INSTITUT FÜR INFORMATIK

Abschlussbericht KMU-innovativ – Verbundprojekt Titan Industrial DevOps Plattform für iterative Prozessintegration und Automatisierung

Wilhelm Hasselbring, Sören Henning, Björn Latte,
Irene Stemmler, Maik Wojcieszak, Uwe Glockmann

Bericht Nr. 2102
Oktober 2021
ISSN 2192-6247

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

ZU KIEL

Institut für Informatik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

Abschlussbericht KMU-innovativ – Verbundprojekt Titan Industrial DevOps Plattform für iterative Prozessintegration und Automatisierung

Wilhelm Hasselbring, Sören Henning, Björn Latte, Irene Stemmler, Maik Wojcieszak, Uwe Glockmann

Bericht Nr. 2102
Oktober 2021
ISSN 2192-6247

e-mail: wha@informatik.uni-kiel.de

Dies ist der Abschlussbericht eines KMU-innovativ Projektes, gefördert durch das BMBF

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

Abschlussbericht

KMU-innovativ - Verbundprojekt Titan Industrial DevOps Plattform für iterative Prozessintegration und Automatisierung

Wilhelm Hasselbring, Sören Henning, Björn Latte, Irene Stemmler, Maik Wojcieszak*, Uwe Glockmann

Verbundpartner
wobe-systems GmbH (WOBE)
Christian-Albrechts-Universität zu Kiel (CAU)
Institut für Informatik
Arbeitsgruppe Software Engineering
Kompetenzsverbund Software System Engineering

* Konsortialleiter

Laufzeit
01.02.2018 - 31.01.2021

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS17084A-B. gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurzfassung

Unternehmensprozesse zu digitalisieren und dabei eine IT-Infrastruktur aufzubauen, ist komplex. Neue, zum Teil teure Technologien werden eingesetzt, jedoch fehlen erprobte Praktiken. Die daraus entstehende Komplexität lässt sich mit dem klassischen Projektmodell nur ungenügend adressieren. Klassische Planungen basieren auf Annahmen, die sich oft zu spät und als falsch erweisen. Mechanismen, den einmal geplanten Weg zum gesetzten Ziel zu korrigieren, bietet das traditionelle Projektmodell nur eingeschränkt.

Ziel des titan-Projekts ist die Integration von Entwicklungswerkzeugen und Betriebs-Technologie in eine Software-Plattform. Kombiniert mit innovativen „Industrial DevOps"-Methoden soll die komplexe Aufgabe einer iterativen Systemintegration im industriellen Umfeld erheblich vereinfacht werden.

Im titan-Projekt ist der Prototyp einer Software-Plattform entstanden, die es industriellen Anwendern erlaubt, diese Praktiken auf Problemstellungen der Digitalisierung anzuwenden. Neben Zielen wie Sicherstellung und Überprüfbarkeit von Qualität, Widerstandsfähigkeit und Skalierbarkeit ist die Eliminierung des Vendor-Lock-In ein zentraler Aspekt des Projekts. Insbesondere werden Prozessanpassungen durch die Anwender mittels Flow Based Automation ermöglicht, neue Softwareversionen und Veränderungen können am System routinemäßig in Betrieb genommen werden und domänenspezifische Komponenten können für komplexe Aufgaben genutzt und verwaltet werden.

Im Rahmen einer Community wird die titan-Open-Source-Plattform weiterentwickelt. Die während des Projekts entstandenen Innovationen werden so verfeinert und in verschiedenen Bereichen angewendet. Die Erfahrungen aus Projekten fließen in die Software ein und werden innerhalb der Community verbreitet.

Inhalt

Abbildungsverzeichnis

1 Kurze Darstellung

„Die digitale Transformation zwingt Unternehmen ins Risiko“ - heißt es in dem Artikel „Was Industrie 4.0 für die Unternehmen bedeutet“ von Bastian Henrichs in „Die Welt“.

Unternehmensprozesse zu digitalisieren und dabei eine IT-Infrastruktur aufzubauen, ist komplex. Neue, zum Teil teure Technologien werden eingesetzt, jedoch fehlen erprobte Praktiken. Die daraus entstehende Komplexität lässt sich mit dem klassischen Projektmodell nur ungenügend adressieren. Klassische Planungen basieren auf Annahmen, die sich oft zu spät und als falsch erweisen. Mechanismen, den einmal geplanten Weg zum gesetzten Ziel zu korrigieren, bietet das traditionelle Projektmodell nur eingeschränkt.

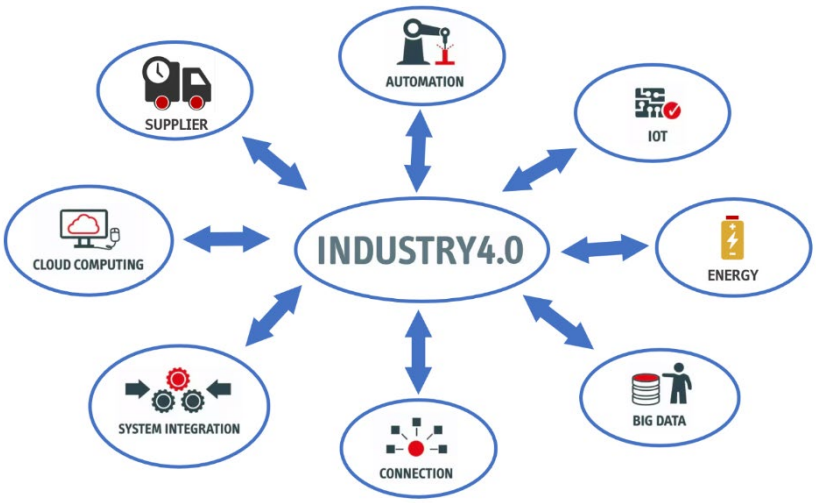

Abbildung 1: Industrie 4.0

Der IT sind komplexe Problemstellungen bekannt und Praktiken wie DevOps, Lean und Agile bieten eine Reihe von Methoden ein gesetztes Ziel in kontrollierten und überschaubaren Schritten zu erreichen und bedarfsorientiert jederzeit die Richtung korrigieren zu können.

Im titan-Projekt ist der Prototyp einer Software-Plattform entstanden, die es industriellen Anwendern erlaubt, diese Praktiken auf Problemstellungen der Digitalisierung anzuwenden. Neben Zielen wie Sicherstellung und Überprüfbarkeit von Qualität, Widerstandsfähigkeit und Skalierbarkeit ist die Eliminierung des Vendor-Lock-In ein zentraler Aspekt des Projekts.

Gelingt es mit titan, neben den Risiken auch gleichzeitig die Kosten der Digitalisierung zu reduzieren, ist das Projekt besonders für mittelständische Unternehmen interessant. Ziele, die bisher zu kostspielig erschienen, werden erreichbar und die Wettbewerbsfähigkeit dieser Unternehmen im Markt dadurch gestärkt.

An Unternehmen angepasste Prozesse stellen ein wichtiges Potenzial für die eigene Wettbewerbsfähigkeit dar. Diese Prozesse zugunsten von Standardsoftware aufzugeben, birgt ein weiteres Risiko. Es wurde untersucht, wie titan Anwendern die Möglichkeit gibt, ihre Prozesse digital zu modellieren, zu prüfen und zu aktualisieren. Eine aufwendige Anforderungsanalyse entfällt, da vorhandenes Wissen im Unternehmen direkt genutzt wird. Ob titan einen Einfluss auf die Digitalisierung haben wird, hängt nur zum Teil von technischen Gesichtspunkten ab. Ein anderer wichtiger Aspekt ist, eine solide Basis von Anwendern und Dienstleistern zu finden, die die Ergebnisse des Projekts für eigene Lösungen nutzen.

1.1 Aufgabenstellung

Ziel des titan-Projekts ist die Integration von Entwicklungswerkzeugen und Betriebs-Technologie in eine Software-Plattform. Kombiniert mit innovativen „Industrial DevOps“-Methoden soll die komplexe Aufgabe einer iterativen Systemintegration im industriellen Umfeld erheblich vereinfacht werden.
Für den Test der organisatorischen und technischen Komponenten wird eine Beispielanwendung zur Evaluation entwickelt und in einer realen Produktionsumgebung eingesetzt.

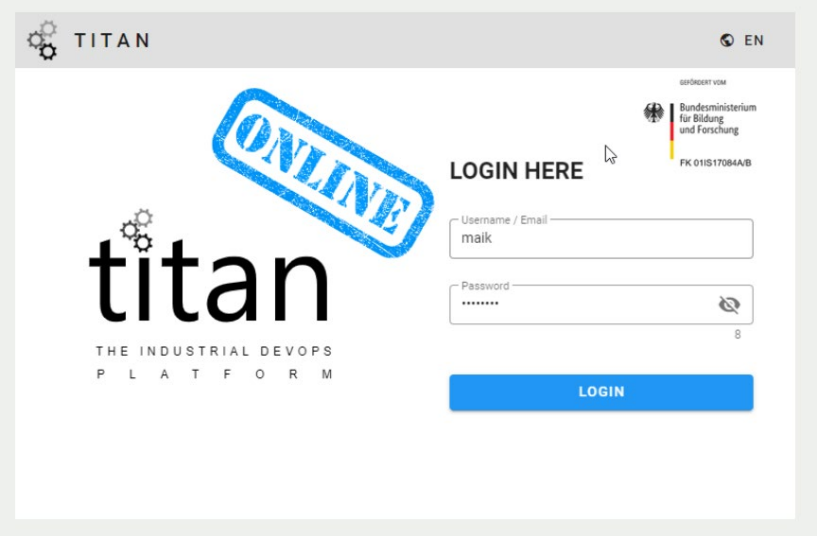

Abbildung 2: titan No-Code Login

Weitere Ziele des Projekts sind:

- Prozessanpassungen durch die Anwender mittels Flow Based Automation (FBA) ermöglichen.

- Neue Softwareversionen und Veränderungen am System routinemäßig in Betrieb nehmen.
- Domänenspezifische Komponenten für komplexe Aufgaben nutzen und verwalten.

Für Digitalisierungsprojekte fehlen Erfahrungswerte. Es ist daher schwer vorherzusagen, welche Investitionen den gewünschten Erfolg bringen. Aufwendige Experimente können sich nicht nur als teuer, sondern auch als existenzbedrohend für kleine und mittelständische Unternehmen (KMU) erweisen. Dazu kommen organisatorische Umstellungen im Unternehmen, Anpassungen der eigenen Arbeitsweise und die Anschaffung einer IT-Infrastruktur, die bisher nicht notwendig war und für deren Betrieb in kleinen und mittleren Unternehmen das Know-how fehlt. Wird am Ende deutlich, dass der eingeschlagene Weg der falsche war, ist eine Umkehr oft nicht mehr möglich.

Hat es ein Unternehmen geschafft, eine hohe Integration und Automatisierung der Systeme zu erreichen, folgt die nächste Herausforderung. Schnelle und sichere Reaktionen auf Veränderungen des Marktes sind notwendig, um mit dem Wettbewerb standzuhalten. Ein einmal eingeführtes System muss flexibel und leicht anpassbar sein. Je größer ein System jedoch ist und je mehr Hersteller Komponenten dazu beitragen und je stärker diese Komponenten miteinander interagieren, desto höher sind die Kosten für Modifizierungen und desto größer ist das Risiko. Die Folge können z.B. Produktionsausfälle sein.

Neben der Komplexität der Systeme stellt auch die mit der Zeit beim Betrieb entstehende „technische Schuld“ ein Risiko dar. Die Ursache dafür sind beispielsweise fehlende Updates von Softwarekomponenten, Betriebssystemen, Firmware oder schlechte Qualität von Integrationscode, wie Skripten. Je öfter ein System verändert wird, desto größer ist die Wahrscheinlichkeit, dass sich „technische Schuld“ aufbaut.

Änderungen am bestehenden System sind jedoch unumgänglich. Die Umstellung einer bestehenden Produktion von Inselsystemen auf ein integriertes System mit einem hohen Maß an Automatisierung kann bestenfalls in wenigen makroskopischen Schritten erfolgen. Eine kontinuierliche Entwicklung in kleinen Schritten, die regelmäßig überprüft und gegebenenfalls rückgängig gemacht werden kann, wäre eine Möglichkeit für Unternehmen, sowohl flexibel zu bleiben wie auch kostengünstig effektives Risikomanagement zu betreiben.
Die Überprüfbarkeit von Veränderungen erfordert belastbare Daten über den Betrieb des Gesamtsystems. Erst wenn alle (oder zumindest alle verfügbaren) Daten in einem System zusammengeführt und kombiniert werden, kann eine schnelle und verlässliche Überprüfung der Effektivität von Veränderungen erfolgen. Natürlich bieten die Daten weitere Potentiale.

Transparenz für Kunden, frühzeitige Erkennung von Problemen im Betrieb, Flexibilisierung von Serviceintervallen und automatische Benachrichtigung von Bereitschaftspersonal sind nur einige Beispiele.

Mit der Einführung einer Software zur Integration unterschiedlicher, vorhandener und neuer Systeme entsteht ein hohes Risiko, das schwer einzuschätzen ist. Anders als Komponenten in der Produktion, kann ein solches System nicht ohne erheblichen Aufwand ausgetauscht werden. Verliert ein Softwarehersteller das Interesse an seinem Produkt, gibt es möglicherweise keine Aktualisierungen mehr und das System veraltet innerhalb kurzer Zeit. Ein Softwareanbieter erlangt mit einem stark integrierten System eine Monopolstellung für potenzielle Änderungen. Das ist zum einen aus Kostengründen ggf. problematisch, zum anderen, weil das Unternehmen (KMU) nicht den Dienstleister beauftragen kann, der die Domäne am besten beherrscht, für die eine Anpassung nötig ist. Beispielsweise ist für die Verknüpfung von kaufmännischen Systemen ein anderes Know-how erforderlich als für die Verknüpfung von eingebetteten Produktionssystemen.

Es sollte ein Prototyp einer Software-Plattform entwickelt und gemeinsam mit den assoziierten Partnern hinsichtlich seiner Anwendbarkeit für iterative Systemintegration getestet und evaluiert werden. Als Vorbild für die technischen und organisatorischen Herausforderungen beim Integrations- und Verbesserungs-Prozess sollten DevOps-Methoden dienen. Es sollte untersucht werden, wie sich DevOps-Methoden auf die Systemintegration in der Industrie anwenden lassen und welche Vorteile sie bringen.

Für die Überprüfung und Optimierung von Iterationsschritten sollten Monitoring- und Logging-Funktionen implementiert werden (Operational Technologie). Technische (CPU, Speicher, Festplatten, Netzwerk, usw.), anwendungsspezifische (Funktion, Ausführungs-geschwindigkeit) und geschäftsrelevante (Stückzahlen, Ausfälle, Qualitätsabweichungen) Telemetrie sollte von dem Prototyp bereitgestellt werden. Damit Anwender aus Fachabteilungen ohne IT-Hintergrund modellierte Prozesse verstehen und selbst Prozesse modellieren können, sollte eine grafische Beschreibungssprache für Flow-basierte Automatisierung geschaffen werden, die intuitiv verständlich ist.

Domänenspezifische Komponenten sollten im Prototyp installiert, verwaltet und verwendet werden können. Es sollte untersucht werden, wie die Kapselung von technischen Details die Anwendung von Personal ohne IT-Hintergrund unterstützt. Es sollte untersucht werden, wie sich eine Grid-Architektur auf die Resilienz und Skalierbarkeit des Systems auswirkt. Dafür wurden Fehlerszenarien provoziert und untersucht. Für die Benutzeroberfläche (UI) sollte ein

Web-Client, der in Internetbrowsern ablaufen kann, entwickelt werden. Dabei spielte die Ergonomie eine wichtige Rolle. Ziel war es, ein leicht verständliches UI zu schaffen, das Fehler bei der Bedienung durch sein Design weitgehend vermeidet. Für den Test der organisatorischen und technischen Komponenten wurde eine Beispielanwendung zur Evaluation entwickelt und in realen Produktionsumgebungen bei den assoziierten Anwendungspartnern (Druckzentrum der Kieler Nachrichten und IBAK) eingesetzt.

1.2 Voraussetzungen, unter denen das Vorhaben durchgeführt wurde

Das Projekt wurde im Rahmen des Förderprogramms „KMU-innovativ“ des BMBF und die darauf erfolgte Bekanntmachung „Richtlinie zur Fördermaßnahme ("KMU-innovativ: Informations- und Kommunikationstechnologie (IKT)".), Bundesanzeiger vom 22.05.2017“ gefördert.
Die Projektpartner Christian-Albrechts-Universität zu Kiel (CAU) und wobe-systems GmbH bilden die in den förderpolitischen Zielen aus der Bekanntmachung (BKM) zum „KMU-innovativ: IKT“ angestrebte Kombination aus Spitzenforschung und erstantragstellendes KMU. Wissenschaft und Wirtschaft arbeiten eng zusammen. 14 assoziierte Partner aus der Industrie liefern Ideen und Anforderungen.

Abbildung 3: Assoziierte Partner im titan Projekt

Die wobe-systems GmbH [WSHP] ist Bestandteil der "wobe-Gruppe". Sie wurde 2003 als GmbH eingetragen. Hervorgegangen ist sie aus der 1999 gegründeten wobe-team GbR.
Die wobe-systems GmbH entwickelt Softwarelösungen für die Automatisierung von Großdruckereien. Basierend auf eigenen Produkten ist das Kerngeschäft die individuelle Integration heterogener Systeme, bestehend aus alten und neuen Maschinen, Planungs- und Verwaltungs-Systemen, sowie deren Automatisierung.
Seit 2016 bietet die wobe-systems GmbH basierend auf über 15 Jahren Erfahrung ihre Dienstleistungen im Bereich der industriellen Automatisierung kleiner und mittelständischer Unternehmen an und entwickelt individuelle Softwarelösungen. Es wurden auch bereits Erfahrungen mit eigenen Open-Source Projekten gesammelt:

- libTML (libtml.org) ist eine Open-Source Bibliothek für skalierbare nachrichtenbasierte Interprozesskommunikation im Netzwerk
- libUJO (libujo.org) ist eine Open-Source Bibliothek für binäre Datenserialisierung für Internet-of-Things Anwendungen

Die Arbeitsgruppe Software Engineering am Institut für Informatik an der Christian-Albrechts-Universität zu Kiel wurde im Oktober 2008 von Prof. Dr. Wilhelm Hasselbring gegründet, der zuvor acht Jahre die Abteilung Software Engineering an der Carl-von-Ossietzky-Universität Oldenburg leitete und Bereichsvorstand am OFFIS-Institut war. An den Universitäten Kiel und Lübeck wurde 2009 der Kompetenzverbund Software System Engineering „KoSSE“ [KOSSE] zum Technologietransfer eingerichtet, in dem Prof. Hasselbring u.a. Sprecher der Kieler Projekte ist. Im Exzellenzcluster „Future Ocean” [FUOC] ist Prof. Hasselbring Principal Investigator.
Die Forschungsschwerpunkte liegen überwiegend im Bereich der Softwarearchitekturen, insbesondere zur Integration und Modernisierung von Informationssystemen. Cloud-basierte Architekturen und skalierbare Microservice-Architekturen stellen aktuelle Forschungsthemen dar. Im Betrieb von Softwaresystemen kommt dem Monitoring (z.B. durch die Software Kieker [KIKR]) und der Visualisierung (z.B. durch die Software ExplorViz [FKH2017]) eine wichtige Rolle zu. DevOps als innovative, agile Softwareentwicklungsmethode zur Integration von Entwicklung und Betrieb komplexer Softwaredienste wird erforscht und eingesetzt.

Die Bereitstellung der während des Projekts entwickelten Software als Open Source und die Veröffentlichung der Ergebnisse gibt KMUs aus unterschiedlichen Industriebereichen die Möglichkeit im Rahmen einer Community eigene Lösungen anzubieten. Das Ziel dabei ist den Austausch von Ideen zu fördern. Nicht jedes KMU muss Digitalisierung neu erfinden. Innovationen können so besser übergreifend genutzt werden. Die Herausforderungen der Digitalisierung, die das titan-Projekt adressiert, werden von Großunternehmen üblicherweise durch die Einführung einer eigenen Softwareentwicklung gelöst. titan zielt besonders auf

KMUs ab, die ihre ersten Schritte in die Digitalisierung machen möchten. Das kontrollierbare Risiko und die geringeren Kosten könnten das Kräfteverhältnis zwischen KMU und Großunternehmen (GU) zugunsten der KMUs verändern.

Der gewählte technologische Ansatz des titan-Projekts erfordert innovative Lösungen in den Bereichen Datenwissenschaft und Informationstechnologie, was ebenfalls den förderpolitischen Zielen entspricht. Für die CAU kommt dem Transfer von Forschungsergebnissen aus der Wissenschaft in die Wirtschaft eine wichtige Rolle zu, um neue Ideen in Innovationen umzusetzen und die digitale Transformation der Industrie aktiv mitzugestalten. Digitalisierung, Prozessautomatisierung und unternehmensweite Systemintegration sind eine Voraussetzung für die unternehmensübergreifende Daten- und Prozess-Integration der Industrie 4.0. titan passt damit sehr gut zu den förderpolitischen Zielen von „KMU-innovativ: IKT“.

Damit die Zusammenarbeit von Anwender und wechselnden Softwareherstellern an einem System, über einen langen Zeitraum gelingt, bedarf es einer Architektur, die komponentenbasiert, modular und so verständlich ist, dass eine effiziente Einarbeitung in kurzer Zeit erfolgen kann. Hilfssysteme wie Datenbanken, Dateiverwaltung oder Managementinformationssysteme, die es in unterschiedlichen Ausprägungen gibt, werden vom titan-Kern durch eine Abstraktionsschicht (Interface) isoliert und so austauschbar gemacht.

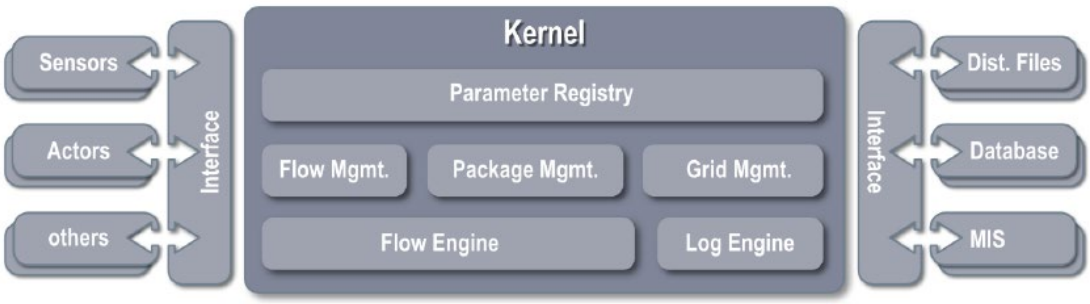

Abbildung 4: titanRS-Kern und angeschlossene Komponenten

Der titan-Kern ist ebenfalls modular aufgebaut. Domänenspezifische Komponenten werden in Paketen zusammengefasst, installiert und verwaltet (Package Mgmt.). Komponentenpakete und Treiber können von unterschiedlichen Herstellern geliefert und gepflegt werden. Klar definierte Schnittstellen verhindern, dass es zu Insellösungen kommt. Eingebaute Regressionstest ermöglichen die Qualitätssicherung, bevor neue Versionen produktiv genutzt werden. Durch den Wegfall des Vendor-Lock-In sind Softwarehersteller

austauschbar. Firmeneigene IT, verschiedene Hersteller und die Produktion können teamübergreifend zusammenarbeiten.

