The present work was submitted to
the Research Group
Software Construction

of the Faculty of Mathematics,
Computer Science, and
Natural Sciences

Bachelor Thesis

A Classification Approach for Metric-based Evaluations of Authentication Design Patterns

presented by

Krisa Carka

Aachen, September 30, 2025

Examiner

Prof. Dr. rer. nat. Horst Lichter

Prof. Dr. rer. nat. Bernhard Rumpe

Supervisor

Alex Mattukat, M.Sc.

Acknowledgment

First of all, I would like to thank Prof. Dr. rer. nat. Horst Lichter for allowing me to write my bachelor's thesis at his chair. I am also grateful to him and Prof. Dr. rer. nat. Bernhard Rumpe for reviewing my thesis.

I extend my deep and sincere gratitude to my supervisor, M.Sc. Alex Mattukat, for his constant guidance. I deeply appreciate his helpful suggestions and constructive feedback throughout the entire course of this thesis.

Last but not least, my heartfelt gratitude goes to my family for their unwavering support throughout my studies. I would like to especially thank my brother, Mishel, for his tremendeous support during the writing of this thesis.

Krisa Carka

Abstract

Authentication Design Patterns provide promising design solutions for software architects to address security risks early in the development process. Despite their potential, the limited guidance in selecting appropriate patterns hinders their practical application. In this regard, Constraint-based Recommender Systems (CBRS) can be utilised to support architects in selecting suitable AuthN DPs that align with their specific requirements. However, to operationalise this approach, reusable knowledge is needed, with which AuthN DPs can be compared with each other. To address this limitation, this thesis proposes a property classification approach to model AuthN DPs through characteristic properties. Following the Design Science Research methodology, we conducted a focused literature review to identify properties and then systematically analysed them, leading to the development of a metamodel outlining the taxonomy of AuthN DP property models. The categorical properties specified by this metamodel can easily be utilised by a CBRS for straightforward comparisons between AuthN DPs. Furthermore, the metamodel outlines an abstraction hierarchy of quality properties, which enables systematic assessment at each abstraction level. To this end, we provide example metrics for each quality property hierarchy and demonstrate the applicability of this approach for metric-based evaluations and comparision of AuthN DPs. This classification approach lays the foundation for the implementation of Knowledge Bases for AuthN DPs that support their recommendation process and enable their evaluation.

Contents

List of Tables

List of Figures

1 Introduction

In today's digital landscape, security is of paramount importance and an integral part of software systems. However, this landscape is characterised by a concerning upwards trend in security vulnerabilities, as evidenced by the significant rise in Common Vulnerabilities and Exposures (CVE) records over the years. These records have escalated from 321 in 1999 to 17,308 in 2019, and finally reached 40,077 in 2024 [CVE24]. Consequently, the financial implications of security incidents due to these vulnerabilities are substantial. In 2025, the estimated worldwide costs related to security incidents are estimated at 10.29 trillion US dollars, with expectations of these costs potentially reaching around 15.63 trillion US dollars by 2029 [Sta24], thus resulting in a 51.89% increase. Such figures highlight the importance of investing in security measures to address security vulnerabilities.

Among the various vulnerabilities, authentication failures are particularly noteworthy. Apart from being listed in OWASP Top 10 Web Application Security Risks [21b] since 2010, they also have a great financial impact. In a recent survey by the Ponemon Institute [Bar24], where a total of 1,917 IT security practitioners worldwide participated, 49% of the respondents reported weak authentication attacks among the costliest attacks. Evidently, the implementation of strong authentication solutions, among other security measures, is indispensable.

To this end, software architects can employ the security-by-design paradigm to integrate security early in the Software Development Life Cycle (SDLC). However, there are many challenges for software engineering in effectively implementing secure solutions. To begin with, the complexity of modern software systems, driven by factors such as distributed architectures and interconnected services, is constantly increasing. Moreover, the lack of security experts alongside the lack of security-relevant information on the one side, as well as the limited practical application of existing approaches to security modelling such as AEGIS, UMLsec, SecureUML, and ASE [Sof23] on the other side, add to the complexity of developing secure systems. These issues become even more pronounced as Insecure Design ranks fourth among the OWASP Top 10 list of security risks, with a call for more use of threat modeling, secure design patterns and principles, and reference architectures [21a].

To address these challenges, the Security-Centered Architecture Modeling (SCAM) project within the Research Group Software Construction at RWTH Aachen, aims to research new approaches to support designing secure software systems [Sof23]. In this regard, in his master's thesis, Lammers [Lam24] has developed two metamodels: the Security Design Pattern (SDP) Description Metamodel and the SDP Knowledge Base Metamodel. The SDP Description Metamodel serves as a foundation for creating SDPs that encapsulate essential security information, thereby assisting architects in their de-

sign and development processes. On the other hand, the SDP Knowledge Base Metamodel is designed to support architects in selecting appropriate SDPs. To facilitate this, a security recommender approach, SecuRe, was proposed, which utilizes a Constraint-based Recommender System (CBRS) to suggest suitable SDPs that align with the specific security requirements by leveraging the information from the SDP Knowledge Bases [Lam24].

1.1 Research questions

In order to implement such SDP Knowledge Bases, first, reusable properties have to be identified that can serve as a basis to describe, compare and choose between different SDPs with the same intended functionality, such as authentication of entities. Focusing on SDPs on the security control Authentication, which we refer to as Authentication Design Patterns (AuthN DPs), this leads to our first research question:

> **RQ1:** *How can AuthN DPs be described through characteristic properties in a differentiated manner?*

Identifying properties that characterize AuthN DPs is essential to create a comprehensive Knowledge Base catalogue that provides a well-rounded overview of AuthN DPs. This catalogue provides a valuable resource for architects and stakeholders by easing the understanding of different AuthN DPs. Furthermore, these properties can serve as a basis to compare them with each other, which leads to our second research question:

> **RQ2:** *How can AuthN DPs be meaningfully compared with each other using such characteristic properties?*

This research question aims to facilitate the recommendation process and enable reasoning about AuthN DPs.

1.2 Contribution

To address the above research questions, we propose a property classification approach to model AuthN DPs through characteristic properties. To this end, we developed a property classification metamodel, which outlines the structure of property models for AuthN DPs, thereby facilitating the implementation of reusable Knowledge Bases that support the recommendation and selection of AuthN DPs.

The categorical properties specified in our metamodel allow for straightforward comparisions between AuthN DPs by categorizing them. These properties can be directly utilised by a CBRS to narrow down suitable AuthN DPs for specific technical and environmental factors.

To evaluate the quality of AuthN DPs for their comparison, our metamodel further outlines an abstraction hierarchy of quality properties, which enables systematic assessment at each abstraction level. We provide example metrics for each quality property hierarchy we defined to showcase their quantification.

To demonstrate the applicability of our concepts, we evaluate three different AuthN DPs on a quality property using the example metrics. While their evaluation is beyond the scope of this thesis, we lay the foundation for future research on metric-based evaluations of AuthN DPs.

1.3 Structure of this thesis

The structure of this thesis is as follows:

- In chapter 2, we introduce the fundamental concepts and terminology relevant for this thesis. This includes an overview on Authentication, Software Engineering, Security Patterns, Metrics and Measurement scales.
- In chapter 3, we present related work including the SDP Catalog Conception by Lammers [Lam24] and the SecuRe recommendation approach by Sabau et al. [SLL25].
- In chapter 4, we present the methodology we followed and its implementation.
- In chapter 5 we present and apply the Property Classification Metamodel for AuthN DPs that we conceptualized and present the results.
- In chapter 6, we demonstrate the applicability of the results by quantifying a quality property on three different AuthN DPs using the example metrics.
- In chapter 7, we discuss how our findings address the research questions and explain limitations.
- In chapter 8, we conclude the thesis by summarizing the main contributions and insights followed by some thoughts and recommendations on future work.

2 Foundations

This chapter presents the definitions and terminology essential to understand the key concepts of this thesis. First, we introduce key authentication terms and principles. Next, we outline the necessary definitions and terminology of software engineering. Then we give an overview on security patterns and their hierarchies. Finally, we provide a brief introduction on measurements and metrics.

2.1 Authentication

This section presents the necessary definitions and terminology on Authentication.

Definition 2.1 (Authentication). Authentication (AuthN) is the process of checking the claimed identity of an entity by validating one or more of the provided authenticators such as passwords, tokens or biometrics [25b].

Definition 2.2 (Authentication factor). An authentication factor refers to a type of evidence used to confirm the claimed identity of an entity.

Authentication factors are commonly classified as follows:

- **knowledge-based factors (something you know):** requires knowledge of secret information, such as a password, passphrase, PIN or answer to a secret security question
- **possession-based factors (something you have):** involves items or devices that the claimant possesses, such as a smart card, mobile device or electronic key
- **inherence-based factors (something you are):** relies on unique physiological or behavioral characteristics of the claimant, such as a fingerprint, retina or facial recognition data

Definition 2.3 (Authentication method). An authentication method is a specific technique employed to verify the claimed identity of an entity.

Authentication methods using only one type of authentication factor are referred to as Single-Factor Authentication (SFA), while those using two or multiple authentication factor types are referred to as Multi-Factor Authentication (MFA).

Definition 2.4 (Password-Based Authentication). Password-Based Authentication is a knowledge-based authentication method that utilizes a secret password linked to a unique user identifier to verify a user's identity. User identifiers and passwords are securely stored by the system, with passwords being protected through cryptographic hashing functions [CDW04].

PBA is one of the most common authentication methods, however, it is vulnerable to many attacks, such as phishing, shoulder surfing, and credential stuffing [AAM15]. Furthermore, it relies on the user's ability to create and remember strong passwords [CDW04]. To enhance security, PBA is frequently combined with additional authentication factors, thereby implementing MFA.

Definition 2.5 (OpenID Connect). OpenID Connect (OIDC) is an authentication protocol that operates on top of OAuth 2.0. It allows a Relying Party (RP) to check the identity of an entity based on the authentication carried out by an Identity Provider (IdP). The IdP issues an identity token in the form of a JSON Web Token (JWT), which includes authentication details, such as the unique identifier of the entity or the time of authentication. The IdP digitally signs the JWT using a private key, which allows the RP to verify the token's integrity and authenticity with the corresponding public key, which the IdP makes publicly accessible [FKS17].

2.2 Software Engineering

This section presents common definitions and terminology used in software engineering.

Definition 2.6 (Software system). A software system is composed of elements designed to satisfy specific requirements and the hardware needed to run these elements on [RW11].

Understanding a software system requires consideration of its architecture.

Definition 2.7 (System architecture). A system's architecture consists of static structures, which include internal design-time elements and their organization, and dynamic structures, which include runtime elements and their interactions [RW11].

An architecture can be depicted in multiple views, each emphasizing different aspects of the system, such as the functionality delivered or the interactions between components.

Definition 2.8 (View). A View depicts the architecture of a software system from a certain perspective that serves to address particular concerns of one or more stakeholders [RW11].

