eines Dashboards sein oder in anderen Formen präsentiert werden. [CSS+08; Int16b]

Zusätzlich zu den bereits genannten Attributen von Metriken haben Chew et al. [CSS+08] unterschiedliche Typen von Metriken und Messwerten definiert. Daher ist in ihrer Dokumentation von Metriken und Messwerten ein weiteres Attribut namens Typ vorhanden.

Metriktypen. Für IT-Sicherheitsprogramme von Organisationen haben Chew et al. [CSS+08] unterschiedliche Typen von Metriken definiert. Je nach Reifegrad des IT-Sicherheitsprogramms finden unterschiedliche Typen Anwendung. Die Reife eines Programms wird definiert durch die etablierten Prozesse und Verfahren. Umso reifer das IT-Sicherheitsprogramm ist, desto detaillierter sind Richtlinien dokumentiert, desto mehr Prozesse sind standardisiert und desto mehr Daten für die Ermittlung von Performance sind verfügbar. In der Abb. 2.1 sind die unterschiedlichen Reifegrade eines IT-Sicherheitsprogramms und die damit verbundenen Metriken dargestellt. Im niedrigsten Stadium müssen IT-Sicherheitsziele definiert werden, um darauf aufbauend Metriken entwickeln zu können. In weiter entwickelten IT-Sicherheitsprogrammen kann der Implementierungsgrad mithilfe von Metriken evaluiert werden. In etablierten Programmen werden Metriken eingesetzt, um die Effizienz, Effektivität und Geschäftsauswirkung von den IT-Sicherheitsprozessen und Vorgehen zu ermitteln.

Grundsätzlich können die unterschiedlichen Typen an Metriken gleichzeitig eingesetzt werden, der Fokus der Metriken variiert je nach Reifegrad des IT-Sicherheitsprogramms.

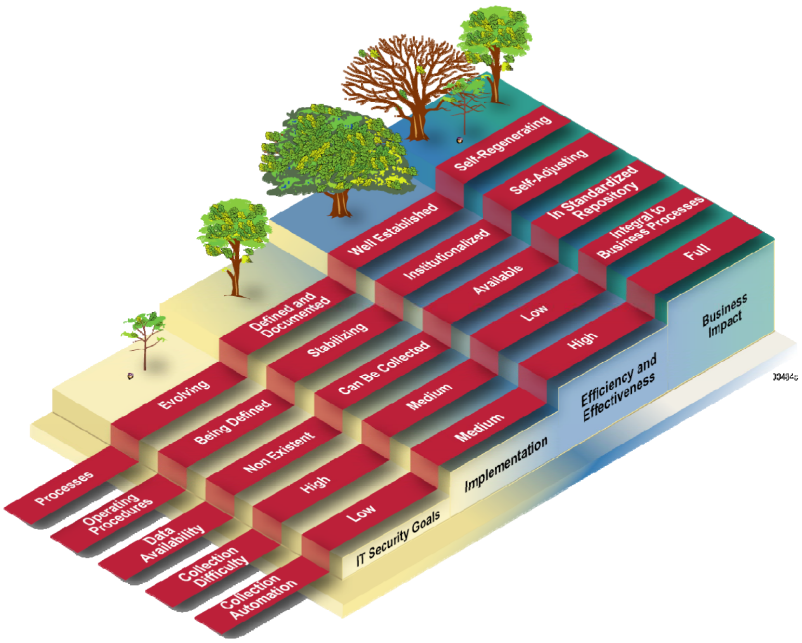

Abb. 2.1 IT-Sicherheitsprogramm Reife und Typen von Metriken [CSS+08, 12]

Den Fortschritt in der Implementierung von IT-Sicherheitsprogrammen, spezifischen Maßnahmen und zugehörigen Richtlinien zeigen Metriken auf, die den Implementierungsgrad wiedergeben. Diese Metriken werden in Prozent angegeben und haben zu Beginn einen Wert unter 100 Prozent. Langfristiges Ziel ist es, in all diesen Metriken einen Wert von 100 Prozent zu erreichen und zu halten. Ist dieses Ziel erreicht, kann der Fokus auf die weiteren Metriktypen gelenkt werden.

Der nächste Metrikentyp untersucht die Effizienz und Effektivität von IT-Sicherheitsprozessen und -maßnahmen. Diese Metriken ermöglichen Rückschlüsse auf die Konformität. Dabei spiegeln Effektivitätsmetriken wider, ob das definierte Ergebnis erreicht wird. Ob das Ergebnis rechtzeitig realisiert wird, zeigen die Effizienzmetriken. Um die Effektivitäts- und Effizienzmetriken zu berechnen, sind meist einzelne Messwerte nicht mehr ausreichend und werden daher aus mehreren Messwerten zusammengesetzt.

Der letzten Metriktyp beschäftigt sich mit dem Einfluss von IT-Sicherheit auf die Mission einer Organisation. Diese Metriken sind per se organisationsspezifisch, da jede Organisation eine eigene Mission verfolgt. Für solche Metriken werden Informationen über IT-Sicherheitsaktivitäten mit der damit verbundenen Ressourcennutzung kombiniert.

Prozesse. Chew et. al [CSS+08] haben zwei Prozesse für die Etablierung und Anwendung von Metriken im Rahmen von IT-Sicherheit vorgestellt: Einerseits einen Prozess zur Entwicklung von Metriken und andererseits einen Prozess zur Implementierung von Metriken. Die Prozesse sollen dabei unterstützen, die am besten geeigneten Metriken auszuwählen, alle relevanten Stakeholder abzuholen und die benötigte technische Infrastruktur sowie Verfahren bereitzustellen.

Der Prozess zur *Entwicklung von Metriken* besteht aus zwei Hauptaktivitäten (siehe

Abb. 2.2): Die erste Aktivität identifiziert und definiert den aktuellen Status der IT-Sicherheit. Die zweite Aktivität entwickelt und wählt spezifische Metriken aus, die es ermöglichen die Implementierung, Effektivität, Effizient und Auswirkung von Sicherheitsmaßnahmen zu bewerten.

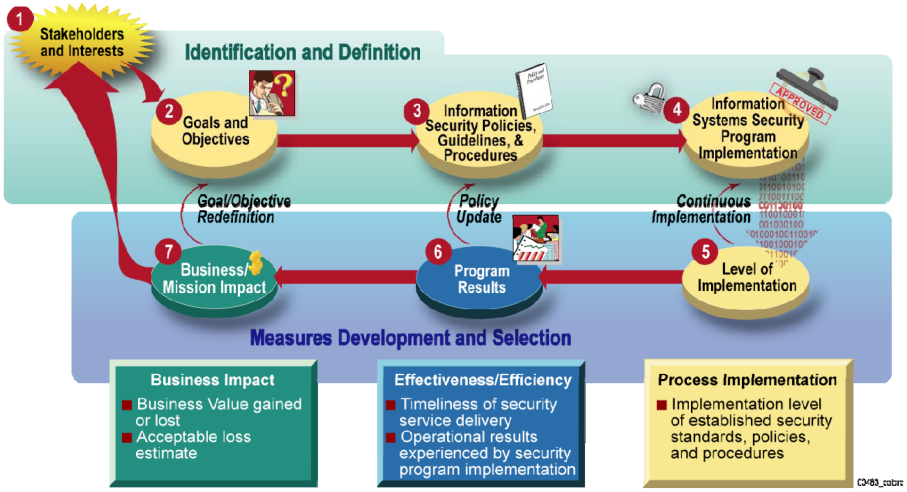

Abb. 2.2 Prozess zur Entwicklung von Metriken [CSS+08, 25]

Im ersten Schritt werden die relevanten Stakeholder und deren Interessen identifiziert. Die Stakeholder sollen für einzelne, wenige Metriken verantwortlich sein und während des gesamten Prozesses der Entwicklung der Metriken eingebunden sein. Im zweiten Schritt werden Ziele und Vorgaben identifiziert und dokumentiert, welche die Implementierung von Sicherheitsmaßnahmen vorgeben. Diese Ziele und Vorgaben können beispielsweise aus übergeordneten Strategien und Anforderungen, Gesetzen, Verordnungen, Richtlinien und Leitlinien abgeleitet werden. Als nächstes werden IT-Sicherheitsvorgaben, -richtlinien und -prozeduren der Organisation betrachtet. Diese Dokumente beschreiben meist wie IT-Sicherheitsziele und -vorgaben durch Sicherheitsmaßnahmen, -anforderungen und -techniken umsgesetzt werden. Im vierten und somit letzten Schritt der ersten Hauptaktivität wird überprüft, ob bereits Messwerte und Daten existieren, aus denen Metriken abgeleitet werden können. Geeignete Messwerte und Daten beweisen, dass Sicherheitsmaßnahmen umgesetzt wurden. Nachdem der aktuelle Status der IT-Sicherheit identifiziert und definiert wurde, werden in den darauffolgenden Schritten Metriken entwickelt und ausgewählt, die den Implementierungsfortschritt, die Effizienz und Effektivität und die Auswirkung auf das Geschäft von Sicherheitsmaßnahmen, Gruppen von Sicherheitsmaßnahmen und Sicherheitsprogrammen widerspiegeln. Die Metriken sollten dabei entweder direkt einzelnen Sicherheitsmaßnahmen oder übergeordneten Zielen zugeordnet werden.

Da der Umfang der zur Verfügung stehenden Messwerte und Metriken meist groß ist, sollten einzelne Metriken priorisiert ausgewählt werden und nach und nach um weitere Metriken erweitert werden. Die ausgewählten Metriken sollten die Verbesserung der Sicherheitsmaßnahmen mit hoher Priorität erleichtern, sowie Daten und Messwerte be-

nutzen, die in bestehenden Quellen enthalten sind und für die bereits Ermittlungsprozesse bestehen.

Nachdem die Metriken erstellt und ausgewählt wurden, sollen Zielvorgaben für jede Metrik festgelegt werden. Anhand der Zielvorgaben kann der Erfolg von beispielsweise Sicherheitsmaßnahmen ermittelt werden. Die Zielvorgabe für den Implementierungsfortschritt sollte generell 100 Prozent sein. Für Effizienz und Effektivität sowie Auswirkung auf das Geschäft müssen die Zielvorgaben individuell festgelegt werden. Hierbei können historische Daten zur Hilfe gezogen werden.

Abgeschlossen wird der Prozess durch die Dokumentation der erarbeiteten Messwerte, Metriken und Zielvorgaben. Die Vorlage zur standardisierten Dokumentation von Messwerten und Metriken umfasst unter anderem die nachfolgenden Felder: Identifikator, Ziele, Beschreibung, Typ, Formel, Zielvorgabe, Nachweise für die Umsetzung, Häufigkeit der Ermittlung und Analyse, verantwortliche Stakeholder, Datenquellen und Berichtsformat.

Das primäre Ziel der Metriken ist es, die Performance zu ermitteln, Ursachen für schlechte Performance und Felder für Verbesserungen aufzudecken. Daneben können die Metriken auch Feedback über die im Prozess untersuchten Ziele, Vorgaben, Richtlinien geben und deren kontinuierliche Weiterentwicklung erleichtern.

Der zweite Prozess zur *Implementierung der Metriken* setzt die Metriken ein, um die Performance der Sicherheitsmaßnahmen zu überwachen und anhand der Ergebnisse die Verbesserung der Performance zu initiieren. Der Prozess besteht aus insgesamt sechs Schritten und stellt dabei die kontinuierliche Verwendung der Metriken sicher (siehe Abb. 2.3).

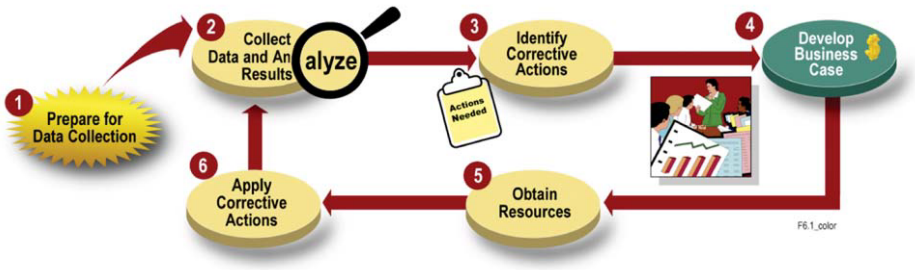

Abb. 2.3 Prozess zur Implementierung von Metriken [CSS+08, 35]

Gestartet wird der Prozess mit vorbereitenden Maßnahmen für die Datenerhebung. Diese beginnen mit der Etablierung eines umfassenden Messprogramms zur Metrikidentifikation, -definition, -entwicklung und -auswahl. Ein Implementierungsplan dokumentiert daraufhin die spezifischen Implementierungsschritte, wie beispielsweise die Messwerte gesammelt, Metriken analysiert und berichtet werden. Der Plan beinhaltet außerdem Bestimmungen, wie das Messprogramm dauerhaft zu überprüfen ist. Im zweiten Schritt werden die Messwerte gesammelt, die darauf aufbauenden Metriken berechnet und gespeichert. Anschließend werden die Metriken analysiert und mit Performancezielen verglichen. Werden die Performanceziele nicht erreicht, sind Ursachen für schlechte Performance und Bereiche mit Verbesserungspotential zu ermitteln. Nachdem Bereiche mit

Verbesserungspotential identifiziert wurden, wird im dritten Schritt ein Plan entwickelt, wie die Lücken geschlossen werden können. Der Plan soll für jeden Problembereich Verbesserungsvorschläge enthalten und diese gemäß ihrem Risiko priorisieren. Basierend auf den Ergebnissen der ersten drei Schritte wird im vierten Schritt ein Geschäftsszenario erstellt, um Kosten und Risiken abzuschätzen. Darin sollte der aktuelle Zustand mit den vorgeschlagenen Alternativen hinsichtlich Kosten, Risiko und Vorteilen verglichen werden. Im fünften Schritt wird das im Geschäftsszenario dargelegte sowie das benötigte Budget beantragt. Nach Erhalt des Budgets sind die erhalten Ressourcen zu priorisieren und Aufgaben zuzuweisen. Im letzten Schritt sind die Korrekturmaßnahmen anzuwenden. Durch die kontinuierliche Überwachung durch das Messprogramm kann zu einem späteren Zeitpunkt festgestellt werden, ob alle Ziele erreicht wurden oder weiterer Handlungsbedarf besteht.

Kapitel 3

Methodik

Die Forschung im Bereich der Informationssysteme ist gekennzeichnet durch zwei Paradigmen: Behavioral Science Research (BSR) und DSR. Während BSR darauf abzielt Theorien zu entwickeln und zu überprüfen, um menschliches oder organisatorisches Verhalten zu erklären oder vorherzusagen, zielt DSR darauf ab, neue und innovative Artefakte zu erstellen und zu evaluieren, um menschliche und organisatorische Fähigkeiten zu erweitern. [HMPR04]

Hevner und Chatterjee [HC10] haben festgestellt, dass BSR und DSR komplementäre Paradigmen sind, um fundamentale Probleme im produktiven Einsatz von Informationstechnologie zu lösen, da Technologie und Verhalten untrennbar sind. Die Theorien von BSR liefern die Wahrheit und sind Basis für DSR. Auf der anderen Stelle liefern Artefakte der DSR Hilfsmittel für BSR.

Um die Forschungsfragen dieser Arbeit zu beantworten, wurde nach der DSR Methodologie nach Hevner et al. [HMPR04] vorgegangen. DSR eignet sich vor allem für Arbeiten wie diese, bei denen für ein bestehendes Problem eine Lösung gesucht wird. Die Problemstellung wurde bereits in der Einleitung erläutert: Organisationen stehen vor der Herausforderung den aktuelle Zustand von IAM zu ermitteln und zu berichten. Außerdem wurde der Wert einer Lösung herausgestellt: Wenn der aktuellen Zustand von IAM bekannt ist, können auf dessen Basis im IAM fundierte Entscheidungen getroffen und die Leistung sowie die Zuständigkeiten erhöht werden. Zur Lösung des Problems wird ein Artefakt erstellt und dessen Performance darauffolgend evaluiert. Ein Artefakt kann eine Konstruktion, ein Modell, eine Methodik oder eine Instanziierung zur Lösung des spezifizierten Problems sein. Im Falle dieser Arbeit ist das Artefakt das konzipierte und prototypisch implementierte IAM Reporting Moduls.

Die Abb. 3.1 zeigt das konzeptionelle Framework von DSR nach [HMPR04]. Basis für die DSR-Forschung bilden die Säulen Environment und Knowledge Base. Das Environment definiert das Problemumfeld, in dem das Problem vorherrscht und für das eine Lösung gesucht wird. Es umfasst Menschen, Organisationen als auch Technologien. Aus dem Environment werden die Anforderungen abgeleitet. Die Knowledge Base ist die Wissensbasis, aus der sich die Forscher bestehender Grundlagen und Methodiken bedienen, um eine Lösung herbeizuführen. In die Säule Design fließen die Anforderungen

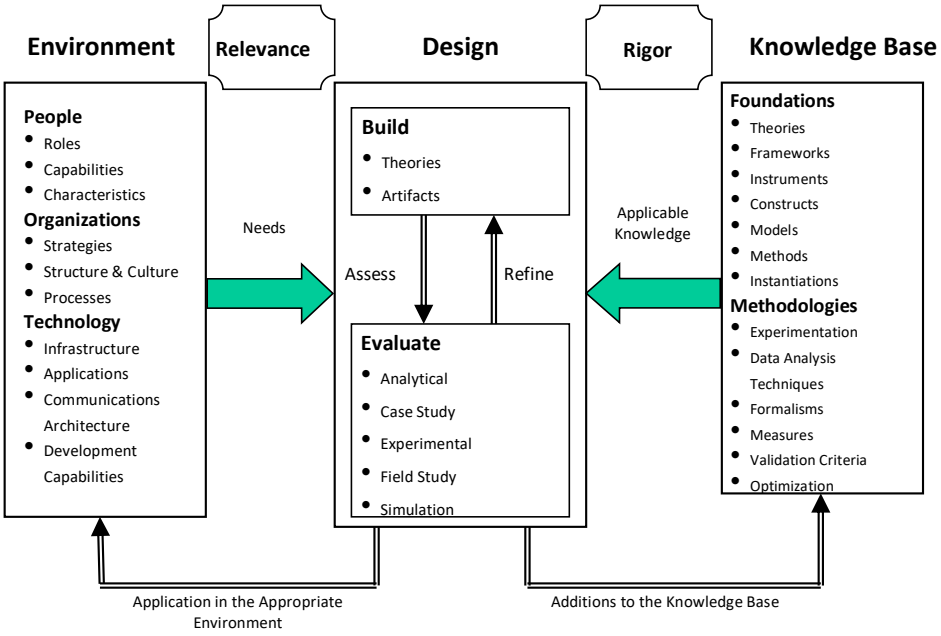

Abb. 3.1 DSR-Framework (Angelehnt an Hevner et al. [HMPR04]) [BHM20, 3]

aus dem Environment und das anwendbare Wissen ein, um ein neues Artefakt zu erstellen und zu evaluieren. Die neuen Artefakte werden im Kontext des Environments angewandt und neu gewonnenes Wissen der Knowledge Base hinzugefügt.

In dieser Arbeit setzt sich das Environment zusammen aus Organisationen, die IAM betreiben, und deren Verantwortliche für IAM Informationen über die Zielerreichung von IAM herausfinden wollen. Weiter bilden unter anderem Veröffentlichungen zu den Themenfeldern IAM, Reporting und Metriken im allgemeinen als auch im Bereich IT-Sicherheit die Knowledge Base.

Hevner et al. [HMPR04] haben Leitlinien zur Anwendung von DSR publiziert. Diese werden im Folgenden kurz vorgestellt sowie deren Anwendung in dieser Arbeit skizziert:

1. *Design as an Artifact:* Erstellung eines brauchbaren Artefakts.
   In dieser Arbeit wurde ein Artefakt – namens IAM Reporting Modul – konzipiert und implementiert, mithilfe dessen Metriken erfasst und berichtet werden können.

2. *Problem Relevance:* Technologiebasierte Lösung eines existierenden, relevanten Praxisproblems.
   Das IAM Reporting Modul erfasst und berichtet den aktuellen Zustand von IAM in Organisationen, um für Maßnahmen im IAM eine fundierte Entscheidungsbasis bereitzustellen.

3. *Design Evaluation:* Stringente Demonstration der Evaluation von Nützlichkeit, Qualität und Effizienz des erstellten Artefakts.
   In der Anforderungsanalyse wurden die Anforderungen an das IAM Reporting Modul aus dem Environment abgeleitet. Auf Grundlage dessen wurde die Evaluation des IAM Reporting Moduls durchgeführt.

4. *Research Contributions:* Beisteuerung von klaren und nachvollziehbaren Forschungsbeiträgen.
   Sowohl das IAM Reporting Modul als Artefakt als auch davon abgeleitete Gestaltungsprinzipien für Reporting Module werden durch diese Arbeit bereitgestellt.

5. *Research Rigor:* Anwendung von stringenten Methoden zur Erstellung und Evaluierung des Artefakts.
   Das IAM Reporting Modul ist beschrieben durch ein konzeptionelles Modell und implementiert als Prototyp. Daraus wurden allgemeine Gestaltungsprinzipien für Reporting Module abgeleitet. Dabei gaben die eingesetzten DSR- und Softwareentwicklungsmethoden Richtlinien zur Erstellung und Evaluierung des IAM Reporting Moduls vor.

6. *Design as a Search Process:* Inkrementelle Weiterentwicklung des Artefakts auf der Suche nach einer effektiven Lösung.
   Während der Erstellung der Arbeit wurde das IAM Reporting Modul in mehreren Iterationen entwickelt und anhand der Anforderungen evaluiert. Die verwendeten Werkzeuge und Methoden zur Softwareentwicklung unterstützen dabei den iterativen Ansatz.

7. *Communication of Research:* Veröffentlichung der Forschungsergebnisse für Fachpublikum.
   Die Ergebnisse werden im Rahmen dieser Masterarbeit an der Universität Regensburg veröffentlicht.

Charakteristisch für DSR ist der iterative Ansatz, welcher in der Abb. 3.1 bereits durch die Pfeile angedeutet ist und ebenfalls in den Leitlinien Erwähnung findet. Von Hevner [Hev07] wurde der iterative Ansatz nochmals genauer spezifiziert: Es handelt sich dabei um die Zyklen Relevance, Rigor und Design. Der Relevance-Zyklus verbindet das Environment mit dem Design. Der Rigor-Zyklus verbindet die Knowledge Base mit dem Design. Der zentrale Design-Zyklus iteriert zwischen den beiden Hauptaktivitäten, dem Erstellen und dem Evaluieren eines Artefakts.

Im Relevance-Zyklus werden Probleme und Potentiale im Environment identifiziert. Daraus abgeleitet werden nicht nur die Anforderungen, sondern auch die Akzeptanzkriterien für Lösungen. Mithilfe der Anforderungen wird im Design ein Artefakt erstellt und nach Abschluss des Designs im Environment anhand der Akzeptanzkriterien evaluiert. In dieser Arbeit wurde zuerst das Environment definiert und daraufhin wurden daraus die Problemstellung, der Wert der Lösung und die Anforderungen an ein IAM Reporting Modul aus dem Environment abgeleitet. Mit dem Ziel eine Lösung für die Problemstellung zu bieten, wurde das IAM Reporting Modul anhand der Anforderungen entwickelt und evaluiert.

Im Rigor-Zyklus werden existierendes Wissen und bestehende Artefakte der Knowledge Base genutzt, um ein neues innovatives Artefakt im Design zu erstellen und Wissen zu sammeln. Weiter wird sichergestellt, dass das gewonnene Wissen aus dem Design

veröffentlicht und in die Knowledge Base übernommen wird. In dieser Arbeit ist das bestehende Wissen über IAM, Reporting und Metriken in den Entstehungsprozess des IAM Reporting Moduls mit eingeflossen. Durch Veröffentlichung dieser Arbeit werden das Wissen über das IAM Reporting Modul und die abgeleiteten Gestaltungsprinzipien in die Knowledge Base übernommen.

Der Design-Zyklus ist der interne Zyklus des Designs und der Mittelpunkt eines jeden DSR Projekts. Dieser Zyklus von Aktivitäten wiederholt sich schneller als die anderen Zyklen. Er iteriert zwischen der Erstellung eines Artefakts und der Evaluierung desselben sowie dem anschließenden Feedback zur weiteren Verfeinerung des Designs. Wie bereits im sechsten Leitfaden beschrieben, wurde dieser Zyklus für das IAM Reporting Modul mehrmals durchlaufen.

Kapitel 4

Anforderungsanalyse

Im Rahmen der Anforderungsanalyse werden die Anforderungen an das IAM Reporting Modul gesammelt und definiert. Diese dienen als Grundlage für das Design und die Implementierung des Systems. Ziel der Anforderungsanalyse ist es, die Anforderungen an das System zu verstehen. Optionen, wie die Anforderungen zu einem späteren Zeitpunkt umgesetzt werden, spielen dabei noch keine oder nur eine sehr untergeordnete Rolle. [Kle18]

Es gibt eine Reihe von Möglichkeiten, wie bei der Anforderungsanalyse vorgegangen wird. Für diese Arbeit wurden die folgenden Aktivitäten ausgewählt: Die Anforderungsanalyse beginnt mit dem übergreifenden Ziel und endet mit der Formulierung der einzelnen Anforderungen. Gestartet wird die Anforderungsanalyse somit mit der Zielbestimmung. Dabei wird festgehalten, was entwickelt werden soll und welche wesentlichen Funktionen das System umfassen soll. Auch Qualitätsziele spielen dabei eine Rolle. Daraufhin erfolgt die Analyse des Systemkontextes und der relevanten Stakeholder. Im Systemkontext werden unter anderem alle Systeme, Akteure, Prozesse und Gesetze nahe des Systems betrachtet. Alle Interessenten an dem System werden in der Stakeholderanalyse zusammen mit deren individuellen Interessen und Zielen erfasst. Darauf aufbauend werden die wesentlichen Use Cases aufgenommen. Die Use Cases werden sowohl graphisch dargestellt als auch textuell beschrieben. Im letzten Schritt werden aus den Use Cases als auch aus der Systemkontext- und Stakeholderanalyse die Anforderungen abgeleitet.

4.1 Zielbestimmung

Ziel der Arbeit ist es ein Reporting Modul für IAM-Metriken zu entwickeln, um den aktuellen Zustand von IAM in Organisationen darzustellen. Die wesentlichen Funktionen sind: Metriken sollen auf Basis von Messwerten toolgestützt erstellt und berechnet werden. Die Messwerte werden hierzu aus verschiedenen Datenquellen importiert. Basierend auf dem in den Metriken hinterlegten Informationsbedarf und der Zielgruppe werden die Metriken in einem Dashboard dargestellt. Die Qualitätsziele können wie folgt zusammengefasst werden: Das erarbeitete Konzept ist mithilfe eines Prototyps zu verifizieren. Dabei ist die

funktionale Eignung der zuvor dargelegten Funktionen zu überprüfen: Im wesentlichen sollen IAM-Metriken korrekt berechnet werden können. Ein generisches Datenmodell ist zu konzipieren, das neben dem Reporting von IAM-Metriken auch für andere Metriken eingesetzt werden könnte. Zuletzt sind die Schnittstelle zu Datenquellen, aus denen die Messwerte importiert werden, so zu modularisieren, sodass unterschiedliche Typen von Datenquellen nach und nach hinzugefügt werden können.

4.2 Systemkontext

Bei der Systemkontextanalyse findet die Umgebung des Systems Betrachtung. Der Systemkontext ist der Teil der Umgebung eines Systems, der für die Definition als auch für das Verständnis der Anforderungen des zu entwickelnden Systems relevant ist. [Gli22] Das Systemkontextdiagramm in Abb. 4.1 setzt das IAM Reporting Modul in Bezug zu seiner Umgebung.

Im Systemkontextdiagramm gibt es zwei Grenzen. Erstens die Systemgrenze zwischen dem geplanten System und seiner Umgebung. An der Systemgrenze sind die externen Schnittstellen zwischen dem System und dem Systemkontext zu definieren. Alle Akteure, Geschäftsprozesse, Systeme und Dokumente, welche das System umgeben und dieses direkt beeinflussen, befinden sich im Systemkontext. Zweitens gibt es die Systemkontextgrenze. Diese schafft eine Grenze zwischen dem Systemkontext und den Entitäten, welche das System nicht direkt beeinflussen und nicht relevant für das System und dessen Anforderungen sind. [Gli22]

Die Akteure sind einerseits reguläre Benutzer des Systems, die Metriken und Reports sichten. Zusätzlich gibt es administrative Benutzer, die im System Datenquellen anbinden, Messwerte importieren und Metriken erstellen.

Standards wie die ISO/IEC 27001 Reihe Informationssicherheit [Int13], ISO/IEC 24760 Identity Management [Int19b] und ISO/IEC 29146 Access Management [Int16a] beschreiben unter anderem den Einsatz von Metriken und eine Referenzarchitektur für Identity Management und Access Management.

Um die beteiligten Systeme zu definieren, wurde die Beispiel-IAM-Architektur vom Federal Identity, Credential, and Access Management (FICAM) herangezogen [Ide21]. Das IAM Reporting Modul interagiert direkt mit IAM-Systemen über Schnittstellen. Hingegen nicht direkt verbunden ist das IAM Reporting Modul mit den Quell- und Endsystemen von IAM-Systemen. Es wird davon ausgegangen, dass sich die für das Reporting benötigten Messwerte entweder in den IAM-Systemen ermittelt werden oder bereits dorthin importiert wurden.

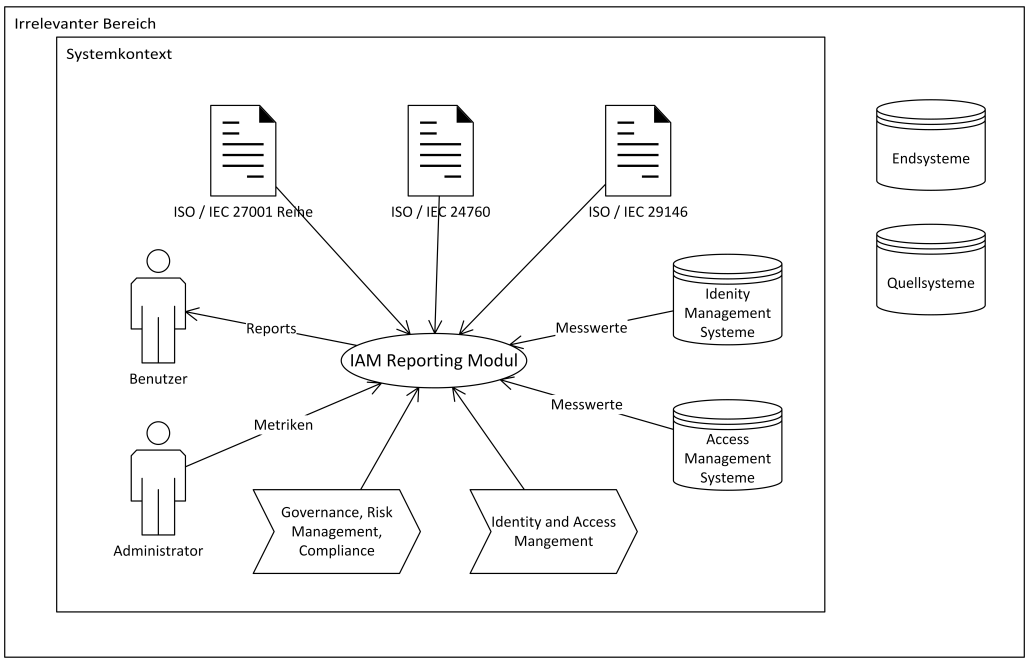

Abb. 4.1 Systemkontextdiagramm

4.3 Stakeholder

Bei der Stakeholderanalyse sind die Interessenten des Systems zu identifizieren und zu dokumentieren. In der Systemkontextanalyse wurden zwei Akteure identifiziert: Benutzer und Administratoren. Diese gilt es in der Stakeholderanalyse weiter zu unterteilen. Ein Stakeholder ist eine Person oder eine Organisation, die von dem System betroffen ist oder die einen Einfluss auf die Anforderungen eines Systems hat [GW07]. In der Tab. 4.1 sind durch Osmanoglu [Osm14, 57 f.] vorgeschlagene und weitere identifizierte Stakeholder von IAM aufgelistet und beispielhafte Interessen und Ziele des Stakeholders vermerkt.

Die Tabelle zeigt die unterschiedlichen Interessen der Stakeholder: Unter anderem sollen die Prozesse optimiert, die Compliance eingehalten und überprüft, das Risikomanagement betrieben, die Kosten gesenkt und die Sicherheit überwacht werden. Die im IAM Reporting Modul bereitgestellten Metriken sollten daher auf die Informationsbedarfe des Benutzers angepasst werden. Um die Rechte der Mitarbeiter zu berücksichtigen, sind die Interessen und Ziele des Datenschutzes, der Rechtsabteilung und ggf. der Arbeitnehmervertretung zu beachten.

