Case Study Timesoft

Inhaltsverzeichnis

Autor

Hansruedi Tremp, MA, MAS, MABTS, dipl. Wirtschaftsinformatiker, Dozent an der OST Ostschweizer Fachhochschule in St. Gallen.

Didaktische Vorbemerkungen

Diese Fallstudie gibt Ihnen einen authentischen Situationsbezug zur Arbeit als Softwarearchitekt. Sie ermöglicht die Anwendung der theoretisch erarbeiteten Konzepte und vorgestellten Referenzarchitekturen. Die konkreten Aufgabenstellungen fördern Kompetenzen, wie Problemlösungsanalyse, Lösungsbewertung sowie Modellierung von problemadäquaten Softwarearchitekturen in UML.

Einführung

Sie sind Softwarearchitekt in der erfolgreichen AllCons AG. Diese hat sich im deutschsprachigen Markt etabliert und betreut dutzende von Kunden von ihren vier Standorten aus. Das bestehende Zeit- und Spesenrapportierungs- sowie Projektabrechnungssystem genügt schon seit geraumer Zeit nicht mehr. Die neu zu entwickelnde verteilte Softwarelösung soll die internen Bedürfnisse sowie die Anforderungen potenzieller Kunden aus verschiedenen Branchen abdecken. Dabei möchte AllCons die Applikation selbst verwenden und im SaaS-Cloudmodell ihren Kunden anbieten.

Unternehmung

Seit mehr als 20 Jahren implementiert die AllCons erfolgreich ERP-Gesamtlösungen mit einer Standard Business Software sowie unterschiedlichste Business und Management Services mit eigenen Software-Plattformen. Dabei unterstützt sie ihre Kunden mit On Premises Installationen und einem SaaS-Cloudangebot. Weiter bieten sie ihren Kunden eine breite Palette an Dienstleistungen und Services zur Digitalisierung von Geschäftsprozessen. Mit den verschiedenen Geschäftsstellen legen sie Wert auf lokale Verankerung.

IT-Strategie

AllCons hat sich ihre IT-Strategie vor Jahren erarbeitet und überprüft diese in einem jährlichen Strategie-Workshop. Nachfolgend sind einige für die Softwarearchitektur relevanten Eckpunkte erwähnt. Die konsequente Umsetzung einer Serviceorientierten Architektur (SOA) steht ganz zuvorderst. Zu entwickelnde Applikationen sollen konsequent auf die Kundenbedürfnisse und gleichzeitig für den Eigenbedarf ausgerichtet sein. Die optimale Integration mit den Umsystemen ermöglichen offene standardisierte Services.

Clientseitig strebt AllCons die Entwicklung von Mobile-First Web Apps an. Je nach Bedürfnislage sind vorranging Hybrid- oder in Ultima Ratio Native Apps zu entwickeln. Serverseitig setzt AllCons auf Microservices. Die Autonomie der einzelnen Microservices soll möglichst hoch

und die Wahl des Technologie-Stacks optimal auf die auszuführende Funktion abgestimmt sein.

Die Softwareentwicklung erfolgt primär mit eigenen Mitarbeitern und mit einem Netzwerk von Subunternehmen, mit welchen eine langfristige Zusammenarbeit angestrebt wird. Bei den nicht selbst entwickelten Applikationen strebt die AllCons, wenn immer möglich Cloudlösungen mit dem Best-of-Breed-Ansatz an.

Analyse

Nach ersten Iterationen im Requirements Engineering liegt ein Use Case Modell vor, welches die Akteure und Anwendungsfälle aufzeigt.

