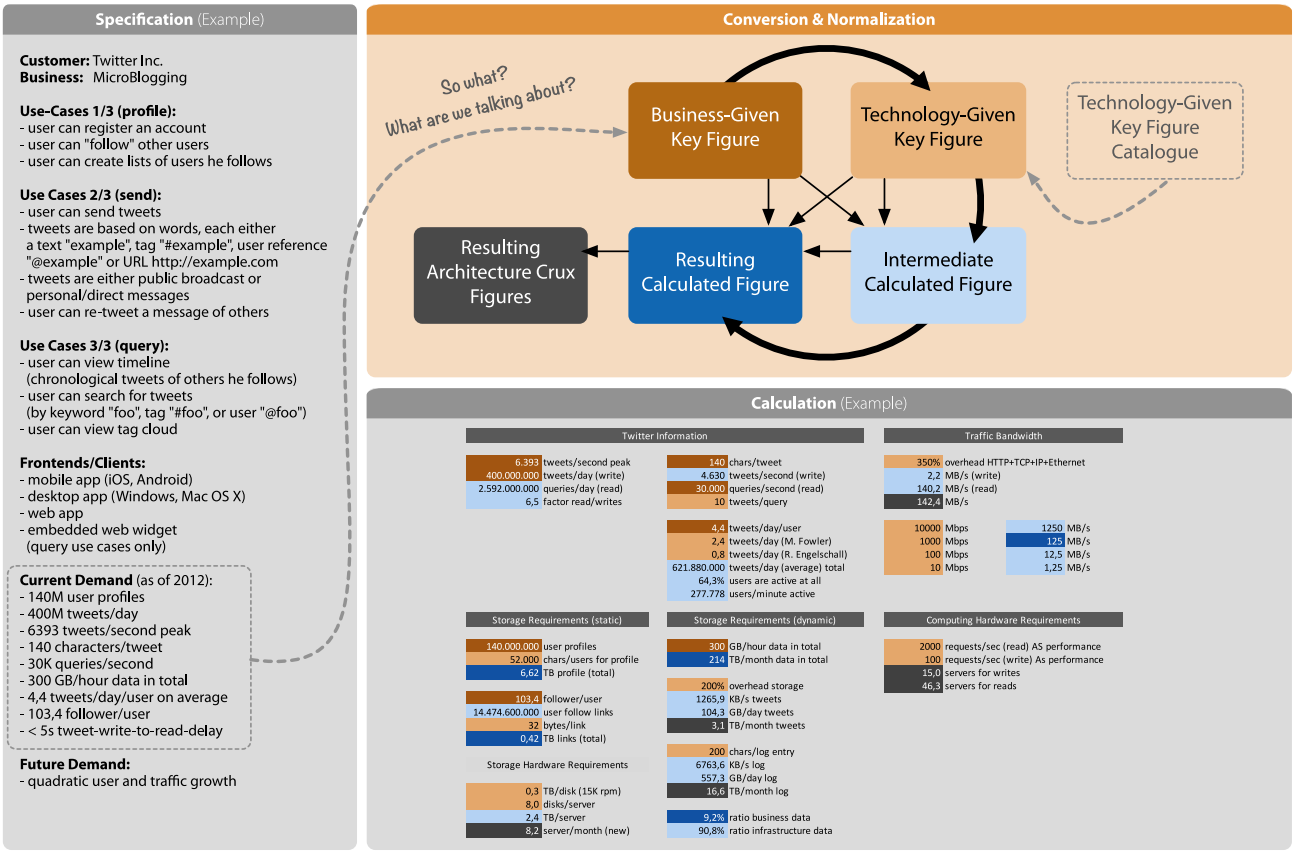


Software Engineering in Industrial Practice (SEIP)

Dr. Ralf S. Engelschall

Back of the Envelope Calculation



To get a better "feeling" for the scope and the degree of difficulty of an architecture to be developed, it is a good idea to do a **Back of the Envelope Calculation** ("rough calculation"). The method is as follows: in a spreadsheet, a two-column table is created in which the first column contains the number and the second column contains the unit.

Now, in the first step, the **Business-Given Key Figures**, i.e., the technically known numbers, are entered into the table as the first rows. They get the first color for differentiation.

Because these numbers usually do not tell enough, in a second step, different **Technology-Given Key Figures** are entered as rows. These may be taken from existing catalogs or are available from your own experience. They are given the second color for differentiation and serve above all as comparative figures to the **Business-Given Key Figures**.