A complex system can be better comprehended through multiple interconnected views

that together depict its functions and quality attributes, than through a single, overly complex model [RW11].

Definition 2.9 (Viewpoint). A Viewpoint is a guideline on how to construct a specific class of views. It defines the purpose of this class of views and the stakeholders and their concerns that it addresses [RW11].

A viewpoint plays an essential role in ensuring that views are both coherent and comprehensible by explicitly outlining the structure and principles that each view should adhere to. Stakeholders who are acquainted with a particular viewpoint can easily grasp any view that conforms to it [RW11].

Definition 2.10 (Design pattern). A design pattern is a conceptual solution that has been validated for addressing recurring design challenges within a specific context [Gam+95].

Design patterns encapsulate best practices and proven strategies that can be applied to similar problems, thereby promoting efficiency and consistency in software design. The abstract solutions they provide can be implemented in various ways within their context.

2.3 Security Patterns

This section provides an overview on security patterns and their hierarchies.

Definition 2.11 (Security Control). A Security Control (SC) is any measure taken to protect the security goals (confidentiality, integrity and availability) of assets in a software system [Joi20].

AuthN is one of the most common SCs. Other common examples include Authorization, Intrusion Detection and Prevention Systems (IDPS), Distributed Denial of Service (DDoS) Prevention, etc.

Definition 2.12 (Security Pattern). A Security Pattern (SP) is a reusable, proven security solution that addresses a specific SC at a conceptual level [SLL25].

SPs must include the Context in which they are applicable, the Problem they address and the high-level Solution they provide for it. Moreover, they must provide an Example that showcases an implementation of the Solution and give Consequences for using this SP [Hey+07]. Some SPs that address the AuthN SC are PBA, OIDC and Fingerprint-based Authentication (FBA).

Definition 2.13 (Security Design Pattern). A Security Design Pattern (SDP) is a

concrete design solution of an SP [Lam24; SLL25].

While SPs offer solutions at a conceptual level, SDPs provide more detailed, concrete solutions at a design level. So, to accommodate different design options of conceptual SPs, each SP can be concretised by multiple SDPs. For example, an important design choice that affects the PBA SP is the password reset mechanism. So, a PBA SDP that uses a SMS Pin for its password reset mechanism requires additional architectural configurations for generating unique, time-limited PINs, handling of secure SMS delivery and creating a restricted session from that PIN that exclusively allows the user to reset their password, while a PBA SDP that uses a Security Question instead, only needs to securely store the user's answer to the Security Question [25c]. Such design choices impact the architecture of SDPs, thereby necessitating specific SDPs for each design option. The SP hierarchy and their relationships are illustrated in figure 2.1.

Definition 2.14 (Authentication Design Pattern). An Authentication Design Pattern (AuthN DP) is a SDP that specifically realises the AuthN SC.

AuthN DPs focus on implementing the AuthN SC within an architecture and provide structured approaches to effectively verify the identity of entities attempting to access a system. They serve as reusable design solutions that can be adapted to various software systems, ensuring that AuthN remains a reliable and secure component of the overall architecture. Some concrete examples of AuthN DPs are Lammer's OIDC SDPs and PBA SDP [Lam24].

2.4 Metrics

This section presents a brief overview of measurements, metrics and measurement scales.

Definition 2.15 (Measurement). A measurement is the process of assigning a numerical value to a characteristic of objects according to specific rules, to represent properties quantitatively or qualitatively [Dal18].

In software engineering, a measurement is used to quantify attributes of software products, processes, or resources.

Definition 2.16 (Metric). A metric is a specific measure that quantifies a characteristic of a system, component or process [Ele90].

Metrics can be defined on different measurement scales according to their characteristics, determining how data can be analyzed and interpreted. In the following, the different scales are presented [Dal18]:

- **Nominal scale:** groups data into separate categories without implying any order.

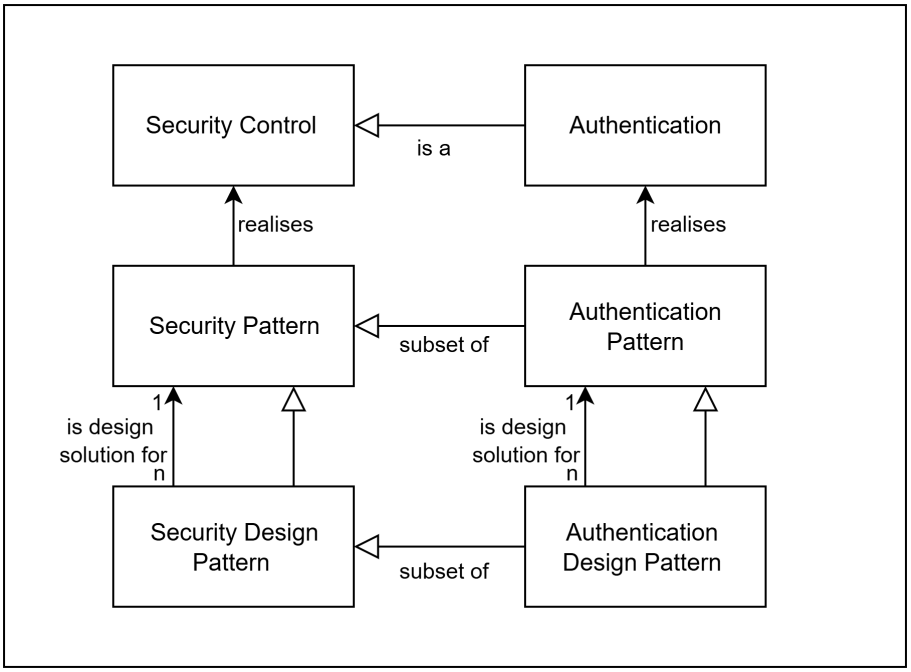

Figure 2.1: Security Pattern Hierarchy and their relationships (left) [SLL25]. Authentication Pattern Hierarchy and their relationships (right).

They are used when the data represent different types or labels, such as defect types like “UI,” “logic,” or “performance” bugs.

- **Ordinal scale:** classifies data with a meaningful order, but unequal intervals. For example, defect severity levels labeled “low”, “medium” and “high” are defined on an ordinal scale.
- **Interval scale:** provides numeric data where the difference between values is meaningful, but there is no true zero. It allows addition and subtraction, but not ratio comparisons. A common example is temperature in Fahrenheit.
- **Ratio scale:** has all the properties of interval scales, but also includes a true zero, making full arithmetic operations possible. Common examples include lines of code (LOC) or execution time, where “twice as much” has a meaningful interpretation.
- **Absolute scale:** is a special case of the ratio scale, used for exact counts with a natural zero and no negative values, such as the number of defects or function points, supporting all arithmetic operations. Counting metrics are defined in this scale.

3 Related Work

In this chapter we present related work for this thesis. We start with a review of the conceptualised SDP catalog by Lammers [Lam24]. Then we discuss the SDP recommender approach proposed by Sabau et al. [SLL25].

3.1 SDP Catalog Conception

In his master thesis, Lammers [Lam24] introduces SDPs designed to help architects securely implement a given security solution, comprising of an usage and a knowledge aspect. The usage aspect provides the information necessary for architects to understand and implement the pattern, while the knowledge aspect provides information necessary to reason about them. The usage aspect is defined by the SDPDM metamodel and the knowledge aspect is defined by the SDPKBM metamodel. In the following, we give a brief overview for each of these metamodels.

3.1.1 SDPDM metamodel by Lammers [Lam24]

The SDPDM builds upon the security pattern structure proposed by Heyman et al. [Hey+07]. Lammers further structures the SDPDM into four key viewpoints, each focusing on a different aspect of SDPs:

- **Conceptual Viewpoint:** This Viewpoint specifies how policies and roles are represented within an SDP, considering role responsibilities and relationships. A Conceptual View provides a high-level understanding of the SDP solution and forms the foundation upon which other views build.
- **Data Viewpoint:** The Data Viewpoint focuses on the modeling of data elements, describing their structure, properties, interrelations, and associated security considerations. It addresses the problem of inadequate data design leading to insecure implementations.
- **Behavioral Viewpoint:** This viewpoint specifies the behavioral models, that detail the interactions between roles defined in the Conceptual View and represent events that occur during these interactions. Each behavioural model depicts a different use case in detail.
- **Structural Viewpoint:** This viewpoint specifies how roles are represented in the architecture, considering both its static and dynamic structure. A Structural View

is intended to provide architects with practical examples and a well-documented foundation for implementing an SDP solution.

Lammers further defined three SDPs conform to this metamodel, which we make use of in chapter 6 for demonstration.

3.1.2 SDPKBM metamodel by Lammers [Lam24]

The SDPKBM metamodel provides the foundation for reasoned recommending reasonably appropriate patterns within a CBRS. It organizes knowledge into knowledge bases that primarily consist of three elements: attributes, recommendation factors and constraints. First, attributes describe the possible design choices for an SDP, such as the password reset mechanism for PBA, along with a set of their possible values, which allows modeling different available design solutions. Second, recommendation factors express how attribute values influence quality properties of a system like security, usability, or maintainability. This way they provide the reasoning behind recommendations by linking design choices to their consequences. Lastly, constraints are rules that ensure only valid combinations of attributes are considered during recommendations, filtering out infeasible or contradictory combinations. Together, these elements allow the knowledge base to represent both the technical characteristics of SDPs and their impact on system qualities, enabling systematic evaluation and transparent recommendations.

However, the SDPDM considers only attributes that differentiate SDPs implementing the same SP (e.g. PBA), assuming that an architect has already chosen an appropriate SP. For instance, assumming an architect has already chosen the PBA SP, the "Password Reset Mechanism" attribute is used to compare different PBA SDPs. We address this gap by providing pattern properties aimed at comparing different AuthN DPs types as well, thus contributing on the creation of an SDP knowledge base for all AuthN DPs.

3.2 SDP Recommender Approach

Building upon the previously introduced work, Sabau et al. [SLL25] propose a security recommender approach, SecuRe, which utilises a CBRS to suggest suitable SDPs that align with the specific security requirements of architects. At its core, SecuRe relies on a set of knowledge bases, specified by the SDPKB metamodel by Lammers [Lam24], that capture both the properties of SPs and the context under which they are applicable. Each knowledge base is tailored to a specific SP and encodes context properties as well as pattern properties. These knowledge bases are further enriched with constraints and filter conditions that ensure only valid and feasible combinations are considered. Building on this foundation, the SecuRe recommendation process consists of eight steps. This process begins with the architect specifying a security requirement and continues with an interactive question and answer loop for the definition of the realization context. To assist in this step, a Large Language Model (LLM) is also used to answer specific questions from the architect's side. Once the context is established, SecuRe uses its

knowledge bases to identify feasible SPs by solving a Constraint Satisfaction Problem (CSP), and then applies the Multi-Attribute Utility Theory (MAUT) to rank these potential SPs by calculating a recommendation score for each one. SecuRe then presents the ranked recommendations with explanations, enabling transparent decision-making, after which the architect can choose a preferred pattern. It then continues this process at a more concrete level by suggesting suitable SDPs for the chosen SP.

