21.10.2026

Vorstellung Teil II xRM-Major-Release „Oktober 2026.10.0“

xRM-Version 2026.10.0



Herzlich willkommen!





Nina
Product Communication



Vinzent
PO Foundation



Lukas
Developer



Ángel
Developer

Ablauf



Präsentation ausgewählter Neuerungen



Q&A



Feedback über MURAL

Agenda

00 Einführung

01 Dokumenten-Editor

02 Digitale Signatur

03 Verlinkung

04 Kontaktsyncronisation

05 LDAP-Modul

06 Q&A

Dokumenten-Editor

Technischer Hintergrund

01

Neuer Dokumenten-Editor

Dokumenten-Editor: docx im Browser bearbeiten

- Neues Modul für die **Bearbeitung** von Dokumenten **direkt im Browser**
- Das Modul startet einen Service auf Basis des **Collabora Editors**
- Mitgelieferte **API zur Konvertierung** in verschiedene Dateiformate
  - Collabora_lib: Collabora.convertTo(pBinary, pToType)
  - Unterstützte Formate und Konvertierungen: Conversion API
- Aktuell im xRM nur für **docx** in Kombination mit dem **docxtemplater**

1

Document-Editor

Startet einen Service

Stellt ein iFrame zur Verwendung bereit

2

Document-Template

Verwaltet Vorlagen

Bietet eine Library zur Ersetzung von Platzhaltern

3

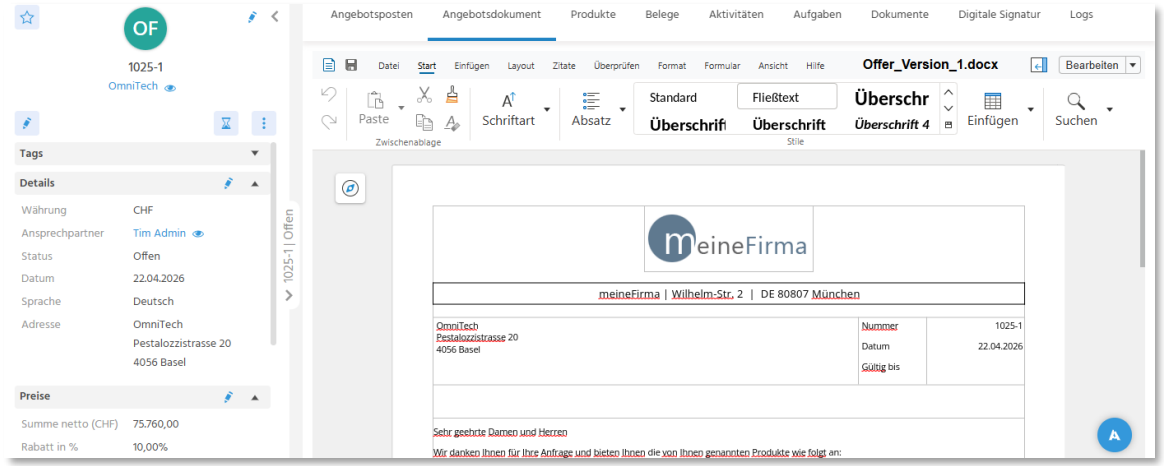
Offer

Wählt eine Vorlage

Übergibt Werte, um Platzhalter zu befüllen

Document-Template und Ablösung der Jasper-Reports





Document-Template: Vorlagen statt Jasper-Reports

- **Document-Template** als zentrales Modul für Vorlagen im System
- Ablauf:
  - **Vorlage mit Platzhaltern im Editor erstellen**
  - **In Angeboten, Belegen usw. per docxtemplater dynamisch befüllen**
- Der Dokumenten-Editor und docxtemplater löst die **Jasper-Reports** ab
- Vorlagen, Angebote und Belege lassen sich vollständig **im Client** bearbeiten, ohne Designer

Migration

- Das neue document-template Modul enthält den **alten und den neuen Context**
- Per **Preference** im einbindenden Modul wird gewählt, ob die neuen Vorlagen oder die alten Jasper-Reports genutzt werden
- Welche Modulversionen mit welcher Logik funktionieren, steht in der document-template **README**

Digitale Signatur

Zukunftssichere Anbindung an Sign8 Signing Portal

02

Zukunftssichere Anbindung an SIGN8 Signing Portal

Einführung

- **Modul Version:** Ab 4.0.0 läuft das Modul nur noch mit SIPO, einen Umschalter gibt es nicht. Wer die Workflow-API braucht, bleibt auf 3.x.
- **Konfiguration:** `custom.sign8.isEnabled`, `systemUrl` (nur der Host, `/api/v2` hängt das Modul selbst an), `apiToken` und `webhookSecret`.
  - **Webhook einrichten:** Im SIPO unter *Team* → *Teameinstellungen* → *Webhooks* und mit demselben Secret wie in der Konfiguration.
  - **API-Token wird nicht mitgeliefert:** Er muss selber im SIPO erstellt und entsprechend im System hinterlegt werden.

