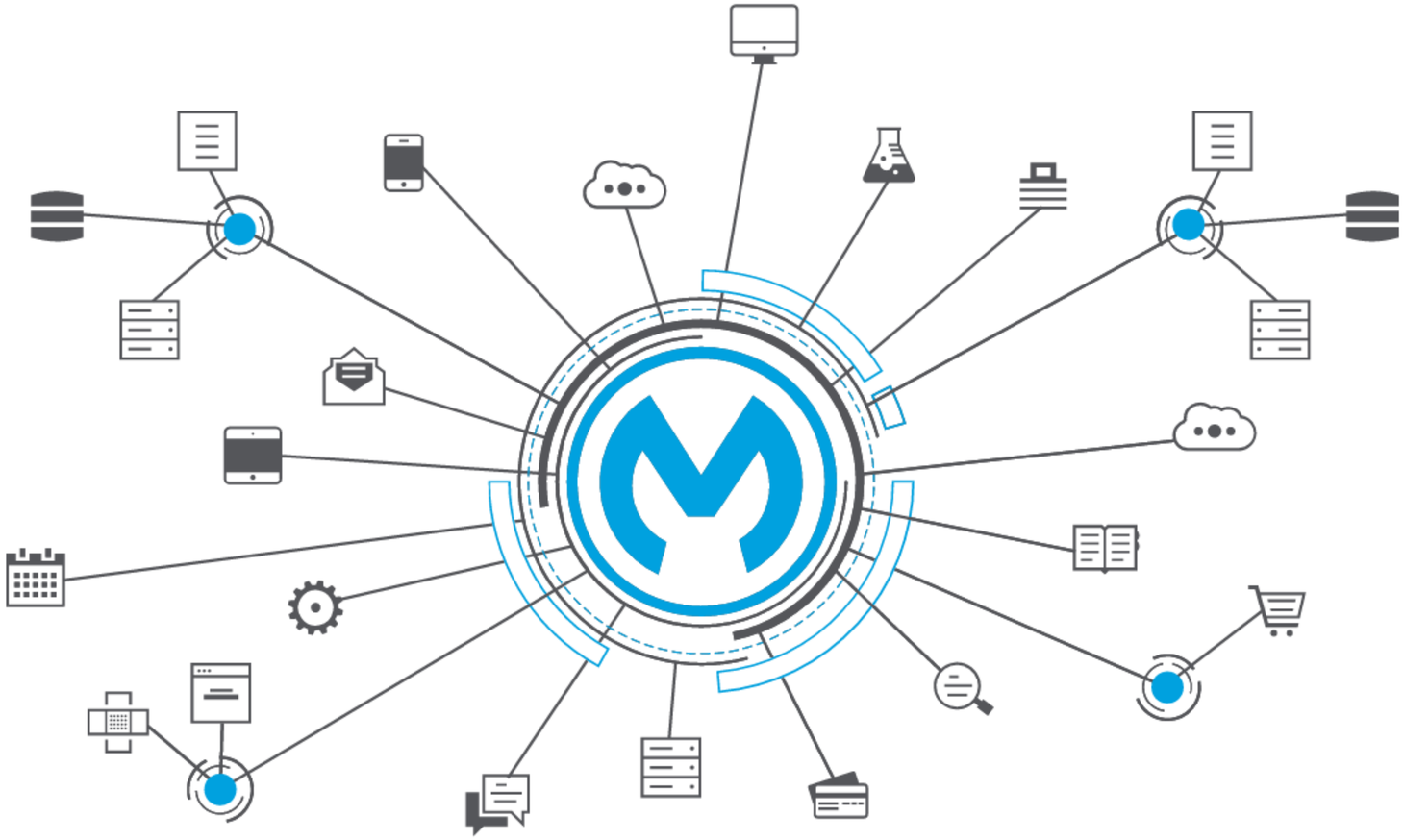
Mulesoft for Java Developers

/

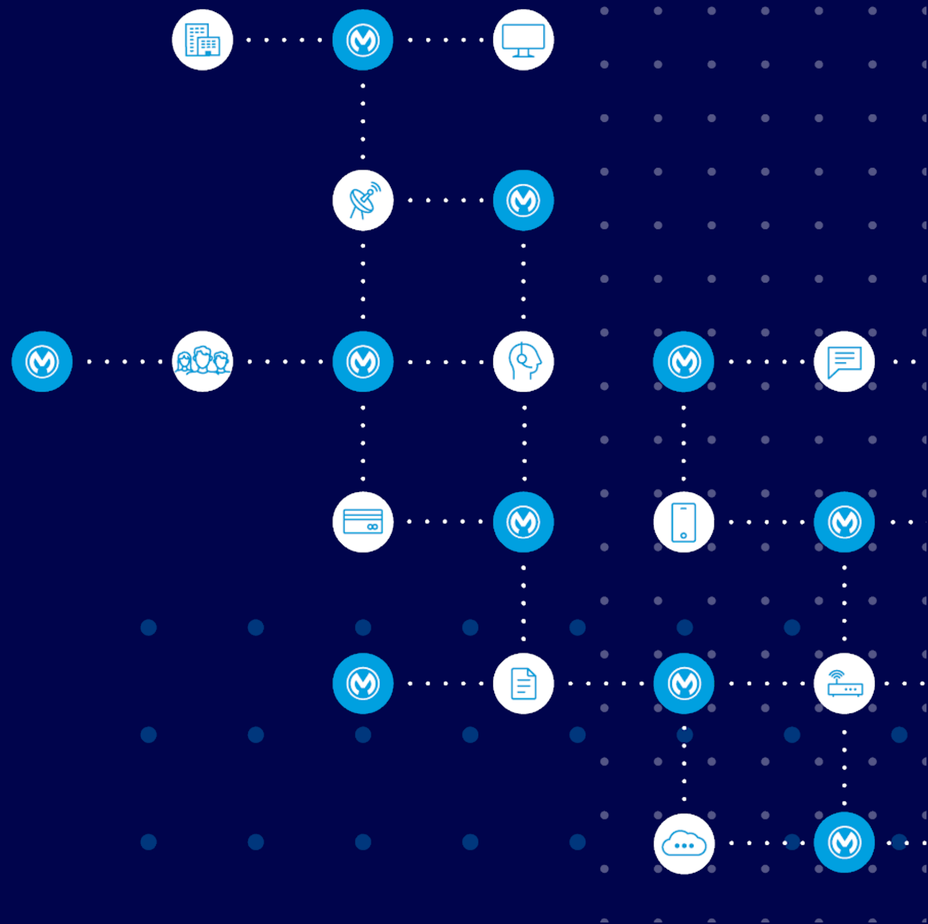


Agenda

- Was sind die Probleme die Firmen - in der Regel - heutzutage in ihrer IT Landschaft haben? Und wie können Sie viele davon lösen?)
- Was ist MuleSoft überhaupt und weswegen hilft es und bei der Lösung dieser Probleme?
  - Wie ist es aufgebaut?
  - Was macht es?
  - Wieso ist es für mich als Java Entwickler so interessant?
- API Led - der etwas andere Architektur Ansatz