Use Cases Modell

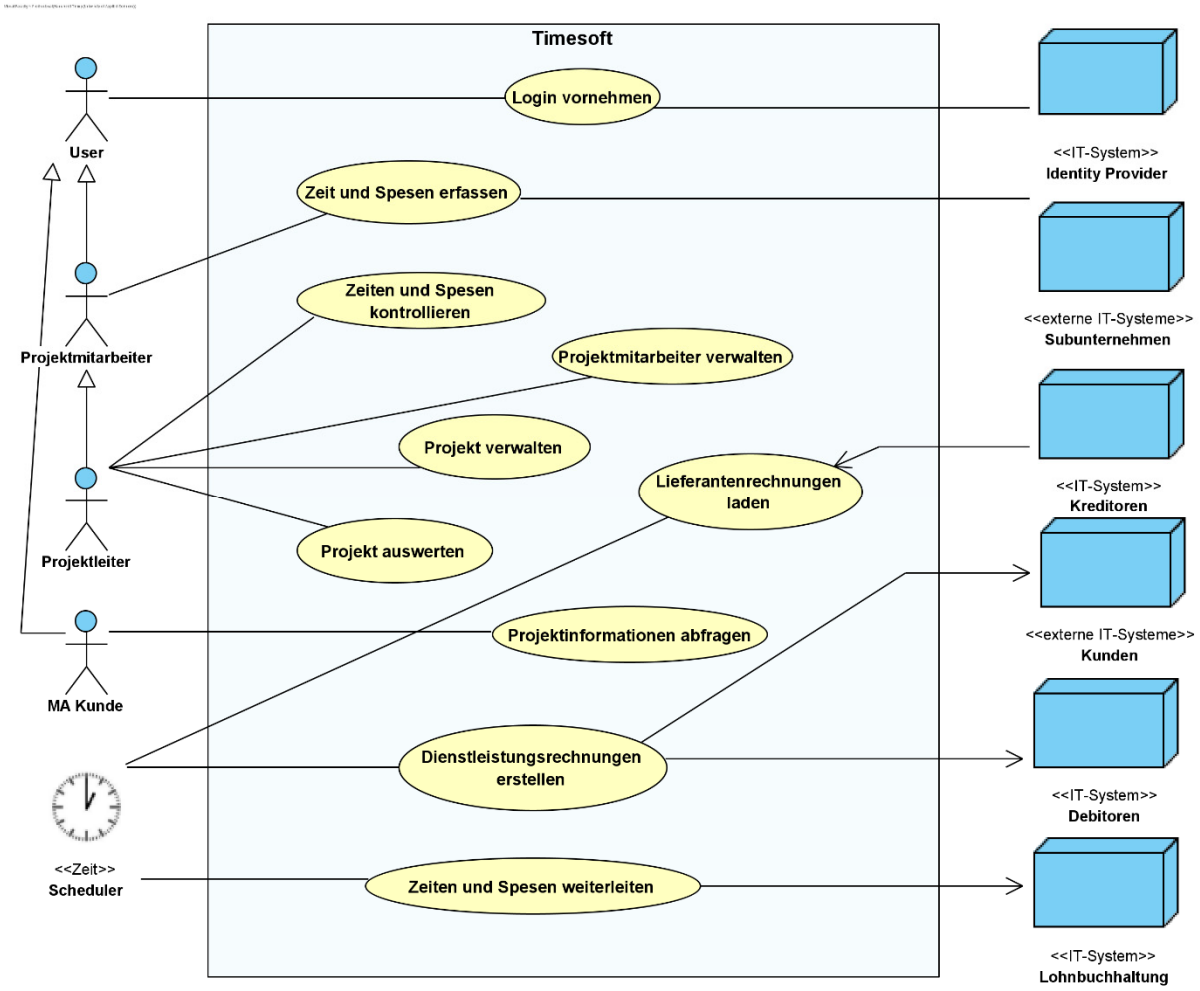

Abb. 1: Use Case Diagramm von Timesoft

UC Login vornehmen

Für die Authentisierung und Autorisierung der User sind die gängigen Identity Provider vorzusehen.

UC Zeit und Spesen erfassen

Die Projektmitarbeiter erfassen die Zeiten und Spesen sowohl inhouse als auch unterwegs. Die Zuordnung zu Kostenstellen und Projekten ist möglichst automatisiert vorzunehmen. Spesenbelege sind einfach zu scannen, auf dessen Inhalt zu analysieren und erkennbare Elemente im Erfassungsformular einzufügen. Die Funktion ist im Intra- und Extranet und für alle gängigen Client-Plattformen als App zur Verfügung zu stellen. Die Projektmitarbeiter müssen auch offline arbeiten können.

Für die Subunternehmen ist zusätzlich ein Webservice für die Übermittlung ihrer Zeiten und Spesen anzubieten.

UC Zeit und Spesen kontrollieren

Der Projektleiter überprüft periodisch die Zeiten und Spesen seines Projektes und gibt diese für die weitere Verarbeitung frei.

UC Projektmitarbeiter verwalten

Der Projektleiter verwaltet die Mitarbeiter seines Projektes. Diese können sowohl eigene Mitarbeiter als auch Mitarbeiter von Subunternehmen sein.

UC Projekt verwalten

Der Projektleiter eröffnet Projekte und pflegt seine Projektdaten.

UC Projekt auswerten

Der Projektleiter wertet mit einem Informations-Cockpit (Intranet Web App) seine Projekt nach unterschiedlichsten Kriterien aus.

Für weitere Ad-Hoc-Auswertungen mit Excel ist ein OLAP-Cube zur Verfügung zu stellen.

UC Lieferantenrechnungen laden

Rechnungen werden in der Kreditoren-Buchhaltung erfasst und dem entsprechenden Projekt zugeordnet. Periodisch holt sich Timesoft die neuen Daten von den Kreditoren.

UC Projektinformationen abfragen

Die erfassten Mitarbeiter des Kunden können bei ihren Projekten verschiedene Informationen abfragen oder Daten herunterladen.

UC Dienstleistungsrechnungen erstellen

Periodisch erstellt Timesoft die Rechnungen für die freigegebenen Stunden und Spesen. Je nach hinterlegter Präferenz im Projekt stellt das System dem Kunden die Rechnung in Papierform oder elektronisch zu. Zu diesem Zweck ist die Schnittstelle zu einem Payment-Provider als Intermediär für die elektronische Zustellung vorgesehen.

Die Debitoren-Buchhaltung erhält die Rechnungsdaten unmittelbar über die angebotene Datenschnittstelle.