Stakeholder	Interessen und Ziele
Auditoren	• Bewertung der Konformität mit Regularien und Gesetzen
Arbeitnehmervertretung	• Rechte der Mitarbeiter (z. B. Schutz der Mitarbeiter vor Leistungsmessung)
Datenschutz	• Sicherstellung der Konformität mit den geltenden Datenschutzgesetzen
Dienstleister	• Erbringung von vertraglich geregelten Leistungen
Führungskräfte	• Überprüfung der Berechtigungen ihrer Mitarbeiter
IAM-Systemverantwortliche	• Administration des IAM-Systems
IT-Helpdesk	• Einhaltung von Service Level Agreements • Lösung von Problemen der Benutzer
IT-Sicherheit	• Überwachung der Sicherheit aller IT-Systeme • Erkennung von Anomalien und potentiellen Cyber-Angriffen
Management	• Verantwortung von Fortschritt, Kosten und Risiken • Budgetierung des IAM-Systems und -Projekts
Personalwesen	• Überprüfung der Daten der Mitarbeiter auf Vollständigkeit • Beobachtung und Verbesserung des Identity Lifecycles
Sicherheit	• Überprüfung der Identität der Benutzer
System- und Datenverantwortliche	• Überwachung der Entwicklung der Zugriffe und Zugriffsberechtigungen in ihrem Verantwortungsbereich • Verantwortung für reibungslosen Betrieb ihrer Systeme • Verantwortung von IT-Sicherheit ihrer Systeme
Rechtsabteilung	• Erstellung von Richtlinien • Beantwortung von Rechtsfragen

Tab. 4.1 Stakeholder

4.4 Use Cases

Use Cases modellieren eine Reihe möglicher Interaktionen zwischen externen Akteuren und einem System. Die Interaktionen erbringen dabei Nutzen für die beteiligten Akteure. Personen, Systeme oder technische Geräte, die in einer bestimmten Rolle agieren, sind in diesem Kontext Akteure. Use Cases beschreiben somit das System aus der Perspektive der Akteure: Ein Use Case beschreibt eine Funktionalität, welche das System für den Akteur bereitstellt. [Gli22]

Use Cases lassen sich in Unified Modeling Language (UML) Use Case-Diagrammen

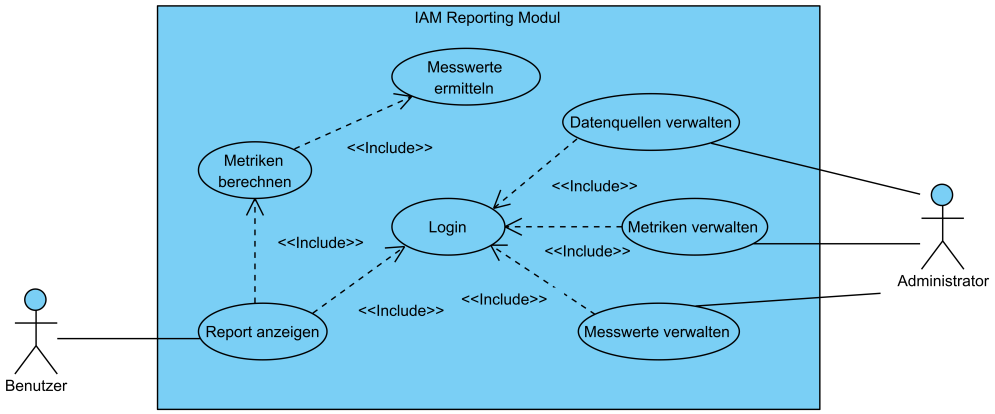

Abb. 4.2 Use Cases des IAM Reporting Moduls

darstellen. In Abb. 4.2 sind die beiden Akteure Benutzer und Administrator und die Use Cases in einem Use Case-Diagramm abgebildet.

Alle Use Cases schließen den Use Case *Login* mit ein. Die Benutzer und den Administratoren benötigen unterschiedliche Informationen und Funktionalitäten, deshalb muss das IAM Reporting Modul zwischen den verschiedenen Anwenderrollen unterscheiden können. Hierbei bietet es sich an, die Authentisierung und die Authentifizierung des Anwenders durchzuführen.

Für den Benutzer konnte ein Use Case identifiziert werden: *Report anzeigen*. Der Benutzer möchte sich zu einem Informationsbedarf einen Report anzeigen lassen. Dazu wählt er einen Informationsbedarf aus und das IAM Reporting Modul generiert einen Report mit Metriken zu dem gewähltem Informationsbedarf.

Damit Metriken im Report angezeigt werden können, müssen diese zuvor aus Messwerten und anderen *Metriken berechnet* werden. Dieser Vorgang soll automatisch durch das IAM Reporting Modul in einer zuvor festgelegten Frequenz erfolgen. Die zur Berechnung benötigte Formel ist ebenso im Voraus festzulegen.

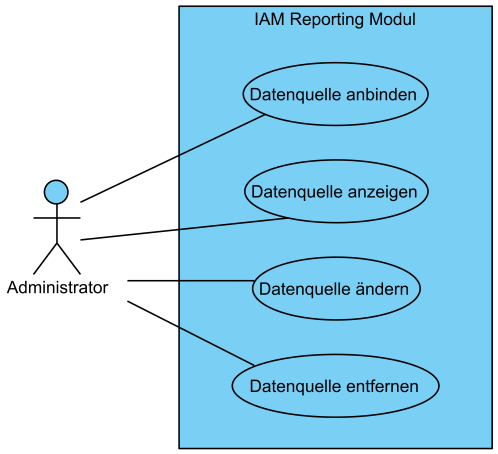

Abb. 4.3 Teilfunktionalität: Datenquellen verwalten

Damit die Metriken wiederum berechnet werden können, müssen Daten zu den in den Formeln referenzierten Messwerte vorliegen. Hierzu soll das IAM Reporting Modul auf Basis einer festgelegten Frequenz *Messwerte* in den jeweiligen Datenquellen *ermitteln*.

Der Administrator benötigt Funktionalitäten, um die *Datenquellen*, *Messwerte* und *Metriken* zu *verwalten* und zu konfigurieren. Diese Funktionalitäten lassen sich noch weiter in Teilfunktionalitäten verfeinern. Um die Übersichtlichkeit des Use Case Diagramms zu wahren, wurden die Teilfunktionalitäten jeweils als ein Use Case dargestellt.

Der erste Use Case des Administrators ist die *Verwaltung von Datenquellen* (siehe Abb. 4.3). Das IAM Reporting Modul soll Messwerte in verschiedenen Datenquellen ermitteln. Der Administrator bindet dazu Datenquellen an, indem er beispielsweise Verbindungsinformationen der Datenquellen angibt. Weiter lässt sich der Administrator die Konfiguration bereits angebundener Datenquellen anzeigen, um diese zu überprüfen. Hat sich die Datenquelle oder deren Konfiguration verändert, ändert der Administrator die Konfiguration der Datenquellen ab. Wird die Datenquelle nicht mehr benötigt, wird diese durch den Administrator entfernt.

Der zweite Use Case des Administrators ist die *Verwaltung von Messwerten* (siehe Abb. 4.4). Der Administrator legt die Konfiguration von Messwerten an oder lässt sich die Konfiguration der Messwerte anzeigen. In der Konfiguration wird beispielsweise festgelegt, in welcher Datenquelle die Messungen ermittelt werden und in welcher Frequenz die Messung stattfindet. Verändert sich die Konfiguration eines Messwerts, ist diese durch den Administrator anzupassen. Wird ein Messwert nicht mehr benötigt, werden die Konfiguration des Messwerts und die ermittelten Werte des Messwerts durch den Administrator gelöscht.

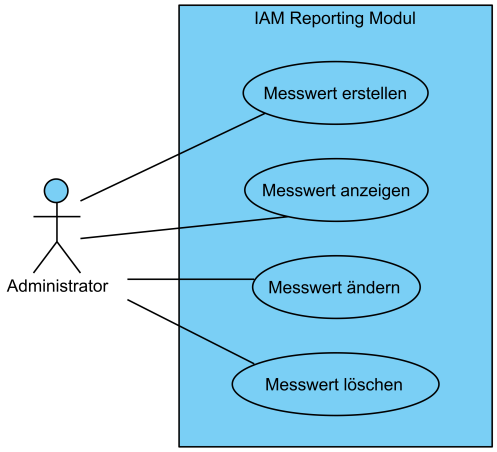

Abb. 4.4 Teilfunktionalität: Messwerte verwalten

Der dritte Use Case des Administrators ist die *Verwaltung von Metriken* (siehe Abb. 4.5). Die Konfiguration der Metriken wird durch den Administrator erstellt. Dabei legt dieser unter anderem fest, aus welchen Messwerten und Metriken sich die Metrik zusammensetzt, wer die Zielgruppe der Metrik ist und in welcher Frequenz diese berechnet werden. Über die Zeit können sich die benötigten Metriken verändern. Deshalb ist es notwendig, dass der Administrator die Konfiguration der Metriken einsehen und verändern kann. Metriken können ebenso gänzlich nicht mehr benötigt werden. Administratoren müssen somit die Konfiguration der Metriken und die berechneten Metriken löschen können.

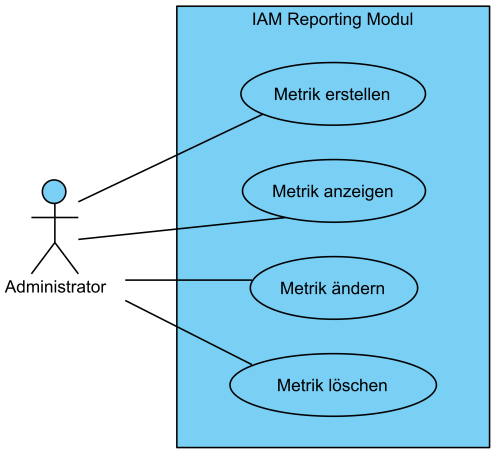

Abb. 4.5 Teilfunktionalität: Metriken verwalten

4.5 Domänenmodell

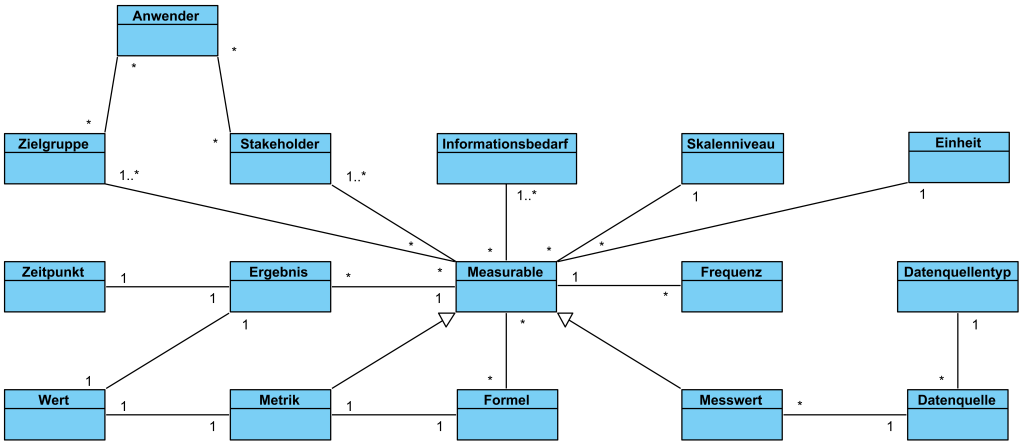

Abb. 4.6 Domänenmodell

Die Abb. 4.6 zeigt das Domänenmodell des IAM Reporting Moduls. Zentral in der Mitte befindet sich das Measureable, von dem sich sowohl die Metrik als auch der Messwert ableiten. Das Measureable hat eine oder mehrere Zielgruppen und Stakeholder. Es wiedergibt Informationen zu einem oder mehreren Informationsbedarfen. Weiter ist die Frequenz, in der es ermittelt oder berechnet wird, angegeben. Die Werte des Measureable besitzen ein Skalenniveau und eine Einheit. In den Ergebnissen sind die Werte und die Zeitpunkte des Measureables abzulegen. Für eine Metrik kann ein Zielwert definiert werden. Eine Metrik kann über eine Formel aus Messwerten und anderen Metriken berechnet werden. Die Messwerte stammen dabei aus Datenquellen. Die Datenquellen sind je nach Typ zu unterscheiden.

4.6 Anforderungen

Bevor das IAM Reporting Modul konzipiert und implementiert wird, müssen die Anforderungen an das System definiert werden. Als eine Anforderung bezeichnet Balzert [Bal09, 455 f.] eine Eigenschaft, die von den Stakeholdern von einem Softwaresystem erwartet wird. Anhand der dokumentierten Anforderungen lässt sich im Rahmen der Evaluation überprüfen, ob das System alle gewünschten Eigenschaften der Stakeholder erfüllt. Zudem beantwortet die Definition der Anforderungen die Forschungsfrage RQ1.

Balzert [Bal09] unterscheidet zwischen mehreren Typen von Anforderungen: funktionale Anforderungen, nicht-funktionale Anforderungen und Rahmenbedingungen. Funktionale Anforderungen beschreiben die von einem System bereitzustellenden Funktionen oder Services [IEE90]. Für nicht-funktionalen Anforderungen gibt es in der Literatur keine einheitliche Definition [Gli07; MZN10]. In dieser Arbeit wird der Begriff wie folgt eingesetzt: Nicht-funktionale Anforderungen beschreiben Bedingungen, die nicht einzelne Funktionen, sondern häufig das gesamte System betreffen [Som16, 105]. Nicht-funktionale Anforderungen beschreiben oft qualitative Aspekte des Systems oder tech-

nische Anforderungen an das System, wie beispielsweise Performance, Zuverlässigkeit, Benutzbarkeit, Sicherheit oder Wartbarkeit [MZN10]. Rahmenbedingungen legen technische und organisatorische Restriktionen für das System und den Entwicklungsprozess fest [Bal09, 459].

Im Institute of Electrical and Electronics Engineers (IEEE) Standard 830-1998 [IEE98] werden Eigenschaften von guten Anforderungen beschrieben. Darunter zählen unter anderem Korrektheit, Eindeutigkeit, Vollständigkeit, Konsistenz, Klassifizierbarkeit nach Wichtigkeit oder Stabilität, Überprüfbarkeit, Modifizierbarkeit und Verfolgbarkeit.

Für die einheitliche und klare Formulierung von Anforderungen, gibt es Anforderungsschablonen z. B. von IEEE [IEE98] oder Rupp und Günther [RG20]. In dieser Arbeit wird auf die Anforderungsschablonen von Rupp und Günther zurückgegriffen. Diese definieren eine Anforderungsschablone als einen "Bauplan, der die Struktur eines einzelnen Anforderungssatzes festlegt". Je nach Typ der Anforderung (funktional, nicht-funktional oder Rahmenbedingung) gibt es unterschiedliche Anforderungsschablonen.

In den Anforderungsschablonen gibt es bestimmte Schlüsselwörter, um die Verbindlichkeit der Anforderung darzustellen. Rupp und Günther [RG20] schlagen folgende die Verwendung der drei Schlüsselwörter MUSS, SOLLTE und WIRD vor.

MUSS: Die Umsetzung der Anforderung ist verpflichtend und muss erfüllt werden.

SOLLTE: Die Umsetzung der Anforderung ist nicht verpflichtend und muss nicht erfüllt werden. Trotzdem ist die Umsetzung wünschenswert und erhöht die Zufriedenheit der Stakeholder.

WIRD: Die Umsetzung der Anforderung ist erst in Zukunft verpflichtend. Diese Anforderungen dienen dazu bei der Konzeption des Systems zukünftige Anforderung zu berücksichtigen.

Die Anforderungen wurden aus den zuvor dargelegten Use Cases, dem Systemkontext, den Stakeholdern und dem allgemeinen theoretischen Hintergrund zu IAM sowie Reporting und Metriken abgeleitet und werden im Folgenden vorgestellt.

4.6.1 Funktionale Anforderungen

Die funktionalen Anforderungen sind orientiert an den Use Cases gruppiert.
Report anzeigen.

/F10/ Das IAM Reporting Modul MUSS dem Benutzer die Möglichkeit bieten, Metriken für einen Informationsbedarf in einem Dashboard darzustellen.

/F11/ Das IAM Reporting Modul SOLLTE die Metriken in Diagrammen visualisiert darstellen.

/F12/ Das IAM Reporting Modul WIRD dem Benutzer die Möglichkeit bieten, den Verlauf einer Metrik darzustellen.

Datenquellen verwalten.

/F20/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die Konfiguration von Datenquellen anzulegen.

/F21/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, in der Konfiguration von Datenquellen Dateien als Datenquelle festzulegen.

/F22/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, in der Konfiguration von Datenquellen Datenbanken als Datenquelle festzulegen.

/F23/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, in der Konfiguration von Datenquellen manuelle Werte als Datenquelle festzulegen.

/F24 Das IAM Reporting Modul MUSS die Konfiguration von Datenquellen persistent speichern.

/F25/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, die Konfiguration von Datenquellen zu editieren.

/F26/ Falls keine Messwerte auf eine Datenquelle verweisen, SOLLTE das IAM Reporting Modul dem Administrator die Möglichkeit bieten, die Konfigurationen von Datenquellen zu löschen.

Messwerte verwalten.

/F30/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, die Konfiguration von Messwerten anzulegen.

/F31/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, die Konfiguration von Messwerten zu editieren.

/F32/ Falls keine Formeln von Metriken auf einen Messwert verweisen, SOLLTE das IAM Reporting Modul dem Administrator die Möglichkeit bieten, die Konfiguration eines Messwerts zusammen mit den ermittelten Messwerten zu löschen.

/F33/ Falls die Konfiguration eines Messwerts gelöscht wird, SOLLTE das IAM Reporting Modul die dazugehörigen ermittelten Messwerten löschen.

/F34/ Das IAM Reporting Modul MUSS die Konfiguration von Messwerten persistent abspeichern.

/F35/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die Zeitpunkte der Ermittlung eines Messwertes oder die Zeitabstände zwischen den Ermittlungen eines Messwertes in einer Konfiguration festzulegen.

Metriken verwalten.

/F40/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die Konfiguration von Metriken anzulegen.

/F41/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die Formel zur Berechnung der Metrikwerte in der Konfiguration anzugeben.

/F42/ Das IAM Reporting Modul WIRD dem Administrator die Möglichkeit bieten, die Formel der Metriken unterstützt durch einen visuellen Editor zu erstellen.

/F43/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die Zeitpunkte der Berechnung einer Metrik oder die Zeitabstände zwischen den Berechnungen einer Metrik in einer Konfiguration festzulegen.

/F44/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, die Konfiguration von Metriken zu editieren.

/F45/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, die Konfiguration einer Metrik zusammen mit den berechneten Metrikwerten zu löschen.

/F46/ Falls die Konfiguration einer Metrik gelöscht wird, SOLLTE das IAM Reporting Modul die dazugehörigen berechneten Metrikwerte löschen.

/F47/ Das IAM Reporting Modul MUSS die Konfiguration von Metriken persistent speichern.

/F48/ Das IAM Reporting Modul MUSS dem Administrator die Möglichkeit bieten, die mathematische Grundrechenarten Addition, Subtraktion, Multiplikation und Division in der Formel zu verwenden.

/F49/ Das IAM Reporting Modul SOLLTE dem Administrator die Möglichkeit bieten, weitere Methoden wie Summe, Produkt, Potenz oder Wurzel in der Formel zu verwenden.

Metriken berechnen.

/F50/ Das IAM Reporting Modul MUSS fähig sein, Metriken auf Basis einer durch den Administrator angegebenen Formel und in der festgelegten Frequenz berechnen.

/F51/ Das IAM Reporting Modul MUSS berechnete Metriken persistent speichern.

Messwerte ermitteln.

/F60/ Das IAM Reporting Modul MUSS fähig sein, Messwerte in der festgelegten Frequenz in Datenquellen zu ermitteln.

/F61/ Das IAM Reporting Modul MUSS ermittelte Messwerte persistent abspeichern.

Login.

/F70/ Das IAM Reporting Modul WIRD eine Autorisierung der Zugriffe durchführen.

/F71/ Das IAM Reporting Modul WIRD dem Administrator die Möglichkeit bieten, einem Benutzer ein oder mehrere Zielgruppen zuzuordnen.

4.6.2 Nicht-funktionale Anforderungen

Zur Kategorisierung der nicht-funktionalen Anforderungen wurde die ISO/IEC 25010 [Int11] herangezogen. Diese unterteilt nicht-funktionale Anforderungen in Wartbarkeit, Kompatibilität, funktionale Eignung, Effizienz, Benutzbarkeit, Zuverlässigkeit und Übertragbarkeit.

Wartbarkeit.

/N10/ Weitere bislang nicht aufgeführte Typen von Datenquellen MÜSSEN zu einem späteren Zeitpunkt zum IAM Reporting Modul hinzufügbar sein.

/N11/ Das IAM Reporting Modul SOLLTE zur Vereinfachung von Fehleranalysen wichtige Ereignisse in ein Log schreiben.

/N12/ Das Dashbaord MUSS um zukünftige Funktionen erweiterbar sein.

/N13/ Der Quellcode des IAM Reporting Moduls SOLLTE dokumentiert sein.

/N14/ Das IAM Reporting Modul MUSS veränderbar sein.

/N15/ Das IAM Reporting Modul MUSS auf einer zeitgemäßen Systemarchitektur basieren.

Kompatibilität.

/N20/ Das IAM Reporting Modul SOLLTE an IAM Systeme anzubinden sein.

Funktionale Eignung.

/N30/ Das IAM Reporting Modul MUSS Metriken anhand der hinterlegten Formel korrekt berechnen.

Effizienz.

/N40/ Wenn zehn Messwerte und drei Metriken im IAM Reporting Modul konfiguriert sind, SOLLTE das IAM Reporting Modul auf aktuellen Standardclients ausführbar sein.

Benutzbarkeit.

/N50/ Die Reaktionszeit des IAM Reporting Moduls auf eine Benutzereingabe SOLLTE kleiner gleich einer Sekunde sein.

/N51/ Wenn der Report drei Metriken umfasst, SOLLTE die Ladezeit des Reports kleiner gleich zehn Sekunden sein.

/N52/ Die Benutzeroberfläche des IAM Reporting Moduls SOLLTE modern sein.

/N53/ Das Design des IAM Reporting Moduls SOLLTE responsive sein.

/N54/ Das IAM Reporting Modul MUSS den Benutzer auf fehlerhafte Benutzereingaben hinweisen.

Zuverlässigkeit.

/N60/ Das IAM Reporting Modul SOLLTE Systemfehler behandeln.

/N61/ Das IAM Reporting Modul WIRD nach einem Absturz oder Datenverlust verlorene gegangene Messwerte neu importieren und Metriken neu berechnen.

Übertragbarkeit.

/N70/ Das IAM Reporting Modul SOLLTE für andere Reporting Anwendungsfälle außerhalb des IAM, die auf Metriken basieren, übertragbar sein.

Sicherheit.

/N80/ Das IAM Reporting Modul MUSS dem Benutzer nur solche Metriken im Dashboard anzeigen, welche der Zielgruppe des Benutzers entsprechen.

4.6.3 Rahmenbedingungen

Die meisten Rahmenbedingungen ergeben sich aus mit dem Betreuer der Arbeit abgestimmten Anforderungen an die Arbeit. Weitere Rahmenbedingungen wurden, ebenso wie die funktionalen und nicht-funktionalen Anforderungen, aus den vorherigen Sektionen und Kapiteln abgeleitet.

/R01/ Die Anforderungen an das IAM Reporting Modul SOLLTEN vollständig formuliert sein.

/R02/ Das Datenmodell des IAM Reporting Moduls MUSS generisch sein.

/R03/ Ein Konzept für das IAM Reporting Moduls MUSS erstellt sein.

/R04/ Das IAM Reporting Modul MUSS als Prototyp implementiert sein.

/R05/ Die Interaktion der Benutzer mit dem IAM Reporting Modul SOLLTE mit einem Webbrowser möglich sein.

/R06/ Die eingesetzte Software oder eingesetzten Softwarebibliotheken des IAM Reporting Moduls MÜSSEN Open Source sein.

/R07/ Die Versionen der eingesetzten Software oder eingesetzten Softwarebibliotheken des IAM Reporting Moduls MÜSSEN gleich dem neusten Long Term Support (LTS) Release sein.

/R08/ Das IAM Reporting Modul WIRD konform zu den geltenden Datenschutzgesetzen sein.

Kapitel 5

Design

Nach der Sammlung der Anforderungen ist der nächste Schritt das Design des IAM Reporting Moduls. Das Design spielt eine wesentliche Rolle, um die Forschungsfrage RQ2 zu beantworten. Dabei werden in diesem Kapitel das Datenmodell, die Prozesse und die Benutzeroberfläche unter Berücksichtigung der Anforderungen entworfen.

Ziel des Softwaredesigns ist es, für die gegebenen fachlichen Anforderungen eine passende Softwarelösung zu entwickeln [Bal11]. Dazu werden die Architektur der zu implementierenden Software, die Systemkomponenten, die Schnittstellen zwischen den Komponenten und weitere Eigenschaften des Systems wie beispielsweise die eingesetzten Algorithmen beschrieben [Int18b; Som16, 56]. Der Begriff Softwaredesign beschreibt dabei sowohl den Prozess als auch das Ergebnis des Prozesses [Int18b]. Softwaredesign ist meist ein iterativer Prozess, bei dem der Detailgrad mit jeder Iteration steigt [Som16, 56]. Typische Aktivitäten im Softwaredesign sind das Design der Architektur, das Design der Datenstrukturen, das Design der Schnittstellen zwischen Systemkomponenten und die Auswahl von bestehenden Komponenten oder das Design von neuen Komponenten [Som16, 56 f.]. Dieser iterative Prozess des Softwaredesigns harmoniert mit dem iterativen Ansatz der DSR Methodologie.

Das Kapitel Design ist wie folgt strukturiert: Als Erstes wird die Architektur des IAM Reporting Moduls festgelegt. Daraufhin folgt die Modellierung des logischen Aufbaus des IAM Reporting Moduls in Form eines Datenmodells. Anschließend werden die in dem IAM Reporting Modul abgebildeten Prozesse entworfen. Zuletzt wird die zur Interaktion mit dem Benutzer benötigte Benutzeroberfläche des IAM Reporting Moduls gestaltet.

5.1 Architektur

Wie in der Anforderungsanalyse im vorangegangenen Kapitel dargelegt, befinden sich sowohl Benutzer und Administratoren als auch externe Systeme wie beispielsweise IAM-Systeme im Systemkontext. Die Benutzer und Administratoren sollen über einen Webbrowser mit dem IAM Reporting Modul interagieren. Die externen Systeme sollen über Schnittstellen angebunden werden.

Wie in Abb. 5.1 skizziert, ist eine Webanwendung zu erstellen. Auf diese können

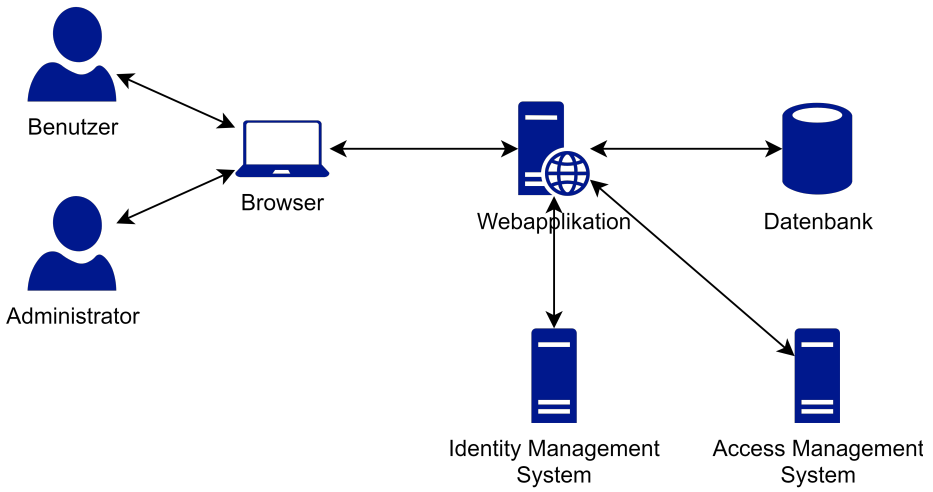

Abb. 5.1 Architektur

Benutzer und Administratoren über einen Webbrowser zugreifen. Die Datenhaltung der Webanwendung soll eine Datenbank übernehmen. Die Webanwendung und die Datenhaltung sind dabei logisch voneinander zu trennen. Physisch können die Webanwendung und die Datenhaltung verteilt auf mehreren Systemen sein als auch auf einem System parallel laufen.

Externe Systeme, wie beispielsweise IAM-Systeme, können über verschiedenste Arten von Schnittstellen als Datenquelle an das IAM Reporting Modul angebunden werden. In den Anforderungen wurden Datenbanken und Dateien (vgl. funktionale Anforderungen F21 und F22) als Arten von Schnittstellen identifiziert. Die einzelnen Arten von Schnittstellen sind generisch zu erstellen, um ohne größere Änderung weitere Systeme anbinden zu können.

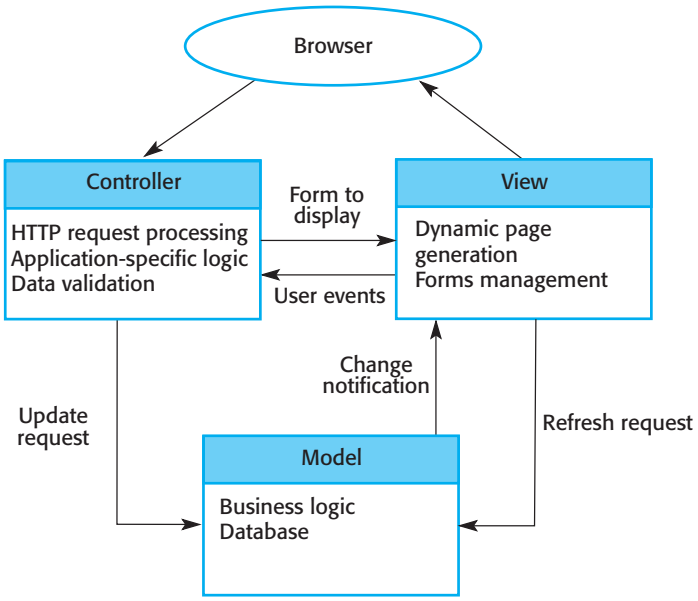

Abb. 5.2 MVC [Som16, 177]

Für die Webanwendung bietet sich das Designmuster Model-View-Controller (MVC) an. MVC separiert die Präsentation und die Interaktion von den Daten eines Systems. Das System ist dabei in drei Komponenten unterteilt, die miteinander interagieren (siehe Abb. 5.2). Die *Model*-Komponente ist verantwortlich für die Daten des Systems und den damit verbundenen Datenoperationen. Die *View*-Komponente ist verantwortlich für die Präsentation der Daten an den Benutzer. Die *Controller*-Komponente ist verantwortlich für die Steuerung der Benutzerinteraktion und gibt diese an Model und View weiter. [Som16, 176 f.]

5.2 Datenmodell

Das Datenmodell beschreibt die statische Sicht des IAM Reporting Moduls. Für diese Arbeit wird das Datenmodell mithilfe eines Unified Modeling Language-(UML)-Klassendiagramms dargestellt. Das Domänenmodell aus der Anforderungsanalyse stellt eine Vorlage für das Datenmodell der Designphase dar. Die im Domänenmodell beschriebenen Klassen sind während der Designphase um Attribute ergänzt worden. Das gesamte Datenmodell ist im Anhang A zu sehen. Die Namen der Klassen und Attribute sind in englischer Sprache formuliert. Die Beschreibung der einzelnen Klassen des Modells folgt in den nachfolgenden Absätzen.

Metrik und Messwert. *Metric* (Metrik) und *Measurement* (Messwert) sind die zentralen Klassen des Datenmodells (siehe Abb. 5.3) und finden Erwähnung in mehreren funktionalen Anforderungen (vgl. F10-F12, F30-F35, F40-F49). Aufgrund gemeinsamer Attribute leiten sich beide Klassen von der abstrakten Klasse *Measurable* ab und besitzen die Attribute *name* (Name) und *description* (Beschreibung) vom Datentyp *string*. Die Klasse *Metric* besitzt ein weiteres Attribut, in dem der Zielwert (*targetValue*) für die Metrik definierbar ist.

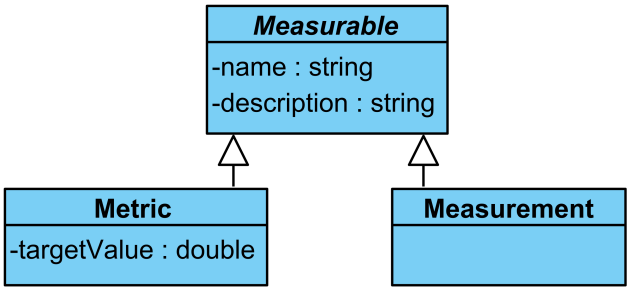

Abb. 5.3 Klassendiagramm: Metrik und Messwert

Formel. Eine Metrik wird anhand einer Formel aus ein oder mehreren Metriken oder Messwerten berechnet (siehe Abb. 5.4). Dies geht aus den funktionalen Anforderungen F41 und F50 hervor. Die Klasse *Metric* besitzt daher eine *Formula* und die Klasse *Formula* referenziert ein oder mehrere *Measurables*. Die Klasse *Formula* besitzt ein Attribut

formula vom Datentyp *string*, in dem die tatsächliche Formel zur Berechnung hinterlegt wird.

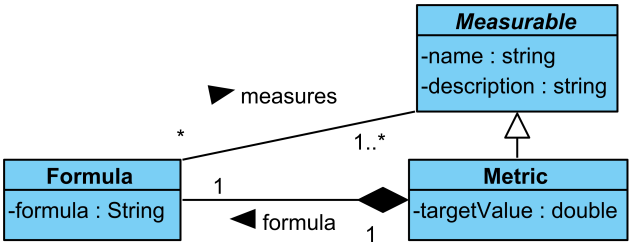

Abb. 5.4 Klassendiagramm: Formel

Frequenz. Metriken und Messwerte sollen gemäß funktionaler Anforderung F35 und F43 zu bestimmten Zeitpunkten oder in bestimmten Zeiträumen berechnet bzw. ermittelt werden. Dazu besitzt die abstrakte Klasse *Measurable* eine *Frequency* (siehe Abb. 5.5). Die Klasse *Frequency* hat ein Attribut *duration* vom Datentyp *Duration*, in dem die Dauer zwischen zwei Berechnungen oder Messungen hinterlegt wird. Die Dauer ermöglicht die Angabe von Zeiträumen, jedoch nicht von bestimmten Zeitpunkten. In einer weiteren Designiteration ist es denkbar, zusätzlich zur *duration* noch ein weiteres Attribut einzuführen, mit dem konkrete Zeitpunkte festgelegt werden können. Dieses Attribut könnte beispielsweise mit einem cron Ausdruck, der aus dem Unix Umfeld bekannt ist, umgesetzt werden [IEE18].

