



# Generische Software

SCHWETER, Clemens

25. Oktober 2023

# Vorstellung

---

## Clemens Schweter @ Alstom

- 2004 Informatik Diplom an der TUB
- 18 Jahre als Freelancer für diverse Unternehmen tätig, u.a. für Alstom
  - Siemens, Thales, Bosch, Wacker, HDI, ...
- Seit 2 Jahren festangestellt für Alstom
- 2010 Plattform „DaMaTo“ entwickelt
- 2011 Plattform „DAISY“ entwickelt



# Wir sind dort, wo Mobilität gefragt ist

Mehr als  
**80.000**  
Mitarbeiter:innen  
weltweit

**70**  
Länder

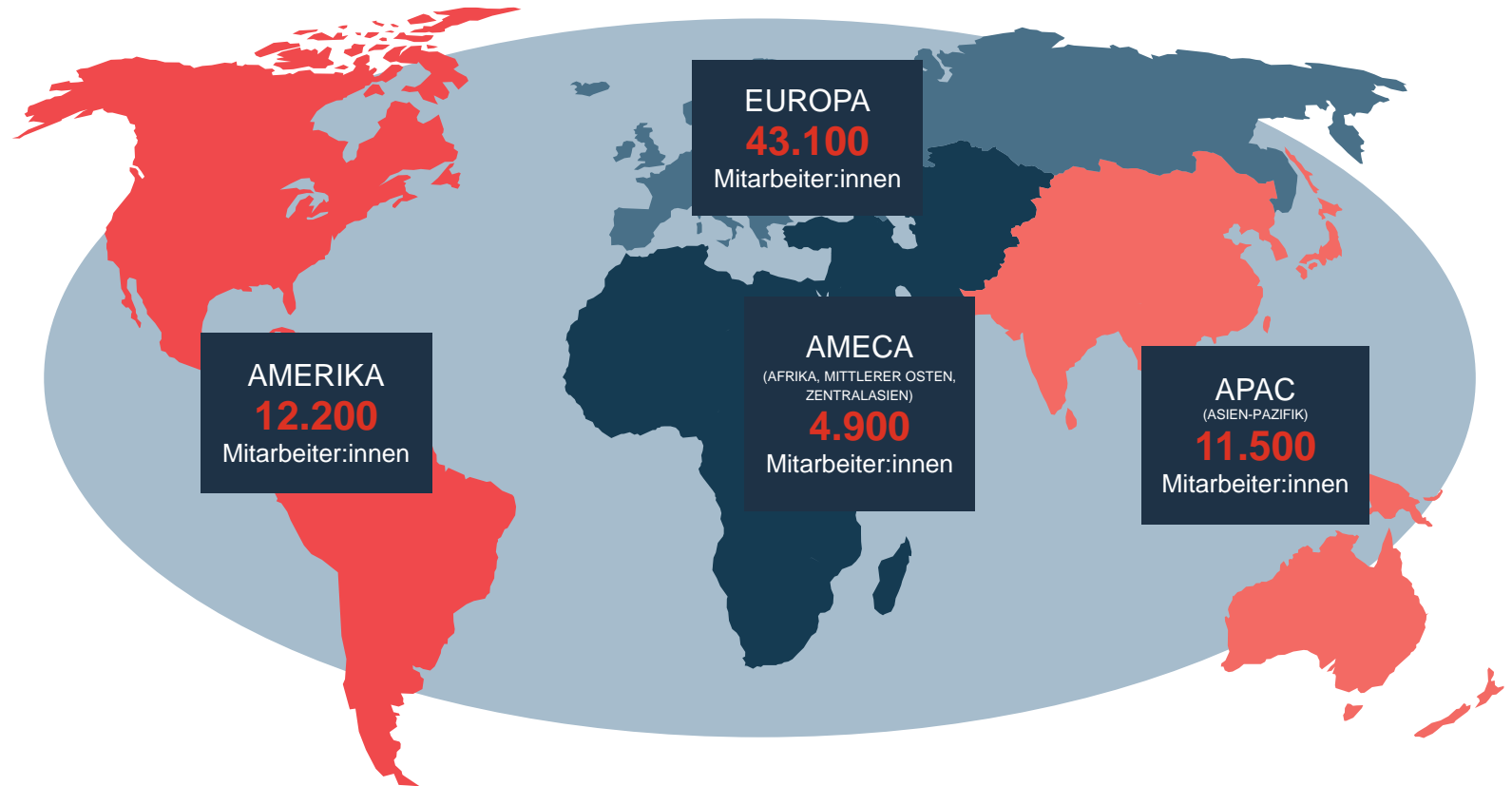
Über  
**250** Standorte

Mehr als **150.000** Fahrzeuge  
