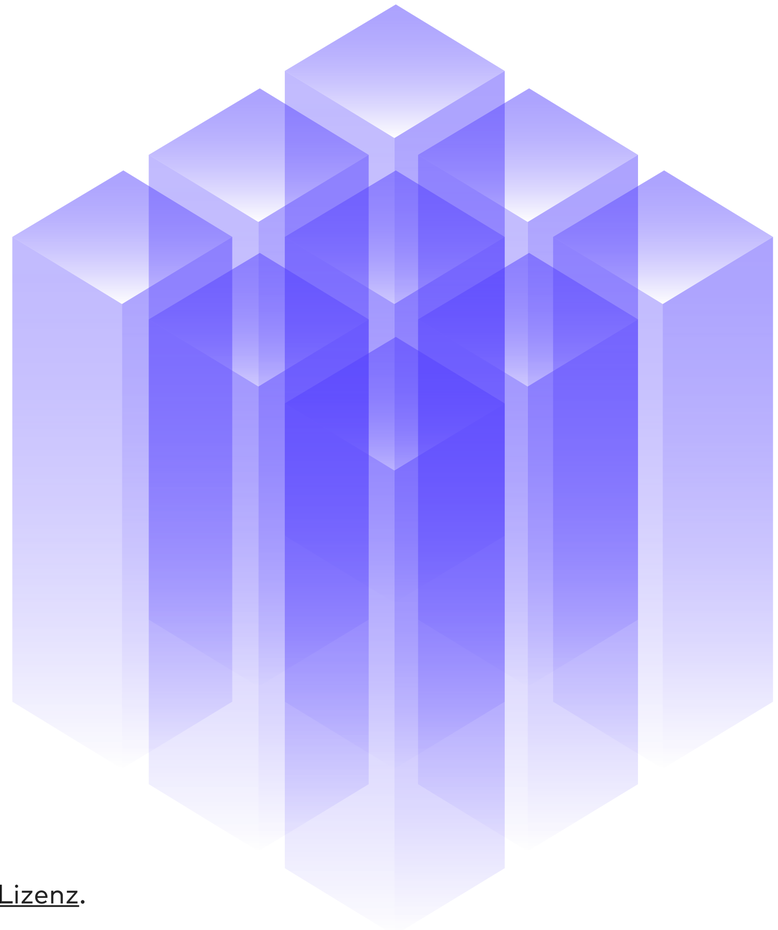
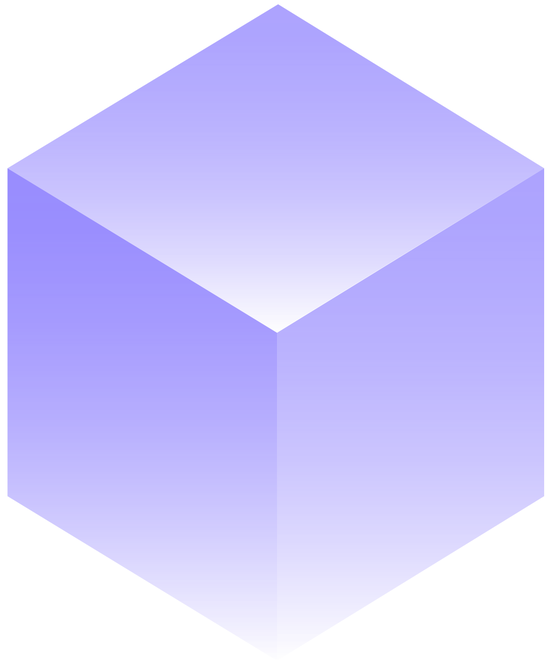
Self-contained Systems (SCS)

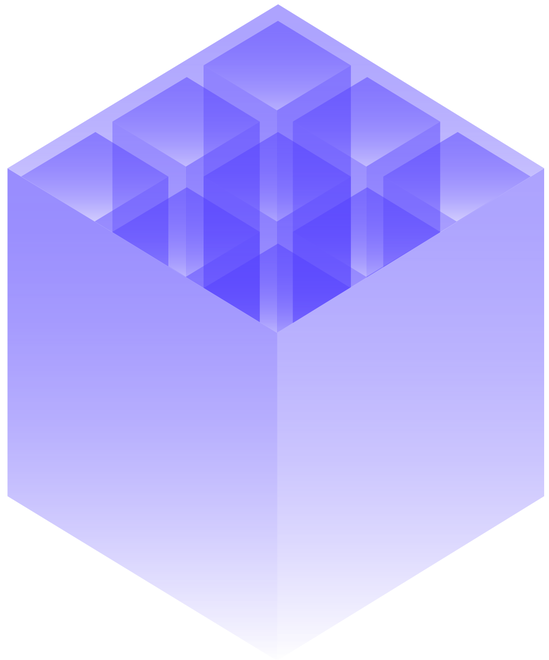
Ein Architekturansatz, der die Funktionalität eines größeren Systems in viele unabhängige, kollaborierende Systeme separiert.



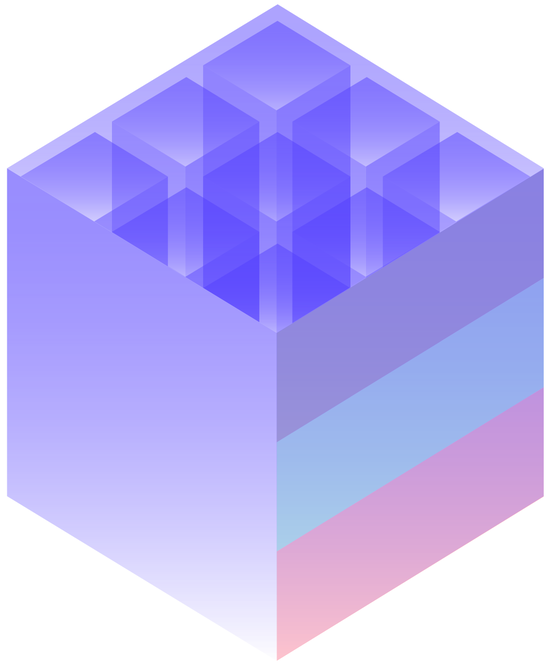
Created by INNOQ. Driven by the Community.



Ein Monolith beherbergt
eine **Vielzahl** von Dingen
in einem System...