MuleSoft

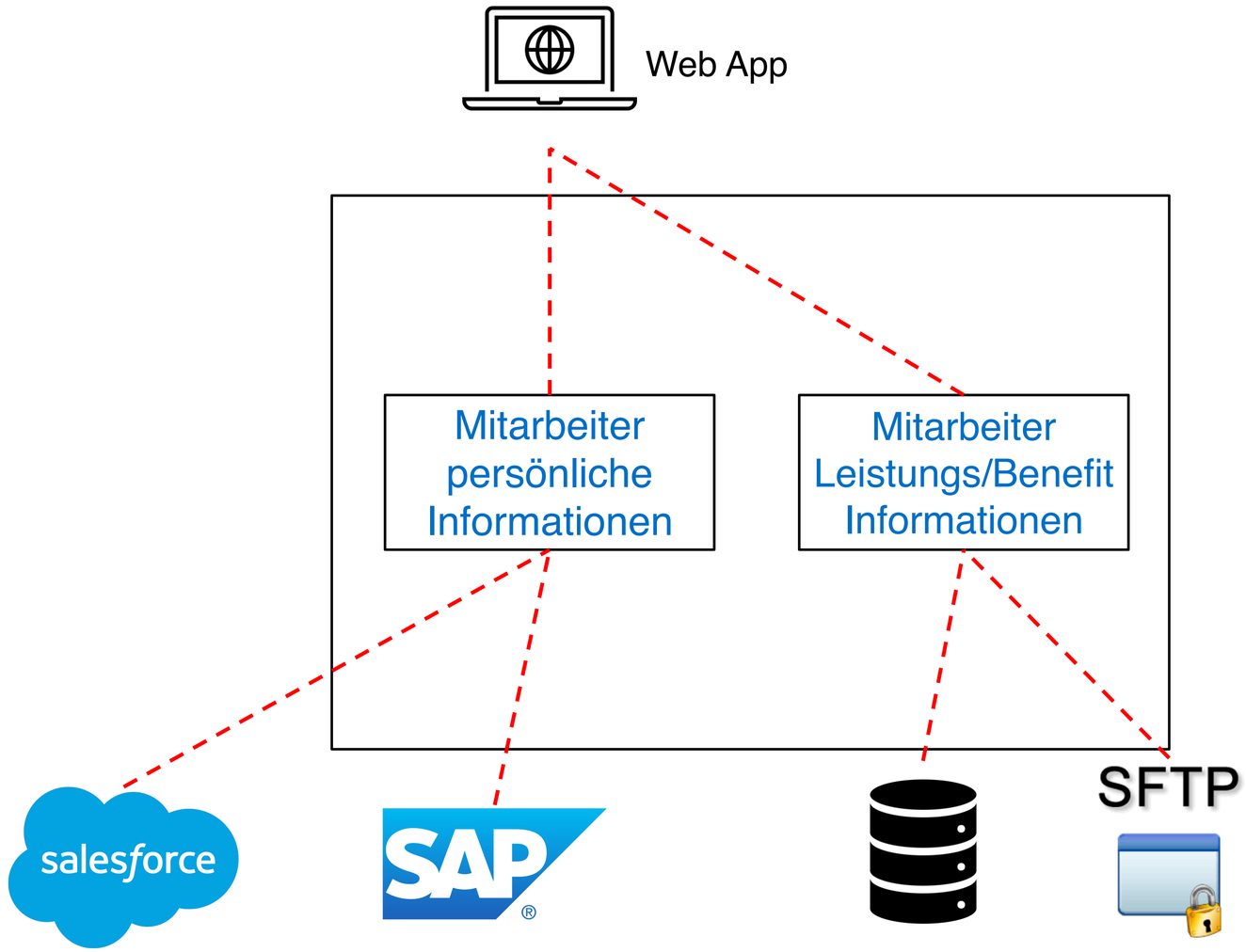
Probleme



Nachteile existierender Point-to-Point Integrationslösungen

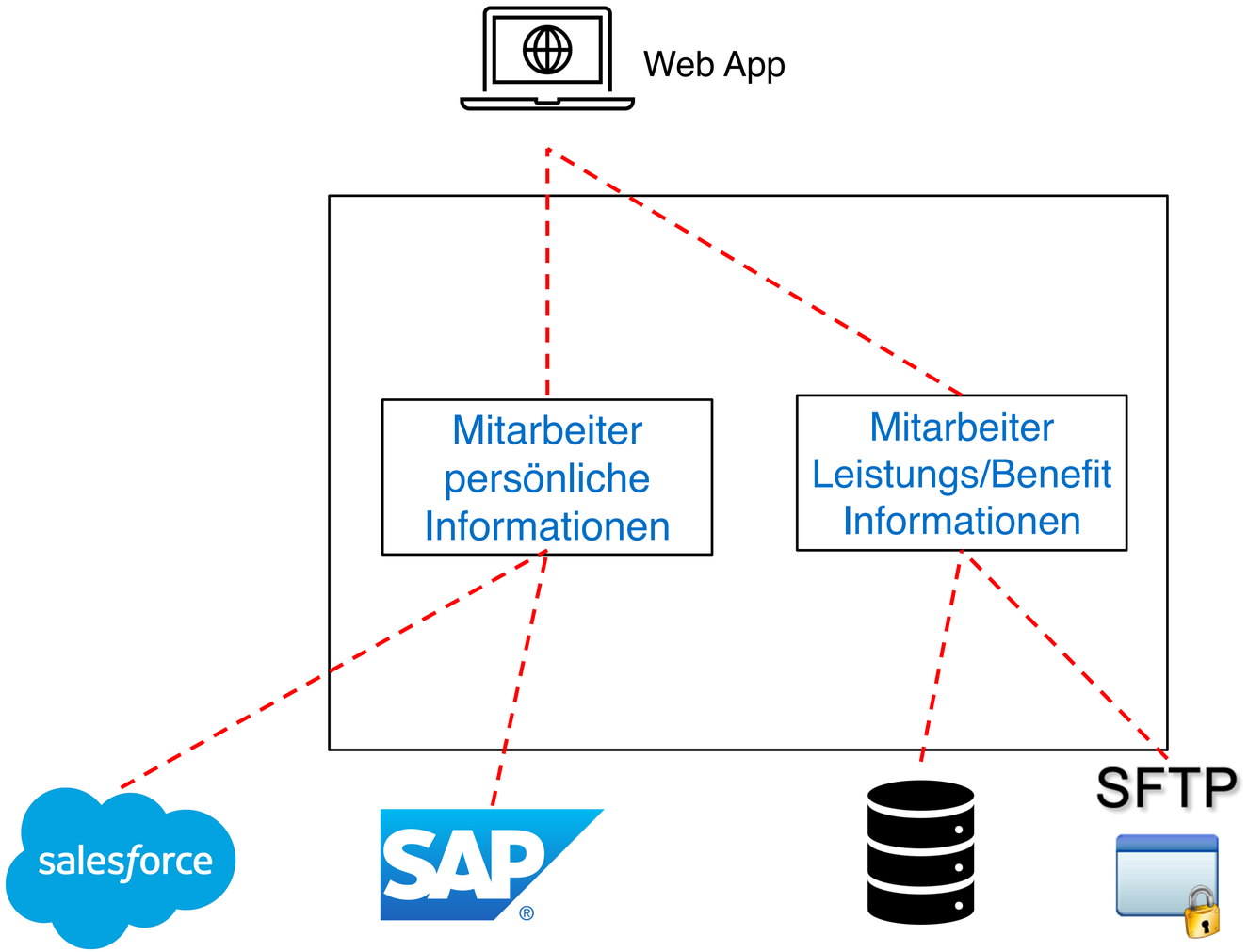
Projekt Übersicht: Eine Web Application die Mitarbeiterinformationen anzeigen soll. In dem Fall zum Beispiel persönliche Daten und Gehaltsinformationen

- Die Daten des Mitarbeiters werden in SAP gespeichert
- Die Informationen für Urlaub und Gehalt liegen im Salesforce System
- Mitarbeiter Werbeaktionen liegen in einer Oracle Datenbank (😨)
- Die Leistungs Analysen jedes Mitarbeiters befinden sich auf einen SFTP server