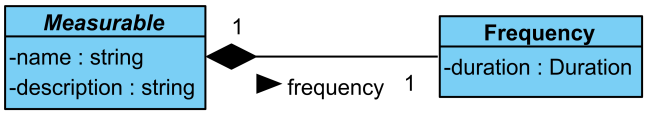

Abb. 5.5 Klassendiagramm: Frequenz

Ergebnis. Ergebnisse von Berechnungen der Metriken und Ermittlungen der Messwerte sind persistent zu speichern (vgl. funktionale Anforderung F51 und F61). Die abstrakte Klasse *Measurables* besitzt deshalb mehrere *Results* (siehe Abb. 5.6). Die Klasse *Result* hat das Attribut *value* vom Datentyp *double*, um den Wert der Berechnung oder der Ermittlung zu speichern. Für einen simpleren Report (vgl. funktionale Anforderung F10), bei dem der aktuellste Wert angezeigt wird, wäre es ausreichend den Wert des Ergebnisses bei jeder Berechnung oder Messung zu überschreiben. Sollen die Metriken – wie in der funktionalen Anforderung F12 beschrieben – im Verlauf dargestellt werden, so sind ältere Ergebnisse ebenso vorzuhalten und mit einem Zeitstempel zu versehen. Deshalb besitzt die Klasse *Result* ein weiteres Attribut *pointInTime* vom Datentyp *Date*, in dem der Zeitpunkt der Berechnung oder Messung festgehalten wird.

Datenquelle. Die Messwerte sind gemäß funktionaler Anforderung F61 in Datenquellen zu ermitteln. Diese Datenquellen können beispielsweise Dateien (funktionale Anforde-

Abb. 5.6 Klassendiagramm: Ergebnis

rung F21), Datenbanken (funktionale Anforderung F22) oder manuelle Eingaben (funktionale Anforderung F23) sein und könnten in Zukunft um weitere Arten von Datenquellen ergänzt werden. Aus diesem Grund gibt es im Datenmodell eine abstrakte Klasse *DataSource* von der sich alle Arten an Datenquellen ableiten (siehe Abb. 5.7). *Measurements* sind Teil einer *DataSource*, somit ist ein Messwert genau einer Datenquelle zugeordnet, in der die Messungen durchgeführt werden. Die abstrakte Klasse *DataSource* besitzt die Attribute *name* und *description* des Datentyps *string*, um der Datenquelle einen Namen und eine Beschreibung zu geben. Weiter besitzt die abstrakte Klasse *DataSource* einen Aufzählungstyp *DataSourceType*. Der Aufzählungstyp *DataSourceType* hat die Werte *DATABASE*, *FILE* und *MANUAL*, welche die unterschiedlichen Arten von Datenquellen symbolisieren. Von der abstrakten Klasse *DataSource* leiten die Klassen *FileDataSource*, *DatabaseDataSource* und *ManualDataSource* ab.

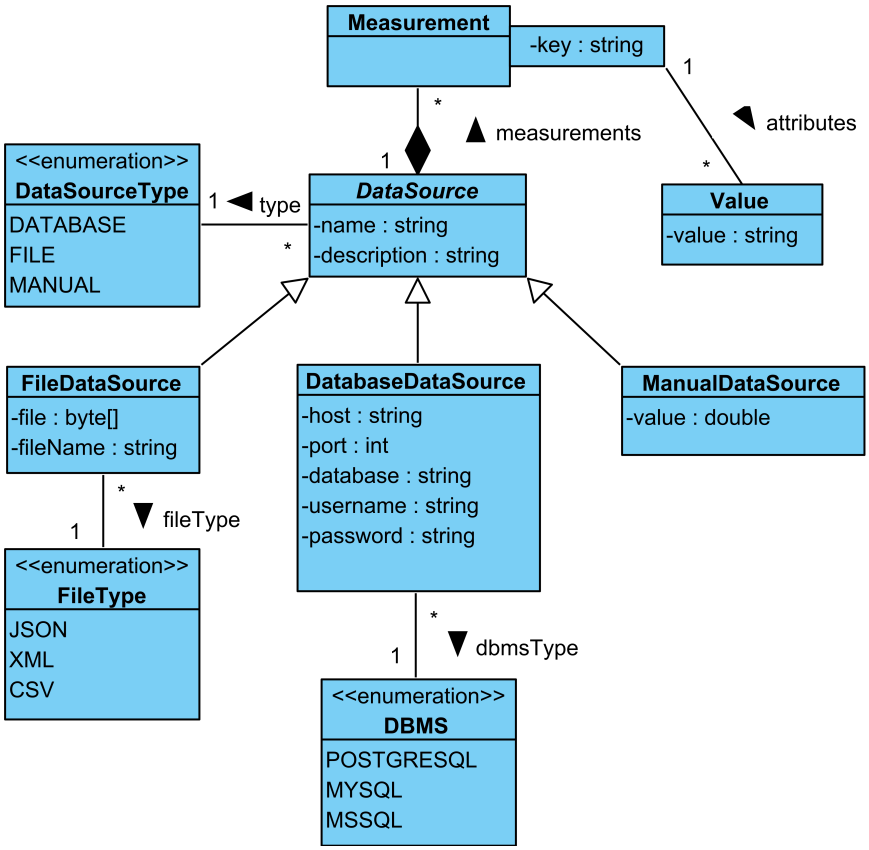

Abb. 5.7 Klassendiagramm: Datenquelle

FileDataSource ist die Klasse für Datei-Datenquellen und besitzt die Attribute *file*, dessen Datentyp eine Sequenz von *Byte*-Werten ist, und *fileName* vom Datentyp *string*.

Das Attribut *file* soll den Inhalt einer Datei speichern und das Attribut *fileName* den Dateinamen. Um zwischen unterschiedlichen Dateitypen zu unterscheiden, hat die Klasse *FileDataSource* einen Aufzählungstyp *FileType*. *FileType* ist ein Aufzählungstyp mit den Werten *JavaScript Object Notation (JSON)*, *eXtensible Markup Language (XML)* und *Comma-separated values (CSV)* und spiegeln die jeweiligen Dateitypen wider. In nachfolgenden Designiterationen könnten weitere Dateitypen ergänzt werden.

DatabaseDataSource ist die Klasse für Datenbank-Datenquellen. Die Klasse *DatabaseDataSource* besitzt Attribute, um die Verbindungs- und Zugangsdaten von Datenbanken zu speichern: Dazu zählen der Hostname oder die IP-Adresse (*host*) vom Datentyp *string*, der Port (*port*) vom Datentyp *int*, der Datenbankname (*database*) vom Datentyp *string*, sowie Benutzername (*username*) und Passwort (*password*) vom Datentyp *string*. Zusätzlich ist zwischen unterschiedlichen Datenbankmanagementsystem (DBMS) zu unterscheiden. Die Klasse *DatabaseDataSource* hat daher einen Aufzählungstyp *DBMS*. Der Aufzählungstyp *DBMS* hat die Werte *POSTGRESQL*, *MYSQL* und *MSSQL* für die unterschiedlichen Arten von *DBMS*.

ManualDataSource ist die Klasse für Datenquellen, deren Werte durch manuell Eingaben gesetzt werden. Die manuelle Eingabe soll im Attribut *value* vom Datentyp *double* in der Klasse *ManualDataSource* gespeichert werden.

Die Ermittlung des Messwerts in einer manuellen Datenquelle kann durch Abfrage des Attributs *value* erfolgen. Für die weiteren Datenquellen sind zusätzliche Angaben in der Konfiguration des Messwerts notwendig, um zu definieren, was gemessen werden soll bzw. wie sich der Messwert zusammengesetzt. Bei Datenbanken handelt es sich z. B. um Structured Query Language (SQL)-Anfragen, die einen Wert zurückgeben. Und bei CSV-Dateien sind es unter anderem Angaben, welche Spalte wie zu aggregieren ist. Diese Zusatzinformationen werden über Schlüssel-Wert-Paare in dem Attribut *attributes* in der Klasse *Measurement* gespeichert. Schlüssel (*key*) und Wert (*value*) sind dabei vom Datentyp *string*.

Informationsbedarf. Messwerte und Metriken sind Informationsbedarfen zugeordnet. Die Klasse *InformationNeed* repräsentiert im Datenmodell Informationsbedarfe und besitzt die Attribute *name* und *description* vom Datentyp *string* (siehe Abb. 5.8). Die Attribute ermöglichen die Erfassung des Namens und der Beschreibung des Informationsbedarfs. Die abstrakte Klasse *Measurable* kann mehrere *InformationNeeds* referenzieren, das heißt eine Metrik oder ein Messwert kann für mehrere Informationsbedarfe Informationen liefern. Genauso können für einen Informationsbedarf mehrere Metriken oder Messwerten Informationen liefern. Somit kann die Klasse *InformationNeed* auch mehrere *Measurables* referenzieren.

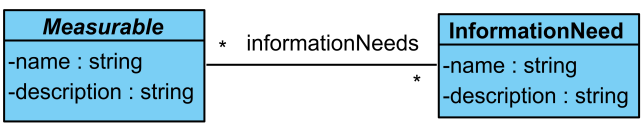

Abb. 5.8 Klassendiagramm: Informationsbedarf

Für diese Arbeit im IAM-Kontext, wäre es ausreichend für die vier relevanten IAM-Ziele von Hummer et al. [HGK+18] eine Aufzählung anstelle einer Klasse dem Datenmodell hinzuzufügen. Die Klasse ermöglicht es jedoch, das Datenmodell ebenso für Metriken und Messwerte außerhalb des IAM-Kontextes einzusetzen, wo andere Ziele existieren.

Zielgruppe und Stakeholder. Metriken und Messwerte besitzen Zielgruppen und Stakeholder. Zielgruppen können sich für mehrere Metriken und Messwerte interessieren und Stakeholder können für mehrere Metriken und Messwerte verantwortlich sein. Aus diesem Grund hat die abstrakte Klasse *Measurable* bidirektionale Assoziation zu den Klassen *Audience* und *Stakeholder* (siehe Abb. 5.9). Die Klassen *Audience* und *Stakeholder* sind Gruppen und erben von der abstrakten Klasse *Group*. Die abstrakte *Group* besitzt ein Attribute *name* vom Datentyp *string* für den Gruppennamen.

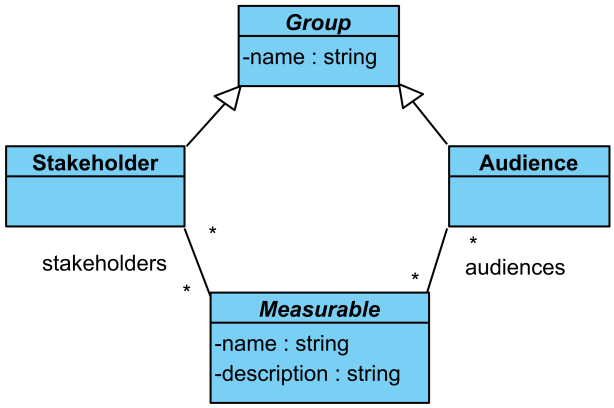

Abb. 5.9 Klassendiagramm: Zielgruppe und Stakeholder

Anwender. Den Zielgruppen und Stakeholdern sind mehrere Anwender zuzuordnen. Ebenso kann ein Anwender ein Teil mehrerer Zielgruppen (vgl. funktionale Anforderung F71) oder Stakeholder sein. Die Klasse *User* hat deswegen eine bidirektionale Assoziation zu den Klassen *Audience* und *Stakeholder* (siehe Abb. 5.10). Die Klasse User besitzt die Attribute *name* vom Datentyp *string* und *admin* vom Datentyp *boolean*. Der Name des Anwenders wird im Attribut *name* gespeichert. Der Wahrheitswert *admin* gibt an, ob der Anwender ein Administrator ist. Die Unterscheidung zwischen Anwendern mit und ohne Administrationszugriff ist notwendig, da es Funktionen gibt, die nur Administratoren durchführen dürfen (vgl. funktionale Anforderungen F20-F22, F24, F25, F30-32, F35, F40-F45, F48, F49).

Skalenniveau. Das Skalenniveau von Metriken und Messwerten gilt es zu erfassen. Aus diesem Grund besitzt die abstrakte Klasse *Measurable* einen Aufzählungstyp *Scale* (siehe Abb. 5.11). Der Aufzählungstyp *Scale* besitzt für die vier Skalenniveaus die Ausprägungen *NOMINAL* für nominalskalierte Werte, *ORDINAL* für ordinalskalierte Werte, *INTERVAL* für intervallskalierte Werte und *RATIO* für verhältnisskalierte Werte [WK18].

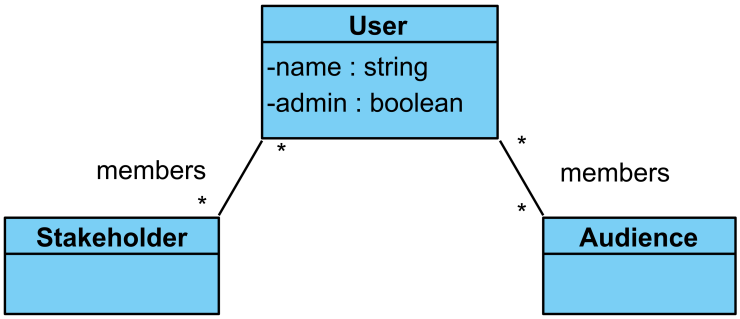

Abb. 5.10 Klassendiagramm: Anwender

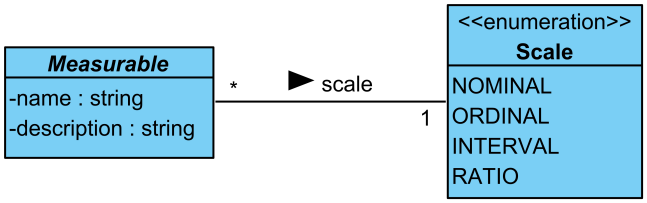

Abb. 5.11 Klassendiagramm: Skalenniveau

Einheit. Neben dem Skalenniveau ist auch die Einheit von Metriken und Messwerten zu erfassen. Die abstrakte Klasse *Measurable* besitzt einen Aufzählungstyp *Unit* (siehe Abb. 5.12). Der Aufzählungstyp *Unit* hat die Merkmale *TOTAL* und *PERCENT*, um zwischen Prozentwerten und absoluten Werten zu unterscheiden. In weiteren Iterationen kann es für eine granularere Unterscheidung sinnvoll sein, den Aufzählungstyp *Unit* um weitere Einheiten zu ergänzen.

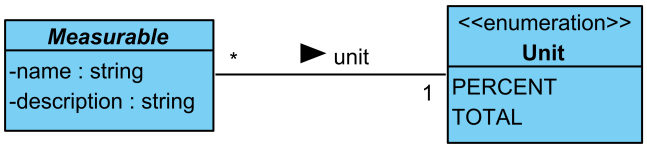

Abb. 5.12 Klassendiagramm: Einheit

Identifikator. Um die Instanzen aller Klassen eindeutig zu identifizieren, besitzen diese einen *Identifikator*. Im Datenmodell leiten sich daher alle Klassen direkt oder indirekt von der abstrakten Klasse *AbstractEntity* ab (siehe Abb. 5.13). Die abstrakte Klasse *AbstractEntity* hat ein Attribut *id* vom Datentyp *int*, um den Identifikator zu speichern.

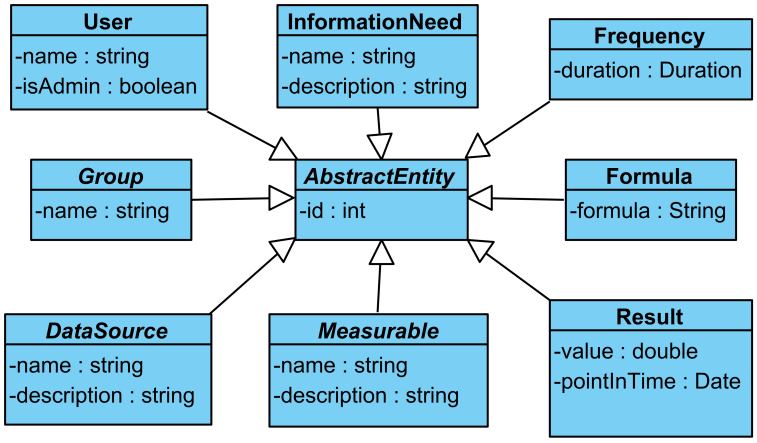

Abb. 5.13 Klassendiagramm: Identifikator

5.3 Prozesse

Im theoretischen Hintergrund zu Reporting und Metriken in Abschnitt 2.2 wurden die von Chew et al. [CSS+08] vorgeschlagene Prozesse zur Entwicklung und zur Implementierung von Metriken vorgestellt. Die Use Cases wurden unter anderem aus diesen Prozessen abgeleitet und skizziert. Dieses Kapitel beschreibt die im IAM Reporting Modul abzubildenden Abläufe und Prozesse.

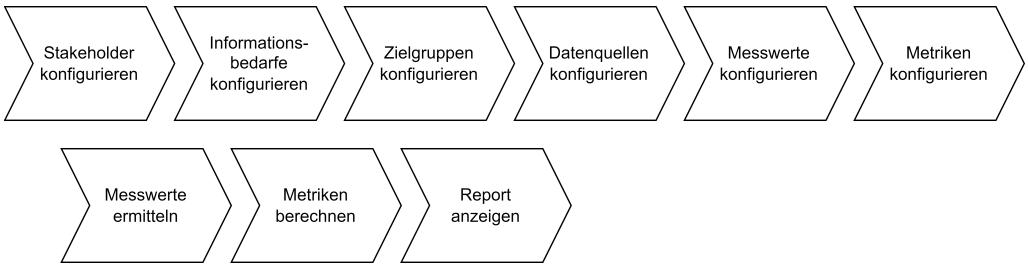

Abb. 5.14 Überblick über Prozesse

In der Abb. 5.14 ist ein Überblick aller Prozesse abgebildet. In der oberen Reihe sind die Prozesse zur Konfiguration der Stakeholder, Informationsbedarfe, Zielgruppen, Datenquellen, Messwerte und Metriken zu sehen. Darunter folgen die Prozesse zur Ermittelung der Messwerte, Berechnung der Metriken und Anzeige der Metriken in einem Report.

Grundsätzlich sollte das IAM Reporting Modul keine Reihenfolge vorgeben, in der die Prozesse nacheinander abzuarbeiten sind. Es kann jedoch aus zwei unterschiedlichen Gründen für Benutzer des IAM Reporting Moduls sinnvoll sein, dass diese die Prozessreihenfolge an bestimmten Stellen einhalten. Einerseits bauen die Konfigurationen von Metriken und Messwerten auf anderen Konfigurationen auf: Die Konfiguration von Metriken referenzieren die von Messwerten, Stakeholdern, Informationsbedarfe, Zielgruppen und ggf. anderer Metriken. Die Konfiguration von Messwerten referenzieren die von Datenquellen, Stakeholdern, Informationsbedarfen und Zielgruppen. Außerdem müssen zur Anzeige des Reports die notwendigen Metriken vorhanden und berechnet sowie

die dafür wiederum notwendigen Messwerte ermittelt sein. Andererseits haben Chew et al. [CSS+08] in der Beschreibung der fachlichen Prozesse auf die Einhaltung der sequentiellen Reihenfolge der Prozessschritte hingewiesen. Im fachlichen Prozesse werden, wie im Abschnitt 2.2 vorgestellt, zuerst die Stakeholder und die Informationsbedürfnisse festgelegt. In den darauffolgenden Schritten unterscheidet sich die Reihenfolge des fachlichen Prozesses von der des IAM Reporting Moduls: Es wird empfohlen, erst die Metriken festzulegen, bevor Messwerte und Datenquellen ausgewählt werden. Abschließend werden die Messwerte gesammelt und die Metriken analysiert.

Im Folgenden werden die einzelnen Prozessschritte mithilfe von UML-Sequenzdiagrammen beschrieben. Dabei wird angenommen, dass sich der Benutzer authentifiziert hat und autorisiert ist für die durchzuführenden Aktionen. Weiter wurden die Sequenzdiagramme unter der Annahme modelliert, dass die Benutzer valide Eingaben tätigen und das System fehlerfrei arbeitet. Am Beispiel von Datenquellen wird die Konfiguration der Elemente Stakeholder, Informationsbedarfe, Zielgruppen, Datenquellen, Messwerte, Metriken aufgezeigt. Die Prozesse sind für diese Elemente dieselben, lediglich die Namen der Elemente und Klassen in den Sequenzdiagrammen unterscheiden sich. Nach den Prozessen zur Konfiguration der Elemente werden die Prozesse zur Planung sich wiederholender Aufgaben zur Ermittlung von Messwerten und Berechnung von Metriken vorgestellt. Zum Schluss folgt der Prozess zur Ansicht eines Reports.

Überblick über Konfigurationen. Wie in der Abb. 5.15 dargestellt startet der Prozess zur Anzeige eines Überblicks aller Konfigurationen der Datenquellen durch Interaktion des Administrators mit der Benutzeroberfläche, indem dieser das Anzeigen aller konfigurierten Datenquellen anfordert. Die Benutzeroberfläche leitet die Anfrage an den entsprechenden Service weiter. Der Service sucht die konfigurierten Datenquellen in der Datenbank und gibt sie der Benutzeroberfläche zurück. Die Benutzeroberfläche zeigt dem Administrator daraufhin die konfigurierten Datenquellen in einer Übersicht an.

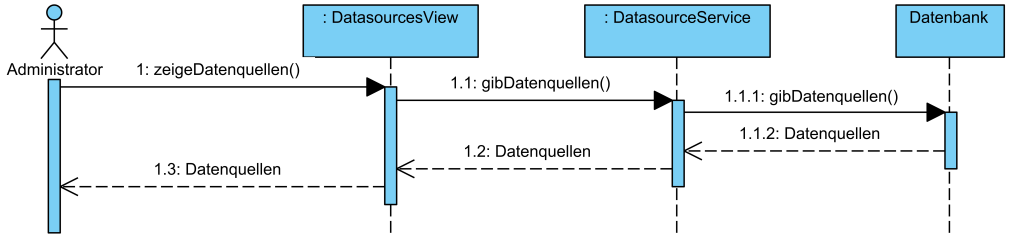

Abb. 5.15 Sequenzdiagramm: Überblick über Konfigurationen

Konfiguration erstellen. In der Abb. 5.16 ist das Sequenzdiagramm mit dem Prozess zum Erstellen einer neuen Konfiguration einer Datenquelle skizziert. Initiiert wird der Prozess durch den Administrator in der Übersicht aller konfigurierten Datenbanken durch Aufruf der Funktion zum Erstellen einer neuen Konfiguration. Die Benutzeroberfläche leitet den Administrator weiter zu einer Benutzeroberfläche mit einem Formular, in dem der Administrator die Konfiguration angeben kann. Das Formular überprüft die Eingaben

auf Validität. Speichert der Administrator das Formular, so wird die eingegebene Konfiguration nach erneuter Überprüfung an den Service zum Speichern weitergegeben. Dieser speichert die Konfiguration in der Datenbank ab und gibt die gespeicherte Konfiguration an die Benutzeroberfläche zurück. Die Benutzeroberfläche zeigt dem Administrator eine Erfolgsmeldung an und leitet diesen zur Übersicht aller konfigurierten Datenquellen zurück. Die Anzeige aller konfigurierter Datenquellen erfolgt wie im Prozess zuvor beschrieben.

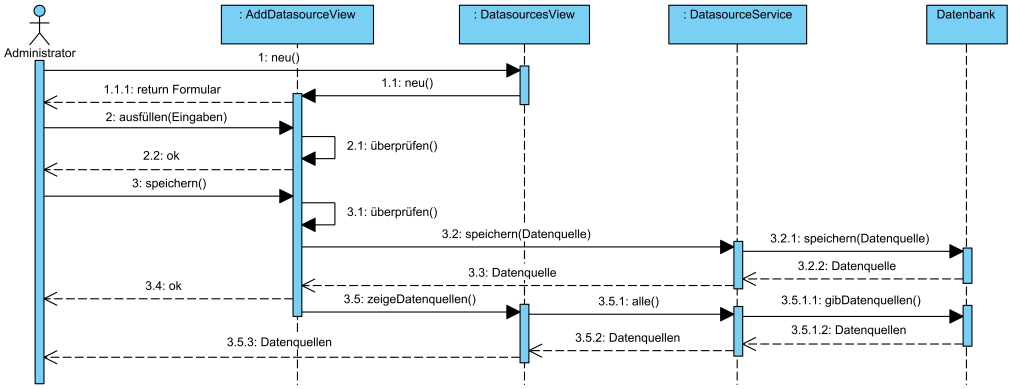

Abb. 5.16 Sequenzdiagramm: Konfiguration erstellen

Konfiguration editieren. Der Prozess zum Editieren der Konfiguration einer Datenquelle ist in Abb. 5.17 zu sehen. Der Administrator startet den Prozess durch Auswahl einer Konfiguration zum Editieren in der Übersicht aller konfigurierten Datenquellen. Die Benutzeroberfläche leitet den Administrator zur Benutzeroberfläche zum Editieren der Konfiguration einer Datenquelle weiter. Diese zeigt dem Administrator die Konfiguration der Datenquelle in einem Formular an. Die weiteren Prozessschritte zum Ausfüllen und Speichern des Formulars sind dieselben wie bei der Anlage einer Konfiguration.

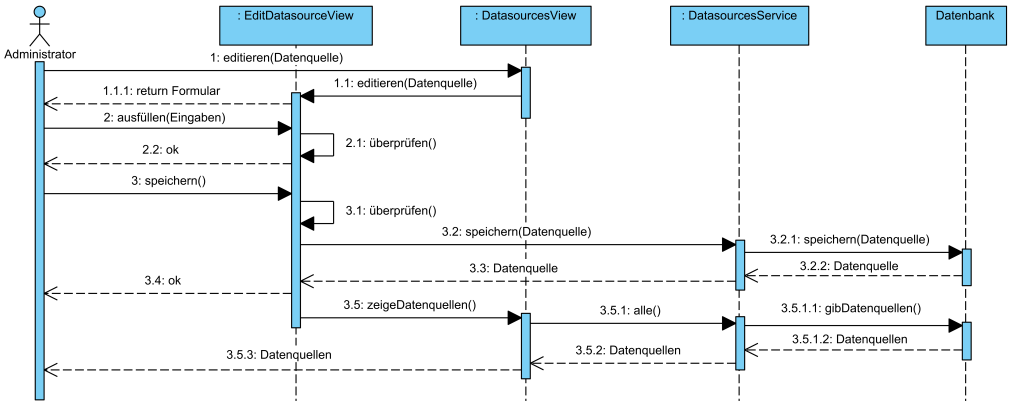

Abb. 5.17 Sequenzdiagramm: Konfiguration editieren

Konfiguration löschen. Um eine Konfiguration zu löschen hat der Administrator dieselben Prozessschritte wie beim Editieren der Konfiguration durchzuführen (siehe Abb. 5.18).

Anstelle das Formular auszufüllen und zu speichern, wählt der Administrator das Löschen der Konfiguration aus. Die Benutzeroberfläche gibt diese Anfrage weiter an den Service. Dieser überprüft, ob ein Löschen der konfigurierten Datenquelle möglich ist. Ein Löschen ist nur möglich, wenn keine anderen Konfigurationen (im Beispiel der Datenquelle: Konfiguration von Messwerten) darauf aufbauen. Die Konfiguration wird daraufhin aus der Datenbank gelöscht und eine Erfolgsmeldung an die Benutzerfläche bzw. den Administrator zurückgegeben. Zuletzt folgen die bereits bekannten Schritte zur Anzeige der Übersicht aller konfigurierten Datenquellen.

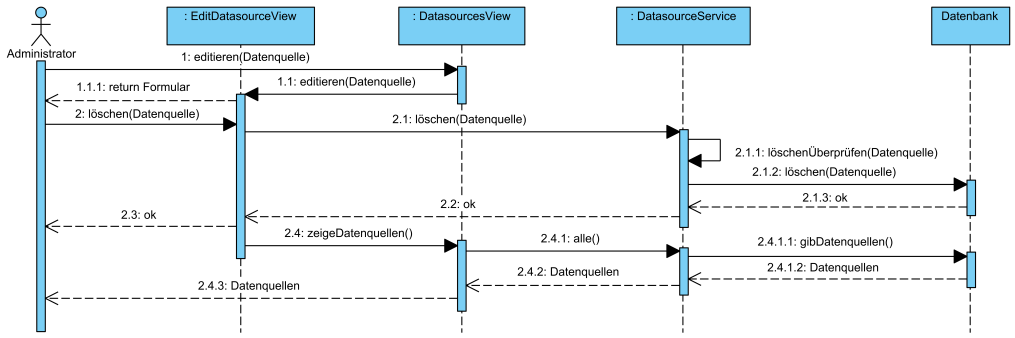

Abb. 5.18 Sequenzdiagramm: Konfiguration löschen

Aufgabe planen. Die Prozesse für sich wiederholende Aufgaben zur Ermittlung von Messwerten und Berechnung von Metriken sind identisch. Lediglich die Details der Aufgabe unterscheiden sich zwischen der Ermittlung der Messwerte und der Berechnung der Metriken. Der Prozess zur Aufgabenplanung wird durch den Service für Messwerte und Metriken angestoßen, sobald die Konfiguration eines Messwerts und einer Metrik in der Datenbank gespeichert wurde. Wie im Sequenzdiagramm in Abb. 5.19 dargestellt, übergibt der Service für Messwerte und Metriken dem Service zur Aufgabenplanung die Konfiguration. Dieser Service aktualisiert eine Aufgabe, falls für die Konfiguration bereits eine Aufgabe vorhanden ist. Ansonsten wird eine neue Aufgabe erstellt. Die Aufgabe wird dem Scheduler übergeben. Der Scheduler speichert die Aufgabe in der Datenbank und gibt die gespeicherte Aufgabe zurück.

Aufgabe ausführen. Der Prozess zur Ausführung von Aufgaben findet wiederholt sich jede Minute (siehe Abb. 5.20. Der Scheduler holt sich alle zum aktuellen Zeitpunkt auszuführenden Aufgaben aus der Datenbank und führt diese nacheinander aus. Das Ergebnis der Ausführung gibt der Scheduler dem Service zur Speicherung weiter. Der Service speichert das Ergebnis in der Datenbank und gibt es dem Scheduler zurück.

Aufgabe löschen. Das Löschen einer Aufgabe erfolgt – ähnlich zur Planung einer Aufgabe – nachdem der Service für Messwerte und Metriken die Konfiguration des Messwerts oder der Metrik erfolgreich aus der Datenbank gelöscht hat. Der Prozess startet mit der Übergabe der gelöschten Konfiguration durch den Service für Messwerte und Metriken

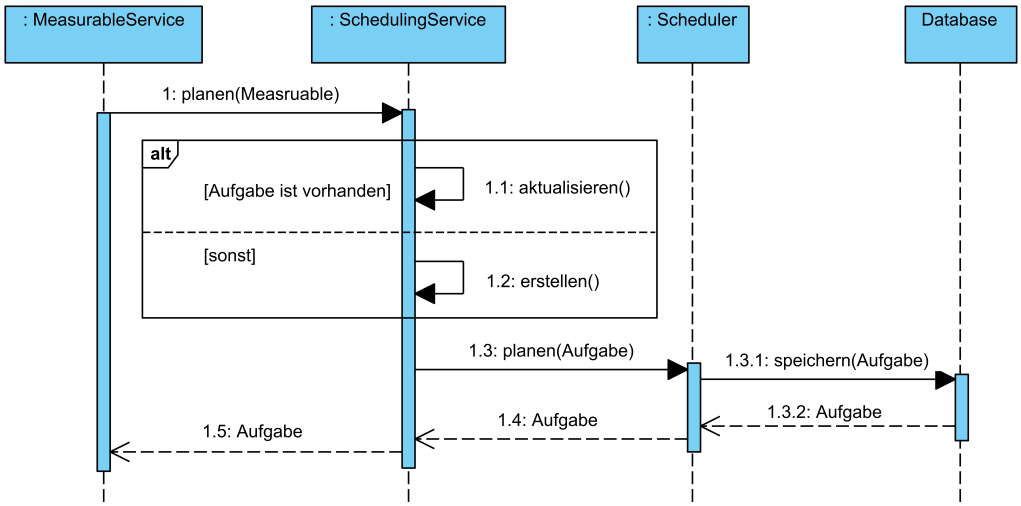

Abb. 5.19 Sequenzdiagramm: Aufgaben planen

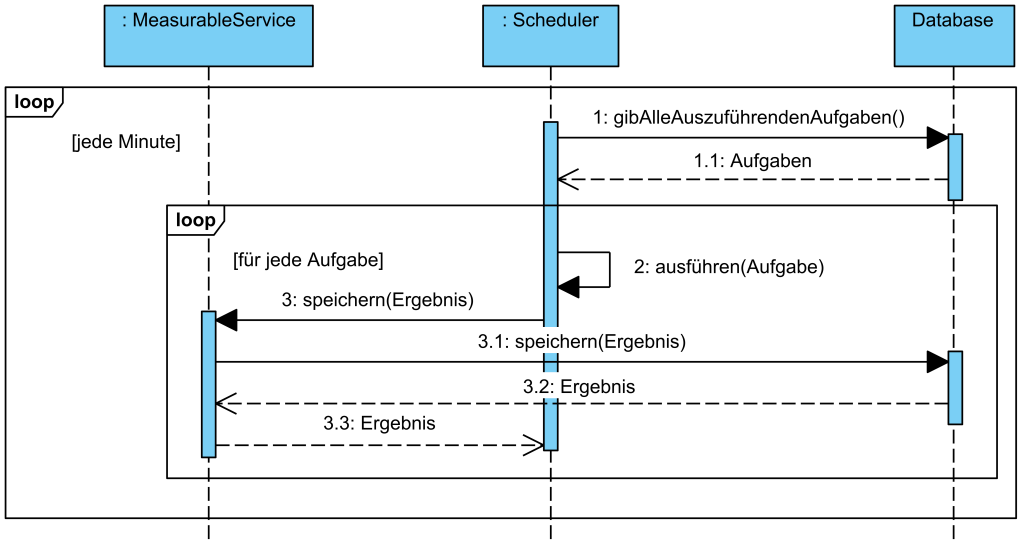

Abb. 5.20 Sequenzdiagramm: Aufgaben ausführen

an den Service zur Aufgabenplanung mit der Information die zugehörige Aufgabe zu löschen (siehe Abb. 5.21). Der Service zur Aufgabenplanung teilt dem Scheduler die Löschung der Aufgabe mit. Dieser löscht die Aufgabe aus der Datenbank und gibt eine Erfolgsmeldung zurück.