im Einsatz

**17.500**  
Ingenieur:innen

Über  
**9.500**  
Patente

Partner von  
mehr als  
**300** Städten



# Ein komplettes Produktportfolio zum Vorteil unserer Kunden

## Schienenfahrzeuge & Komponenten



Peplemover und Monorail

Straßen- und Stadtbahnen

U-Bahnen

Nahverkehrszüge

Regional- und Intercity-Züge

Hochgeschwindigkeitszüge

Lokomotiven

Komponenten

## Digitale & Integrierte Systeme



Signaltechnik für den Nahverkehr

Signaltechnik für den Fernverkehr

Schlüsselfertige Lösungen

Smart Mobility

Cybersicherheit

Serviceleistungen Signaltechnik und  
Infrastruktur

Infrastruktur und Telekommunikation

## Serviceleistungen



Fahrzeuginstandhaltung

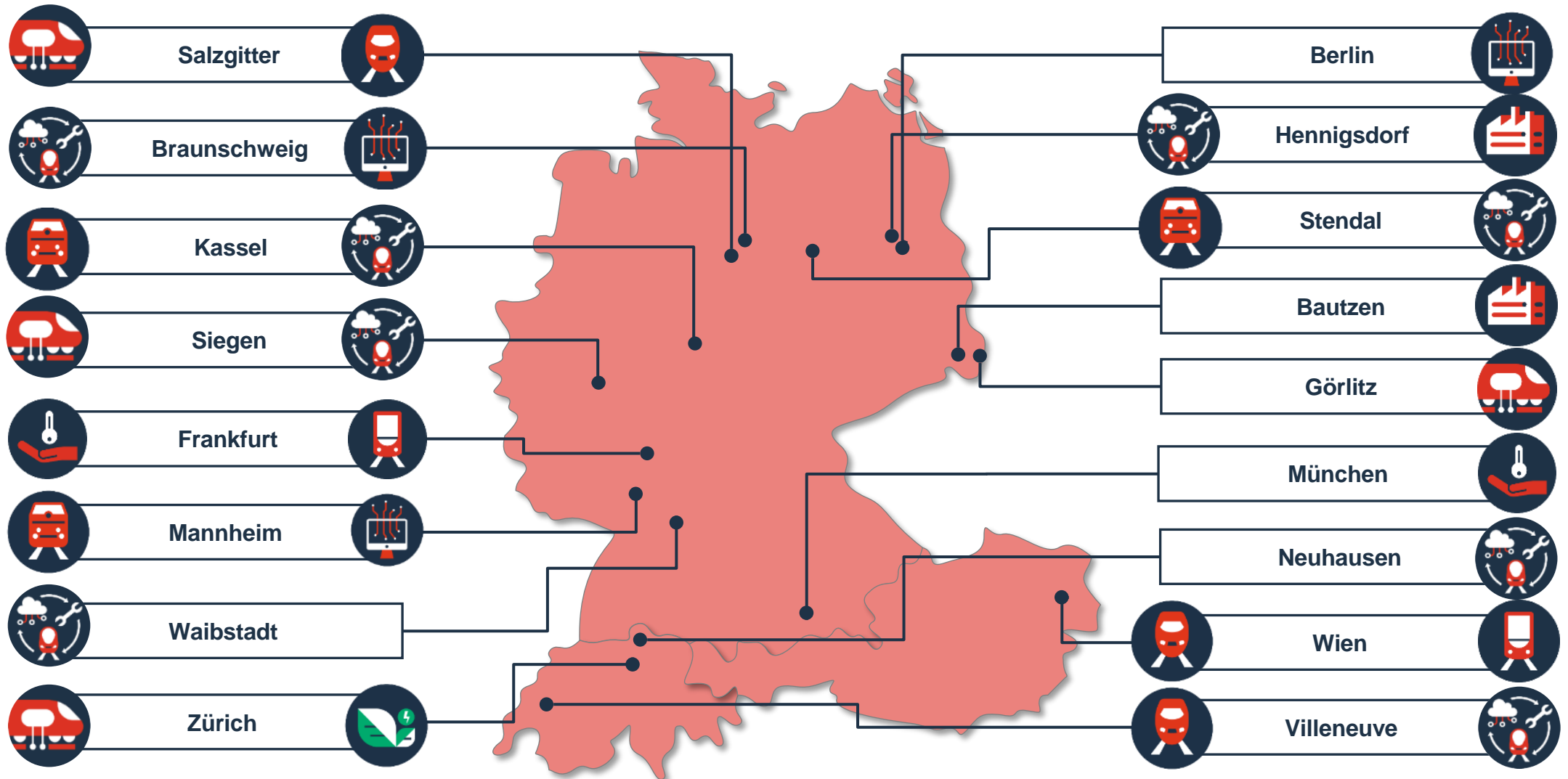
Asset Life Management

Ersatzteile & Komponentenüberholung

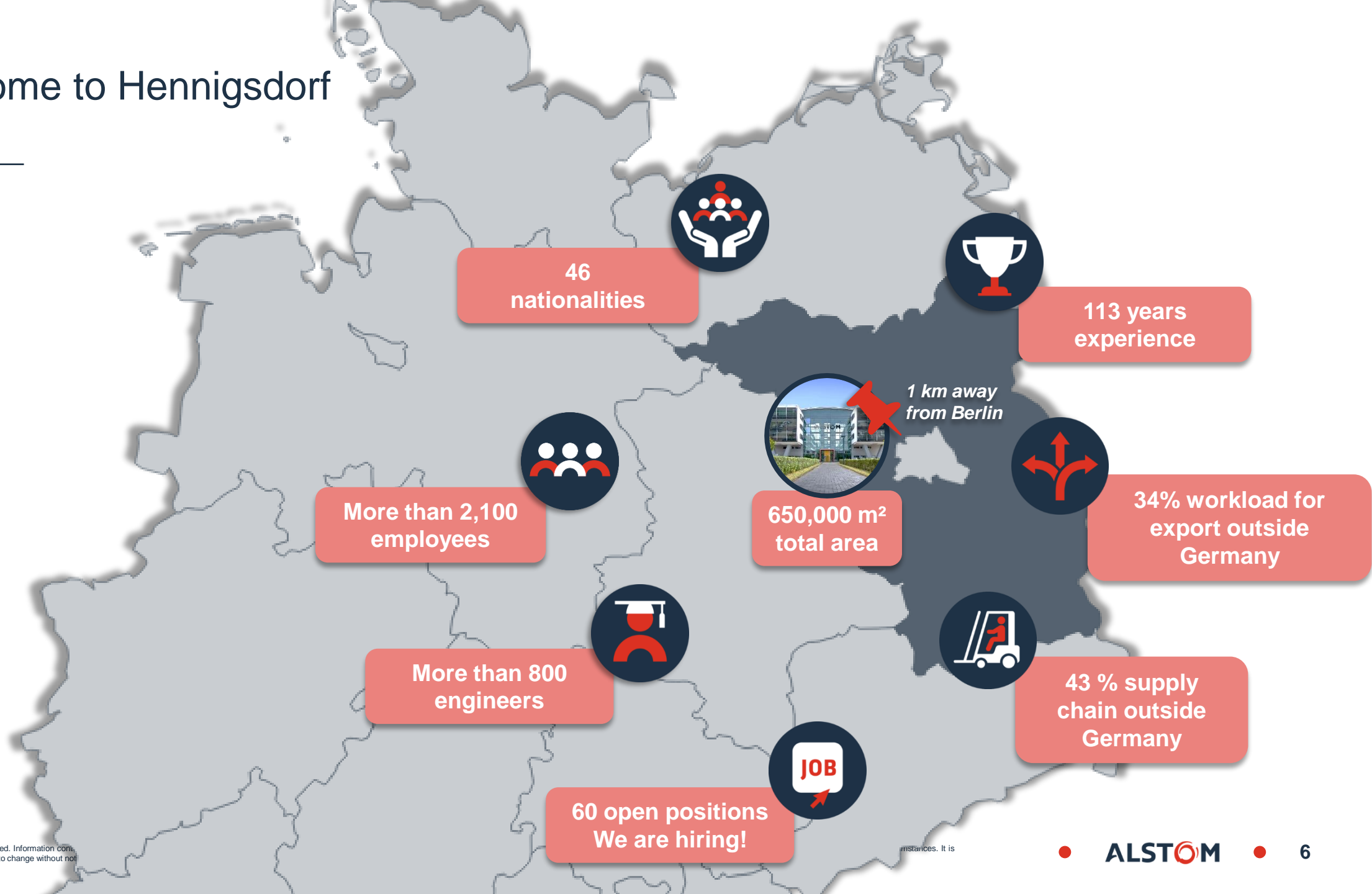
Digitale Lösungen

Betrieb und Systemwartung

# Standorte DACH-Region



# Welcome to Hennigsdorf



# Hennigsdorf: Entwicklung, Prototypen-Fertigung, Testing und Service



- Umfangreiche Kompetenzen in der Entwicklung von Vollbahnen
- Produktion von u.a. Prototypen, Vor- und Kleinserien
- Service Center