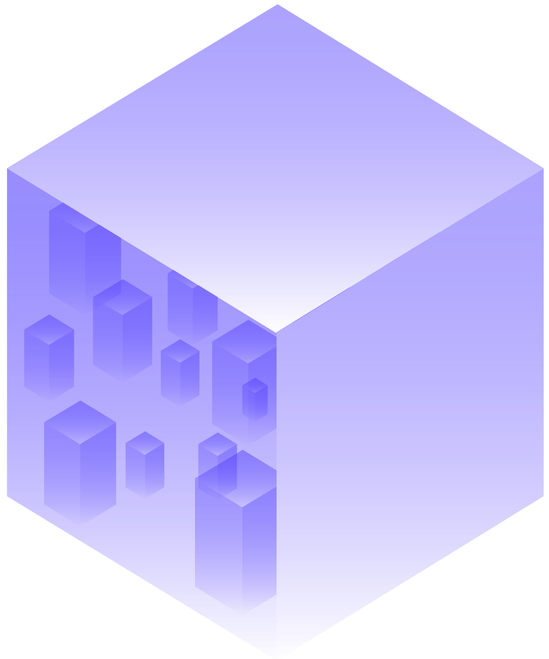
Fachlichkeit und verschiedene Domänen



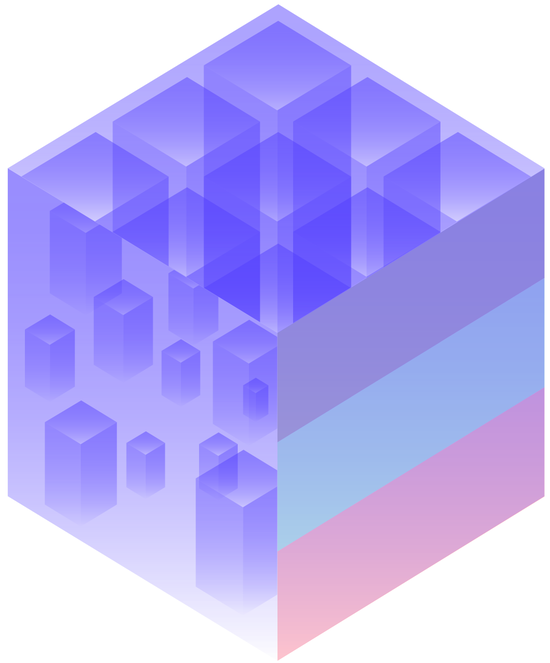
Benutzer:innenoberfläche

Geschäftslogik

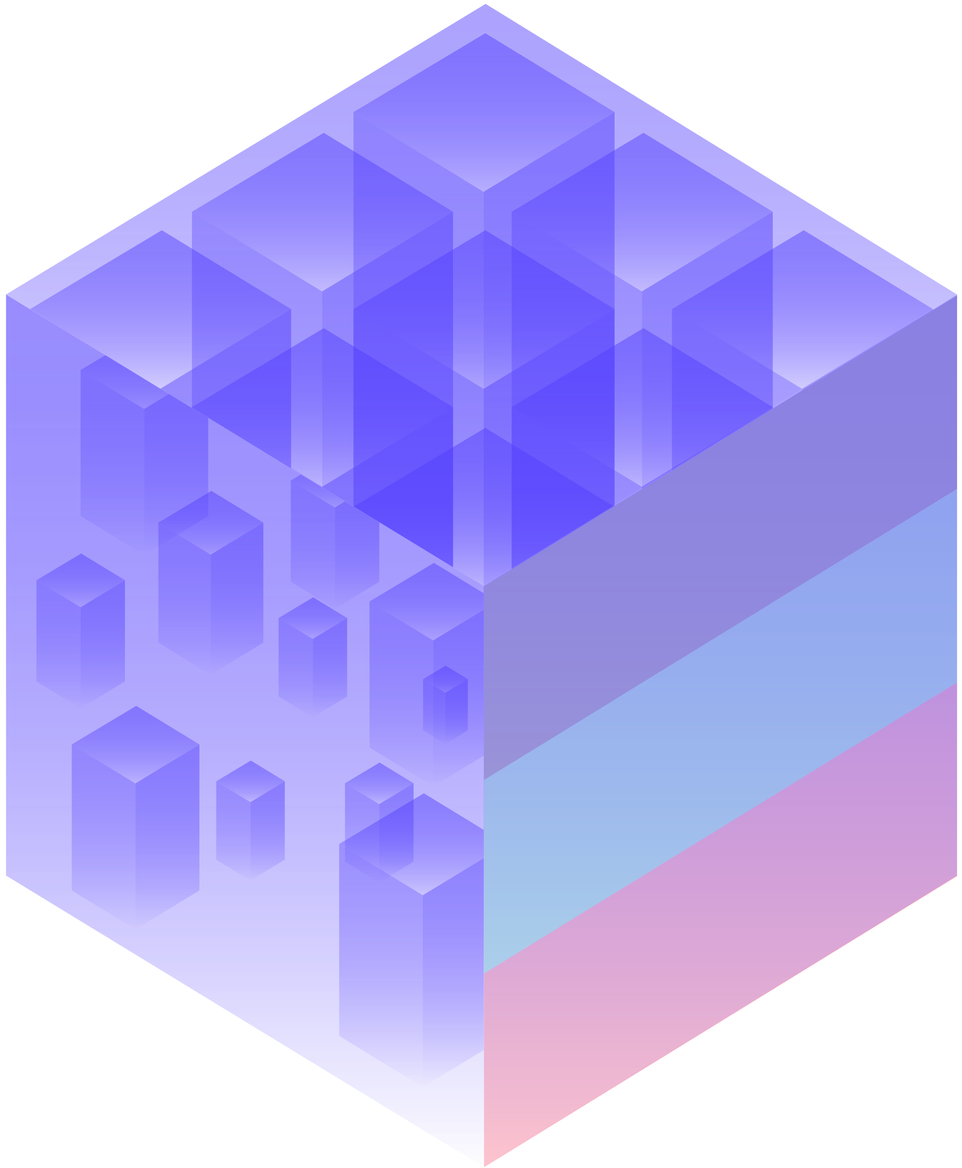
Datenhaltung



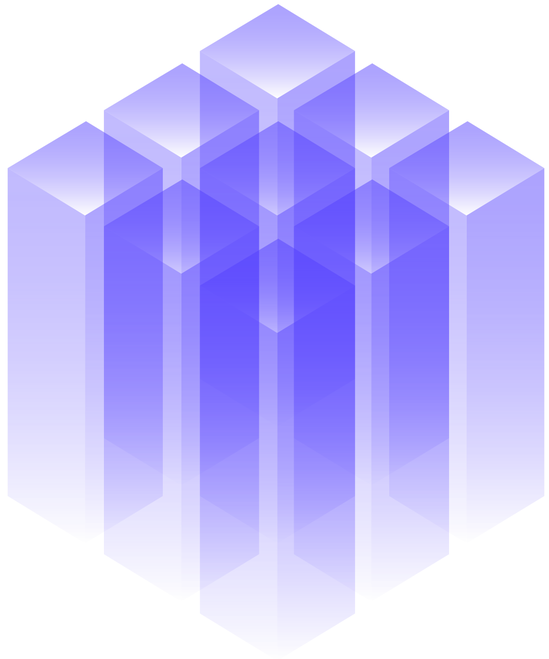
... sowie **verschiedene** Module, Komponenten, Frameworks und Bibliotheken.



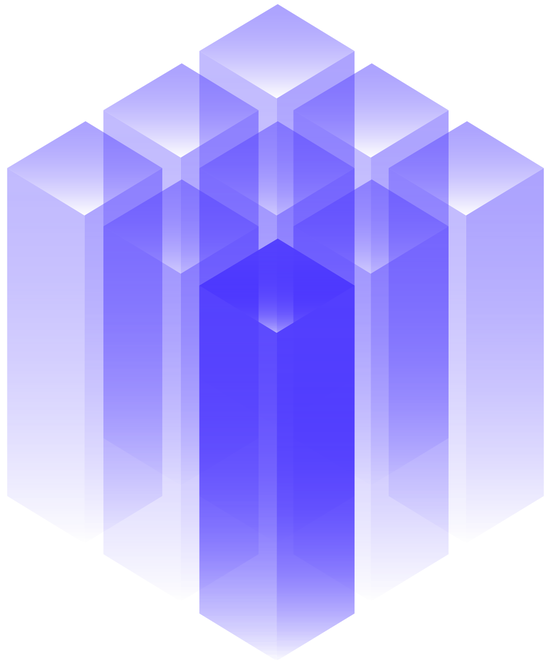
Dabei tendiert ein
Monolith dazu,
über die Zeit zu **wachsen**.



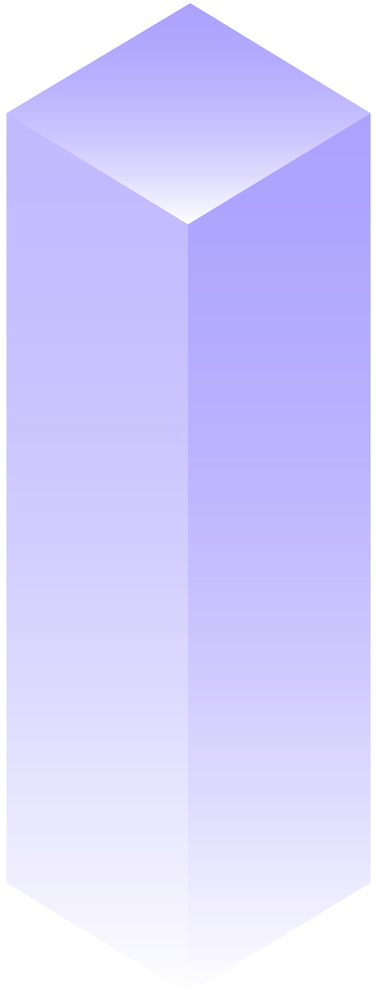
Dabei tendiert ein
Monolith dazu,
über die Zeit zu **wachsen**.



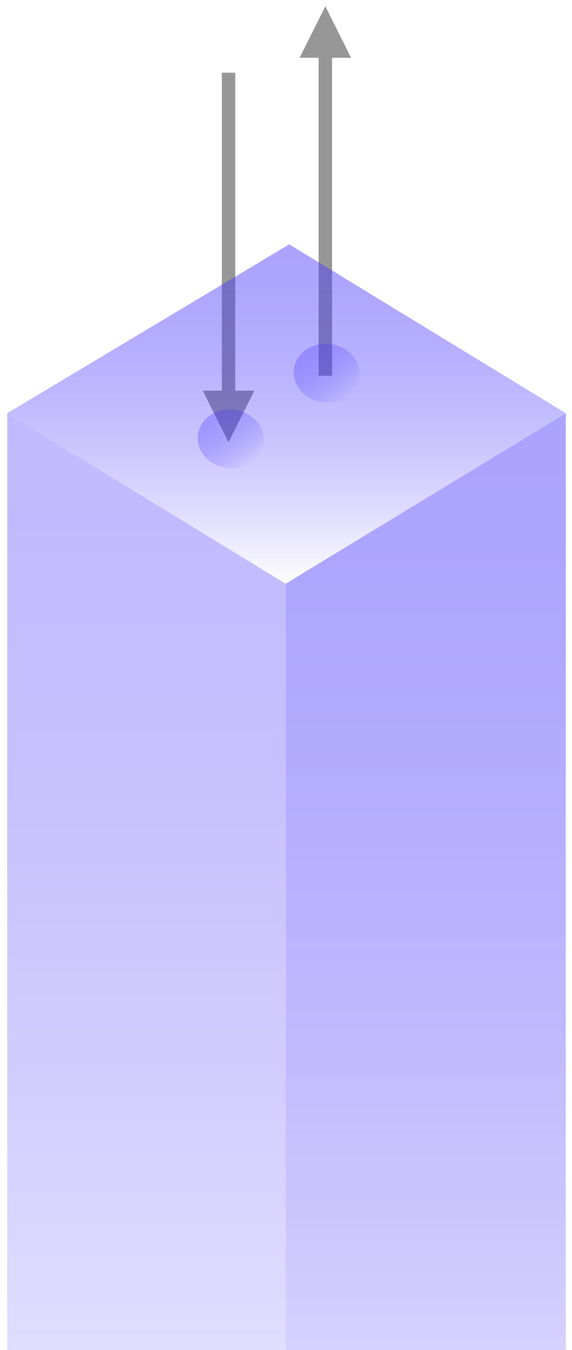
Zerschneidet man ein solches monolithisches System entlang seiner fachlichen Grenzen...