UC Zeiten und Spesen weiterleiten

Monatlich übergibt Timesoft die Spesen an die Lohnbuchhaltung zur Auszahlung im nächsten Lohnlauf. Für Mitarbeiter im Stundenlohn übergibt das System die geleisteten Stunden.

Status

Die Projektleitung diskutiert mit dem Entwicklungsteam die Lösungsmöglichkeiten für die neue Software. Dabei melden sich viele Stimmen und drei Szenarien zeichnen sich ab.

Eine Gruppe vertritt die klare Meinung, ganz auf .NET Core zu setzen und den klassischen Mehrschichtenarchitektur-Ansatz zu verfolgen, welcher eine Skalierung auf dem Web-, Business- und Data-Tier ermöglicht. Dies impliziert eine zentrale relationale Datenbank im MS SQL-Server.

Eine zweite Gruppe möchte auf eine Reihe von unabhängigen Microservices setzen. Jedes Team übernimmt dabei die Verantwortung für die Wahl des Technologie-Stacks sowie die weitere Pflege der Software. Jeder Service muss so gebaut sein, dass er für sich alleine voll skalierbar ist.

Eine Dritte Gruppe setzt auf den Ansatz einer Serverless Architecture mit der Azure Cloud. Dies ergeben die höchste Flexibilität und Skalierbarkeit.

Die Projektleitung gibt Ihnen als Softwarearchitekt den Auftrag, die Sache in die Hand zu nehmen und begründete Lösungsvorschläge zu unterbreiten.

Aufgaben

Alle nachfolgenden Aufgaben zu dieser Case Study müssen die oben beschriebene Situation berücksichtigen und die im Anhang befindlichen Referenzarchitekturen korrekt im Kontext anwenden. Die Architekturen sind als UML Komponentendiagramm zu modellieren.

Makroarchitektur

Legen Sie für Timesoft ca. 5 sinnvolle fachliche Microservices fest. Identifizieren Sie die Service Provider und Consumer von Timesoft. Modellieren Sie danach die Makroarchitektur unter Berücksichtigung folgender Punkte:

- die Client-Apps sind noch nicht zu modellieren
- Die Kommunikation mit den Service Consumer bzw. Provider erfolgt über synchrone Web Services
- Die Kommunikation unter den Microservices erfolgt asynchron

Clientseitige Architektur

Legen Sie die clientseitige Architektur basierend auf den oben beschriebenen Anforderungen fest. Verwenden Sie alle vier nachfolgenden Modelle jeweils genau einmal:

- Mobile Native App für Phones und eine Wearable App für SmartWatches (iOS und Android)
- Mobile Cross Platform App für Tablets (iOS, Android und Windows 10)
- Eine Web App für die Intra-/Extranet-Applikation (Projektmitarbeiter)
- Eine PWA/SPA Web App für die Kunden

Ergänzen Sie die Makroarchitektur mit den relevanten Komponenten im Client sowie Presentation Tier und den notwendigen Services im Web API. Begründen Sie kurz Ihre Entscheidungen und geben Sie konkrete Vorteile für die Benutzter der vier verschiedenen Client-Anwendungen.

Mikroarchitekturen der Microservices

Sie haben den Auftrag, drei ausgewählte Microservices in je einem anderen Technologie-Stack zu modellieren. Wählen Sie zu jedem der folgenden Technologie-Stack einen passenden fachlichen Microservice aus:

- **Skriptsprachenbasiert**
- **.NET-basiert**
- **Jakarta EE-basiert**

Entscheiden Sie sich dann je Technologie-Stack für passende Persistenz-Lösungen. Jede Technologie muss mindestens in einem Microservice Verwendung finden, Sie können aber auch in einem Microservice mehr als eine Persistenz-Lösung einsetzen:

- RDBMS (SQL-basiert)
- Dokumentenorientiertes DBMS
- Cache-DB

Begründen Sie die jeweilige Auswahl des Microservice und der Persistenz-Lösung in 1 - 2 ganzen Sätzen. Geben Sie jeweils stichwortartig Vor- und Nachteile des jeweiligen Technologie-Stacks im Anwendungskontext an.

Integrationsarchitektur

Identifizieren Sie alle in der Ausgangslage erwähnten Applikationen und ordnen Sie diese in sinnvolle Kategorien zu.

Schnittstellen

Analysieren Sie die erwähnten Schnittstellen und erstellen Sie eine Tabelle mit folgenden Spalten:

- Provider Applikation (Schnittstellenanbieter)
- Transport-Protokoll: SOAP, REST, AMQP, …
- verwendete E-Message: Normen in Semantik, Struktur und Codierung
- Consumer Applikation

Wählen Sie die Protokolle und Normen anhand der Anforderungen bzw. des beschriebenen Kontextes und begründen Sie kurz Ihre Wahl.

Middleware und Cloud-Serviceprovider

Wählen Sie ausgehend von der Makroarchitektur und den vorhin beschriebenen Schnittstellen adäquate Software bzw. Cloud-Angebote. Begründen Sie kurz Ihre Wahl.