4 Methodology

Design Science Research (DSR) is a well-accepted research paradigm within the field of Information Systems Research [Hev+04; Pef+07]. Its applicability within Software Engineering research has been demonstrated by several publications, both as a research paradigm and as a practical methodology for artifact creation, evaluation and knowledge building [Sto+17; Eng+20; RES20]. This aligned with the nature of our thesis, hence, we follow the DSR methodology, as proposed by Peffers et al. [Pef+07], which includes six steps: problem identification and motivation, definition of objectives of a solution, design and development, demonstration, evaluation and communication. This thesis focuses primarily on the first four steps, laying the groundwork for future implementation and evaluation while including initial demonstration. In the following sections we describe how we implemented these steps.

4.1 Identify Problem and Motivate

As presented in chapter 1, authentication failures, along with insecure design, are in the OWASP TOP 10 list of security risks and have a great financial impact. To mitigate these risks, the security-by-design paradigm can be employed, using security patterns to integrate security early in the SDLC. However, the increasing complexity of modern software systems, the lack of security experts and security-relevant information as well as the limited practical application of existing approaches to security modelling present major challenges for software engineering in effectively implementing secure solutions [Sof23]. To address these challenges, a security recommender approach, SecuRe [SLL25], was proposed as part of the SCAM Project [Sof23]. This approach was developed through an amalgamation of various theses within the SCAM research project. One of these theses is the work of Lammers, named "Conception of a Security Design Pattern Catalog for Constraint-based Recommender Systems" [Lam24]. In this thesis two metamodels were developed: the SDPDM, which provides a foundation for creating SDP descriptions that encapsulate essential security information, and the SDPKM, that aims to support architects in selecting appropriate SDPs [Lam24]. The SecuRe approach utilises a Constraint-based Recommender System to suggest suitable SDPs that align with the specific security requirements by leveraging the information from SDP Knowledge Bases [SLL25]. However, to operationalise the SecuRe approach, first reusable knowledge is needed to develop such Knowledge Bases, with which SDPs can be compared with one another.

4.2 Define Objectives of a Solutions

This thesis focuses specifically on SDPs for Authentication (AuthN DPs). The ultimate goal is to make AuthN DPs comparable with one another. To this aim, we propose a property classification approach for metric-based evaluations of AuthN DPs. To meet this goal, we formulated a set of requirements, with the aim to ensure the comparability between AuthN DPs when these requirements are fulfilled. They were defined as follows:

- **R1: Universal -** A property should be applicable to all AuthN DPs.

This requirement ensures that a property can be used as a comparision basis between all AuthN DPs.

- **R2: Measureable -** A property should either be categorizable on a nominal scale or quantifiable by measurements on at least an ordinal scale.

This requirement ensures that AuthN DPs can be compared based on a property, either through categorizing or through ranking. Properties with non-categorical values are excluded by this requirement.

4.3 Design and Development

To address RQ1, we followed a bottom-up approach, where we conducted a focused literature review on existing standards and guidelines on authentication solutions, as well as publications on authentication patterns and solutions. We then analyzed and compared authentications solutions with one another to extract properties that they had in common. Furthermore, we synthesized properties from requirements and suggestions of standards and guidelines on authentication solutions. In the end, we obtained a set of properties characterising AuthN DPs, which we assessed and filtered according to the two requirements of our solution objective defined in the previous step.

To address RQ2, we systematically analysed the properties from the obtained set and thereby identified through subjective assessments that these properties can be classified into two major groups based on the scale their values are defined. The first group included categorical properties, whose values are on a nominal scale. The second group included properties, whose values can be quantified on at least an ordinal scale. This insight led to the conceptualisation of a metamodel for property classifications of AuthN DPs.

With the goal of quantifying the second group of properties, which we identified to mostly be quality properties, we decided on an iterative approach, to focus on each quality property individually, as illustrated in figure 4.1. In the following, we present the methodology we implemented for each iteration to quantify a quality property.

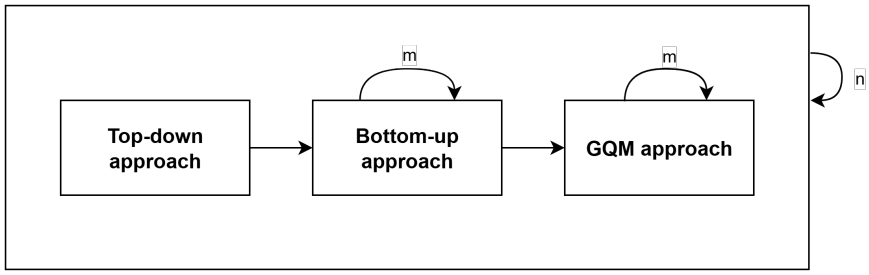

Figure 4.1: Iterative approach over n quality properties. The top-down approach breaks a quality property into m granular properties. The bottom-up approach and then the GQM approach are applied over each of these m granular properties.

Top-down approach

First, we employed a top-down approach to break down a quality property into more granular ones. This approach allowed us to get a better understanding of these properties and ease their quantification. Moreover, this resulted in an abstraction hierarchy of these quality propeties, which further refined the property classification metamodel we conceptualised.

Bottom-up approach

Next, we followed a bottom-up approach to identify basis data and measures, with which we could measure each of these granular properties. We did this iteratively over each granular property by conducting a targeted literature review and utilising information from the SDPDM and the AuthN DP examples by Lammers [Lam24].

Goal-Question-Metric approach

We adopted the Goal-Question-Metric (GQM) approach by Basili et al. [BCR94], to connect the results from the top-down approach with the results from the bottom-up approach. With the goal of quantifying each granular property, we formulated questions to get a better understanding of these properties. Using the basis data and measures resulting from the bottom-up approach, we defined and assigned example metrics for each granular property. If a question was too complex or abstract to directly assign a metric, we broke it down to subquestions until a metric could be directly assigned. This approach resulted in example metrics for each granular property.

4.4 Demonstration

We demonstrate the applicability of our results by quantifying a quality property on three different AuthN DPs using the example metrics (see chapter 6). We then do a

comparision of these AuthN DPs on the basis of this quality property. For the demonstration we use the AuthN DPs from/defined in Lammers' master thesis [Lam24].

4.5 Evaluation

While evaluation is an important step of DSR, due to the time constraints of a bachelor thesis, an evaluation of our findings is out of scope. Instead, we suggest this as future work through expert studies.

4.6 Communication

We communicate our findings through this thesis report. Furthermore, we plan on publishing a paper that will include the work of this thesis and its findings.

5 Authentication Design Pattern Property Classification

This chapter presents a Property Classification of AuthN DPs. As the ultimate goal of this thesis is to make AuthN DPs comparable with one another, our approach is to model AuthN DPs through a set of characteristic properties that can be used for comparision. For this, we conceptualised the Property Classification Metamodel for AuthN DPs, which is introduced in section 5.1. This metamodel lays the ground work to define properties that make AuthN DPs comparable. We then applied the metamodel and present the results in section 5.2 and section 5.3.

5.1 Property Classification Metamodel for AuthN DPs

This section introduces the Property Classification Metamodel (PCM) for AuthN DPs, which outlines how to model AuthN DPs through characteristic properties. By clearly specifying the properties that define these patterns, the PCM ensures that property models of AuthN DPs follow a consistent structure, thereby enhancing their comprehensibility and applicability.

We first classify properties into two distinct groups:

- Properties, whose values are on a nominal scale
- Properties, that are quantifiable or measurable on at least an ordinal scale

In the following, we provide formal definitions of each group and their taxonomy, as illustrated in figure 5.1. In the subsequent sections, we present the identified properties conform to this taxonomy.

Definition 5.1 (Categorical Property). A Categorical Property is a discrete characteristic of AuthN DPs that takes on a finite set of predefined values. Its values are categorised on a nominal scale.

An example of Categorical Properties (CPs) is Reauthentication Enforcing, which takes on binary values “Yes” for AuthN DPs which enforce reauthentication of entities and “No” for AuthN DPs that do not enforce it. This property is defined on a nominal scale, as it does not imply any order among the values.

Categorical Properties can be directly used to compare AuthN DPs, as their values are fixed and well-defined. Quantifiable Properties, on the other hand, need prior quantification (e.g. through metrics) in order to allow for comparision between AuthN DPs on their basis. Among quantifiable properties, this thesis focuses exclusively on those quantifiable properties that describe a specific qualitative aspect of AuthN DPs. In this context, we adapt the quality model of ISO/IEC 25010 [23] for SPs and AuthN DPs, as it is a widely recognized standard in software engineering for assessing product quality and models both external and internal properties.

Definition 5.2 (Quality Property). A Quality Property describes a specific high-level quality characteristic that SPs have in common.

A typical example of Quality Properties for SPs is Complexity, which refers to the overall degree of intricacy and interdependence of the different elements in the architecture of an SP. This is a crucial quality characteristic as it significantly influences the understandability, maintainability and scalability of an SP.

Quality Properties can moreover be specific to a SC. For instance, AuthN DPs can be characterised by their AuthN strength, which e.g. does not characterise the DDoS Prevention SC. Therefore we need an AuthN-specific definition of Quality Properties.

Definition 5.3 (AuthN Quality Property). An AuthN Quality Property describes a specific high-level quality characteristic of AuthN DPs to facilitate their comparability.

As this thesis focuses on AuthN DPs and not on SPs, for the sake of simplicity, we will not distinguish in the remainder of this thesis whether a Quality Property is generally applicable to all SPs or to AuthN DPs only. Instead, we will consider all Quality Properties to be applicable to AuthN DPs. We do this without loss of generality: while some of the Quality Properties may be generally applicable to all SPs, we leave a detailed investigation of this question open for future work. However, since AuthN DPs are more specialized SPs and can be modeled by a specific Quality Property Q, this semantics is preserved if this Q is also generally applicable to all SPs and is thus defined as a Quality Property for SPs instead of AuthN DPs. So, in the remainder of this thesis we will use the term Quality Properties (QPs) to refer to AuthN Quality Properties (AuthN QPs), such that all subsequent references of QPs will adhere to the definition of AuthN QPs.

Definition 5.4 (Quality Attribute). A Quality Attribute represents a more granular subcharacteristic of one or more QPs.

For example, AuthN Correctness, which refers to the degree of correctly identified legitimate and illegitimate users, and Resistance, which refers to the ability to detect, withstand and recover attacks, can both be modeled as Quality Attributes (QAs) of the AuthN Strength QP. Additionally, AuthN Correctness can also be modeled as a QA

of the Reliability QP. Identifying and defining QAs of each QP enables the quantification of QPs, thereby facilitating meaningful comparisons between different AuthN DPs. The concept of Quality Attributes corresponds to the quality sub-characteristics of the ISO/IEC 25010 quality model [23].