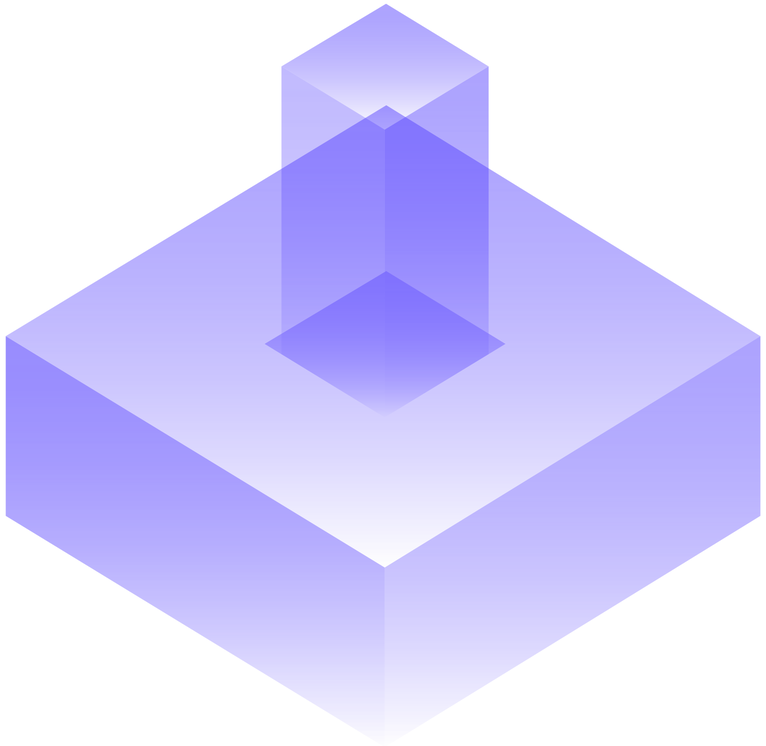
...und entwickelt jede Domäne in einer **eigenständigen, austauschbaren Webanwendung**...



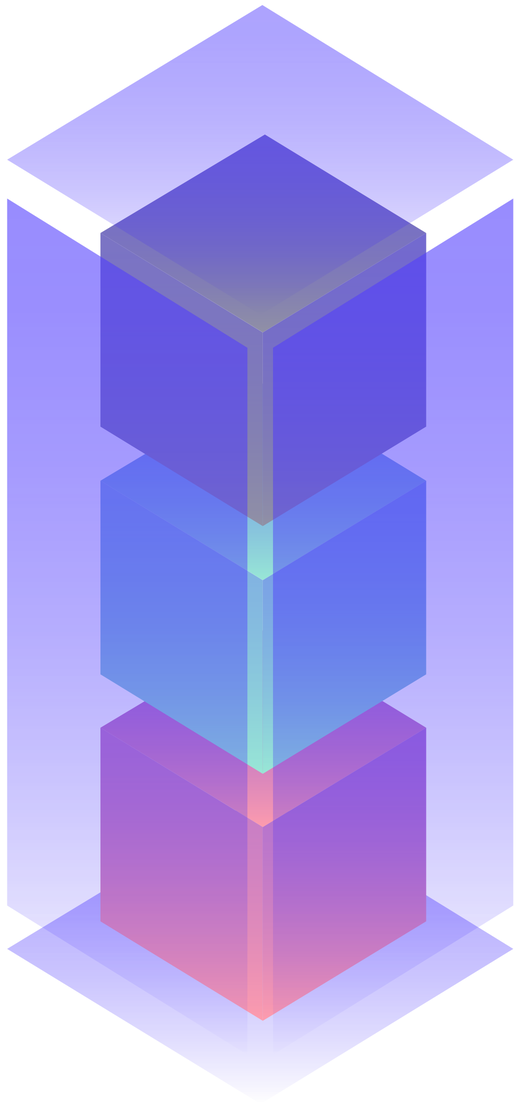
...so kann man diese Anwendung als ein **Self-contained System** (SCS) betrachten.



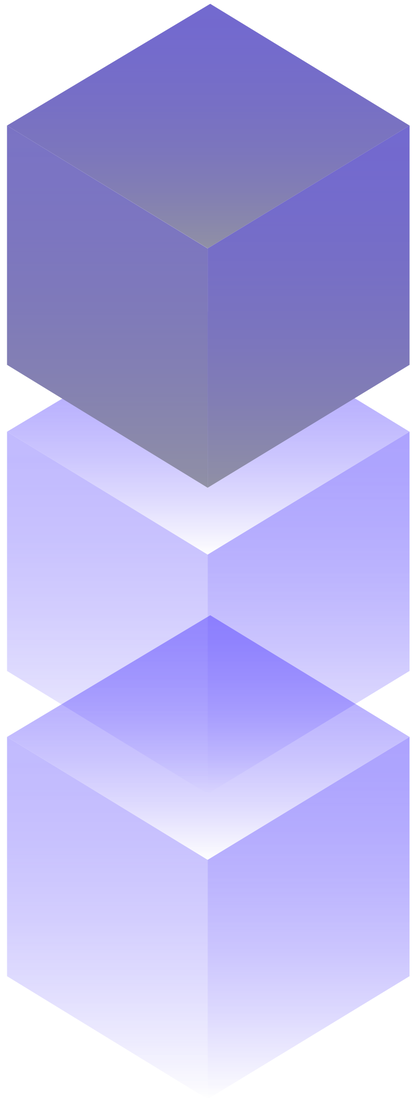
Nach außen bildet ein SCS eine dezentrale Einheit, die nur über **RESTful HTTP** oder **leichtgewichtiges Messaging** mit anderen Systemen kommuniziert.



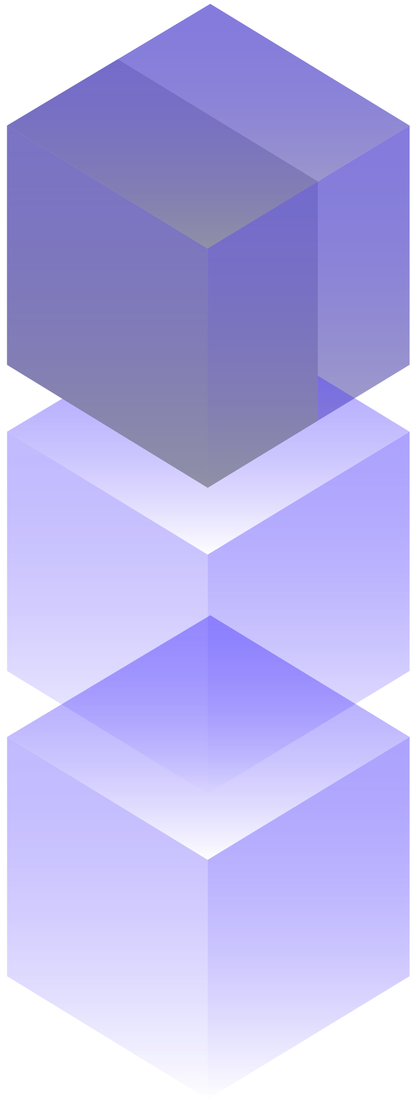
Unabhängig davon kann für jedes SCS individuell entschieden werden, welche Systemplattform verwendet werden soll.



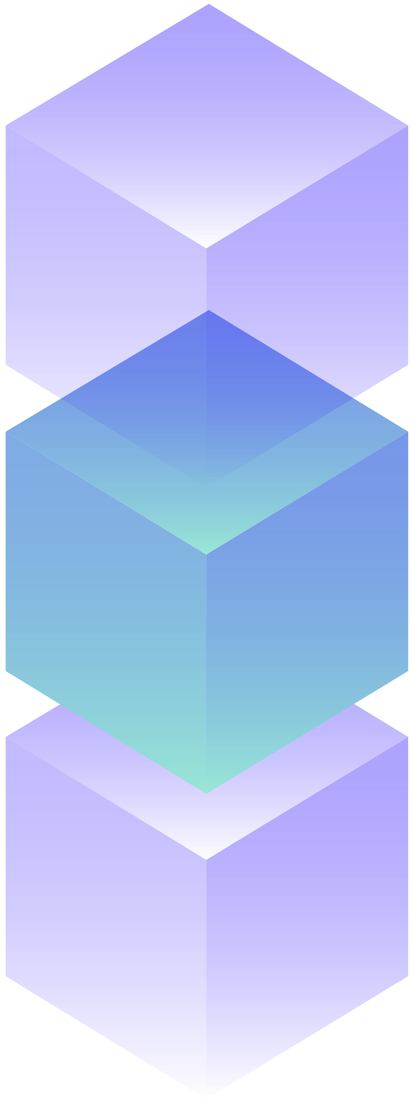
Jedes SCS enthält ein eigenes **UI**, eine für seine Fachlichkeit spezifische **Geschäftslogik** sowie separate **Datenhaltung**.



Das UI besteht aus Web-technologien, die nach den **ROCA-Prinzipien** eingesetzt werden.