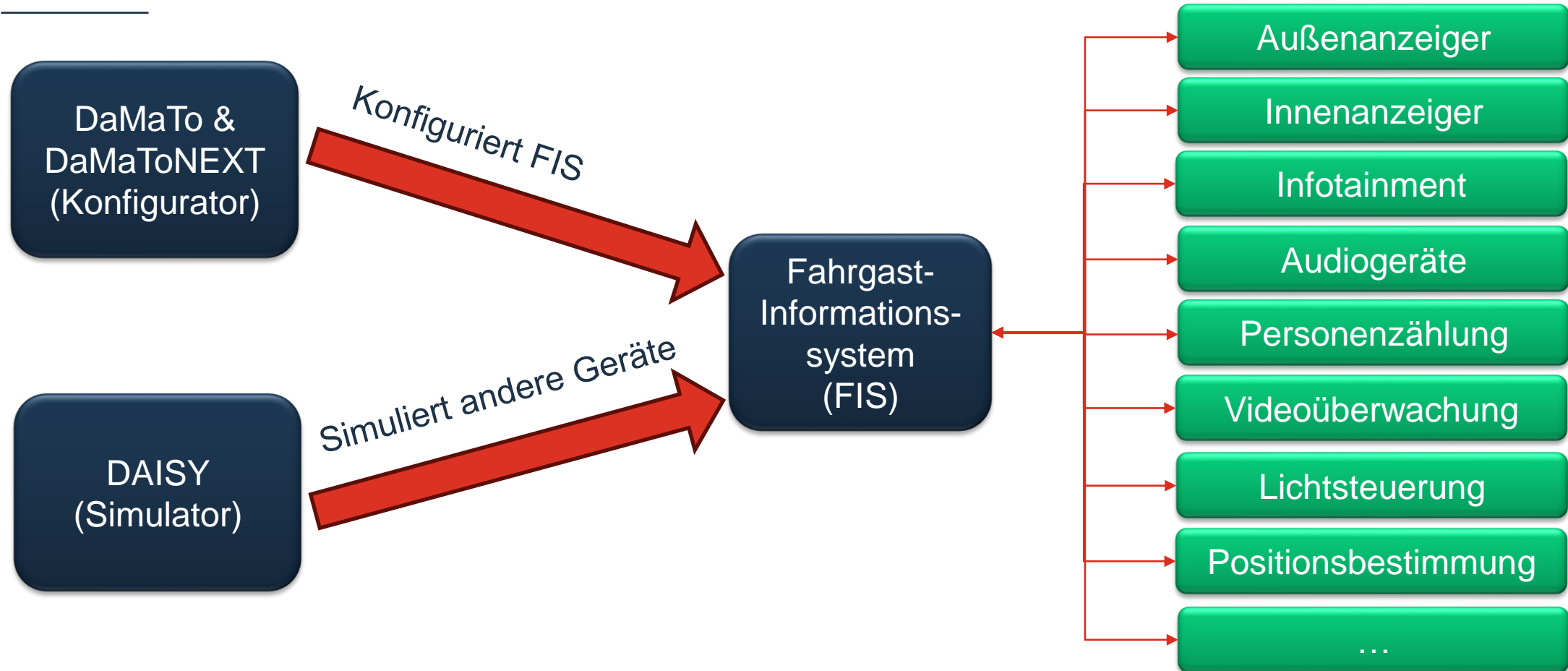


- Ca. 2.100 Mitarbeiter:innen
- 620.000 m<sup>2</sup> Gesamtfläche



- 3 interne Teststrecken (70 km/h)
- 1 externe Teststrecke (140 km/h)
- Typ-, System- und Zulassungstests

# Funktionen in einem Zug





# DaMaTo – Strecke zwischen zwei Bahnhöfen konfigurieren

The screenshot displays the ALSTOM DaMaTo software interface. The main window shows a table of segments with columns for 'Abfahrtsbahnhof', 'Ankunftsbahnhof', 'Ausgangsseite', 'Entfernung [m]', 'Entfernung Luft...', and 'Wenden mögl?'. The table lists various station-to-station connections, such as Verden(Aller) - HV (HV) to Nienburg(Weser) - HNBG (HNBG) with a distance of 3,115m. A warning icon is present in the 'Entfernung [m]' column for the segment Oldenburg(Oldb) - HOLD (HOLD) to Bad Zwischenahn - HZWI (HZWI).