- **SIPO-Doku** stehen in der README des Moduls:

  Swagger UI: https://sipo.sign8.eu/api/v2/docs
  OpenAPI-Schema (JSON): https://sipo.sign8.eu/api/v2/openapi.json
  Konfiguration in SIPO: Dokumentation

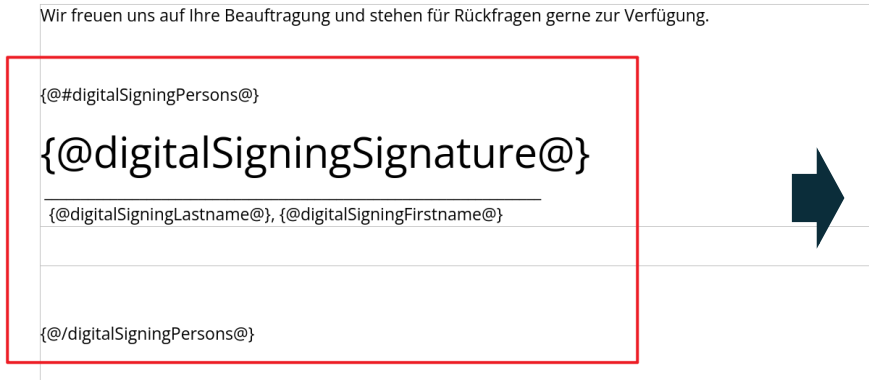
  Wissens: Demosystem - Digitale Signatur mit SIGN8 | Wissen | Neon Prod

Quelle: Grafik und Logo SIGN8

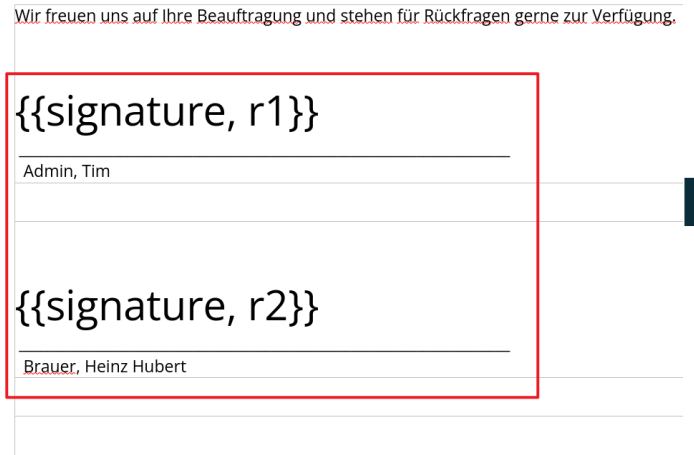
Dokumente

- **Standardweg ist der Dokumenten-Editor:** Reports werden nicht unterstützt, können im Projekt aber individuell eingebunden werden.
- **Ein Platzhalter je Unterzeichner:** Ohne Platzhalter lehnt SIGN8 ab mit „Signers must have at least one signature field“. Platzhalter ist wie folgt {{signature, rN}}
- **Platzhalter bestimmt das Feld:** Die Schriftgröße bestimmt die Feldgröße, und der Platzhalter darf nicht umbrechen.