In the third step, both the **Business-Given Key Figures** and the **Technology-Given Key Figures** are converted unit-consistently and compared to each other. The intermediate results, called **Intermediate Calculated Figures**, are spreadsheet cells with formulas, which get the third color to distinguish them.

Whenever an **Intermediate Calculated Figure** (or possibly already a **Business-Given Key Figure** or a **Technology-Given Key Figure**) provides a decisive hint or insight, you change the row to the fourth color. If this insight has potential relevance for the subsequent architecture (and thus represents a key point), the row is changed to the fifth color, which shows the **Resulting Architecture Crux Figure**.

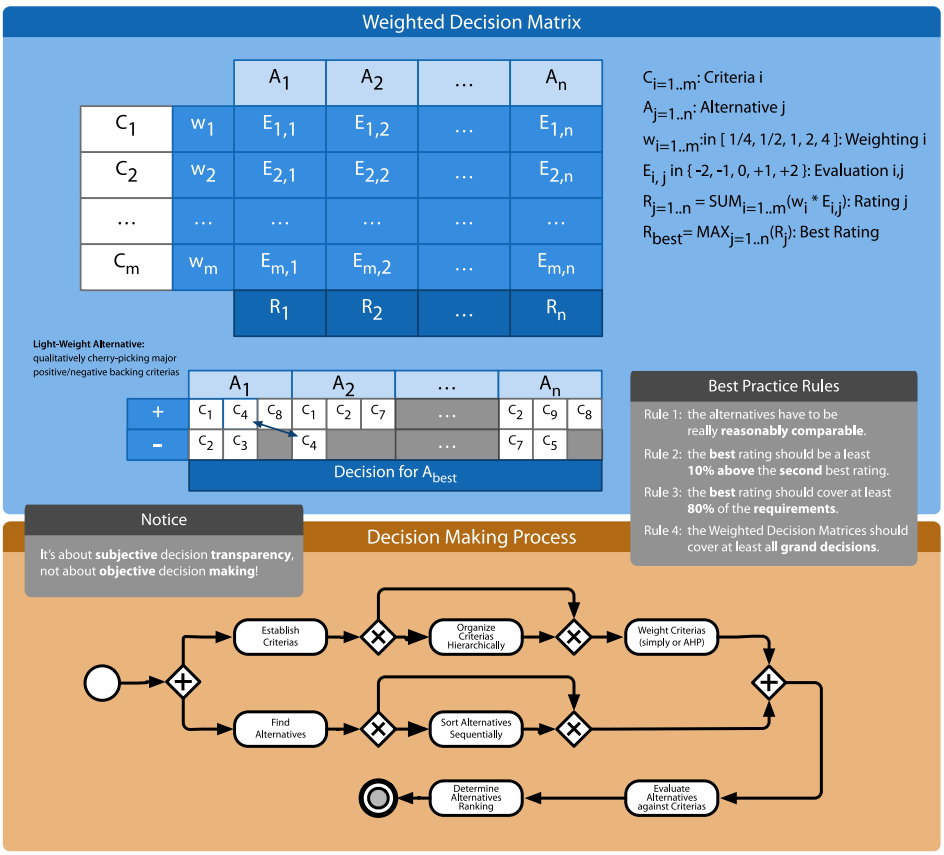
Afterwards, you can optionally bundle the different rows into logical groups in the spreadsheet to make the spreadsheet clearer.

Questions

? What method can be used to get a better "feel" for the scope and difficulty of an architecture to be developed?

Intellectual Content: Version 1.1.0 (2021-11-23), Authored 2011-2021 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.3.0 (2022-10-10), Copyright © 2012-2022 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

Weighted Decision Matrix



Standard Criteria Catalogs

Software Selection:
Suitable Functionality
Available Usage Examples
Reasonable Documentation
Reasonable Support
Permissive License
Long-Term Release Track Record
Current Market Momentum

Software Selection (Open Source):
+ Clean Source Code
+ Clean Build Process
+ Open Source License

Software Selection (Library):
+ Non-Invasive Programming Model
+ Orthogonal Application Programming Interface
+ Minimum/No Dependencies
+ Non-Copyleft Open Source License

Software Selection (Framework):
+ Orthogonal Application Programming Interface
+ Adequate Dependencies
+ Non-Overlapping Scope
+ Non-Copyleft Open Source License

Software Selection (Tool):
+ Clean Deployment Procedure
+ Pleasant Command-Line Interface

Software Selection (Application):
+ Clean Deployment Procedure
+ Pleasant Graphical User Interface

Software Architecture Evaluation:
Meets Functional Requirements
Meets Non-Functional Requirements
Adequate Technology Overhead
Single Dependency Direction
Distance to State of the Art ("modern")
Distance to Most Simple Approach ("adequate")
Distance to Mainstream Approach ("mainstream")
Documented Architecture Decisions ("rationales")
Documented Architecture Views
Documented Architecture Perspectives (NFR)

In order to make qualitative decisions transparently and comprehensibly (but not necessarily objectively) and at the same time to document the decision-making process, one can apply the method of the **Weighted Decision Matrix**.