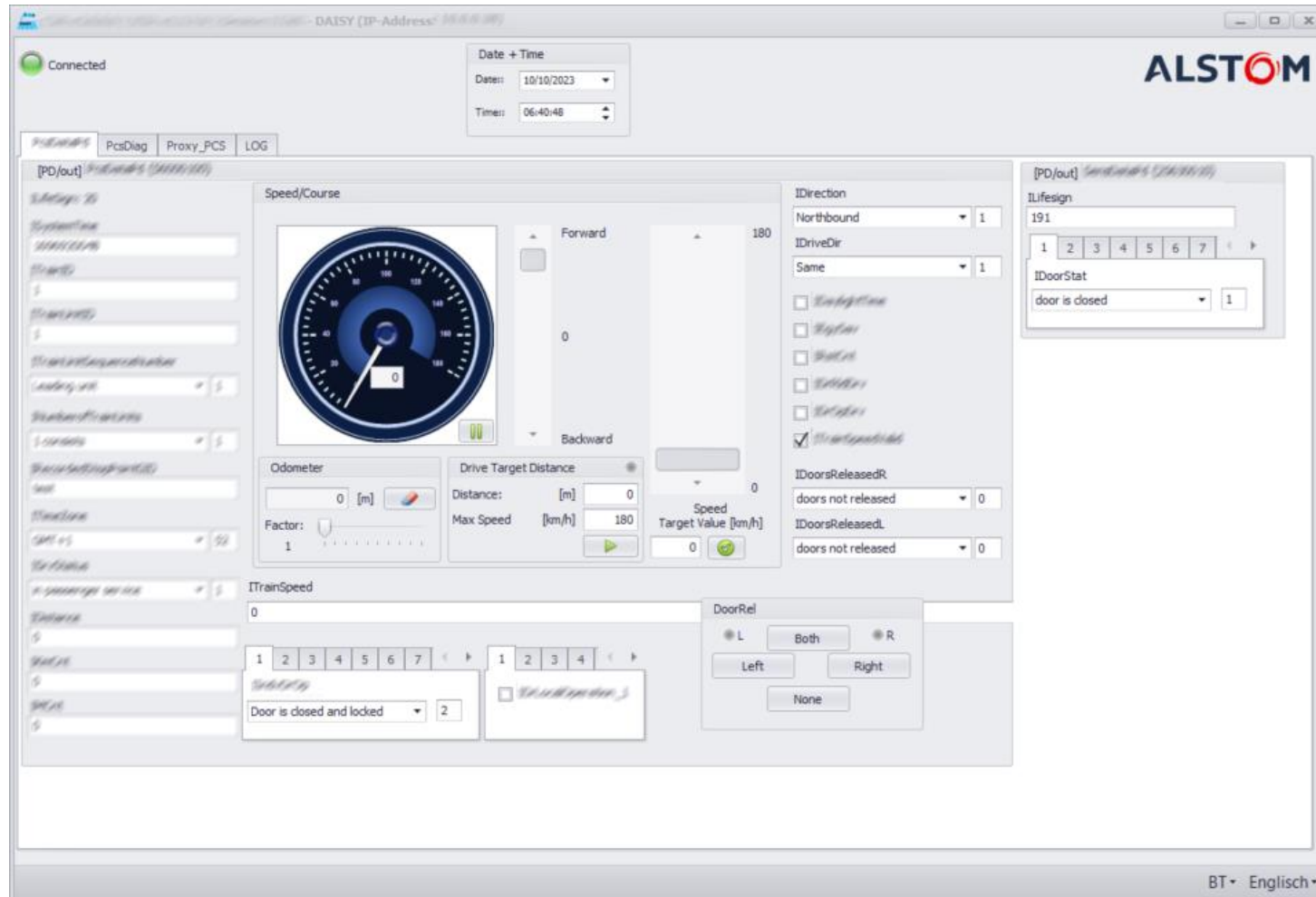
Abfahrtsbahnhof	Ankunftsbahnhof	Ausgangsseite	Entfernung [m]	Entfernung Luft...	Wenden mögl?
Verden(Aller) - HV (HV)	Nienburg(Weser) - HNBG (HNBG)		3,115	3,115	<input type="checkbox"/>
Dresden Hbf - DH (DH)	Dresden-Neustadt - DN (DN)		1,345	1,345	<input type="checkbox"/>
Hude - HHUD (HHUD)	Delmenhorst - HD (HD)		1,385	1,385	<input type="checkbox"/>
Hamm(Westf) - EHM (EHM)	Dortmund Hbf - EDO (EDO)		3,125	3,125	<input type="checkbox"/>
Emden Außenh. - HEA (HEA)	Emden Hbf - HE (HE)		1,815	1,815	<input checked="" type="checkbox"/>
Augustfehn - HAUG (HAUG)	Leer (Ostfriesl) - HLEE (HLEE)		2,145	2,145	<input type="checkbox"/>
Magdeburg Hbf - LM (LM)	Brandenburg Hbf - LB (LB)		8,015	8,015	<input type="checkbox"/>
Helmstedt - HHLM (HHLM)	Magdeburg Hbf - LM (LM)		4,755	4,755	<input checked="" type="checkbox"/>
Köln Hbf - KK (KK)	Düsseldorf Hbf - KD (KD)		4,015	4,015	<input type="checkbox"/>
Leipzig Hbf - LL (LL)	Bitzerfeld - LBT (LBT)		3,955	3,955	<input type="checkbox"/>
Dessau Hbf - LD (LD)	Bitzerfeld - LBT (LBT)		2,155	2,155	<input type="checkbox"/>
Nienburg(Weser) - HNBG (HNBG)	Verden(Aller) - HV (HV)		3,115	3,115	<input type="checkbox"/>
Magdeburg Hbf - LM (LM)	Braunschweig Hbf - HBS (HBS)		8,015	8,015	<input type="checkbox"/>
Königs Wusterh. - BKW (BKW)	Berlin-Lichtenberg - BLO (BLO)		3,015	3,015	<input type="checkbox"/>
Oldenburg(Oldb) - HOLD (HOLD)	Bad Zwischenahn - HZWI (HZWI)		1,455	1,455	<input type="checkbox"/>
Bad Oeynhausen - HOY (HOY)	Minden(Westf)Hbf - HM (HM)		1,145	1,145	<input checked="" type="checkbox"/>
Wuppertal Hbf - KW (KW)	Hagen Hbf - EHG (EHG)		2,615	2,615	<input type="checkbox"/>
Königs Wusterh. - BKW (BKW)	Berlin Ostbahn. - BHF (BHF)		2,655	2,655	<input checked="" type="checkbox"/>
Solingen Hbf - KSO (KSO)	Köln Hbf - KK (KK)		2,615	2,615	<input checked="" type="checkbox"/>
Berlin Wannsee - BWS (BWS)	Berlin Gesundbr. - BGS (BGS)		2,195	2,195	<input checked="" type="checkbox"/>
Emden Hbf - HE (HE)	Leer(Ostfriesl) - HLEE (HLEE)		2,145	2,145	<input type="checkbox"/>
Berlin Hbf - BLS (BLS)	Berlin Wannsee - BWS (BWS)		1,915	1,915	<input type="checkbox"/>
Helmstedt - HHLM (HHLM)	Braunschweig Hbf - HBS (HBS)		3,915	3,915	<input type="checkbox"/>
Königs Wusterh. - BKW (BKW)	Berlin Hbf - BLS (BL)		3,855	3,855	<input checked="" type="checkbox"/>

