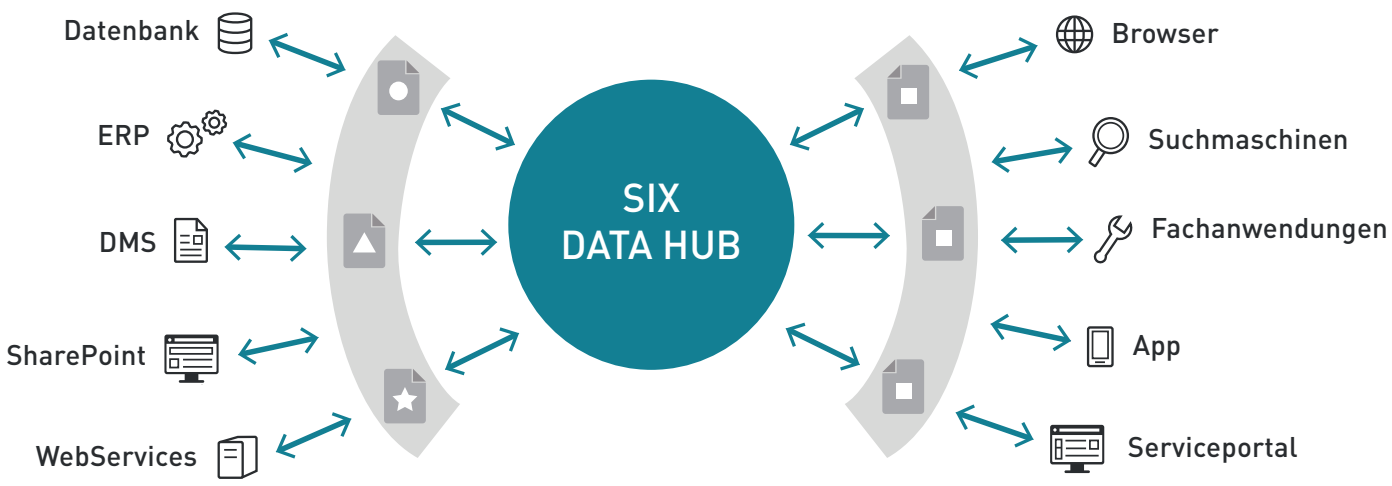
EINER FÜR ALLE: DER SIX DATA HUB

Datenquellen und Systeme unkompliziert miteinander verbinden



Sie möchten mehrere Systeme miteinander verbinden? Zum Beispiel eine Produktdatenbank oder ein Fachverfahren mit einem Content Management System? Und das mit geringem Programmieraufwand? Dann ist der Six Data Hub genau das Richtige für Sie.

EINHEITLICHE KOMMUNIKATION

Der Six Data Hub ist eine von der Six Offene Systeme GmbH entwickelte High-Level-Schnittstelle, die eine einheitliche Kommunikation zwischen verschiedenen Systemen ermöglicht. Sie dient als Datendrehscheibe zwischen unterschiedlichen Datenquellen. Mithilfe des Six Data Hub werden beliebige Datensätze mit einer einheitlichen Zugriffsmethode ausgelesen und weitergegeben: beispielsweise aus einer Fachanwendung, einer Produktdatenbank, einem Media Asset Management System oder einem Web-CMS.

Der Six Data Hub gewährleistet einen ungehinderten Datenaustausch. Er sorgt dafür, dass sich alle Systeme und Anwendungen miteinander verstehen, obwohl die unterschiedlichen Datenquellen oftmals über eine eigene Datenstruktur und voneinander abweichende APIs verfügen. Six-Kunden wie die Polizei Brandenburg, die Stadt Rostock und CEWE, Europas größter Fotodienstleister, haben den Six Data Hub bereits im Einsatz.

WIE FUNKTIONIERT DER SIX DATA HUB?

Der Six Data Hub basiert auf dem Data Access Object Pattern (DAO), ein in der Informatik verwendetes Implementierungs-Pattern, das „den Zugriff auf unterschiedliche Arten von Datenquellen kapselt. Dadurch wird garantiert, dass die angesprochenen Datenquellen ausgetauscht werden können, ohne dass der aufrufende Code geändert werden muss“ (Wikipedia). Die Programmierlogik wird somit von technischen Details der Datenspeicherung befreit.

Die Installation des Six Data Hub erfolgt unkompliziert über den De-facto-Standard ‚Composer‘. Nach der Konfiguration (Tabellen, Felder etc.) benötigt ein Entwickler kein spezifisches Wissen über die einzelnen Systeme, die mit dem Six Data Hub verbunden sind, wenn für die anzubindende Datenquelle ein Treiber existiert. Die bezogenen Werte werden vom Six Data Hub automatisch normalisiert und gegebenenfalls transformiert. Das heißt: Sämtliche Datensätze, Felder, Fehlermeldungen etc. werden in ein einheitliches Rückgabeformat umgewandelt.

SIX DATA HUB: DIE KOMPONENTEN

Der Treiber stellt die Verbindung zwischen der Datenquelle und dem Repository her. Er normalisiert und transformiert die Daten und liefert ein RepositoryResult-Objekt.

Das Entity enthält ein Datenabbild der Datenquelle. Die Daten liegen in normalisierter und transformierter Form vor. Entity ist eine einfache Klasse, die keine Business-Logik enthält.

Das Modell besteht aus Entity plus Business-Logik.

Die Mapping-Datei definiert das Mapping zwischen Quelldaten und Entity. Sie enthält Informationen zu Feldern, Normalisierungsfunktionen, Transformatoren, Verbindungsparametern und Fieldsets (Felddefinitionen, die jederzeit wiederverwendet werden können).

Das Repository filtert Daten und bietet Methoden, um Daten aus einer Datenquelle zu lesen und zu schreiben. Beispiele: *readAll()*, *readByFilter()*, *delete()*, *save()*, ...

MEHR ERFAHREN?

six.de/sixdatahub

DIE VORTEILE IM ÜBERBLICK

- **Mehr Flexibilität:** Die Konfiguration der Repositories lässt sich jederzeit flexibel anpassen und erweitern, ohne dass es notwendig ist, direkt in den Code einzugreifen. D. h. der Datenaustauch bleibt gewährleistet, auch wenn die Strukturen der Datenquelle sich verändern.
- **Schnelle Konvertierung:** Ausgelesene Werte lassen sich per Konfiguration einfach konvertieren.
- **Effiziente Zusammenarbeit:** Die verteilte Entwicklung in großen Teams wird optimiert, da z. B. der Entwickler, der den Controller programmiert, nichts über die API wissen muss. Ist der entsprechende Treiber einmal implementiert, muss sich der Entwickler der Business-Logik keine Gedanken darüber machen.
- **Hohe Datensicherheit:** Die Entwicklung ist ohne Zugriff auf die eigentliche Datenquelle – z. B. eine Behördendatenbank mit vertraulichen Informationen – möglich, da Drittsysteme simuliert werden können (‚MockDriver').
- **Spürbare Zeitersparnis:** Durch die Implementierung der entsprechenden Treiber wird viel Entwicklungszeit eingespart, da bei Änderungen in den Datenquellen nur der Treiber aktualisiert werden muss.
- **Effektive Qualitätssicherung:** Systeme und Programme können sicherer und gründlicher getestet werden und lassen sich besser warten.