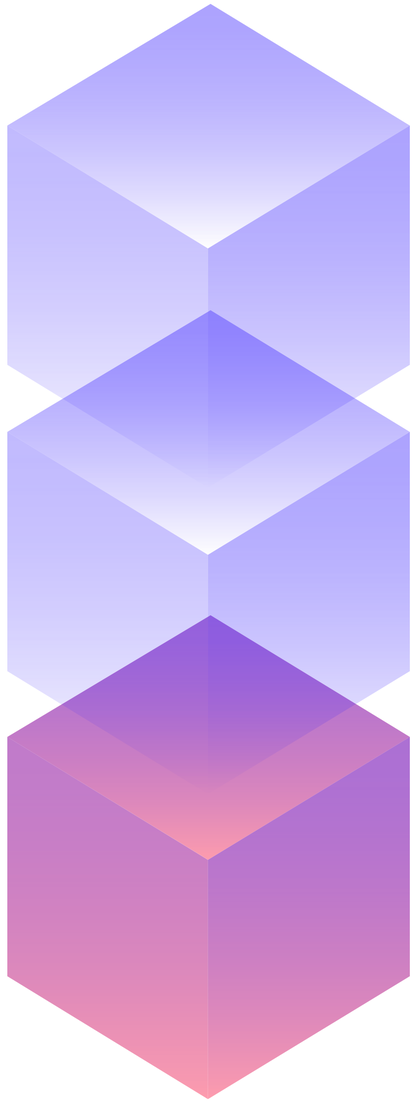


Neben der Weboberfläche kann ein Self-contained System auch ein **optionales API** anbieten.



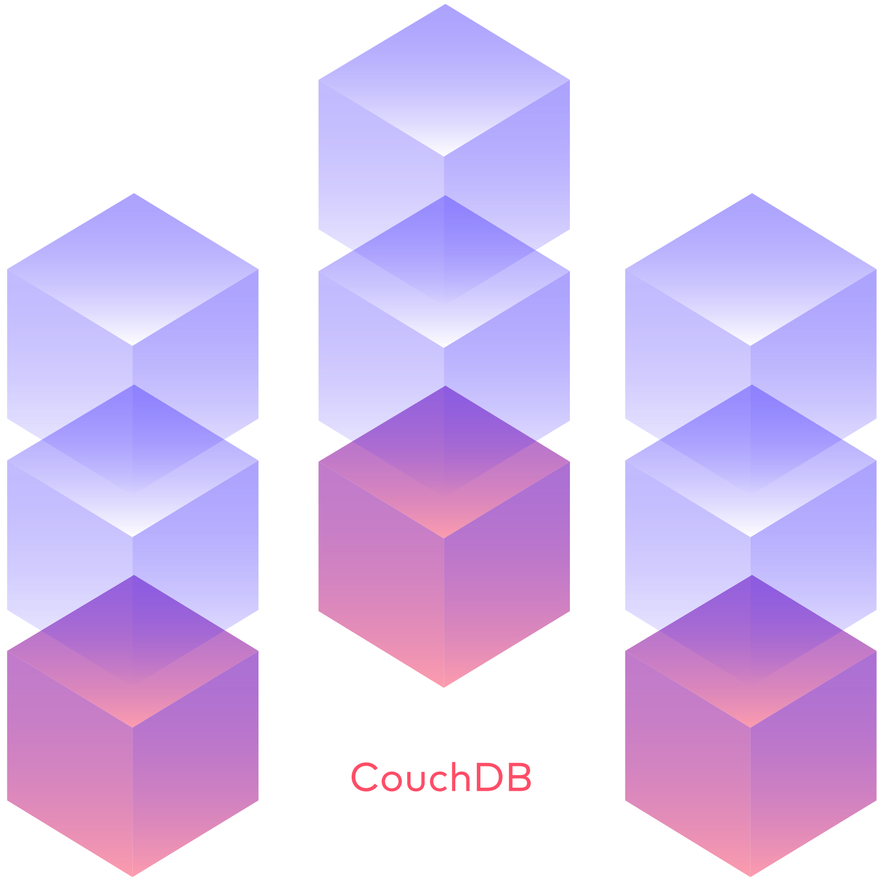
Die Geschäftslogik eines SCS löst **ausschließlich** Probleme der zu Grunde liegenden Domäne. Diese Logik wird **nur** über eine klar definierte Schnittstelle mit anderen Systemen geteilt.

Innerhalb der Geschäftslogik können für die Lösung eines fachlichen Problems u. a. auch **Microservices** eingesetzt werden.

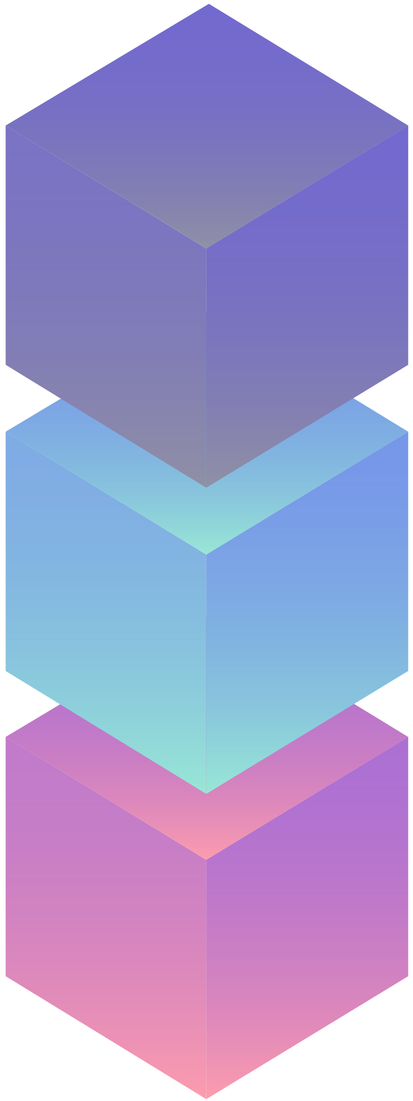


Jedes SCS besitzt eine **separate Datenhaltung**, die unter Umständen eine für die Domäne erforderliche Redundanz aufweist.

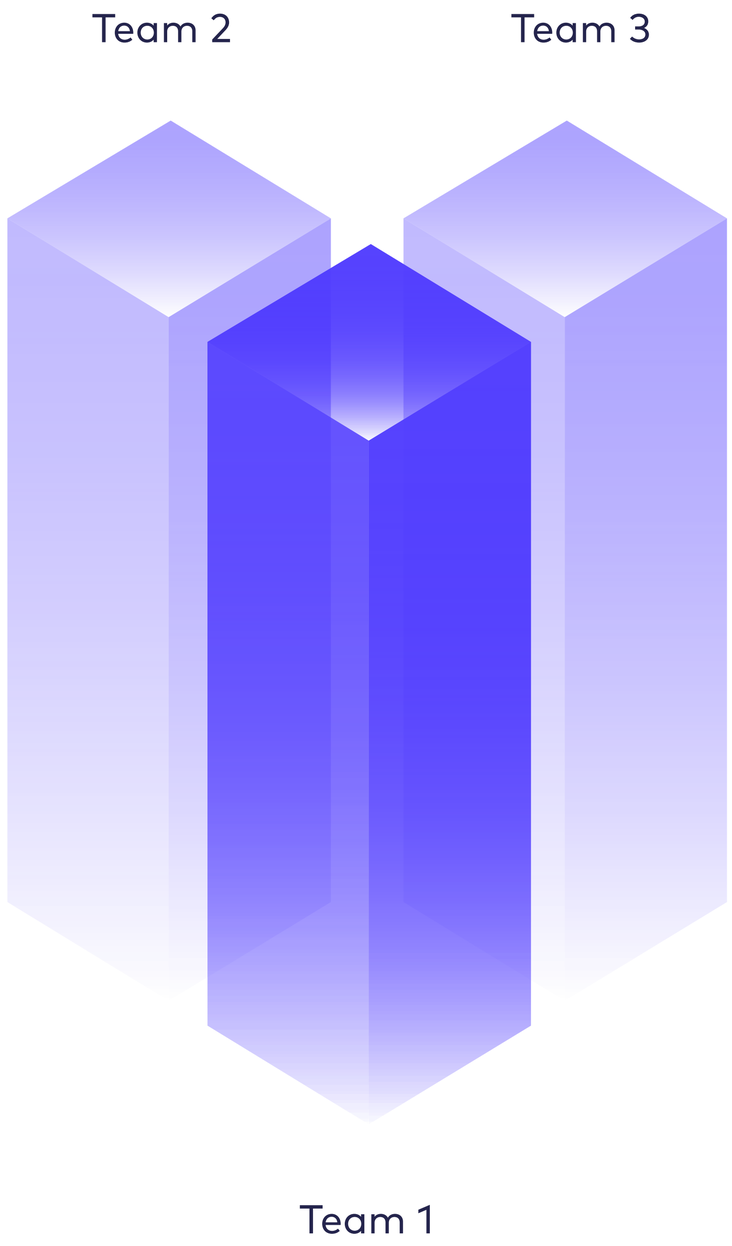
Auch wenn Redundanzen von Daten in Kauf genommen werden, beeinflusst dies nicht die **Datenhoheit** des führenden Systems.