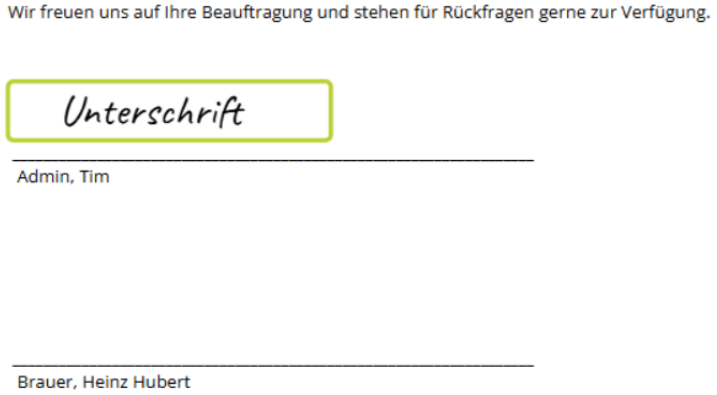
Vorlage



Angebot



SIPO



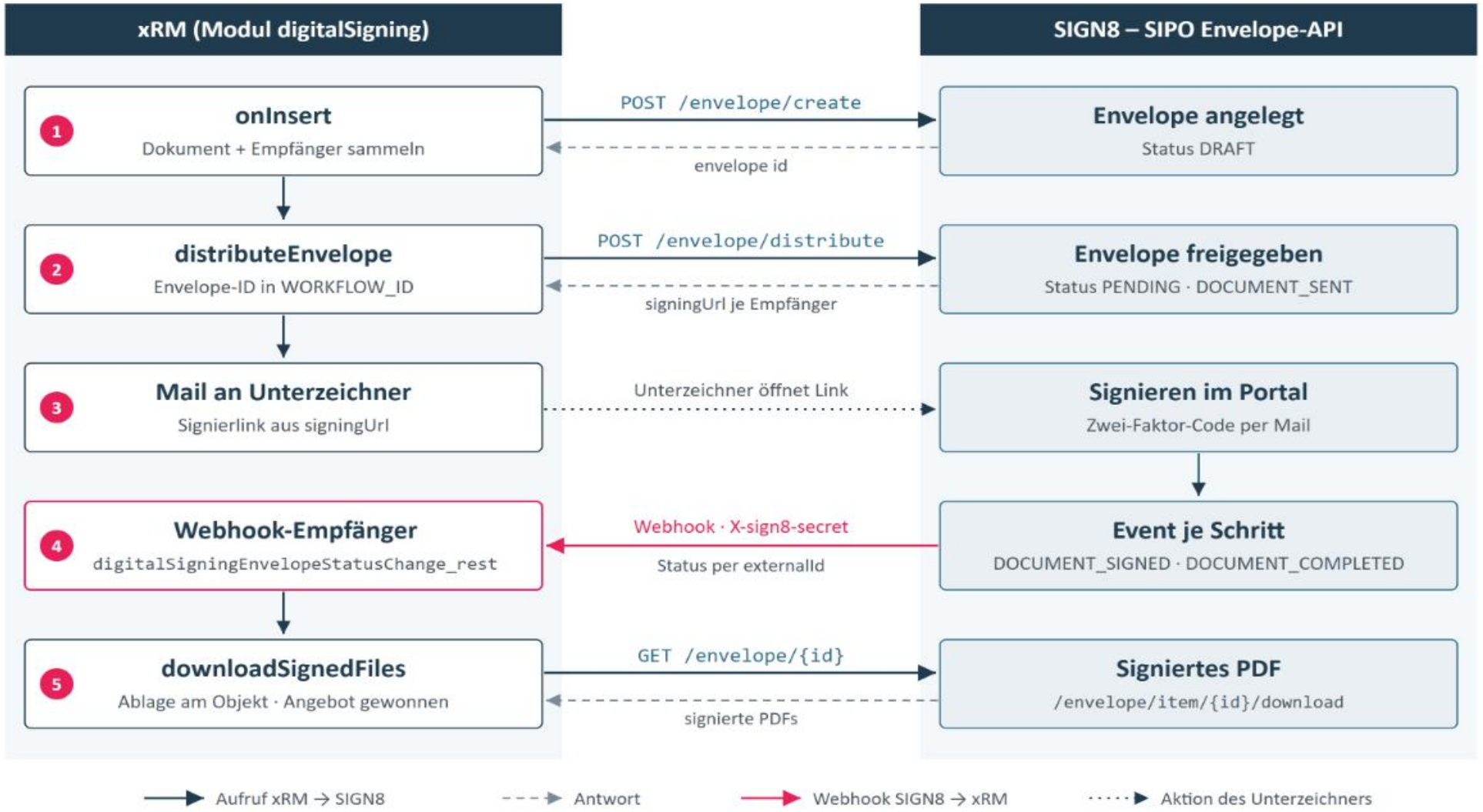
Was hat sich gegenüber v1 geändert?