The prerequisite is that the decision to be made is the choice of one of many alternatives. These are entered in a spreadsheet as columns. Optionally, the alternatives can be put into a meaningful order.

Then one specifies different criteria, which are to be used for the decision. The goal is to distinguish the alternatives with as few criteria as possible. Each criterion is given a weighting. Optionally, the criteria can be grouped into a hierarchy and the groups into a meaningful order.

Subsequently, one determines a weighting for each criterion, which indicates how strongly the criterion is considered in the decision-making process.

Now all alternatives are evaluated against all criteria. With few criteria and many alternatives, one can evaluate all criteria per alternative. If there are many criteria, it is advisable to evaluate all alternatives per criterion. As an evaluation scale, it is advisable to use -2, -1, 0, +1, +2, in order to have a middle (0), positive/negative evaluations (+1, -1) and positive/negative superlatives (+2, -2).

Finally, for each alternative, the product sum of the criterion weighting column and the alternative rating column is calculated. The decision is then made for the alternative with the maximum amount in the product sum.

Questions

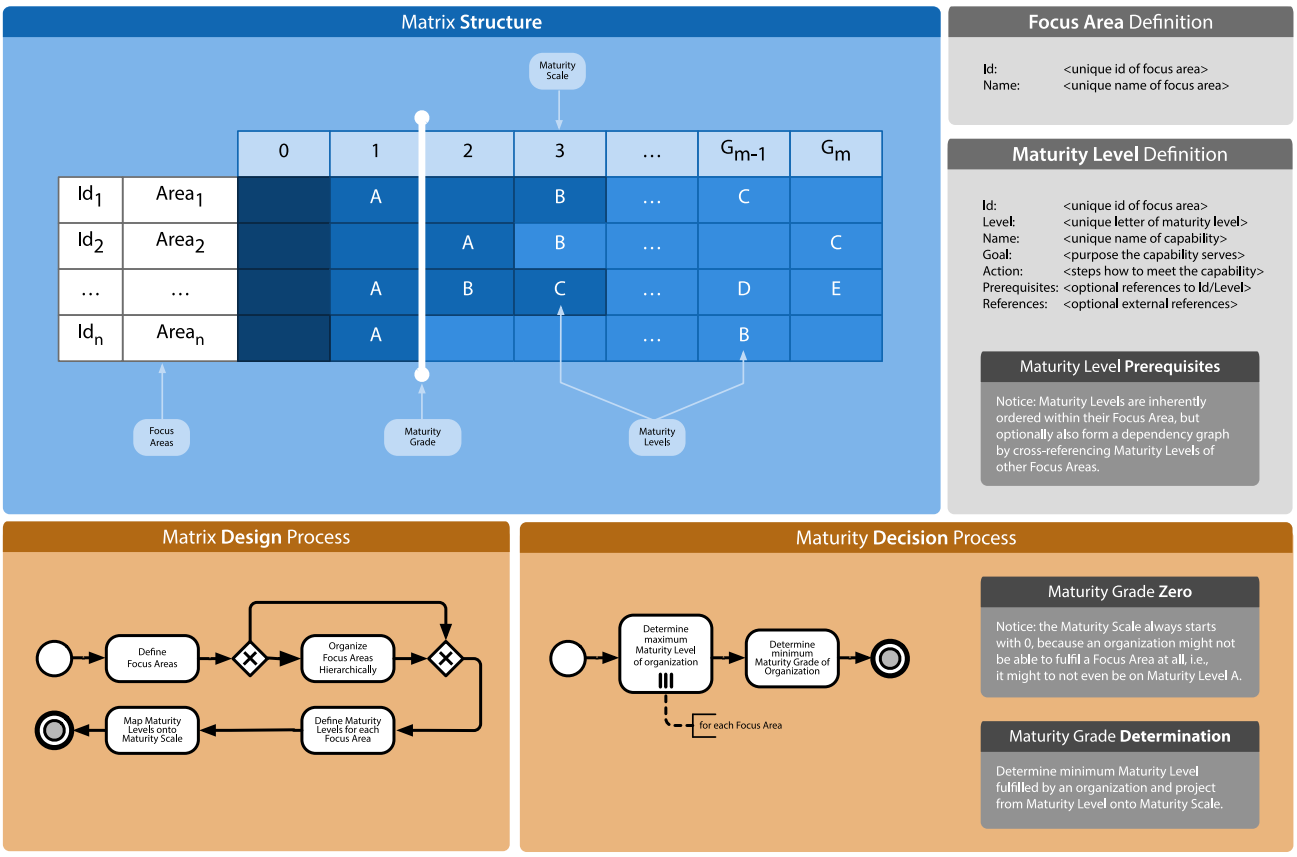
? How can the qualitative decision to choose one of many alternatives be made transparently and comprehensibly and documented at the same time?