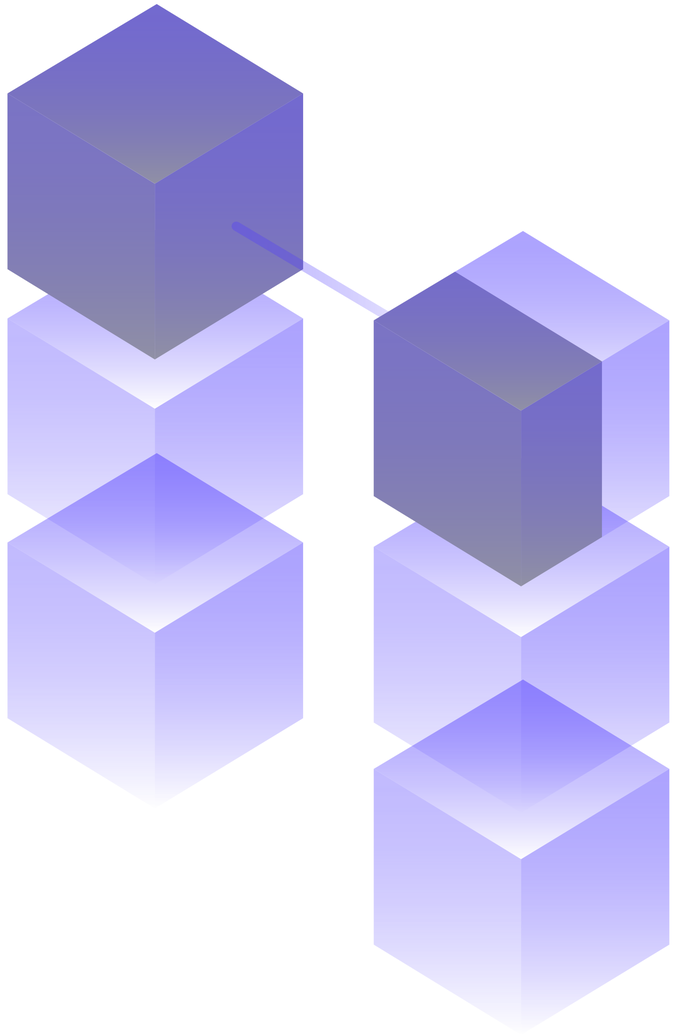
Dies ermöglicht **polyglotte Persistenz**, da unabhängig von anderen Systemen ein **passendes** DBMS für ein konkretes fachliches Problem eingesetzt werden kann.



Innerhalb eines SCS können auch weitere **flexible technische Entscheidungen** getroffen werden. So etwa die Wahl der Programmiersprache, der eingesetzten Frameworks oder der Entwicklungswerkzeuge.



Durch den klaren fachlichen Scope kann ein SCS von **einem Team** entwickelt, betrieben und gewartet werden.

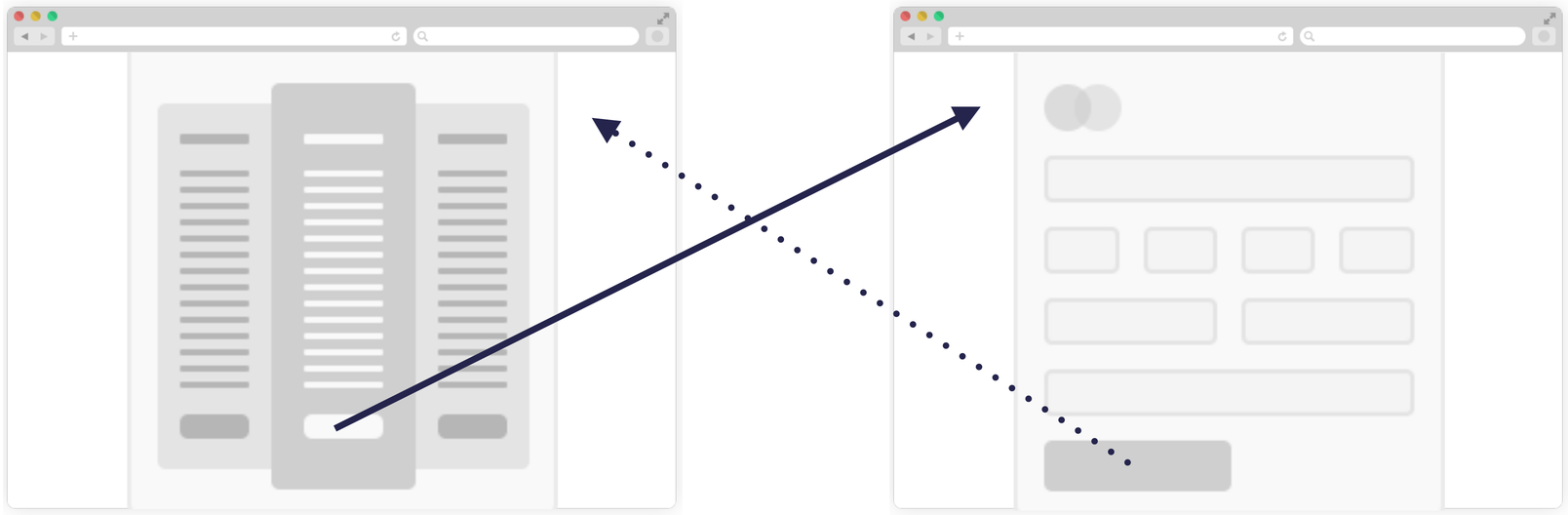


Self-Contained Systems sollten vorzugsweise über ihre **Weboberflächen** integriert werden, um die Kopplung an andere Systeme minimal zu halten.

System 1

System 2

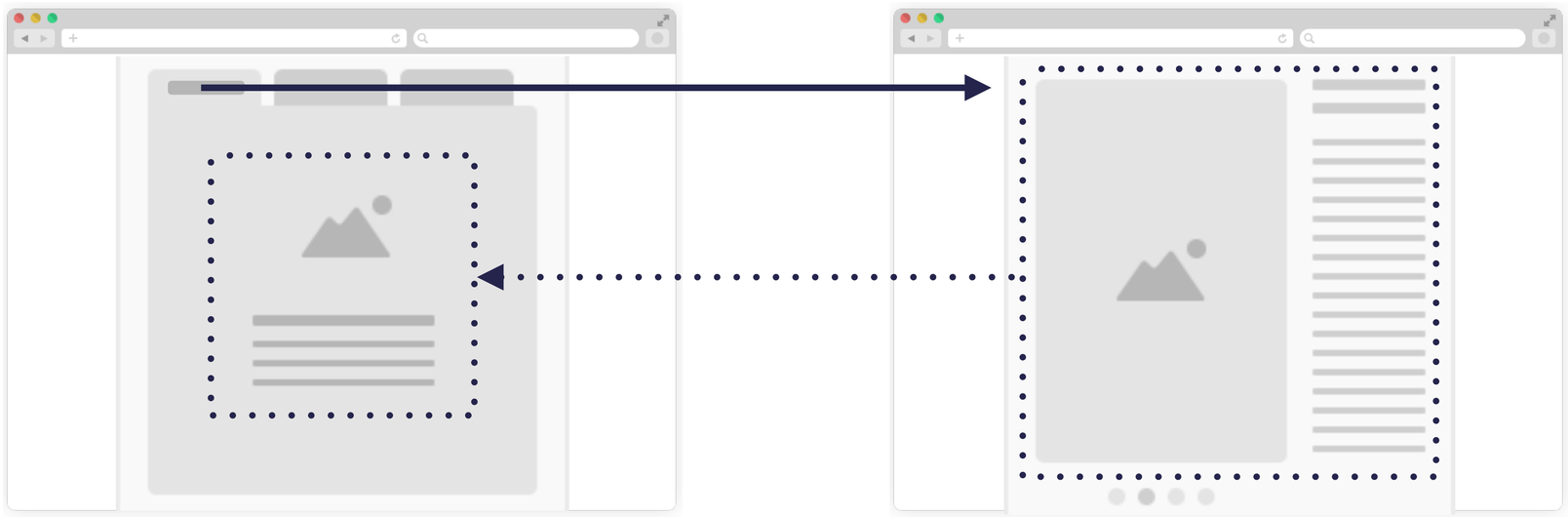
Dabei werden z. B. einfache Links benutzt, um zwischen verschiedenen Anwendungen zu navigieren.