Report anzeigen. Der Prozess einen Report anzuzeigen erfolgt – wie im Sequenzdiagramm in Abb. 5.22 aufgezeigt – mittels Aufruf des Dashboards durch den Benutzer. Der Benutzer wählt einen Informationsbedarf aus. Die Benutzeroberfläche gibt den Informationsbedarf an den Service für Messwerte und Metriken weiter und fragt alle dazugehörigen Metriken an. Der Service sucht die entsprechenden Metriken in der Datenbank und gibt diese der Benutzeroberfläche zurück. Schließlich zeigt die Benutzeroberfläche den Report mit den Metriken an.

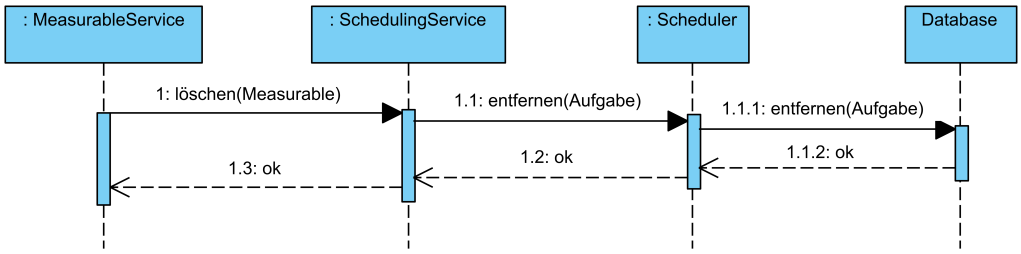

Abb. 5.21 Sequenzdiagramm: Aufgabe löschen

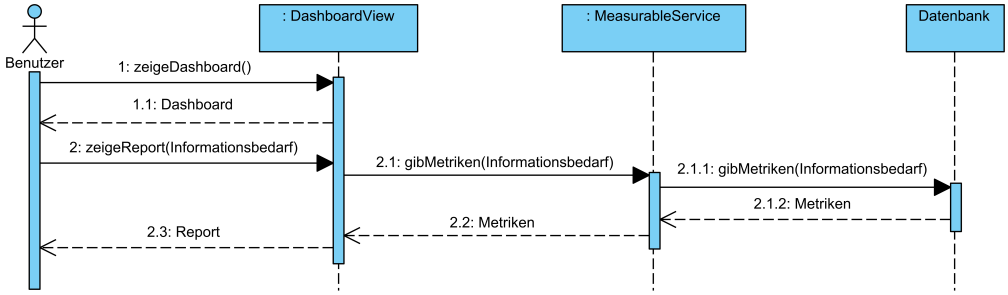

Abb. 5.22 Sequenzdiagramm: Report anzeigen

5.4 Benutzeroberfläche

Nachdem die Konzepte für das Datenmodell und die Prozesse des IAM Reporting Moduls beschrieben wurden, folgt nun die Skizzierung der Benutzeroberfläche. Die Anwender des IAM Reporting Moduls interagieren mit der Software über die Benutzeroberfläche, um die beschriebenen Schritte der Prozesse durchzuführen. Bei den skizzierten Benutzeroberflächen handelt es sich um die für das Design relevanten Teile der Benutzeroberfläche: Das Dashboard der Metriken sowie die Oberflächen zur Auflistung, Anlage, Bearbeitung und Löschung der Datenquellen, Messwerte und Metriken.

Benutzeroberflächen können in verschiedenen Detaillierungsgraden entworfen werden. Die Detaillierungsgrade reichen von Entwürfen einzelner Oberflächen bestimmter Funktionen über Designs gesamter Oberflächen bis hin zu Designprototypen und Pilotsystemen, mit denen bereits eine Interaktion möglich ist. [BBLZ96]

Bei den nachfolgend dargestellten Oberflächen handelt es sich um Mockups. Ein Mockup ist eine Skizze einer möglichen Benutzeroberfläche der Anwendung [RRG+10]. Mockups helfen dabei die Anforderungen an die Anwendung in einer gemeinsamen, verständlichen Sprache für Kunden und Entwickler darzustellen [RRG+11]. Mockups sind im mittleren Bereich der Detaillierungsgrade anzusiedeln und skizzieren neben der Oberfläche für eine bestimmte Funktion ebenso die Einbettung in die gesamte Oberfläche inklusive beispielsweise der Navigationselemente.

Dashboard. Das Mockup des Dashboards ist in Abb. 5.23 dargestellt. Auf der linken Seite ist die Navigationsleiste zu sehen, mithilfe derer die Navigation zu den Seiten Dashboard, Metriken, Messwerte und Datenquellen sowie eine Abmeldung des aktuellen Anwenders möglich sind. Die Navigation zu den Seiten Metriken, Messwerte und

Datenquellen sind dabei den Administratoren vorbehalten. Im oberen Teil des Mockups befindet sich der Titel der aktuellen Seite. Die Navigationsleiste und der Titel der Seite ist auf allen beschriebenen Seiten aufzufinden.

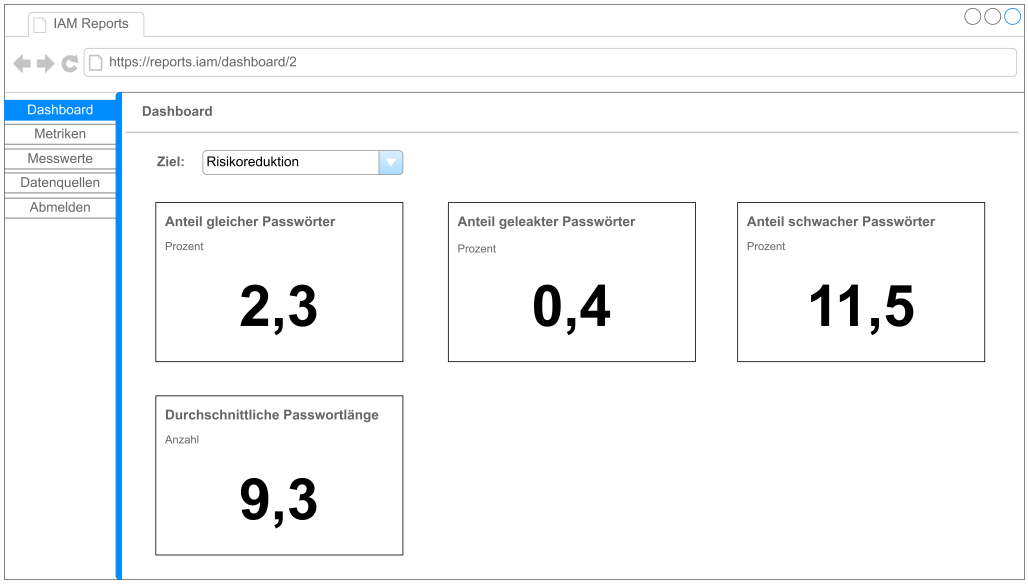

Abb. 5.23 Mockup: Dashboard

In der Mitte ist das Dashboard zu sehen. Es besteht aus der Auswahl des Informationsbedarfs im obereren Teil und der Darstellung der Metriken zum gewählten Informationsbedarf auf Karten in einem Raster (vgl. funktionale Anforderung F10). Die Karte einer Metrik ist wie folgt aufgebaut: Links oben ist der Name der Metrik in fetter Schrift abgedruckt. Direkt darunter folgt die Einheit der Metrik. Mittig ist der aktuellste Metrikwert in großer, fetter Schrift dargestellt. Um die funktionale Anforderung F12 zu erfüllen, könnte in weiteren Designiterationen eine Detailansicht der Metriken hinzugefügt werden, die sich bei Auswahl einer Metrik bzw. Karte öffnet und den Verlauf der Metrikwerte darstellt.

Auflisten der Datenquellen. Wechselt der Anwender in der Navigation auf die Ansicht der Datenquellen, so werden, wie in Abb. 5.24 dargestellt, alle konfigurierten Datenquellen in einer Tabelle aufgelistet. Die Tabelle hat drei Spalten: Die erste Spalte ist für den Namen der Datenquelle, die Zweite für den Typ der Datenquelle und die Dritte zeigt die Anzahl der Messwertkonfigurationen, welche die Datenquelle verwenden, an. Um die Konfiguration einer Datenquelle zu editieren, ist die Zeile der Datenquelle in der Tabelle anzuklicken. Ist eine neue Konfiguration einer Datenquelle anzulegen, ist der Button „neue Datenquelle“ rechts neben dem Titel der Seite zu betätigen.

Anlegen einer Datenquelle. Wählt der Anwender das Anlegen einer neuen Konfiguration einer Datenquelle aus, öffnet sich das in Abb. 5.25 dargestellt Formular. Im Formular kann die Konfiguration einer neuen Datenquelle gemäß der funktionalen Anforderung F20 vorgenommen werden.

Im Formular sind für alle Typen von Datenquellen der Name, die Beschreibung und

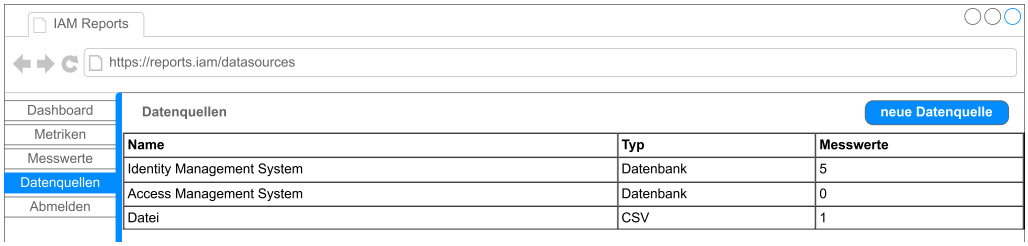

Abb. 5.24 Mockup: Auflisten der Datenquellen

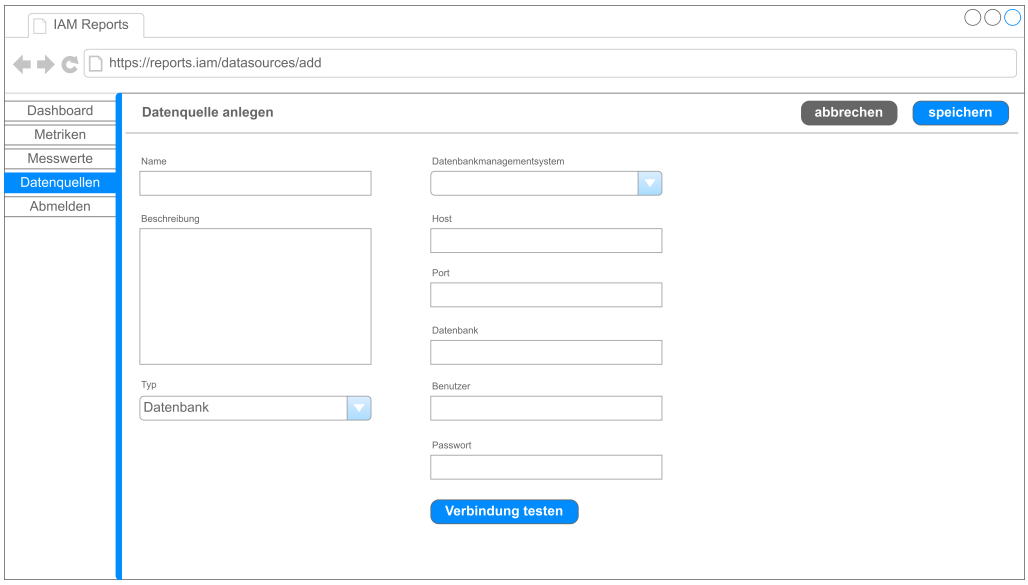

Abb. 5.25 Mockup: Anlegen einer Datenbank als Datenquelle

der Typ der Datenquelle anzugeben. Je nach Typ der Datenquelle können zusätzliche Informationen erforderlich sein (vgl. funktionale Anforderungen F21-F23). In dem dargestellten Beispiel handelt es sich eine Datenquelle vom Typ Datenbank. Aus diesem Grund werden die Verbindungsdaten der Datenbank benötigt: das DBMS, der Host der Datenbank, der Port unter welchem die Datenbank erreichbar ist, der Name der Datenbank sowie die Anmeldedaten Benutzer und Passwort. Um die Konfiguration zu überprüfen, befindet sich im Falle des Typs Datenbank am Ende des Formulars ein Button zum Testen der Verbindung zur Datenbank. Für die weiteren Typen von Datenquellen – Datei und manuell – befinden sich die Mockups im Anhang B.

Rechts neben dem Titel befinden sich die Buttons „abbrechen“ und „speichern“, um die Konfiguration der Datenquelle zu verwerfen oder persistent zu speichern (vgl. funktionale Anforderung F24). Nach Durchführung der jeweiligen Aktion wird der Anwender zurück zur Übersicht aller konfigurierten Datenquellen geleitet.

Editieren einer Datenquelle. Die Benutzeroberfläche zum Editieren der Konfiguration einer Datenquelle (vgl. funktionale Anforderung F25) ist sehr ähnlich zu der Benutzeroberfläche zur Anlage einer Datenquelle gestaltet. Wie in Abb. 5.26 dargestellt, ist das Formular zur Eingabe der Konfiguration identisch. Es befindet sich lediglich neben den

Buttons „abbrechen“ und „speichern“ ein zusätzlicher Button „löschen“ zum Löschen der Datenquelle.

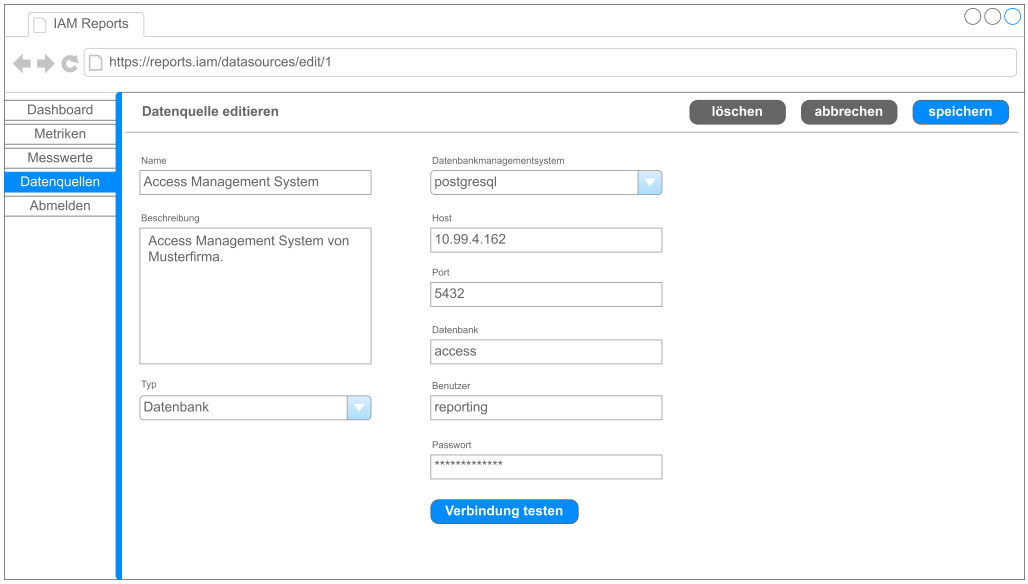

Abb. 5.26 Mockup: Editieren einer Datenquelle

Löschen einer Datenquelle. Bei Betätigung des Buttons „löschen“ öffnet sich das in Abb. 5.27 skizzierte Pop-up Fenster. In dem Pop-up Fenster wird der Anwender aufgefordert, die Aktion zu bestätigen. Der Anwender kann das Entfernen der konfigurierten Datenquelle durch Auswahl von „löschen“ bestätigen oder die Aktion durch Auswahl von „abbrechen“ abbrechen. Wird die Aktion bestätigt, ist die Konfiguration der Datenquelle zu löschen (vgl. funktionale Anforderung F26) und der Anwender zurück zur Übersicht aller konfigurierten Datenquellen zu leiten. Wird die Aktion abgebrochen, ist das Pop-up Fenster zu schließen.

Auflisten der Messwerte. Ähnlich zur Übersicht der konfigurierten Datenquellen, besteht für den Anwender die Möglichkeit, alle konfigurierten Messwerte aufzulisten (siehe Abb. 5.28). Die Auflistung der Konfiguration der Messwerte erfolgt ebenso tabellarisch. Die Tabelle hat die Spalten Name, Datenquelle und Metriken. In der ersten Spalte ist der Name des Messwerts angegeben. Die Spalte Datenquelle stellt den Namen der Datenquelle, in der die Messwerte ermittelt werden, dar. Die letzte Spalte zeigt die Anzahl an Metrikkonfigurationen an, die in ihrer Formel den Messwert referenzieren. Genauso wie bei der Auflistung der konfigurierten Datenquellen können Konfigurationen von Messwerten durch Klick auf die Zeile editiert oder durch Klick auf den Button „neuer Messwert“ neu angelegt werden.

Editieren eines Messwerts. Wie schon bei der Konfiguration der Datenquellen sind die Benutzeroberflächen zur Anlage und und zum Editieren der Konfiguration von Messwerten sehr ähnlich. Aus diesem Grund wird im Folgenden die Benutzeroberfläche zum

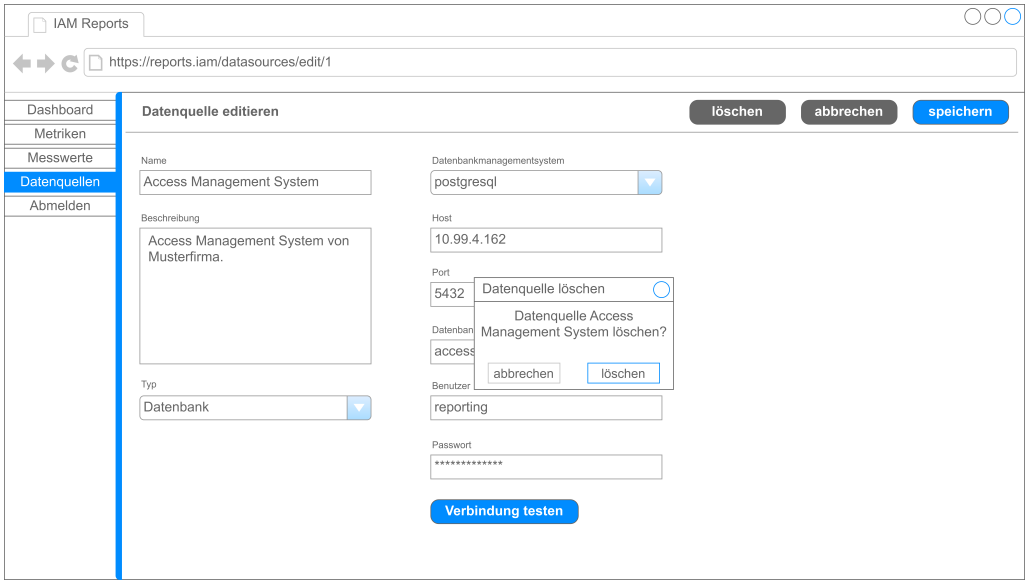

Abb. 5.27 Mockup: Löschen einer Datenquelle

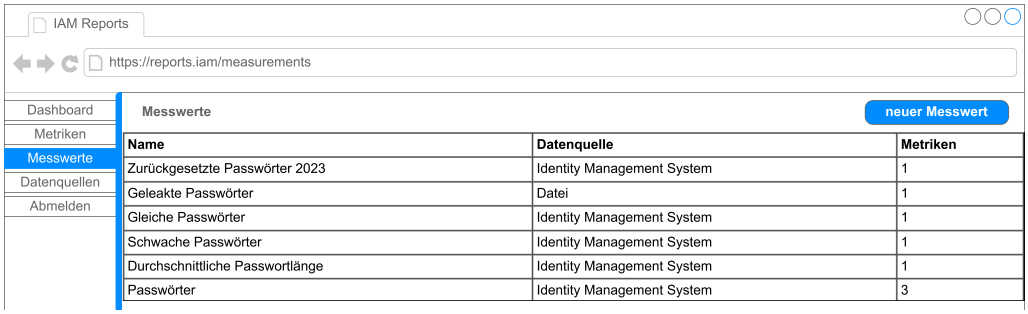

Abb. 5.28 Mockup: Auflisten der Messwerte

Editieren der Konfiguration der Messwerte beschrieben (vgl. funktionale Anforderung F31). Das Mockup zur Anlage der Konfiguration von Messwerten ist im Anhang B zu finden (vgl. funktionale Anforderung F30).

Das Formular zur Anlage und zum Editieren beinhaltet alle notwendigen Felder zur Konfiguration eines Messwertes (siehe Abb. 5.29): Den Namen, die Beschreibung, den Informationsbedarf, die Zielgruppen, die Stakeholder, die Frequenz (vgl. funktionale Anforderung F35), die Einheit, die Skala und die Datenquelle des Messwertes.

Je nach Typ der gewählten Datenquelle können zusätzliche Felder erforderlich sein. Im gewählten Beispiel handelt es sich beim Typ der Datenquelle um eine Datenbank. Aus diesem Grund gibt es ein zusätzliches Feld zur Angabe einer SQL-Anfrage und einen Button zum Testen der SQL-Anfrage. Die Mockups von Messwertkonfigurationen mit den Datenquellentypen Datei und manuell sind im Anhang B beigefügt.

Neben dem Titel befinden sich – wie beim Editieren der Konfiguration einer Datenquelle – Buttons zum Löschen, Abbrechen und Speichern. Dabei ist der Button „löschen“ ausschließlich der Seite zum Editieren der Konfiguration von Messwerten vorbehalten. Das Pop-up zum Löschen der Konfiguration von Messwerten ist analog zu dem Pop-up zum Löschen der Konfiguration von Datenquellen aufgebaut und wird daher nicht

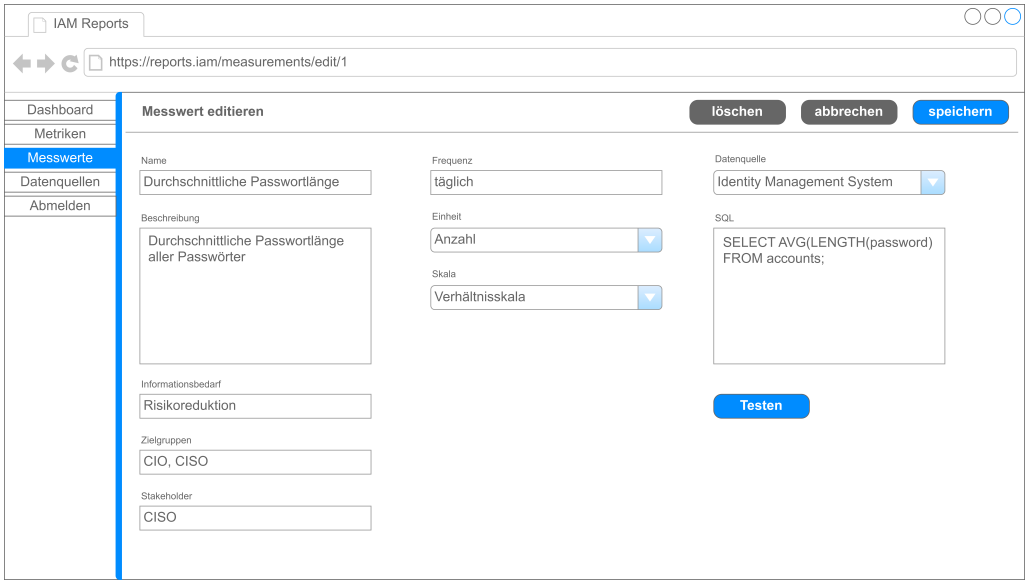

Abb. 5.29 Mockup: Editieren eines Messwerts mit einer Datenbank als Datenquelle

wiederholt beschrieben. Das Mockup der Benutzeroberfläche zum Löschen der Konfiguration von Messwerten gemäß der funktionalen Anforderung F32 ist im Anhang B zu finden.

Auflisten der Metriken. Auch die Konfiguration aller Metriken ist tabellarisch dargestellt (vgl. Abb. 5.30). In der Tabelle sind von den konfigurierten Metriken der Name, der Informationsbedarf und die Formel angezeigt. Die Seiten zur Anlage und zum Editieren der Konfiguration einer Metrik erreicht der Anwender abermals durch Klicken des Buttons „neue Metrik“ bzw. durch Auswahl einer Zeile in der Tabelle.

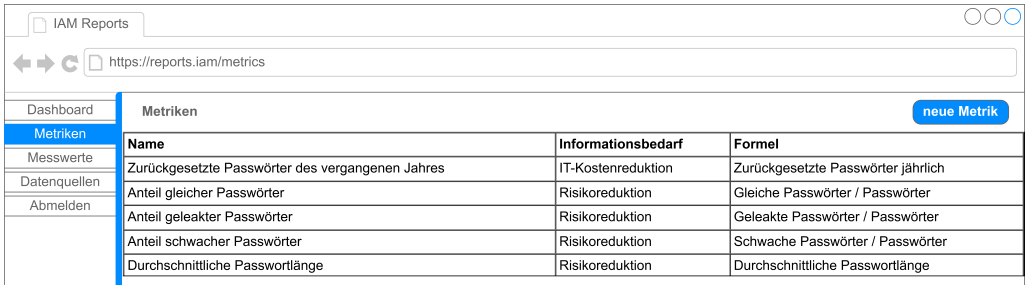

Abb. 5.30 Mockup: Auflisten der Metriken

Editieren einer Metrik. Wie bei der Konfiguration der Datenquellen und der Messwerte ähneln sich die Benutzeroberflächen zur Anlage und zum Editieren der Konfiguration der Metriken (vgl. funktionale Anforderungen F40 und F44). Deswegen wird nur die Benutzeroberfläche zum Editieren der Konfiguration einer Metrik – wie in Abb. 5.31 dargestellt – vorgestellt. Das Mockup der Benutzeroberfläche zur Anlage der Konfiguration einer Metrik ist im Anhang B zu finden.

Im Formular zum Editieren der Metriken sind die Felder Name, Beschreibung, Informationsbedarf, Zielgruppen, Stakeholder, Frequenz (vgl. funktionale Anforderung F43),

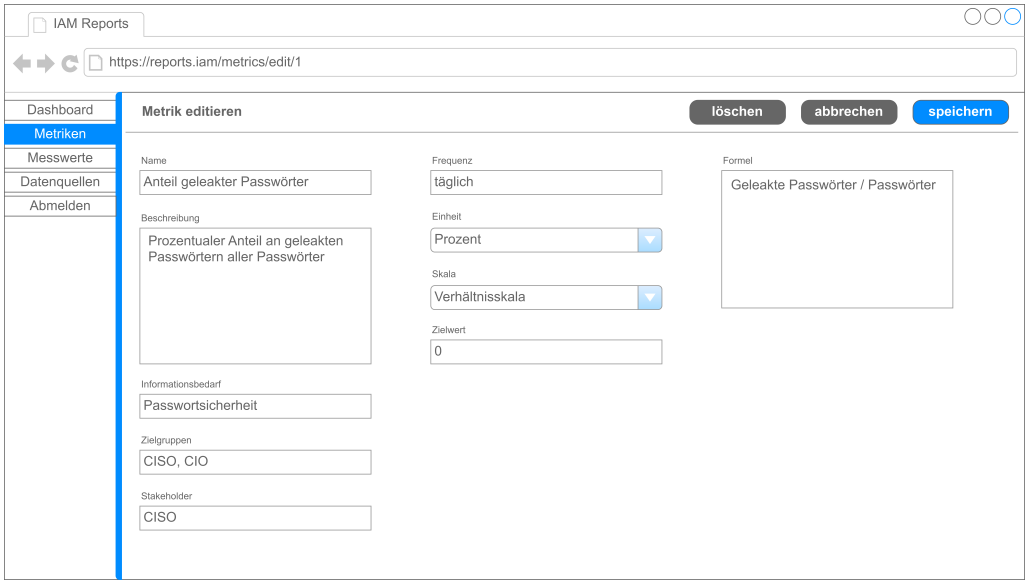

Abb. 5.31 Mockup: Editieren einer Metrik

Einheit, Skala, Zielwert und Formel (vgl. funktionale Anfroderung F41) auszufüllen. Die Buttons zum Löschen, Abbrechen und Speichern sind wie in den vorherigen Benutzeroberflächen rechts neben dem Titel zu finden. Der Button „löschen“ ist ausschließlich beim Editieren der Konfiguration einer Metrik sichtbar. Beim Löschen der Konfiguration einer Metrik (vgl. funktionale Anforderung F45) erscheint das bereits für Datenquellen und Messwerte vorgestellte Pop-up. Das Mockup zum Löschen der Konfiguration einer Metrik ist ebenfalls im Anhang B angehängt.

Kapitel 6

Implementierung des Prototyps

Nachdem das Design des IAM Reporting Moduls entworfen wurde, folgte zur Verifizierung des Designs eine prototypische Implementierung. Die prototypische Implementierung trägt den bislang noch unbeantworteten Teil zur Forschungsfrage RQ2 bei. In diesem Kapitel wird die Implementierung des Prototyps des IAM Reporting Moduls beschrieben.

Ein Prototyp ist eine frühzeitige Version eines Systems. Er kann zur Demonstration von Konzepten, Testen von Designmöglichkeiten und zur Exploration des Problems und der möglichen Lösungen eingesetzt werden. Die iterative und schnelle Entwicklung des Prototyps ist essentiell, um die Kosten gering zu halten und Stakeholdern frühzeitig ein System zum Experimentieren zur Verfügung zu stellen. In der Softwareentwicklung kann der Prototyp sowohl in der Anforderungsphase bei der Ermittlung und der Validierung von Anforderungen unterstützen, als auch in der Designphase, um Softwarelösungen und die Entwicklung der Benutzeroberflächen zu untersuchen. [Som16, 62 f.]

Ziel dieses Prototyps ist es, die grundlegenden Anforderungen – gekennzeichnet durch das Schlüsselwort „MUSS“ – zu implementieren. Weitere Anforderungen mit dem Schlüsselwort „SOLLTE“ sind umzusetzen, falls dies der Zeitrahmen erlaubt. Ist eine Umsetzung nicht möglich, so sind diese Anforderungen trotzdem bei der Implementierung zu berücksichtigen, um eine spätere Erweiterung zu ermöglichen. Letzteres gilt ebenso für Anforderungen mit dem Schlüsselwort „WIRD “.

Im Folgenden wird der verwendete Softwarestack vorgestellt und auf die Implementierungsdetails wesentlicher Funktionen eingegangen. Zuletzt wird die Installation der Entwicklungsumgebung beschrieben.

6.1 Softwarestack

Der Prototyp wird als Webanwendung implementiert. Zum Einsatz kommen dabei die Programmiersprache Java, das Webframework Vaadin Flow, das Anwendungsframework Spring Boot, das Framework Hibernate zur objektrelationalen Abbilung, die DBMSs PostgreSQL und H2, das Build-Werkzeug Maven sowie weitere Java-Bibliotheken.

Java. Zur Programmierung des Prototyps wird die Programmiersprache Java verwendet. Java ist eine objektorientierte Sprache, die in Bytecode übersetzt wird. Ausgeführt wird der Bytecode in einer Laufzeitumgebung, der so genannten Java Virtual Machine (JVM). Die JVM wandelt den Bytecode in Maschinencode um und bringt Zusatzdienste wie z. B. automatische Speicherbereinigung oder Typprüfung mit. Durch die JVM ist Java plattformunabhängig. [Ull21, 49-65] Für diese Arbeit wird die Open Source-Implementierung Eclipse Temurin Open Java Development Kit (OpenJDK) der Java Platform, Standard Edition eingesetzt. Diese ist unter der GPLv2+CPE (GNU Gerneral Public License, version 2, with the Classpath Exception) lizenziert [Fre91].

Vaadin Flow. Um das Frontend des Protoyps zu programmieren wird Vaadin Flow eingesetzt. Vaadin Flow ist ein Framework, das es erlaubt, Webanwendungen mit Java zu programmieren ohne die für Webanwendungen typischen Sprachen wie Hypertext Markup Language (HTML), Cascading Style Sheets (CSS) oder JavaScript einsetzen zu müssen. Die Benutzeroberfläche wird dabei aus Komponenten zusammengebaut. Die Komponenten können mit Datenquellen verbunden und Aktionen bei Interaktion durch den Benutzer definiert werden. Es ist möglich, vordefinierte Komponenten einzusetzen, als auch eigene Komponenten zu definieren. Eine Vaadin Flow-Applikation wird in der JVM ausgeführt und als HTML gerendert. [Vaa23a] Vaadin Flow und die standardmäßig mitgelieferten Komponenten sind Open Source und unter der Apache License 2.0 lizenziert [Vaa16]. Vaadin bietet zusätzliche Komponenten unter einer proprietären Lizenz an [Vaa22]. Auf den Einsatz solcher proprietär lizenzierten Komponenten wurde in dieser Arbeit verzichtet.

Spring Boot. Das Backend des Prototyps ist mit Spring Boot programmiert und stellt dem Vaadin Flow-Frontend die Daten aus der Datenbank zur Verfügung. Spring ist ein Framework für die Anwendungungsentwicklung mit Java und reduziert die Komplexität der Java-Programierung. Spring Boot ist ein auf Spring aufbauendes Projekt, das Leitlinien gibt, wie Anwendungen mit Spring und weiteren Bibliotheken erstellt und konfiguriert werden. Durch die vorgegebenen Leitlinien ist eine weitestgehend automatische Konfiguration der Spring Anwendung möglich. Spring Boot Applikationen sind eigenständige Spring-Applikationen mit eingebettetem Java-Applikationsserver. [VMw23] Sowohl Spring Boot als auch das Spring Framework sind Open Source und unter der Apache License 2.0 lizenziert [VMw19a; VMw19b].

Node.js. Um die Benutzeroberflächen von Vaadin Flow zu bauen, wird Node.js benötigt. „Node.js ist eine JavaScript-Laufzeitumgebung, die auf Chromes V8 JavaScript-Engine basiert“ [Ope23]. Node.js ist Open Source und mit der MIT Lizenz lizenziert [Ope22].