Intellectual Content: Version 1.1.1 (2022-05-20), Authored 2018-2022 by Dr. Ralf S. Engelschall, Graphical Illustration: Version 1.1.0 (2021-05-26), Copyright © 2018-2021 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved. Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Focus Area Maturity Model

TECHNISCHE UNIVERSITÄT MÜNCHEN



The Focus Area Maturity Model (FAMM) is a method to assess the maturity of an organization with respect to a specific topic area.

The structure of a FAMM is a matrix of horizontal Focus Areas and their Maturity Levels and possible vertical Maturity Grades on a Maturity Scale. Per Focus Area there can be one or any number of Maturity Levels and their positions on the Maturity Scale are based on the importance of the Focus Areas and the relationship between the Focus Areas and their Maturity Levels. This matrix is designed in a first step for a topic area and is then fixed.

In order to determine the Maturity Level of an organization, determine for each Focus Area the maximum Maturity Level the organization fulfills. The Maturity Level of the organization is then derived from the minimum Maturity Level across all Focus Areas and the projection of this Maturity Level onto the Maturity Scale.

Since the Maturity Levels should only ever be positioned in the matrix above Maturity Grade 0, in the worst case, an organization has a Maturity Grade 0 if it does not fulfill a Focus Area at all.

Questions

? From whom can you determine the maturity level with the Focus Area Maturity Model (FAMM)?

Intellectual Content: Version 1.0.1 (2023-12-01), Authored 2023-2023 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.0.0 (2023-05-31), Copyright © 2022 Dr. Ralf S. Engelschall (http://engelschall.com) - All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contents only.

Big Picture (8-D)

Vote is a portable mobile-first designed application for easily performing anonymous online votings within a small group of people to figure out their opinions or moods.

Votings are created in advance, executed at a certain time, conducted by the users, and then finally reported.

1 Elevator Pitch (RE, D)

Name, Purpose, Motivation, Actors, Devices.

Rationale: Roughly describe the purpose and primary motivation.

Format: Prose Abstract

2 Crux Flash (RE, D)

Functionality, Cruxes.

Rationale: Roughly describe the functionality and the cruxes.

Format: Prose Abstract

Votings can be quickly accessed by QR-code or URL and are based on one or more questions and corresponding multiple-choice-based answers.

Votings are interactively conducted, and answers are received and reported either asynchronously in batches (offline voting) or even synchronously in real-time (online voting).

Actor Role	Use-Case
User	Register Account
	Recover Account
	Configure Account
	Login Account
	Logout Account
Author	Create Voting
	Grant Voting Access
Supervisor	Execute Voting
	Enable Question
	Display Result
Voter	Vote Question
	Display Result

3 Customer Journey (UX, D)

Actor Roles, Use-Cases.

Rationale: Sketch the customer journey through major use-cases.

Format: 2xN Table or UML UC Diag.

4 Dialog Storyboard (UX, D)

Dialogs, Interaction, Control Flow.

Rationale: Illustrate the major user interface dialogs (or dialog types).

Format: Wireframe Graph Diagram



Quality	Expectation
Cross-Platform Client	yes
Non-Cleartext Password Storage	yes
Minimum Concurrent Voters (people)	50
Maximum Display Result Latency (sec)	1

5 Quality Requirements (RE, T)

Qualities, Expectations.

Rationale: List requirements on the major non-functional qualities.