System 1

System 2

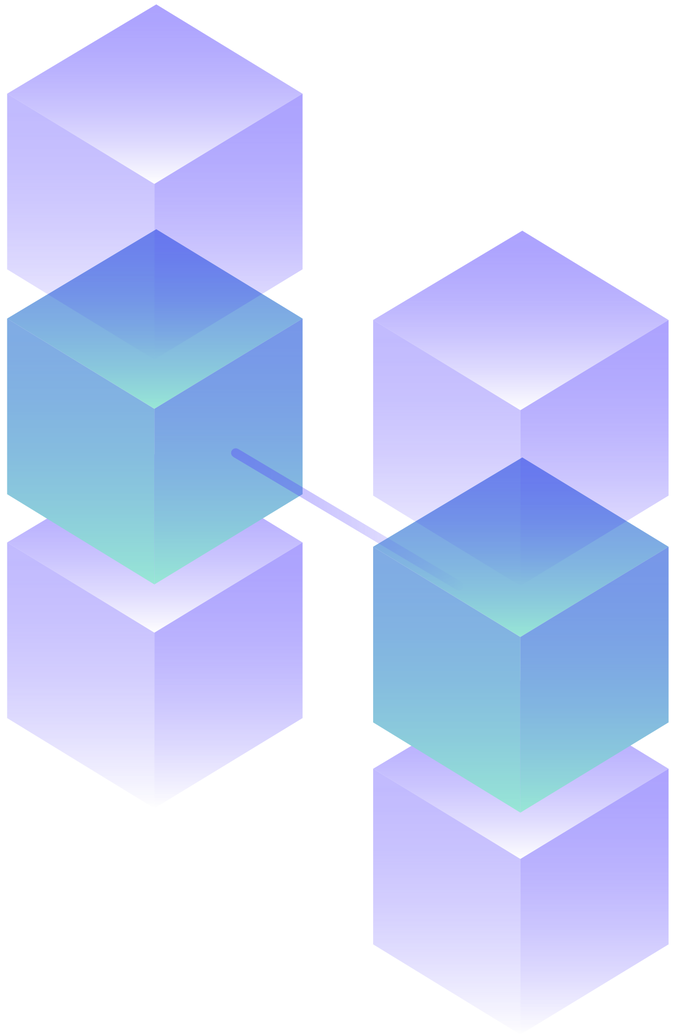
Soll die Navigation in beide Richtungen möglich sein, kann eine **Rücksprung-Adresse** mitgegeben werden.



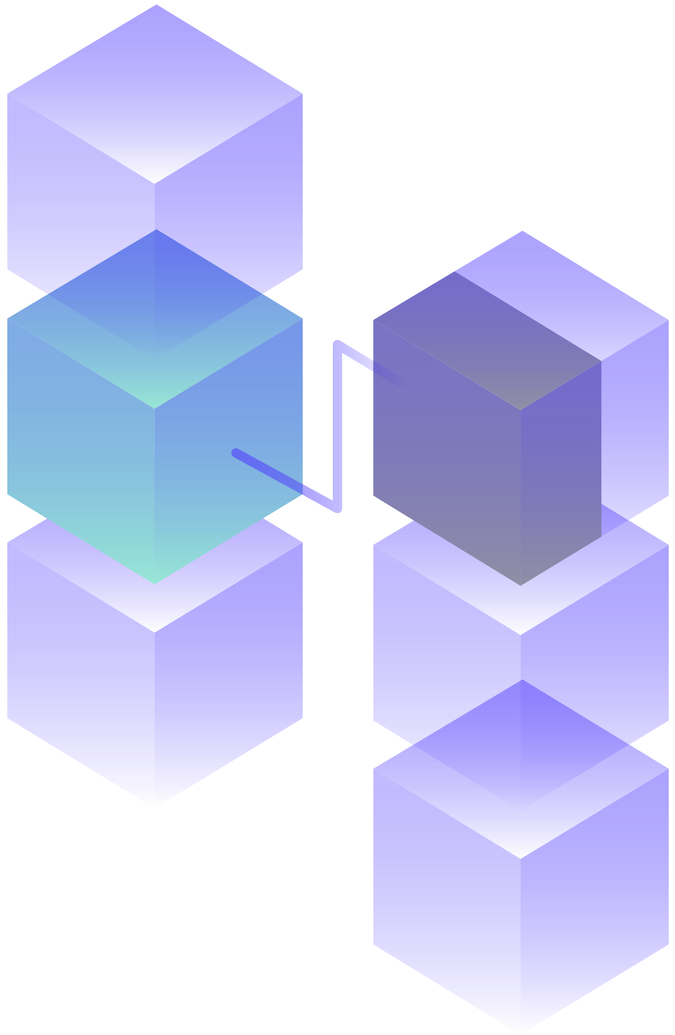
System 1

System 2

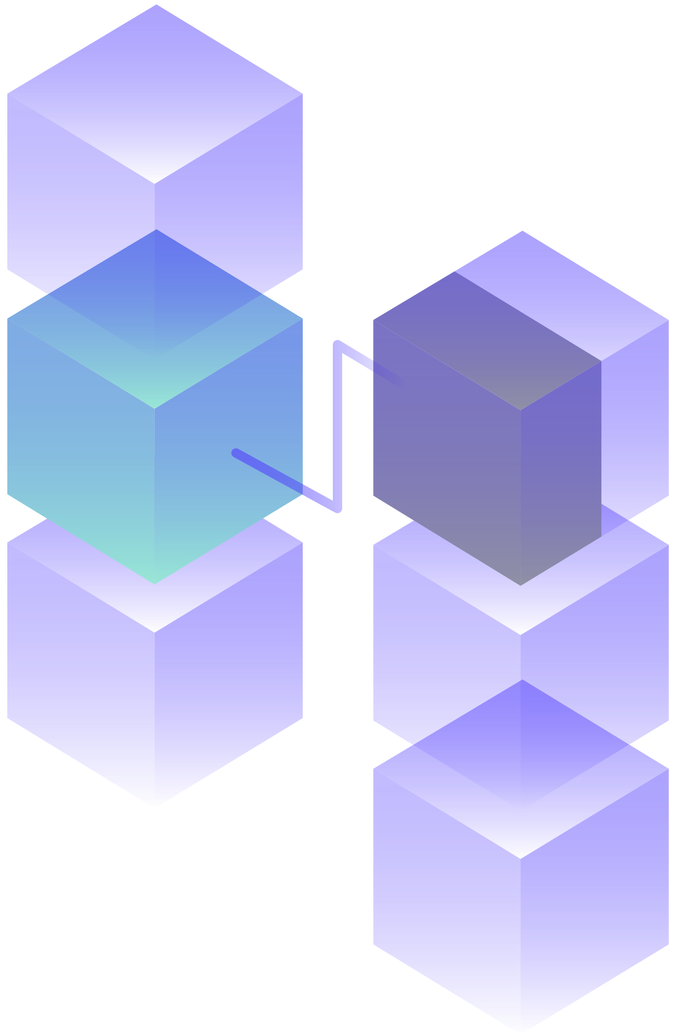
Darüber hinaus können Links auch dazu genutzt werden, einzelne Seitenbereiche eines anderen SCS mit Hilfe von JavaScript dynamisch in das UI einzubetten.



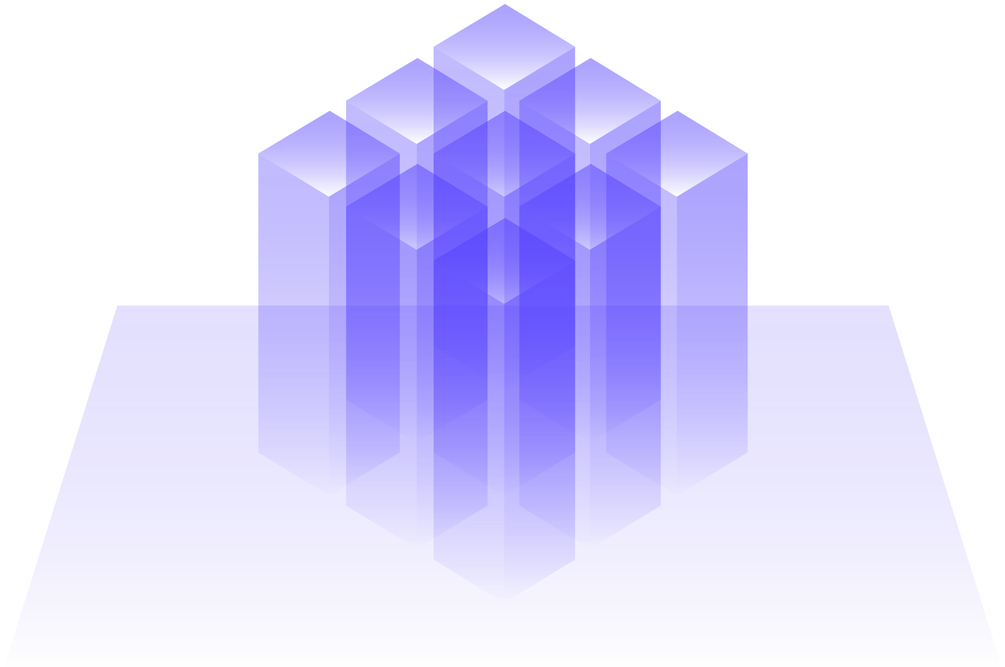
Um die Kopplung zu anderen Systemen **minimal** zu halten sollte auf entfernte, synchrone Aufrufe innerhalb der Geschäftslogik verzichtet werden.



