Daniel Liebhart

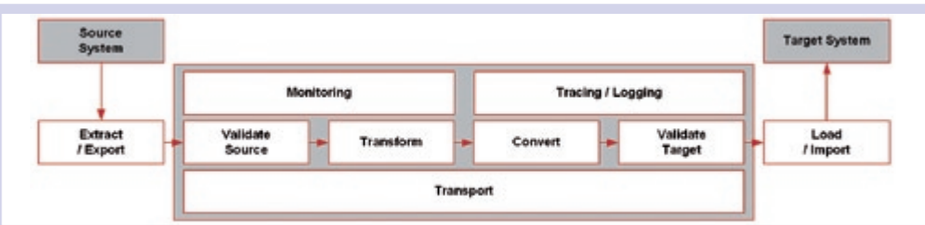
Schnittstellen sind ein Albtraum

Die Erstellung von Schnittstellen zwischen verschiedenen Systemen oder Subsystemen sind eine uninspirierte Fleissarbeit, sie sind selten gut spezifiziert oder durchdacht und oftmals die Ursache von schwierig zu findenden Fehlern. Doch der vernünftige Einsatz der richtigen Schnittstellenart schafft Abhilfe.

Es existieren keine Informationssysteme ohne Schnittstellen. Damit sind das User Interface als auch die Schnittstellen zum Betriebssystem, zur Datenbank und zu anderen Systemen gemeint. Diejenigen Schnittstellen, die uns jedoch Kopfzerbrechen bereiten, sind die zum Datenaustausch zwischen verschiedenen Systemen. Die Schwierigkeit ist, dass bestehende Systeme mit neuen verbunden werden müssen. In vielen Fällen sind die Beschreibungen der Daten, der Formate und der technischen Möglichkeiten bestehender Systeme unvollständig oder falsch. Dazu kommt, dass die Daten in verschiedenen Systemen verschieden granular verwendet werden. Laut einer Analyse von Forrester Research umfasst die Entwicklung von Schnittstellen 35 bis 40 Prozent des Gesamtaufwandes der Programmierung. Bei Punkt-zu-Punkt-Schnittstellen entfallen sogar 70 Prozent des Gesamtaufwandes auf den Informationsaustausch. Dazu kommt, dass zwischen 39 und 50 Prozent (je nach Studie) aller Fehler in einem System in der Schnittstelle auftreten. Die am häufigsten auftretenden Interface-Fehler sind mangelhafte Fehlerbehandlung (15 Prozent), unzureichende Umsetzung der durch die Schnittstelle spezifizierten Funktionalität (13 Prozent), mangelhaftes Postprocessing nach einem Aufruf (10 Prozent) und Änderungen in der Datenstruktur (10 Prozent).

Die gängige Umsetzung

Die gängige Umsetzung ist die schrittweise Realisierung derjenigen Schnittstellen, die gerade gebraucht werden. Dies erscheint zunächst relativ einfach. Es gilt Daten, die in einem bestimmten Format vorliegen, in ein System zu importieren. Die Schnittstelle kann also im Rahmen der Systemrealisierung nebenbei umgesetzt werden. Sie wird ausprogrammiert. Damit ist das Problem jedoch nur scheinbar gelöst. Die Qualität der Umsetzung wird erst im laufenden Betrieb sichtbar. Die Wahrscheinlichkeit, dass die einmal spezifizierten Daten und Formate dieselben bleiben, ist sehr klein. Die Wahrscheinlichkeit, dass die Schnittstelle auch mit unvollständigen Daten umgehen kann, ist klein. Die Wahrscheinlichkeit, dass die während der Entwicklung vorliegenden Testdaten den tatsächlichen produktiven Daten entsprechen, ist klein. Die Änderungsresistenz einer ausprogrammierten Schnittstelle ist jedoch gross.



Diese Schritte sind für den typischen Ablauf einer Schnittstellentransaktion notwendig.

Was passiert in einer Schnittstelle?

In einer Schnittstelle zwischen zwei Systemen geschieht aus logischer Sicht eigentlich immer dasselbe. Daten werden aus dem einen System extrahiert, transformiert und/oder konvertiert und anschliessend in das Zielsystem importiert. Selbst ein einfacher Aufruf mit Datenübergabe realisiert diese Mechanismen. Das Extrahieren und Importieren geschieht in einem Schritt, und mögliche Transformationen und Konversionen werden implizit vom aufrufenden System oder vom Zielsystem übernommen. Etwas genauer betrachtet sind folgende Schritte für den typischen Ablauf einer Schnittstellentransaktion notwendig (siehe Grafik):

1. Daten werden aus den Source-System extrahiert.
2. Die extrahierten Daten werden validiert, also auf Vollständigkeit und Korrektheit überprüft.
3. Die Daten werden transformiert, das heisst, es wird derjenige Teil der extrahierten Daten herausgelöst, die vom Target-System tatsächlich benötigt werden.
4. Die Daten werden anschliessend konvertiert, es findet eine Formatumwandlung statt.
5. Die Daten werden validiert, um sicherzustellen, dass das Target-System die Daten auch übernehmen kann.
6. Die Daten werden geladen, also in das Target-System übernommen.