QAs can moreover be broken down into more granular QAs. For instance, Resilience, which refers to the ability to detect, withstand and recover from faults, failures and security breaches, can be broken down to two more granular QAs: Resistance and Fault Tolerance, where the latter refers to the ability to continue functioning correctly and securely in the event of errors, faults or failures. Meanwhile, Resilience itself can be modeled as a QA of the Reliability QP.

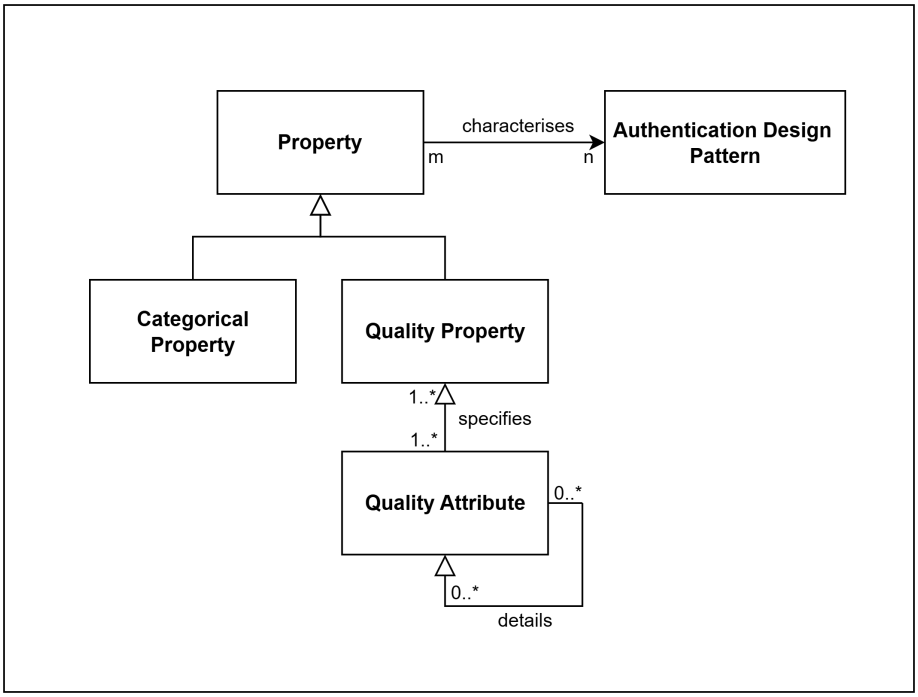

Figure 5.1: Overview of the property taxonomy defined by the Property Classification Metamodel for AuthN DPs.

Together, QPs and QAs form an abstraction hierarchy (illustrated in figure 5.1) that not only allows for a better understanding of the quality of AuthN DPs, but also aids in the evaluation of existing AuthN DPs by enabling a systematic assesment at each level of abstraction.

5.2 PCM Application for Categorical Properties

In the following we present the identified Categorical Properties specified by the PCM introduced in the previous section.

Reauthentication Enforcing: specifies whether the AuthN DP Solution enforces users to reauthenticate during a session.

This property can take the following values:

- **Yes:** Reauthentication is required after a specified amount of time or upon certain actions.
- **No:** No reauthentication is required during the session.

Reauthentication Enforcing is important for maintaining security in long-lived sessions. For instance, an AuthN DP that enforces reauthentication after a predefined time can help ensure that the user remains authorized throughout the session, reducing the risk of unauthorized access. A CBRS can utilise this property to narrow down AuthN DPs based on application context an architect provides. For example, in banking applications, where transactions involve significant amounts of money, it is encouraged to enforce reauthentication, thus AuthN DPs that take the value "Yes" should be prioritised in recommendations. In contrast, for social media platforms, where user engagement is prioritized, reauthentication is discouraged, thus AuthN DPs that take the value "No" should be recommended.

Compliance: specifies which regulatory laws and standards the AuthN DP Solution adheres to.

It includes the following regulatory laws and standards:

- **ISO 27001:** Adherence to the International Organization for Standardization's standard for Information Security Management Systems (ISMS).
- **NIST 800-63:** Adherence to the NIST Special Publication 800-63 Digital Identity Guidelines.
- **ASVS:** Adherence to the Application Security Verification Standard.
- **NIST 800-53:** Compliance with the NIST Special Publication 800-53 Security and Privacy Controls for Information Systems and Organizations.
- **HIPAA:** Compliance with the Health Insurance Portability and Accountability Act.
- **GDPR:** Compliance with the General Data Protection Regulation.
- **CCPA:** Adherence to the California Consumer Privacy Act.
- **FISMA:** Compliance with the Federal Information Security Management Act.
- **PCI DSS:** Adherence to the Payment Card Industry Data Security Standard.
- **PSD2:** Compliance with the Revised Payment Services Directive.

AuthN DPs take as values for this property a set containing regulatory laws and standards from the ones listed above, where an empty set implies that the AuthN DP Solution adheres to none of the above.

Compliance with regulatory laws and standards is essential for organizations to mitigate legal risks, ensure data protection and maintain trust with customers and stakeholders. For example, an organization handling healthcare data must comply with HIPAA to protect patient information, while a financial institution must adhere to PCI DSS to secure credit card transactions. A CBRS can filter AuthN DPs based on their compliance with these standards, ensuring that the suggested AuthN DPs meet legal and industry requirements.

Credential Revocation: refers to whether the AuthN DP Solution includes the possibility to revoke previously issued credentials.

This property can take the following values:

- **Provided:** Credential revocation is supported by the the AuthN DP Solution, allowing administrators or users to revoke credentials when needed.
- **Not Provided:** The AuthN DP Solution does not provide credential revocation, meaning that once credentials are issued, they remain valid indefinitely.

For example, in banking applications, it is essential for users to be able to revoke their credentials, if they suspect their banking credentials have been compromised. Thus, a CBRS should recommend AuthN DPs that provide ceredential revocation.

5.3 PCM Application for Quality Properties & Example Metrics

In the following we introduce the identified QPs. For each QP we present an abstraction hierarchy conform to the PCM defined in section 5.1. To show how these properties can be quantified, we present example metrics for them.

5.3.1 Complexity QP

Definition 5.5 (Complexity). The overall degree of intricacy and interdependence of the different elements in the architecture of the AuthN DP.

Complexity is an important intrinsic quality characteristic for AuthN DPs, which highly impacts the understanding of the AuthN DP Solution for architects and stakeholders. When considering the complexity of AuthN DPs, there are two different and very important aspects to regard, namely Design and Implementation Complexity, which we model as QAs of the Complexity QP.

Definition 5.6 (Design Complexity). Degree of complexity of the design models of an AuthN DP.

In this regard, the Conceptual, Data, Behavioural and Structural Views, included in the Architectural Description of an AuthN DP conform to the SDPDM by Lammers [Lam24], represent design models of the AuthN DP. As such, their complexity needs to be considered for the Design Complexity QA. So, we model and define Conceptual Complexity, Data Complexity, Behavioural Complexity and Structural Complexity as more granular QAs of the Design Complexity QA.

Definition 5.7 (Conceptual Complexity). Degree of complexity of the underlying policies of the AuthN Solution and the roles that implement them.

We can measure the Conceptual Complexity of an AuthN DP, using the following metric, which considers the roles, their relationships and the policy points they implement, as defined in the Conceptual Viewpoint:

$$Complexity_{Conceptual} = |Roles| + |Policy\_Points| + |Relationships|$$

Definition 5.8 (Data Complexity). Degree of complexity of the data models and their relationships.

To measure the Data Complexity of an AuthN DP, we can use the following metric, which considers the data models as defined in the Data Viewpoint:

$$Complexity_{Data} = |M_D| * avg_{m \in M_D}(Complexity_{DM}(m))$$

with M_D the set of data models of the AuthN DP (found in the Data View of the SDPDM) and the following counting metric:

$$Complexity_{DM}(m) = |F_m| + |Ref_m| + |TF_m| + |R_m|$$

with F_m the set of fields, Ref_m the set of references to other data models, TF_m set of transformation fields and R_m set of data rules applied to the data model m.

Definition 5.9 (Behavioural Complexity). Degree of complexity of the behavioural models of an AuthN DP.

We can measure the Behavioural Complexity of an AuthN DP, using the following metric, which considers the behavioural models as defined in the Behavioural Viewpoint:

$$Complexity_{Behavioural} = |M_B| * avg_{m \in M_B}(Complexity_{BM}(m)) + |E|$$

with M_B the set of behavioural models of the AuthN DP and $Complexity_{BM}(m)$ a counting metric for the number of steps in a behavioural model m.

Definition 5.10 (Structural Complexity). Degree of complexity of the architectural models and their relationships.

To measure the Structural Complexity of an AuthN DP, we can use the following metric, which considers the architectural models, their elements and relationships as defined in the Structural Viewpoint:

$$Complexity_{Structural} = |M_S| * avg_{m \in M_S}(Complexity_{SM}(m))$$

with M_S the set of architectural models of the AuthN DP (found in the Structural View of the SDPDM) and the following counting metric:

$$Complexity_{SM}(m) = |AME_m| + |Relationships_m|$$

with AME_m the set of architectural modeling elements (AME) and $Relationships_m$ the set of relationships between AMEs in the architectural model m.

As the elements of the models in each View and their relationships are specified in the respective Viewpoints in the SDPDM, the input data for the above metrics is easy to obtain for any AuthN DP defined conform to the SDPDM. Using the assessments from the metrics of these four QAs, we can quantify Design Complexity as follows:

$$\begin{aligned} Complexity_{Design} = \quad & a_1 * Complexity_{Conceptual} \\ & + a_2 * Complexity_{Data} \\ & + a_3 * Complexity_{Behavioural} \\ & + a_4 * Complexity_{Structural} \end{aligned}$$

with a_1, a_2, a_3, a_4 their respective weights.

Definition 5.11 (Implementation Complexity). Degree of complexity of an implementation of the AuthN DP.