On Premises Middleware:

- Integration Middleware: I-Bus/I-Broker
- API-Management
- Message Broker (MOM)

Cloud Service-Provider:

- Identity-Provider
- Message Service Provider (für E-Mail, SMS usw.)
- EDI Service Provider
- Payment Provider

Integrationsmodell

Modellieren Sie die Integrationsarchitektur als UML-Komponentendiagramm unter Berücksichtigung aller Applikationen sowie ausgewählten Integrationsmiddleware sowie Cloud-Service-Provider.
Schreiben Sie alle Komponenten jeweils so an:

- Typ, konkreter Produktname
- z.B. Integration Middleware, MS BizTalk Server

Schreiben Sie insbesondere auch alle Schnittstellen gemäss den Vorgaben der Referenzarchitektur an.

Bezug zum Lehrbuch

Der Inhalt der Aufgabenstellungen bezieht sich auf das Lehrbuch «Architekturen Verteilter Softwaresysteme», welches im Springer Vieweg in der neuen Reihe «erfolgreich studieren» im Juli 2021 erscheint. Weitere Ausführungen zu den entsprechenden Architekturthematiken sind dort dargelegt.

Quellen

Der Autor der Fallstudie ist Alex Brunner, CTO und Head of Software Engineering, zu Dank verpflichtet für die bereitwillige Unterstützung. Die Inhalte sind an die Situation der ALL CONSULTING AG (www.all-consulting.ch) angelehnt und in einigen Bereichen zu didaktischen Zwecken vereinfacht.

Cloud-Referenzarchitekturen:

https://docs.microsoft.com/en-us/azure/architecture/browse/

https://cloud.google.com/migrate/compute-engine/docs/4.5/concepts/architecture/gcp-reference-architecture?hl=de

https://aws.amazon.com/de/architecture/

Anhang - Referenzarchitekturen

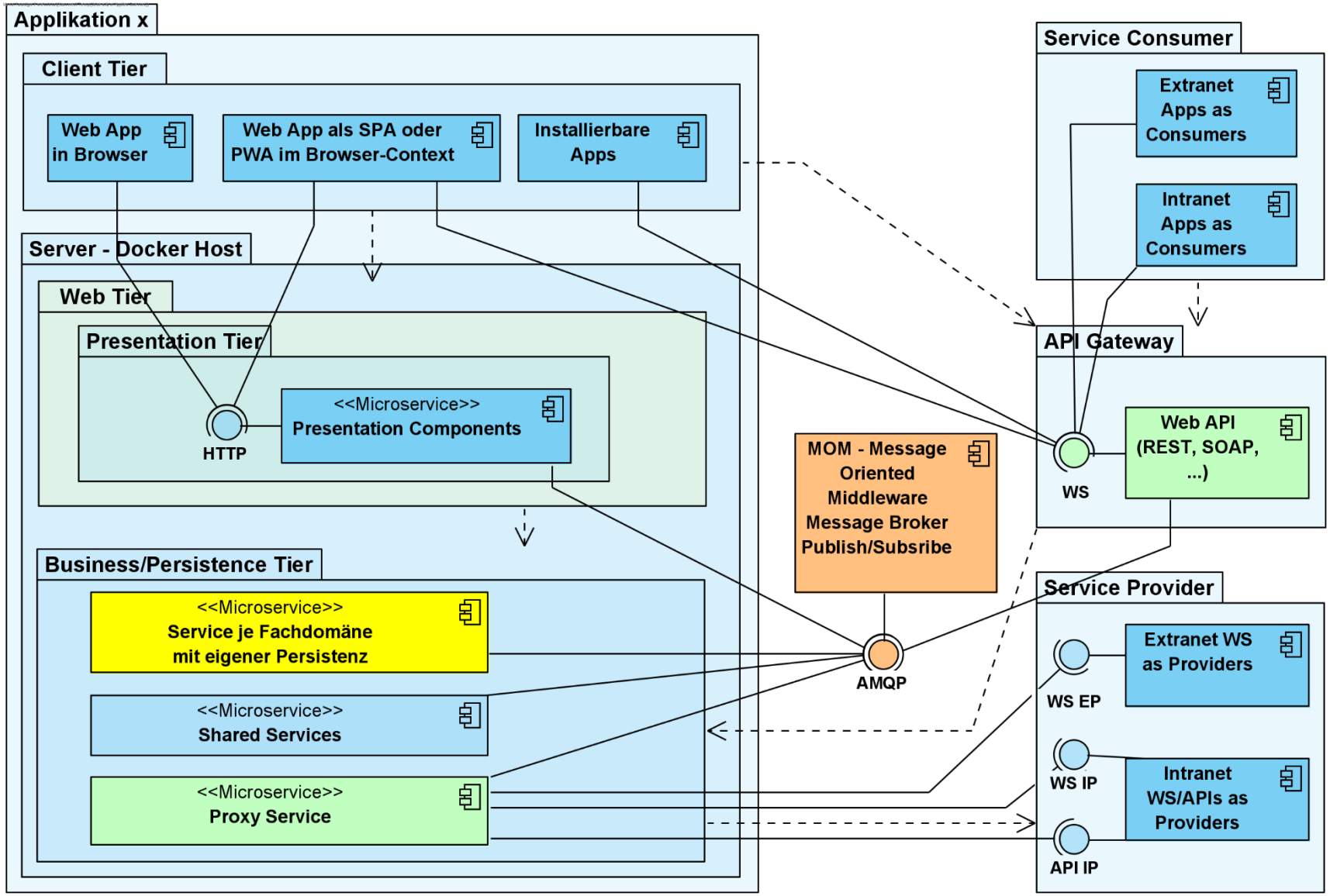

Abb. 2: Referenzarchitektur Makroarchitektur mit Clientseitiger Architektur und Microservices