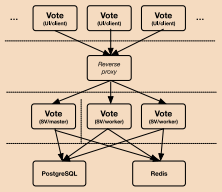
Format: 2xN Table

6 System Architecture (SY, T)

Actors, Systems, Zones, Programs.

Rationale: Illustrate the major system architecture components.

Format: Boxes'n'Lines Diagram



Aspect	Amount	Size	Total Size	Unit
Account Data	10.000	256	2.560.000	B
Voting Data	10.000	1.024	10.240.000	B
Server RAM Usage	200	20	4.000	MB

8 Sizing Sketch (SW, T)

Aspects, Amounts, Sizes, Total Sizes, Units.

Rationale: Sketch the sizing of major entities and system parts.

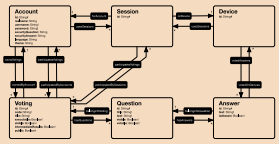
Format: 5xN Table

7 Data Model (SW, T)

Entities, Relationships.

Rationale: Model major data entities and their relationships.

Format: UML Class Diagram



(Example) (Method) (Example)

(1) Creation Step (RE) Requirements Engineering (UX) User Experience (SY) Systems Architecture (SW) Software Architecture (D) Domain Scope (T) Technology Scope

Intellectual Content: Version 1.0.3 (2020-01-20), Authored 2015-2020 by Dr. Ralf S. Engelschall (http://engelschall.com)
Graphical Illustration: Version 1.0.3 (2020-01-20), Copyright © 2015-2020 Dr. Ralf S. Engelschall (http://engelschall.com) – All rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

In order to understand a Business Information System to be created at a very early stage, a so-called **Big Picture** should be created. This can be developed according to the method **8-D** of Dr. Ralf S. Engelschall, which consists of 8 dimensions (4 domain-specific and 4 technical ones).

In the first step, an **Elevator Pitch** is created, which is defined by **Name**, **Purpose**, **Actors** and **Devices** to give a rough overview of the solution. In the second step, an additional **Crux Flash** is created, which roughly describes the functionality and its main cruxes. Both steps usually consist of only a few prose sentences.

In the third step, a **Customer Journey** is outlined, which describes, via **Actor Roles** and **Use Cases**, which use cases the different actor roles experience. This usually consists of only a 2-column table.

In the fourth step, a **Dialog Storyboard** is illustrated, which is shown via **Dialogs**, **Interactions** and **Control Flow**, and which illustrates the user interface the application offers. This usually consists of a graph of **Wireframes** (intentionally very rough and fuzzy sketches of the dialogs).

In the fifth step, the **Quality Requirements** are listed via **Qualities** and **Expectations**, to show which expectations exist concerning non-functional quality properties. These are usually executed as a 2-column table.

In the sixth step, the **System Architecture** is illustrated, which via **Actors**, **Systems**, **Tiers/Areas** and **Programs** shows which main components will exist on the level of System Architecture. This usually consists of a "Boxes'n'Lines" diagram.