Workflow-API (v1, bis Modulversion 3.x) im Vergleich zur SIPO Envelope-API (ab 4.0.0)

Thema	v1 – Workflow-API	SIPO – Envelope-API
Ablauf	Fünf Aufrufe: Workflow anlegen → Datei hochladen → Unterzeichner anlegen → Signaturen setzen → Workflow starten	Zwei Aufrufe: createEnvelope (Dokument und Empfänger in einem Upload, als manuell gebautes Multipart*) → distributeEnvelope *SIGN8 läuft auf Node 22 (undici 6.x),Deshalb baut _postMultipartManual. Der Workaround kann entfallen, sobald SIGN8 auf undici ≥ 7.17.0 (Node 24) aktualisiert.
Authentifizierung	x-api-key + secret	Ein Authorization-Header mit dem Team-Token
Signaturfelder	Feste Koordinaten (xfdf) im Code	Platzhalter {{signature, rN}} im Dokument – SIGN8 erzeugt die Felder daraus
Signaturniveau	Einfach oder fortgeschritten (AES)	SES, AES oder QES je Vorgang; Aktuell im xRM Angebot: SES mit Zwei-Faktor-Code
Rückkanal	Webhook-URL je Workflow, zwei Empfänger-Prozesse	Ein Webhook in SIGN8, ein Empfänger-Prozess - Zuordnung über externalId - abgesichert per Secret
Signiertes Dokument	Lädt das Projekt selbst im Finished-Service	Lädt und legt das Modul selbst ab, setzt das Angebot auf „gewonnen“
Code-Struktur	Alles in DigitalSigning_lib	Eigene SIPO_lib

Wie läuft eine Signaturanfrage technisch ab?

Vom Klick auf „Signatur anfordern“ bis zum signierten PDF am Angebot



Verlinkung

Termine per Link-Feld einbinden

03

Termine per Link-Feld einbinden

Problem

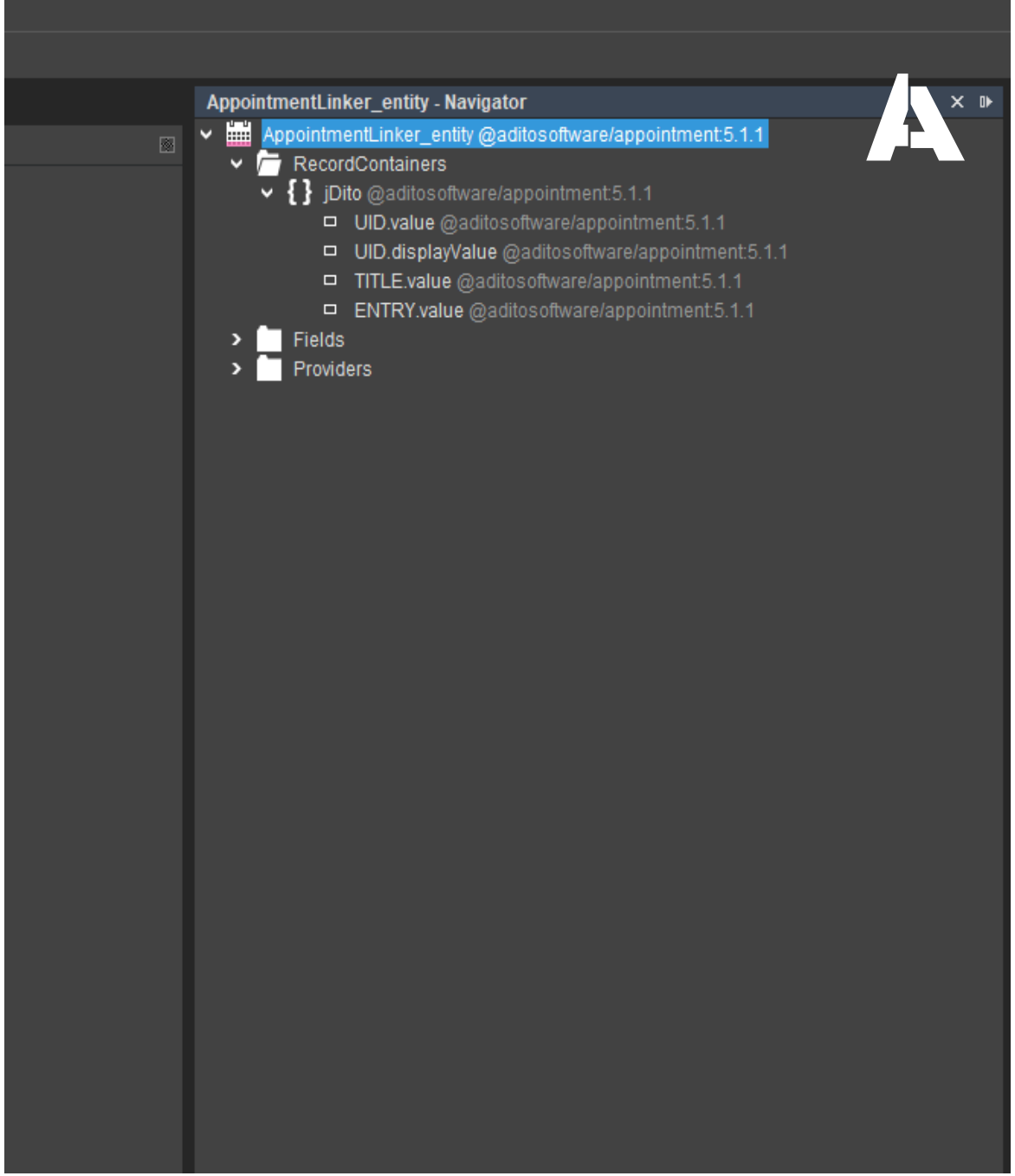
Termine ließen sich nicht konsistent wie andere Objekte per Link-Feld in Projekte einbinden.