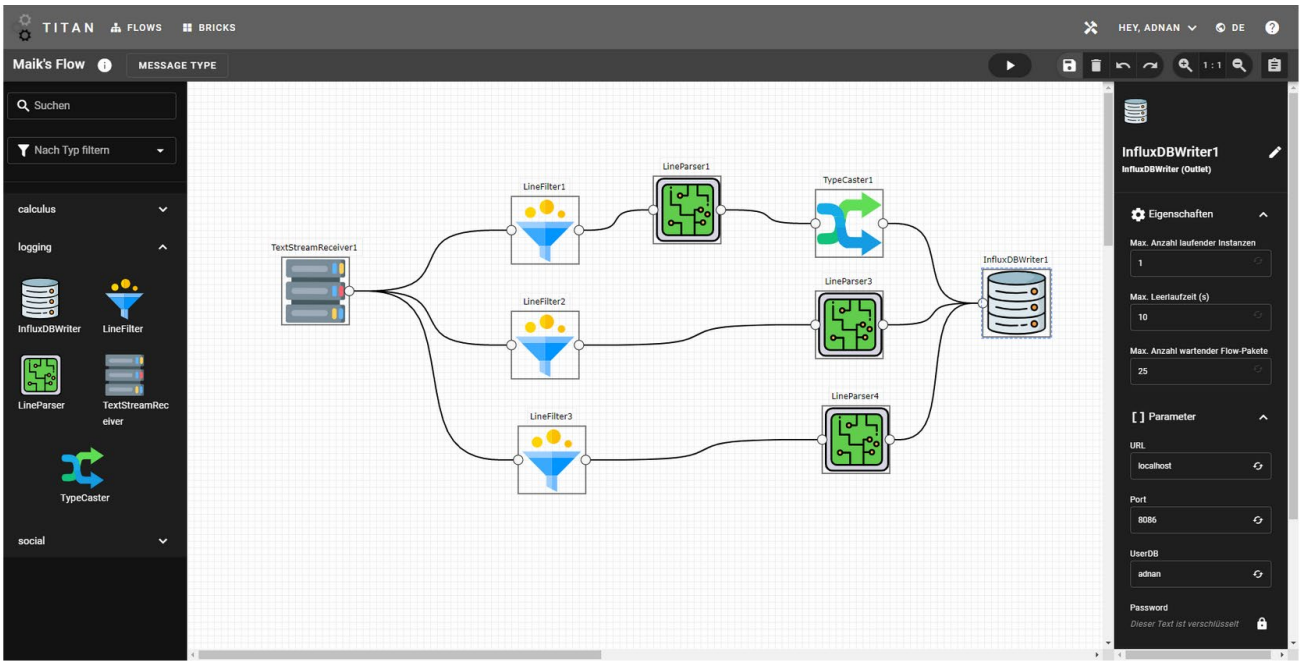

Abbildung 5: Flow Based Automation in titan No-Code

Komponenten werden mittels Flow Based Automation (FBA) für die Modellierung von Geschäfts- und System-Prozessen in Flows miteinander verbunden. Die Flow-Architektur ermöglicht es, Systeme elastisch skalierbar zu gestalten. Einzelne Funktionen können auf beliebig viele Rechner in einem Netzwerk verteilt werden. Parallelität erfordert dabei keine spezielle Kenntnis des Entwicklers oder Prozessdesigners, sondern ist Systembestandteil des Kerns. Die Komponenten werden als skalierbare Microservices realisiert, die unabhängig installiert werden können.

Die klare Abgrenzung und Struktur von Funktionskomponenten, Interfaces und Flows, ermöglicht eine standardisierte Protokollierung und ein effektives Monitoring von Systemfunktionen. Neben potenziellen Systemproblemen können Informationen über Häufigkeit der Verwendung, Veränderungen und Upgrades von Flows dargestellt werden. Geeignete Metriken und KPIs zeigen zu jeder Zeit den Zustand des Systems und ermöglichen, neben der für industrielle Anwendungen dringend erforderlichen Nachvollziehbarkeit (ISO900x), auch schnelle und gezielte Reaktionen auf potenzielle Probleme.

Das Pilotprojekt Control Center dient der laufenden Überprüfung neuer Softwareversionen in einem realen Produktionsumfeld. Es werden Problemfelder adressiert, die naturgemäß

Individualentwicklung erfordern und ständige Veränderung verlangen. Für die spätere Anwendbarkeit von titan ist dieses Feedback von hoher Bedeutung. Zusammen mit der Kieler Zeitungsdruckerei werden Produktionssysteme mit titan integriert. Das Ziel ist die Speicherung und Auswertung von Daten über den Ressourcenverbrauch und die Nutzung der Systeme. Es soll untersucht werden, wie diese Daten für die Flexibilisierung von Service Intervallen durch Predictive Maintenance genutzt werden können.

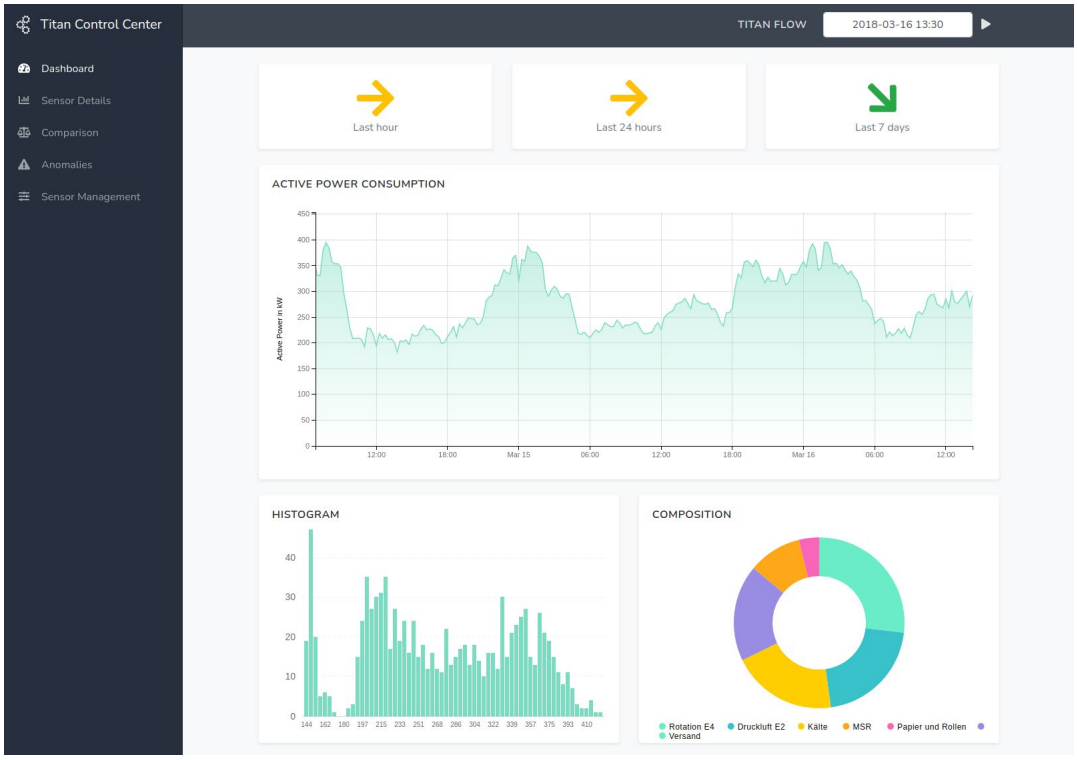

Abbildung 6: titan Control Center

Ein weiterer Aspekt ist die Optimierung des Ressourcenverbrauchs in der Produktion, sowohl in der Kieler Zeitungsdruckerei als auch in der Produktion von IBAK. Im Rahmen eines Energiemanagements nach ISO 50001 soll der Stromverbrauch optimiert und Einsparpotentiale erkundet werden. Um dieses Ziel zu erreichen, ist es wichtig, mittels empirischer Daten laufend zu überprüfen, ob bereits getroffene Maßnahmen effektiv sind. Darüber hinaus sollen Prognosen anhand gesammelter Daten neue Einsparpotentiale aufzeigen. Das titan Control Center soll im Rahmen des Projekts entsprechende Daten liefern und so die Zeit zwischen den Iterationen im Deming-Kreis (Plan-Do-Check-Act-Zyklus) verkürzen.

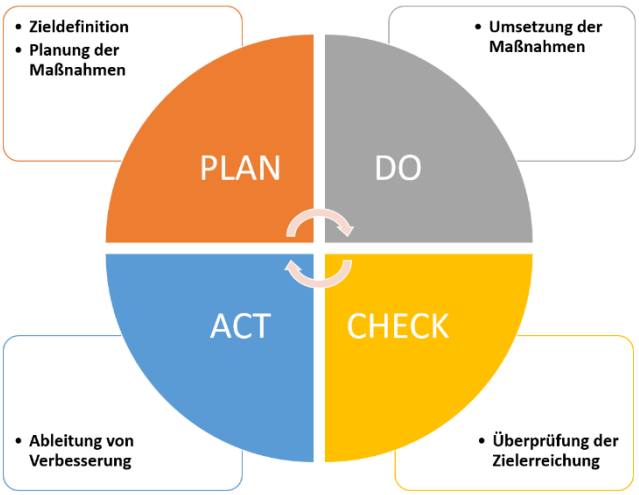

Abbildung 7: PDCA-Zyklus

Der vorgeschlagene Lösungsansatz, Industrial DevOps und die titan-Plattform, geben den Anwendern Werkzeuge an die Hand, die es ihnen ermöglichen die Qualität von Softwarekomponenten kontinuierlich sicherzustellen und effizient Automatisierungsprozesse umzusetzen. So werden die Anwender eng als aktive und kreative Know-how-Lieferanten in den Entwicklungsprozess eingebunden. Ermöglicht wird das durch eine Flow-Architektur, die Prozesslogik als Pipelines mit Funktionsblöcken darstellt. Solche Ablaufpläne sind intuitiv und leicht verständlich, vergleichbar etwa mit dem Kopieren von Dateien über „Hotfolder“ von einem Werkzeug zum anderen.

Die Flow-Architektur ermöglicht darüber hinaus eine hohe Skalierbarkeit, Parallelität und Elastizität, sowie Redundanz für Ausfallsicherheit. Der Test auf dem Produktivsystem wird zur Regel und Bestandteil der täglichen Arbeit. Änderungen am System werden automatisch protokolliert und Testergebnisse revisionssicher gespeichert. Die Überprüfung der internen Qualität der titan-Plattform selbst wird durch die Bereitstellung des Kerns als Open-Source-Projekt ermöglicht.
Automatisierung von Tests und die Übernahme von neuen Softwareversionen in den produktiven Einsatz sowie Monitoring und Protokollierung des Systems zu jedem Zeitpunkt sind ebenfalls wichtige Funktionen der titan-Plattform.

Basierend auf DevOps-Methoden werden diese Funktionen genutzt, die Qualität und Funktionsfähigkeit des Systems zu visualisieren und Optimierungspotenziale aufzuzeigen. Veränderungen am System werden zur täglichen Routine, ohne Störungen des Ablaufs im Betrieb zu verursachen. Die Trennung von Flow und Funktionskomponenten ermöglicht eine schnelle Anpassung an organisatorische Gegebenheiten in Unternehmen. Egal, ob mit

internen oder externen Dienstleistern am System gearbeitet wird oder wie oft neue Dienstleister mit Änderungen am System beauftragt werden, der Anwender bleibt autonom und behält die Kontrolle über die Qualität seines Systems.

Zusammengefasst besteht die Neuheit des Lösungsansatzes in der konsequenten Anwendung moderner Architektur, den Möglichkeiten moderner IT-Systeme und organisatorischer Methoden zur Schaffung eines Systems, das die Verantwortung und die Kontrolle für individuelle und sich dynamisch verändernde Software in die Hände des Anwenders legt. Lean-Prinzipien und Methoden agiler Softwareentwicklung bilden das Fundament von DevOps. Mit Industrial DevOps wird dies auf das industrielle Umfeld zugeschnitten.

1.3 Planung und Ablauf des Vorhabens

Abbildung 8

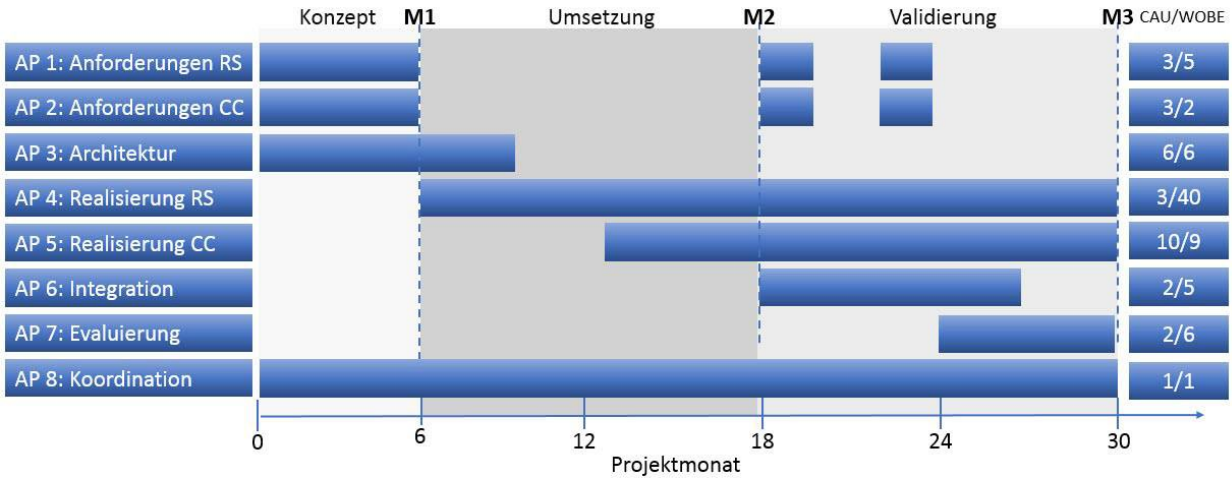

Abbildung 8: Zeitplanung und Projektaufwand

Das titan-Projekt besteht aus acht Arbeitspaketen und drei Meilensteinen. Die Zeitplanung und der Projektaufwand sind in Abbildung 8: Zeitplanung und Projektaufwand Abbildung 8 : Zeitplanung und Projektaufwand, dargestellt. Der Aufwand ist in Personenmonaten angegeben.

1.3.1 Arbeitspaket 1: Anforderungsanalyse titan-Plattform

AP1	**Anforderungsanalyse titan-Plattform**
Leitung:	WOBE **Aufwand:** CAU 3 PM, WOBE 5 PM
Aufgabe:	
Es sind die Funktionen der titan-Plattform bezüglich User-Interface und Backend zu ermitteln und zu dokumentieren. Die Dokumentation muss zu Beginn festgelegten Qualitätsmerkmalen entsprechen. Neben den funktionalen Anforderungen sind auch die so genannten nicht-funktionalen Anforderungen wie Performance, Skalierbarkeit, Resilienz, usw. zu beschreiben.	
Ergebnisse:	

WOBE
• Geeignete Dokumentation aller Anforderungen
  ○ User-Stories für Text basierte Dokumentation
  ○ Grafische Darstellungen von UI Anforderungen
  ○ Weitere grafische Darstellungen z.B. Use-Case Diagramme
  ○ Akzeptanzkriterien für Anforderungen festlegen
  ○ Beschreibung nichtfunktionaler Anforderungen mit Risikoeinschätzung für das Projekt
• Relative Abschätzung des Aufwands für die Implementierung der Anforderungen
• Priorisiertes Backlog für die Umsetzung der Anforderungen
CAU & WOBE
• **Katalog mit Anwendungsszenarien**
• Dokumentation einer Stakeholder Analyse	

Tätigkeiten:	WOBE & CAU
• Regelmäßige Treffen der Projektpartner zu Workshops • Formulierung von Anforderungen und Überprüfung gegenüber vereinbarter Qualitätsrichtlinien • Regelmäßige Präsentation und Diskussion der formulierten Anforderungen mit den assoziierten Partnern KN und IBAK im Kontext des jeweiligen Anwendungsfalls	
Unteraufgaben:	
AP1-1:	Nicht-funktionale Anforderungen der titan-Plattform
AP1-2:	Funktionale Anforderungen titan-UI
AP1-3:	Funktionale Anforderungen titan-Backend

AP1-1	**Nicht-funktionale Anforderungen der titan-Plattform**
Leitung: WOBE	**Aufwand:** CAU 0,45 PM, WOBE 0,75 PM

Aufgabe:

Die für den Einsatz von titan im industriellen Umfeld erforderlichen nicht-funktionalen Anforderungen sollen ermittelt, mit den assoziierten Projektpartnern abgestimmt und geeignet dokumentiert werden.

Ergebnisse: CAU & WOBE

- Performance und Skalierbarkeit
- Verfügbarkeit, Resilienz
- Sicherheit
- Compliance
- Einsetzbarkeit (welcher Aufwand ist nötig um die Software zu betreiben?)

Tätigkeiten:

WOBE
- Dokumentation der Anforderungen
- Erstellen von Akzeptanzkriterien

CAU & WOBE
- Festlegung der Anforderungen und Vergleich mit ähnlichen Lösungen
- Abstimmung der gefundenen Anforderungen mit den assoziierten Projektpartnern

AP1-2	**Funktionale Anforderungen titan-UI**
Leitung: WOBE	**Aufwand:** CAU 1,5 PM, WOBE 2,5 PM

Aufgabe:

Die Anforderungen der Benutzerschnittstelle werden ermittelt und geeignet dokumentiert.

Ergebnisse: CAU & WOBE

- User-Stories in einem priorisierten Backlog
- Grafische Darstellung des UIs (Mockups) und Beschreibung der Benutzerinteraktionen

Tätigkeiten:

WOBE
- Rahmengebende Ziele für das Benutzerinterface formulieren und dokumentieren
- Erstellen von Mockups und Beschreibungen der Benutzerinteraktionen

CAU & WOBE
- Anforderungen im Projektteam ermitteln und dokumentieren
- Überprüfung der Anforderungen mir den assoziierten Projektpartnern

AP-Nr.:	AP1-3	**Titel:**	**Funktionale Anforderungen titan-Backend**
Leitung:	WOBE	**Aufwand:**	CAU 1,05 PM, WOBE 1,75 PM

Aufgabe:

Das titan-Backend, also die auf den Servern ablaufenden Prozesse, bietet alle Funktionen, die für einen Industrial DevOps Workflow bei der Systemintegration erforderliche sind. Diese Funktionen werden in AP1-3 dokumentiert und abgestimmt.

Ergebnisse: CAU & WOBE

- User-Stories in einem priorisierten Backlog
- Grafische Darstellungen von Abläufen falls erforderlich

Tätigkeiten: CAU & WOBE

- Anforderungen im Projektteam ermitteln und dokumentieren
- Überprüfung der Anforderungen mit den assoziierten Projektpartnern

1.3.2 Arbeitspaket 2: Anforderungsanalyse titan CC

AP2		**Anforderungsanalyse titan Control Center**	
Leitung:	CAU	**Aufwand**:	CAU 3 PM, WOBE 2 PM

Aufgabe:

Es sind die Funktionen des titan Control Centers zu ermitteln und zu dokumentieren. Die Anforderungen werden gemeinsam mit den assoziierten Partnern KN und IBAK ermittelt und abgestimmt. Die Dokumentation muss zu Beginn festgelegten Qualitätsmerkmalen entsprechen.

Ergebnisse:

CAU

- Geeignete Dokumentation aller Anforderungen
  - User-Stories für Text basierte Dokumentation
  - Beschreibung des Workflows
  - Beschreibung der erfassten Daten und deren Darstellung
  - Akzeptanzkriterien für Anforderungen festlegen
  - Beschreibung nichtfunktionaler Anforderungen mit Risikoeinschätzung für das Projekt
- Dokumentation einer Stakeholder-Analyse
- Relative Abschätzung des Aufwands für die Implementierung der Anforderungen
- Priorisiertes Backlog für die Umsetzung der Anforderungen

Tätigkeiten:

CAU & WOBE

- Regelmäßige Treffen der Projektpartner zu Workshops
- Regelmäßige Refinement-Workshops gemeinsam mit den assoziierten Partnern KN und IBAK
  CAU
- Formulierung von Anforderungen und Überprüfung gegenüber vereinbarter Qualitätsrichtlinien

1.3.3 Arbeitspaket 3: Architektur

AP3	**Architektur**
Leitung: CAU	**Aufwand:** CAU 6 PM, WOBE 6 PM

Aufgabe:

Basierend auf der Anforderungsanalyse aus AP1 und AP2 wird eine Architektur für das titan-System erarbeitet und dokumentiert. Die Umsetzung des Projekts mit agilen Methoden soll berücksichtigt werden. Die Architektur muss dafür flexibel genug sein, damit Änderungen der Anforderungen zu jeder Zeit möglich sind.

Ergebnisse: CAU & WOBE

- Geeignete Dokumentation der Architektur
  - Darstellung der verschiedenen Abstraktionsebenen
  - Grafische Darstellung der Module, Komponenten und Klassen
  - Beschreibung von Schnittstellen, Daten und Prozessen

Tätigkeiten: CAU & WOBE

- Regelmäßige Treffen der Projektpartner zu Workshops
- Dokumentation der Architektur

Unteraufgaben:

AP3-1:	Architektur der titan-Laufzeitumgebung
AP3-2:	Architektur der titan-Entwicklungs- und Test-Umgebung
AP3-3:	Architektur der Benutzerschnittstelle
AP3-4:	Installierbare Komponenten
AP3-5:	Grafische Beschreibungssprache für Flows
AP3-6:	Control Center Systemarchitektur

AP3-1	**Architektur der titan-Laufzeitumgebung**
Leitung: CAU	**Aufwand:** CAU 1,2 PM, WOBE 1,2 PM

Aufgabe:

Basierend auf den wesentlichen Anforderungen bezüglich Ausfallsicherheit und Skalierbarkeit wird eine geeignete Architektur für die titan-Laufzeitumgebung festgelegt. Prozesse, Module, Komponenten, Klassen und Daten werden geeignet dokumentiert.

Ergebnisse:

WOBE
- Dokumentation der Prozesse, Module, Komponenten und Klassen der titan-Laufzeitumgebung (bevorzugt grafische Darstellungen)
- Beschreibung des Toolstacks für die Umsetzung

CAU & WOBE
- Systemarchitektur mit Datenbanken, verteilten Dateisystemen usw.
- Beschreibung von geeigneten Metriken zur Überwachung der Laufzeitumgebung

Tätigkeiten:

CAU & WOBE
- Workshops der Projektpartner CAU und WOBE
- Regelmäßige Überprüfung und Verfeinerung

WOBE
- Dokumentation der Architektur

AP3-2	**Architektur der titan-Entwicklungs- und Test-Umgebung**
Leitung: CAU	**Aufwand:** CAU 1,2 PM, WOBE 1,2 PM

Aufgabe:

titan integriert eine Entwicklungs- und Test-Umgebung für Flows und Werkzeuge zur Qualitätsüberprüfung von Komponenten (Unit Tests, Statische Codeanalyse, usw.). Die Architektur der dafür benötigten Systembestandteile sollen festgelegt und dokumentiert werden.