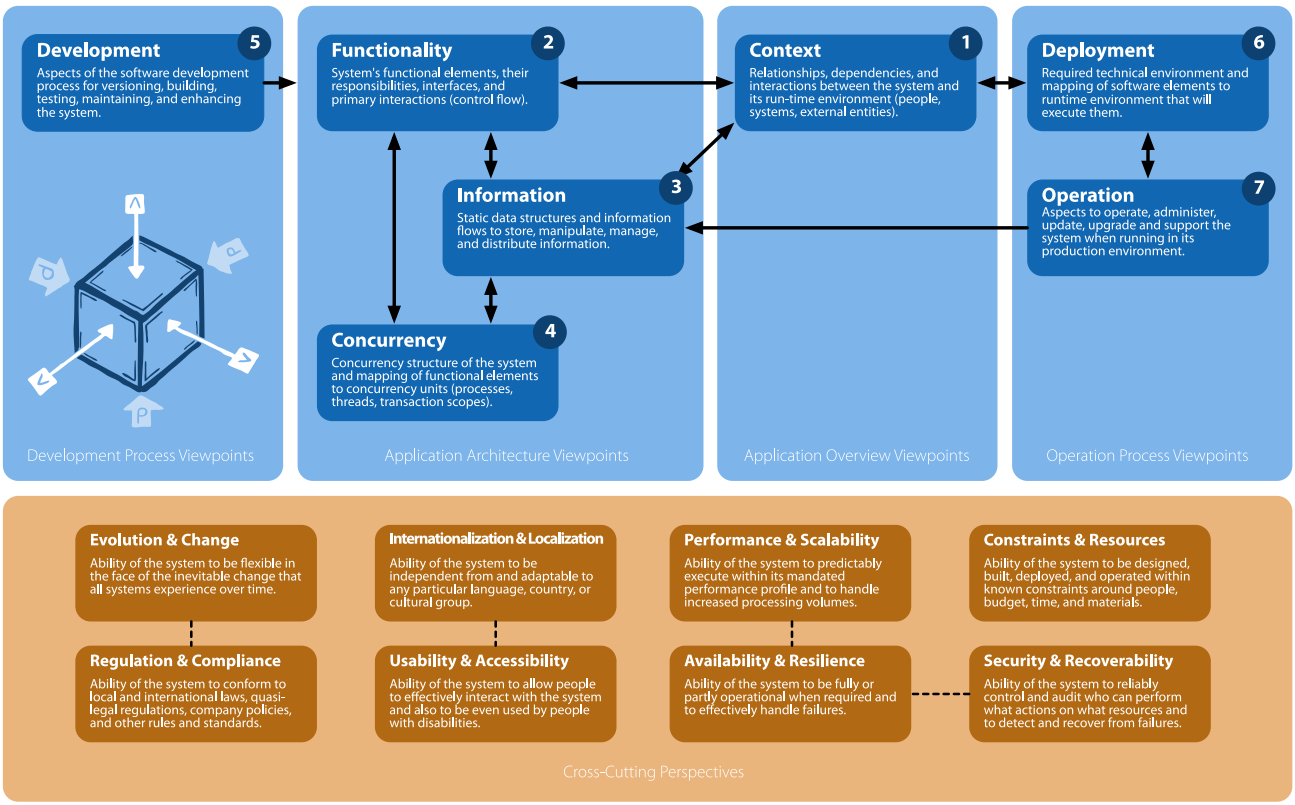
In the seventh step, the **Data Model** is modeled, which, via **Entities** and **Relationships**, shows which main domain-specific classes and relationships the data of the application has. This is usually represented as a "UML Class Diagram".

In the eighth and last step, a **Sizing Sketch** is created, which shows via **Aspects**, **Amounts**, **Sizes**, **Total Sizes** and **Units** which orders of magnitude (in the "worst case") are to be expected with the data and the system components. This usually consists of a 5-column table.

Questions

- What are the eight dimensions of the **8-D** model supposed to do?

Viewpoints & Perspectives



Intellectual Content: Version 1.0.4 (2019-09-15), Authored 2011-2019 by Dr. Ralf S. Engelschall, based on Nick Rozanski, Eoin Woods, "Software Systems Architecture", 2nd ed. 2011, ISBN 978-0-321-71833-4
Graphical Illustration: Version 1.0.0 (2018-09-15), Copyright © 2011-2018 Dr. Ralf S. Engelschall, http://engelschall.com/, All Rights Reserved.
Unauthorized Reproduction Prohibited, Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

In order to create an **Architecture Description** (in the German context usually called **IT-Konzept**) for an application, one documents methodically via **Viewpoints** and **Perspectives**.

The former are 2-dimensional diagrams with an explanation of a single specific fact. The latter are explanations of an overall issue, all of which are derived from **Non-Functional Requirements**.

How many **Viewpoints** and **Perspectives** actually need to be documented depends on the **Concern** of the **Stakeholders**! A standard set of **Viewpoints** and **Perspectives** is the following.

For the **Viewpoints** it makes sense to document for instance 7 particular **Views**: **Context**, **Functionality**, **Information**, **Concurrency**, **Development**, **Deployment** and **Operation**. The two viewpoints **Functionality** and **Information** are the two most important ones to document the architecture of an application. They should always be documented and, therefore, never omitted!

For the **Perspectives**, it makes sense to document for instance 8x2 particular **Aspects** which are based on common **Non-Functional Requirements**: **Evolution & Change**, **Regulation & Compliance**, **Internationalization & Localization**, **Usability & Accessibility**, **Performance & Scalability**, **Availability & Resilience**, **Constraints & Resources** and **Security & Recoverability**. For the **Perspectives** in practice one has to take into account all the relevant **Non-Functional Requirements** relevant to the project and must not limit oneself to just these usual ones.

Questions

? Which two **Viewpoints** in the **Architecture Description** of an application should always be documented?