Apache Tomcat. Spring Boot verwendet im Prototyp den Java-Applikationsserver Apache Tomcat. Apache Tomcat ist ein Applikaationsserver für Java-Anwendungen und

implementiert einige der Jakarta Enterprise Edition (EE, früher Java EE) Spezifikationen. Apache Tomcat ist Open Source und mit der Apache License 2.0 lizenziert.

Hibernate. Um das Domänenmodell des Prototyps in einer relationalen Datenbank zu implementieren, wird Hibernate eingesetzt. Hibernate ist ein Framework für Java. Hibernate Object-Relational Mapping (ORM) übernimmt die objektrelationale Abbildung von Domänenmodellen in relationale Datenbanken und implementiert die Java Persistence API (JPA) [HiboJa]. Hibernate Validator ist ein weiteres Projekt von Hibernate, das Objekte validiert und Bean Validation 2.0 implementiert [HiboJb]. Sowohl Hibernate ORM als auch Hibernate Validator sind Open Source: Hibernate ORM ist mit der LGPL 2.1 (GNU Lesser General Public License Version 2.1) [Hib18] und Hibernate Validator mit der Apache License 2.0 [Hib09] lizenziert.

H2. Während der Entwicklung des Prototyps werden die Daten in einer H2-Datenbank gespeichert. H2 ist ein relationales DBMS, das in Java geschrieben ist und die Datenbanken im Arbeitsspeicher hält [H2oJa]. Es ist lizenziert unter den Open Source Lizenzen MPL 2.0 (Mozilla Public License Version 2.0) oder EPL 1.0 (Eclipse Public License) [H2oJb]. Während der Entwicklung von Java Anwendungen ist der wesentliche Vorteil von H2 gegenüber klassischen relationalen Datenbanksystemen, dass H2 nicht separat aufgesetzt und konfiguriert werden muss. Für den Einsatz in produktiven Anwendungen ist H2 weniger geeignet.

PostgreSQL. PostgreSQL ist ein objektrelationales DBMS, welches die SQL-Sprache verwendet und erweitert sowie mit weiteren Funktionen kombiniert, um Daten sicher zu speichern. Im Gegensatz zu H2 eignet sich PostgreSQL für den produktiven Einsatz. PostgreSQL läuft auf allen großen Betriebssystemen und ist vollständig konform zu den Eigenschaften Atomarität, Konsitenzerhaltung, Isolation und Dauerhaftigkeit. [The23b] PostgreSQL ist Open Source und mit der PostgreSQL License lizenziert [The23c].

Apache Maven. Zum Bauen des Prototypen und zur Verwaltung der eingesetzten Bibliotheken wird die Software Apache Maven eingesetzt. Apache Maven ist eine Software, um auf Java-basierende Projekte zu bauen und zu managen: Apache Maven vereinfacht den Bauprozess, vereinheitlicht das Bausystem, gibt Überblick über Projektinformationen und gibt Leitlinien für die Entwicklung von Software. Apache Maven ist Open Source und mit der Apache License 2.0 lizenziert. [The23a]

Codebasis. Ausgangspunkt für die Codebasis ist das Standard-Template für Vaadin Flow-Projekte in Verbindung mit Spring Boot, verfügbar unter https://github.com/vaadin/skeleton-starter-flow-spring/tree/v23. Dabei handelt es sich um ein Maven Java Projekt mit den notwendigen Abhängigkeiten für Vaadin Flow und Spring Boot:

- *Vaadin*
- *Vaadin Spring Boot Starter*
- *Spring Boot Starter Validation*
- *Spring Boot Devtools*
- *Spring Boot Starter Test*
- *Vaadin Testbench*
- *JUnit Vintage Engine*
- *Webdrivermanager*

Die Abhängigkeit von *Vaadin* wurde durch *Vaadin-Core* ersetzt, um ausschließlich Open Source Komponenten einzusetzen.

Zusätzliche Bibliotheken. Für den Prototypen wurde die Maven-Konfiguration um zusätzliche Bibliotheken ergänzt:

- *H2 Databse*: Das In-Memory DBMS H2 dient als Datenbank während der Entwicklung.
- *PostgreSQL*: Die PostgreSQL-Bibliothek findet Verwendung zur Anbindung von PostgreSQL-Datenbanken als Datenquelle als auch der Datenbank im Live Betrieb.
- *Spring Boot Starter Data JPA*: Diese Bibliothek dient zur Persistierung von Applikationsdaten und inkludiert Hibernate.
- *Spring Boot Starter Security*: Spring Security fügt Loginmechanismen zum Projekt hinzu.
- *Sprint Boot Starter Quratz*: Quartz ist eine Bibliothek zur Planung von wiederkehrenden Aufgaben.
- *EvalEx*: EvalEx ist eine Bibliothek zum Analysieren und Auswerten von algebraischen Ausdrücken.
- *Lumogridlayout*: Lumogridlayout ist eine Bibliothek mit Vaadin-Komponenten, um CSS Grid-Layouts in Vaadin zu verwenden.

Versionsübersicht. Die Tab. 6.1 gibt einen Überblick über die Versionen der eingesetzten Software und Bibliotheken. Die Tabelle ist unterteilt in installierte Software, Maven-Bibliotheken und zusätzliche Bibliotheken, die Teil bereits aufgelisteter Maven-Bibliotheken sind, jedoch aufgrund vorheriger Erwähnung nochmals separat aufgelistet werden.

Software / Bibliothek	**Version**	**Anmerkung**
Software		
Apache Maven	3.9.1	
Eclipse Temurin OpenJDK	17.0.6+10	
Node.js	18.15.0	
PostgreSQL	15.2	
Maven-Bibliotheken		
EvalEx	3.0.3	
H2 Database	2.1.214	
JUnit Vintage Engine	5.8.2	
Lumogridlayout	1.2.2	
PostgreSQL	42.6.0	
Spring Boot Devtools	2.7.9	
Sprint Boot Starter	2.7.9	
Spring Boot Starter Data JPA	2.7.9	
Spring Boot Starter Quartz	2.7.9	
Spring Boot Starter Security	2.7.9	
Sprint Boot Starter Test	2.7.9	
Sprint Boot Starter Validation	2.7.9	
Vaadin-Core	23.3.8	
Vaadin Spring Boot Starter	23.3.8	
Vaadin Testbench	23.3.8	
Webdirvermanager	5.3.2	
Zusätzliche Bibliotheken		*inkludiert in*
Apache Tomcat	9.0.71	Vaadin Spring Boot Starter
Hibernate	5.6.15.Final	Sprint Boot Starter Data JPA

Tab. 6.1 Versionsübersicht

6.2 Projektstruktur

In Abb. 6.1 ist die Struktur des Maven Projekts abgebildet. Im Hauptordner / befindet sich die Maven Konfigurationsdatei `pom.xml`. Weitere Konfigurationsdateien für Spring, Quartz und Hibernate, befinden sich unter `resources`. Dabei kommen unterschiedliche Konfigurationsdateien für die lokale Entwicklung, das Testen und das Bauen der produktiven Anwendung sowie eine für alle Szenarien allgemeingültige Konfigurationen zum Einsatz.

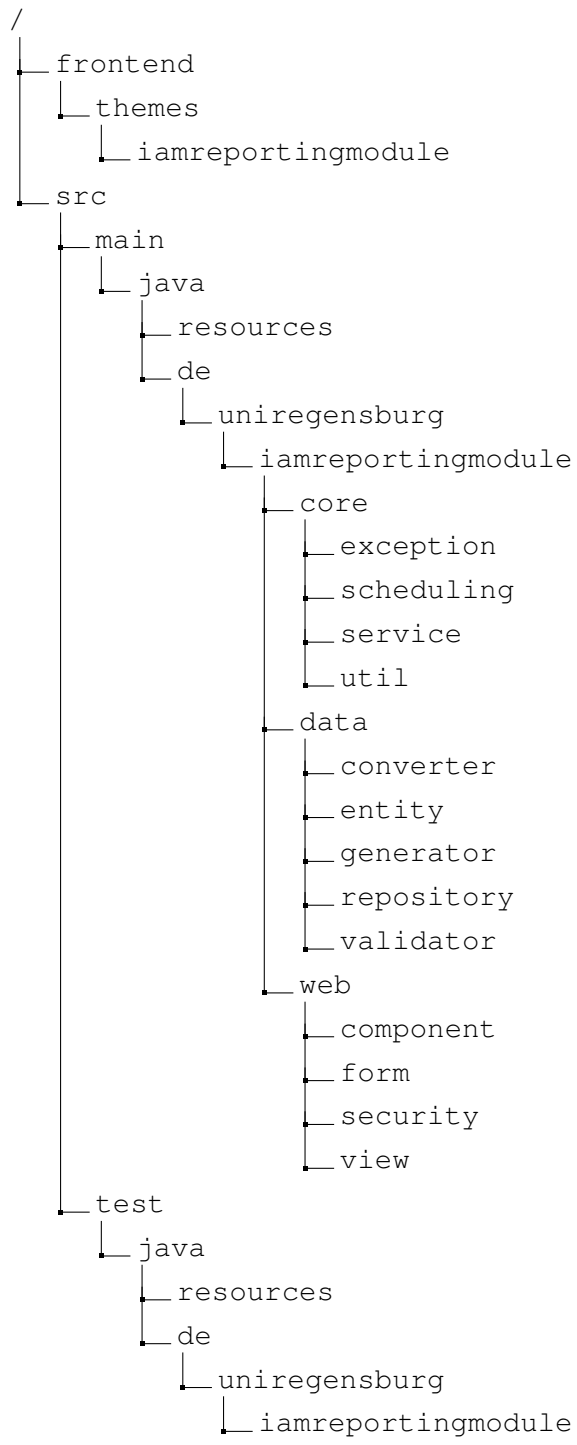

Abb. 6.1 Projektstruktur

Die Ordnerstruktur der Klassen der Anwendung (unter `main`) und der Klassen zum Testen der Anwendung (unter `test`) folgen demselben Schema, das im folgenden erläutert wird. Die Java-Packages sind gemäß dem Architekturmuster MVC aufgeteilt: Die Modelle sind im Package `data`, die Steuerung im Package `core` und die Präsentation im Package `web` aufzufinden. Im Package `core` wird zwischen Exceptions (`exception`), Quartz-Aufgaben (`scheduling`), Spring Services (`service`) und allgemeinen Hilfsklassen (`util`) weiter untergliedert. Das Package `data` strukturiert sich

in Klassen zur Datenkonvertierung (`converter`), Datenvalidierung (`validator`), Datengeneration (`generator`) sowie die Entitäten (`entity`) und deren JPA-Repositorys (`repository`). Die Vaadin Frontend-Klassen befinden sich im Package `web`. Es ist untergliedert in Komponenten (`component`), Formulare (`form`), Sicherheitskonfigurationen (`security`) und Ansichten (`view`). Das Design der Vaadin Komponenten kann unter `/frontend/themes/iamreportingmodule` durch Anpassung des CSS konfiguriert werden.

6.3 Funktionen

6.3.1 Authentifizierung und Autorisierung

Um zwischen Administratoren und normalen Benutzern im Prototyp zu unterscheiden, werden Authentifizierung und Autorisierung rudimentär umgesetzt. Die Bibliothek Spring Boot Starter Security unterstützt dabei, indem sie grundlegende Funktionen zur Authentifizierung und Autorisierung mit sich bringt.

Zum Einsatz kommt die Klasse User (siehe Datenmodell in Kapitel 5.2), um die Anmeldedaten der Anwender persistent abzuspeichern. Bevor ein Anwender mit der Anwendung interagieren kann, muss sich dieser mit dessen Benutzername und Passwort authentifizieren (siehe Abb. 6.2). Die Klasse `LoginView` zeigt dabei die Vaadin Flow-Komponente `LoginForm` an.

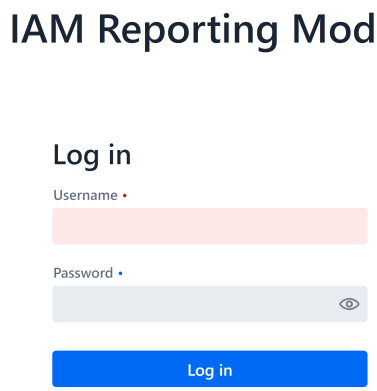

Abb. 6.2 Implementierung: Login

Zur Unterscheidung zwischen Benutzern und Administratoren wird das Attribut *admin* verwendet. Besitzt ein Benutzer das Attribut *admin* mit dem Wahrheitswert *wahr*, so wird diesem die Rolle *ADMIN* zugewiesen. Alle Benutzer können das Dashboard einsehen. Benutzer mit der Rolle *ADMIN* können zusätzlich die Konfiguration verändern.

6.3.2 Layout

In der Abb. 6.3 ist das Layout der Anwendung abgebildet. In der Mitte wird der Inhalt der jeweiligen Seite angezeigt. Auf der linken Seite befindet sich eine vertikale Navigationsleiste, um zwischen dem Dashboard und den Konfigurationsseiten zu wechseln.

Oben in der Mitte wird der aktuelle Seitentitel angezeigt. Links neben dem Titel ist ein Button, um die Navigationsleiste ein- und auszublenden. Oben rechts können seitenspezifische Buttons angezeigt werden, um weitere Aktionen für eine Seite hinzuzufügen. Dieses Layout ist in der Klasse `MainLayout` definiert und wird von allen weiteren Benutzeroberflächen verwendet.

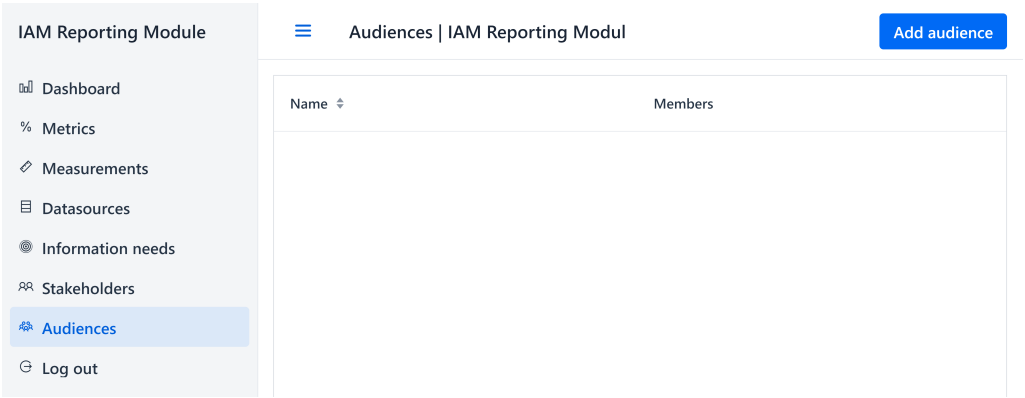

Abb. 6.3 Implementierung: Layout

6.3.3 Konfiguration

Bevor Metriken im Dashboard angezeigt werden können, müssen diese und die weiteren Objekte des Typs Messwert, Datenquelle, Informationsbedarf, Zielgruppe und Stakeholder durch einen Administrator konfiguriert werden. Das IAM Reporting Modul bietet für jeden Typ Benutzeroberflächen an, um alle Objekte des Typs anzuzeigen, ein neues Objekt zu erstellen, ein Objekt zu editieren und ein Objekt zu löschen.

Da sich der Aufbau und die Funktionsweise der Benutzeroberflächen der unterschiedlichen Typen ähnelt, wird diese im Folgenden am Beispiel vom Typ Metrik erläutert. Beim Aufbau der Benutzeroberfläche wurde sich an den Mockups des Kapitels 5.4 orientiert. Die Funktionsweise lehnt sich an die im Kapitel 5.3 entworfenen Prozesse an. Auf spezielle Eigenschaften bei der Konfiguration von Datenquellen, Messwerten und Metriken wird zu einem späteren Zeitpunkt eingegangen.

Übersicht aller konfigurierten Metriken. Über den Navigationspunkt „Metrics“ gelangt der Administrator zur Übersicht aller Metrikkonfigurationen (siehe Abb. 6.4). Die Benutzeroberfläche ist in der Klasse `MetricsView` definiert. Diese enthält ein `Grid` (Vaadin Flow-Komponente für Tabellen), das über Methodenaufruf des Services für Metriken mit allen Metrikkonfigurationen befüllt wird. Folglich werden in der Übersicht die Konfigurationen der Metriken tabellarisch dargestellt. Um eine Konfiguration zu editieren, ist die entsprechende Zeile anzuklicken. Ein entsprechender Listener leitet den Administrator zum Editieren der Konfiguration weiter. Zur Anlage einer neuen Konfiguration ist rechts oben der Button „neue Metrik“ auszuwählen, der zur Anlage einer neuen Metrikkonfiguration verlinkt.

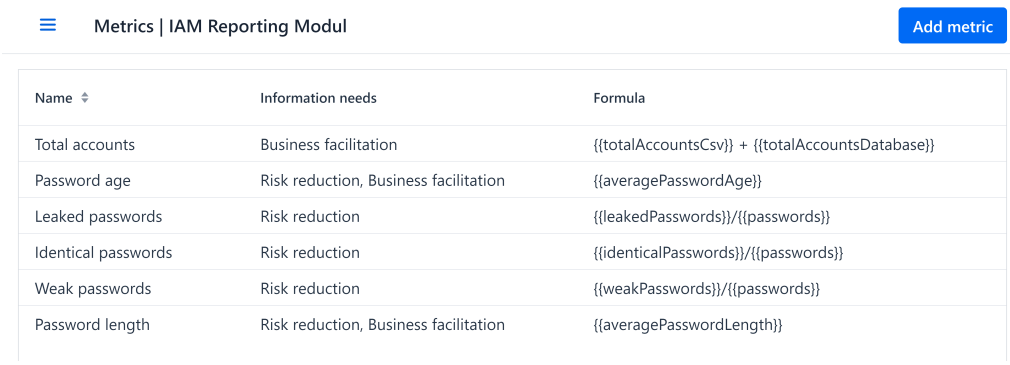

Abb. 6.4 Implementierung: Übersicht aller Metrikkonfigurationen

Anlage der Konfiguration einer Metrik. Zur Anlage der Konfiguration einer Metrik wird dem Administrator ein Formular mit allen konfigurierbaren Feldern einer Metrik angezeigt und kann durch diesen ausgefüllt werden. Die Klasse `AddMetricView` zeigt dazu das Formular `MetricForm` an. Weiter werden Buttons zum Abbrechen und Speichern des Formulars angezeigt. Bricht der Administrator ab, werden die Änderungen verworfen. Wird die im Formular angegebene Konfiguration gespeichert, wird der Service zum Speichern der Konfiguration der Metrik aufgerufen. In beiden Fällen wird der Administrator zurück zur Übersicht aller Konfigurationen geleitet. Ausschließlich für die Konfiguration der Metrik ist ein weiterer Button zum Testen der Formel vorhanden. Auf die Formel der Metrik wird im Abschnitt 6.3.6 eingegangen.

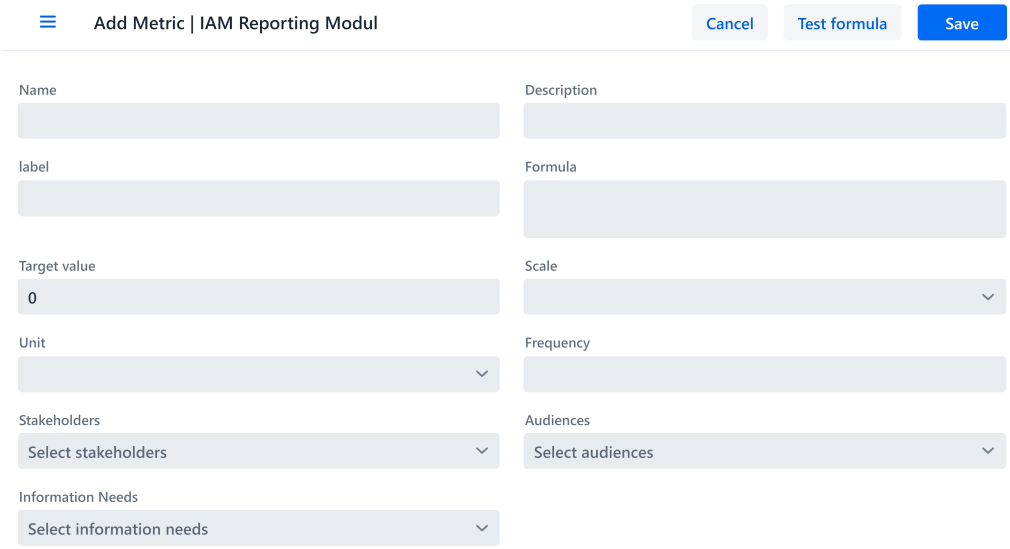

Abb. 6.5 Implementierung: Anlage der Konfiguration einer Metrik

Das Formular `MetricForm` beinhaltet alle Felder zur Konfiguration von Metriken. Im Falle der Metrik handelt es sich bei den Feldern um Vaadin-Komponenten für Textfelder (`TextField`), Textbereiche (`TextArea`) und Dropdown-Menüs (`ComboBox` und `MultiSelectComboBox`). Die Validierung und Konvertierung der Formularfelder erfolgt mithilfe einer Instanz der Klasse `BeanValidationBinder`.

Die Klasse `BeanValidationBinder` ist Teil der Vaadin-Bibliotheken. Instanzen der Klasse `BeanValidationBinder` ermöglichen es, eine Klasse – wie beispielsweise die Klasse `Metric` – mit einem Formular zu verbinden. Verbinden bedeutet in diesem Fall, dass die Attributwerte von Objekten der angegebenen Klasse automatisch in die vorgegebenen Formularfelder übernommen werden. Gleichzeitig ermöglicht die Klasse `BeanValidationBinder` auch, die Formularfelder anhand der Klassendefinition zu validieren, umzuwandeln und aus den Formularwerten ein Klassenobjekt zu erstellen. Die Funktionalität zur Validierung und Konvertierung von simplen und einigen weiteren Datentypen bringt Vaadin bereits mit. Für eigene komplexe Datentypen – wie beispielsweise die Klassen `Formula` oder `Frequency` – mussten zuerst Funktionen zur Konvertierung oder auch Validierung definiert werden.

Editieren der Konfiguration einer Metrik. Die Benutzeroberfläche zum Editieren der Konfiguration einer Metrik (siehe Abb. 6.6) baut auf der Benutzeroberfläche zur Anlage der Konfiguration einer Metrik auf und ist ergänzt um einen Button zum Löschen der Konfiguration. Weiter gibt die Klasse `EditMetricView` dem Formular ein Objekt der Klasse `Metric` mit. Somit wird das Formular mit der aktuellen Konfiguration befüllt. Außerdem werden im speziellen Fall der Konfiguration der Metrik und des Messwerts unterhalb des Formulars die neusten zehn Metrik- bzw. Messwerte und deren Erstellungszeitstempel angezeigt.

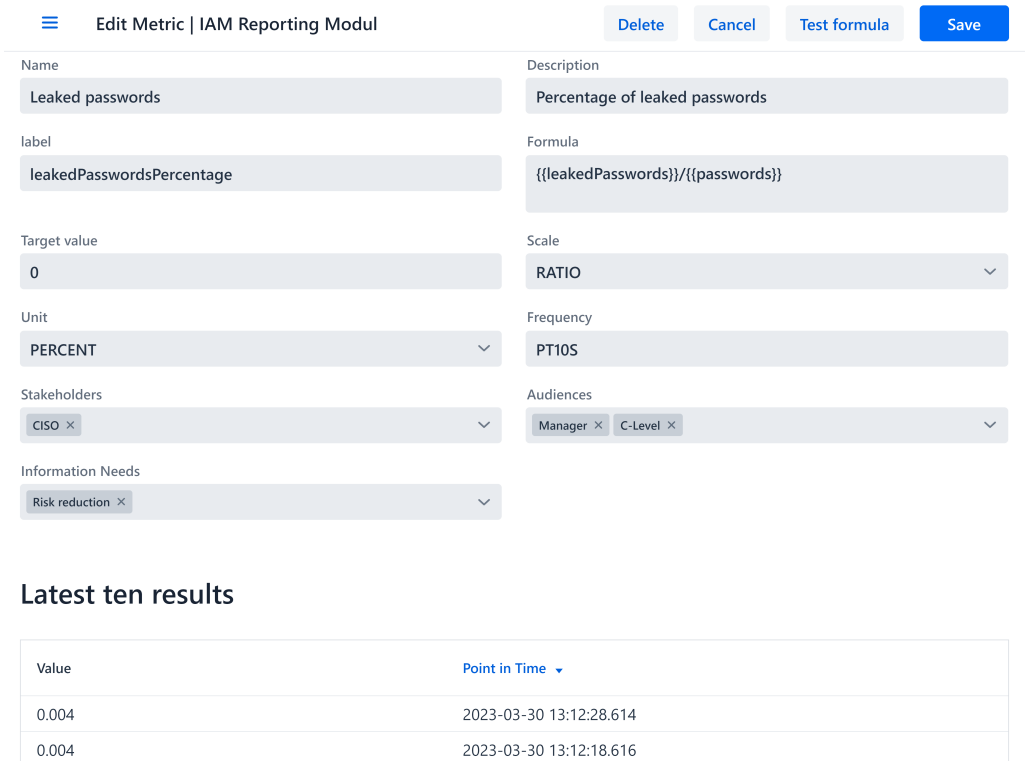

Abb. 6.6 Implementierung: Editieren der Konfiguration einer Metrik

Löschen der Konfiguration einer Metrik. Zusätzlich zu den Aktionen Abbrechen und Speichern ist in der Klasse `EditMetricView` auch eine Aktion zum Löschen der Konfiguration der Metrik definiert: Bei der Auswahl des Buttons „Delete“ wird der in Abb. 6.7 dargestellte Dialog zur Bestätigung des Löschvorgangs geöffnet.

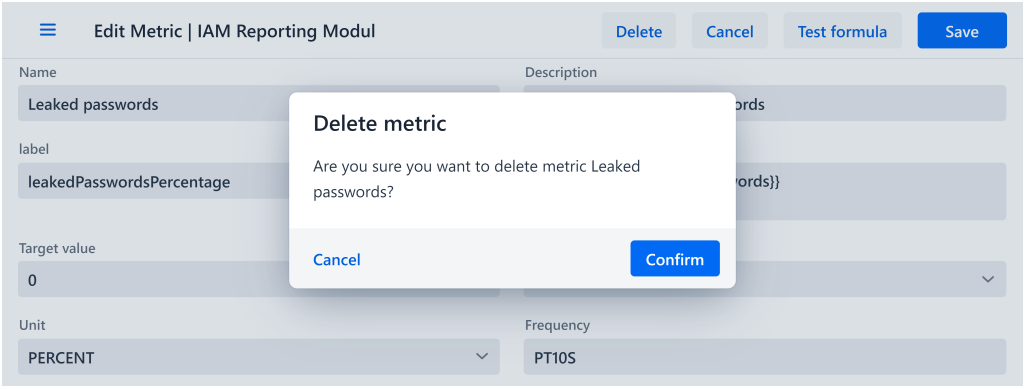

Abb. 6.7 Implementierung: Löschen der Konfiguration einer Metrik

Bestätigt der Administrator den Löschvorgang, wird die Konfiguration der Metrik gelöscht und der Administrator zur Übersicht der konfigurierten Metriken weitergeleitet. Im Falle der Konfiguration der Metrik und des Messwerts werden zusätzlich zur Konfiguration auch die berechneten oder ermittelten Werte gelöscht. Bricht der Administrator den Dialog ab, so kann dieser die Konfiguration weiter bearbeiten.

Ein Löschen ist im Allgemeinen nur möglich, wenn keine anderen Konfigurationen auf der zu löschenden Konfiguration aufbauen. Verwendet beispielsweise eine andere Metrikkonfiguration die zu löschende Metrik in ihrer Formel, so kann eine Löschung nicht durchgeführt werden. Ist eine Löschung nicht möglich, so wird der Administrator mit einem Hinweis darauf aufmerksam gemacht und der Dialog geschlossen.

6.3.4 Anbindung von Datenquellen

Zur Konfiguration von Datenquellen wurde im Design ein einheitliches Formular für alle Typen von Datenquellen vorgeschlagen, das nach Auswahl des Typs um weitere Felder erweitert wird (vgl. Anlegen einer Datenquelle in Abschnitt 5.4). Um weiterhin die volle Funktionalität von der Klasse `BeanValidationBinder` zum Verbinden, Konvertieren und Validieren der Formularfelder nutzen zu können, ist es erforderlich für jeden Typ von Datenquelle ein eigenes Formular und separate Benutzeroberflächen zur Anlage, zum Editieren und zum Löschen der Konfiguration von Datenquellen anzulegen. Deswegen wird nach Klicken des Buttons „Add datasource“ zuerst in einem Dialog der Typ der Datenquelle abgefragt (siehe Abb. 6.8), bevor der Administrator zur Benutzeroberfläche zur Anlage der Konfiguration der jeweiligen Datenquelle weitergeleitet wird.

Im Falle einer manuellen Datenquelle ist neben den allgemeinen Angaben für Datenquellen im Formular `ManualDataSourceForm` der manuelle Wert einzugeben.

Bei Datenquellen vom Typ Datenbank sind im Formular `DatabaseDataSourceForm` zusätzlich zu den allgemeinen Angaben das DBMS

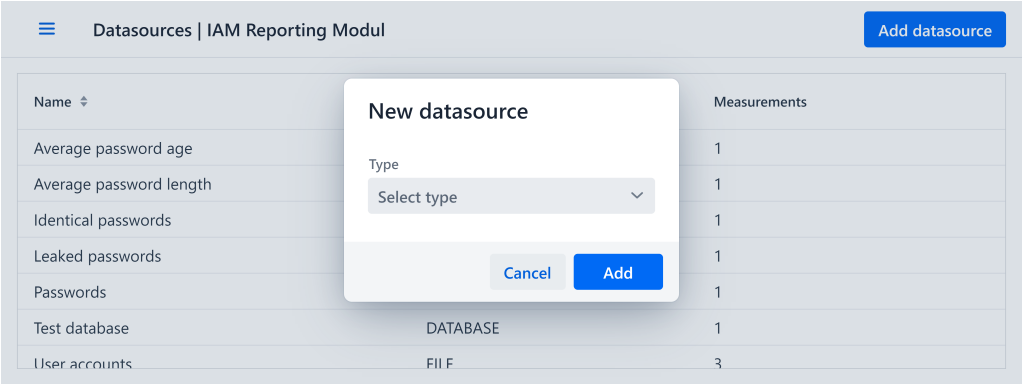

Abb. 6.8 Implementierung: Auswahl des Typs der Datenquelle

auszuwählen und die Verbindungsdaten der Datenbank anzugeben sowie die Verbindung zur Datenbank zu Testen. Im Prototyp ist die Anbindung von PostgreSQL-DBMS implementiert. Weitere DBMS können in zukünftigen Entwicklungsiterationen hinzugefügt werden. Dies kann erreicht werden durch die Einbindung von für das DBMS notwendige Java-Bibliotheken und durch Hinzufügen der DBMS-spezifischen Funktionen, um Datenbankverbindungen zu dem DBMS aufzubauen.

Um eine Datei als Datenquelle hinzuzufügen, ist – wie in Abb. 6.9 dargestellt – im Formular `FileDataSourceForm` zusätzlich zu den allgemeinen Angaben eine Datei hochzuladen. Die Implementierung erkennt den Dateityp automatisch anhand des Multipurpose Internet Mail Extensions (MIME)-Typs und erlaubt nur das Hochladen von Dateien zugelassener Dateitypen. Im Prototyp ist aktuell ausschließlich der MIME-Typ `text/csv` (CSV-Dateien) erlaubt. Die Felder Dateiname und Dateityp sind schreibgeschützt und werden automatisch nach Hochladen der Datei befüllt. Die Datei wird – wie im Datenmodell (vgl. Kapitel 5.2) modelliert – in der Klasse `FileDataSource` als Attribut gespeichert und somit auch in der Datenbank persistent abgelegt.

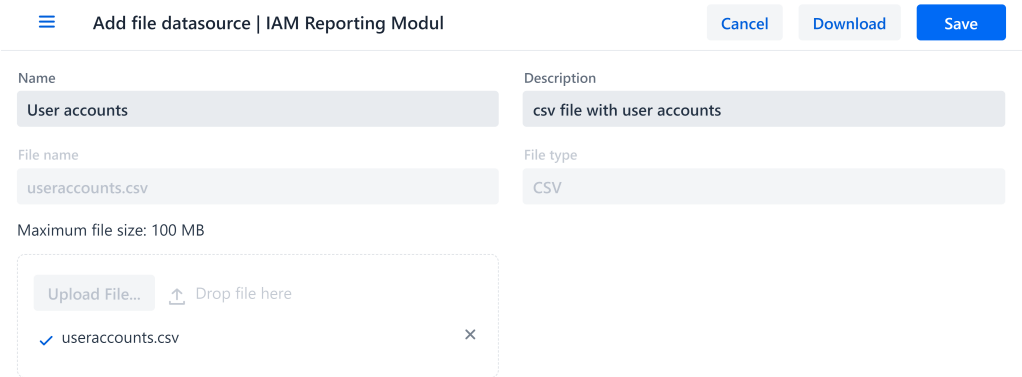

Abb. 6.9 Implementierung: Anlage einer Datei als Datenquelle

6.3.5 Ermittlung von Messwerten

In der Konfiguration der Messwerte sind neben den allgemeinen Angaben für alle Messwerte auch – je nach ausgewählter Datenquelle – weitere spezifische Angaben zu tätigen.

Die spezifischen Angaben unterscheiden sich je nach Typ der Datenquelle: Bei Datenbanken ist eine SQL-Abfrage, bei CSV-Dateien sind die Spalte, das Trennzeichen und die Aggregationsmethode anzugeben und bei manuellen Eingaben als Datenquelle sind keine zusätzlichen Angaben notwendig.