Ergebnisse:

WOBE
- Dokumentation der Prozesse, Module, Komponenten und Klassen der titan-Entwicklungs- und Test-Umgebung (bevorzugt grafische Darstellungen)
- Beschreibung des Toolstacks für die Umsetzung

CAU & WOBE
- Systemarchitektur, integrierte Werkzeuge, Datenbanken, usw.
- Beschreibung von geeigneten Metriken zur Qualitätsüberprüfung

Tätigkeiten:

CAU & WOBE
- Workshops der Projektpartner CAU und WOBE
- Regelmäßige Überprüfung und Verfeinerung

WOBE

- Dokumentation der Architektur

AP-Nr.:	AP3-3	**Architektur der Benutzerschnittstelle**	
Leitung:	CAU	**Aufwand:**	CAU 0,6 PM, WOBE 0,6 PM

Aufgabe:

Für die titan-Benutzerschnittstelle wird basierend auf den Anforderungen aus AP1 eine geeignete Architektur festgelegt und dokumentiert.

Ergebnisse:

WOBE
- Dokumentation der Module, Komponenten und Klassen des titan-UIs (bevorzugt grafische Darstellungen)
- Beschreibung der Schnittstellen und der Kommunikation mit dem titan-Kern
- Beschreibung des Toolstacks für die Umsetzung
- Systemarchitektur im Betrieb

CAU & WOBE
- Beschreibung von geeigneten Metriken für die Überwachung des UIs im Betrieb

Tätigkeiten:

WOBE
- Dokumentation der Architektur

CAU & WOBE
- Workshops der Projektpartner CAU und WOBE
- Regelmäßige Überprüfung und Verfeinerung

AP3-4		**Installierbare Komponenten**	
Leitung:	CAU	**Aufwand:**	CAU 0,6 PM, WOBE 0,6 PM

Aufgabe:

Beschreibung der Schnittstelle und Datenstruktur für installierbare Komponenten.

Ergebnisse: CAU & WOBE

- Dokumentation der Datenstruktur installierbarer Komponenten
- Dokumentation des Versionsmanagements von Komponenten
- Dokumentation des Qualitätsmanagements von Komponenten

Tätigkeiten: CAU & WOBE

- Workshops der Projektpartner CAU und WOBE
- Dokumentation der Architektur
- Regelmäßige Überprüfung und Verfeinerung

AP-Nr.:	AP3-5	**Grafische Beschreibungssprache für Flows**

Leitung: CAU **Aufwand:** CAU 1,8 PM, WOBE 1,8 PM

Aufgabe:

Für die Beschreibung von Flows im titan-System wird eine Datenstruktur und eine geeignete grafische Beschreibungssprache festgelegt und dokumentiert.

Ergebnisse: CAU & WOBE

- Dokumentation der Flow-Daten im System
- Dokumentation der Daten im Flow und deren Harmonisierung zwischen Knoten
- Dokumentation der schematischen, grafischen Darstellung in einer allgemeinen Form

Tätigkeiten:

WOBE
- Dokumentation der Architektur

CAU & WOBE
- Workshops der Projektpartner CAU und WOBE
- Regelmäßige Überprüfung und Verfeinerung

AP3-6 **Control Center System-Architektur**

Leitung: CAU **Aufwand:** CAU 0,6 PM, WOBE 0,6 PM

Aufgabe:

Basierend auf den Ergebnissen aus AP2 wird für die Control Center Anwendung bei den assoziierten Partnern KN und IBAK wird eine Systemarchitektur festgelegt und dokumentiert. Die für die Integration von Sensoren und Datenquelle, sowie Management Information Systemen benötigten titan-Komponenten werden beschrieben.

Ergebnisse: CAU

- Dokumentation der Systemarchitektur inkl. integrierter Systemkomponenten
  - Dokumentation der Unterschiede bei IBAK und KN
- Dokumentation der benötigten Komponenten
- Dokumentation der Daten

Tätigkeiten:

CAU & WOBE
- Workshops der Projektpartner CAU und WOBE
- Regelmäßige Überprüfung und Verfeinerung

CAU
- Dokumentation der Architektur

1.3.4 Arbeitspaket 4: Realisierung der titan-Plattform

AP4	**Realisierung der titan-Plattform**
Leitung:	WOBE **Aufwand:** CAU 3 PM, WOBE 40 PM

Aufgabe:

Basierend auf den Ergebnissen von AP1 und AP3 wird ein Prototyp des titan-Systems realisiert. Die Arbeit wird agil organisiert und die Software iterativ entwickelt. Neben der Programmierung der eigentlichen Funktionalität, wird eine Testsuite für automatische Regressionstests entwickelt, die Tests nicht-funktionaler Anforderungen wir Performance und Sicherheit einschließt.
Für die wissenschaftliche Untersuchung der Grid-Architektur werden Tests und Werkzeuge erstellt, angewendet und ausgewertet.

Ergebnisse:

WOBE
- Software des titan-Kernsystems
- Testsuite für Regressionstests
- Dokumentation des Codes für Entwickler (eng)
- Dokumentation des Systems (Installation, Betrieb)
- Dokumentation der Flow-Programmierung

CAU & WOBE
- Tests und Werkzeuge für die Untersuchung der Grid-Architektur und des Flows

Tätigkeiten:

WOBE
- Meetings
  - Refinement – Verfeinern der Anforderungen im Backlog
  - Sprint-Planung – Festlegen der Arbeiten für die nächste Iteration
  - Stand-up – Tägliche, kurze Diskussion der anstehenden Arbeiten
  - Retrospektiven – Diskussionen von Verbesserungen der aktuellen Arbeitsweisen

CAU & WOBE
- Entwicklung neuer Funktionen
- Refactoring – Anwenden neuer Erkenntnisse auf bereits implementierte Codezeilen

Unteraufgaben:

AP4-1:	Implementierung der titan-Laufzeitumgebung
AP4-2:	Implementierung der titan-Entwicklungs- und Test-Umgebung
AP4-3:	Bibliothek für Visuelle Programmierung des Flows erstellen
AP4-4:	Implementierung der titan-Benutzerschnittstelle (UI)

AP4-5:	Datensicherheit implementieren
AP4-6:	Entwicklung von Tests und Werkzeugen für die Untersuchung der Grid-Architektur und des Flows

AP4-1	**Implementierung der titan-Laufzeitumgebung**		
Leitung:	WOBE	**Aufwand:**	CAU 0 PM, WOBE 8,8 PM

Aufgabe:

Implementieren der titan-Laufzeitumgebung

Ergebnisse: WOBE

- Services und Prozesse für die Ausführung von Flows und die Sammlung ausführungsspezifischer Metriken

Tätigkeiten: WOBE

- Implementierung der erforderlichen Dienste und Prozesse
- Erstellen und implementieren geeigneter Datenstrukturen
- Erstellen geeigneter Dokumentation
- Regelmäßige Überprüfung und Verfeinerung

AP4-2	**Implementierung der titan-Entwicklungs- und Test-Umgebung**		
Leitung:	WOBE	**Aufwand:**	CAU 0 PM, WOBE 10,7 PM

Aufgabe:

Implementieren der titan-Entwicklungs- und Test-Umgebung auf den Servern.

Ergebnisse: WOBE

- Services und Prozesse für Paketmanagement, Tests von Komponenten und Flows und Datensandbox für produktionsähnliche Tests implementieren und dokumentieren.

Tätigkeiten: WOBE

- Implementierung der erforderlichen Dienste und Prozesse
- Erstellen und implementieren geeigneter Datenstrukturen
- Erstellen geeigneter Dokumentation
- Regelmäßige Überprüfung und Verfeinerung

AP4-3	**Bibliothek für Visuelle Programmierung des Flows erstellen**		
Leitung:	WOBE	**Aufwand:**	CAU 0 PM, WOBE 10,7 PM

Aufgabe:

Grafische Controls und Komponenten für die Erstellung und Verwaltung von Flows entwickeln.

Ergebnisse: WOBE

- Bibliothek grafischer Komponenten für die UI-Entwicklung
- Dokumentation
- Beispiele

Tätigkeiten: WOBE

- Design der Komponenten und Bedienkonzepte basierend auf den Ergebnissen von AP1-2
- Implementierung und Test der Komponenten
- Erstellen geeigneter Dokumentation
- Regelmäßige Überprüfung und Verfeinerung

AP4-4 **Implementierung der titan-Benutzerschnittstelle (UI)**

Leitung: WOBE **Aufwand:** CAU 0 PM, WOBE 4,9 PM

Aufgabe:

Grafische Benutzeroberfläche für den Einsatz auf Desktop und mobilen Geräten mit durchgängiger Touch- und Gestenbedienung. Neben der Laufzeitüberwachung und Steuerung wird die Verwaltung des Systems und die Entwicklung von Flows integriert.

Ergebnisse: WOBE

- Software der grafischen Benutzeroberfläche (UI)
- Dokumentation

Tätigkeiten: WOBE

- Implementierung des UIs basierend auf den Ergebnissen von AP1-2 und AP3-3.
- Erstellen geeigneter Dokumentation
- Regelmäßige Überprüfung und Verfeinerung

AP4-5 **Datensicherheit implementieren**

Leitung: WOBE **Aufwand:** CAU 0 PM, WOBE 2,45 PM

Aufgabe:

Für die im System verarbeiteten Daten wird Datensicherheit mittels Verschlüsselung implementiert.

Ergebnisse: WOBE

- Funktionen für Ende zu Ende Verschlüsselung von zu übertragenden Daten
- Funktionen für die Verschlüsselung auf der Nachrichtenschicht
- Funktionen für die sichere Authentifizierung von Benutzern am System
- Dokumentation

Tätigkeiten: WOBE

- Implementierung und Test der Verschlüsselungsfunktionen
- Erstellen geeigneter Dokumentation
- Regelmäßige Überprüfung und Verfeinerung

AP4-6	**Entwicklung von Tests und Werkzeugen für die Untersuchung der Grid-Architektur und des Flows**		
Leitung:	CAU	**Aufwand:**	CAU 3 PM, WOBE 2,45 PM

Aufgabe:

Für die wissenschaftliche Überprüfung der experimentellen Architektur des titan-Systems werden Tests und Werkzeuge entwickelt.

Ergebnisse:

- Test und Werkzeuge für die wissenschaftliche Überprüfung der experimentellen Architektur hinsichtlich Wartbarkeit, Skalierbarkeit und Ausfallsicherheit

Tätigkeiten:

- Design von Tests
- Entwicklung geeigneter Werkzeuge für die Durchführung der Tests

1.3.5 Arbeitspaket 5: Realisierung titan-CC

AP5	**Realisierung titan-CC**		
Leitung:	CAU	**Aufwand:**	CAU 10 PM, WOBE 10 PM

Aufgabe:

Basierend auf den Ergebnissen von AP2 und AP3 wird ein Prototyp des titan Control Centers basierend auf dem titan System implementiert. Die Arbeit wird agil organisiert und die Software iterativ entwickelt.

Ergebnisse:

- Domänenspezifische Komponentenpakete und Plugins
- Anwendungsspezifisches UI + Dokumentation
- Flows für die Abbildung der Anforderungen

Tätigkeiten:

CAU

- Implementierung der Komponenten

- Implementierung des UIs
- Refactoring – Anwenden neuer Erkenntnisse auf bereits implementierte Codezeilen

WOBE & CAU

- Enge Zusammenarbeit von CAU und wobe
- Regelmäßige Abstimmung (Review) der Arbeitsergebnisse mit den Partnern KN und IBAK

Bemerkung:

Das titan Control Center wird basierend auf der titan-Plattform entwickelt. Es wird kontinuierlich geprüft, wie das Control Center mit der Plattform interagiert. Die daraus gewonnenen Erkenntnisse fließen in die jeweils nächste Iteration des Projektes ein.

Unteraufgaben:

AP5-1:	Anwendungsspezifische Komponenten entwickeln
AP5-2:	Testsuite für die titan-Qualitätssicherung erstellen
AP5-3:	Anwendungsspezifisches UI Entwickeln

AP5-1	**Anwendungsspezifische Komponenten entwickeln**		
Leitung:	CAU	**Aufwand:**	CAU 2,6 PM, WOBE 2 PM

Aufgabe:

Entwickeln anwendungsspezifischer titan Komponenten für die Integration der benötigten Systeme.

Ergebnisse: WOBE & CAU

- Packages mit Komponenten für den Einsatz von titan im Kontext des Energiemanagements

Tätigkeiten: WOBE & CAU

- Entwicklung der Komponenten

AP5-2	**Testsuite für die titan Qualitätssicherung erstellen**		
Leitung:	CAU	**Aufwand:**	CAU 2,6 PM, WOBE 2 PM

Aufgabe:

Für die Komponenten aus AP5-1 wird eine Testsuite für die Qualitätssicherung in das Paket integriert.

Ergebnisse: WOBE & CAU

- Packages mit integrierter Testsuite

Tätigkeiten: WOBE &CAU

- Entwicklung der Testfälle für die Komponenten

AP5-3	**Anwendungsspezifisches UI entwickeln**

Leitung: CAU **Aufwand:** CAU 6,5 PM, WOBE 6 PM

Aufgabe:

Für den Fall des Energiemanagements wird ein anwendungsspezifisches UI entwickelt, das Anwendern als zentrale Informationsschnittstelle dient.

Ergebnisse: WOBE & CAU

- Anwendungsspezifisches UI
- Dokumentation

Tätigkeiten: WOBE & CAU

- UI Entwickeln und Testbar machen
- Dokumentation erstellen
- Feedback von Anwendern einholen und Anforderungen verfeinern

1.3.6 Arbeitspaket 6: Integration

AP6	**Integration**

Leitung: WOBE **Aufwand:** CAU 2 PM, WOBE 5 PM

Aufgabe:

Der titan-Prototyp wird gemeinsam mit den Anwendern KN und IBAK in den produktiven Betrieb übernommen. Dabei ist es wichtig für die spätere Auswertung und für praxisnahe Verbesserungen, das verantwortliche Personal zu schulen und eng mit einzubeziehen.

Ergebnisse:

- Produktiv eingesetztes System bei IBAK und KN
- Verfeinerungen der Anforderungen basierend auf den Erfahrungen des Produktiven Einsatzes

Tätigkeiten:

WOBE

- Installation von Systembestandteilen bei den Anwendern
- Ausbildung des IT-Personals
- Ausbildung der Prozessverantwortlichen

WOBE & CAU

- Feedback dokumentieren und Verbesserungsvorschläge für die Implementierung in der nächsten Iteration vorbereiten

- Das System im laufenden Betrieb beobachten und basierend auf den gesammelten Daten regelmäßig verbessern

1.3.7 Evaluierung

AP7	**Evaluierung**		
Leitung:	CAU	**Aufwand:**	CAU 2 PM, WOBE 6 PM

Aufgabe:

Evaluierung der gefundenen Projektergebnisse gegenüber den gesetzten Zielen. Vergleich der in AP1 gefundenen Anwendungsszenarien mit dem erreichten Stand der Software.

Ergebnisse:

CAU & WOBE
- Dokumentation der Projektergebnisse
- Implementierungen von Flows und Komponenten als Beispiele für die gefundenen Anwendungsszenarien
- Auswertung der Erkenntnisse bezüglich der Grid-Architektur hinsichtlich Wartbarkeit, Skalierbarkeit und Ausfallsicherheit.
- Ergebnisse der Untersuchung über die Anwendung von DevOps Methoden auf die Problemstellung in der Industrie
- Dokumentation weiterer Schritte nach Projektende
- Beurteilung des Systems hinsichtlich Compliance und Sicherheit

WOBE
- Erfassung und Auswertung der Benutzererfahrung mit dem UI und Flow basierter Automatisierung

Tätigkeiten:

CAU
- Interviews mit den Anwendern IBAK und KN bezüglich ihrer Sicht auf die Projektergebnisse
- Erfassung der Effektivität des Systems beim Einsatz im Energiemanagement
- Interviews mit den Anwendern IBAK und KN bezüglich ihrer Sicht auf die Projektergebnisse

WOBE & CAU
- Vergleich des Systems mit gängigen Standards und Vorschriften
- Untersuchung der Sicherheitsaspekte des Systems
- Durchführung von Tests in der Produktionsumgebung
- Workshops der Projektbeteiligten und Sammlung der rückblickenden Beurteilung sowie Einschätzungen der Zukunftsperspektiven zum Zeitpunkt der Evaluierung

1.3.8 Koordination und Projektmanagement

AP8	**Koordination und Projektmanagement**		
Leitung:	WOBE	**Aufwand:**	CAU 1 PM, WOBE 1 PM
Aufgabe:			
Durchführung des Projektmanagements, agiler Methoden und geeigneter Visualisierungstechniken.			
Ergebnisse:	WOBE		
• Team Charta • Kommunikationsinfrastruktur • Visualisierung von Projektfortschritt und Status			
Tätigkeiten:	WOBE & CAU		
• Koordination der Kommunikation innerhalb der Teams und mit den Stakeholdern. • Aufbau einer Teamkultur für einen reibungslosen Ablauf des Informationsflusses. • Visualisierung von Projektfortschritt und Status • Organisation von Dokumentationen			

1.3.9 Meilensteine (Festlegung von Meilensteinen und ggf. Abbruchkriterien; interne im Projektkonsortium und externe mit Beteiligung des BMBF und des PT)

Das titan-Projekt besteht aus drei Phasen, die jeweils durch einen Meilenstein abgeschlossen werden.

1.3.9.1 M1 Konzept

Während der Konzeptionsphase konzentriert sich die Arbeit auf die Ermittlung der Anforderungen, die Dokumentation der Architektur des Systems und die Schaffung der Rahmenbedingungen für die Implementierung der Software. Diese Phase endet mit dem Beginn der Implementierung.

1.3.9.2 M2 Umsetzung

Der Schwerpunkt der Umsetzungsphase ist die Implementierung des Systems. Kontinuierliche Integration der Komponenten und häufige automatische Tests stellen dabei die Qualität sicher und ermöglichen es, den Stand des Projekts jederzeit zu überprüfen. Die Umsetzungsphase endet mit dem Beginn der Inbetriebnahme der Prototypen bei den Kieler Nachrichten und IBAK.

1.3.9.3 M3 Validierung

Während der Validierungsphase werden die Anforderungen des Systems gemeinsam mit den Kieler Nachrichten und IBAK weiter verfeinert. Grundlage dafür sind die im Betrieb gesammelten Daten und Erfahrungen. Die kontinuierliche Verbesserung und die häufige

Inbetriebnahme neuer Funktionen ermöglicht eine anschließende Evaluierung des Gesamtprojekts bezogen auf ein praktisches Beispiel. Die Validierungsphase endet mit der Projektlaufzeit. Auf Grundlage der Validierungserfahrungen wird der Verwertungsplan überprüft und verfeinert.

1.4 Wissenschaftlicher und technischer Stand zum Beginn des Projekts

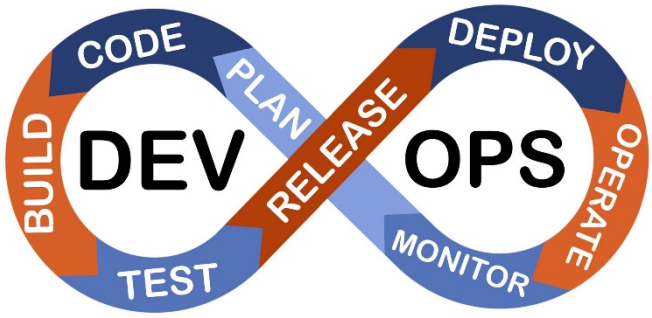

Abbildung 9: DevOps Darstellung

DevOps bezeichnet ein Vorgehen, bei dem die für den Betrieb eines Softwaresystems zuständigen Abteilungen eng in den Entwicklungs- und Verbesserungs-Prozess eingebunden sind.

Werden weitere Abteilungen wie Finanzen, Vertrieb oder Marketing einbezogen, spricht man von „BizDevOps“. DevOps macht bei der Integration nicht an den Schnittstellen der IT-Systeme halt, sondern behandelt ein System als Ganzes. Insbesondere werden die Bedürfnisse der Menschen, die in und mit einem System arbeiten, berücksichtigt. Bisher wird DevOps überwiegend für Internet-Anwendungen, wie z.B. eCommerce eingesetzt. Mit titan planen wir dies auf das industrielle Umfeld auszudehnen.

Speziell im industriellen Umfeld stellen die Produktionsabteilungen eine wichtige Gruppe dar, die in den DevOps-Zyklus, der in Abbildung 9 dargestellt ist, eingebunden werden muss. DevOps-Tools und -Praktiken werden an die speziellen Anforderungen der Industrie angepasst. Stabilität und Flexibilität integrierter Systeme können so erheblich verbessert werden.

Die titan-Plattform unterstützt diese Technik und integriert Entwicklung und Betrieb in ein System. Verbesserungen können kontinuierlich hinzugefügt, evaluiert und in Betrieb genommen werden (Continuous Integration CI, Continuous Deployment CD).

Abbildung 10: Lebensphasen von Softwaresystemen

Individualsoftware: Die Abbildung 10 zeigt die traditionellen Phasen der Lebensspanne von Individualsoftware, wie sie aktuell typisch sind. Auf eine isolierte Entwicklungsphase folgt eine Phase der aktiven Softwarepflege und danach eine Phase, in der die Wartung der Software aufrechterhalten wird. Oft wird die Software jedoch auch nach dem Ende der letzten Phase weiter betrieben. Dadurch entsteht ein hohes Risiko für den Anwender. Änderungen sind schon während der Servicephase nicht oder nur eingeschränkt möglich.

Je länger ein System im Einsatz ist, desto mehr verschlechtert sich dessen Wartbarkeit. Ein möglicher und häufiger Grund dafür ist die mangelhafte innere Qualität von Software. Werden neue Projekte für die Erweiterung eines bestehenden Systems durchgeführt, müssen sich diese in bestehende Architekturkonzepten eingliedern. Diese sogenannten Brownfield Projekte erfordern zeitintensive Untersuchungen von Legacy Code, der strukturelle Rahmenbedingungen vorgibt und teilweise so alt ist, dass die notwendigen Fähigkeiten diesen Code zu verstehen, in Entwicklungsteams nicht, oder kaum noch vorhanden sind.

Agile Softwareentwicklung: Um komplexe Anforderungen, die sich stetig ändern, in den Griff zu bekommen, werden die Methoden der agilen Softwareentwicklung eingesetzt. Enge Zusammenarbeit aller Beteiligten und klare strukturierte Kommunikation basierend auf gemeinsam verabredeten Regeln und Werten steigern die Effizienz von Softwareprojekten und sind geeignet, Wunschdenken und unrealistische Erwartungen zu reduzieren. Der Aufwand, der nötig ist, das dringend benötigte Alignment, also die Ausrichtung aller Beteiligten auf gemeinsame Ziele, herzustellen und aufrechtzuerhalten, ist für extrem langlaufende Projekte problematisch und nur dann realistisch, wenn ein Unternehmen bereit ist, eigene Entwicklungsteams für seine Systeme dauerhaft einzusetzen und agil zu

managen. Dieser Weg steht kleinen und mittleren Betrieben aufgrund der damit verbundenen Kosten und des Risikos jedoch meist nicht offen.