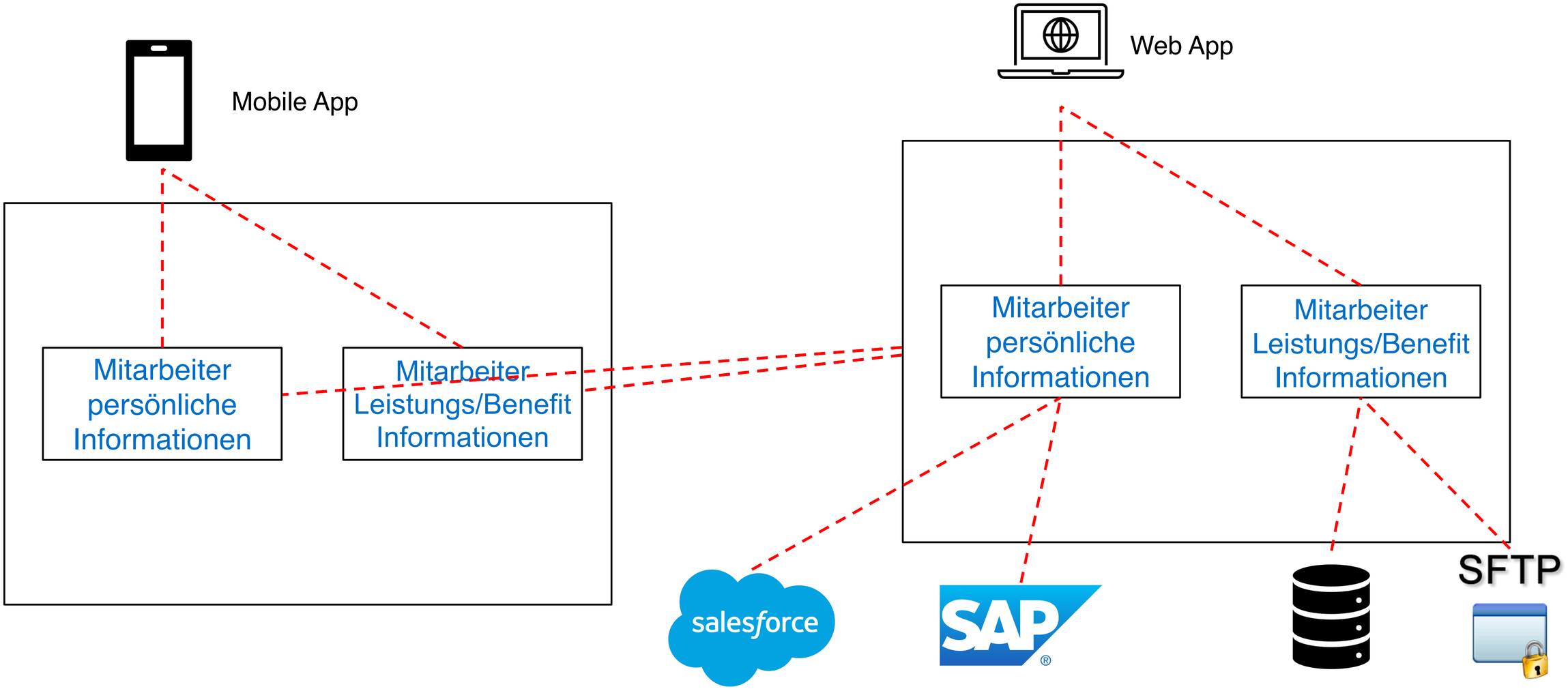


Vorteile und Nachteile existierender Point-to-Point Integrationslösungen

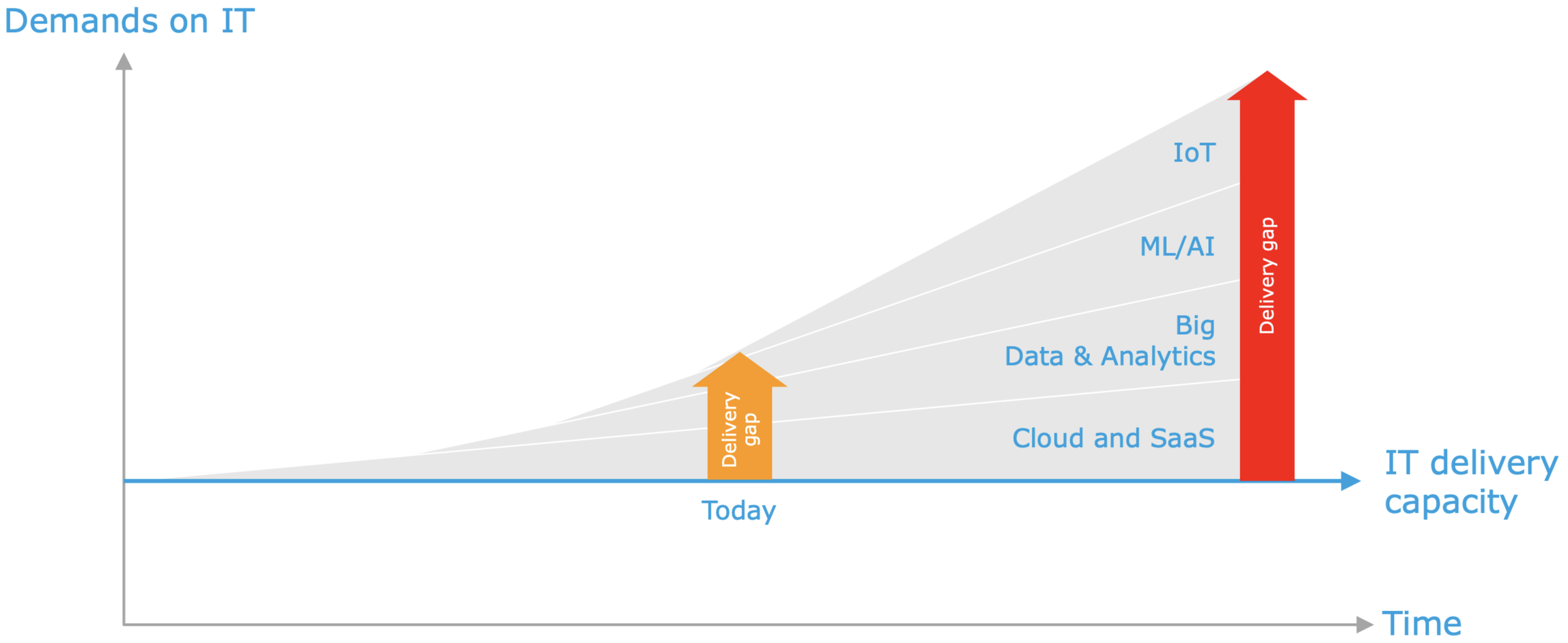
- Pünktlich und im Budget
- Begrenzte Wiederverwendbarkeit
- Eine feste Kopplung der Komponenten
- Schwierig auf Änderungen erweiterbar