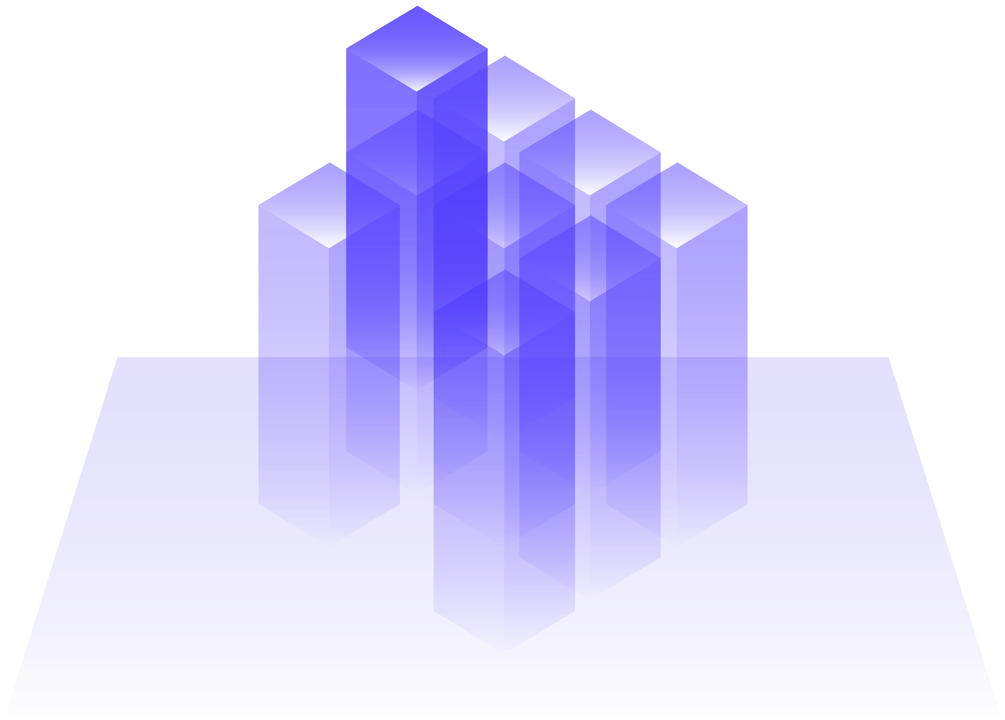
Falls sich entfernte Aufrufe eines APIs nicht vermeiden lassen, sollten diese **asynchron** erfolgen, um Abhängigkeiten zu reduzieren und Fehlerkaskaden vorzubeugen.



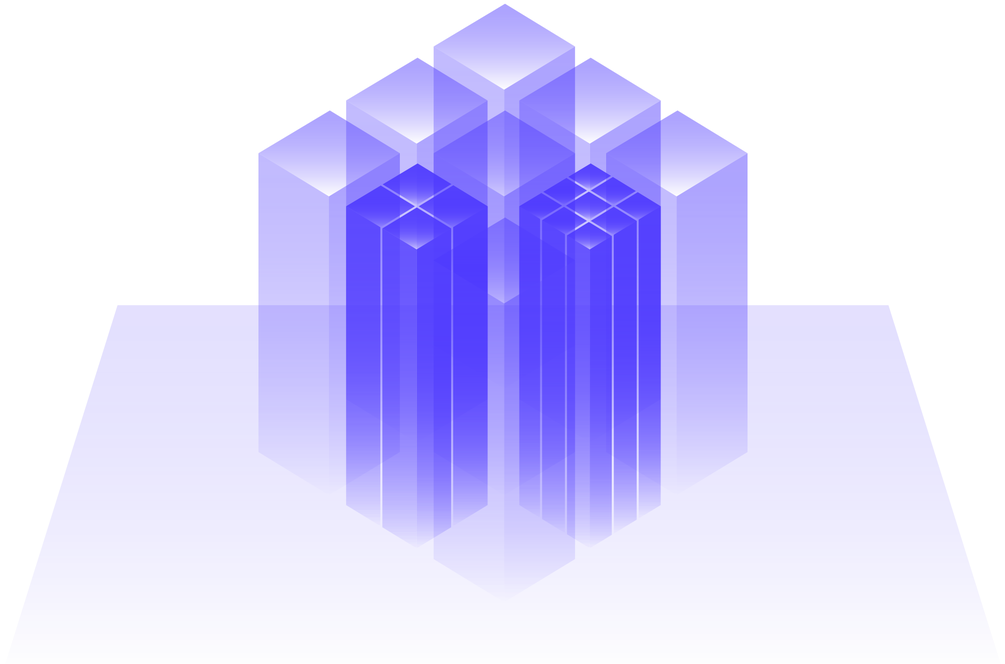
Dies setzt jedoch voraus, dass Änderungshäufigkeit und Aktualitätserwartung des Datenbestandes einen „**Eventual Consistency**“ Ansatz erlauben.



Ein auf diese Weise integriertes
System of Systems
besitzt viele Vorteile.



Die **Entkopplung** einzelner, austauschbarer Systeme führt zu einem stabileren Gesamtsystem.

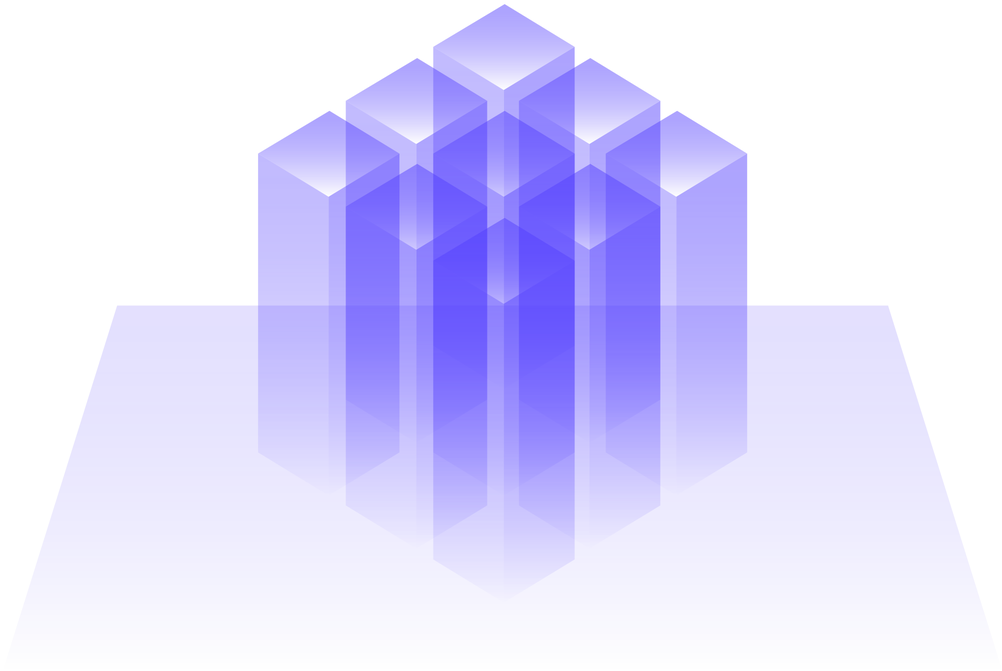


Darüber hinaus können
einzelne Systeme nach Bedarf
individuell skaliert und auf die zu
erwartende Last angepasst werden.

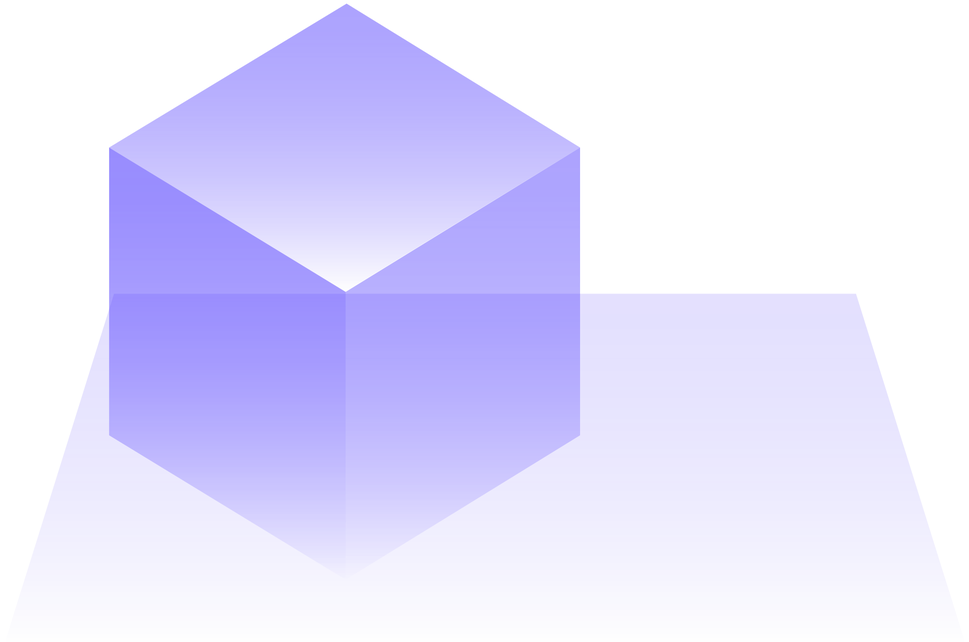
Version 1

Um ein altes, monolithisches System in ein System of Systems zu überführen, empfiehlt sich in der Regel kein **Big Bang Release**, da dieser insbesondere bei großen Systemen besonders fehleranfällig und riskant ist.