Zusätzlich sind begleitende Mechanismen notwendig. So ist der Transport der Daten zu garantieren, sofern die beiden Systeme physisch getrennt sind, und es sind Monitoring, Logging und Tracing Facilities notwendig, um den Gesamtablauf zu überwachen, Fehler zu analysieren und Laufzeittests zu fahren.

Die Komponenten einer Schnittstelle

Aus dem typischen Ablauf einer Schnittstellentransaktion lassen sind sämtliche Komponenten einer Schnittstelle isolieren.

Extract/Export: Daten werden aus einem Quellsystem (Source System) exportiert, respektive durch das Source-System bereitgestellt. Falls das System einen Service zur Verfügung stellt, so können die Daten darüber angefordert werden.

Validate Source: Die Validierung erfolgt entweder implizit durch die interne Konsistenz des Quellsystems oder jedoch explizit durch die Bereitstellung oder Verfügbarkeit von Metadaten. Eine automatische Prüfung der Daten ist nur dann möglich, wenn eine explizite Datenbeschreibung vorliegt.

Transform: Die Transformation ist für die Anreicherung respektive für die Filterung von Daten verantwortlich. Diese Komponente wird zur Zusammenstellung von Daten für ein Zielsystem aus Daten mehrerer Quellsysteme eingesetzt.

Daniel Liebhart ist Dozent für Informatik an der Hochschule für Technik in Zürich und Solution Manager der Trivadis AG.

Convert: Die Konvertierung umfasst die reine Umwandlung von Daten von einem Format in das andere.

Validate Target: Die Validierung erfolgt entweder implizit durch die interne Konsistenz des Zielsystems oder jedoch explizit durch die Bereitstellung oder Verfügbarkeit von Metadaten. Eine automatische Prüfung der Daten ist nur dann möglich, wenn eine explizite Datenbeschreibung vorliegt.

Load/Import: Daten werden in ein Zielsystem (Target-System) geladen, respektive dadurch bereitgestellt, um vom Target-System selbst importiert zu werden. Falls das System einen Service zur Verfügung stellt, so können die Daten darüber angeliefert werden.

Transport: Der Datentransport wird durch implizite oder explizite Transportmechanismen garantiert. Die meisten Punkt-zu-Punkt-Schnittstellen arbeiten mit impliziten Transportmechanismen. Explizite Transportmechanismen bilden den Datentransport als Prozess ab. Idealerweise sorgt die Transportkomponente auch für die garantierte Lieferung der Daten an das Zielsystem und ist fähig, geeignete Massnahmen zu ergreifen, falls das Zielsystem nicht erreichbar ist.

Monitoring: Die Überwachung des gesamten Ablaufes eines Datentransfers über eine Schnittstelle vom Quellsystem bis hin zum Zielssystem ist insbesondere in einer verteilten Umgebung von zentraler Bedeutung.

Tracing/Logging: Tracing und Logging Facilities erlauben die Protokollierung der einzelnen Schritte des Gesamtablaufes und stellen Mechanismen bereit, um Fehlerprotokolle zu erstellen oder um gezielte Testläufe mit speziellen Daten zu fahren.

Konventionelle Lösungen zur Realisierung von Schnittstellen

Zur Umsetzung von Schnittstellen existieren verschiedene konventionelle Lösungen: die ausprogrammierte Schnittstelle, der Einsatz von Softwarekonvertern, von ETL-Tools oder Frameworks oder aber die Abwicklung über Messaging oder anderen Integrationsinfrastrukturen. Die heute am meisten verbreitete und leider auch fehlerhafteste Lösung ist die Ausprogrammierung einer Schnittstelle. Jede Schnittstelle inklusive alle notwendigen Komponenten wird einzeln ausprogrammiert. Vorteile sind das schnelle Resultat und die Kosten der Umsetzung. Leider ist diese Umsetzung nur für sehr einfache Schnittstellen geeignet. Die Lösung ist änderungsresistent, hat keine logische Strukturierung und ist selten integrierbar in eine bestehende Umgebung.

Für komplexe Punkt-zu-Punkt-Schnittstellen ist der Einsatz von Softwarekonvertern geeignet. Es existiert eine Vielzahl von Produkten auf dem Markt. Softwarekonverter sind logische Maschinen, die einem Compiler sehr ähnlich sind, jedoch nicht die Semantik des zu konvertierenden Inhalts garantieren. Der Konverter führt einen Umwandlungsprozess durch, bei dem es sowohl zu Informationsverlust als auch zur Informationsanreicherung kommen kann. Ein weitgehend normierte Art und Weise, Schnittstellen zu realisieren, hat sich im Bereich Business Intelligence durchgesetzt: ETL (Extract Transport und Load). Diese Schnittstellentechnik realisiert weitgehend das Konvertermodell, erweitert dies jedoch mit Data-Cleansing- und Data-Consolidation-Mechanismen, die für die Schnittstellen zwischen operativen Systemen und einem Data Warehouse notwendig sind. Leider ist dies sehr aufwendig und eignet sich nur für Massendaten. Ein weiterer Ansatz ist der Einsatz von generischen Frameworks zur rationellen Realisierung vieler Interfaces. Der grosse Initialaufwand dieser Lösung lohnt sich nur dann, wenn viele gleichzeitig zu realisierende Schnittstellen umzusetzen sind. Eine standardisierte Lösung für Schnittstellen ist der Einsatz von Integrationsinfrastrukturen. Ein wichtiges Prinzip dieser Middleware-Software ist der Einsatz standardisierter Schnittstellen wie beispielsweise derjenige von Webservices. Ein typischer Anwendungsfall ist der Austausch von Daten zwischen verschiedenen Systemen in einem Unternehmen. Zentrale Informationen, wie beispielsweise Kundendaten und Bestellungen, werden von verschiedensten Applikationen in aktueller Form benötigt.