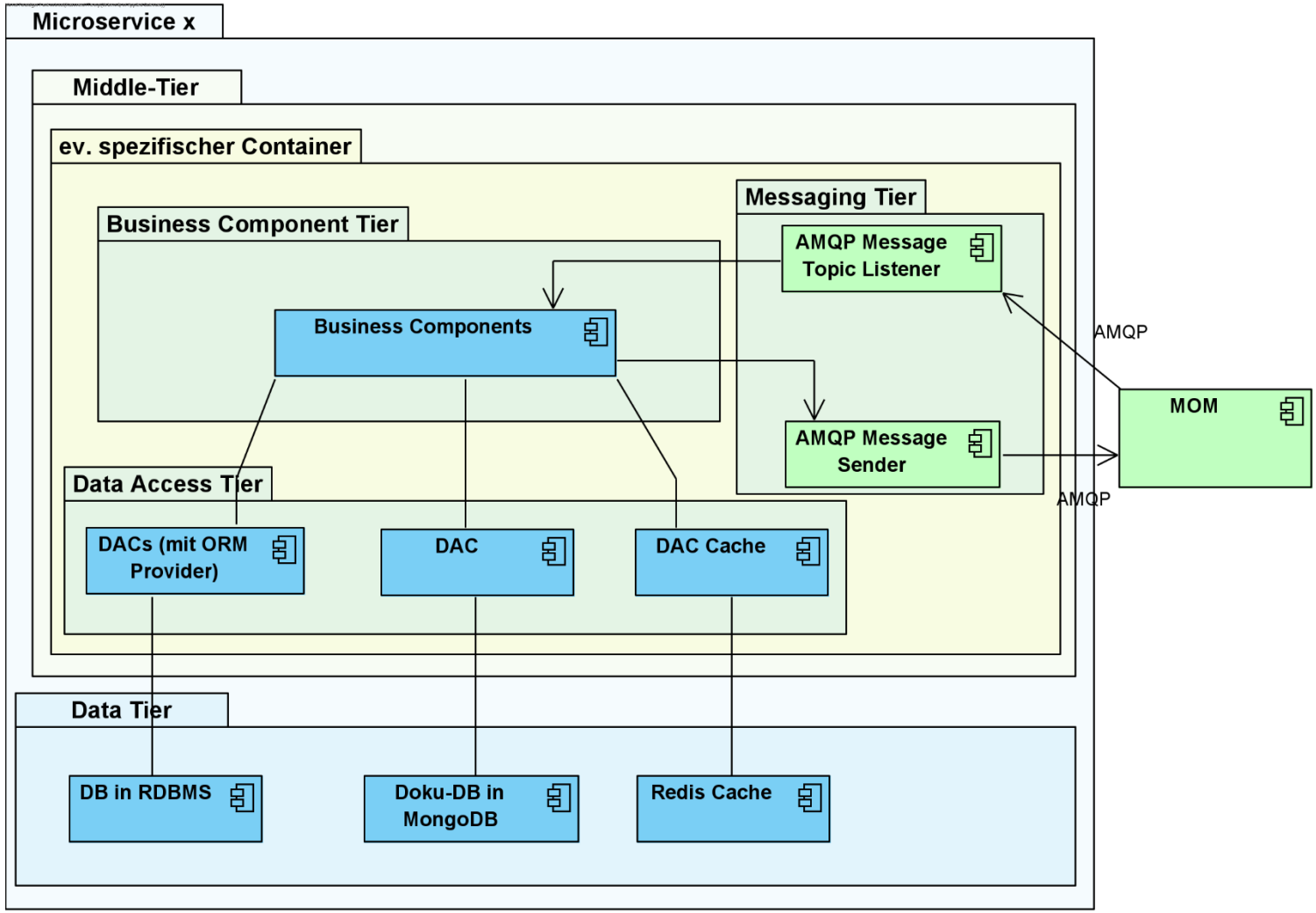

Abb. 3: Referenzarchitektur Mikroarchitektur eines Microservice mit asynchroner Kommunikation