Anders als bei der Konfiguration der Datenquellen gibt es für die Konfiguration der Messwerte nur ein Formular: `MeasurementForm`. Nach Auswahl der Datenquelle werden für den Typ der Datenquelle spezifische Felder eingeblendet (siehe Abb. 6.10). Diese Felder können nicht durch die Klasse `BeanValidationBinder` validiert und konvertiert werden. Die Validierung und die Konvertierung der zusätzlichen Felder sind im Formular `MeasurementForm` manuell implementiert. Gespeichert werden die Werte der zusätzlichen Felder im Attribut `Map<String, String> attributes` der Klasse `Measurement`.

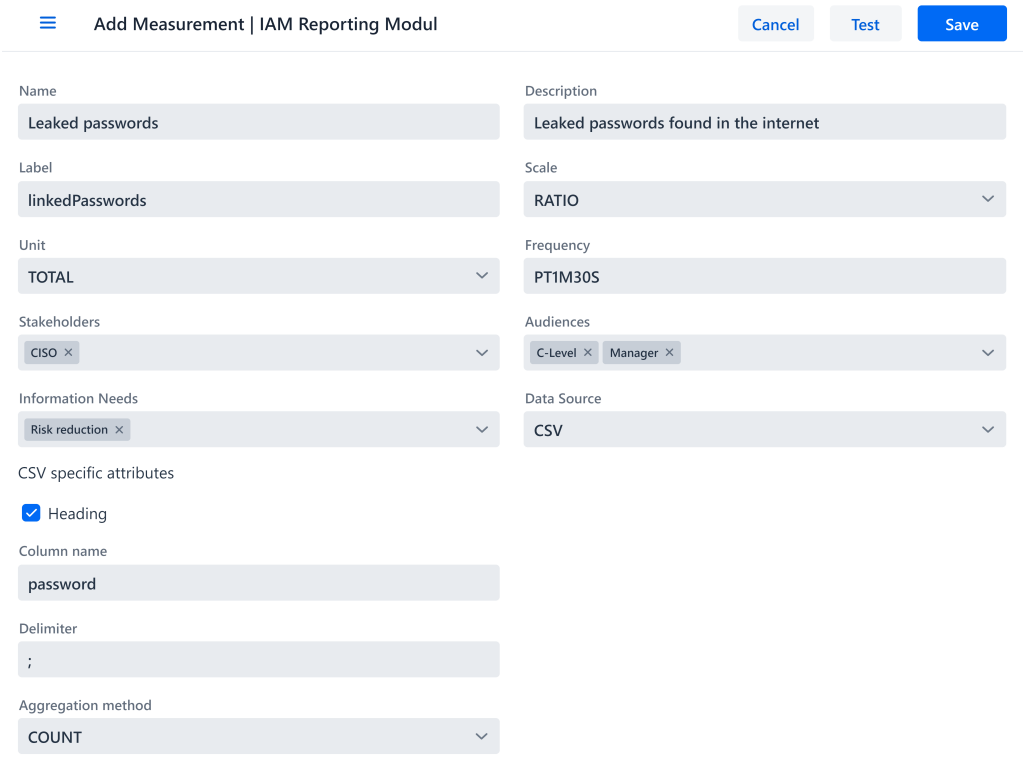

Abb. 6.10 Implementierung: Anlage der Konfiguration eines Messwerts mit einer CSV-Datei als Datenquelle

Die Benutzeroberfläche zur Konfiguration von Messwerten ermöglicht das Testen der angegebenen Konfiguration. Bei dem Test versucht das IAM Reporting Modul eine Messung durchzuführen. Im Falle einer manuellen Datenquelle wird der Wert der manuellen Datenquelle zurückgegeben. Komplexer gestaltet sich die Ermittlung von Messwerten in Datenbanken und CSV-Dateien. Für Datenbanken wurde eine Klasse `DatabaseUtil` implementiert, die Methoden zum Aufbau einer Datenbankverbindung und Abfrage einer Datenbank zur Verfügung stellt. Ähnlich dazu gibt es für CSV-Dateien eine Klasse `CsvUtil`, um Messungen in CSV-Dateien durchzuführen.

Der Klasse `DatabaseUtil` wird bei Instanziierung die Datenquelle (beinhaltet die

Verbindungsdaten) mitgegeben. Bei Methodenaufruf zur Durchführung der Messung wird die Variable `attributes` der Klasse `Measurement` (beinhaltet die SQL-Abfrage) angegeben. Ist die Ausführung der SQL-Abfrage erfolgreich, wird dem Administrator der Wert der Messung angezeigt. Andernfalls werden dem Administrator die Hintergründe des Scheiterns der Abfrage gemeldet.

Ähnlich zur Klasse `DatabaseUtil` wird auch der Klasse `CsvUtil` bei Instanziierung die Datenquelle mitgegeben. Bei Aufruf der Methode `measure` wird die Variable `attributes` der Klasse `Measurement` übergeben. Anhand der angegeben Spalte und der Aggregationsmethode wird der Wert berechnet. Dem Administrator wird daraufhin der Wert oder im Fehlerfall die Fehlermeldung angezeigt. Folgende Aggregationsmethoden sind im Prototyp implementiert: Zählen aller Zeilen, Summieren aller Werte der Zeilen, kleinster Wert aller Zeilen, größter Wert aller Zeilen, Durchschnittswert aller Zeilen und Median aller Zeilen.

6.3.6 Berechnung von Metrikwerten

Wie in Abb. 6.6 dargestellt, ist zur Berechnung der Metrikwerte durch den Administrator eine Formel anzugeben. Die Herausforderung bei der Implementierung der Berechnung der Metrikwerte ist, die Formel – welche als Zeichenkette angegeben ist – als mathematische Formel korrekt zu interpretieren und zu berechnen. Neben den mathematischen Grundrechenarten (vgl. funktionale Anforderung F48) und weiteren algebraischen Methoden (vgl. funktionale Anforderung F49) sind ebenso Variablen für Messwerte und Metriken Teil der Formel und müssen berücksichtigt werden. Die Klasse `FormulaUtil` implementiert die Funktionalität zur Berechnung von Formeln.

Die Implementierung der mathematischen Grundrechenarten und Methoden greift auf bestehende Bibliotheken zur Berechnung von Formeln zurück. Die Anforderungen an eine solche Java Bibliothek sind:

/M10/ Die Bibliothek MUSS das Ergebnis einer Formel – angegeben als Zeichenkette – korrekt berechnen.

/M11/ Die Bibliothek MUSS die Syntax einer Formel – angegeben als Zeichenkette – validieren.

/M12/ Die Bibliothek MUSS Open Source sein.

/M13/ Das letzte Release der Bibliothek SOLLTE höchstens drei Jahre zurück liegen.

/M14/ Das letzte Release der Bibliothek MUSS keine bekannten Schwachstellen aufweisen.

In der Tab. 6.2 sind die evaluierten Java Bibliothek aufgestellt. Die Erfüllung einer Anforderung ist mit ✓ und ein Nichterfüllen mit ✗ markiert.

Bibliothek	M10	M11	M12	M13	M14
EvalEx [a]	✓	✓	✓	✓	✓
exp4j [b]	✗	✓	✓	✗	✗
Javaluator [c]	✗	✗	✓	✗	✓
mXparser [d]	✓	✓	✗	✓	✓

Tab. 6.2 Mathematische Bibliotheken

[a] https://github.com/ezylang/EvalEx
[b] https://www.objecthunter.net/exp4j
[c] https://javaluator.sourceforge.net/en/home
[d] https://mathparser.org

Die Bibliothek *EvalEx* erfüllt alle Anforderungen zur Berechnung und Validierung von Formeln, ist Open Source und unter der Apache 2.0 Lizenz lizenziert, das letzte Release liegt keinen Monat zurück (19.03.2023) und weist keine bekannten Schwachstellen auf [Kli22; Mvn23b].

In der Bibliothek *exp4j* ist keine Verwendung von Summe und Produkt in der Formel möglich, eine Methode zur Validierung der Formel wird mitgebracht, sie ist Open Source unter der Apache 2.0 Lizenz lizenziert, das letzte Release liegt über drei Jahre zurück (30.01.2017) und verwendete Bibliotheken weisen bekannte Schwachstellen auf [Ass17; Mvn23c].

Die Bibliothek *Javaluator* ermöglicht keine Verwendung eines Produkts in der Formel und bringt keine Validierung der Formel mit. Sie ist Open Source unter der LGPL 3.0 Lizenz lizenziert, das letzte Release liegt über drei Jahre zurück (15.08.2019) und weist keine bekannten Schwachstellen auf [JavoJ; Mvn23d].

Die letzte evaluierte Bibliothek ist *mXparser*: Sie erfüllt alle Anforderungen zur Berechnung und Validierung von Formeln, ist Open Source mit einem Dual Licence Agreement lizenziert, das letzte Release liegt weniger als ein Jahr zurück (08.02.2023) und weist keine bekannten Schwachstellen auf [Gro23a; Mvn23f]. Das Dual Licence Agreement ermöglicht nur eine nicht-kommerzielle Nutzung, für eine kommerzielle Nutzung muss eine kommerzielle Lizenz erworben werden [Gro23b].

Für das IAM Reporting Tool wurde sich für den Einsatz der Bibliothek *EvalEx* entschieden, da sie als einzige Bibliothek alle Anforderungen M10 bis M13 erfüllt. Die Herausforderungen zur Berechnung einer Formel bestehend aus einer Zeichenkette unter Verwendung von mathematischen Grundrechenarten und Methoden sind somit gelöst. Es bleibt noch eine Lösung der Herausforderung – Variablen der Metriken und Messwerte in den Formeln durch ihre tatsächlichen Werte bei Berechnung zu ersetzen – zu finden. Hierzu müssen die Variablen in der Formel gekennzeichnet werden und die Variablennamen eindeutig sein.

In der Implementierung wurde sich für eine Kennzeichnung von Variablen mit zwei sich öffnenden geschweiften Klammern `{{` am Anfang und mit zwei sich schließenden geschweiften Klammern `}}` am Ende entschieden. Als eindeutiger Variablenname könnte das Attribut `id: int` (vgl. Klasse `AbstractEntity` in Abb. 5.13) Einsatz finden.

Dieser Identifikator ist jedoch für die Benutzer des IAM Reporting Tools nicht einfach zu merken. Aus diesem Grund wurde ein weiteres Attribut `label: String` eingeführt, in dem der Anwender selbst einen Variablennamen vergeben kann. Dieser Variablennamen muss eindeutig sein und der Namenskonvention „lowerCamelCase“ folgen.

In der Klasse `FormulaUtil` wird vor Berechnung des Wertes mit *EvalEx* die Zeichenfolge nach Variablen durchsucht und die Variablen mit dem aktuellsten Wert der Metrik oder des Messwerts ersetzt.

6.3.7 Wiederkehrende Ermittlung von Messwerten und Berechnung von Metrikwerten

Messwerte und Metriken werden nicht nur einmalig beim Abspeichern oder durch manuelle Aktion berechnet oder gespeichert. Anstelle dessen wird eine Frequenz konfiguriert, in der die Messwerte und Metriken wiederkehrend ermittelt und berechnet werden. Das IAM Reporting Modul benötigt daher eine Funktionalität, um die Ermittlung der Messwerte und die Berechnung der Metrikwerte im Hintergrund mit der konfigurierten Frequenz durchzuführen (vgl. funktionale Anforderungen F50 und F60).

Auf der Suche nach einer passenden Java-Bibliothek zur Ausführung der Aufgaben im Hintergrund wurden die folgenden Anforderungen definiert:

/S10/ Die Bibliothek MUSS Aufgaben wiederkehrend anhand einer festgelegten Frequenz durchführen.

/S11/ Die Bibliothek MUSS die wiederkehrenden Aufgaben persistent über einen Systemneustart hinweg speichern.

/S12/ Die Bibliothek MUSS Open Source sein.

/S13/ Das letzte Release der Bibliothek SOLLTE höchstens drei Jahre zurück liegen.

/S14/ Das letzte Release der Bibliothek MUSS keine bekannten Schwachstellen aufweisen.

Vier Java Bibliotheken wurden als mögliche Lösung identifiziert und die Erfüllung der Anforderungen S10 bis S14 zum Vergleich in Tab. 6.3 gegenübergestellt. Die Erfüllung einer Anforderung ist mit dem bereits vorgestellten Schema ✓ für die Erfüllung und ✗ für die Nichterfüllung der Anforderung markiert. Wie der Tabelle zu entnehmen ist, erfüllt keine der Bibliotheken alle Anforderungen.

Bibliothek	S10	S11	S12	S13	S14
DB Scheduler [a]	✓	✓	✓	✓	✗
JobRunr [b]	✓	✓	✗	✓	✓
Quartz Scheduler [c]	✓	✓	✓	✗	✓
Wisp Scheduler [d]	✓	✗	✓	✓	✓

Tab. 6.3 Bibliotheken zur Planung wiederkehrender Aufgaben

[a] https://github.com/kagkarlsson/db-scheduler
[b] https://www.jobrunr.io/en
[c] https://www.quartz-scheduler.org
[d] https://github.com/Coreoz/Wisp

Die Bibliothek *DB Scheduler* bringt alle Funktionen mit, ist Open Source mit der Apache 2.0 Lizenz lizenziert, das letzte Release liegt kein Jahr (18.11.2022) zurück, verwendet jedoch mehrere Bibliotheken, die bekannte Schwachstellen aufweisen [Kar22; Mvn23a].

JobRunr erfüllt alle Anforderungen bezüglich der Funktionalität, das letzte Release liegt keinen Monat (17.03.2023) zurück, weist keine bekannten Schwachstellen auf und ist Open Source mit der GNU Lesser General Public License Version 3 (LGPLv3) Lizenz lizenziert [JoboJa; Mvn23e]. Diese LGPLv3 Lizenz kommt mit einer Einschränkung einher, die den Einsatz von maximal 100 wiederkehrenden Aufgaben erlaubt. Für mehr als 100 wiederkehrende Aufgaben ist eine kommerzielle Lizenz zu erwerben [JoboJb].

Die Bibliothek *Quartz Scheduler* erfüllt alle funktionalen Anforderungen, das letzte Release liegt über drei Jahre (23.10.2019) zurück, ist Open Source mit der Apache 2.0 Lizenz lizenziert und weist keine bekannten Schwachstellen auf [TeroJ; Mvn23g].

Die letzte untersuchte Bibliothek ist *Wisp Scheduler*. Diese Bibliothek ermöglicht das Ausführen wiederkehrender Aufgaben, diese werden jedoch nicht persistent über einen Systemneustart hinweg gespeichert [Cor22]. Das letzte Release der Bibliothek liegt weniger als ein Jahr (12.09.2022) zurück, die Bibliothek ist Open Source mit der Apache 2.0 Lizenz lizenziert und weist keine bekannten Schwachstellen auf [Mvn23h].

Da keine der untersuchten Bibliotheken alle Anforderungen erfüllt, wurde die Bibliothek gewählt, die alle Anforderungen mit dem Schlüsselwort MUSS erfüllt. Dabei handelt es sich um die Bibliothek *Quartz Scheduler*. Die Anforderung S13 mit dem Schlüsselwort SOLLTE ist nicht erfüllt: Das letzte Release wurde vor über drei Jahren veröffentlicht, im Quellcode auf GitHub wurden jedoch seitdem Änderungen durchgeführt [Ter22].

Die Bibliothek Quartz Scheduler bringt eine Integration für Spring Boot mit, welche den Einsatz vom Quartz Scheduler unter minimaler Konfiguration ermöglicht. Einzig die Datenbank für den Quartz Scheduler musste konfiguriert werden.

Für die beiden Aufgabentypen „Ermittlung von Messwerten“ und „Berechnung von Metrikwerten“ wurden zwei Klassen `MeasureJob` und `CalculateJob` erstellt. In beiden Klassen sind die durchzuführenden Aktionen bei Ausführung der Aufgabe definiert: Ermittlung des Messwerts oder Berechnung des Metrikwerts sowie Speicherung des Ergebnisses in der Datenbank.

Bei der Anlage, der Änderung oder der Löschung der Konfiguration von Messwerten und Metriken werden Methoden der Klasse `JobSchedulingService` aufgerufen, um die Aufgaben zu planen, zu aktualisieren oder zu löschen. Die Klasse `JobSchedulingService` bildet dabei die Schnittstelle zum Quartz-Scheduler, erstellt die Aufgaben anhand der Klassen `MeasureJob` und `CalculateJob`, plant deren Ausführung anhand der gegebenen Frequenz und aktualisiert bzw. löscht die Aufgabendaten bei Aktualisierungen und Löschungen. Das Wiederholungsintervall ist in der Konfiguration von Metriken und Messwerten als Zeitspanne gemäß ISO 8601-Standard [Int19a] anzugeben.

6.3.8 Dashboard

Wie in Abb. 6.11 dargestellt, zeigt das Dashboard die aktuellen Werte aller Metriken für einen Informationsbedarf an. Den Informationsbedarf wählt der Benutzer im Dropdown-Menü aus, das IAM Reporting Modul zeigt daraufhin die Metriken an. Dabei werden nur die Metriken angezeigt, die für eine Zielgruppe bestimmt sind, welcher der aktuelle Benutzer angehört.

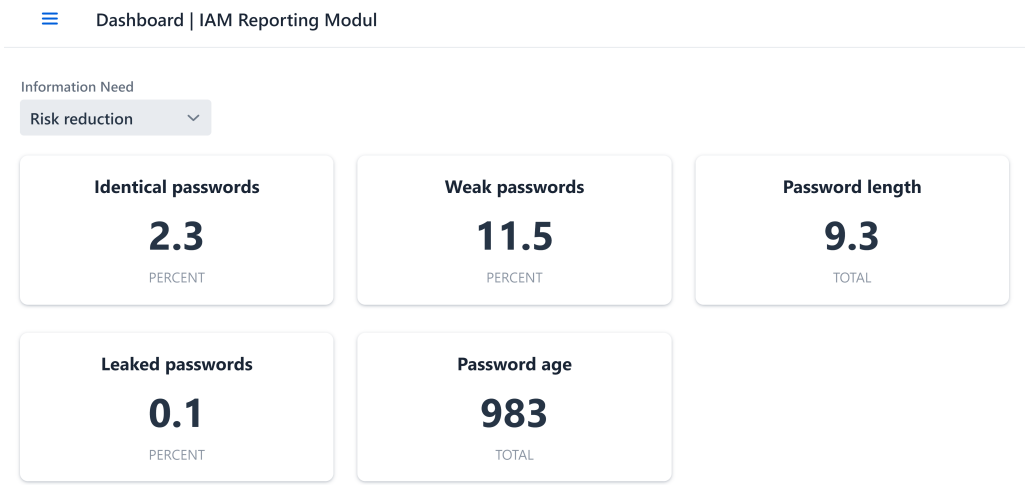

Abb. 6.11 Implementierung: Dashboard

Die Klasse `DashboardView` ist zuständig für das Dashboard. Neben dem Dropdown-Menü (`ComboBox`) kommt ein `GridLayout` der Bibliothek Lumogridlayout zum Einsatz. Das GridLayout ermöglicht die Verwendung eines CSS-Grids. Das Grid besitzt drei Spalten, in denen die Metriken auf Karten abgebildet werden.

Die Karten sind eine benutzerdefinierte Vaadin-Komponente `Indicator`. Es handelt sich dabei um ein `Div`-Element mit den drei Textfeldern (`Text`) Titel, Wert und Einheit sowie einen Link (`Anchor`). Im Feld Titel wird der Name der Metrik angezeigt, im Feld Wert der neuste berechnete Wert und im Feld Einheit die Einheit der Metrik. Der Link erstreckt sich über die Fläche der gesamten Karte und verlinkt auf die Detailansicht der Metrik. Neben der Klasse `Indicator` gibt es eine CSS-Datei `indicator.css`, welche die Beschreibung des Designs der Komponente beinhaltet.

In der Detailansicht der Metrik (siehe Abb. 6.12) sind die neusten zehn Werte und der Zeitstempel der Berechnung in einer Tabelle aufgelistet.

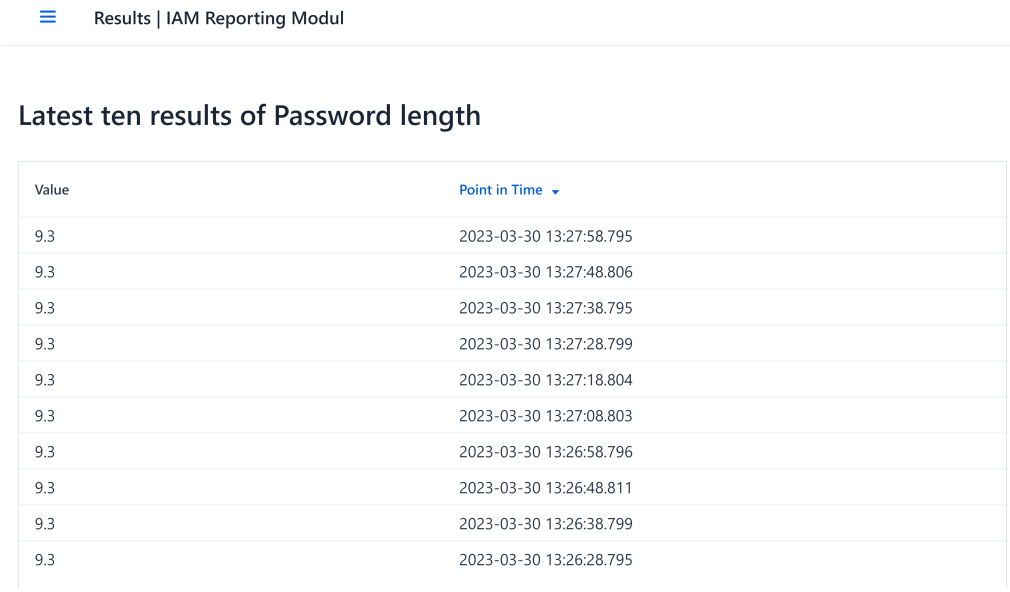

Abb. 6.12 Implementierung: Detailansicht einer Metrik

6.4 Installation der Entwicklungsumgebung

Installation. Der Quellcode des Prototyps ist auf GitHub (https://github.com/baua1310/iam-reporting-module) unter der MIT Lizenz veröffentlicht und kann von dort heruntergeladen werden. Ebenso ist der Quellcode im digitalen Anhang zu finden. Alle nachfolgend beschriebenen Befehle sind im Hauptverzeichnis des Quellcodes auszuführen. Um das IAM Reporting Modul aus dem Quellcode zu bauen und zu starten, ist die folgende Software auf dem lokalen Rechner notwendig:

- Java (https://adoptium.net/de/installation/)
- Apache Maven (https://maven.apache.org/install.html)
- PostgreSQL (https://www.postgresql.org/download/)

Getestet wurde das Bauen und das Starten des IAM Reporting Moduls mit den im Kapitel 6.1 beschriebenen Versionen der Software.

Entwicklung. Zur Entwicklung des IAM Reporting Moduls kann ein beliebiger Editor oder eine beliebige Entwicklungsumgebung eingesetzt werden. Mit dem Maven-Befehl `mvn spring-boot:run` wird das IAM Reporting Modul im Entwicklungsmodus gestartet und ist unter der URL http://localhost:8080 in einem beliebigen, zeitgemäßen Webbrowser erreichbar. Eine Anmeldung ist möglich mit einem Standardbenutzer, unter

Angabe des Benutzernamens `user` und des Passworts `password`, oder mit einem Benutzer mit Administratorenrechten, unter Angabe des Benutzernamens `admin` und des Passworts `password`.

Die Datenbank wird dabei bei jedem Start neu erstellt und mit Musterdaten gefüllt. Zum Einsatz kommt dabei das DBMS H2. Die Anwendung unterstützt Live Reload. Bei den meisten Codeänderungen ist es daher ausreichend das Projekt neu zu kompilieren. Dies kann entweder mit integrierten Tools der Entwicklungsumgebung oder mit dem Maven-Befehl `mvn compile` erfolgen.

Betrieb. Um das IAM Reporting Modul für den Betrieb zu bauen, ist folgender Maven-Befehl auszuführen: `mvn clean install -Pproduction -DskipTests`. Die Anwendung wird daraufhin als Java Archive-(JAR-)Datei gepackt und kann mit dem Befehl `java -jar target/iamreportingmodule-1.0-SNAPSHOT.jar` gestartet werden. Das IAM Reporting Modul ist daraufhin unter http://localhost:8080 erreichbar.

Im Betrieb findet das DBMS PostgreSQL Verwendung. Vor dem ersten Start ist es notwendig, eine Datenbank und einen Benutzer mit Passwort und ausreichenden Berechtigungen für das IAM Reporting Modul anzulegen. Die im Listing 6.1 aufgelisteten Befehle müssen dazu in der Datenbank ausgeführt werden:

```
1 CREATE USER <user> WITH ENCRYPTED PASSWORD '<password>';
2 CREATE DATABASE <database> WITH OWNER <user>;
```

Listing 6.1 Datenbank und Datenbankbenutzer erstellen

Weiter sind die Datenbankverbindungsdaten in der Konfiguration der Anwendung (`application-production.properties`) anzugeben. Die anzupassenden Zeilen fünf bis sieben sind im Listing 6.2 aufgelistet: Die Platzhalter `<database>`, `<user>` und `<password>` müssen dabei durch den Namen der Datenbank, den Namen des Datenbankbenutzers und das Passwort des Datenbankbenutzers ersetzt werden. Die benötigten Datenbanktabellen erstellt die Anwendung beim ersten Start.

```
5 spring.datasource.url=jdbc:postgresql://localhost:5432/<database>
6 spring.datasource.username=<user>
7 spring.datasource.password=<password>
```

Listing 6.2 application-production.properties

Neue Benutzer des IAM Reporting Moduls sind manuell in der Datenbank zu pflegen. In Listing 6.3 ist der Befehl zur Anlage eines neuen Benutzers abgedruckt. Die Platzhalter – gekennzeichnet durch eckige Klammern (<>) – sind wie folgt zu ersetzen: `<id>` ist ein im System eindeutiger Identifikator des Benutzers in Form eines Universally Unique Identifiers (UUIDs). `<username>` ist der eindeutige Benutzername und `<first_name>` sowie `<last_name>` sind Vor- und Nachname des Benutzers. `<password>` ist das mit bcrypt gehashte Passwort des Benutzers. Der Wahrheitswert `<admin>` gibt an, ob ein Benutzer Administrator ist oder nicht.

```
INSERT INTO user_account (
id, username, password, admin, first_name, last_name
) VALUES (
'<id>','<username>','<password>',<admin>,'<first_name>','<last_name>'
);
```

Listing 6.3 IAM Reporting Modul Benutzer erstellen

Musterdaten. Während der Entwicklung werden beim Start der Anwendung Musterdaten generiert und in der Datenbank gespeichert. Somit können durchgeführte Änderungen in der Anwendung überprüft werden, ohne dass manuell Daten angelegt werden müssen. Die Klasse `DataGenerator` besitzt eine Methode `generateData`, die beim Starten der Anwendung ausgeführt wird. Wird die Anwendung für den Betrieb gebaut, gilt dies nicht. Die Methode `generateData` wird nicht beim Start der für den Betrieb gebauten Anwendung ausgeführt.

Tests. Die Anwendung besitzt zum aktuellen Zeitpunkt bis auf wenige Ausnahmen keine automatisierten Tests. Die Ausnahmen sind einige Methoden der Hilfsklassen im Package `util`. Das Testen anhand der Anforderungen im Kapitel 4 kann nur manuell erfolgen.

Kapitel 7

Evaluation

In diesem Kapitel wird die Arbeit evaluiert. Die Evaluation belegt den Nutzen des Artefakts. Hierbei kann die Gültigkeit, Nützlichkeit, Qualität und Effizienz des Artefakts herausgestellt werden. [GH13] In DSR bestehen eine Vielzahl an Methoden, um Artefakte zu evaluieren [HMPR04]. In dieser Arbeit wird anhand des implementierten Prototyps des IAM Reporting Moduls das in dem Kapitel 5 entworfene Design eines IAM Reporting Moduls evaluiert. Zu diesem Zweck wurde der in dem vorangegangen Kapiteln 6 implementierte Prototyp mit den Musterdaten gestartet und auf Konformität mit den Anforderungen aus Abschnitt 4.6 geprüft.

Bei den Musterdaten handelt es sich um fiktive Daten mit zwei Zielgruppen, zwei Stakeholdern, vier Informationsbedarfen, acht Datenquellen, zehn Messwerten und sechs Metriken. Eine der Datenquellen ist eine PostgreSQL-Datenbank, eine weitere ist eine CSV-Datei und die restlichen sieben sind manuelle Datenquellen. Dabei werden drei der Messwerte aus der CSV-Datei erhoben, ein Messwert aus der Datenbank und sechs Messwerte aus den manuellen Datenquellen. Vier der Metriken sind kombiniert aus Messwerten unterschiedlicher Datenquellen, zwei der Metriken aus Messwerten der gleichen Datenquelle.

In der Tab. 7.1 sind die Ergebnisse aufgelistet: Jede Anforderung ist repräsentiert durch seinen Identifikator, dem Schlüsselwort der Anforderung und der Erfüllung der Anforderung in einer Zeile der Tabelle dargestellt. Das Zeichen ✓ spiegelt ein Erfüllen der Anforderung wider, ✗ die Nichterfüllung.

Von insgesamt 58 definierten Anforderungen wurden 53 erfüllt. Von den fünf nicht erfüllten Anforderungen ist eine Anforderung gekennzeichnet durch das Schlüsselwort MUSS, eine Anforderung durch das Schlüsselwort SOLLTE und drei Anforderungen durch das Schlüsselwort WIRD. Von den insgesamt 32 funktionalen Anforderungen sind alle 16 mit dem Schlüsselwort MUSS gekennzeichnete Anforderungen erfüllt. Elf von zwölf der funktionalen Anforderungen mit dem Schlüsselwort SOLLTE sind erfüllt, sowie drei der vier funktionalen Anforderungen mit dem Schlüsselwort WIRD. Von den 18 nicht-funktionalen Anforderungen sind alle Anforderungen bis auf eine mit dem Schlüsselwort WIRD erfüllt. Von den acht Rahmenbedingungen sind sechs erfüllt. Je eine Rahmenbedingung mit den Schlüsselwörtern MUSS und WIRD ist nicht erfüllt.

Anforderung	Schlüsselwort	erfüllt	Anforderung	Schlüsselwort	erfüllt
F10	MUSS	✓	F61	MUSS	✓
F11	SOLLTE	✗	F70	WIRD	✓
F12	WIRD	✓	F71	WIRD	✓
F20	MUSS	✓	N10	MUSS	✓
F21	MUSS	✓	N11	SOLLTE	✓
F22	MUSS	✓	N12	MUSS	✓
F23	SOLLTE	✓	N13	SOLLTE	✓
F24	MUSS	✓	N14	MUSS	✓
F25	SOLLTE	✓	N15	MUSS	✓
F26	SOLLTE	✓	N20	SOLLTE	✓
F30	SOLLTE	✓	N30	MUSS	✓
F31	SOLLTE	✓	N40	SOLLTE	✓
F32	SOLLTE	✓	N50	SOLLTE	✓
F33	SOLLTE	✓	N51	SOLLTE	✓
F34	MUSS	✓	N52	SOLLTE	✓
F35	MUSS	✓	N53	SOLLTE	✓
F40	MUSS	✓	N54	MUSS	✓
F41	MUSS	✓	N60	SOLLTE	✓
F42	WIRD	✗	N61	WIRD	✗
F43	MUSS	✓	N70	SOLLTE	✓
F44	SOLLTE	✓	N80	MUSS	✓
F45	SOLLTE	✓	R01	SOLLTE	✓
F46	SOLLTE	✓	R02	MUSS	✓
F47	MUSS	✓	R03	MUSS	✓
F48	MUSS	✓	R04	MUSS	✓
F49	SOLLTE	✓	R05	SOLLTE	✓
F50	MUSS	✓	R06	MUSS	✓
F51	MUSS	✓	R07	MUSS	✗
F60	MUSS	✓	R08	WIRD	✗

Tab. 7.1 Erfüllung der Anforderungen

Die funktionalen Anforderungen (siehe Abschnitt 4.6.1) wurden überprüft, indem die beschriebenen Funktionen im Prototyp durchgeführt wurden. Die aus den Use Cases Datenquellen verwalten (F20-F26), Messwerte verwalten (F30-F35), Metriken berechnen (F50, F51), Messwerte ermitteln (F60, F61) und Login (F70, F71) abgeleiteten Anforderungen wurden allesamt erfüllt. Beim Use Case Report anzeigen, können die aktuellen Metrikwerte in einem Dashboard (F10) sowie der Verlauf der letzten zehn Metrikwerte (F12) angezeigt werden. Die Metrikwerte werden jedoch nicht in Diagrammen visualisiert (F11). Das Verwalten einer Metrik ist gemäß der Anforderungen (F40, F41, F43-F49) möglich. Einzig die Eingabe der Formel ist bislang in der Benutzeroberfläche nur als einfache Zeichenkette möglich. Ein visueller Formeleditor (F42) zur Unterstützung des Anwenders bei der Eingabe der Formel ist nicht implementiert.

Nicht-funktionale Anforderungen werden oft auch als Anforderungen an die Qualität eines Systems suggeriert [MZN10]. Im Gegensatz zu den funktionalen Anforderungen ist es bei den nicht-funktionalen Anforderungen nicht möglich, durch Überprüfung der beschriebenen Funktion die Erfüllung der Anforderung festzustellen. Je nach Anforderung ist ein Abgleich mit dem Design oder der Implementierung und dem Quellcode erforderlich.

Alle Anforderungen an die Wartbarkeit sind erfüllt: Der Prototyp basiert auf dem Designmuster MVC (vgl. Abschnitt 5.1) und ist mithilfe eines aktuellen Softwarestacks (vgl. Abschnitt 6.1) implementiert. Anforderung N15 ist somit erfüllt. Der Quellcode ist mit einer Open Source Lizenz auf GitHub veröffentlicht (vgl. Abschnitt 6.4). Ebenso wird diese Arbeit veröffentlicht. Durch Veröffentlichung dieser Arbeit und Klassenkommentaren im Quellcode ist die Anforderung N13 erfüllt. Auch die Anforderungen N12 und N12 sind aufgrund der vorangegangenen Beschreibungen erfüllt, da der Prototyp des IAM Reporting Moduls auf einem aktuellen Softwarestack basiert, der Quellcode dokumentiert und anpassbar ist. Die Anforderung N10 ist erfüllt, da im Design des Datenmodells in Kapitel 5.2 berücksichtigt wurde, dass es unterschiedliche Typen von Datenquellen gibt. Diese können – wie beispielhaft anhand der Datenquellen-Typen Datenbank, Datei und manuell aufgezeigt – im Datenmodell und in der Benutzeroberfläche ergänzt werden. Während der Prototyp ausgeführt wird, loggt dieser die durchgeführten Aktionen im Befehlsfenster und der Logdatei `iamreportingmodule.log`. Die Anforderung N11 ist damit erfüllt.