Below the main table, there is a 'Fehler' (Errors) section with a table of error messages:

Lfd. Nr.	Feld	Beschreibung
4	Fe... Phrase [Benutzer::Flügel] - Sprachvarianten [de] - Audio-Gruppe [Innen] - Phrasenbaus...	Sie müssen in genau einer der Spalten (PRM-Datei oder RtdElement PRM) ei...
5	Fe... Phrase [Benutzer::Flügel] - Sprachvarianten [de] - Audio-Gruppe [Innen] - Phrasenbaus...	Sie müssen in genau einer der Spalten (PRM-Datei oder RtdElement PRM) ei...
6	Fe... Phrase [Benutzer::Flügel] - Sprachvarianten [de] - Audio-Gruppe [Innen] - Phrasenbaus...	Sie müssen in genau einer der Spalten (PRM-Datei oder RtdElement PRM) ei...

The interface also includes a left sidebar with navigation options like 'Einstellungen', 'Versionierung', and 'Routen', and a bottom status bar showing 'Fehler (23) - Warnungen (72) - Informationen (53)'.

# DAISY - Beispielprojekt



# DaMaToNEXT - Datenbearbeitung

The screenshot shows the DaMaToNEXT DataTypes configuration interface. The sidebar on the left contains the following menu items:

- General
  - Settings
  - Sergeant
  - Foreach
  - Datum-Zeit
- Test
  - External Files
  - Audio files
  - Image files
  - Video files
  - DataTypes**
  - DataTypes Null
  - Calculations
- Fahrplan
  - Routen
  - Routen2
  - Fahrten

The main configuration area displays the following fields:

- Name: A
- Boolean:
- BooleanWithDefault:
- DecimalValue: 44.4
- Enum: A
- Integer: 37
- IntegerRequired: 33
- Text2to10: 33
- TextRequired: 666
- Datum Vergangenheit: 01.08.2022
- Zeit früh: 04:55:12
- Datum Zukunft: 01.08.2072
- Zeit spät: 23:59:59

A modal dialog is open over the form, displaying a validation error: "Boolean and BooleanDefault need to be equal." The dialog also shows a table with the following data:

BooleanWithDefault	DecimalValue
<input checked="" type="checkbox"/>	44.4
<input type="checkbox"/>	76
<input checked="" type="checkbox"/>	44.4
<input type="checkbox"/>	76

Buttons for "Update" and "Cancel" are visible at the bottom of the modal dialog.

# Problem

- Viele Kundenprojekte gleichzeitig.
- Alle Projekte irgendwie gleich, aber doch nicht.
- Projekte laufen sehr lang, sind komplex und Anforderungen ändern sich.
- Projekte haben lange Wartungszeit (spätere Erweiterungen, Änderungswünsche, ...).
- Zeit für Benutzerhandbücher und Dokumentation oft nicht mehr vorhanden.
- Anforderungen und Quellcode konsistent halten, erfordert manuelle Disziplin.



**Ein generisches Konzept muss her!**

# Ziele und Wünsche

- Geringer Entwicklungsaufwand!
- Geringe Fehlerquote!
- Anforderungen ohne manuellen Aufwand konsistent zu Programm halten.
- Benutzerhandbücher und Dokumentation entsteht von allein.
- Auch alte Projekte können jederzeit erweitert oder geändert werden.
- Probleme nur einmal lösen, nicht „das Rad jedes Mal neu erfinden“.