Tests und Qualität: Ein Systemtest soll eine Software in einer Testumgebung verifizieren, die der späteren Produktionsumgebung sehr ähnlich ist. Ein isoliertes Testsystem zur Verfügung zu stellen, um eine verteilte und integrierte Produktionsumgebung zu simulieren, ist jedoch nur schwer realisierbar, teuer und birgt das Risiko, dass die Ergebnisse des Systemtests zwar die Funktionsfähigkeit der Software auf dem Testsystem bestätigen, jedoch das Risiko für ein fehlerhaftes Verhalten auf dem Produktivsystem nicht, oder nicht ausreichend geprüft werden kann.

Auf Tests einzelner Komponenten, Unittests, Integrationstests und funktionale Tests hat der Anwender in der Regel keinen Einfluss. Sie werden vom Softwarehersteller in mehr oder weniger ausreichendem Umfang durchgeführt.

Bezüglich der inneren Qualität von Software bleibt dem Anwender ebenfalls nichts weiter übrig, als dem Softwarehersteller seiner Wahl zu vertrauen.

Microservices: Große Internetsysteme wie die von Amazon, Google oder Otto.de stehen vor ähnlichen Problemen und nutzen die Aufteilung ihrer Webanwendung in domänenspezifische Dienste (Microservices), die keine, oder nur wenige Abhängigkeiten haben. Die Vermeidung von Abhängigkeiten reduziert dabei den Kommunikationsaufwand und erleichtert die Einführung neuer Dienste durch weitere, neue Teams. Microservices lassen sich leicht durch komplett neue Implementierungen ersetzen, wenn der Aufwand Legacy Code zu pflegen unrealistisch hoch werden würde. Systemtests lassen sich durch gezielte, gekapselte Untersuchungen auf dem Produktivsystem durchführen. Das Problem des Alignments lösen diese Unternehmen durch den Einsatz eigener Entwicklungsteams.

1.4.1 Angabe bekannter Konstruktionen, Verfahren und Schutzrechte, die für die Durchführung des Vorhabens benutzt wurden

Um das gemeinsame Verständnis zu verbessern, fahren wir mit den bereits schon genutzten Methoden fort:

- Definition von weiteren projekt-bezogenen Begriffen als geteiltes Verständnis

Um eine bessere Kommunikation zu gewährleisten, wurden für das Projekt relevante Begriffe diskutiert und definiert. Die Begriffe und Definitionen sind dokumentiert. Dieser Prozess wird fortlaufend weitergeführt.

- Visuelle Darstellung von technischen / software-spezifischen Aspekten

Zum besseren Verständnis versuchen wir, alle technischen und software-spezifischen Aspekte zu visualisieren.

Hier folgen einige Beispiele:

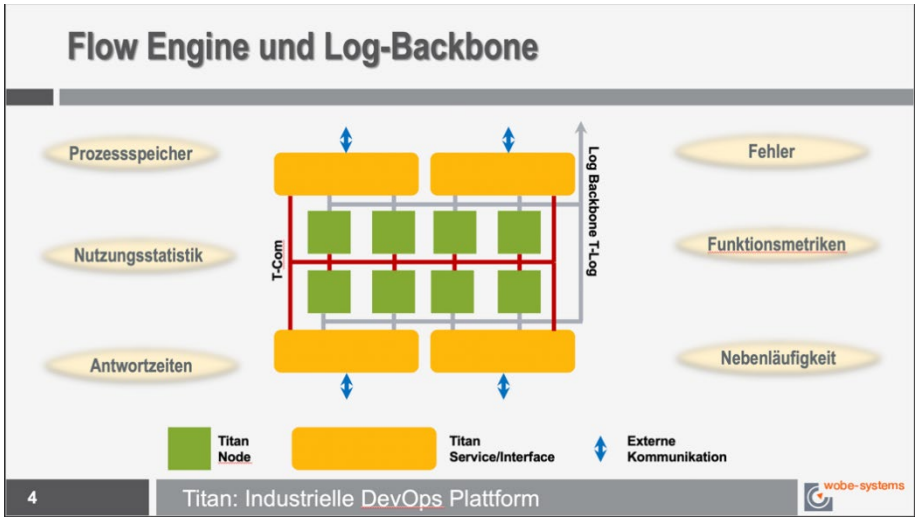

Abbildung 11: Flow Engine Log-Backbone

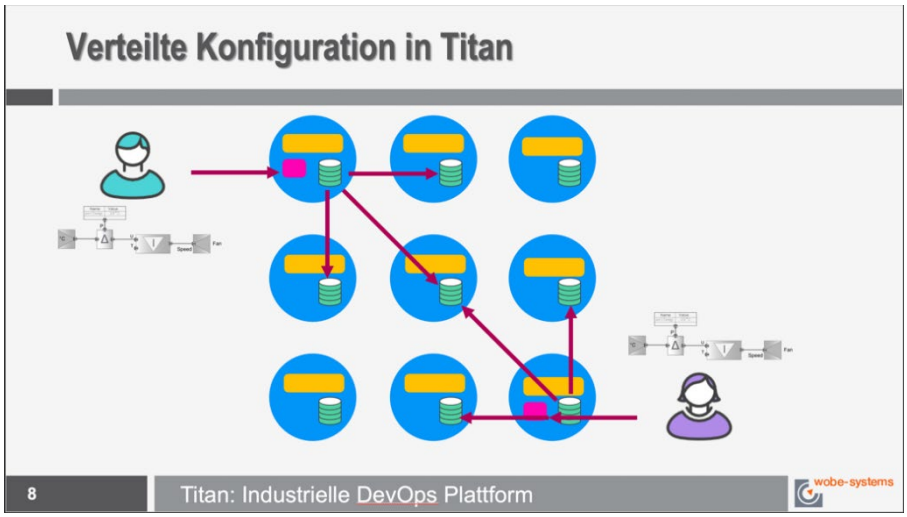

Abbildung 12: Verteilte Konfiguration im titan-Projekt

- Erarbeitung von Clean Code Praktiken

  Um mit der titan-Plattform ein langlebiges Softwaresystem zu schaffen, werden bereits frühzeitig Praktiken und Tools integriert, die die Entwicklung von Clean Code fördern sollen. Großer Wert wurde hierbei auf die Berücksichtigung von Ergebnissen sowohl aus der Forschung als auch aus der industriellen Praxis gelegt. Die Ergebnisse hieraus wurden mit allen Projektbeteiligten im Rahmen eines regulären Statusmeetings diskutiert.

1.4.2 Schaffung technischer Voraussetzungen

Es wurden weitere technische Voraussetzungen für das titan Projekt geschaffen. Dazu gehören folgende Aspekte:

- Begleitung der Master Arbeit:
  „Simulating Actor Based Execution of Flow Models in Cloud Environments”. Untersuchung zweier Routing Mechanismen durch Simulation für die Weiterleitung und Verteilung von Flow Packets innerhalb eines Multi Node Grids mit Hinblick auf den Aufbau der titan Software Plattform. Die Simulation ergab, dass bei der Implementierung eines eigenen Routing Mechanismus für titan bei Nutzung eines Round Robin Verfahrens eine gleichmäßigere Auslastung der Knoten innerhalb des Grids zu erwarten ist.

- Zusammenführung der prototypischen Arbeiten am Control Center und an der Flow Engine

  Der im Rahmen einer Masterarbeit entstandene Prototyp des Control Centers sollte frühzeitig während der Implementierung der Flow Engine eingebunden werden. Notwendige Schritte wurden dazu auch während des Off-Sites besprochen und in den folgenden Monaten umgesetzt.

- Erweiterung der titan Infrastruktur

  Mit einer Monitoring Lösung auf Basis des ELK-Stack im Rahmen eines studentischen Master-Projektes. Identifizierung von Metriken, die auch für die titan Plattform von Relevanz sind. Erarbeitung des Verständnisses, wie Business Value Metriken im Rahmen des Monitorings erfasst werden können.

- Weiterentwicklung UjoTypes-C

  Erweiterung um die nach den Anforderungen der titan Plattform benötigten Datentypen. Reduzierung der Komplexität des Sourcecodes in Bezug auf verbesserte Lesbarkeit, Wartbarkeit und Langlebigkeit. Überarbeitung der durch die Statische Code Analyse aufgedeckten Schwachpunkte in Bezug auf Secure Coding.

- Weiterentwicklung UjoTypes-py

  Erweiterung um die nach den Anforderungen der titan Plattform benötigten Datentypen. Programmierung eines Klassenwrappers der eine sprachtypischere Verwendung von UjoTypes in Python ermöglicht.

- Erarbeitung eines Beschreibungssprache für Datenmodelle

  Für eine einheitliche Definition von Daten wird eine Schemabeschreibung entwickelt.

- Entwicklung von tLog-py

  Basis Bibliothek zum Versenden von Log- und Metrik-Daten innerhalb der titan Software Plattform. Die von der Bibliothek erzeugten Daten werden in dem dafür erarbeiten UjoSchema kodiert und über Message Streaming übertragen.

- Entwicklung von tLogView

  Kommandozeilenanwendung zur Visualisierung des Log- und Metrik Datenstroms für einfache Debugging- und Untersuchungszwecke. Einfache Filtermöglichkeiten nach Meldungstyp und Meldungsschwere.

- Erweiterung der Build Pipelines aller Unterprojekte

  Zur automatisierten statischen Code Analyse auf Basis der definierten Anforderungen für Code-Qualität.

- Entwicklung erster prototypischer Flows

  Entwicklung erster Softwarekomponenten durch eine wissenschaftliche Hilfskraft der CAU unter Nutzung der zur Verfügung gestellten Bibliotheken: LibUjo-C, UjoTypes-C und UjoTypes-py. Einholung von Feedback zur „User Experience“ bezüglich der Bibliotheken im Rahmen des Dezember Status Meetings. Die entstandenen Komponenten wurden später in Bricks der Flow Engine überführt werden.

- Weiterentwicklung des prototypischen Control Centers mit den Anwendungspartnern

  Der Prototyp des Control Centers aus der Masterarbeit wurde weiterentwickelt und in Bezug auf Skalierbarkeit evaluiert. Des Weiteren wird die jeweils aktuelle Version des Control Centers beim assoziierten Partner IBAK eingesetzt, wo der Stromverbrauch von anfänglich 16, mittlerweile 28 Servern überwacht wird. Die Erkennung von Verbrauchschwankungen im Laufe des Tages wird von IBAK als hilfreich erachtet. Die Softwarearchitektur des Control Centers mit den Resultaten aus Skalierbarkeitsevaluation und den Ergebnissen des Einsatzes bei IBAK wurden als Konferenzbeitrag auf der IEEE International Conference on Fog Computing 2019 präsentiert [HHM19].

- Vorarbeiten in Form von Experimenten

  bezüglich der Umsetzung der Komponenten „Control Peer“ und „Brick Runner“ der titan Software Plattform. Ziel ist die Definition des Kommunikationsprotokolls zwischen den Baustein, sowie die Erkundung dazu geeigneter Übertragungstechniken.

Das bereits implementierte Datenformat Ujo wurde gegen bestehende Lösungen wie zum Beispiel Protocol Buffer von Google weiter abgegrenzt. Als einer der Hauptunterschiede wurde die Möglichkeit identifiziert, variante Datentypen zu verwenden und je nach Anwendung entweder stark und schwach einzuschränken.

Die Implementierung der Flow-Engine, also den Komponenten, die den Flow ausführen, stellte das Team vor Herausforderungen. In gemeinsamen Workshops wurde das zugrunde liegende Probleme weiter verfeinert, so dass anschließend eine machbare Aufgabe vorlag, die zeitnah umgesetzt wurde. Eine bereits vorbereitete Integration des Flows für die Übernahme von Energieverbrauchsdaten in das titan Control Center konnte so frühzeitig umgesetzt werden.

Zusammengefasst sind wir auf erhebliche Herausforderungen auf allen Ebenen gestoßen, die die Projektpartner jedoch strukturiert in neue Erkenntnisse umgewandelt und bereits als Veröffentlichung vorbereitet haben.

1.4.3 Angabe der verwendeten Fachliteratur sowie der benutzten Informations- und Dokumentationsdienste

Kim, Gene ; Humble, Jez ; Debois, Patrick ; Willis, John: The DevOps Handbook : How to Create World-Class Agility, Reliability, and Security in Technology Organizations. United States: IT Revolution, 2016. -ISBN 978-1-942-78807-2. S. 1-480

Morrison, John Paul: Flow-based Programming : A New Approach to Application Development. New York: Van Nostrand Reinhold, 1994. -ISBN 978-0-442-01771-2. S. 1-316

Humble, Jez ; Molesky, Joanne ; O'Reilly, Barry: Lean Enterprise : Mit agilen Methoden zum innovativen Unternehmen. Sebastopol: O'Reilly, 2017. -ISBN 978-3-960-10078-2. S. 1-330

Humble, Jez ; Farley, David: Continuous Delivery : Reliable Software Releases Through Build, Test, and Deployment Automation. Amsterdam: Addison-Wesley, 2010. -ISBN 978-0-321-60191-9. S. 1-463

Skelton, Matthew ; Pais, Manuel: Team Topologies : Organizing Business and Technology Teams for Fast Flow. United States: IT Revolution, 2019. -ISBN 978-1-942-78881-2. S. 1-240

Reussner, Ralf; Hasselbring, Wilhelm: Handbuch der Software-Architektur. Köln: Dpunkt-Verlag, 2009. -ISBN 978-3-898-64559-1. S. 1-555

LeBlanc, David ; Howard, Michael: Writing Secure Code. Amsterdam: Pearson Education, 2002. -ISBN 978-0-735-63740-5. S. 1-800

Bastos, Joel ; Araújo, Pedro: Hands-On Infrastructure Monitoring with Prometheus : Implement and Scale Queries, Dashboards, and Alerting Across Machines and Containers. Birmingham: Packt Publishing, Limited, 2019. -ISBN 978-1-789-61234-9. S. 1-430

Kruse, Peter: Next practice : erfolgreiches Management von Instabilität ; Veränderung durch Vernetzung. Offenbach: GABAL, 2004. -ISBN 978-3-897-49439-8. S. 1-220

Fowler, Martin: Refactoring : wie Sie das Design vorhandener Software verbessern. Amsterdam: Addison-Wesley, 2005. -ISBN 978-3-827-32278-4. S. 1-440

Goucher, Adam ; Riley, Tim: Beautiful Testing : Leading Professionals Reveal How They Improve Software. Sebastopol: O'Reilly Media, 2009. -ISBN 978-0-596-15981-8. S. 1-352

Adzic, Gojko: Specification by Example : How Successful Teams Deliver the Right Software. Birmingham: Manning Publications, 2011. -ISBN 978-1-617-29008-4. S. 1-296

McConnell, Steve: Code Complete. München: Microsoft Press, 2004. -ISBN 978-0-735-61967-8. S. 1-914

Martin, Robert C.: Clean Code : Refactoring, Patterns, Testen und Techniken für sauberen Code ; [Kommentare, Formatierung, Strukturierung ; Fehler-Handling und Unit-Tests ; zahlreiche Fallstudien, Best Practices, Heuristiken und Code Smells]. Bonn: MITP, 2009. -ISBN 978-3-826-65548-7. S. 1-475

Turnbull, James: The Art of Monitoring. : James Turnbull, 2014. -ISBN 978-0-988-82024-1. S. 1-750

Martin, Robert C.: Clean Architecture : A Craftsman's Guide to Software Structure and Design. London: Prentice Hall, 2018. -ISBN 978-0-134-49416-6. S. 1-404

Humble, Jez ; O'Reilly, Barry ; Molesky, Joanne: Lean Enterprise : How High Performance Organizations Innovate at Scale. Sebastopol: O'Reilly Media, 2015. -ISBN 978-1-449-36842-5. S. 1-317

Pariès, Jean ; Wreathall, John: Resilience Engineering in Practice : A Guidebook. Boca Raton, Fla: CRC Press, 2017. -ISBN 978-1-317-06525-8. S. 1-362

Hubbard, Douglas W.: How to Measure Anything : Finding the Value of Intangibles in Business. New York: John Wiley & Sons, 2014. -ISBN 978-1-118-53927-9. S. 1-432

Tarlinder, Alexander: Developer Testing : Building Quality Into Software. Amsterdam: Addison-Wesley, 2016. -ISBN 978-0-134-29106-2. S. 1-313

Patton, Jeff ; Economy, Peter: User Story Mapping - Die Technik für besseres Nutzerverständnis in der agilen Produktentwicklung. Sebastopol: O'Reilly, 2015. -ISBN 978-3-958-75067-8. S. 1-302

Wohinz, Josef W. ; Moor, Michael: Betriebliches Energiemanagement : Aktuelle Investition in die Zukunft. Wien u.a.: Springer Vienna, 2012. -ISBN 978-3-709-19039-5. S. 1-315

Senge, Peter M.: The Fifth Discipline : The Art and Practice of the Learning Organization. : Doubleday/Currency, 1990. -ISBN 978-0-385-26094-7. S. 1-424

1.5 Zusammenarbeit mit anderen Stellen.

Wir erachten es als sehr wichtig, dass alle am Projekt beteiligten Partner stets eingebunden sind in die Aktivitäten. Dafür ist es notwendig, dass wir Arbeitsergebnisse regelmäßig präsentieren und diskutieren. Dies geschieht sowohl im Rahmen der Status-Meetings mit allen Beteiligten als auch in kleineren Gruppen zwischen wobe und CAU oder auch unter Einbeziehung eines der assoziierten Partner.

Besonders die Darstellung der technischen Herausforderungen und Ergebnisse in einer für andere Berufsgruppen verständlichen Sprache liegen uns dabei am Herzen. Die Rezeption und Verständlichkeit sind ein Gradmesser dafür, ob wir uns auf dem richtigen Weg befinden. Schließlich ist es das Ziel, mit der titan-Plattform eine Möglichkeit zu schaffen, betriebliche Anforderungen möglichst barrierefrei in Software-Prozesse zu modellieren.

Für ein besseres gegenseitiges Verständnis haben wir die Idee der Projektbeteiligten aus der Retrospektive im ersten Projektzeitraum aufgegriffen und versuchten, regelmäßig Partnerbetriebe und mögliche Partnerbetriebe im Rahmen der Status-Meetings zu besuchen und zu besichtigen.

2 Eingehende Darstellung

In jedem Projekt ist es wichtig, ein gemeinsames Verständnis von den Zielen und der Sprache des Projektes zu schaffen. Nur so kann gewährleistet werden, gemeinsam in die geplante Richtung zu arbeiten und auch die Anforderungen aller Beteiligten korrekt verstanden zu haben. Doch genau das fehlende Verständnis für die Projektziele, eine nicht einheitliche Sprache und die mangelhafte Erhebung sowie eine falsche Dokumentation der Anforderungen führen häufig dazu, dass Projekte in Schieflage geraten oder gar scheitern.

Um dieses möglichst zu vermeiden, haben wir uns im titan Projekt sehr intensiv mit der Thematik des gemeinsamen Verständnisses und den Anforderungen auseinandergesetzt. Folgende Schritte wurden von uns eingeleitet:

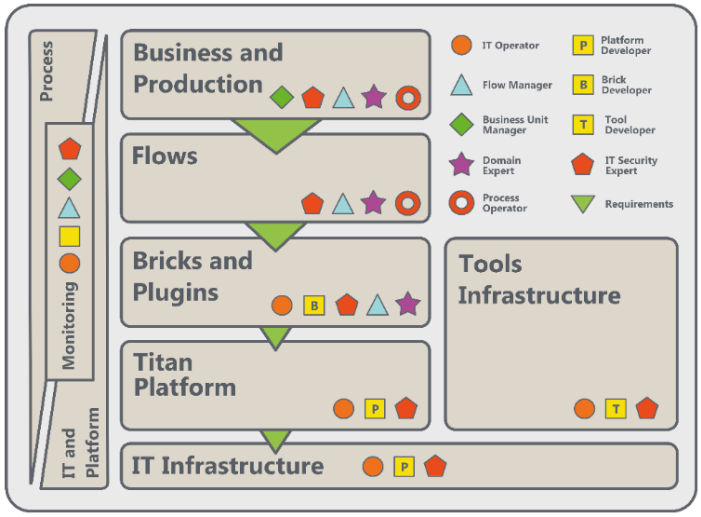

Abbildung 13: Das Industrial DevOps Rollenmodell

- *Erstellung eines Industrial DevOps-Rollenmodels zum besseren Verständnis der Ziele des titan-Projektes*

  Dieses Modell ist mit den Projektpartnern diskutiert und weiterentwickelt worden. Das Rollenmodell definiert die verschiedenen Rollen innerhalb eines Industriebetriebes. Es zeigt auch, wie die titan-Plattform dort einzuordnen ist und welchen Einfluss die definierten Rollen auf mögliche Flows im Betrieb nehmen können.

- *Definition von projektbezogenen Begriffen als geteiltes Verständnis*

  Um eine bessere Kommunikation zu gewährleisten, wurden für das Projekt relevante Begriffe diskutiert und definiert. Die Begriffe und Definitionen sind dokumentiert. Dieser Prozess wird fortlaufend weitergeführt.

- *Aufnahme und Diskussion der Anforderungen mit den assoziierten Projektpartnern*

  Um ein geteiltes Verständnis über die Anforderungen der assoziierten Partner zu bekommen, wurde mit jedem assoziierten Partner ein individuelles Meeting durchgeführt. In diesen Meetings sind deren Anforderungen und Zielsetzung aufgenommen worden.

Einige Beispiele für die Anforderungen sind:

- Anbindung von Stromverteilerleisten, die Informationen über den Verbrauch liefern
- Verwaltung von Verbrauchern
- Veränderungen der Infrastruktur z.B. Verbraucher wechselt den Stromanschluss
- Abfrage von bereits gesammelten Verbrauchsdaten aus einer SQL-Datenbank
- Auswertung von historischen Daten für ISO50001 Audits – Nachweis der Wirksamkeit von Maßnahmen zur Energieeinsparung
- Überprüfung von Maßnahmen zur Reduzierung von Spitzenlast
- Sammeln von Sensordaten verteilter Systeme (weltweit)
- Auswertung der gesammelten Daten für die Planung von Wartungseinätzen
- Verwaltung von titan System-Komponenten mit eingeschränkter Netzanbindung

- *Modellierung und Erstellung von ersten Bricks zum besseren Verständnis*
  Zur Visualisierung des Flows sind erste Bricks erstellt und innerhalb des Teams diskutiert worden. Diese werden fortlaufend weiterentwickelt.