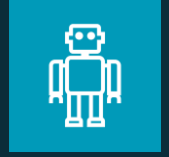
Lösung

- Neue Zwischen-Entität AppointmentLinker_entity als Sprungziel
- Diese Entität ermittelt aus einer UID die drei nötigen Parameter
  - Entry_param
  - MasterEntry_param
  - CalenderOwner_param
- Der normale Termin-Context öffnet sich
- Die Anzeige eines verlinkten Termins erfolgt im gewohnten Format, z. B. „08.09.2026 | 15:13 – 16:13 Uhr"

Vorteil

- Einheitliches Vorgehen für Customizer: Termine werden wie jedes andere verlinkte Objekt eingebunden
- Für User:
  - Vertraute Darstellung für den User
  - Direkter Sprung aus dem Link-Feld in den Termin





Kontaktsynchronisation

Zuverlässige Synchronisierung großer Kontaktfilter unter MariaDB

04

Zuverlässige Synchronisierung großer Kontaktfilter unter MariaDB

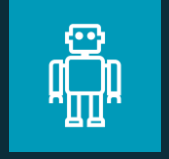
Problem:

Zuvor konnte die bei der Synchronisierung erzeugte SQL-Anweisung bei großen Datenmengen so umfangreich werden, dass MariaDB sie nicht mehr verarbeiten konnte, die Synchronisierung schlug dann fehl. Außerdem wurden Preferences nicht mitgeliefert, sondern mussten manuell angelegt werden.

Unsere Lösung:

- Nutzung von Batching
- Lange WHERE mit NOT_EQUAL für Löschung der Kontakte wurde durch IN ersetzt, weil dieser inperformant war
- Die benötigten **Preferences** für den Kontakte-Sync werden nun inklusive Beschreibung und Default-Wert direkt mitgeliefert und müssen nicht mehr manuell angelegt werden.
  - Microsoft-graph-api: BaseURL, version, defaultAuthAlias, Scope
  - Microsoft-graph-contactsync: active, syncSize, folderName
- Der Kontakte-Sync ist **standardmäßig deaktiviert**, um einen versehentlichen Sync von Testdaten in produktive Exchange-Umgebungen zu vermeiden
- Über den neuen Service **graphSyncUserIdList_service** lässt sich der Sync zudem optional auf einzelne User einschränken, was das gezielte Debugging deutlich erleichtert.

Microsoft Graph API



LDAP-Modul

Projektspezifisch erweiterbar ohne Customizing-Fork

05

LDAP-Modul: Projektspezifisch erweiterbar ohne Customizing-Fork

Problem:

Das LDAP-Modul ließ sich bisher nicht über Services anpassen, was für zusätzliche Entwicklungsaufwände beim Customizing des Moduls sorgte.

Unsere Lösung:

- Folgende Bereiche lassen sich nun über eigene, projektspezifische Erweiterungen (Services) anpassen, statt **activeDirectoryImport_serverProcess** direkt zu verändern:
  - Rollenzuordnung
  - Feld-Mapping (LDAP-Attribute)
  - Kommunikationstypen
  - Liste zu ignorierender User
  - zusätzliche Suchkriterien für bestehende ADITO-User
- CONTACTID des eigenen Unternehmens wird nicht mehr als Platzhalterwert im Code hinterlegt, sondern aus der bereits etablierten **Projekt-Preference** myCompany.contactID gelesen.
- Service, die angepasst werden können, haben eine ordentliche **Dokumentation** hinterlegt, welche Informationen übergeben werden können

Q&A

06


ADITO
Einzigartige Beziehungen