**Ein generisches Konzept muss unbedingt her!**

# Lösungsansätze

## Jedes Projekt „auf der grünen Wiese“ starten

- Mit jedem Projekt wird komplett vorne begonnen.
- Man hat keine Altlasten, muss aber jedes Problem erneut lösen.
- Keine Wiederverwendbarkeit, keine Synergien, ...
- Unter Umständen werden sogar unterschiedliche Teams für unterschiedliche Projekte beauftragt.



Keine Lösung

# Lösungsansätze

## #ifdef

- Eine Codebasis, aber je nach Compiler-einstellungen ergeben sich unterschiedliche Ergebnisse.
- Quellcode ist schwer wartbar.
- Änderungen an einem Projekt haben u. U. zur Folge, dass sich alte Projekte nicht einmal mehr übersetzen lassen.
- Jedes Projekt muss komplett von vorn bis hinten durchgetestet werden, da die Auswirkungen nicht abschätzbar sind.



**Diese Projekte gibt es wirklich!**

# Lösungsansätze

---

## Unterschiedliche Entwicklungszweige

- Die generelle Software wird auf einer Entwicklungslinie gestartet.
- Werden Änderungen vorgenommen, so wird der entsprechende Code Teil in einem separaten Branch weiterentwickelt.
- Problem: Branche laufen mit der Zeit auseinander.



**Irgendwann hat man etliche Projekte, die nichts mehr gemeinsam haben!**



# Lösungsansätze

## Objektorientierte Architektur

- Funktionen werden in Basisklassen abstrahiert und pro Projekt angepasst.
- Teil des Quellcodes ist in allen Projekten gleich und nur die Anpassungen werden pro Projekt implementiert.
- Prinzipiell ein guter Ansatz, ABER
  - Die Abstraktion muss ständig angepasst werden, was doch wieder Änderungen in anderen Projekten zur Folge hat.
  - Segen für die Entwickler und Fluch für die Tester.



**Ansatz ist schon besser, aber hat noch viele Nachteile und ist kompliziert!**

# Lösungsansätze

---

## Generische Software über Konfiguration

- Software bietet nur Funktionsblöcke an, die mit einer Konfigurationsdatei parametrierbar werden.
- Die Funktionsblöcke selbst können projektunabhängig getestet werden (bspw. mit (automatischen) Unit Tests).



**Dieser Ansatz wird genauer betrachtet!**

# Funktionsweise

---

- Parameter der Funktionsblöcke werden in einer Konfigurationsdatei abgelegt.
- Softwarearchitektur:
  - Aus der Konfigurationsdatei wird Quellcode generiert, der dann kompiliert wird.
  - Für jeden Funktionsblock gibt es Quellcode, der dann ausgeführt wird.
  - Jeder Funktionsblock enthält die Funktion zum Generieren der Textbauteile für die Dokumentation.
- Automatische Versionsupdates
  - Bspw. XML-Konfiguration mit XSLT auf neue Struktur konvertieren.
- Benutzerhandbuch setzt sich zusammen aus projektspezifischen Textbausteinen und den generierten Textbausteinen aus den Funktionsblöcken.

**XML als Beschreibungssprache für die Anwendung!**

# Bewertung

---

## Generische Software über Konfiguration - Vorteile

- Funktionsblöcke sind unabhängig zueinander.
- Neue Funktionsblöcke können entwickelt und benutzt werden, ohne Auswirkungen auf andere Projekte.
- Funktionsblöcke als kleine Einheiten können ausgiebig getestet werden. Die parametrisierte Anwendung ist dann sehr fehlerarm.



**Wiederverwendbare konfigurierbare Funktionsblöcke!**

# Bewertung

---

## Generische Software über Konfiguration - Nachteile

- Es wird sozusagen eine proprietäre Entwicklungssprache geschaffen.
- Keine Dokumentation und Hilfe im globalen Internet verfügbar.
- Interne Dokumentation oft nur spärlich vorhanden.
- Debug-Möglichkeiten müssen in der Plattform selber entwickelt werden (und können gar nicht so umfangreich sein, wie bei einer IDE "von der Stange")



**Dieser Ansatz hat aber auch seinen Preis, aber es lohnt sich!**

# Grenzen von generischer Software

---

- Projekte müssen ähnlich sein.
- Unter Umständen nicht performant genug (im embedded Umfeld).
- Unter Umständen zu hoher Speicherverbrauch (im embedded Umfeld).
- Hohes Abstraktionsvermögen nötig.
- Diszipliniertes Arbeiten (Kosten, Zeit und Qualität).
- Gerade am Anfang ist Investitionsaufwand nötig.