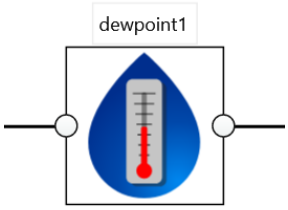

Abbildung 15: "Wetterdaten" Brick

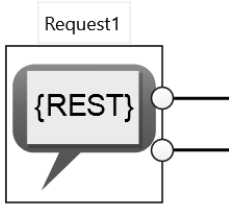

Abbildung 14: "REST-Request" Brick

Übersicht über die Tätigkeiten im Rahmen der Flow-Sprache und des Flow-Konzeptes:

- Grundlegende Elemente wurden beschrieben
- Ein grafisches Design wurde erarbeitet
- Die Flow-Sprache wurde anhand von Beispielen konzeptionell evaluiert
- Die Flow-Sprache abstrahiert Funktionalität und Daten, damit der Anwender beide für die Systemintegration erforderlichen Elemente unabhängig von der vorhandenen Infrastruktur modellieren kann.
- Für die Modellierung von Daten wurde eine Beschreibungssprache (UJOschemes) definiert.

- *Erstellung einer ausführlichen Dokumentation der Arbeitsergebnisse über git*

  Um eine gemeinsame Plattform zum Austausch aller relevanten Dokumente zu haben, haben wir ein git-Projekt erstellt. Folgende Themen werden hier bislang adressiert:

  - Dokumentation der Arbeitsschritte und technischen Entwürfe
  - Erstellte Dokumente wie Rollenmodell, Bricks, Fotos etc.
  - Related Projects wie UjoTypes und UjoSchema
  - Guidelines wie Coding Style Guide etc.

Während der gemeinsamen Planung der Anforderungen haben Prof. Hasselbring und sein Team von der CAU die eigenen Forschungsergebnisse, die für das Projekt relevant sind, vorgestellt. Hier ist bereits eine umfangreiche Arbeit geleistet worden, die verschiedene Aspekte des Projekts titan betrifft. Insbesondere war uns das Projekt TeeTime (https://teetime-framework.github.io/) zu Beginn des Projekts noch nicht bekannt. Die in diesem Projekt durchgeführten Arbeiten halfen, schneller bei der Arbeit am Flow voranzukommen.

Für das titan Projekt wurde eine eigene Terminologie entwickelt, mit der die Objekte in einem Flow beschrieben werden. Die Terminologie wurde gemeinsam mit den Partnern erörtert und beschlossen. Dieser Schritt war nötig, da eine einheitliche Terminologie für die gemeinsame Arbeit unerlässlich ist und weil bereits existierende Terminologien aus verschiedenen Gründen ungeeignet waren. Ein Grund war zum Beispiel, Flow Elemente für Stakeholder ohne umfangreiches IT Wissen verständlich zu machen.

Ein weiteres wichtiges Ergebnis der Arbeit an den Anforderungen war, die Signifikanz der Daten im System zu erkennen. Daten werden vom Flow behandelt, während sie durch das System laufen. Daten müssen temporär oder permanent gespeichert und manipuliert werden. Ein Anwender des titan Systems muss also nicht nur den Flow verstehen, sondern er muss auch in der Lage sein, Datenstrukturen zu erstellen und zu pflegen, ohne Probleme im Ablauf zu verursachen. Vorhandene Datenbeschreibungssprachen konnten den Anforderungen an Einfachheit bei hoher Flexibilität nicht gerecht werden. Es wurde daher beschlossen, eine eigene Sprache für die Beschreibung objekthierarchischer Daten zu entwickeln. Die Sprache wird im ersten Schritt genutzt, um interne Datenstrukturen des Systems zu beschreiben und in eine verständliche Dokumentation umzusetzen.

Die Architektur des Systems wurde basierend auf den Ideen aus der Projektbeschreibung verfeinert und erweitert. Die gefundenen Erkenntnisse führten zu einer strikten Trennung von Log-Infrastruktur und titan Laufzeitumgebung. Für die Log-Infrastruktur wurde ein Ebenen-Modell entworfen, dessen Ziel es ist, Daten aus verschiedenen Bereichen zu standardisieren. Ein modularer Aufbau des titan Log Backbones erlaubt es, Datenspeicherung und Auswertung nach Belieben zu implementieren. Der Austausch des Nachrichtentransports ist ebenfalls vorgesehen.

Eine Infrastruktur für Entwicklung, Codeverwaltung und Automatisierung von Test, statischen Analysen und Builds wurde eingerichtet. Richtlinien für Codequalität wurden erstellt und Werkzeuge für die Überprüfung der Richtlinien wurden evaluiert und integriert. Das Ziel, sich beim verwendeten Tool-Stack auf Open Source Werkzeuge zu beschränken, wurde erreicht.

Gemeinsam mit den Partnern wurde festgestellt, dass in realen Produktionsumgebungen ein einheitlicher Standard nicht erwartet werden kann. Dennoch wurde festgestellt, dass ein einheitlicher Standard von Daten und Schnittstellen die Integration erheblich vereinfachen würde. titan soll deshalb dem Anwender dabei unterstützen, einen internen Datenstandard zu definieren, um Abläufe zu modellieren, ohne die Spezifika der diversen Schnittstellen berücksichtigen zu müssen. Dieses Vorgehen wurde als Grundprinzip des titan-Systems vereinbart.

Ablauf des Vorhabens

Nach einer anfänglichen Orientierungsphase zu Beginn des Projekts, in der die Projektmitarbeiter die Arbeiten des Kerngeschäfts übergeben haben, ist das Projekt mit der geplanten Teamstärke begonnen worden. Während der Orientierungsphase sind somit weniger Kosten angefallen als geplant. In der zweiten Hälfte der Planungsphase standen dann alle eingeplanten Mitarbeiter voll zur Verfügung. Die gesetzten Ziele sind Ende der Planungsphase bis zum ersten Meilenstein weitgehend erreicht worden, so dass die Implementierung voll starten konnte. Einige kleine Entwicklungsprojekte wurden bereits während der Planungsphase gestartet, damit die technischen Vorrausetzungen für die Implementierung getestet werden konnte. Dieses Vorgehen hat in Phase 2 das Risiko eines Verzugs reduziert.

Innerhalb der Projektgruppe sind 14 Partner (siehe Abbildung 3) aus unterschiedlichen Industrien vertreten.

- Grafische Industrie
- Fertigende Industrie B2B
- Maritime Industrie
- Systemhaus aus dem Bereich Gebäudeautomatisierung
- Systemanbieter für Industrieautomatisierung

Die Projektpartner Christian-Albrechts-Universität zu Kiel (CAU) und wobe-systems GmbH (wobe) arbeiten eng zusammen. Die Zusammenarbeit geht weit über einen reinen Austausch von Informationen über den Stand des jeweiligen Arbeitsbereichs hinaus. Tatsächlich werden zum Beispiel gemeinsam Paper für Wissenschaftliche Konferenzen verfasst. Dabei profitieren beide Teams und lernen voneinander. Diese enge Zusammenarbeit sorgt für einen Fluss immer neuer Ideen und Betrachtungsweisen der jeweils aktuellen Herausforderungen, was für das Projekt ein großer Gewinn ist.

Obwohl die Implementierung der für die Systemintegration mit dem Control Center benötigten Komponenten zu Anfang noch nicht verwendbar war, hat das Team der CAU früh begonnen, die benötigten Bricks zu entwickeln. Während der Projektlaufzeit wurden diese Bricks stätig an den jeweiligen Entwicklungsstand der Flow-Engine angepasst. Auch dieses Vorgehen fördert einen ständigen Austausch zwischen den Teams und damit gemeinsames Lernen.

Eine große Herausforderung bei der Implementierung der Flow-Engine ist ihr experimenteller Charakter. Analogien zu bestehenden Technologien sind zwar vorhanden, lassen sich aber nicht direkt anwenden. Innovative Lösungen sind gefragt, die das Ziel verfolgen, eben diese Architektur wissenschaftlich zu untersuchen und mit den gegebenen Ressourcen Aufgabenbereiche anzugehen, die alles andere als trivial sind. Beispielsweise ist eine asynchrone Implementierung eines Queue-Managements beim Datentransport eines der Probleme, die im Projektverlauf bereits untersucht wurden, jedoch immer wieder diskutiert werden mussten.

Das Interesse der assoziierten Partner am Fortschritt der Entwicklung ist vorhanden, was sich besonders in der regen Beteiligung an Statusmeetings ausdrückt. Trotz der sehr technischen Themen kommt es dabei zu spannenden Diskussionen, von denen alle Beteiligten profitieren.

Der im Antrag für die Implementierung gewählte Clean Code Ansatz wurde während des Projektverlaufs intensiv untersucht. Die Ergebnisse waren teilweise überraschend. Besonders haben uns die Schwierigkeiten beschäftigt, Code auf Qualität zu prüfen. Es musste zunächst ein Weg gefunden werden, innere Qualität von Software zu definieren und ein gemeinsames Verständnis zu schaffen. Der zweite Schritt bestand darin, die Qualitätssicherung organisatorisch und technisch umzusetzen. Die im Projekt gefundenen Erkenntnisse stießen auch außerhalb des Projekts auf Interesse. Eines der eingereichten Papers für einen Workshop zum Thema langlebige Software beschäftigt sich damit [LHW19]. Es ist geplant, das Thema während des Projekts weiter zu vertiefen, da wir darin eine hohe Relevanz für Softwarelösungen für industrielle Anwendungen sehen.

Unser Ziel war es, im Verlauf des Projekts bereits vorhandene Teilprojekte, zum Beispiel unser binäres Datenformat als Open Source zu veröffentlichen und so eine allgemeine Diskussion zu erreichen. Themen wie die Erfahrungen mit Clean Code sollten auch auf Community Konferenzen diskutiert werden. Und natürlich ging es darum, die Flow-Engine so weit zu implementieren, dass sie mit dem Control Center integriert werden konnte, um Messungen für die ersten Flows zu liefern.

Erste Schritte, die Flow Engine und das Control Center zu integrieren, erforderten eine noch engere Zusammenarbeit der Teams der Christian-Albrechts-Universität zu Kiel (CAU) und der wobe-systems GmbH (wobe). Es wurde wertvolles Feedback ausgetauscht, das maßgeblich dazu beigetragen hat, gewählte Ansätze immer wieder zu überdenken und die resultierenden Lösungen zu verbessern.

Die guten Fortschritte, die das Team der CAU mit dem Control Center gemacht hat, ermöglichten es, bei der Integration und dem Test des Flow Engine (FE) Prototypen zu unterstützen. So war es dem Team von wobe möglich, die Entscheidung umzusetzen, den ersten Prototypen der FE zu verwerfen und neu zu implementieren. Der neue Ansatz ermöglichte eine erhebliche Steigerung der Ausführungsgeschwindigkeit der Flows und ist Ausgangspunkt für eine spätere Skalierung besser geeignet. Die Entscheidung, die Arbeit von mehreren Wochen zu verwerfen und neu anzufangen, zeigt den experimentellen Charakter des Projekts. Dennoch sind solche Entscheidungen für die Teams und die Projektleitung nicht einfach. Dass dieser Schritt möglich war und mit vollem Einsatz aller Beteiligten getragen wurde, zeigt aus Sicht der Projektleitung die Qualität und die hohe Motivation der Teams.

Es hatte sich herausgestellt, dass es während der Implementierung zunehmend schwierig wurde, neue und für die assoziierten Partner interessante Inhalte zu präsentieren. Gemeinsam wurde jedoch entschieden, dass es wichtig ist, die Partner mit einzubeziehen, weil sie wertvolle Beiträge zum Projekt liefern. Anwender aus der Praxis helfen, den Fokus auf der praktischen Anwendbarkeit der Ergebnisse zu behalten. Es wurde entschieden, Statusmeetings nicht hauptsächlich für die Präsentation von Ergebnissen zu nutzen, sondern in Form von Workshops weitere Inhalte zu erarbeiten oder zu überprüfen.
Das Interesse an den im titan Projekt behandelten Themen ist hoch. Besonders Clean Code und Softwarequalität sind als Anknüpfungspunkt geeignet. Das Zeigen die Gründung der Gruppe Softwerkskammer Region Kiel, die Gastvorlesung von Björn Latte an der CAU und der Erfolg des Papers “Clean Code: On the Use of Practices and Tools to Produce Maintainable Code for Long-Living Software” [LHW19].

Es war für das Projekt aus unserer Sicht wichtig, den Kontakt zu unseren Partnern und zu potenziellen Anwendern zu pflegen. Wir haben wichtige Impulse und weiteres wertvolles Feedback bekommen, wenn konkrete Beispiele umgesetzt wurden.

Mit Erreichen des Meilensteins M2 ging es wie geplant in die Evaluierungsphase. Die Arbeitsergebnisse der Partner wurden integriert und in Beispielanwendungen getestet. Bei der Entwicklung die Prototypen der Flow Engine hatten sich bei der Arbeit Herausforderungen ergeben, die die Entwicklung gebremst haben.

Insbesondere erforderte die Integration des Frontends mit dem Backend der Flow Engine noch weiteren Entwicklungsaufwand. Dies spiegelte sich jedoch in der ursprünglichen Planung wider und stand dem Beginn der Integrationsphase nicht entgegen.

Neben den organisatorischen Herausforderungen gab es auch technische Herausforderungen. Die neue Implementierung der Flow Engine und immer wieder auftretende Schwierigkeiten des Teams, Qualitätsstandards konsequent einzuhalten, erfordern Zeit und machen Nacharbeiten erforderlich. Diese Herausforderungen sind jedoch typisch für ein Entwicklungsteam und wurden bis zu einem gewissen Grad bei der Planung des Projekts berücksichtigt. Um das Thema Qualität besser in den Griff zu bekommen, hat das Team entschieden, von Kanban auf Scrum umzustellen.

Während des Berichtzeitraums zeigte sich, dass Daten im titan Projekt eine entscheidende

Rolle spielen. Zum einen ist das Thema Daten sehr gut geeignet, die Möglichkeiten der Plattform zu erläutern. Das zeigte sich bei einem Vortrag in der Starterkitchen in Kiel https://yimeo.com/374102719/f36034a3b9).

Zum anderen stellen besonders kleinteilige Daten und damit verbundene unterschiedliche oder nicht vorhandene Standards im Einsatzbereich der Plattform eine große Herausforderung dar. Diese Erkenntnis hat dazu geführt, die Datenbeschreibungssprache UjoSchema zu vereinfachen und zu überarbeiten.

Abbildung 16: Big Data basierend auf [Shaqiri, Bledi. (2017). Exploring Techniques of Improving Security and Privacy in Big Data. 10.13140/RG.2.2.23201.10089.j

Aus Sicht des Projektmanagements ist der Austausch mit den assoziierten Partnern, der Austausch mit Messebesuchern und die Aktivitäten auf Netzwerkveranstaltungen wichtig für das Projekt. Das Feedback führt zu immer neuen Sichtweisen und Ideen, die es ermöglichen titan besser zu erklären und einfacher anwendbar zu machen.

Die Ergebnisse aus dem Bereich Control Center wurden auf wissenschaftlichen Konferenzen auch international vorgestellt und brachten positives Feedback und spannende Diskussion ein. Hier ist besonders zu erwähnen, dass die skalierbare Architektur und die damit gezeigte schnelle Verarbeitungsgeschwindigkeit großer Datenmengen aktuell auf großes Interesse stößt.

Wie bereits angemerkt, konnte das geplante Personal nicht sofort eingesetzt werden, woraus sich eine Verzögerung beim Projektstart ergab. Während des Projektverlaufs mussten wir das Team personell verändern, was zu Verzögerungen durch die Einarbeitung einer neuen Kollegin führte. Der fristgerechte Start des Meilensteins M2 und der Integration der Gesamtlösung zeigte, dass die Gesamtlösung noch nicht zufriedenstellend evaluiert werden kann. Insbesondere die Akzeptanz der Lösung durch geschultes Fachpersonal im praktischen Einsatz ist hier betroffen. Dieser Punkt war aber aus unserer Sicht für den Erfolg des Projekts wichtig.

Da der Verbundpartner Christian-Albrechts-Universität zu Kiel (CAU) Personal erst verspätet einsetzen konnte, wurden von beiden Verbundpartnern jeweils ein Antrag auf kostenneutrale Verlängerung der Projektlaufzeit gestellt. Die gewonnene Zeit wurde genutzt, die Implementierung des Systems voranzutreiben. Integration und Evaluierung haben sich verschoben. Die übrige Projektplanung blieb jedoch unberührt.

Die Projektlaufzeit war geprägt vom Corona-Lockdown. Das Team musste sich umstellen auf Home-Office und verteiltes Arbeiten. Aus Sicht des Projektmanagements, hat das sehr gut funktioniert. Das Team wurde bereits vor Corona auf mobiles Arbeiten vorbereitet. Sowohl die Ausstattung als auch die Übung online zu kommunizieren half schnell wieder zum Tagesgeschäft übergehen zu können.

Da die Arbeit an der grafischen Oberfläche hinter dem Zeitplan lag, wurde das Team zu Beginn des Jahres 2020 vergrößert. Zwei zusätzliche Entwickler und zwei Studenten mussten in die Arbeit am System integriert werden. Damit bestand das Team aus insgesamt 7 Personen. Da der Fortschritt auf sich warten ließ, kam aus dem Team der Vorschlag die agile Methode Scrum statt Kanban einzusetzen. Wesentlicher Bestandteil von Scrum sind kurze Sprints, in denen ein gemeinsam verabredetes Ziel erreicht wird. Was dazu nötig ist, um dieses Ziel zu erreichen, organisiert das Team selbst. Das Projektmanagement zieht sich auf die Rolle des Product Owners zurück und arbeitet gemeinsam mit dem Team an der Festlegung der Ziele.

Durch die Einführung von Scrum in der letzten Projektphase konnte das Team eine gleichbleibende Geschwindigkeit bei der Entwicklung neuer Funktionen etablieren. Das Wissen über Entwicklung (Dev) und IT-Betrieb (Ops) war im Team vorhanden. Es wurde dennoch beobachtet, dass die Aufgaben des IT-Betriebs während des Sprints zweitrangig behandelt wurden. In Retrospektiven verstärkte sich der Druck aus dem Team nach einer klaren Rollenverteilung.

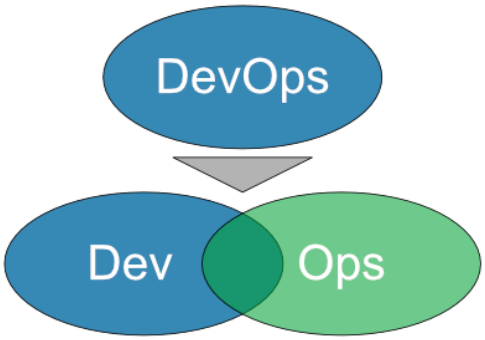

Abbildung 17: Team-Struktur für DevOps

Als Konsequenz wurde das Team in Entwicklung und IT-Betrieb geteilt. Dabei wurde berücksichtigt, dass beide Teams eng zusammenarbeiten und gegenseitig unterstützen. Dafür war es nötig Wissen über Entwicklung im IT-Betriebsteam zu haben, und umgekehrt.

Als Ergebnis der Umstrukturierung wurden beide Themenbereiche mit gleicher Priorität bearbeitet. Gleichzeitig sank die mentale Belastung in den Teams.

Als Ergebnis arbeitete das Team deutlich fokussierter an der Implementierung. Während eines Sprints wurden Diskussionen auf technische Details beschränkt, während Entscheidungen über Architektur und/oder der Einsatz von neuen Tools und Techniken basierend auf den im Sprint gelernten Lektionen reflektiert und zwischen den Sprints diskutiert wurden. Die grafische Oberfläche machte schnell Fortschritte und entwickelte sich zum Kern weiterführender Diskussionen. Es war mit den Ergebnissen vor Augen wesentliche leichter die nächsten Schritte aus Sicht eines möglichen Anwenders zu betrachten. Damit sich das Team voll auf die Entwicklung konzentrieren konnte, wurde entschieden einen externen Scrum-Master für die effektive Durchführung von Meetings und als Ansprechpartner für das Team hinzuzuziehen.

Es hatte sich gezeigt, dass auch die letzte Phase im Projekt durch die Pandemie-Situation geprägt war. Anfangs wurde versäumt die kleinen Erfolge des Projekts ausreichend zu würdigen und zu feiern. Das haben wir zumindest teilweise nachgeholt und online gemeinsam die Erfolge des Teams gewürdigt. Nicht zuletzt der erfolgreiche Projektabschluss gab Anlass zur Freude. Gemeinsam mit dem Team der CAU werden wir diesen Erfolg nachträglich feiern, wenn die Umstände es wieder erlauben. Alle Teammitglieder arbeiteten motiviert und kompetent an Lösungen, um das gemeinsame Ziel zu erreichen. Diesen Schwung nehmen wir mit in die Phase der Verwertung der Projektergebnisse.

2.1 Verwendung der Zuwendung und des erzielten Ergebnisses im Einzelnen, mit Gegenüberstellung der vorgegebenen Ziele

Die folgenden Kapitel beschreiben die Ergebnisse der einzelnen Arbeitspakete und stellen sie den geplanten Ergebnissen der Antragstellung gegenüber. Die Arbeiten wurden gemeinschaftlich von den Projektpartner CAU und WOBE durchgeführt. Eine Partner-spezifische Listung der Beiträge zu den Ergebnissen und Tätigkeiten findet sich in der tabellarischen Kurzdarstellung der Arbeitspakete in Kapitel 1.3.

2.1.1 Vorarbeiten

Clean Code

Clean Code als Methode, die innere Qualität von Software zu steigern, wurde eingehend untersucht und diskutiert. Das überraschende Ergebnis war, dass die Clean Code Prinzipien bei der Herstellung von Software bislang in anderen Projekten kaum eine Rolle spielen. Die Projektpartner haben jedoch eine Reihe von Punkten gefunden, die Clean Code in Software motivieren sollten:

- Bessere Wartbarkeit
- Vermeidung von technischer Schuld
- Weniger Aufwand und daher kostengünstigere Änderungen und Anpassungen bei langlebiger Software (Cost of Change)
- Stabilere Systeme
- Reduzierung der Wahrscheinlichkeit beim Beheben von Fehlern neue Fehler zu machen
- Leichteres Onboarding neuer Teammitglieder
- Zufriedenere Anwender
- In Open-Source Software bessere Einbindung von Contributern

Im titan Projekt werden Clean Code Prinzipien kontinuierlich angewendet. Pfeiler für den erfolgreichen Einsatz sind zum Beispiel Weiterbildungen zum Thema Code Qualität, technische Maßnahmen wie statische Code Analyse, Style Guides und Code Reviews als Quality Gate.

Feedback zur Weiterentwicklung der Code Hosting und Build Plattform

Sämtliche Software-Teilprojekte, die im Rahmen des titan-Projektes entstehen und weiterentwickelt werden, werden in einer eigens eingerichteten Plattform (GitLab) verwaltet. Sie ermöglicht Versionsmanagement, die Koordination zwischen Entwicklern und das automatische Erzeugen von Artefakten. Die Plattform wird auf der Serverinfrastruktur von wobe ausgeführt. Änderungen und zusätzliche benötigte Funktionen ergeben sich fortlaufend, werden diskutiert und umgesetzt.