Version 2

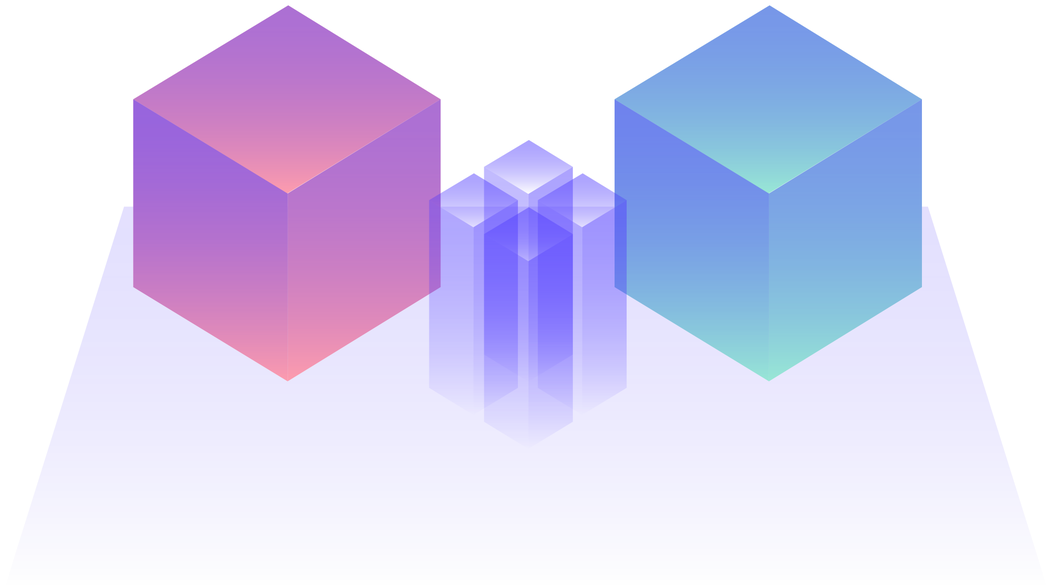


Um ein altes, monolithisches System in ein System of Systems zu überführen, empfiehlt sich in der Regel kein **Big Bang Release**, da dieser insbesondere bei großen Systemen besonders fehleranfällig und riskant ist.

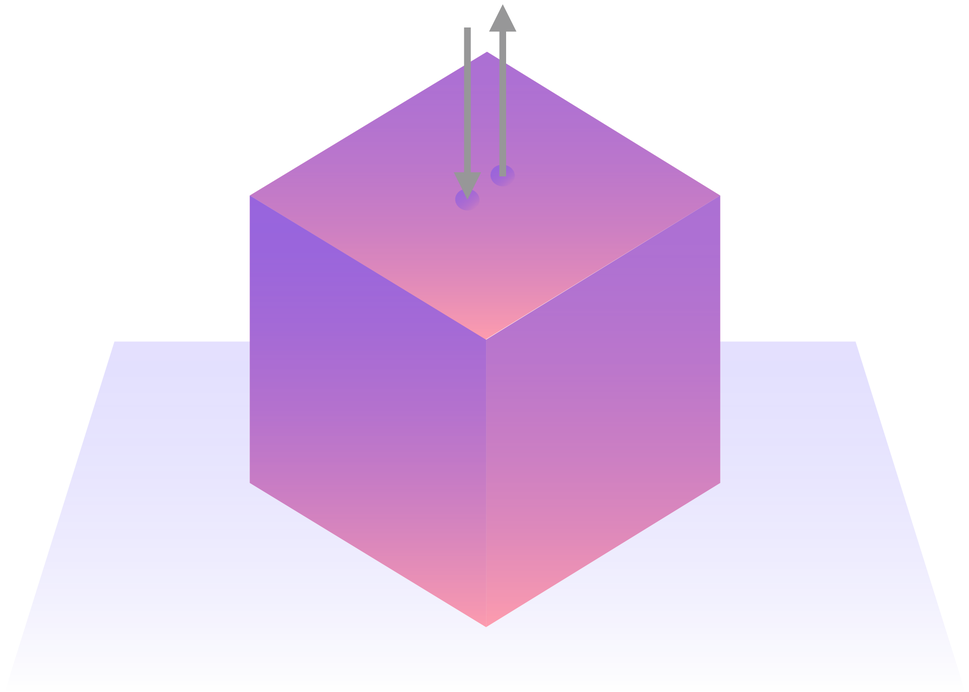


Statt dessen führen Anpassungen in häufigen, kleinen Schritten dazu, große und komplexe Systeme **evolutionär** zu modernisieren. Das Risiko eines Fehlschlags wird dabei erheblich minimiert.

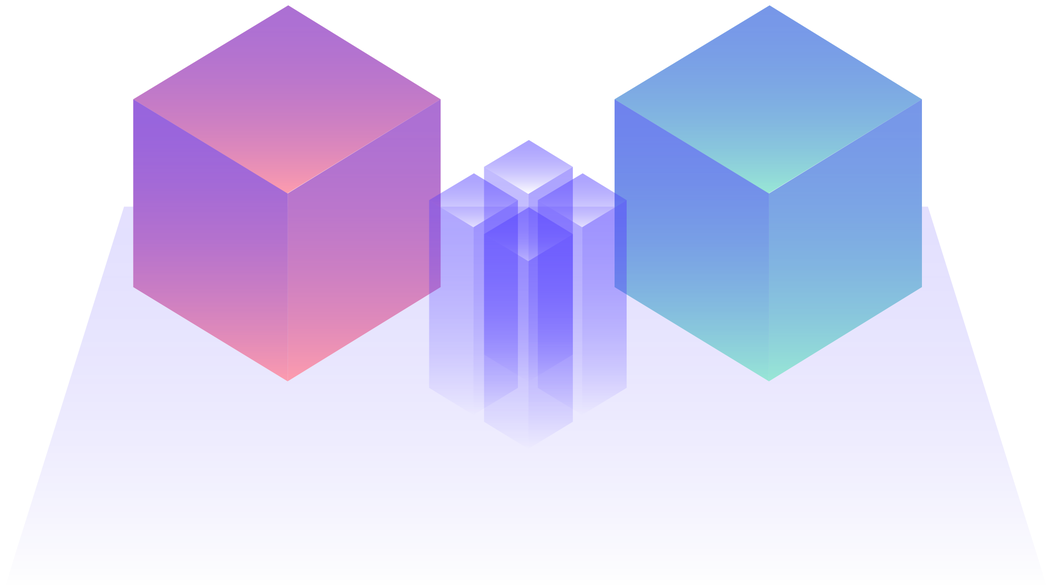
Statt dessen führen Anpassungen in häufigen, kleinen Schritten dazu, große und komplexe Systeme **evolutionär** zu modernisieren. Das Risiko eines Fehlschlags wird dabei erheblich minimiert.



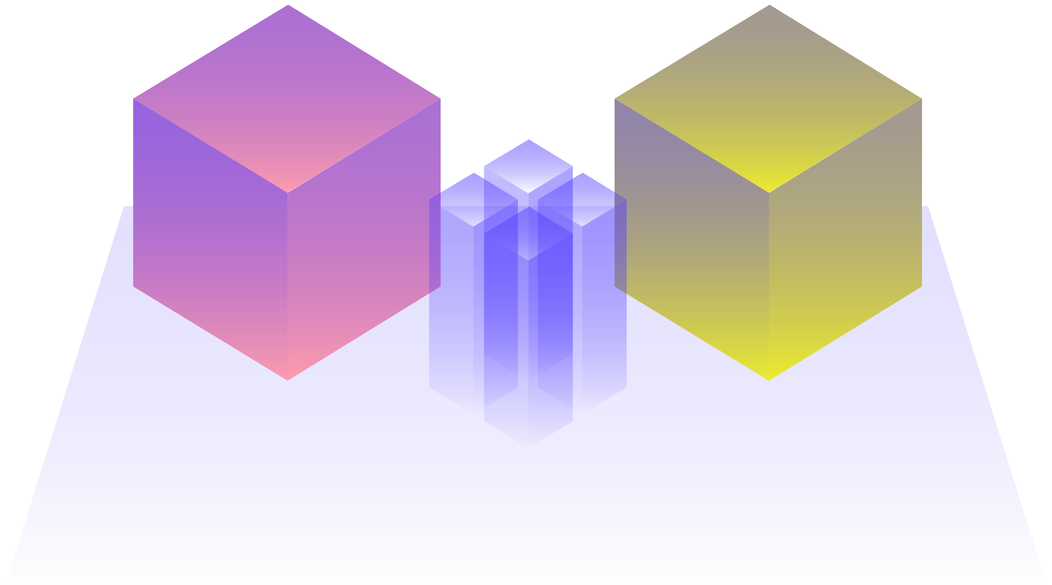
In der Realität besteht ein System of Systems in Teilen sowohl aus Individual-Software als auch aus Standardprodukten.



Bei der Auswahl von Produkten ist darauf zu achten, dass die Software klar definierte Aufgaben erfüllt und sich über dieselben Mechanismen wie individuell entwickelte Self-Contained Systems integrieren lässt.



Dies sorgt dafür, dass sich auch Standardprodukte nach Ablauf ihrer Nutzungszeit risikoarm durch ein nachfolgendes System ersetzen lassen.



Dies sorgt dafür, dass sich auch Standardprodukte nach Ablauf ihrer Nutzungszeit risikoarm durch ein nachfolgendes System ersetzen lassen.

Findet sich kein Produkt, dass sich nach den angestrebten Kriterien in das Gesamtsystem integrieren lässt, sollte eine Alternative gewählt werden, die sich zumindest um einheitliche Schnittstellen erweitern lässt.

INNOQ

www.innoq.com

Sie finden mehr tiefer gehenden Inhalt zu Self-contained Systems, Microservices, Monolithen, REST oder ROCA unter innoq.com/scs

Sie möchten Ihre IT-Landschaft modernisieren?
Oder etwas Neues entwickeln?
Wir unterstützen Sie gerne. Schreiben Sie uns!