As the Structural View of an AuthN DP showcases an implementation example of its Solution [Lam24], the Structural Complexity of an AuthN DP is an essential, more granular QA of Implementation Complexity. Moreover, the Data and Behavioural Views illustrate the structure and relationships between the data and behavioural models that are to be implemented. As such, we also model Data Complexity and Behavioural Complexity as more granular QAs of the Implementation Complexity QA (see figure 5.2). By breaking down the Implementation Complexity QA into more granular QAs, we leave room for a more granular assessment. The following is an example metric of how we can quantify it:

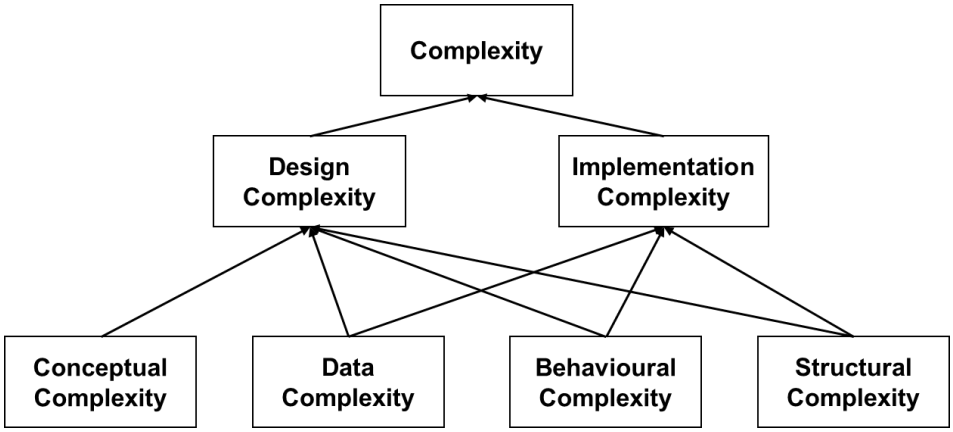

Figure 5.2: Abstraction hierarchy of the Complexity QP conform to the PCM

$$\begin{aligned} Complexity_{Implementation} = \quad & b_1 * Complexity_{Data} \\ & + b_2 * Complexity_{Behavioural} \\ & + b_3 * Complexity_{Structural} \end{aligned}$$

with weights b_1, b_2, b_3.

Finally, we can use the following example metric to quantify the Complexity QP using the assessments for the Design Complexity and Implementation Complexity QAs:

$$Complexity = w_1 * Complexity_{Design} + w_2 * Complexity_{Implementation}$$

5.3.2 AuthN Strength QP

Definition 5.12 (AuthN Strength). Robustness of the AuthN DP Solution in verifying the identity of users.

AuthN Strength reflects the effectiveness of the core functionality of AuthN DPs, making it an essential QP. We model AuthN Strength through three QAs: AuthN Correctness, Resistance and Authenticator strength (see figure 5.3).

Definition 5.13 (AuthN Correctness). Degree to which the AuthN DP Solution correctly identifies legitimate and illegitimate users.

AuthN Correctness reflects the effectiveness of the AuthN DP Solution in verifying the identity of users. High correctness rates indicate that the AuthN DP Solution is functioning effectively, thereby enhancing its strength. A common metric for quantifying

AuthN Correctness is the following:

$$Correctness = \frac{TP + TN}{TP + FP + TN + FN}$$

where

- TP (True Positives) represent the number of correctly identified legitimate users,
- TN (True Negatives) represent the number of correctly identified illegitimate users,
- FP (False Positives) represent the number of incorrectly identified illegitimate users as legitimate, and
- FN (False Negatives) represent the number of incorrectly identified legitimate users as illegitimate.

Definition 5.14 (Resistance). Ability of the AuthN DP Solution to withstand and recover from security breaches.

A robust AuthN DP Solution must not only function correctly under normal circumstances, but also demonstrate resilience when faced with attacks. To quantify the Resistance of an AuthN DP Solution we can use the following vulnerability-based example metric $Resistance \rightarrow [0, 20]$:

$$Resistance = \frac{1}{n} * \sum_{v \in V} (2 - D(v)) * CVSS(v)$$

with

- $V :=$ set of known vulnerabilities of the AuthN DP Solution
- $CVSS(v) \rightarrow [1, 10] :=$ the CVSS [25a] severity score of the vulnerability $v \in V$
- $M(v) \rightarrow [0, 2] :=$ the score of defense strategies of the AuthN DP Solution against the vulnerability $v \in V$ with

  $0 :=$ no defense, $1 :=$ mitigation or recovery, $2 :=$ prevention

A lower score of this metric indicates higher Resistance. To illustrate the application of this metric, consider the following fictitious example: AuthN DP A has the following known vulnerabilities: SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF) with respective CVSS scores 8.8, 5.4, and 8.8. A prevents XSS, and mitigates SQL Injection and CSRF. We calculate the Resistance score according to

the above metric:

$$\begin{aligned} Resistance &= \frac{1}{n} \times \sum_{v \in V} (2 - D(v)) \times CVSS(v) \\ &= \frac{1}{3} \times [(2 - D(\text{SQL})) \times CVSS(\text{SQL}) + (2 - D(\text{XSS})) \times CVSS(\text{XSS}) \\ &\quad + (2 - D(\text{CSRF})) \times CVSS(\text{CSRF})] \\ &= \frac{1}{3} \times [(2-1) \times 8.8 + (2-2) \times 6.5 + (2-1) \times 8.8] \\ &= \frac{1}{3} \times [1 \times 8.8 + 0 \times 6.5 + 1 \times 8.8] \approx 5.87 \end{aligned}$$

A has a Resistance score of 5.87, which can be considered as a high level of Resistance.

Definition 5.15 (Authenticator strength). Effectiveness of the AuthN factor(s) used in the AuthN DP Solution against forging.

The strength of the authenticator(s) used in the AuthN DP Solution plays a significant role in the robustness of the whole AuthN DP Solution. The most common way to measure the strength of different authenticator types is through their entropy [KD19]. Thus, we provide the following entropy-based example metric for quantifying the Authenticator strength QA:

$$E_{total} = \begin{cases} E(f), & |F| == 1 \\ min_{f \in F}(E(f) & |F| > 1 \quad and \quad options > 1 \\ max_{f \in F} E(f) & |F| > 1 \quad and \quad options == 1 \end{cases}$$

with F set of AuthN factors used and $E(f)$ entropy of the AuthN factor $f \in F$:

$$E(f) = \begin{cases} \frac{1}{FMR(f)} & f\, biometric\, AuthN\, factor \\ log_2(N^L) & else \end{cases}$$

with L length of the AuthN factor used and N the number of possible characters.

The first case in this metric considers AuthN DPs that implement SFA using only one authenticator and directly assigns its entropy. The second case considers AuthN DPs that also implement SFA but provide different authenticator options. Here, following the "as secure as weakest link" principle, the lowest entropy across all authenticators is assigned. Finally, the last case considers AuthN DPs that implement MFA, and assigns the highest entropy across all authenticators. As the entropy of biometric authenticators is difficult to measure, their entropy is approximated in practice as $\frac{1}{FMR(f)}$ where $FMR(f)$ is the False-Match rate of the authenticator f.

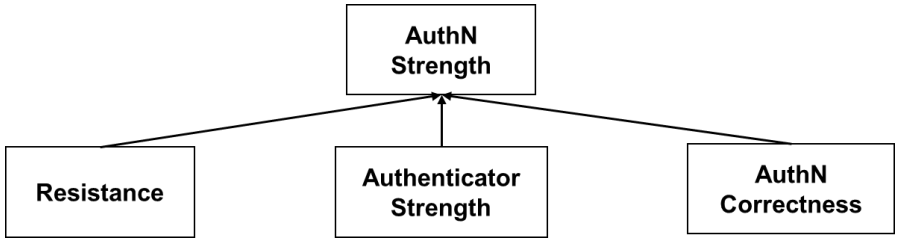

Figure 5.3: Abstraction hierarchy of the AuthN Strength QP conform to the PCM

5.3.3 Reliability QP

Definition 5.16 (Reliability). Ability of the AuthN DP Solution to correctly and consistently verify user identities and perform authentication.

Reliability is crucial for maintaining user trust and system integrity. It comprises three key components: AuthN Correctness, Resilience and Consistency, which we model as QAs (see figure 5.4).

Definition 5.17 (Consistency). Degree to which the AuthN DP Solution yields the same outcome upon repeated AuthN requests by the same user.

Consistency is essential for ensuring a reliable user experience and maintaining security. Different AuthN factors exhibit varying levels of consistency. While verifying knowledge- and possession-based authenticators like passwords or tokens is deterministic, recognition of biometric authenticators is probabilistic, and thus can lead to inconsistent AuthN request outcomes. To measure the Consistency of the AuthN DP Solution, we need to consider its Matching Correctness, which is defined as follows:

$$MatchingCorrectness = \begin{cases} 1 & F_b == \emptyset \\ avg_{f \in F_b}(FMR(f) + FnMR(f) & |F_b| >= 1 \end{cases}$$

with F set of AuthN factors used in the AuthN DP, $F_b \subseteq F$ set of biometric AuthN factors used, $FMR(f)$ and $FnMR(f)$ the False Match and False Non-Match rate of the biometric AuthN factor f.

Definition 5.18 (Resilience). Ability of the AuthN DP Solution to withstand and recover from faults, failures and security breaches.

We break Resilience down into two more granular QAs: Resistance and Fault Tolerance. Using measures of these granular QAs, we can quantify Resilience as follows:

$$Resilience = w_1 * Resistance + w_2 * Fault\,Tolerance$$

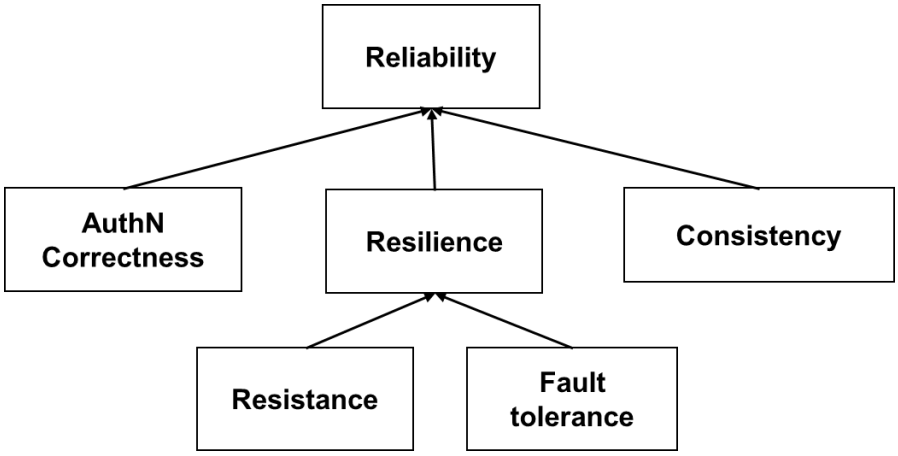

Figure 5.4: Abstraction hierarchy of the Reliability QP conform to the PCM

Definition 5.19 (Fault tolerance). Ability of the AuthN DP Solution to continue functioning correctly and securely in the event of errors, faults or failures.

To quantify Fault tolerance, we can use the following example metric:

$$Fault\ tolerance = \frac{|E_{FB}|}{|E|}$$

with $Fault\ tolerance \rightarrow [0, 1]$, E set of all Error Events (included in the Behavioural View of an AuthN DP) and $E_{FB} \subseteq E$ set of Error Events including a fallback behaviour.

5.3.4 Usability QP

Definition 5.20 (Usability). Ease with which end users can effectively and efficiently complete AuthN tasks.

Usability is essential for user adoption and satisfaction. An AuthN process with a low level of Usability can lead to user frustration and abandonment. We model this QP through four QAs: Response time, User Friendliness, Accessibility and Reset Usability (see figure 5.5).