Der konkrete Anwendungsfall bestimmt den Einsatz der Lösung für den Schnittstellenbau. Allen Lösungen gemeinsam ist jedoch, dass die typischen Komponenten immer vorhanden sind. Ob sie nun explizit als isolierbare Bestandteile oder jedoch impliziert als versteckte Funktionalität vorhanden sind – sie sind da und sie sind unbedingt als solche zu beachten und bewusst zu gestalten. Das hilft, den Aufwand und die Fehler zu reduzieren.

Schnittstellentypen und deren Einsatzgebiet

Neben der Vergegenwärtigung des genauen Ablaufs und der notwendigen logischen Komponenten gibt es einen zweiten zentralen Aspekt, der beim Schnittstellenbau zu beachten ist: die Schnittstellentypen. Davon existieren drei: die methodenorientierte, die meldungsorientierte und die ressourcenorientierte Schnittstelle. Die methodenorientierten Schnittstellen werden auch Control-Interfaces genannt und sind an Funktions- und Methodenaufrufen im Call&Return-Stil orientiert. Sie eignen sich für den Austausch einfacher Daten und auch für die Realisierung von Schnittstellen zu bestehenden Systemen. Meldungsorientierte Schnittstellen tauschen strukturierte Daten als Meldungen aus und werden auch Document Oriented Interfaces genannt. Der Einsatz von Integrationsinfrastrukturen bedeutet oftmals die Umsetzung dieser Art von Schnittstelle, die sich vor allem für den Austausch komplexer Datenstrukturen eignet. Das typische Beispiel einer ressourcenorientierten Schnittstelle sind RESTful Services. Dieser Schnittstellentyp ist relativ neu und aufgrund seiner Flexibilität sehr mächtig. So kann beispielsweise ein Dokument als Ressource angesehen werden. Der Schnittstellenaufruf ist dann lediglich die Anforderung der Ressource. Dieser Aufruf löst auch sämtliche notwenigen Operationen implizit aus und stellt die Daten im richtigen Format für das Zielsystem bereit. Die bewusste Auswahl des richtigen Schnittstellentyps hilft bei der Umsetzung. Allerdings ist darauf zu achten, dass eine Vermischung der Typen in einem System zu Schwierigkeiten führen kann und oftmals nicht einfach zu realisieren ist.

Lösung	Kurzbeschrieb	Vorteile	Nachteile
Ausprogrammierte Schnittstelle	Jede Schnittstelle inklusive alle notwendigen Komponenten wird einzeln ausprogrammiert.	Schnelles Resultat und Geschwindigkeit. Für einfache Schnittstellen geeignet.	Änderungsresistent, keine logische Strukturierung, nicht integrierbar in eine bestehende Umgebung.
Softwarekonverter	Standardarchitektur für die Kernfunktionen Extract/Export, Validate, Transform, Convert	Möglicher Einsatz von Standardprodukten. Für komplexe Point-to-Point-Schnittstellen geeignet.	Realisiert keine begleitenden Mechanismen (Transport, Monitoring, etc.) und Geschwindigkeit.
ETL	Standardmechanismus für die Schnittstellen zwischen operativen Systemen und einem DWH.	Viele Standardprodukte verfügbar. Geeignet für Business Intelligence Infrastrukturen (Massendaten).	Komplexe Realisierung und oft durch den Hersteller eingeschränkte Anwendung.
Frameworks	Generische Frameworks zur rationellen Realisierung vieler Interfaces.	Produktivität. Geeignet für viele gleichzeitig zu realisierende Interfaces.	Grosser Initialaufwand und proprietäre Lösung
Messaging-Infrastrukturen	Eine Middleware wird als Basis für eine Schnittstellenarchitektur eingesetzt	Flexibilität und Standardprodukte mit sehr grossem Funktionsumfang verfügbar. Geeignet für viele gleichzeitig zu realisierende Interfaces.	Kosten, Herstellerabhängigkeit und grosser Umsetzungs- und Betriebsaufwand. Nur für sehr grosse Unternehmen geeignet.

Zur Umsetzung von Schnittstellen existieren verschiedene konventionelle Lösungen, die jeweils Vor- und Nachteile haben.