Security Basics

Information Protection Requirements

Confidentiality Information and systems are available to authorized users only.	**Vertraulichkeit** (germ.) Informationen und Systeme stehen nur autorisierten Benutzern zur Verfügung.	
Integrity Correctness of information and functionality of systems is given.	**Integrität** (germ.) Die Korrektheit der Informationen und die Funktionsweise von Systemen ist gegeben.	
Availability Information and systems are available to authorized users without hindrance.	**Verfügbarkeit** (germ.) Informationen und Systeme stehen autorisierten Benutzen ungehindert zur Verfügung.	

Information Protection Assessment

	Confidentiality			Integrity & Availability
	Military:	Gov. (germ.):	Industry:	Recoverable within…
very high	top secret	streng geheim	(secret)	4 hours
high	secret	geheim	(strictly confidential)	1 day
normal	confidential	vertraulich	confidential	1 week
low	(restricted)	nur für den Dienstgebrauch	internal	2 week
none	(public)	(öffentlich)	public	N.A.

Privacy by Design Aspects

TRY Processing Transparency Full transparency of all person-related data processing actions and the rationales for the processing.	**DIF Differential Privacy** Introduced randomness into the results of queries on underlying confidential or person-related data to ensure ambiguity in the deduced underlying data.
CST Processing Consent Explicit consent from the data owner for the data processing, including the right to contradict the consent at any later time.	**STR Secure Storage** Secure storage of all person-related information to prevent data loss or exposure while the information is at rest.
MIN Data Minimization Minimization of the gathered, processed and stored person-related data.	**TRS Secure Transfer** Secure transfer of all person-related information to prevent data loss or exposure while the information is transferred.
PSN Data Pseudonymization Substitute person-related information with artificial identifiers (pseudonyms) while still keeping information for backtracking to the original information.	**ERA Right to Erasure** Ensure the data owners, at any time, have the right and operational means to lookup and optionally erase all their person-related information.
ANO Data Anonymization Substitute person-related information with artificial identifiers without keeping any information for backtracking to the original information.	**PRT Right to Portability** Ensure the data owners, at any time, have the right and operational means to lookup and optionally export all their person-related information.
AGG Data Aggregation Always report person-related statistical information only as aggregated numbers which each does not cover less than about 10 involved persons.	**NOT Breach Notification** Transparently provide notifications in case of any protection breach of confidential or person-related information.

Intellectual Content: Version 1.0.0 (2018-10-31). Authored 2018 by Dr. Ralf S. Engelschall.
Graphical Illustration: Version 1.0.0 (2018-10-31). Copyright © 2018 Dr. Ralf S. Engelschall <http://engelschall.com>. All Rights Reserved.
Unauthorized Reproduction Prohibited. Licenced to Technische Universität München (TUM) for reproduction in Computer Science lecture contents only.

In the case of protection requirements, a distinction is made between three requirements (**Information Protection Requirements**): **Confidentiality** ensures that information and systems are available to authorized users only; **Integrity** ensures the correctness of information and the functioning of systems; and **Availability** ensures that information and systems are freely available to authorized users.