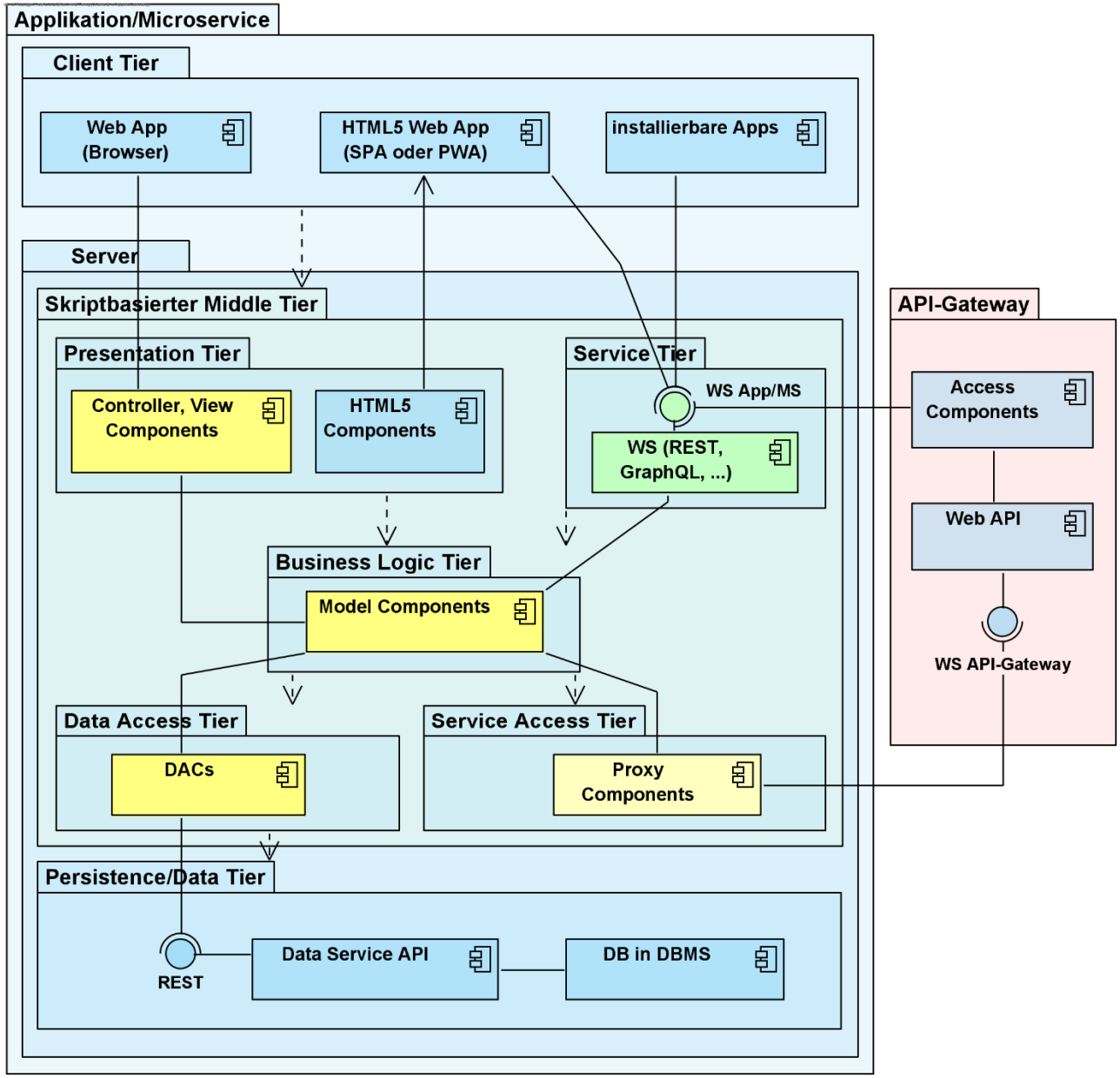

Abb. 4: Referenzarchitektur mit skriptbasierter Sprache