Folgende Themen werden hier adressiert:

- Software-Teilprojekte
- Dokumentation der Arbeitsschritte und technischen Entwürfe
- Erstellte Dokumente wie Rollenmodell, Bricks, Fotos etc.
- Related Projects wie UjoTypes und UjoSchema
- Guidelines wie Coding Style Guide
- Standards in Bezug auf Code Quality
- Automatisiertes Erzeugen der Build Artefakte und Packaging für das Deployment
- Verkettung von aufeinander aufbauenden Builds von Unterprojekten durch Abruf der Artefakte des jeweils vorgelagerten Projekts
- Nutzung der Pipeline als „Quality Gates“ zur Überprüfung und Sicherstellung der aufgestellten Guidelines zu Code Stil und Code Qualität

2.1.2 AP1: Anforderungsanalyse titan-Plattform

Geeignete Dokumentation aller Anforderungen

Den Kontext für die Anforderungen der Plattform liefert eine für alle Projektbeteiligten zugängliche Dokumentation im Web. Sie dient während des Projekts und danach als Referenz für grundlegende Prinzipien der Flow-Programmierung und Umsetzung der Plattform. Änderungen an der Dokumentation werden mittels Source-Code-Management (SCM) via Git nachverfolgbar. Aktualisierungen werden automatisch veröffentlicht, damit das Team sich immer auf den aktuellen Stand beziehen kann: https://doc.industrial-devops.org/titanDocumentation/.

Neben der allgemeinen Dokumentation der Anforderungen enthält die Plattform Dokumentation detaillierte Informationen zur technischen Umsetzung: https://doc.industrial-devops.org/titanPlatform/.

User-Stories für Text basierte Dokumentation

Für die effektive Verwaltung der Anforderungen im Team, wurde für das Projekt eigens eine Jira-Software Instanz verwendet: https://tracker.industrial-devops.org.

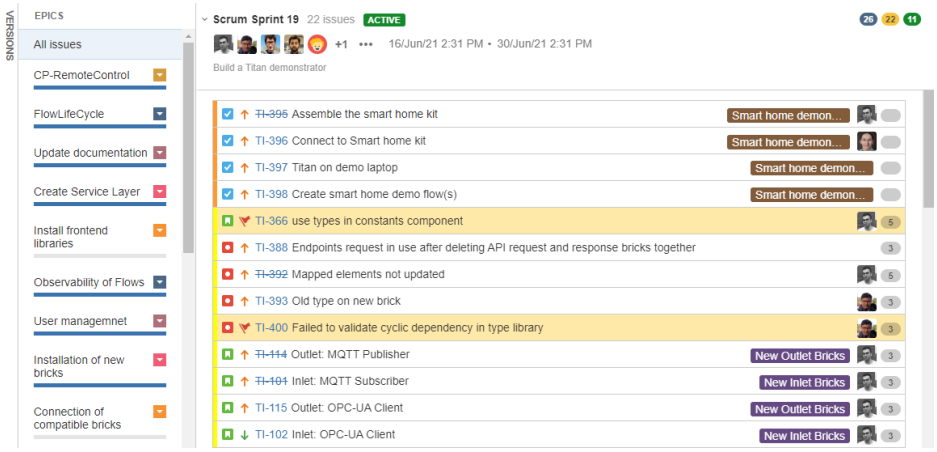

Abbildung 18: Sprint Backlog

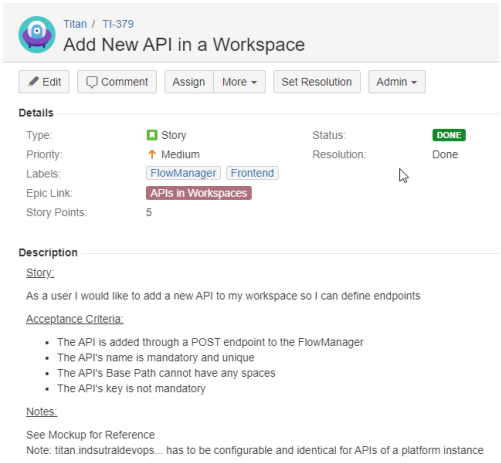

Abbildung 19: Example User Story

Grafische Darstellungen von UI Anforderungen

Für die Visualisierung von Anforderungen des titan-UI wurden Mock-Ups erstellt, die im Jira mit den Anforderungen verknüpft sind.

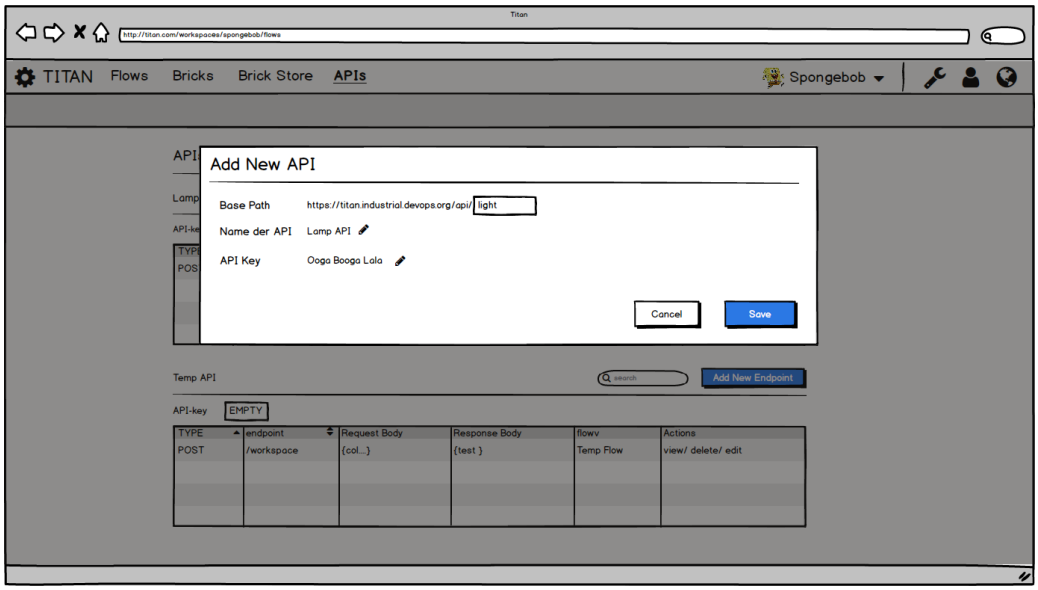

Abbildung 20: Beispiel Mock-Up titan-UI

Weitere grafische Darstellungen z.B. Use-Case Diagramme

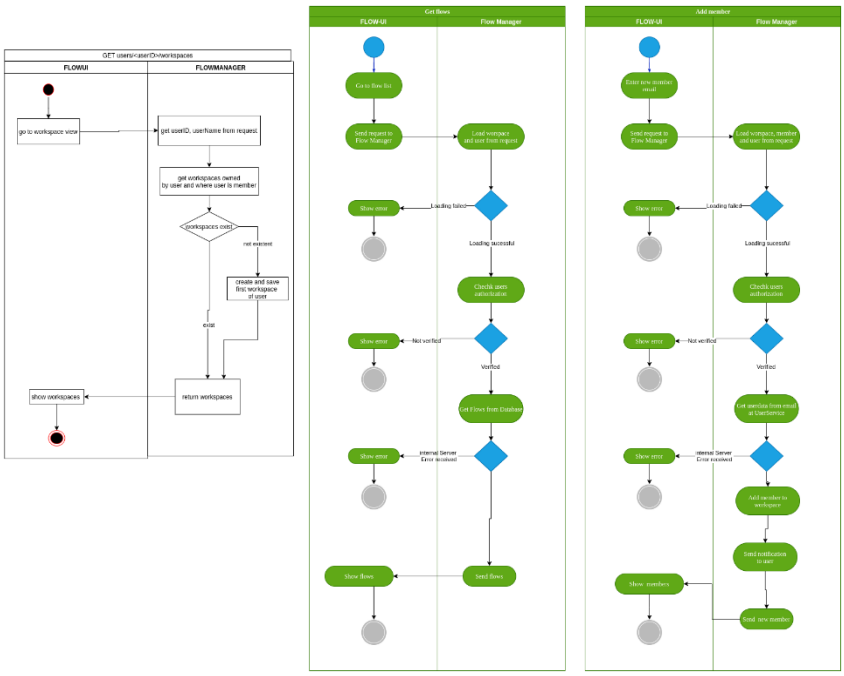

Abbildung 21: Beispiel für Backend-Prozess

Akzeptanzkriterien für Anforderungen festlegen

Wie in Abbildung 19 gezeigt wurden Anforderungen mit Akzeptanzkriterien versehen.

Dokumentation einer Stakeholder Analyse

Die Stakeholder Analyse wurde in Form eines Rollenmodells erstellt und mit den technischen Aspekten kombiniert. So entstand eine grafische Darstellung der Sozio-technischen Sichtweise, die für DevOps typisch ist.

Fehler! Verweisquelle konnte nicht gefunden werden. zeigt das Rollenmodell. Eine Beschreibung des Rollenmodells ist im Internet zu finden: https://doc.industrial-devops.org/titanDocumentation/industrial-devops/role-model/

Relative Abschätzung des Aufwands für die Implementierung der Anforderungen

Wie Abbildung 19 zeigt wurden Aufwandsabschätzungen an den jeweiligen Anforderungen mit Story-Punkten vermerkt und während der Bearbeitung im Sprint verwendet.

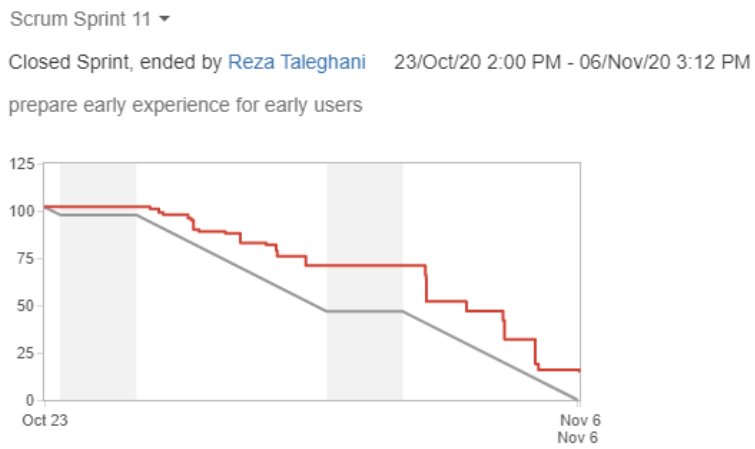

Abbildung 22: Burndown Chart Scrum Sprint

Priorisiertes Backlog für die Umsetzung der Anforderungen

Das Priorisierte Backlog zum Zeitpunkt der Berichterstellung enthält mehr als 600 Vorgänge.

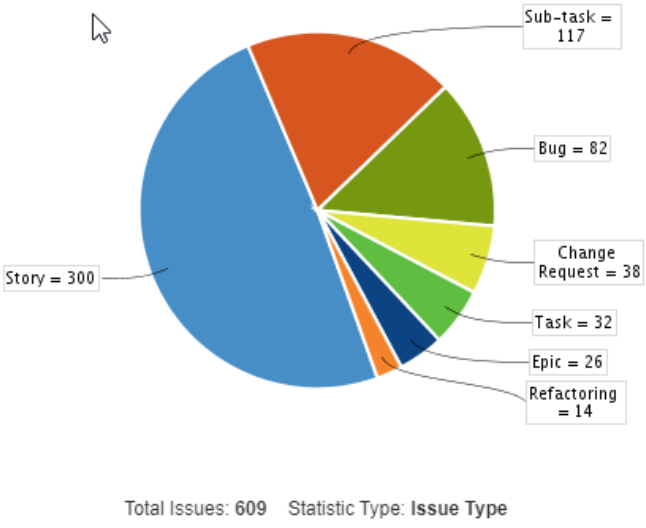

Abbildung 23: Verteilung der Vorgänge im Backlog

2.1.3 AP1-1: Nicht-funktionale Anforderungen der titan-Plattform

Performance und Skalierbarkeit

Performance und Skalierbarkeit sind kritische Faktoren der Plattform. Konkrete Anforderungen wurden in verschiedenen Bereichen eingebracht:

Die Skalierbarkeit der Flows wird durch parallele Ausführung von Bricks ermöglicht: https://doc.industrial-devops.org/titanDocumentation/titan-flow/concurrency/.

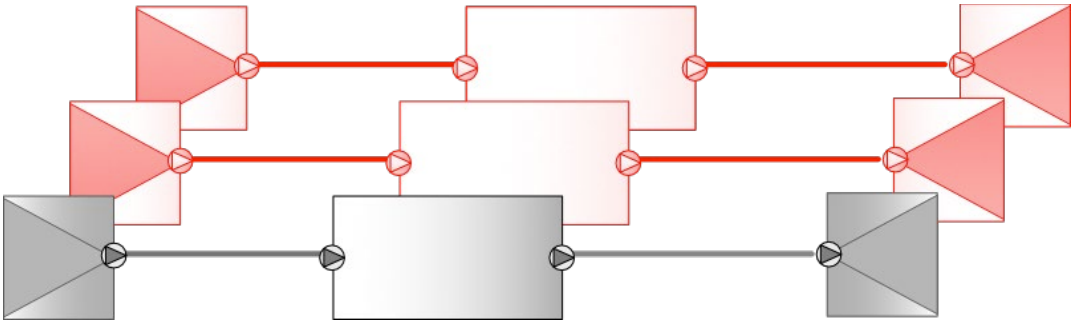

Abbildung 24: Flow Concurrency

Für die effiziente Verarbeitung von Datenpaketen wird ein eigenes, binäres Format verwendet: https://ujotypes-py.readthedocs.io/en/latest/.

Verfügbarkeit, Resilienz

Die Anwendungsfälle von titan als „Glue-Code“ zur Systemintegration implizieren die Anforderung einer hohen Verfügbarkeit, weil ein Ausfall bedeutet, dass automatisierte Prozesse nicht ausgeführt werden können. Bei stark integrierten Systemen kann das die Gesamtfunktionalität gefährden.

Sicherheit

Sicherheit spielt eine große Rolle. Das titan System ist verbunden mit vielen anderen Komponenten. Ein Angriffsvektor titan bietet die Möglichkeit größten Schaden zu verursachen.

Compliance

Zwei grundlegende Anforderungen wurden identifiziert und teilweise im Prototypen umgesetzt.

1. DSGVO konforme Verarbeitung von Personenbezogenen Daten
2. Nachvollziehbarkeit und Protokollierung von Änderungen

Einsetzbarkeit (welcher Aufwand ist nötig, um die Software zu betreiben?)

Modularisierung und Skalierbarkeit bringt eine Steigerung der Komplexität mit sich. Dennoch ist eine Anforderung den Aufwand für den Betrieb möglichst gering zu halten. Dabei ist vorgesehen, dass der Aufwand steigt, je größer das System ausgelegt wird. Kleine Anwendungen sollen jedoch einfach zu realisieren sein.

2.1.4 AP1-2: Funktionale Anforderungen titan-UI

Anforderungen im Projektteam ermitteln und dokumentieren

Anforderungen wurden im Laufe des Projekts verfeinert. Jira wurde für das Backlog und die Verwaltung der Anforderungen genutzt.

Erstellen von Mockups und Beschreibungen der Benutzerinteraktionen

Abbildung 20 zeigt ein Beispiel für ein Mockup. Die Mockups sind mit den Anforderungen im Backlog verknüpft.

Überprüfung der Anforderungen mir den assoziierten Projektpartnern

Den Assoziierten Partnern wurden die Anforderungen regelmäßig in gemeinsamen Statusmeetings vorgestellt und gemeinsam diskutiert. In Workshops wurden bestimmte Aspekte der Flow-Programmierung getestet (z.B. Papier-Prototyp)

Abbildung 25: Workshop mit Partnern

2.1.5 AP2: Anforderungsanalyse titan CC

Die Anforderungen für das titan Control Center wurden anhand des Anwendungsfall „Industrielles Energiedatenmanagement“ aufgenommen. Fachliche sowie technische Anforderungen und Rahmenbedingungen wurde hierzu in Workshops mit den assoziierten Partnern IBAK und KN Druckzentrum sowie im Rahmen der regulären Statusmeetings diskutiert. Besonderer Fokus lag auf dem Monitoring und der Analyse von Stromverbrauchsdaten.

Fachliche Anforderungen

Fachliche Anforderungen wurden organisiert in vier Ziele, die produzierende Unternehmen durch ein umfassenden Energiemanagement erreichen möchten. Die identifizierten Ziele sind:

- Reporting: Aufgezeichnete sowie analysierte Stromverbrauchsdaten sollen in Form von Dashboards und Berichten verschiedenen Stakeholdern zur Verfügung gestellt werden. Eine solche Berichterstattung ist beispielsweise für eine ISO 50001 Zertifizierung erforderlich, die sowohl IBAK als auch das KN Druckzentrum anstreben.
- Optimierung: Unternehmen wollen ihren Energieverbrauch sowohl aus ökologischen als auch aus ökonomischen Gründen reduzieren. Das beinhaltet unter anderem die Reduzierung Gesamtverbrauchs, die Reduzierung von Lastspitzen, die überproportional teuer sind (besonders relevant für IBAK und das KN Druckzentrum), sowie die Optimierung des Verbrauchs von einzelnen Maschinen oder Produktionsanlagen.
- Fehlererkennung: Ein abweichender Energieverbrauch kann ein Indikator für Fehler von Maschinen oder des Produktionsbetriebs sein. Ein Beispiel hierfür existiert im KN Druckzentrum, wo ein erhöhter Stromverbrauch der zentralen Druckluftverteilung häufig auf das Vorhandensein von Leckagen hindeutet
- Predictive Maintenance: Ein abweichender Energieverbrauch kann als Indikator für notwendige Wartung dienen. Typisches Beispiel hierfür sind Kühlkreisläufe, in denen ein erhöhter Verbrauch von Pumpen auf eine notwenige Reinigung von Filtern hindeuten kann. Im Falle des KN Druckzentrums ist es besonders wichtig, Produktionsfehler bzw. -stillstand zu vermeiden, da die Zeitungsproduktion sehr zeitkritisch ist.

Technische Anforderungen

Aufbauend auf den fachlichen Anforderungen in Form von Zielen, haben wir technische Maßnahmen in Form von acht „Maßnahmen“ organisiert. Diese Maßnahmen sind:

- Echtzeit-Monitoring und Auswertung: Auf Besonderheiten im Energieverbrauch soll schnell reagiert werden können, zum Beispiel vor dem Auftreten von Lastspitzen, bei der Erkennung von Fehlern oder notwenigen Wartungsarbeiten
- Monitoring und Analyse auf unterschiedlichen Aggregationsebenen: Einzelne Verbraucher sollen zu größeren Einheiten zusammengefasst werden, z.B. alle Netzteile eines Servers oder alle Maschinen eines Typs. Beim KN Druckzentrum ist es zudem notwendig, eine Organisation von Verbrauchern sowohl noch Maschinentypen als auch nach geografischem Standort vorzunehmen.
- Zeitliche Aggregation: Gespeicherte Daten sollen in verschiedenen Auflösungen archiviert werden können. Zusätzlich sind weitere Aggregationen für Visualisierungen notwendig.

- Korrelation (von Zeitreihen): Dies betrifft Zeitreihen von verschiedenen Verbrauchern, Verbrauchern und der Umwelt (z.B. Wetterdaten), Verbrauchern und Ereignisse (z.B. Wartungsarbeiten) und Verbrauchern und Produktionsdaten (z.B. Produktionsgeschwindigkeit).
- Erkennung von Anomalien: Dies ermöglicht beispielsweise die Erkennung von Fehlern in der Produktion. Besonders anspruchsvoll ist eine Anomalieerkennung, wenn, wie im Beispiel des KN Druckzentrums, der Verbrauch stark abhängig ist von Tages- und Wochenzeit.
- Treffen von Vorhersagen: Vorhersagen über den Energieverbrauch können bei der automatischen Wartung oder der Vermeidung von Lastspitzen helfen. Ähnliche Herausforderungen wie bei der Anomalieerkennung gelten auch hier.
- Visualisierung: Monitoringdaten und Analyseergebnissen sollen für verschiedene Stakeholder (z.B. Management, Produktionsleitung, Facharbeiter) bereitgestellt werden. Dabei benötigen unterschiedliche Stakeholder unterschiedlich aufgelöste Daten. Typische Arten von Visualisierungen sind farbliche Indikatoren für hohen/niedrigen Verbrauch, Anzeigen von Anomalien, interaktive Zeitreihen, etc.
- Benachrichtigungen: In Ergänzung zur Visualisierung sollen automatische Benachrichtigungen verschickt werden können, zum Beispiel beim Auftreten von Anomalien.

Im Rahmen der regulären Statusmeetings wurde mit den assoziierten Partnern eine Bewertung vorgenommen, welche Maßnahmen inwiefern welche Ziele unterstützen (siehe Abbildung 26: Ziele von industrieller Energiedatenanalyse sowie zugehörige Maßnahmen). Die Ergebnisse dieser Arbeiten im *Journal of Data, Information and Management* publiziert [HHB+21].

Technische Rahmenbedingungen

Von IBAK wurde Zugriff auf Stromverbrauchsdaten der Server gegeben. Diese sind über Stromversorgungsleisten der Firma *Raritan* angebunden und stellen Daten über die Protokolle SNMP (poll) sowie HTTP/JSON (push) bereit, wobei sich für letzteres favorisierte Technik entschieden wurde. Über mehrere Monate wurde eine dauerhafte Weiterleitung der IBAK-Messdaten an die Server der Uni Kiel zur kontinuierlichen Auswertung eingerichtet.

Vom KN Druckzentrum wurde Zugriff auf das bereits installierte Energieverbrauchs-Monitoringsystem gegeben. Mittels kontinuierlicher Abfrage einer Firebird-Datenbank wurde so eine einfache Integration in die bestehenden Systeme ermöglicht. Zur Evaluierung wurde wobe und der CAU ein Datenbank-Backup mit Verbrauchsdaten der letzten 4,5 Jahre sowie außerdem Produktionsdaten zur Verfügung gestellt.