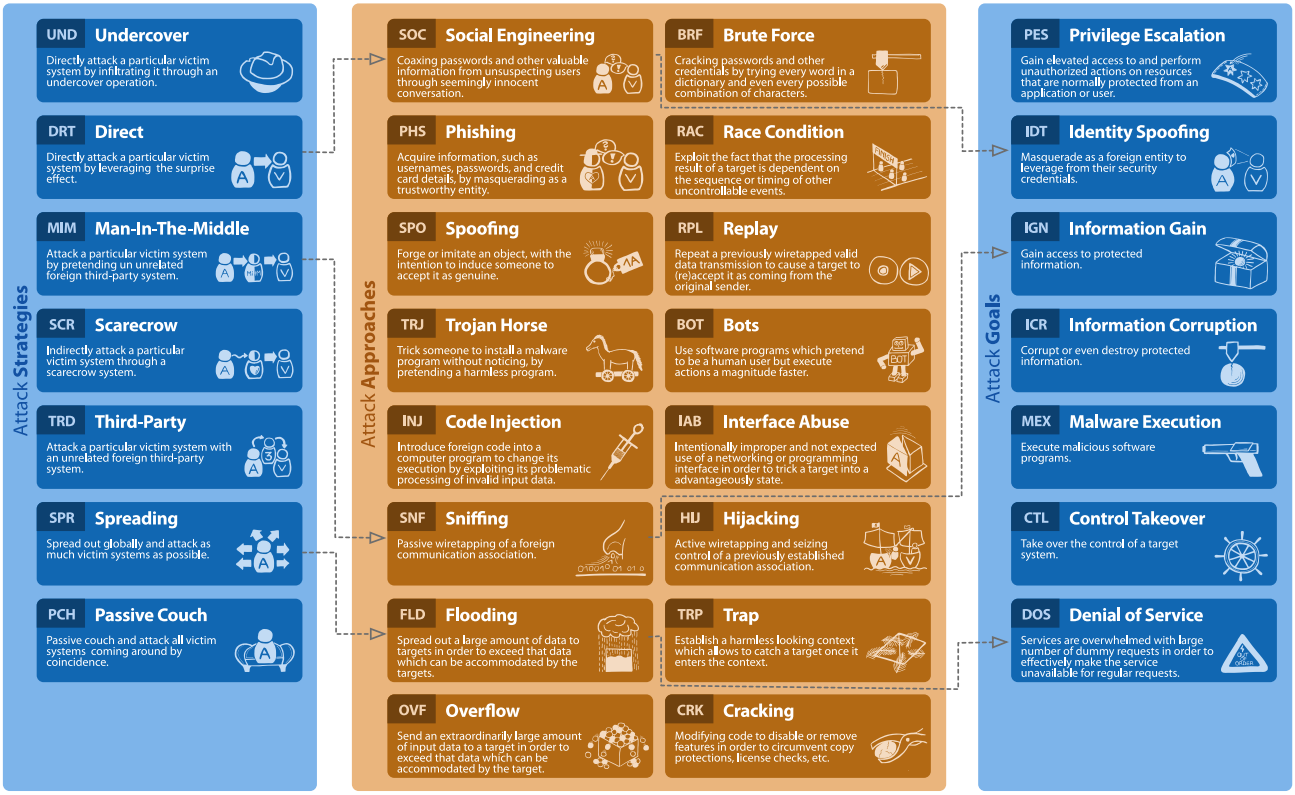
In an **Information Protection Assessment**, the information and systems are classified and attributed according to these three requirements into various strengths (**very high**, **high**, **normal**, **low**, **none**).

With **Privacy by Design**, various security-relevant aspects should be taken into account directly in the design of the solution. In some cases, these are even required by law (e.g., transparency in processing, data economy, right to be forgotten, and obligation to report violations by the GDPR).

Questions

- Which three requirements are distinguished for **Information Protection Requirements**?

Security Attacks



Attack = Strategy x Approach x Goal

The space of **Security Attacks** can be spanned across three dimensions: which **Strategy** the attacker chooses, which **Approach** the attacker chooses, and what **Goal** the attacker is pursuing. Every attack is a particular combination of these three dimensions.

For an **Identity Spoofing** attack, an attacker chooses as strategy a **Direct** attack on individuals, using the **Social Engineering** approach to gain access to personal information.

For an **Information Gain** attack, an attacker chooses, for example, a **Man-In-The-Middle** strategy, in which the attacker hooks into a foreign communication and uses the passive approach of **Sniffing** to obtain information.

For a **Denial of Service** attack, an attacker chooses, for example, the **Spreading** strategy, in which he uses the **Flooding** approach to overload a target system with requests.

Questions

- Which three dimensions can be used to understand **Security Attacks**?

Security Defenses





Intellectual Content: Version 1.0.2 (2012-10-17). Authored 2012 by Dr. Ralf S. Engelschall. Graphical Illustration: Version 1.0.4 (2020-04-23). Copyright © 2012-2020 Dr. Ralf S. Engelschall <http://engelschall.com>. All rights Reserved. Unauthorized Reproduction Prohibited. Licenced to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

There are numerous **Security Defenses**. In practice, however, these are divided into two areas: those that are based at the level of the **Network Protocols** and those that start at the **Software Application** level.

At the level of the **Network Protocols** classic defense measures are the use of a **Secure Network Protocol** such as the very widely used **Transport Layer Security (TLS)**, the use of a **Protocol Proxy** such as NGINX or PGProxy, the **Peer Identification**, **Peer Authentication** and **Digital Signatures** via X.509 certificates and the **Peer Whitelisting** via an Access Control List (ACL).

At the level of the **Software Application**, classic defense measures are the specification of a **Strong Password Policy** with, for example, a minimum length specification, the use of a **Role-Based Authorization** such as Linux RBAC, and **Execution Sandboxing** using, for example, virtual machines like JVM or containers like Docker.

Questions

- Name a very widely used **Secure Network Protocol**, which is one of the most common **Security Defenses** at the **Network Protocol** level!