Die Anforderung bezüglich der Kompatibilität N20 ist durch die generische und erweiterbare Schnittstelle zu Datenquellen (vgl. N10) erfüllt: Im aktuellen Implementierungsstand des Prototyps ist eine Anbindung von IAM-Systemen mit PostgreSQL Datenbanken oder über CSV-Dateien möglich.

Die Anforderung N30 fordert die korrekte Berechnung von Metriken auf Grundlage der hinterlegten Formel. Die Testklasse `FormulaCalculation` überprüft erfolgreich die Berechnung von Formeln mittels mehrerer Testmethoden. Eine Testmethode ist die Berechnung einer in einer Metrik hinterlegten Formel. Die Anforderung N30 ist somit erfüllt.

Die Anforderung N40 ist der Kategorie Effizienz zugeordnet und fordert die Ausführbarkeit des IAM Reporting Modul auf einem aktuellen Standardclients, wenn zehn Messwerte und drei Metriken konfiguriert sind. Die Entwicklung des Prototyps des IAM Reporting Moduls hat auf einem aktuellen Standardlaptop stattgefunden. Dieser besitzt einen Intel Core i5 Prozessor mit acht logischen Kernen, 16 GB Arbeitsspeicher, eine 256 GB Non-Volatile Memory Express Solid State Disk sowie einen integrierten Intel Grafikprozessor. Die Ausführung des Prototyps mit den Musterdaten ist auf diesem Standardlaptop möglich.

Ebenso wurde die Benutzbarkeit anhand der Anforderungen N50 und N51 auf diesem Standardlaptop geprüft. Um die Zeiten zu messen wurden die Entwicklerungstools des Webbrowsers Microsoft Edge verwendet. Im Tab „Netzwerk“ wurde der Cache deaktiviert und vor jeder Messung alle Netzwerkaktivitäten aus dem Protokoll gelöscht. Das Messergebnis wurde nach der Durchführung einer Aktion aus der Summe aller Zeiten der Netzwerkaktivitäten gebildet. Jede Messung wurde zehnmal durchgeführt.

Gemessen wurde die Reaktionszeit (N50) von drei unterschiedlichen Aktionen:

1. Zeitraum ab dem Klick auf den Menüpunkt „Metrics“ in der Navigation startend von der Seite „Dashboard“ bis zum vollständigen Laden der Übersicht der Metriken.

2. Zeitraum ab dem Klick auf den Button „Test formula“ im Editiermodus der Metrik „Leaked passwords“ bis zum Erscheinen des Ergebnisses der Testberechnung.

3. Zeitraum ab dem Speichern der Konfiguration der Metrik „Identical passwords“ bis zum vollständigen Laden der Übersicht der Metriken und bis zum Erscheinen der Benachrichtigung des erfolgreichen Speicherns.

Im ersten Szenario lag das arithmetische Mittel bei 40,6 ms, im zweiten Szenario bei 65,3 ms und im dritten Szenario bei 400,9 ms. Die Anforderung N50 ist somit erfüllt.

Die Ladezeit eines Reports mit drei Metriken (N51) wurde anhand des Informationsbedarfs „Business facilitation“ gemessen. Dieser umfasst in den Musterdaten drei Metriken. Gemessen wurde der Zeitraum ab der Auswahl des Informationsbedarfs „Business facilitation“ bis zum vollständigen Anzeigen aller Metriken des Reports. Das arithmetische Mittel aller Messungen beträgt 48,4 ms. Damit ist die Anforderung N51 erfüllt. Alle Messergebnisse sind in der Tab. C.1 im Anhang C eingetragen.

Die weiteren Anforderungen N52-N34 bezüglich der Benutzbarkeit sind aufgrund des Einsatzes des Frontendframeworks Vaadin Flow erfüllt: In Vaadin Flow werden von Vaadin zur Verfügung gestellte Komponenten eingesetzt, die sowohl modern (N52) als auch responsive (N53) sind [Vaa23b]. Weiter kommt die Klasse `BeanValidationBinder` zum Einsatz, die Benutzereingaben überprüft und den Benutzer auf fehlerhafte Eingaben hinweist (N54).

Bezüglich der Zuverlässigkeit des IAM Reporting Moduls wird nur eine der beiden Anforderungen erfüllt. Systemfehler werden durch Einsatz von Java-Exceptions behandelt (N60), geloggt und je nach Fehler dem Benutzer direkt mitgeteilt. Falls dennoch ein Absturz nicht verhindert werden kann oder es zu einem Datenverlust kommt, wurden

keine Routinen entworfen oder implementiert, um verloren gegangene Messwerte erneut zu importieren bzw. Metriken erneut zu berechnen (N61).

Die Übertragbarkeit gemäß der Anforderung N70 auf andere auf Metriken basierende Reporting-Anwendungsfälle außerhalb des IAM, ist durch das Datenmodell (vgl. Abschnitt 5.2) gegeben, da dieses generisch gehalten und nicht – beispielsweise durch Verankerung der konkreten Ziele des IAMs im Datenmodell – auf den IAM-Kontext zugeschnitten wurde.

Die Anforderung N80 an die Sicherheit ist erfüllt: Das IAM Reporting Modul zeigt dem Benutzer nur Metriken an, welche der Zielgruppe des Benutzers entsprechen (vgl. Abschnitt 6.3.8).

Neben der Erfüllung von nicht-funktionalen Anforderungen zeigen auch die Rahmenbedingungen die Qualität des Designs bzw. des implementierten Prototyps auf. Ebenso wie bei den nicht-funktionalen Anforderungen können die Rahmenbedingungen nicht durch Überprüfung der beschriebenen Funktionalität belegt werden. Die Erfüllung der Rahmenbedingung ist durch diese Arbeit oder den Quellcode zu belegen.

In dieser Arbeit wurden die Anforderungen an das IAM Reporting Modul in Abschnitt 4.6 vollständig formuliert (R01), ein Konzept des IAM Reporting Moduls entworfen und in Kapitel 5 beschrieben (R03) sowie prototypisch implementiert und die Implementierung in Kapitel 6 beschrieben (R04).

Wie bereits in bei der Evaluation der nicht-funktionalen Anforderung N70 beschrieben, ist das Datenmodell generisch gehalten. Somit ist die Rahmenbedingung R02 erfüllt. Die Erfüllung der nicht-funktionalen Anforderung N70 und der Rahmenbedingung R02 zeigt außerdem, dass das entworfene Konzept auch auf ähnliche Anwendungsfälle zum Reporting von Metriken außerhalb des IAM-Kontextes übertragbar ist.

Die Rahmenbedingung R05 ist erfüllt, da das IAM Reporting Modul als eine Webanwendung entworfen und implementiert ist. Der Zugriff auf die Webanwendung wurde mit dem Webbrowser Microsoft Edge erfolgreich getestet.

Wie im Abschnitt 6.1 beschrieben, sind alle eingesetzten Softwares und Softwarebibliotheken Open Source. Damit ist die Rahmenbedingung R06 erfüllt. Der Tab. 6.1 ist jedoch zu entnehmen, dass die Versionen der eingesetzten Software und Softwarebibliotheken nicht immer dem neusten LTS Release entsprechen, da z. B. die neusten LTS Releases von Spring Boot, Apache Tomcat und Hibernate nicht mit dem aktuell neusten LTS Release von Vaadin kompatibel sind. Die Rahmenbedingung R07 ist somit nicht erfüllt.

Die letzte Rahmenbedingung R08 fordert die Konformität zu den geltenden Datenschutzgesetzen. Der Punkt Datenschutz wurde bei dem Design und der Implementierung nicht explizit betrachtet. Das IAM Reporting Modul erhebt beispielsweise zur Benutzerverwaltung den Vor- und Nachnamen des Benutzers sowie einen Benutzernamen. Löschroutinen dieser personenbezogenen Daten wurden jedoch nicht entworfen und implementiert. Weitere für den Datenschutz relevante Daten könnten möglicherweise in

den ermittelten Messwerten oder Metriken enthalten sein. Die Rahmenbedingung R08 ist somit nicht erfüllt.

Zusammengefasst zeigt der implementierte Prototyp, dass es auf Grundlage der Anforderungen und des Designs möglich ist, den aktuellen Zustand von IAM in Organisationen mithilfe von Metriken zu berichten.

Kapitel 8

Diskussion

Um den aktuellen Zustand von IAM zu berichten, haben Hummer et al. [HGK+18] vier relevante IAM-Ziele als ersten Schritt zu einem ganzheitlichen Framework zur Messung von IAM präsentiert, auf deren Basis ein Tool zur nachhaltigen Messung und Pflege von IAM entwickelt werden kann. In der vorliegenden Arbeit wurden die Anforderungen an ein solches Tool – namens IAM Reporting Modul – definiert sowie ein Konzept für dieses entworfen und prototypisch implementiert.

Zuerst wurde im Rahmen einer Anforderungsanalyse die Zielsetzung des IAM Reporting Moduls definiert, das System in seinen Kontext eingeordnet, die relevanten Stakeholder identifiziert, die Use Cases beschrieben, das Domänenmodell skizziert und daraus abgeleitet die funktionalen und nicht-funktionalen Anforderungen sowie Rahmenbedingungen formuliert. Auf Basis der Anforderungen wurden die Architektur, das Datenmodell, die Prozesse und die Benutzeroberflächen des IAM Reporting Modul entworfen. Die prototypische Implementierung des IAM Reporting Moduls mit anschließender Evaluation des Prototyps anhand der definierten Anforderungen verifiziert erfolgreich das entworfene Design des IAM Reporting Moduls.

Auch wenn fünf der 58 definierten Anforderungen nicht durch den Prototyp des IAM Reporting Moduls erfüllt sind, zeigt dieser erfolgreich, wie mithilfe eines Tools IAM-Metriken zur Ermittlung des Zustands von IAM in Organisationen berichtet werden können: Im IAM Reporting Modul stellt das Dashboard einen Report mit den aktuellsten berechneten Metrikwerten zu dem durch den Benutzer ausgewählten Informationsbedarf dar. Aufgrund des modernen Softwarestacks basierend auf Java und PostgreSQL und des erweiterbaren Datenmodells ist es möglich, das IAM Reporting Modul an die unterschiedlichen und meist heterogenen Systemlandschaften verschiedener Organisationen anzupassen und darin einzusetzen. Das Deployment als Webanwendung mit Authentifizierung und Autorisierung ermöglicht einen zentralen Betrieb des IAM Reporting Moduls in einer Organisation und ohne der Notwendigkeit der Installation von zusätzlicher Software – bis auf einen zeitgemäßen Webbrowser – auf den Geräten der Benutzer.

Neben dem praktischen Nutzen des IAM Reporting Moduls in Organisationen, wurden die folgenden vier Gestaltungsprinzipien für Reporting Module basierend auf Metriken aus der vorliegenden Arbeit abgeleitet:

Regelmäßige Berechnung und Ermittlung: Um die Zielerreichung in einer Organisation mithilfe eines Reporting Moduls basierend auf Metriken zu berichten, sind die Metriken regelmäßig aus Messwerten und anderen Metriken anhand einer Formel zu berechnen sowie die Messwerte regelmäßig zu ermitteln (vgl. Abschnitt 6.3.7). Denn um den Fortschritt der Zielerreichung vergleichen zu können, müssen die Werte der Metriken und Messwerte in gleichbleibenden, in den für das Ziel erforderlichen Zeitabständen vorliegen [CSS+08; Int16b]. Beispielsweise kann der langfristige Effekt einer implementierten Maßnahme nur sinnvoll evaluiert werden, wenn eine für diese passende Metrik sowohl vor als auch nach der Umsetzung der Maßnahme regelmäßig ermittelt wurde.

Prozess zur Konfiguration: Um im Reporting Modul die Konfiguration vorzunehmen, sind bestimmte Konfiguration vor anderen durchzuführen. Dabei unterscheidet sich die Reihenfolge der durchzuführenden Schritte zur Konfiguration im Reporting Modul (vgl. Abschnitt 5.3) von der Reihenfolge der Prozessschritte im fachlichen Prozess (vgl. Abschnitt 2.2). Sowohl im Reporting Modul als auch im fachlichen Prozess sind zuerst die grundlegenden Konfigurationen der Stakeholder, der Informationsbedarfe und der Zielgruppen festzulegen. Danach sind im fachlichen Prozess als Erstes die Metriken zu definieren, bevor Messwerte und Datenquellen ausgewählt werden. Konträr dazu folgt im Reporting Modul zuerst die Konfiguration der Datenquellen, daraufhin die der Messwerte und abschließend die der Metriken. Das liegt daran, dass die Konfigurationen der Metriken und Messwerte im Datenmodell eine 1:N Beziehung zu anderen Konfigurationen haben (vgl. Kapitel 5.2). Beispielsweise muss vor der Konfiguration eines Messwerts die Datenquelle konfiguriert werden (siehe Abb. 5.7). Ein weiteres Beispiel ist die Konfiguration einer Metrik: vor dieser müssen die in der Formel der Metrik referenzierten Messwerte oder Metriken konfiguriert werden. (siehe Abb. 5.4).

Zusatzinformationen über den Messwert je nach Typ der Datenquelle: Um in einem Reporting Modul Messwerte aus Datenquellen mit einer Vielzahl an Daten zu ermitteln, kann die Konfiguration von zusätzlichen Informationen über den Messwert je nach Typ der Datenquelle erforderlich sein (vgl. Datenmodell in Kapitel 5.2 und Abb. 5.7). Denn sofern in der Datenquelle nicht nur ein einzelnes Datum vorliegt, ist weiter zu spezifizieren, welches Datum in der Datenquelle gemessen werde soll. Bei Datenbanken ist z. B. eine Abfrage, die den gewünschten Messwert zurückgibt, zu definieren. Bei Dateien hingegen sind Angaben zu machen, wo sich der Messwert in der Datei befindet.

Datenmodell: Um die für ein Reporting Modul erforderlichen Daten zu strukturieren, ist das in Kapitel 5.2 entworfene Datenmodell einzusetzen. Dieses ermöglicht die Trennung der Konfiguration von den ermittelten oder berechneten Ergebnissen, um die Historie der Ergebnisse beizubehalten (siehe Abb. 5.6). Weiter kann in der Formel auf Messwerte und Metriken verwiesen werden, um eine Metrik sowohl aus Messwerten und als auch aus anderen Metriken berechnen zu können (siehe Abb. 5.4). Zuletzt ist das entworfene Datenmodell um weitere Typen von Datenquellen erweiterbar, um bis dato nicht berücksichtigte Typen von Datenquellen hinzuzufügen (siehe Abb. 5.7).

Folgende Limitierungen der Arbeit sind festzuhalten: Diese Arbeit behandelt aus-

schließlich die Definition der Anforderungen, des Designs und eine prototypische Implementierung des IAM Reporting Moduls. Deswegen kann keine Aussage darüber getroffen werden, wie sich das IAM Reporting Modul im produktiven Einsatz über einen längeren Zeitraum mit einer Vielzahl an Datenquellen, Messwerten und Metriken verhält. Weiter ist zu berücksichtigen, dass sich diese Arbeit nicht mit dem Entwerfen von Visualisierungen von Metriken mithilfe von Diagrammen und anderen Visualisierungsmöglichkeiten sowie dem Design der Benutzeroberfläche nach Aspekten des User Experience Designs (UXD) beschäftigt hat. Ebenfalls außerhalb des Rahmens der vorliegenden Arbeit ist die Konformität mit dem Datenschutz. Der entworfene und implementierte Prototyp des IAM Reporting Moduls erfasst, wie in der Evaluation dargelegt, wenige für den Datenschutz relevante Daten. Eine explizite Überprüfung des Datenschutzes ist ausstehend.

Insgesamt hat diese Arbeit die Grundlagen eines IAM Reporting Moduls basierend auf Metriken geliefert. Zukünftige Forschungen können auf der geschaffenen Grundlage aufsetzen, um beispielsweise das IAM Reporting Modul in einer zu produktiven Systemlandschaften ähnlichen Umgebung zu evaluieren. Zudem kann die Erweiterbarkeit des Datenmodells durch weitere Typen von Datenquellen wie z. B. Not only Structured Query Language (SQL)-(NoSQL)-Datenbanken und Webservices überprüft werden. Die Konformität mit Datenschutz und auch die Datensicherheit des IAM Reporting Moduls können weitere Aspekte zukünftiger Forschungen sein. Im Allgemeinen stehen Organisationen im Bereich der IT-Sicherheit oft vor der Fragestellung, ob die getroffenen Maßnahmen die gewünschten Wirkungen zeigen. Deshalb kann es für zukünftige Forschungen interessant sein, wie das IAM Reporting Modul und die Gestaltungsprinzipien auf andere Use Cases im Bereich der IT-Sicherheit angewandt werden können, um die Zielerreichung mithilfe von Metriken zu ermitteln. Der Schwerpunkt dieser Arbeit lag nicht auf dem Design der Benutzeroberflächen. Diese können in weiteren Iterationen unter Zuhilfenahme von UXD-Methoden optimiert werden: Zum einen können die Formulare zur Konfiguration beispielsweise durch visuelle Editoren für die Formel und die Frequenz erweitert und die Anordnung der Felder überprüft werden. Zum anderen kann der bislang schlicht gehaltene Report im Dashboard durch erweiterte Visualisierungen der Metriken unter Zuhilfenahme von Diagrammen und Indikatoren für Trends und die Erreichung von Zielen, durch beispielsweise Farben oder Pfeile, verbessert werden.

Kapitel 9

Fazit

In dieser Arbeit wurde untersucht, welche Anforderungen an ein IAM Reporting Modul basierend auf Metriken bestehen, wie das IAM Reporting Modul zu konzipieren und zu implementieren ist und welche Gestaltungsprinzipien sich daraus ableiten lassen. Unter Einbezug bestehender Veröffentlichungen wurden die Anforderungen definiert und mittels der Anwendung von Softwareentwicklungsmethoden ein Konzept für ein IAM Reporting Modul basierend auf Metriken erstellt. Im weiteren Verlauf wurde das IAM Reporting Modul als Prototyp implementiert und anhand der Anforderungen evaluiert.

Die Evaluation belegt, dass mithilfe des IAM Reporting Moduls Metriken mit Messwerten aus unterschiedlichen Quellen erfolgreich ermittelt, berechnet und berichtet werden können. Im Verlauf der Arbeit wurden alle drei Forschungsfragen beantwortet: Zunächst wurden die funktionalen und nicht-funktionalen Anforderungen an ein IAM Reporting Modul aus wissenschaftlichen Veröffentlichungen abgeleitet und definiert (RQ1). Die daraufhin erfolgte Konzeption und Implementierung des Prototyps zeigt auf, wie mithilfe des IAM Reporting Moduls der aktuelle Zustand von IAM in Organisationen berichtet werden kann (RQ2). Abschließend wurden vier Gestaltungsprinzipien für Reporting Module aus den Ergebnissen abgeleitet (RQ3): (1) Als wesentliche Anforderung hat sich herausgestellt, die Metriken und Messwerte in regelmäßigen Zeitabständen zu berechnen bzw. zu ermitteln, um die Performance vergleichen zu können. (2) Weiter wurden Differenzen zwischen dem fachlichen Prozess zur Konfiguration und der Reihenfolge der zur Konfiguration notwendigen Schritte im Reporting Modul festgestellt. Während sich die Reihenfolge der ersten Schritte zur Konfiguration von den Stakeholdern, den Informationsbedarfen und den Zielgruppen gleichen sind die darauffolgenden Schritte konträr. Im fachlichen Prozess werden zunächst die Metriken festgelegt und daraufhin werden die Messwerte und die Datenquellen ausgewählt. Im Reporting Modul ist diese Reihenfolge umgekehrt. (3) Da die Messwerte in technologisch unterschiedlichen Datenquellen ermittelt werden, sind im Messwert zusätzliche Informationen je nach Typ der Datenquelle zu erfassen. In den Zusatzinformationen ist vermerkt, wie der Messwert in der jeweiligen Datenquelle ermittelt wird. (4) Des Weiteren haben die Implementierung und die Evaluation des IAM Reporting Moduls gezeigt, dass das konzipierte Datenmodell erfolgreich für Reporting Module eingesetzt werden kann.

Diese Arbeit ist ein weiterer Beitrag dazu, Maßnahmen der IT-Sicherheit messbar zu machen. Die Ergebnisse dieser Arbeit knüpfen an die bisherigen Veröffentlichungen zur Messung der Performance im Gebiet der IT-Sicherheit und Zielen im IAM an und zeigen wie die Performance und Ziele mithilfe eines Tools basierend auf Metriken gemessen und berichtet werden können. Aus den Ergebnissen ergeben sich Möglichkeiten für zukünftige Arbeiten: Die Evaluation des IAM Reporting Moduls in einer produktiven Umgebung, das Hinzufügen weiterer Typen von Datenquellen, der Einsatz des Reporting Moduls in weiteren Teilgebieten der IT-Sicherheit und die Konzeption erweiterter Visualisierungen der Metriken im Report.