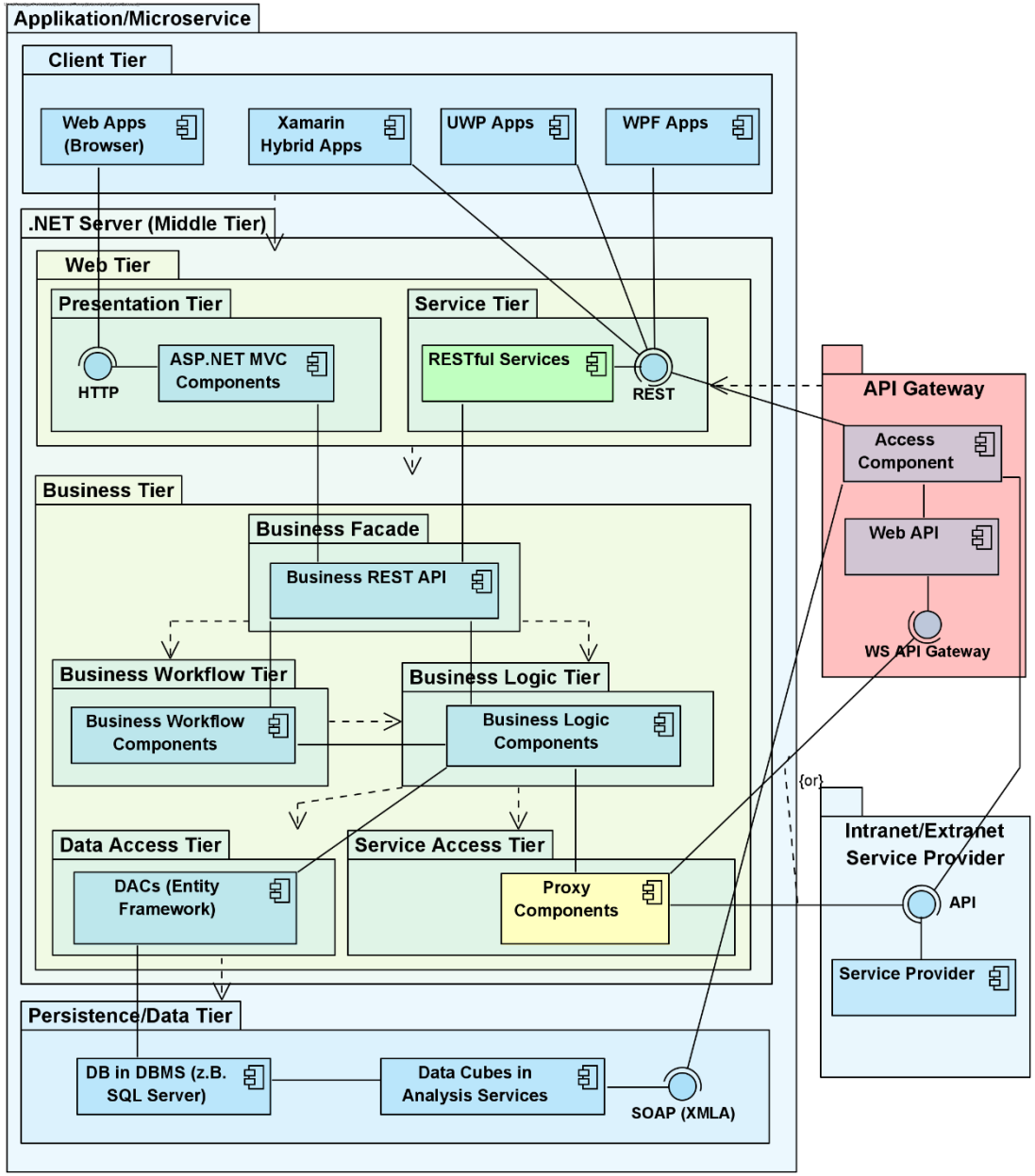

Abb. 5: Referenzarchitektur .NET

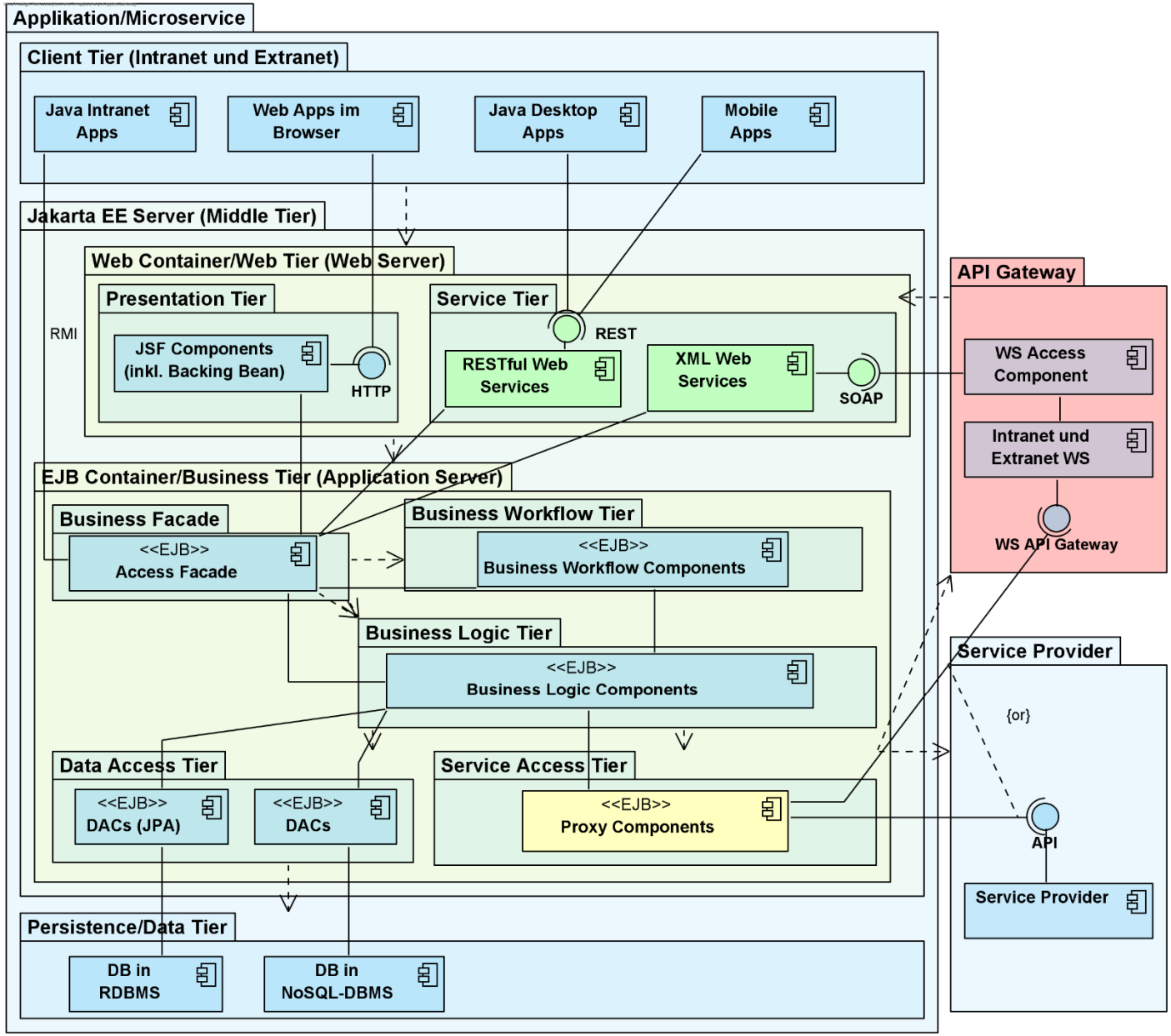

Abb. 6: Referenzarchitektur