Definition 5.21 (Response time). The time required from the AuthN DP Solution to process an AuthN request and provide feedback to the end user.

Response time encompasses how efficiently users can perform authentication. Users expect quick feedback during authentication, and delays can lead to a perception of inefficiency. The Response time of an AuthN DP Solution can be quantified by the following metric provided by U.M.A. Technology [UMA25] that measures the total latency of the AuthN flow:

$$L_{total} = L_{network} + L_{crypto} + L_{processing} + L_{audit} + L_{MFA}$$

with:

- $L_{network}$ representing network transmission delay (client-server round trip),
- L_{crypto} representing cryptographic verification delay,
- $L_{processing}$ representing server and backend processing time,
- L_{audit} representing logging or auditing delay, and
- L_{MFA} representing multi-factor verification delay

Definition 5.22 (User Friendliness). Degree to which end users can effectively authenticate themselves with minimal effort and cognitive load.

A user-friendly AuthN process minimizes cognitive load and reduces the likelihood of errors, thus improving user experience. We break User Friendliness down to two more granular QAs: Simplicity and Memorability (see figure 5.5).

Definition 5.23 (Simplicity). Straightforwardness of the AuthN process the AuthN DP offers for end users.

The Simplicity of the AuthN process allows users to authenticate themselves with ease, reducing the chances of user errors and improving overall user satisfaction. Complex processes can deter users from completing authentication. An example metric to quantify the Simplicity QA is a counting metric of the user steps required by the AuthN DP Solution, where a higher number of user steps indicates a reduced level of Simplicity.

Definition 5.24 (Memorability). Ease with which users can recall their credentials.

Poor memorability of AuthN credentials increases the effort required for users to authenticate themselves. According to NIST [GFN+17], the probability of a recall failure rises when there are more items for users to remember. Thus, a simple and easy to obtain metric for Memorability is a counting metric of the credentials users need to recall in order to authenticate, where a higher number of credentials to recall indicates lower Memorability. A more precise metric could be the sum of the entropies of the credentials to recall. However, user credential entropy is usually very difficult to obtain [GFN+17].

Definition 5.25 (Accessibility). Degree to which the AuthN process is accessable to all users.

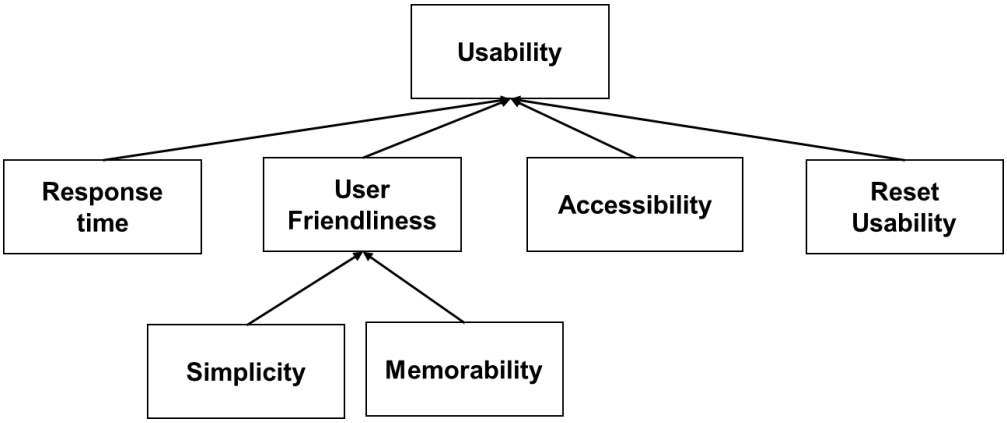

Figure 5.5: Abstraction hierarchy of the Usability QP conform to the PCM

An accessible AuthN DP Solution that accommodates all individuals, including those with impairments, promotes inclusivity and ensures compliance with legal standards. Furnell, Helkala, and Woods [FHW22] has assessed the potential usability impacts that the impairment of nine different International Classification of Functioning, Disability and Health (ICF) functions is likely to introduce in relation to different AuthN methods. Based on these findings, we can consider a simple counting metric of the ICF impairment classes that are likely to affect the usability of the AuthN method employed in the AuthN DP to quantify the Accessibility QA. In this regard, a higher number of the ICF impairment classes indicates a lower level of Accessibility.

Definition 5.26 (Reset Usability). Ease with which users can reset their credentials.

Reset Usability is critical for allowing users to regain access to their accounts without excessive difficulty. A complicated reset process can lead to user frustration and increased support costs.

The following example metric can be used to quantify the Reset Usability QA:

$$Usability_{Reset} = Opt_{Reset} * (Dev_{Reset} + Steps_{Reset} + Memorability_{Reset})$$

with Opt_{Reset} the number of reset options provided by the AuthN DP, Dev_{Reset} the number of standby devices required for the reset process, $Steps_{Reset}$ the number of user steps required and $Memorability_{Reset}$ the memorability of the reset process (quantified by the example metric of the Memorability QA).

A higher $Usability_{Reset}$ value indicates a lower level of the Reset Usability QA.

5.3.5 Performance QP

Definition 5.27 (Performance). Efficiency and effectiveness of the AuthN DP Solution.

We model Performance through three QAs (see figure 5.6): Response time and Resource Consumption, which reflect the efficiency of the AuthN DP Solution, and AuthN Correctness, which reflects the effectiveness of the AuthN DP Solution.

Definition 5.28 (Resource Consumption). Efficiency of the AuthN process in terms of memory, computation and communication.

While Response time concerns the time efficiency of the AuthN DP Solution, Resource Consumption represents its operational efficiency. It reflects how well the AuthN DP Solution utilised system resources. High levels of Resource Consumption can create bottlenecks, hindering the AuthN DP Solution's capacity to handle AuthN requests promptly and degrading its Performance. We break Resource Consumption down to three more granular QAs (see figure 5.6): Memory Consumption, Computational Consumption and Communication Consumption. Using measures of these granular QAs, we can quantify Resource Consumption as follows:

$$\begin{aligned} ResourceConsumption = \quad & Consumption_{Memory} + Consumption_{Computational} \\ & + Consumption_{Communication} \end{aligned}$$

Definition 5.29 (Memory Consumption). Amount of system memory used during authentication.

Memory Consumption reflects how efficiently the AuthN DP Solution utilises memory resources. Excessive memory usage can limit the number of concurrent AuthN requests that the system can handle. We can quantify it using the following example metric:

$$Consumption_{Memory} = \sum_{d \in D} S_d$$

with D the set of all stored data items by the AuthN Solution and S_d the size of each stored data item d.

Definition 5.30 (Computational Consumption). Processing power required for cryptographic operations.

Computational Consumption reflects the computational overhead of the AuthN DP Solution. We can quantify it as follows:

$$Consumption_{Computational} = \sum_{O_i \in C} P(O_i)$$

where C is the set of cryptographic operations used in the AuthN DP Solution, O_i represents the i-th cryptographic operation and $P(O_i)$ is its processing power.

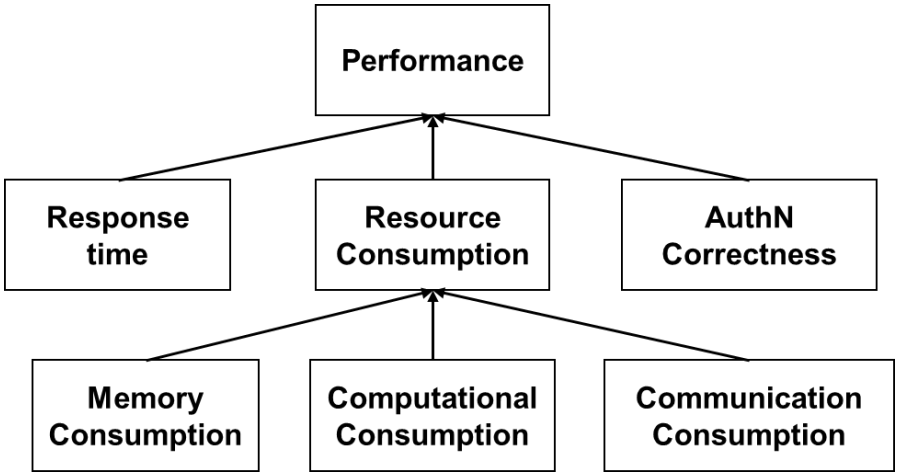

Figure 5.6: Abstraction hierarchy of the Performance QP conform to the PCM

Definition 5.31 (Communication Consumption). Amount of data exchanged during authentication.

Communication Consumption reflects how efficiently the AuthN DP Solution utilises the network bandwidth. This QA is critical in environments with limited bandwidth. We can quantify it using the following example metric:

$$Consumption_{Communication} = N \times (S_{avg} + O)$$

with N the exchanged message count, S_{avg} the average exchanged data size and O the protocol overhead.

6 Demonstration

In this chapter, we demonstrate the application of the example metrics for the Complexity QP and its QAs on the three AuthN DPs defined by Lammers [Lam24]: two OIDC AuthN DPs, which we refer to as OIDC1 and OIDC2, and one PBA AuthN DP. We then compare these AuthN DPs with each other using the calculated scores from the metrics.

Conceptual Complexity QA

We begin by quantifying the Conceptual Complexity QA on each AuthN DP. The basis data neccesary for the application of this metric is shown in table 6.1.

$$\begin{aligned} Complexity_{Conceptual}(OIDC1) = &\quad |Roles_{OIDC1}| + |Policy\_Points_{OIDC1}| \\ &+ |Relationships_{OIDC1}| \\ = &\quad 10 + 11 + 14 = 35 \end{aligned}$$

$$\begin{aligned} Complexity_{Conceptual}(OIDC2) = &\quad |Roles_{OIDC2}| + |Policy\_Points_{OIDC2}| \\ &+ |Relationships_{OIDC2}| \\ = &\quad 11 + 8 + 15 = 34 \end{aligned}$$

$$\begin{aligned} Complexity_{Conceptual}(PBA) &= |Roles_{PBA}| + |Policy\_Points_{PBA}| + |Relationships_{PBA}| \\ &= 14 + 17 + 26 = 55 \end{aligned}$$

These results indicate that PBA has the highest level of Conceptual Complexity out of the three, followed by OIDC1 and in the end OIDC2.

Data Complexity QA

Next, we quantify the Data Complexity QA on each AuthN DP. The average complexity of the data models, included in the $Complexity_{Data}$ metric, can also be equivalently expressed as follows:

$$\begin{aligned} avg_{m \in M_D}(Complexity_{DM}(m)) = & \quad avg_{m \in M_D}(|F_m| + |Ref_m| + |TF_m| + |R_m|) \\ = & \quad \frac{|F| + |Ref| + |TF| + |R|}{|M_D|} \end{aligned}$$

where F, Ref, TF, and R are the set of fields, references, transformation fields, and data rules across all data models, and M_D the set of all data models (see table 6.2).