Anhang

Anhang A

Datenmodell

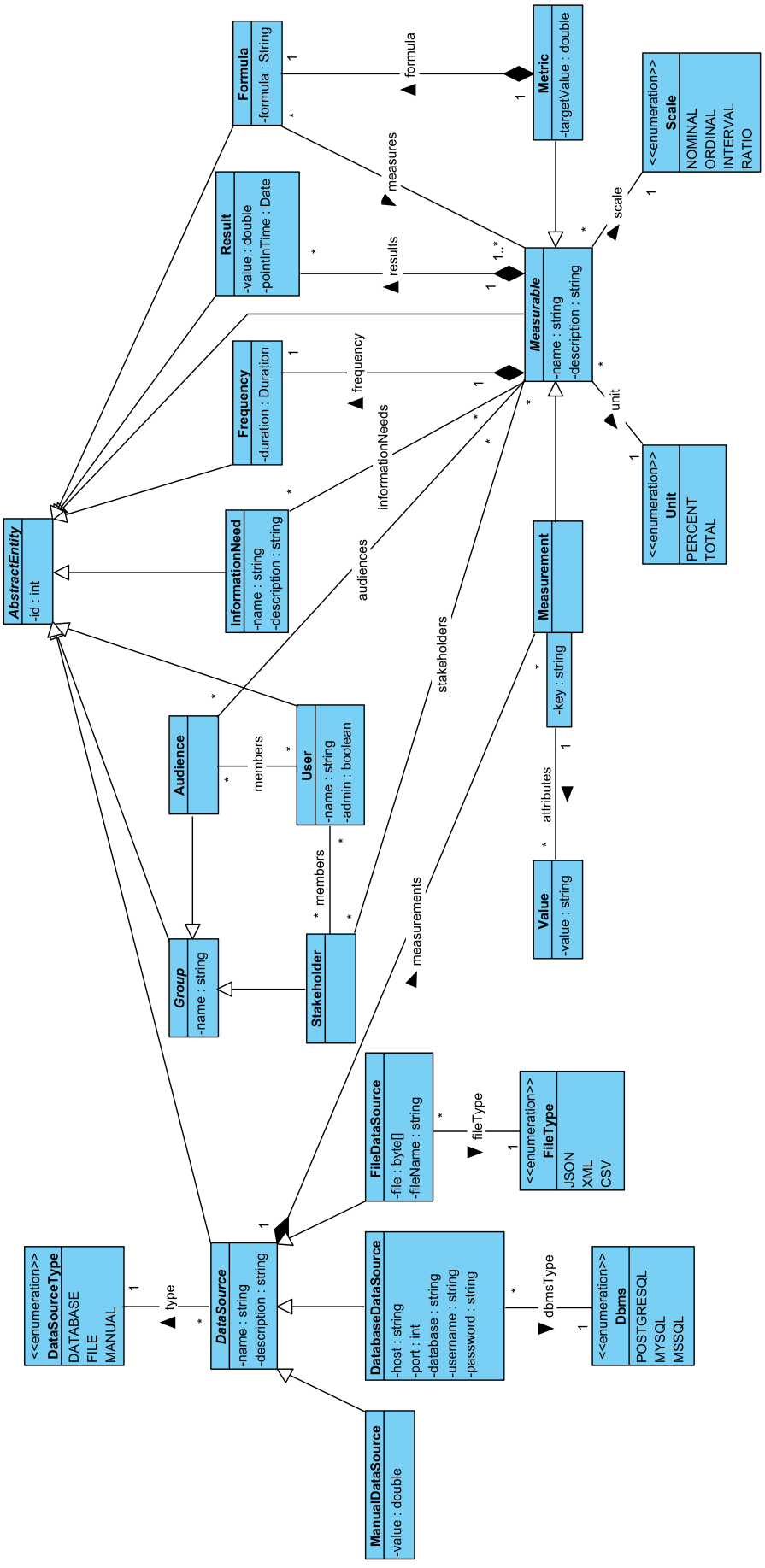

Abb. A.1 Klassendiagramm

Anhang B

Mockups

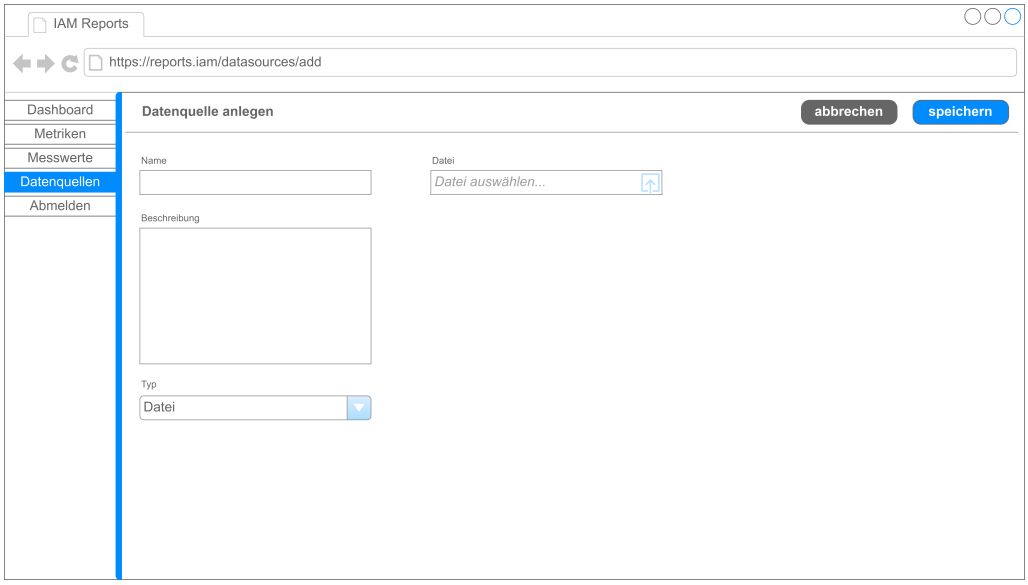

Abb. B.1 Mockup: Anlegen einer Datei als Datenquelle

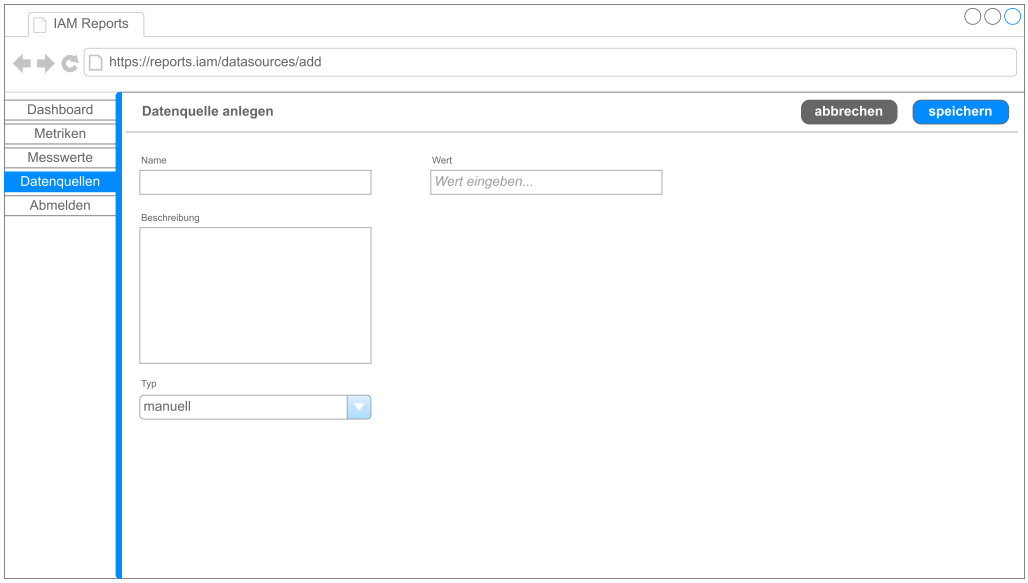

Abb. B.2 Mockup: Anlegen einer manuellen Datenquelle

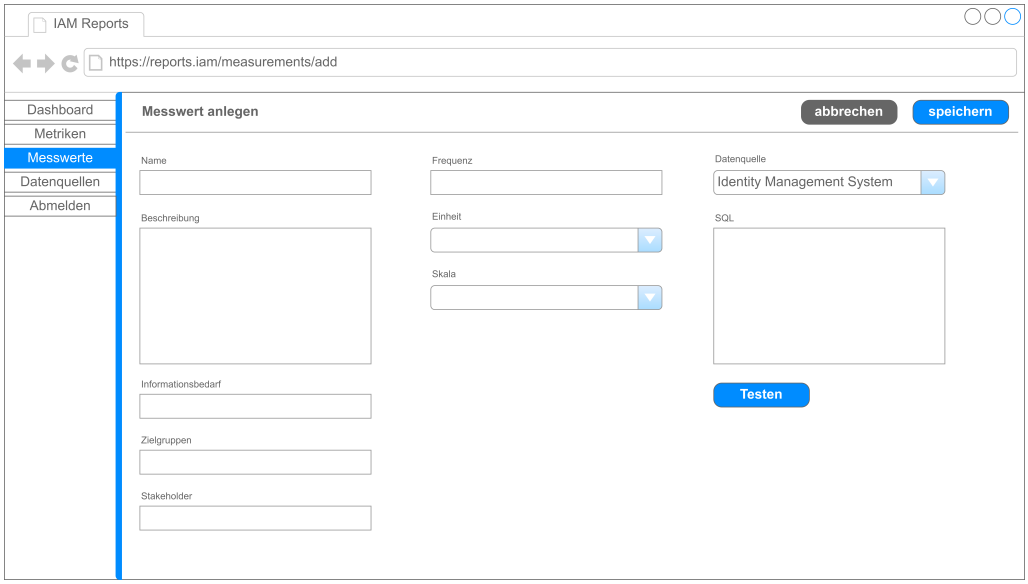

Abb. B.3 Mockup: Anlegen eines Messwerts

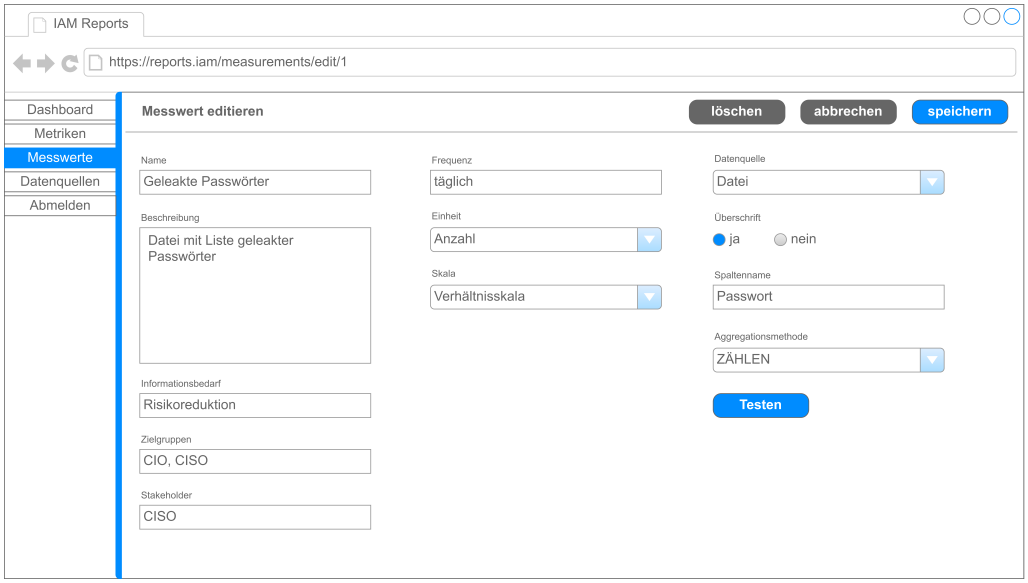

Abb. B.4 Mockup: Editieren eines Messwerts mit einer Datei als Datenquelle

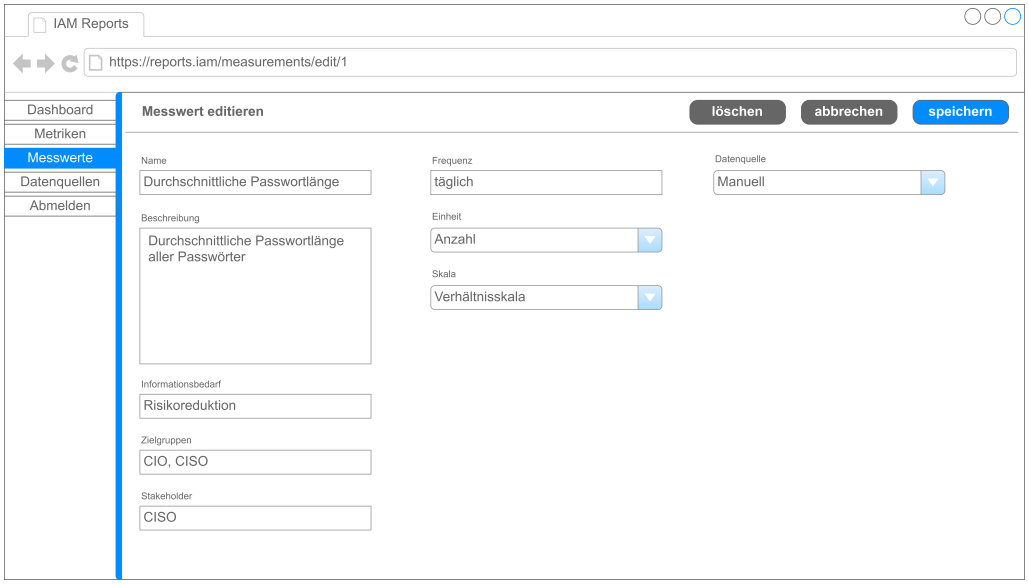

Abb. B.5 Mockup: Editieren eines Messwerts mit einer manuellen Datenquelle

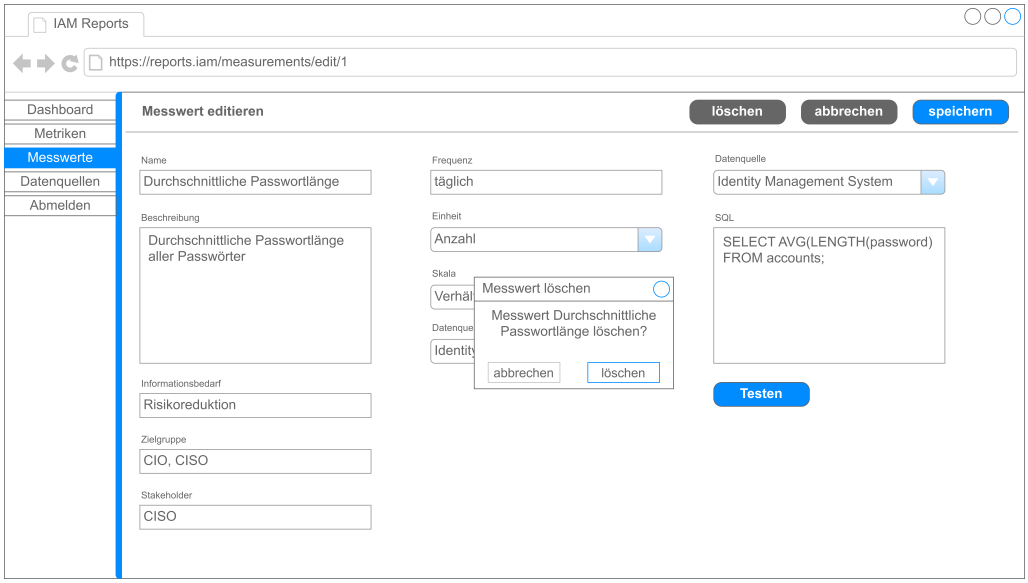

Abb. B.6 Mockup: Messwert löschen

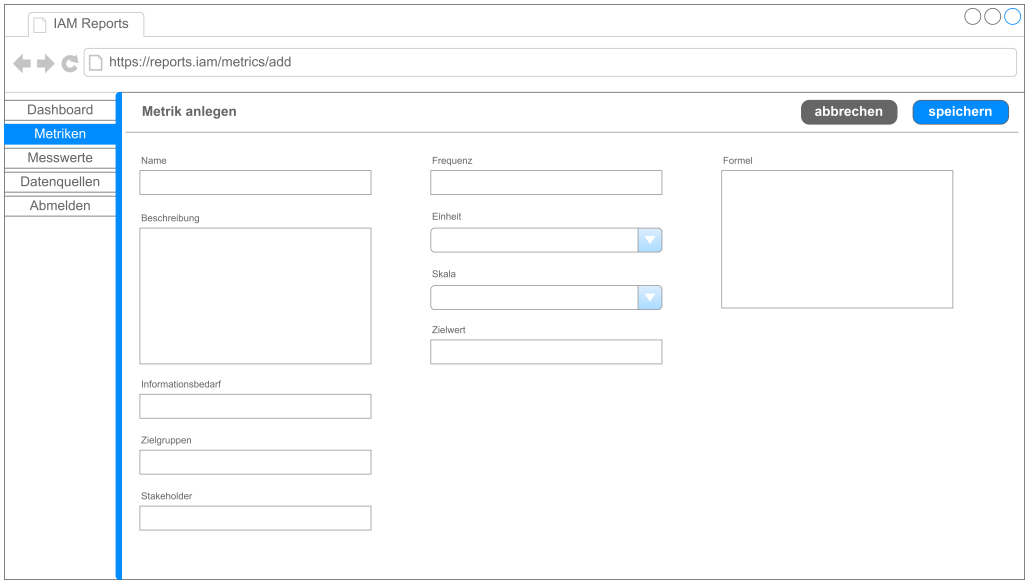

Abb. B.7 Mockup: Metrik anlegen

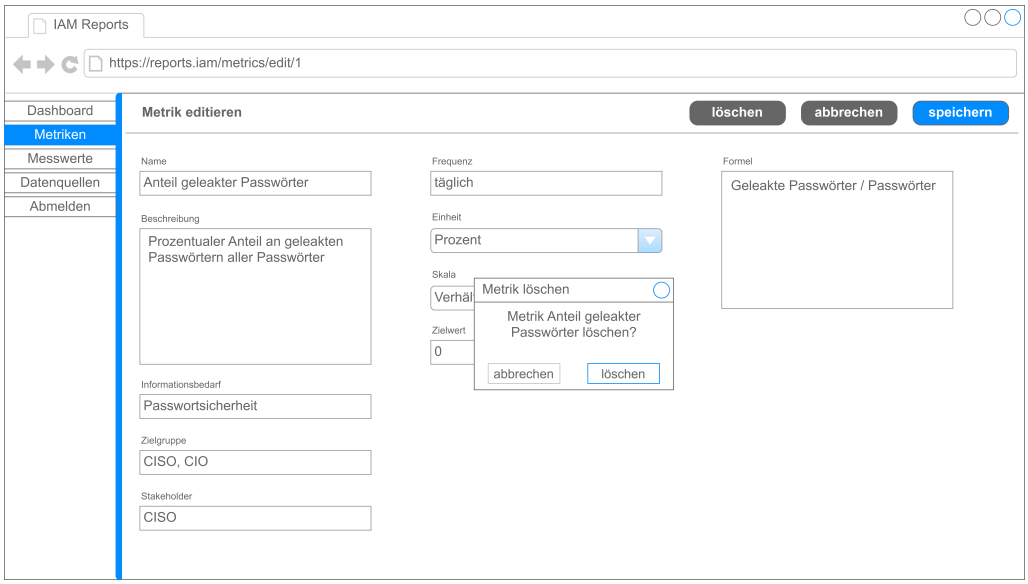

Abb. B.8 Mockup: Metrik löschen

Anhang C

Messergebnisse

Szenario	N50 (1. Szenario)	N50 (2. Szenario)	N50 (3. Szenario)	N51
Messung 1	44	55	439	44
Messung 2	34	67	413	45
Messung 3	42	54	410	48
Messung 4	40	56	376	59
Messung 5	34	72	377	55
Messung 6	48	54	433	45
Messung 7	43	73	395	46
Messung 8	42	85	361	45
Messung 9	41	53	372	41
Messung 10	38	84	433	56
arithmetische Mittel	40,6	65,3	400,9	48,4
Median	41,5	61,5	402,5	45,5
Standardabweichung	4,351245033	12,66710526	28,66445348	6,040603355

Tab. C.1 Messergebnisse in ms

Anhang D

Digitaler Anhang

PDF-Version dieser Arbeit
Dateipfad: Masterarbeit.pdf

Quellcode des Prototyps
Dateipfad: Quellcode/

Literaturverzeichnis

[Ahn09] AHN, Gail-Joon: Discretionary Access Control. In: LUI, Ling (Hrsg.) ; ÖZSU, M. T. (Hrsg.): Encyclopedia of Database Systems. Springer US, 2009, S. 864–866

[Ass17] ASSEG, Frank: exp4j. https://www.objecthunter.net/exp4j/. Version: 2017. – Abruf am: 2023-02-05

[Bal09] BALZERT, Helmut: Anforderungen und Anforderungsarten. In: Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering. Heidelberg : Spektrum Akademischer Verlag, 2009, S. 455–474

[Bal11] BALZERT, Helmut: Der Entwurfsprozess. In: Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb. Heidelberg : Spektrum Akademischer Verlag, 2011, S. 481–485

[Bas22] BASEL COMMITTEE ON BANKING SUPERVISION: The Basel Framework. 2022 https://www.bis.org/basel_framework

[BBLZ96] BÄUMER, Dirk ; BISCHOFBERGER, Walter R. ; LICHTER, Horst ; ZÜLLIGHOVEN, Heinz: User Interface Prototyping—Concepts, Tools, and Experience. In: Proceedings of the 18th International Conference on Software Engineering. USA : IEEE Computer Society, 1996 (ICSE '96), S. 532–541

[BHM20] Kapitel Introduction to Design Science Research. In: BROCKE, Jan ; HEVNER, Alan R. ; MAEDCHE, Alexander: Introduction to Design Science Research. Cham : Springer International Publishing, 2020, S. 1–13

[Bit22] BITKOM: Ausgaben für IT-Sicherheit in Deutschland in den Jahren 2017 bis 2021 und Prognose bis 2025 (in Milliarden Euro) [Graph]. Zitiert nach de.statista.com. https://de.statista.com/statistik/daten/studie/1041736/umfrage/ausgaben-fuer-it-security-in-deutschland/. Version: Oktober 2022. – Abruf am: 2023-03-24

[BKY11] BARABANOV, Rostyslav ; KOWALSKI, Stewart ; YNGSTRÖM, Louise: Information security metrics: State of the Art. In: DSV Report series 2011 (2011)

[Blo74] BLOHM, Hans: Die Gestaltung des betrieblichen Berichtswesens als Problem der Leitungsorganisation: Organisation, Verwaltung und Arbeitswissenschaft. Bd. 2. Berlin : Herne, 1974

[Blo75] BLOHM, Hans: Organisation des Informationswesen. In: GROCHLA, Erwin (Hrsg.) ; WITTMANN, Waldemar (Hrsg.): Handwörterbuch der Betriebswirtschaft Bd. 2. 4. Stuttgart : Poeschel, 1975, S. 1924–1930

[BS21] BERG, Achim ; SELEN, Sinan: Wirtschaftsschutz 2021. https://www.bitkom.org/sites/default/files/2021-08/bitkom-slides-wirtschaftsschutz-cybercrime-05-08-2021.pdf. Version: August 2021. – Abruf am: 2023-03-24

[BSS09] BLACK, Paul ; SCARFONE, Karen ; SOUPPAYA, Murugiah: Cyber Security Metrics and Measures. Hoboken, NJ : John Wiley & Sons, Inc., 2009

[Bun22] BUNDESAMT FÜR SICHERHEIT IN DER INFORMATIONSTECHNIK: ORP.4: Identitäts- und Berechtigungsmanagement. In: IT-Grundschutz-Kompendium. Reguvis Fachmedien GmbH, Februar 2022, S. 1–8

[CG16] CHAMONI, Peter ; GLUCHOWSKI, Peter: Analytische Informationssysteme – Einordnung und Überblick. In: GLUCHOWSKI, Peter (Hrsg.) ; CHAMONI, Peter (Hrsg.): Analytische Informationssysteme: Business Intelligence-Technologien und -Anwendungen. Berlin, Heidelberg : Springer Berlin Heidelberg, 2016, Kapitel 1, S. 3–12

[CG22] CAMERON, Andrew ; GREWE, Olaf: An Overview of the Digital Identity Lifecycle (v2). In: IDPro Body of Knowledge 1 (2022). https://bok.idpro.org/article/31/galley/121/view/

[Cla94] CLARKE, Roger: Human identification in information systems: Management challenges and public policy issues. In: Information Technology & People 7 (1994), Nr. 4, S. 6–37

[Cla10] CLARKE, Roger: A Sufficiently Rich Model of (Id)entity, Authentication and Authorisation. http://www.rogerclarke.com/ID/IdModel-1002.html. Version: Februar 2010. – Abruf am: 2023-02-14

[Cor22] COREOZ: Wisp Scheduler. https://github.com/Coreoz/Wisp. Version: 2022. – Abruf am: 2023-02-05

[CSS+08] CHEW, Elizabeth ; SWANSON, Marianne. ; STINE, Kevin M. ; BARTOL, Nadya. ; BROWN, Anthony. ; ROBINSON, Will.: Performance measurement guide for information security. Gaithersburg, MD : National Institute of Standards and Technology, Juli 2008 (NIST SP 800-55r1). – Forschungsbericht

[Den76] DENNING, Dorothy E.: A Lattice Model of Secure Information Flow. In: Commun. ACM 19 (1976), Mai, S. 236–243

[DLSA15] DEBATTISTA, Jeremy ; LANGE, Christoph ; SCERRI, Simon ; AUER, Sören: Linked 'Big' Data: Towards a Manifold Increase in Big Data Value and Veracity. In: 2015 IEEE/ACM 2nd International Symposium on Big Data Computing (BDC), 2015, S. 92–98

[FK92] FERRAIOLO, David ; KUHN, David: Role-Based Access Controls. In: Proceedings of the 15th National Computer Security Conference, National Institute of Standards and Technology, Oktober 1992, S. 554–563

[FP07] FUCHS, Ludwig ; PERNUL, Günther: Supporting Compliant and Secure User Handling - A Structured Approach for In-House Identity Management. In: The Second International Conference on Availability, Reliability and Security (ARES'07), 2007, S. 374–384

[Fre91] FREE SOFTWARE FOUNDATION, INC: GNU General Public License, version 2, with the Classpath Exception. https://github.com/adoptium/jdk17u/blob/master/LICENSE. Version: Juni 1991. – Abruf am: 2023-01-24

[GD71] GRAHAM, G S. ; DENNING, Peter J.: Protection: Principles and Practice. In: Proceedings of the May 16-18, 1972, Spring Joint Computer Conference, Association for Computing Machinery, 1971, S. 417–429

[GGF17] GRASSI, Paul A. ; GARCIA, Michael E. ; FENTON, James L.: Digital identity guidelines: revision 3. Gaithersburg, MD : National Institute of Standards and Technology, Juni 2017 (NIST SP 800-63-3). – Forschungsbericht

[GH13] GREGOR, Shirley ; HEVNER, Alan R.: Positioning and Presenting Design Science Research for Maximum Impact. In: MIS Q. 37 (2013), Juni, Nr. 2, S. 337–356

[GHM08] GLEICH, Ronald ; HORVÁTH, Péter ; MICHEL, Uwe: Management Reporting: Grundlagen, Praxis und Perspektiven. Haufe Verlag, 2008 (Haufe Fachpraxis). https://www.wiso-net.de/document/HAUF__31e2e457e8f2b1d2cebce8587a4d526f05b346a8

[Gli07] GLINZ, Martin: On Non-Functional Requirements. In: 15th IEEE International Requirements Engineering Conference (RE 2007), 2007, S. 21–26

[Gli22] GLINZ, Martin: A Glossary of Requirements Engineering Terminology, Version 2.0.1. International Requirements Engineering Board (IREB). Available at https://www.ireb.org/en/cpre/cpre-glossary/, 2022

[Gro23a] GROMADA, Mariusz: mXparser. https://mathparser.org/. Version: 2023. – Abruf am: 2023-02-03

[Gro23b] GROMADA, Mariusz: mXparser – LICENSE. https://mathparser.org/mxparser-license/. Version: 2023. – Abruf am: 2023-02-03

[GW07] GLINZ, Martin ; WIERINGA, Roel J.: Guest Editors' Introduction: Stakeholders in Requirements Engineering. In: IEEE Software 24 (2007), Nr. 2, S. 18–20

[Gö06] GÖPFERT, Ingrid: Berichtswesen. In: HANDELSBLATT (Hrsg.): Wirtschaftslexikon Bd. 2. Stuttgart : Poeschel, 2006, S. 692–702

[H2oJa] H2: H2 Database Engine. https://www.h2database.com/html/main.html. Version: oJ. – Abruf am: 2023-01-18

[H2oJb] H2: H2 License. https://www.h2database.com/html/license.html. Version: oJ. – Abruf am: 2023-01-18

[HC10] HEVNER, Alan R. ; CHATTERJEE, Samir: Design Science Research in Information Systems. In: Design Research in Information Systems: Theory and Practice. Boston, MA : Springer US, 2010, S. 9–22

[Hev07] HEVNER, Alan R.: A Three Cycle View of Design Science Research. 19 (2007), Nr. 2

[HFK+14] HU, Vincent C. ; FERRAIOLO, David ; KUHN, Rick ; SCHNITZER, Adam ; SANDLIN, Kenneth ; MILLER, Robert ; SCARFONE, Karen: Guide to Attribute Based Access Control (ABAC) Definition and Considerations. National Institute of Standards and Technology, Januar 2014 (NIST SP 800-162). – Forschungsbericht

[HGK+18] HUMMER, Matthias ; GROLL, Sebastian ; KUNZ, Michael ; FUCHS, Ludwig ; PERNUL, Günther: Measuring Identity and Access Management Performance - An Expert Survey on Possible Performance Indicators. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018) Bd. 2018-January, SciTePress, Januar 2018, S. 233–240

[HHK+18] HEINRICH, Bernd ; HRISTOVA, Diana ; KLIER, Mathias ; SCHILLER, Alexander ; SZUBARTOWICZ, Michael: Requirements for data quality metrics. In: ACM Journal of Data and Information Quality 9 (2018), Januar, S. 12:1–12:32

[Hib09] HIBERNATE: hibernate-validator/license.txt at main. https://github.com/hibernate/hibernate-validator/blob/main/license.txt. Version: 2009. – Abruf am: 2023-01-19

[Hib18] HIBERNATE: hibernate-orm/lgpl.txt at main. https://github.com/hibernate/hibernate-orm/blob/main/lgpl.txt. Version: 2018. – Abruf am: 2023-01-19

[HiboJa] HIBERNATE: Hibernate ORM. https://hibernate.org/orm/. Version: oJ. – Abruf am: 2023-01-19

[HiboJb] HIBERNATE: Hibernate Validator. https://hibernate.org/validator/. Version: oJ. – Abruf am: 2023-01-20

[HMPR04] HEVNER, Alan R. ; MARCH, Salvatore T. ; PARK, Jinsoo ; RAM, Sudha: Design Science in Information Systems Research. In: MIS Quarterly 28 (2004), Nr. 1, 75–105. http://www.jstor.org/stable/25148625

[Hor08] HORVÁTH, Péter: Controlling. 11. Verlag Franz Vahlen, 2008

[IAB18] INDU, I. ; ANAND, P. M. ; BHASKAR, Vidhyacharan: Identity and access management in cloud environment: Mechanisms and challenges. In: Engineering Science and Technology, an International Journal 21 (2018), Mai, S. 574–588

[Ide21] IDENTITY, CREDENTIAL, AND ACCESS MANAGEMENT SUBCOMMITTEE (ICAMSC): The Federal Idenity, Credential, and Access Management Architecture. https://playbooks.idmanagement.gov/docs/ficam-arch.pdf. Version: Januar 2021. – Abruf am: 2022-12-07

[IEE90] IEEE Standard Glossary of Software Engineering Terminology. In: IEEE Std 610.12-1990 (1990), S. 1–84

[IEE98] IEEE Recommended Practice for Software Requirements Specifications. In: IEEE Std 830-1998 (1998), S. 1–40

[IEE18] IEEE AND THE OPEN GROUP: The Open Group Base Specifications Issue 7, 2018 Edition. https://pubs.opengroup.org/miscpubs/9699919799/. Version: 2018. – Abruf am: 2023-01-24

[Int11] INTERNATIONALE ORGANISATION FÜR NORMUNG: Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. Version: März 2011. https://www.iso.org/standard/35733.html. Geneva, CH : Internationale Organisation für Normung, März 2011 (ISO/IEC/IEEE 25010:2011). – Standard

[Int13] INTERNATIONALE ORGANISATION FÜR NORMUNG: Information security, cybersecurity and privacy protection — Information security management systems — Requirements. Version: Oktober 2013. https://www.iso.org/standard/54534.html. Geneva, CH : Internationale Organisation für Normung, Oktober 2013 (ISO/IEC 27001:2013). – Standard

[Int16a] INTERNATIONALE ORGANISATION FÜR NORMUNG: Information technology — Security techniques — A framework for access management. Version: Juni 2016. https://www.iso.org/standard/45169.html. Geneva, CH : Internationale Organisation für Normung, Juni 2016 (ISO/IEC 29146:2016). – Standard

[Int16b] INTERNATIONALE ORGANISATION FÜR NORMUNG: Information technology — Security techniques — Information security management — Monitoring, measurement, analysis and evaluation. Version: Dezember 2016. https://www.iso.org/standard/64120.html. Geneva, CH : Internationale Organisation für Normung, Dezember 2016 (ISO/IEC 27004:2016). – Standard

[Int17] INTERNATIONALE ORGANISATION FÜR NORMUNG: Systems and software engineering – Measurement process. Version: Mai 2017. https://www.iso.org/standard/71197.html. Geneva, CH : Internationale Organisation für Normung, Mai 2017 (ISO/IEC/IEEE 15939:2017). – Standard

[Int18a] INTERNATIONALE ORGANISATION FÜR NORMUNG: Information technology — Security techniques — Information security management systems — Overview and vocabulary. Version: Februar 2018. https://www.iso.org/standard/73906.html. Geneva, CH : Internationale Organisation für Normung, Februar 2018 (ISO/IEC 27000:2018). – Standard

[Int18b] INTERNATIONALE ORGANISATION FÜR NORMUNG: Systems and software engineering — Life cycle management — Part 1: Guidelines for life cycle management. Version: November 2018. https://www.iso.org/standard/72896.html. Geneva, CH : Internationale Organisation für Normung, November 2018 (ISO/IEC/IEEE 24748-1:2018). – Standard

[Int19a] INTERNATIONALE ORGANISATION FÜR NORMUNG: Date and time — Representations for information interchange — Part 1: Basic rules. Version: Februar 2019. https://www.iso.org/standard/70907.html. Geneva, CH : Internationale Organisation für Normung, Februar 2019 (ISO 8601-1:2019). – Standard

[Int19b] INTERNATIONALE ORGANISATION FÜR NORMUNG: IT Security and Privacy — A framework for identity management — Part 1: Terminology and concepts. Version: Mai 2019. https://www.iso.org/standard/77582.html. Geneva, CH : Internationale Organisation für Normung, Mai 2019 (ISO/IEC 24760-1:2019). – Standard

[JavoJ] Javaluator. https://javaluator.sourceforge.net/en/home/. Version: oJ. – Abruf am: 2023-02-05

[JoboJa] JOBRUNR: JobRunr. https://www.jobrunr.io/en/. Version: oJ. – Abruf am: 2023-02-05

[JoboJb] JOBRUNR: JobRunr Pro. https://www.jobrunr.io/en/pricing/. Version: oJ. – Abruf am: 2023-02-05

[Kar22] KARLSSON, Gustav: db-scheduler. https://github.com/kagkarlsson/db-scheduler. Version: 2022. – Abruf am: 2023-02-05

[KBM10] KEMPER, Hans-Georg ; BAARS, Henning ; MEHANNA, Walid: Informationsgenerierung, -speicherung, -distribution und -zugriff. In: Business Intelligence – Grundlagen und praktische Anwendungen: Eine Einführung in die IT-basierte Managementunterstützung. Wiesbaden : Vieweg+Teubner, 2010, Kapitel 3, S. 85–161

[Kle18] KLEUKER, Stephan: Anforderungsanalyse. In: Grundkurs Software-Engineering mit UML: Der pragmatische Weg zu erfolgreichen Softwareprojekten. Wiesbaden : Springer Fachmedien Wiesbaden, 2018, S. 55–92

[Kli22] KLIMASCHEWSKI, Udo: EvalEx - Java Expression Evaluator. https://github.com/ezylang/EvalEx. Version: 2022. – Abruf am: 2023-02-05

[Koc94] KOCH, Rembert: Betriebliches Berichtswesen als informations- und steuerungsinstrument. Lang, 1994

[Lam74] LAMPSON, Butler W.: Protection. In: SIGOPS Oper. Syst. Rev. 8 (1974), Januar, S. 18–24

[Mvn23a] MVNREPOSITORY: DB Scheduler: Core » 11.6. https://mvnrepository.com/artifact/com.github.kagkarlsson/db-scheduler/11.6. Version: 2023. – Abruf am: 2023-02-05

[Mvn23b] MVNREPOSITORY: EvalEx » 3.0.1. https://mvnrepository.com/artifact/com.ezylang/EvalEx/3.0.1. Version: 2023. – Abruf am: 2023-02-05

[Mvn23c] MVNREPOSITORY: Exp4j » 0.4.8. https://mvnrepository.com/artifact/net.objecthunter/exp4j/0.4.8. Version: 2023. – Abruf am: 2023-02-05

[Mvn23d] MVNREPOSITORY: Javulator » 3.0.3. https://mvnrepository.com/artifact/com.fathzer/javaluator/3.0.3. Version: 2023. – Abruf am: 2023-02-05

[Mvn23e] MVNREPOSITORY: JobRunr » 6.0.0. https://mvnrepository.com/artifact/org.jobrunr/jobrunr/6.0.0. Version: 2023. – Abruf am: 2023-02-05

[Mvn23f] MVNREPOSITORY: MathParser Org MXparser » 5.2.0. https://mvnrepository.com/artifact/org.mariuszgromada.math/MathParser.org-mXparser/5.2.0. Version: 2023. – Abruf am: 2023-02-05

[Mvn23g] MVNREPOSITORY: Quartz » 2.3.2. https://mvnrepository.com/artifact/org.quartz-scheduler/quartz/2.3.2. Version: 2023. – Abruf am: 2023-02-05

[Mvn23h] MVNREPOSITORY: Wisp Scheduler » 2.3.0. https://mvnrepository.com/artifact/com.coreoz/wisp/2.3.0. Version: 2023. – Abruf am: 2023-02-05

[MZN10] MAIRIZA, Dewi ; ZOWGHI, Didar ; NURMULIANI, Nurie: An Investigation into the Notion of Non-Functional Requirements. In: Proceedings of the 2010 ACM Symposium on Applied Computing. New York, NY, USA : Association for Computing Machinery, 2010 (SAC '10), S. 311–317

[NO96] NYANCHAMA, Matunda ; OSBORN, Sylvia: Modeling Mandatory Access Control in Role-Based Security Systems. In: SPOONER, David L. (Hrsg.) ; DEMURJIAN, Steven A. (Hrsg.) ; DOBSON, John E. (Hrsg.): Database Security IX: Status and prospects. Boston, MA : Springer US, 1996, S. 129–144

[Ope22] OPENJS FOUNDATION: node/LICENSE at main. https://github.com/nodejs/node/blob/main/LICENSE. Version: 2022. – Abruf am: 2023-02-01

[Ope23] OPENJS FOUNDATION: Node.js. https://nodejs.org/de/. Version: 2023. – Abruf am: 2023-02-01

[Osm14] OSMANOGLU, Ertem: Identity and Access Management. Waltham : Syngress, 2014

[RG20] RUPP, Christine ; GÜNTHER, Andreas: Schablonen für Anforderungen und User-Storys – MAST<sc>e</sc>R und andere Templates. In: Requirements-Engineering und -Management. 2020, Kapitel 19, S. 357–388

[Roy08] ROYER, Denis: Enterprise Identity Management. In: FISCHER-HÜBNER, Simone (Hrsg.) ; DUQUENOY, Penny (Hrsg.) ; ZUCCATO, Albin (Hrsg.) ; MARTUCCI, Leonardo (Hrsg.): The Future of Identity in the Information Society. Boston, MA : Springer US, 2008, S. 433–446

[RRG+10] RIVERO, José M. ; ROSSI, Gustavo ; GRIGERA, Julián ; BURELLA, Juan ; LUNA, Esteban R. ; GORDILLO, Silvia: From Mockups to User Interface Models: An Extensible Model Driven Approach. In: DANIEL, Florian (Hrsg.) ; FACCA, Federico M. (Hrsg.): Current Trends in Web Engineering. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, S. 13–24

[RRG+11] RIVERO, José M. ; ROSSI, Gustavo ; GRIGERA, Julián ; ROBLES LUNA, Esteban ; NAVARRO, Antonio: From Interface Mockups to Web Application Models. In: BOUGUETTAYA, Athman (Hrsg.) ; HAUSWIRTH, Manfred (Hrsg.) ; LIU, Ling (Hrsg.): Web Information System Engineering – WISE 2011. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011, S. 257–264

[RS21] ROMEIKE, Frank ; STALLINGER, Manfred: Risikomaße. In: Stochastische Szenariosimulation in der Unternehmenspraxis : Risikomodellierung,

Fallstudien, Umsetzung in R. Wiesbaden : Springer Fachmedien Wiesbaden, 2021, S. 235–261

[San93] SANDHU, Ravi S.: Lattice-Based Access Control Models. In: Computer 26 (1993), S. 9–19

[San96] SANDHU, Ravi S.: Role hierarchies and constraints for lattice-based access controls. In: BERTINO, Elisa (Hrsg.) ; KURTH, Helmut (Hrsg.) ; MARTELLA, Giancarlo (Hrsg.) ; MONTOLIVO, Emilio (Hrsg.): Computer Security — ESORICS 96. Berlin, Heidelberg : Springer Berlin Heidelberg, 1996, S. 65–79

[SCFY96] SANDHU, Ravi S. ; COYNE, E J. ; FEINSTEIN, H L. ; YOUMAN, C E.: Role-based access control models. In: Computer 29 (1996), S. 38–47

[SM22] STREIM, Andreas ; MANN, Simran: 7,8 Milliarden Euro: Markt für IT-Sicherheit wächst 2022 um 13 Prozent. https://www.bitkom.org/Presse/Presseinformation/IT-Sicherheit-waechst-2022#_. Version: Oktober 2022. – Abruf am: 2023-03-24

[Som16] SOMMERVILLE, Ian: Software Engineering, Global Edition. 10. Pearson Education, 2016

[SS75] SALTZER, J H. ; SCHROEDER, M D.: The protection of information in computer systems. In: Proceedings of the IEEE 63 (1975), S. 1278–1308

[SS94] SANDHU, Ravi S. ; SAMARATI, P: Access control: principle and practice. In: IEEE Communications Magazine 32 (1994), S. 40–48

[TDA19] THUAN, Nguyen H. ; DRECHSLER, Andreas ; ANTUNES, Pedro: Construction of design science research questions. In: Communications of the Association for Information Systems 44 (2019), März, S. 332–363

[Ter22] TERRACOTTA, INC: Quartz Scheduler. https://github.com/quartz-scheduler/quartz. Version: 2022. – Abruf am: 2023-02-05

[TeroJ] TERRACOTTA, INC: Quartz Job Scheduler. https://www.quartz-scheduler.org/. Version: oJ. – Abruf am: 2023-02-05

[The23a] THE APACHE SOFTWARE FOUNDATION: Maven - Introduction. https://maven.apache.org/what-is-maven.html. Version: 2023. – Abruf am: 2023-01-19

[The23b] THE POSTGRESQL GLOBAL DEVELOPMENT GROUP: PostgreSQL: About. https://www.postgresql.org/about/. Version: 2023. – Abruf am: 2023-01-19

[The23c] THE POSTGRESQL GLOBAL DEVELOPMENT GROUP: PostgreSQL: License. https://www.postgresql.org/about/licence/. Version: 2023. – Abruf am: 2023-01-19

[Ull21] ULLENBOOM, Christian: Java ist auch eine Insel: Einführung, Ausbildung, Praxis. 16. Bonn : Rheinwerk Verlag, 2021

[Vaa16] VAADIN LTD.: vaadin-core/LICENSE at master. https://github.com/vaadin/vaadin-core/blob/master/LICENSE. Version: 2016. – Abruf am: 2023-01-31

[Vaa22] VAADIN LTD.: Vaadin Commercial License and Service Terms. https://vaadin.com/commercial-license-and-service-terms. Version: November 2022. – Abruf am: 2023-01-31

[Vaa23a] VAADIN LTD.: Vaadin Flow. https://vaadin.com/flow. Version: 2023. – Abruf am: 2023-01-18

[Vaa23b] VAADIN LTD.: Vaadin UI Components. https://vaadin.com/components. Version: 2023. – Abruf am: 2023-02-17

[VMw19a] VMWARE, INC.: spring-boot/LICENSE.txt at main. https://github.com/spring-projects/spring-boot/blob/main/LICENSE.txt. Version: 2019. – Abruf am: 2023-01-19

[VMw19b] VMWARE, INC.: spring-framework/LICENSE.txt at main. https://github.com/spring-projects/spring-framework/blob/main/LICENSE.txt. Version: 2019. – Abruf am: 2023-01-19

[VMw23] VMWARE, INC.: Spring Boot. https://spring.io/projects/spring-boot#overview. Version: 2023. – Abruf am: 2023-01-19

[WK18] WÜBBENHORST, Klaus ; KAMPS, Udo: Revision von Skalenniveau vom 16.02.2018 - 15:59. https://wirtschaftslexikon.gabler.de/definition/skalenniveau-46555/version-269833. Version: 2018. – Abruf am: 2023-01-26

[YS15] YASAIN, Emrah ; SCHRYEN, Guido: Requirements for IT Security Metrics - an Argumentation Theory Based Approach. In: ECIS 2015 Completed Research Papers, 2015, S. 1–16

Erklärung an Eides statt

Ich habe die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Die Arbeit wurde bisher an keiner anderen Hochschule zur Erlangung eines akademischen Grades eingereicht. Die vorgelegten Druckexemplare und die dem Prüfer zur Verfügung gestellte elektronische Version (PDF-Datei) der Arbeit sind identisch.
Von den in §13 Abs. 3 PO 2015 vorgesehenen Rechtsfolgen habe ich Kenntnis.

Bad Griesbach i. Rottal, den 12. April 2023

Julian Bauer
Matrikelnummer 2259323