6 Monate später

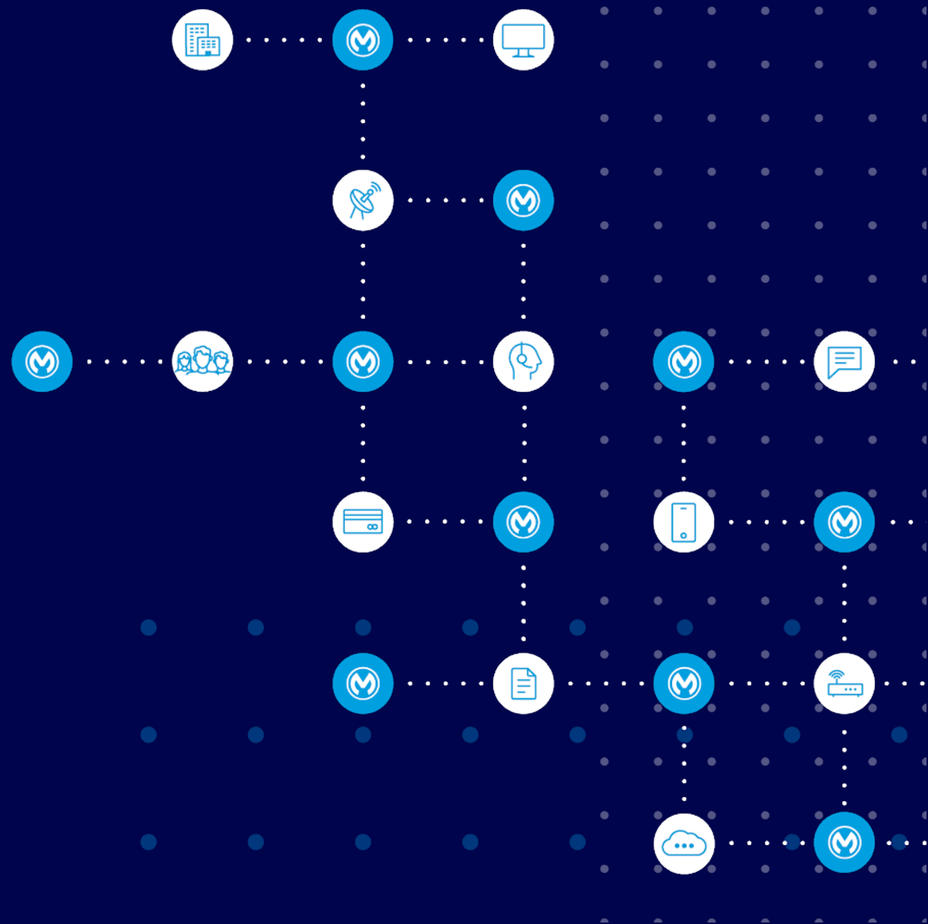


The Current Trend



MuleSoft

Lösung ==> MuleSoft



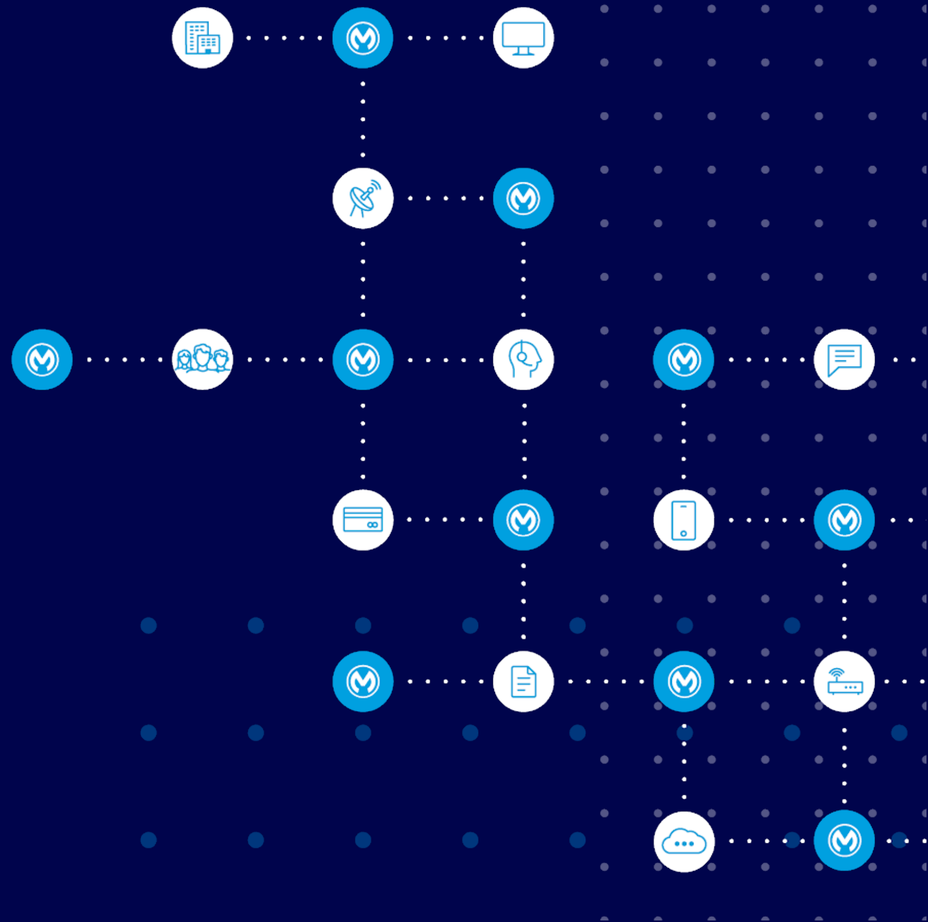
- **Integrationslösung von Salesforce, mit deren Hilfe effizient und schnell Apis geschrieben werden (nach dem WYSIWYG Prinzip)**
- **Die unterschiedlichsten Systeme und Protokolle können genutzt und angebunden werden (eine kleine Übersicht) - als Beispiele**
  - **Protokolle: REST, SOAP**
  - **Datenbanken: MongoDB, IBM, Microsoft SQL**
  - **Programme/Tools/Cloud: Salesforce, SAP, Office365, Amazon AWS**
  - **Queues: IBM Queue, Websphere, JMS Queues**
- **Im Kern ==> Systeme, die sonst nicht miteinander sprechen würden, können über spezielle Transformationen und Anbindungen miteinander arbeiten (über Transformationen) => Dataweave Language**

Warum ist es für mich als Java Entwickler interessant?