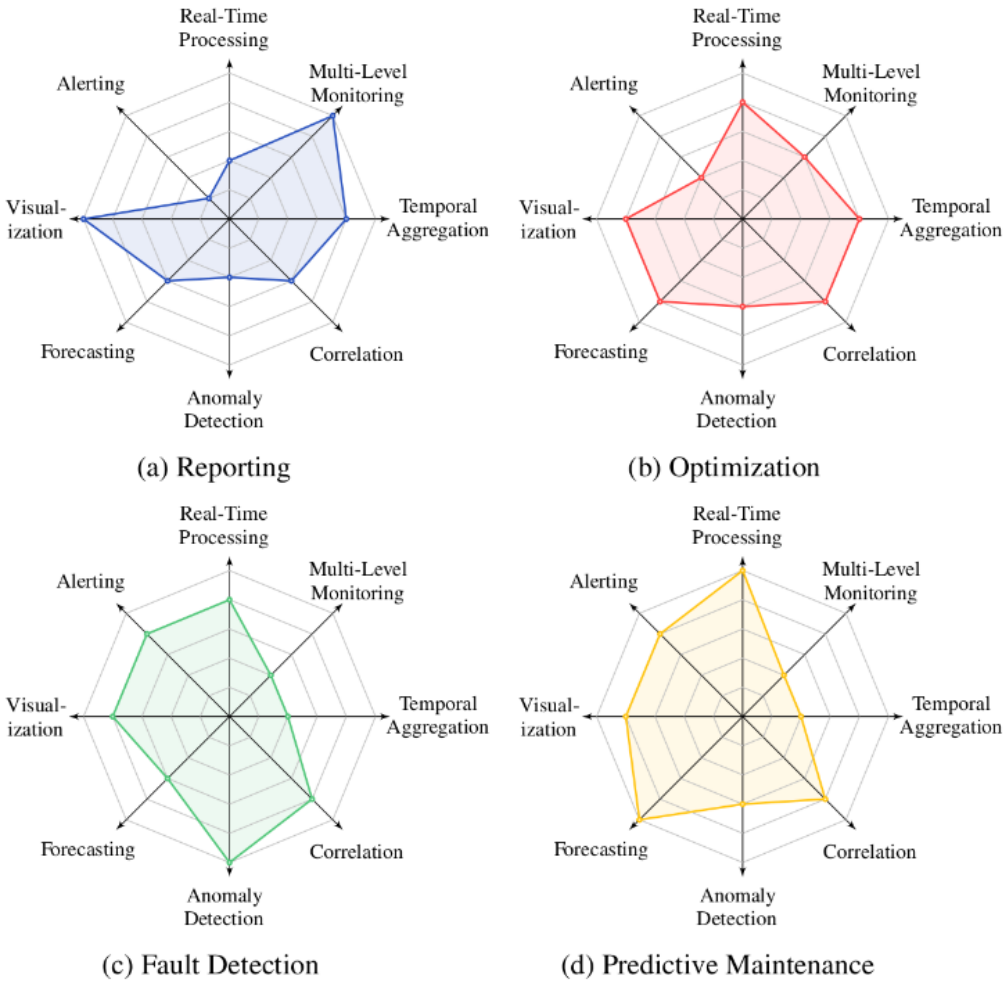

Abbildung 26: Ziele von industrieller Energiedatenanalyse sowie zugehörige Maßnahmen [HHB+21]

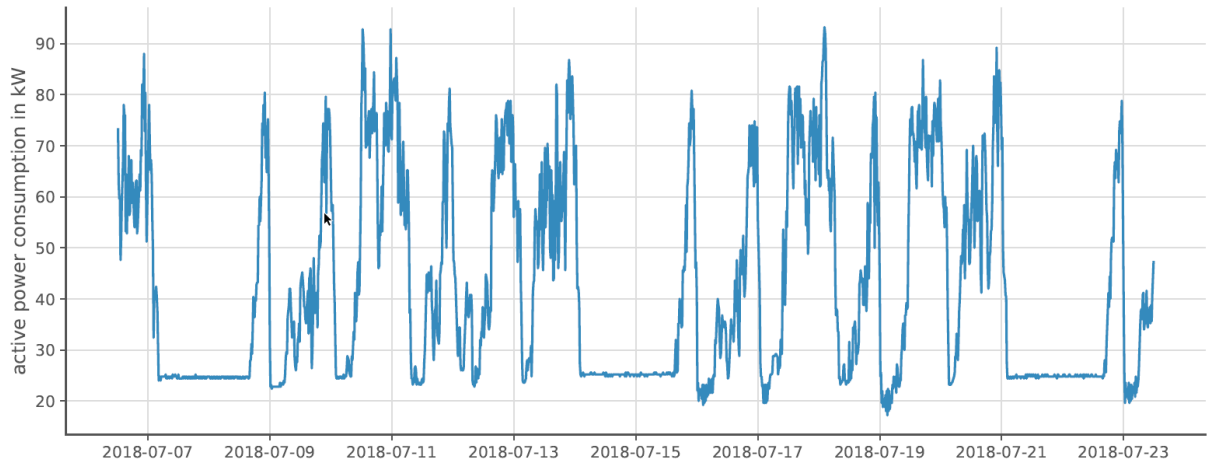

Abbildung 27: Stromverbrauch des KN Druckzentrums

2.1.6 AP3: Architektur

- Geeignete Dokumentation der Architektur

Die verschiedenen Systemkomponenten werden in den folgenden Abschnitten (AP3-1 bis AP3-6) beschrieben. Es wurde jeweils die Darstellung gewählt, die für kontinuierliche Arbeit am effektivsten war.

2.1.7 AP3-1: Architektur der titan-Laufzeitumgebung

Die titan-Laufzeitumgebung wird im folgenden auch Flow-Engine genannt und stellt den Systemteil dar, der für die effiziente Ausführung von Flows zuständig ist.

- Dokumentation der Prozesse, Module, Komponenten und Klassen der titan-Laufzeitumgebung (bevorzugt grafische Darstellungen)

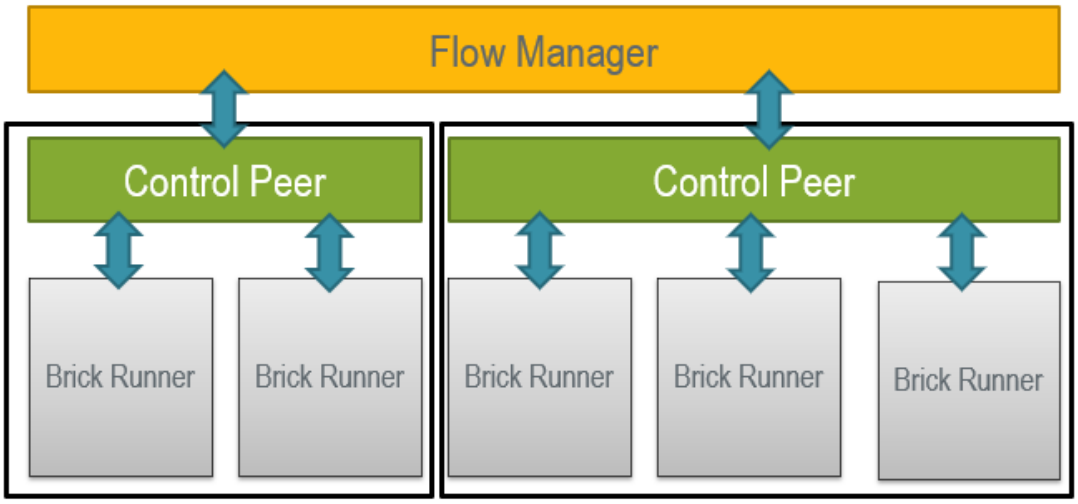

Abbildung 28: Komponenten der Flow Engine

Die Beschreibung der Kommunikation der Komponenten ist in der Dokumentation beschrieben: https://titanfe.readthedocs.io/en/latest/HowTos/communication-in-titanfe.html#.

- Beschreibung des Tool Stacks für die Umsetzung

Der Prototyp wird in Python umgesetzt.

- Systemarchitektur mit Datenbanken, verteilten Dateisystemen usw.

Die Flow-Engine ist eingebettet in das Gesamtsystem:

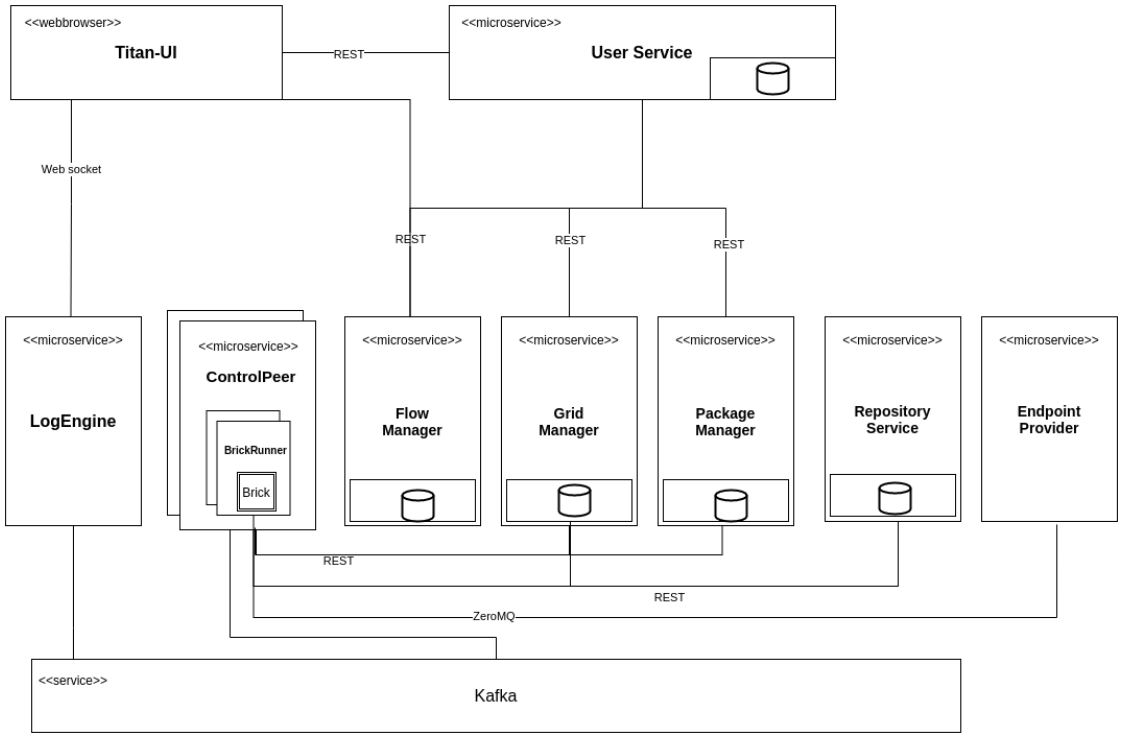

Abbildung 29: titan Architektur

- Beschreibung von geeigneten Metriken zur Überwachung der Laufzeitumgebung

Für die Überwachung werden zwei Blickwinkel benötigt:

1. Die Überwachung der Server/Container und der darunter liegenden Ausführungsplattform.

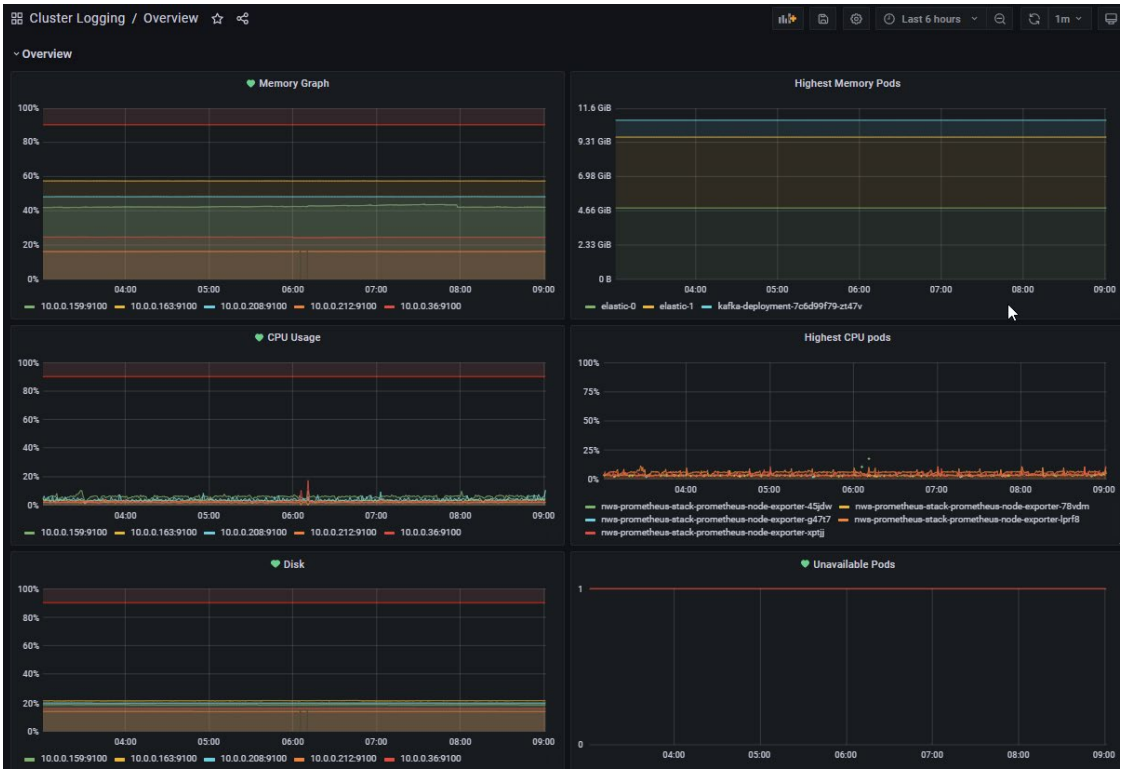

Abbildung 30: Beispiel Dashboard Umgebungsmetriken

Folgende Metriken werden betrachtet:

Speicher, CPU, Fehler, Netzwerk, Verfügbarkeit

2. Die Überwachung einzelner Flows.

Folgende Metriken werden betrachtet:
Paketlaufzeit, Ausführungszeit von Bricks, Zeit in Queue, Füllstand der Queues, Anzahl FPs (Flow-Packages).

2.1.8 AP3-2: Architektur der titan-Entwicklungs- und Test-Umgebung

- Ziele von DevOps
  - Continuous Integration (CI)
  - Continuous Deployment/Delivery (CD)
  - Fokus auf Qualität
- Tool Stack
  - Quellcode Verwaltung mit GitLab (https://about.gitlab.com)
  - Anforderungsmanagement mit Jira (https://www.atlassian.com/de/software/jira)
  - Automatische Qualitätsprüfung

Für die automatische Prüfung der Qualität werden verschiedene Tools eingesetzt, die im Einzelnen untersucht und beschrieben wurden [LHW19].

- Laufzeitumgebung (Environments) mit Docker (https://www.docker.com)

2.1.9 AP3-3: Architektur der Benutzerschnittstelle

Die Architektur des Gesamtsystems ist in Abbildung 29 dargestellt. Das UI kommuniziert via REST Schnittstellen mit den Diensten der Plattform.

Beschreibung der Schnittstellen und der Kommunikation mit dem titan-Kern

Die Beschreibung der Schnittstellen, die für die Kommunikation des UIs verwendet werden, sind in der Service-Dokumentation zu finden: https://doc.industrial-devops.org/titanPlatform/.

2.1.10 AP3-4: Installierbare Komponenten

Installierbare Komponenten sind eine Schlüsselfunktion der titan Plattform. Mit installierbaren Komponenten werden Domainspezifische Anpassungen möglich.

- Dokumentation der Datenstruktur installierbarer Komponenten
  https://doc.industrial-devops.org/titanDocumentation/titan-flow/how-to-write-Python-bricks/
- Dokumentation des Versionsmanagements von Komponenten
  Das Versionsmanagement der Bricks und Brick-Pakete wird mit GitLab realisiert.
- Dokumentation des Qualitätsmanagements von Komponenten
  Für das Qualitätsmanagement werden die gleichen Methoden, Tool und Prinzipien eingesetzt, die in 2.1.8 beschrieben sind.

2.1.11 AP3-5: Grafische Beschreibungssprache für Flows

Die grafische Beschreibungssprache für Flows soll die intuitive Umsetzung von Lösungen durch Fachkräfte mit geringen Programmierkenntnissen unterstützen.

- Dokumentation der Flow-Daten im System

Für die Beschreibung und Dokumentation von Flow-Daten wird die eigens dafür entwickelte Schema-Sprache UjoSchema verwendet. Die Sprache ist in folgenden Dokumenten beschrieben:

- https://git.industrial-devops.org/titan/related-projects/ujoschema-py/-/blob/master/doc/markdown/ujo-schema.md
- https://www.industrial-devops.org/datenstrukturen-dokumentieren/
- https://www.industrial-devops.org/grenzen-setzen/
- https://www.industrial-devops.org/daten-als-schluessel-zu-nocode/
- https://www.industrial-devops.org/typen-gibts/

- Daten-Mapping und Matching

Damit Bricks miteinander verbunden werden können, ohne bereits bei deren Erstellung die verwendeten Datenstrukturen zu kennen, müssen Daten im Flow auf die jeweiligen Schnittstellen der Bricks gemappt werden.

Ein Matching, also die automatische Zuweisung von Daten ist ebenfalls vorgesehen, kann aber nur funktionieren, wenn sie vom Anwender verifiziert und betätigt werden.

- Dokumentation der schematischen, grafischen Darstellung in einer allgemeinen Form
  - https://doc.industrial-devops.org/titanDocumentation/titan-flow/flow_elements/
  - https://doc.industrial-devops.org/titanDocumentation/titan-flow/fundamental_brick_types/

2.1.12 AP3-6: Control Center Systemarchitektur

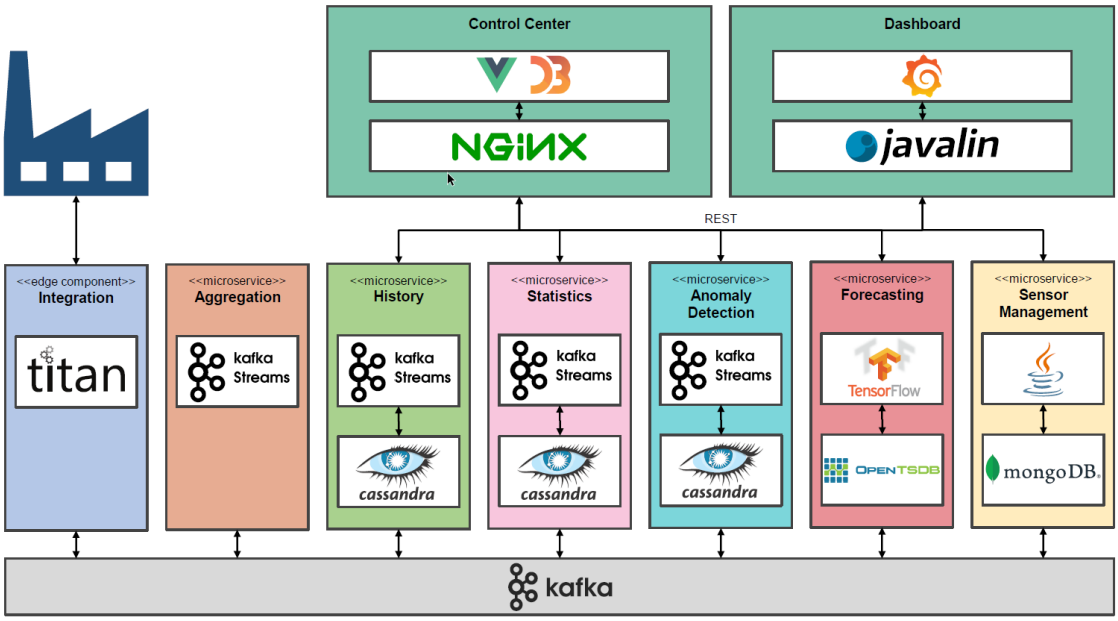

Abbildung 31: titan CC Architektur

Für die Architektur des titan Control Centers wurde eine Event-getriebene, auf Microservices basierende Architektur vorgesehen. Dabei werden verschiedene Analyse-Aufgaben in jeweils eigenständigen, lose gekoppelten Komponenten vorgesehen, den „Microservices“. Einzelne Microservices können unabhängig voneinander entwickelt und den Betrieb gebracht werden, sowie unterschiedliche Technologien in der Umsetzung verwenden. Viele Microservices des titan CC verwenden dabei Konzepte und Technologien aus dem „Big Data“ Umfeld. Die Kommunikation zwischen Microservices erfolgt ausschließlich asynchron über Events, die von einem verteilten Messaging-System (Apache Kafka in unserem Fall) verwaltet werden. Die getroffenen Architekturentscheidungen sollen insbesondere dazu dienen eine hohe Skalierbarkeit, Erweiterbarkeit und Fehlertoleranz des Control Centers zu erreichen [HHM19]. Umgesetzte Microservices sind im Einzelnen (siehe Abbildung 31: titan CC Architektur):

- *Aggregation* Microservice: Führt eine hierarchische Aggregation von Sensor-Messdaten zu Verbrauchsdaten für Gruppen von Sensoren und Maschinen durch [HH20a].
- *History* Microservice: Archiviert Verbrauchsdaten in unterschiedlichen Auflösungen für unterschiedlich lange Zeiträume und bietet geeignete Abfragemöglichkeiten für die Visualisierung hiervon an.

- *Statistics* Microservice: Berechnet kontinuierlich Statistiken, wie z.B. den durchschnittlichen Verbrauch pro Wochentag.
- *Anomaly Detection* Microservice: Untersucht die Datenströme von Sensoren oder aggregierten Daten kontinuierlich auf Anomalien, speichert diese und stellt sie für über geeignete Schnittstellen zur Verfügung.
- *Forecasting* Microservice: Erstellt kontinuierlich Vorhersagen über zukünftige Sensormessdaten, speichert diese und stellt sie für über geeignete Schnittstellen zur Verfügung.
- *Sensor Management* Microservice: Verwaltet Sensoren und ihre Zuordnung zu Gruppen.

Die Visualisierung von Analyseergebnisse der einzelnen Microservices sowie notwendige Konfiguration erfolgt über entsprechende grafische Benutzeroberflächen. Hierzu sieht die Architektur des titan Control Center sieht zwei Arten von Visualisierungen vor, die unterschiedliche Zielgruppen adressiert [HH21a].

Das titan Control Center kann nahtlos mit der titan Plattform integriert werden. Domain-Experten, wie z.B. Facharbeiter, können über eigene Bricks und Flows die Stromverbrauchsdatenströme von Sensoren abgreifen und auf diese Art die Datenaufnahme, Konvertierung sowie Benutzer-spezifische Aggregationen in der titan Plattform umsetzen. Analog werden Analyse-Ergebnisse, wie z.B. erkannte Anomalien, an die titan Plattform zurückgeführt. Nach den Prinzipien des „Fog Computings“ erfolgen die rechenintensiven Analysen des Control Centers in Cloud Umgebungen (private oder public) um den Anforderungen an Skalierbarkeit gerecht zu werden. Die Datenintegration erfolgt wahlweise auch in der Cloud, oder bereits an den Rändern des „Netzes“, d.h. auf den Maschinen der Produktion selbst oder auf Hardware innerhalb der Produktionsumgebung.

Stromverbrauchsdaten werden von Hersteller-spezifischen Formaten mittels der titan Plattform in ein allgemeines Format überführt. Hierzu kommt Ujo zur Beschreibung der Datenschemata und Kodierung der Messwert-Datensätze zum Einsatz. Im Kontext des Anwendungsfalls „Energiemanagement“ wird der Fokus auf Wirkleistungsdaten („Active Power“ in W oder kW) gelegt, wobei andere Messgrößen analog modelliert werden können. Eine Umwandlung von Daten ist beispielsweise im Falle des KN Druckzentrums erforderlich, wo Verbrauchsdaten in KWh mittels der titan Plattform in W umgewandelt werden.