**Muss genau geprüft werden, ob dieser Ansatz für das System geeignet ist.**

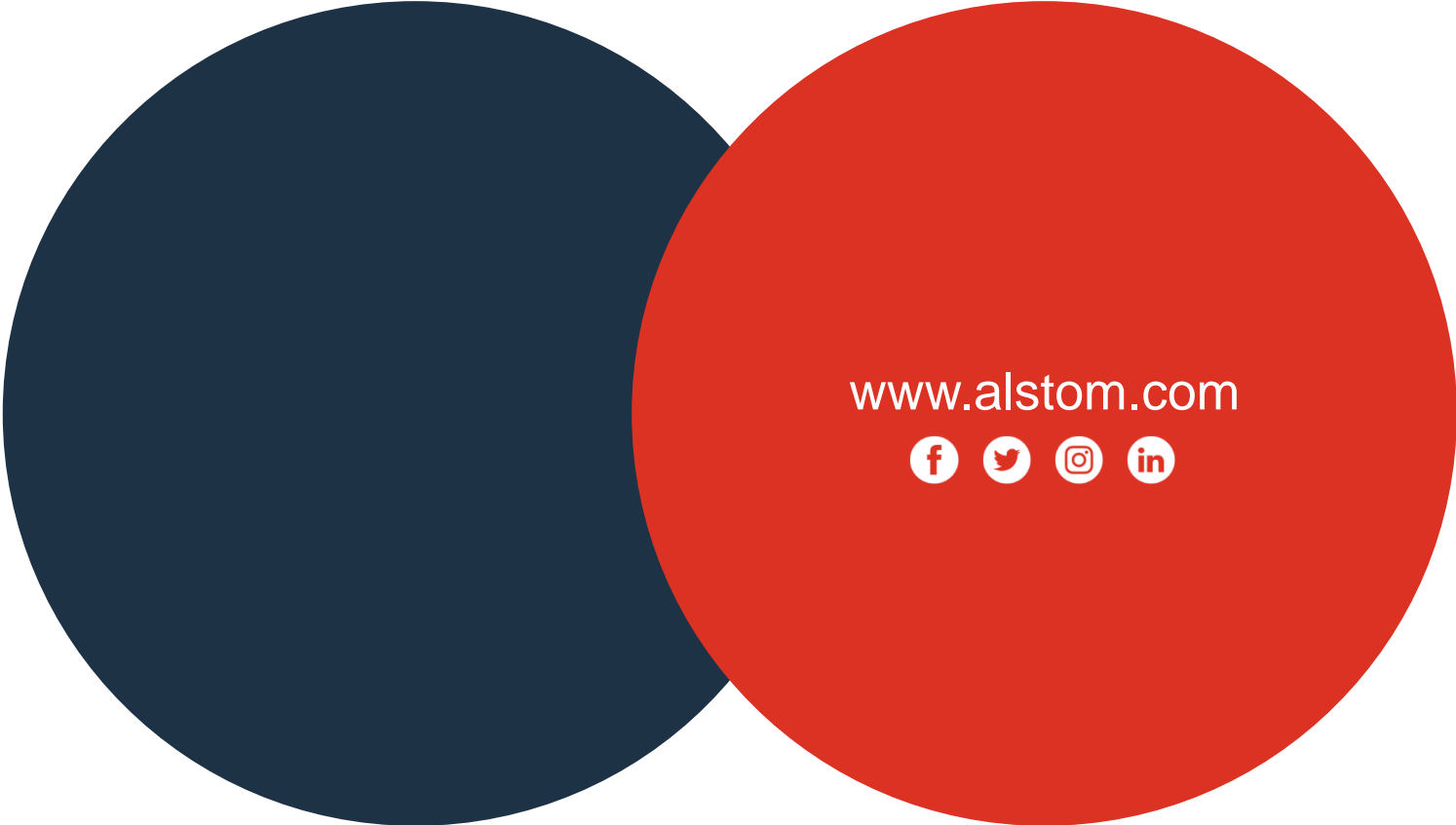
# Qualitätssicherungsmaßnahmen

---

## Unit Tests und Anwendungstest für den langfristigen Erfolg!

- Unit Tests der Funktionsblöcke (Test-driven development).
- 4 Augen-Prinzip!
- Auch die Testabdeckung ständig im Auge behalten (80% für nicht-sicherheitsrelevante Software)
- Manuelle „Chaos-Konfigurationen“ durch Test-Ingenieur.
- Bei der Entwicklung der Funktionsblöcke darf an Zeit nicht gespart werden.
- Stabiles Fundament = Stabile Software





[www.alstom.com](http://www.alstom.com)



**ALSTOM**  
• mobility by nature •



# Join #TeamAlstom

Get in  
Touch!

- Opportunities for Working Students
  - at our engineering site in Hennigsdorf
  - at our DACH headquarter at Ernst-Reuter-Platz (signalling & corporate functions)
- Internship
- Thesis
- VIE (French International Internship Program)
- Many exciting junior positions!