Thus, we can simplify $Complexity_{Data}$ as follows:

$$Complexity_{Data} = |M_D| * avg_{m \in M_D}(Complexity_{DM}(m)) = |F| + |Ref| + |TF| + |R|$$

Now, we can quantify the Data Complexity QA easily on each AuthN DP:

$$\begin{aligned} Complexity_{Data}(OIDC1) &= |F_{OIDC1}| + |Ref_{OIDC1}| + |TF_{OIDC1}| + |R_{OIDC1}| \\ &= 65 + 16 + 1 + 6 = 88 \\ Complexity_{Data}(OIDC2) &= |F_{OIDC2}| + |Ref_{OIDC2}| + |TF_{OIDC2}| + |R_{OIDC2}| \\ &= 64 + 15 + 2 + 7 = 88 \\ Complexity_{Data}(PBA) &= |F_{PBA}| + |Ref_{PBA}| + |TF_{PBA}| + |R_{PBA}| \\ &= 25 + 13 + 3 + 8 = 49 \end{aligned}$$

These scores indicate that OIDC1 and OIDC2 have an equal level of Data Complexity, and PBA has the lowest Data Complexity level out of all.

Behavioural Complexity QA

We move on to quantify the Behavioural Complexity of these AuthN DPs. Again, we can simplify the average complexity of the behavioural models, included in the Behavioural Complexity metric to the total number of steps across each behavioural model (see table 6.3). Thus, we can simplify $Complexity_{Behavioural}$ as follows:

$$Complexity_{Behavioural} = |Steps| + |E|$$

This metric yields the following scores:

$$\begin{aligned} Complexity_{Behavioural}(OIDC1) &= |Steps_{OIDC1}| + |E_{OIDC1}| \\ &= 77 + 17 = 94 \\ Complexity_{Behavioural}(OIDC2) &= |Steps_{OIDC2}| + |E_{OIDC2}| \\ &= 63 + 17 = 90 \\ Complexity_{Behavioural}(PBA) &= |Steps_{PBA}| + |E_{PBA}| \end{aligned}$$

$$= 117 + 9 = 126$$

We see that PBA has the highest score for Behavioural Complexity out of the three, followed by OIDC1 and OIDC2.

Structural Complexity QA

At last, we can quantify the Structural Complexity QA, concluding the third layer of the Complexity QP abstraction hierarchy (see figure 5.2). As all three AuthN DPs have only one architectural model in their Structural View (see table 6.4), the metric is simplified as follows:

$$Complexity_{Structural} = |AME| + |Relationships|$$

Applying this metric to the AuthN DPs yields the following results:

$$\begin{aligned} Complexity_{Structural}(OIDC1) &= |AME_{OIDC1}| + |Relationships_{OIDC1}| \\ &= 21 + 30 = 51 \\ Complexity_{Structural}(OIDC2) &= |AME_{OIDC2}| + |Relationships_{OIDC2}| \\ &= 20 + 30 = 50 \\ Complexity_{Structural}(PBA) &= |AME_{PBA}| + |Relationships_{PBA}| \\ &= 17 + 25 = 42 \end{aligned}$$

These scores rank OIDC1 on top, directly followed by OIDC2 and at the end PBA.

Design Complexity QA

Using the scores from the quantification of the lowest abstraction level, we can now quantify the Design and Implementation Complexity QAs. For demonstration purposes, we use equal weights ($a_1 = a_2 = a_3 = a_4 = \frac{1}{4}$) for the Design Complexity metric, under the assumption that all Views of an AuthN DP are equally relevant concerning the Design Complexity of an AuthN DP. However, architects can choose different weights, depending on the View they are more interested in. Using these weights, the scores result as follows:

$$Complexity_{Design}(OIDC1) = \frac{1}{4} \times (35 + 88 + 94 + 51) = 67$$

$$Complexity_{Design}(OIDC2) = \frac{1}{4} \times (34 + 88 + 90 + 50) = 65.5$$

$$Complexity_{Design}(PBA) = \frac{1}{4} \times (55 + 49 + 126 + 42) = 68$$

Implementation Complexity QA

Concerning the weights for the Implementation Complexity metric, we consider the Data and Behaviuoral Complexities to have about equal impact on the Implementation Complexity ($b_1 = b_2 = \frac{1}{4}$). As the Structural View represents an implementation example of the AuthN DP [Lam24], we weight it with $b_3 = \frac{1}{2}$). With that in mind, the calculation scores for Implementation Complexity QA result as follows:

$$Complexity_{Implementation}(OIDC1) = \frac{1}{4} \times 88 + \frac{1}{4} \times 94 + \frac{1}{2} \times 51 = 71$$

$$Complexity_{Implementation}(OIDC2) = \frac{1}{4} \times 88 + \frac{1}{4} \times 90 + \frac{1}{2} \times 50 = 69.5$$

$$Complexity_{Implementation}(PBA) = \frac{1}{4} \times 49 + \frac{1}{4} \times 126 + \frac{1}{2} \times 42 = 64.75$$

Complexity QP

As Design and Implementation are both very important aspects, we balance the final Complexity score using equal weights ($w_1 = w_2 = \frac{1}{2}$):

$$Complexity(OIDC1) = \frac{1}{2} \times 67 + \frac{1}{2} \times 71 = 69$$

$$Complexity(OIDC2) = \frac{1}{2} \times 65.5 + \frac{1}{2} \times 69.5 = 67.5$$

$$Complexity(PBA) = \frac{1}{2} \times 68 + \frac{1}{2} \times 64.75 = 66.375$$

These final scores suggest that OIDC1 has a higher overall Complexity than the other two, and PBA the lowest Complexity of all.

	OIDC1	OIDC2	PBA
|Roles|	10	11	14
|Policy_Points|	11	8	17
|Relationships|	14	15	26

Table 6.1: Basis Data for $Complexity_{Conceptual}$ metric for OIDC1, OIDC2 and PBA AuthN DPs

	OIDC1	OIDC2	PBA
|F|	65	64	25
|Ref|	16	15	13
|TF|	1	2	3
|R|	6	7	8

Table 6.2: Basis Data for $Complexity_{Data}$ metric for OIDC1, OIDC2 and PBA AuthN DPs

	OIDC1	OIDC2	PBA
|Steps|	77	63	117
|E|	17	17	9

Table 6.3: Basis Data for $Complexity_{Behavioural}$ metric for OIDC1, OIDC2 and PBA AuthN DPs

	OIDC1	OIDC2	PBA
|AME|	21	20	17
|Relationships|	30	30	25

Table 6.4: Basis Data for $Complexity_{Structural}$ metric for OIDC1, OIDC2 and PBA AuthN DPs

7 Discussion

In this chapter, we discuss the findings that address the research questions posed in this thesis.

> **RQ1:** *How can AuthN DPs be described through characteristic properties in a differentiated manner?*

> **RQ2:** *How can AuthN DPs be meaningfully compared with each other using such characteristic properties?*

Following the DSR methodology outlined in Section 4.2, we defined the objective of our solution as making AuthN DPs comparable with one another, specifically aimed at answering these research questions. To achieve this objective, we formulated two specific requirements intended to ensure that AuthN DPs can be effectively compared when these requirements are fulfilled. In the following, we discuss how our results meet these requirements.

- **R1: Universal -** A property should be applicable to all AuthN DPs.

The challenge in proving this requirement lies in the necessity to test the property against every possible AuthN DP. Since it is impractical to evaluate all (existing and potentially emerging) AuthN DPs, following the falsifiability principle, we have to operate under the assumption that a property is universal until it is demonstrated to be false in a specific instance. Up until now, we have not identified any contradicting examples to this requirement, which supports the idea that the defined properties can be consistently applied to a wide range of AuthN DPs. While we aim to define properties that are universally applicable, we must remain open to the possibility that new AuthN DPs may emerge or that existing ones may eventually contradict this requirement. Until such evidence arises, these properties may be effectively utilized for comparison.

- **R2: Measureable -** A property should either be categorizable on a nominal scale or quantifiable by measurements on at least an ordinal scale.

The PCM metamodel we developed specifies that properties of AuthN DPs can be classified into Categorical Properties and QPs, with all defined properties conforming to this structure. By definition, Categorical Properties take on predefined and fixed values that categorize AuthN DPs on a nominal scale, thus allowing for straightforward

comparisons. On the other hand, the abstraction hierarchy of QPs defined in the PCM enables a more granular assessment of QPs, thereby facilitating quantitative evaluations of AuthN DPs. The example metrics we provide for QAs and QPs demonstrate that these properties are measurable on at least an ordinal scale, thus making them suitable for comparison and ranking of AuthN DPs. Furthermore, since QPs represent specific qualitative aspects of AuthN DPs, comparisons based on these properties are also considered relevant and meaningful.

7.1 Limitations

While the findings of this thesis offer valuable insights for software architects and serve as a basis for quantitative evaluations of AuthN DPs, there are limitations that must be acknowledged.

A key limitation of metric-based evaluations lies in the necessity for consistent data gathering and storage practices, which is essential for a meaningful application of metrics. For example, to apply a counting metric of threats and attacks against an AuthN DP solution, one must first have full knowledge over all such possible threats. Otherwise, the application of this metric is impractical and hinders reproducibility.

Additionally, the metrics defined in this thesis are only provided as examples to demonstrate that QPs are quantifiable. Their suitability, applicability and reproducibility are yet to be evaluated, and this is suggested as future work.

Moreover, it should also be noted that the proposed properties are not intended to provide an exhaustive catalogue. Instead, they serve as a starting point for further research on quantitative evaluations of AuthN DPs that facilitate their comparision.

For instance, we also considered Maintainability as a relevant QP of AuthN DPs. However, due to time constraints, we were unable to fully model it conform to the PCM and provide example metrics for its quantification. Thus, we suggest its inclusion as future work. Furthermore, Costs may also be considered as a QP for the evaluation of AuthN DPs. However, since Costs are rather relevant for AuthN DPs for the enterprise layer, which were not in the scope of this thesis, they were not included in our results.

Among the quantifiable properties, this thesis only considers those that describe a quality aspect of AuthN DPs. There are, however, other quantifiable properties that may also be relevant for comparisions of AuthN DPs. We identified *Credential Validity Period* as an example of such properties. As AuthN DPs can be ordered based on the duration of the validity period for the credentials utilised within them, this property is defined on at least an ordinal scale (e.g. with values “short-term”, “medium-term” and “long-term”), and as such is a quantifiable property. Credential Validity Period does not describe any quality aspect of AuthN DPs, but represents rather a design choice that depends on the context of the application in which an AuthN DP is deployed. As such, a CBRS could utilise this property to narrow down suitable AuthN DPs for specific technical and environmental factors.

8 Conclusion

This chapter summarizes the findings of this thesis, and outlines opportunities for future research.