- **Basiert auf Java - viele Dinge die man dort kennen und „lieben“ gelernt hat gibt es auch dort**
  - **Maven**
  - **Spring Integration (ESB)**
  - **UnitTests die als Java Anwendung geschrieben werden können**
  - **Stage/Common Properties**
  - **Keystore/TrustStore Anbindung und Security Verwaltung**
  - **Eine Entwicklungsumgebung die quasi eine Erweiterung von Eclipse ist (nein, es gibt leider keine brauchbare IntelliJ Erweiterung 🥲)**

MuleSoft

API-led Architektur

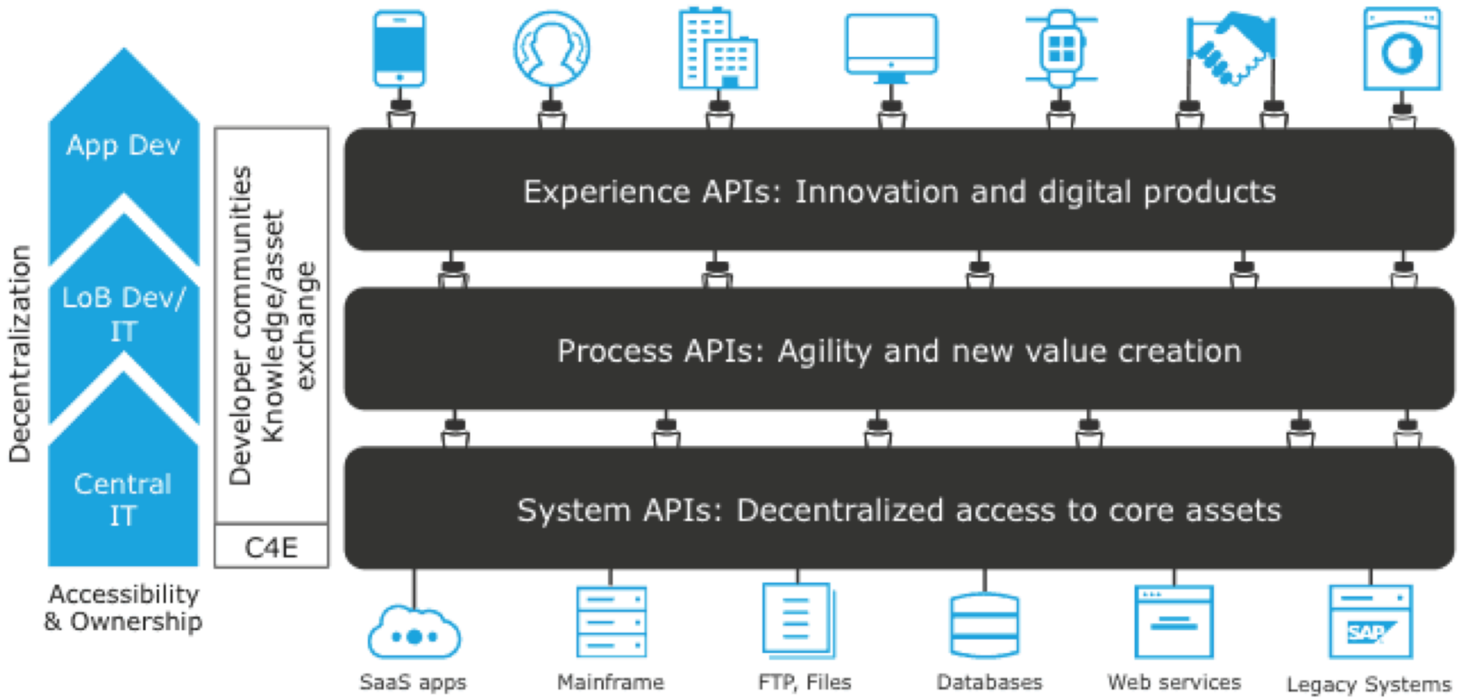


Was ist die API-LED Architektur

System APIs: Greifen auf die Kern Systeme zu und stellen die Daten den darüber liegenden APIs zur Verfügung. Die darüber liegende API muss dabei nicht auf Änderungen oder Komplexitäten der Kern Schnittstelle reagieren oder diese beachten.

Process APIs: Diese APIs kümmern sich um die Geschäftslogik. Sie sammelt die Daten von den darunter liegenden System APIs ein und verarbeitet diese entsprechend der Business Anforderung weiter. Dadurch kann die gleiche Logik von verschiedenen Experience APIs eingebunden und genutzt werden - ohne die dafür notwendige Logik zu kennen.

Experience APIs: Die Experience API greift auf die Process API zu und verarbeitet diese so, das ein Client diese verarbeiten kann



Das Problem mit API-LED Architektur Ansatz lösen