Jakarta (Java) EE

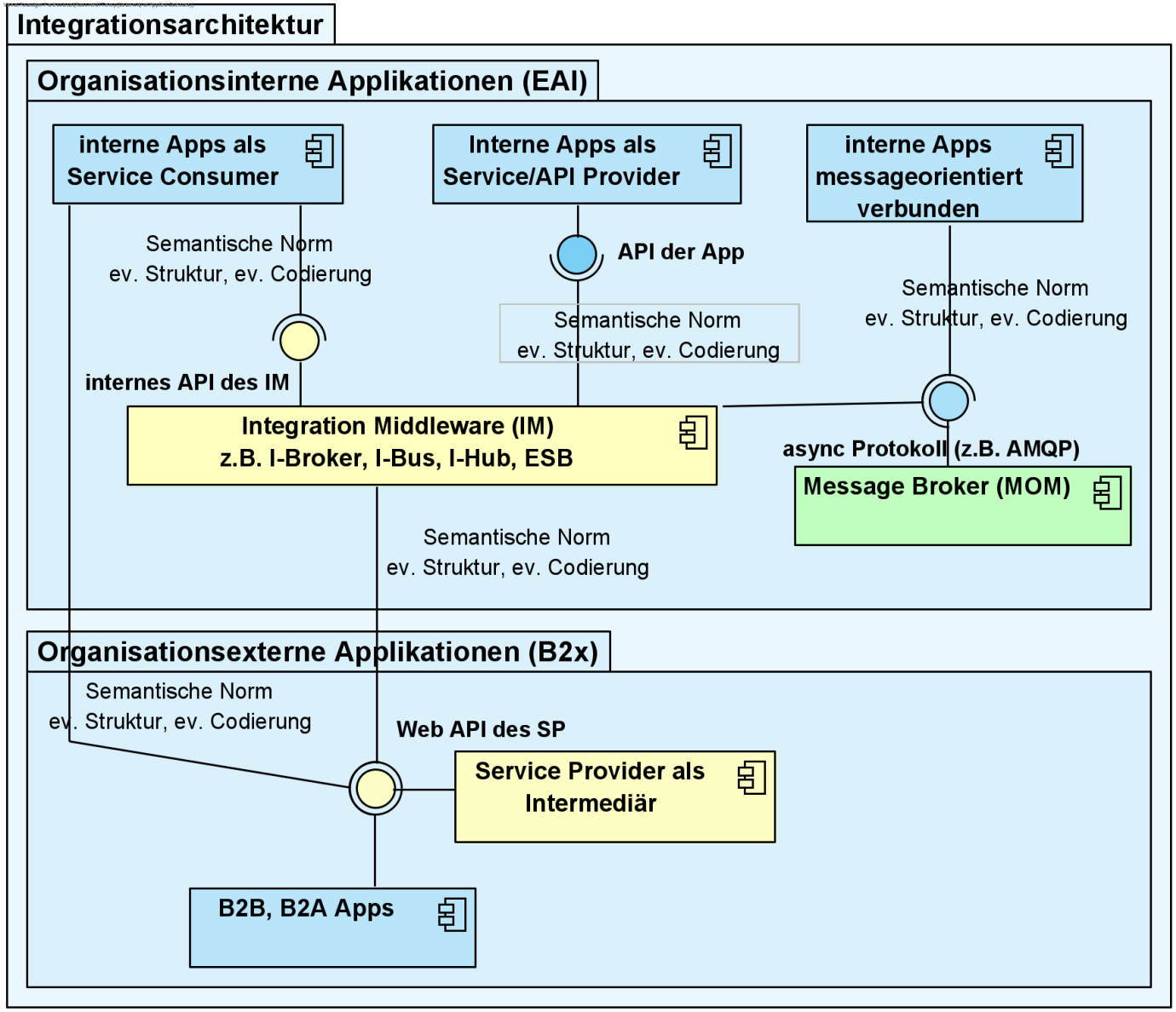

Abb. 7: Referenzarchitektur Applikationsintegration