8.1 Summary

This thesis addresses the need for reusable knowledge to effectively evaluate and compare AuthN DPs with. We proposed a property classification approach to model AuthN DPs through characteristic properties. To this end, we developed the PCM metamodel, which outlines the structure of property models for AuthN DPs, and we defined properties that conform to this metamodel.

The Categorized Properties allow for straightforward comparisions as they take on predefined and fixed values that categorize AuthN DPs on a nominal scale. For example, Compliance categorizes AuthN DPs based on the regulatory frameworks they adhere to. A comparision based on this property helps architects identify which AuthN DPs align with the legal and regulatory requirements of their application.

In contrast, QPs describe specific qualitative aspects of AuthN DPs. Their abstraction hierarchies specified by the PCM metamodel not only allow for a better understanding of the quality of AuthN DPs, but also provide a foundation for quantitative evaluations of AuthN DPs. By enabling a systematic assesment at each level of abstraction, these hierarchies support the development of metrics that quantify QAs and QPs, as we illustrated with the provided example metrics. We demonstrated the applicability of this metric-based quantification approach by applying example metrics for a QP on three different AuthN DPs, which ultimately resulted in a ranking of these AuthN DPs.

In conclusion, we provide a systematic approach to describing, evaluating, and comparing AuthN DPs. The defined Categorized Properties can be utilized for straightforward comparisions between AuthN DPs, which may offer valuable insights for achitects and stakeholders. Meanwhile, the defined QP abstraction hierarchies can be used for systematic, metric-based evaluations of AuthN DPs. Through these evaluations, AuthN DPs can be easily and effectively compared with each other.

8.2 Future Work

A significant area for future work is the demonstration and evaluation of the applicability of the defined QPs and their abstraction hierarchies for metric-based evaluations and comparision of AuthN DPs. This could involve case studies or comparative analyses that illustrate how the defined QPs and QAs can be utilized in diverse contexts. In addition, the suitability, applicability and reproducibility of their example metrics should be evaluated.

Furthermore, future work could focus on extending the AuthN DP property catalog. While this thesis has established a starting point, there other QPs such as Maintainability that could provide valuable insights on the quality of AuthN DPs.

Another important area for future work involves the implementation of a knowledge base to systematically catalog the defined properties and metrics. This knowledge base could serve as a resource for software architects and researchers.

Finally, given that the object of this thesis were AuthN DPs, the properties we defined were focused on AuthN DPs and not SPs. However, QPs such as Complexity may be generally applicable to all SPs. Future work could investigate the applicability of these QPs for the SP hierarchy abstraction level.

Bibliography

[21a] *A04:2021 – Insecure Design*. Accessed: 2025-09-25. OWASP Foundation. 2021. URL: https://owasp.org/Top10/A04_2021-Insecure_Design/ (cit. on p. 1).

[21b] *A07:2021 – Identification and Authentication Failures*. Accessed: 2025-09-25. OWASP Foundation. 2021. URL: https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/ (cit. on p. 1).

[23] *ISO/IEC 25010:2023 — Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Product quality model*. Accessed: 2025-09-28. 2023. URL: https://www.iso.org/standard/78176.html (cit. on pp. 20, 21).

[25a] *Common Vulnerability Scoring System (CVSS) SIG*. Accessed: 2025-09-28. FIRST. 2025. URL: https://www.first.org/cvss/ (cit. on p. 27).

[25b] *OWASP Authentication Cheat Sheet*. Accessed: 2025-09-28. OWASP Foundation. 2025. URL: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html (cit. on p. 5).

[25c] *OWASP Forgot Password Cheat Sheet*. Accessed: 2025-09-28. OWASP Foundation. 2025. URL: https://cheatsheetseries.owasp.org/cheatsheets/Forgot_Password_Cheat_Sheet.html (cit. on p. 8).

[AAM15] S. N. Abdulkader, A. Atia, and M.-S. M. Mostafa. "Authentication systems: Principles and threats." In: *Computer and Information Science* 8.3 (2015), p. 155 (cit. on p. 6).

[Bar24] Barracuda Networks. *Cybernomics 101: Uncovering the Financial Forces Driving Cyberattacks*. Accessed: 2025-09-25. 2024. URL: https://assets.barracuda.com/assets/docs/dms/barracuda-cybernomics-report.pdf (cit. on p. 1).

[BCR94] V. R. Basili, G. Caldiera, and D. H. Rombach. "The Goal Question Metric Approach." In: vol. I. John Wiley & Sons, 1994 (cit. on p. 17).

[CDW04] A. Conklin, G. Dietrich, and D. Walz. "Password-based authentication: a system perspective." In: *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*. 2004. DOI: 10.1109/HICSS.2004.1265412 (cit. on p. 6).

[CVE24] CVE Program. *Metrics – CVE: Common Vulnerabilities and Exposures*. Accessed: 2025-09-25. 2024. URL: https://www.cve.org/about/Metrics (cit. on p. 1).

[Dal18] S. Dalati. "Measurement and Measurement Scales." In: *Modernizing the Academic Teaching and Research Environment: Methodologies and Cases in Business Research*. Ed. by J. Marx Gómez and S. Mouselli. Cham: Springer International Publishing, 2018, pp. 79–96. ISBN: 978-3-319-74173-4. DOI: 10.1007/978-3-319-74173-4_5. URL: https://doi.org/10.1007/978-3-319-74173-4_5 (cit. on p. 8).

[Ele90] I. Electronics Engingeers. "IEEE Standard Glossary of Software Engineering Terminology." In: *Office* 121990 (1990), p. 84 (cit. on p. 8).

[Eng+20] E. Engström et al. "How software engineering research aligns with design science: a review." In: *Empirical Software Engineering* 25.4 (Apr. 2020), pp. 2630–2660. ISSN: 1573-7616. DOI: 10.1007/s10664-020-09818-7. URL: http://dx.doi.org/10.1007/s10664-020-09818-7 (cit. on p. 15).

[FHW22] S. Furnell, K. Helkala, and N. Woods. "Accessible authentication: Assessing the applicability for users with disabilities." In: *Computers & Security* 113 (2022), p. 102561. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2021.102561. URL: https://www.sciencedirect.com/science/article/pii/S0167404821003850 (cit. on p. 32).

[FKS17] D. Fett, R. Küsters, and G. Schmitz. "The Web SSO Standard OpenID Connect: In-depth Formal Security Analysis and Security Guidelines." In: *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. 2017, pp. 189–202. DOI: 10.1109/CSF.2017.20 (cit. on p. 6).

[Gam+95] E. Gamma et al. *Design Patterns*. Vol. 47. Addison Wesley Professional Computing Series February. 1995, pp. 1–429 (cit. on p. 7).

[GFN+17] P. A. Grassi, J. L. Fenton, E. M. Newton, et al. *Digital Identity Guidelines: Authentication and Lifecycle Management (SP 800-63B)*. Special Publication 800-63B. Accessed: 2025-09-28. National Institute of Standards and Technology, 2017. DOI: 10.6028/NIST.SP.800-63b. URL: https://pages.nist.gov/800-63-3/sp800-63b.html (cit. on p. 31).

[Hev+04] A. R. Hevner et al. "Design Science in Information Systems Research." In: *MIS Quarterly* 28.1 (2004), pp. 75–105. ISSN: 02767783. URL: http://www.jstor.org/stable/25148625 (visited on 09/30/2025) (cit. on p. 15).

[Hey+07] T. Heyman et al. "An Analysis of the Security Patterns Landscape." In: *Third International Workshop on Software Engineering for Secure Systems (SESS'07: ICSE Workshops 2007)*. 2007, pp. 3–3. DOI: 10.1109/SESS.2007.4 (cit. on pp. 7, 11).

[Joi20] Joint Task Force. *Control Baselines for Information Systems and Organizations (SP 800-53B)*. Special Publication 800-53B. Accessed: 2025-09-28. National Institute of Standards and Technology, 2020. DOI: 10.6028/NIST.SP.800-53B. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53B.pdf (cit. on p. 7).

[KD19] K. Kishore Kumar and A. M. Deepthishree. "Comparison-Based Analysis of Different Authenticators." In: *ICCCE 2018*. Ed. by A. Kumar and S. Mozar. Singapore: Springer Singapore, 2019, pp. 583–590. ISBN: 978-981-13-0212-1 (cit. on p. 28).

[Lam24] D. Lammers. "Conception of a Security Design Pattern Catalog for Constraint-based Recommender Systems." Accessed: 2025-09-25. Master's Thesis. Rheinisch-Westfälische Technische Hochschule Aachen, Dec. 12, 2024. URL: https://swc.rwth-aachen.de/theses/conception-of-a-security-design-pattern-catalog-for-constraint-based-recommender-systems/ (cit. on pp. 1–3, 8, 11, 12, 15, 17, 18, 24, 25, 35, 38).

[Pef+07] K. Peffers et al. "A design science research methodology for information systems research." In: *Journal of Management Information Systems* 24 (Jan. 2007), pp. 45–77 (cit. on p. 15).

[RES20] P. Runeson, E. Engström, and M.-A. Storey. "The Design Science Paradigm as a Frame for Empirical Software Engineering." In: *Contemporary Empirical Methods in Software Engineering*. Ed. by M. Felderer and G. H. Travassos. Cham: Springer International Publishing, 2020, pp. 127–147. ISBN: 978-3-030-32489-6. DOI: 10.1007/978-3-030-32489-6_5. URL: https://doi.org/10.1007/978-3-030-32489-6_5 (cit. on p. 15).

[RW11] N. Rozanski and E. Woods. *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. 2nd ed. Addison-Wesley Professional, 2011. ISBN: 032171833X (cit. on pp. 6, 7).

[SLL25] A. R. Sabau, D. Lammers, and H. Lichter. *SecuRe – An Approach to Recommending Security Design Patterns*. 2025. arXiv: 2501.14973 [cs.SE]. URL: https://arxiv.org/abs/2501.14973 (cit. on pp. 3, 7–9, 11, 12, 15).

[Sof23] Software Construction Group. *SCAM – Security-Centric Architecture Modelling*. Accessed: 2025-09-25. 2023. URL: https://swc.rwth-aachen.de/research/projects/scam-security-centric-architecture-modelling/ (cit. on pp. 1, 15).

[Sta24] Statista. *Forecast: Cost of Cybercrime Worldwide*. Accessed: 2025-09-25. 2024. URL: https://www.statista.com/forecasts/1280009/cost-cybercrime-worldwide (cit. on p. 1).

[Sto+17] M.-A. Storey et al. "Using a Visual Abstract as a Lens for Communicating and Promoting Design Science Research in Software Engineering." In: *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 2017, pp. 181–186. DOI: 10.1109/ESEM.2017.28 (cit. on p. 15).

[UMA25] UMA Technology. *Latency Analysis for API Authentication Flows Under Compliance Zones*. Accessed: 2025-09-28. 2025. URL: https://umatechnology.org/latency-analysis-for-api-auth-flows-under-compliance-zones/ (cit. on p. 30).