2.1.13 AP4: Realisierung der titan-Plattform

- Methode

  Die titan Plattform wurde mit der Scrum Methode vom Team umgesetzt, nachdem Kanban deutliche Schwächen bei der Zielverfolgung und Nachhaltung von Teilzielen zeigte.
- Quellen, Entwicklerdokumentation und Testsuite
  - https://git.industrial-devops.org/titan/flow-ui/flow-ui
  - https://git.industrial-devops.org/titan/DataFlowEngine/flowengine-go
  - https://git.industrial-devops.org/titan/DataFlowEngine/flowengine-py
- Downloads und Dockerfiles
  - https://pypi.org/user/indevops/
  - Docker und Installation: https://doc.industrial-devops.org/titanPlatform/Docker/serviceimages/
- API Dokumentation
  - https://doc.industrial-devops.org/titanPlatform/

2.1.14 AP4-1: Implementierung der titan-Laufzeitumgebung

Die titan Laufzeitumgebung oder Flow-Engine (FE) wurde für den Prototyp in Python implementiert. Die Flow Engine steht auf dem Python Package Index (PyPI) zum Download zur Verfügung: https://pypi.org/user/indevops/

Für Test und Evaluierung wurde ein Dashboard entwickelt, das die Vorgänge in der FE in Echtzeit darstellt.

Abbildung 32: Dashboard Flow-Metriken

Folgende Metriken werden betrachtet:

- Paketlaufzeit,
- Ausführungszeit von Bricks,
- Zeit in Queue,
- Füllstand der Queues,
- Anzahl FPs (Flow-Packages).

2.1.15 AP4-2: Implementierung der titan-Entwicklungs- und Test-Umgebung

Docker Container werden für Tests und Produktivbetrieb lokal mit Docker-Compose (https://docs.docker.com/compose/) und Produktiv mit Kubernetes betrieben (https://kubernetes.io).

  - Für den Betrieb notwendige Anwendungen:
    - Kafka (https://kafka.apache.org)
    - Grafana (https://grafana.com)
    - Elastic (https://www.elastic.co)
    - InfluxDB (https://www.influxdata.com)
    - MongoDB (https://www.mongodb.com)
- Umsetzung
  - Automatisierung mittels Pipelines in GitLab

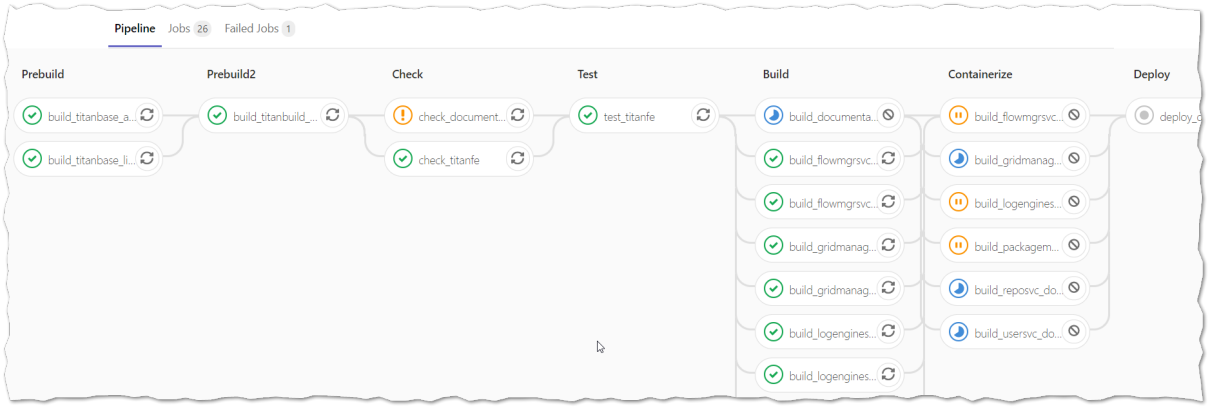

Abbildung 33: Beispiel Pipeline in GitLab

2.1.16 AP4-3: Bibliothek für Visuelle Programmierung des Flows erstellen

Im Projekt sind einige Brick-Pakete entstanden, die für einzelne Arbeiten und Evaluierungen verwendet wurden.

Abbildung 34: Brick-Pakete im Brick-Store

- Basics

  Allgemeine Bricks für grundlegende Standardaufgaben.

  https://doc.industrial-devops.org/titanPackages/basics
- Colors

  Bricks zur Umrechnung von Farbwerten.

  https://doc.industrial-devops.org/titanPackages/colors/
- Databases

  Das Databases Paket bietet Bricks, um Daten aus einem Flow in einer Datenbank zu speichern oder von dort zu lesen.

  https://doc.industrial-devops.org/titanPackages/databases/
- Internet-protocols

  Das Internetprotokoll Paket besitzt verschiedene Bricks mit unterschiedlichen Internetprotokollen für das Versenden und Pollen von Daten.

  https://doc.industrial-devops.org/titanPackages/internet-protocols/
- IoT

  Das IoT Paket enthält Bricks für die Vernetzung von physischen und virtuellen Objekten. Es werden Protokolle des Internet of Things (IoT) verwendet.

  https://doc.industrial-devops.org/titanPackages/iot/
- Socialmedia

  Das Socialmedia-Paket bietet Funktionalität für das Schreiben und Lesen von Nachrichten auf Sozialen-Plattformen.

  https://doc.industrial-devops.org/titanPackages/socialmedia/

- testing

  Das Testing Paket enthält Bricks um einfache Tests und Debugging für die titan Platform durchzuführen.
  https://doc.industrial-devops.org/titanPackages/testing/
- textprocessing

  Das Textprocessing-Paket bietet Funktionalität, um Strings und Texte zu verarbeiten und zu generieren.
  https://doc.industrial-devops.org/titanPackages/textprocessing/
- trello

  Das Trello Package enthält Bricks, die Änderungen an Trello Boards (https://trello.com/en) ermöglichen.
  https://doc.industrial-devops.org/titanPackages/trello/
- weather

  Das Paket beeinhaltet Bricks zum Erfassen und Auswerten von Innenraum- und Aussenbedingung (Wetter).
  https://doc.industrial-devops.org/titanPackages/weather/

2.1.17 AP4-4: Implementierung der titan-Benutzerschnittstelle (UI)

Bei der Implementierung der titan Benutzerschnittstelle kam es auf Übersichtlichkeit und Einfachheit an. Eine universelle Anwendbarkeit war ebenfalls Ziel.

Benutzerführung und Profile

Benutzer der Plattform müssen sich registrieren und ein Nutzerkonto eröffnen.

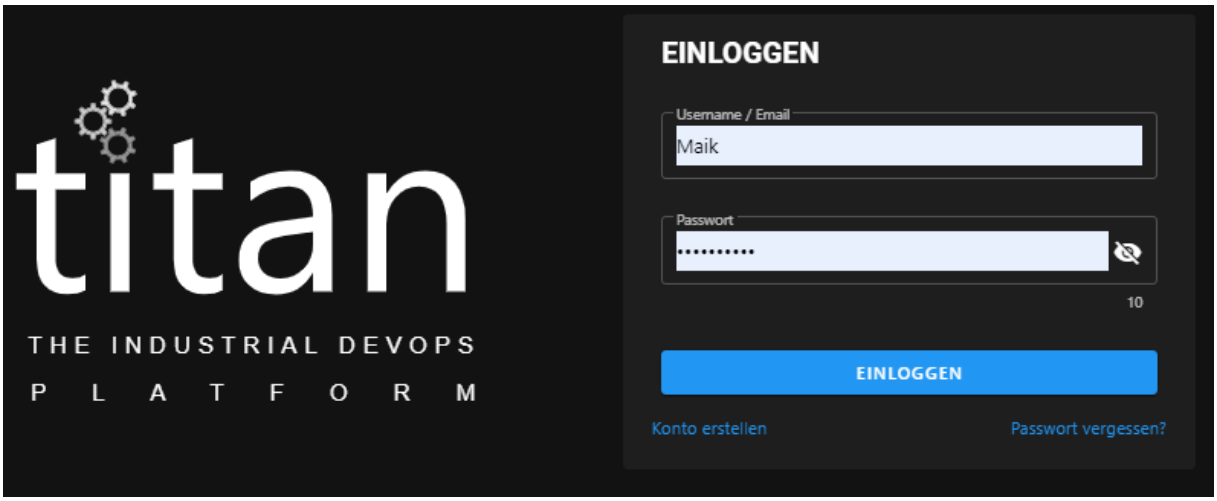

Abbildung 35: UI Login

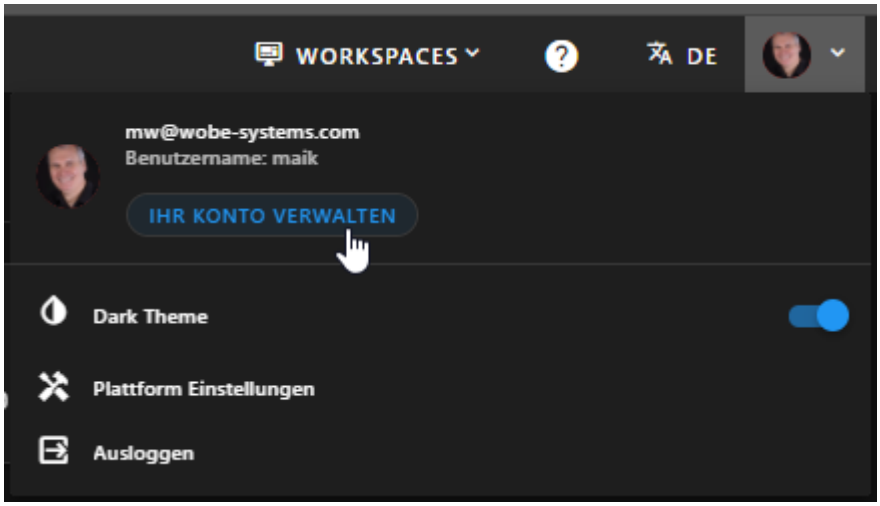

Abbildung 36: Nutzerkonto Verwalten

Für die Einstellung persönlicher Präferenzen bietet das UI die Möglichkeit das eigene Konto zu verwalten. Natürlich kann das Konto dort auch gelöscht werden. Die Löschung erfolgt DSGVO-konform.

Arbeitsbereiche

Die Plattform unterstützt Arbeitsbereiche. Ein Arbeitsbereich ermöglicht es Nutzern Teams zu bilden und Lösungen für unterschiedliche Unternehmensbereiche zu Clustern, damit auch Umfangreiche Anwendungen Übersichtlich und Pflegbar bleiben.

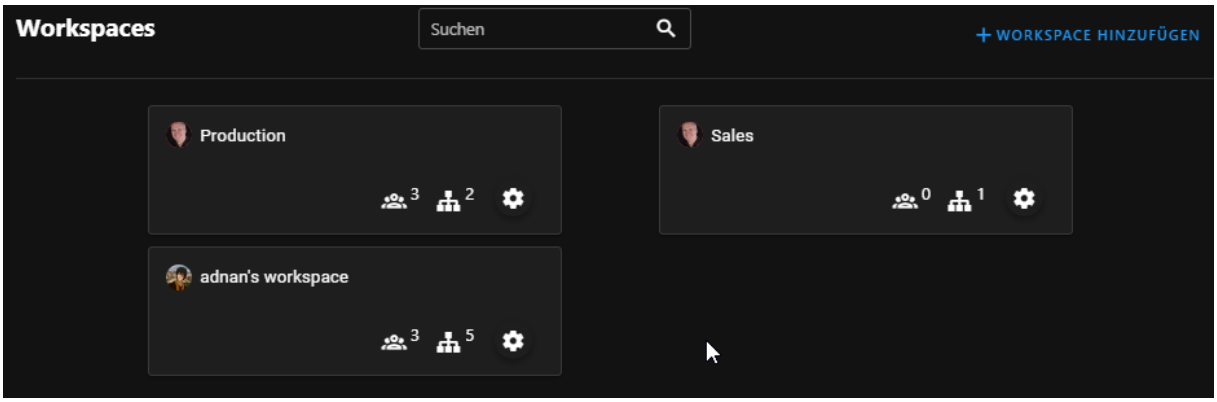

Abbildung 37: Arbeitsbereiche

Flows Editieren

Das UI Modul für die Erstellung und Bearbeitung von Flows ist natürlich ein zentraler Bestandteil des UIs.

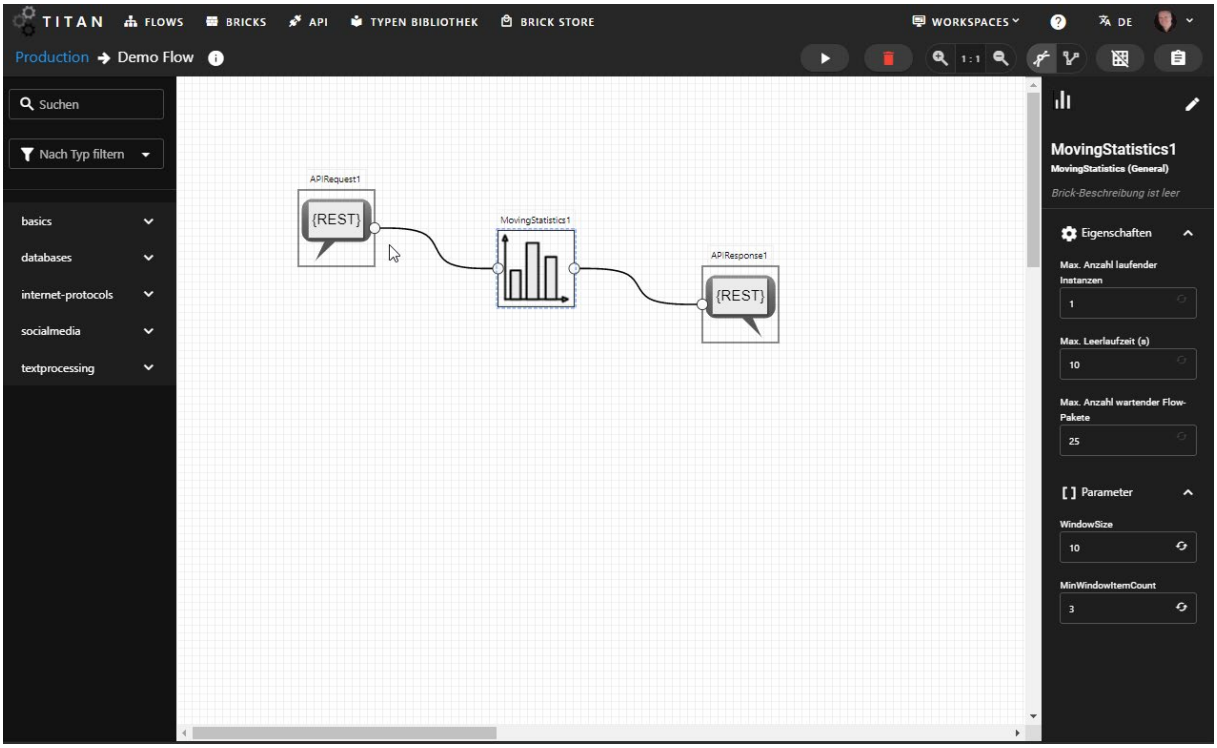

Abbildung 38: Flow Editor

Typen Bibliothek

Für die Verwaltung von Daten erlaubt die Plattform den Nutzern in ihrem Arbeitsbereich Wiederverwendbare Datentypen zu definieren.

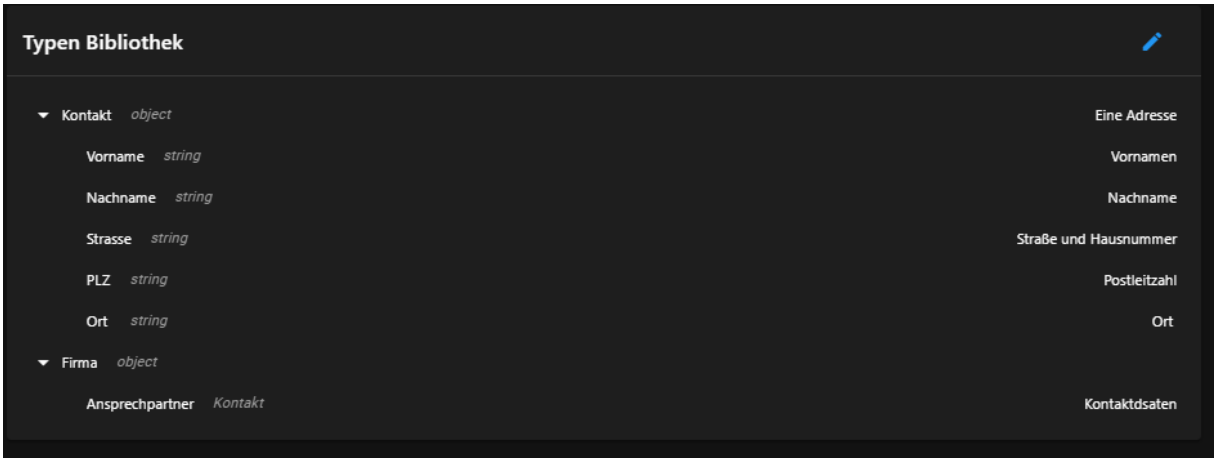

Abbildung 39: Wiederverwendbare Datentypen

APIs

Die Plattform ermöglicht die Definition und Verwaltung von APIs. Aktuell ist es möglich Web-Services zu definieren, die dann mit Flows, implementiert werden.

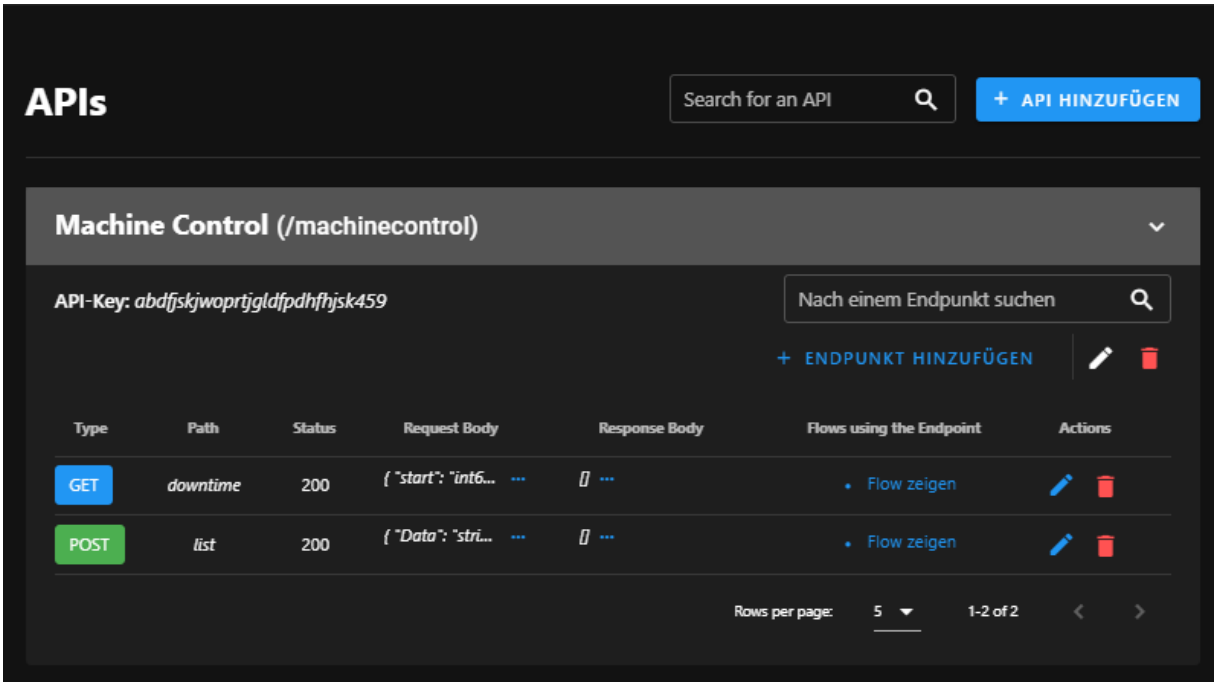

Abbildung 40: Definition von Web-APIs

Brick-Pakete verwalten

Nutzer können in ihrem Arbeitsbereich Brick-Pakete installieren und nutzen. Das UI stellt dafür ein Interface zur Verfügung.

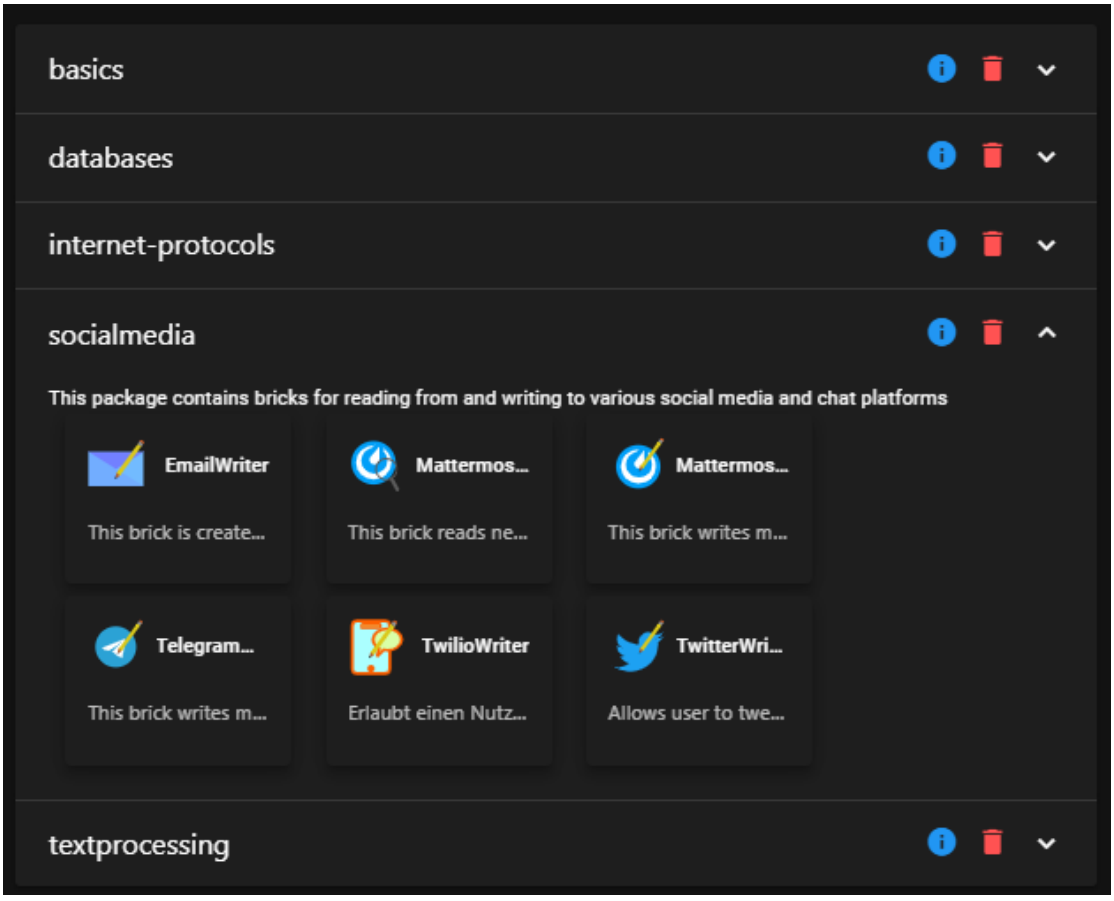

Abbildung 41: Brick-Pakete verwalten

2 Themes

Für besseren Kontrast und Lesbarkeit kann zwischen einem hellen und einem Dunklen Theme gewählt werden.

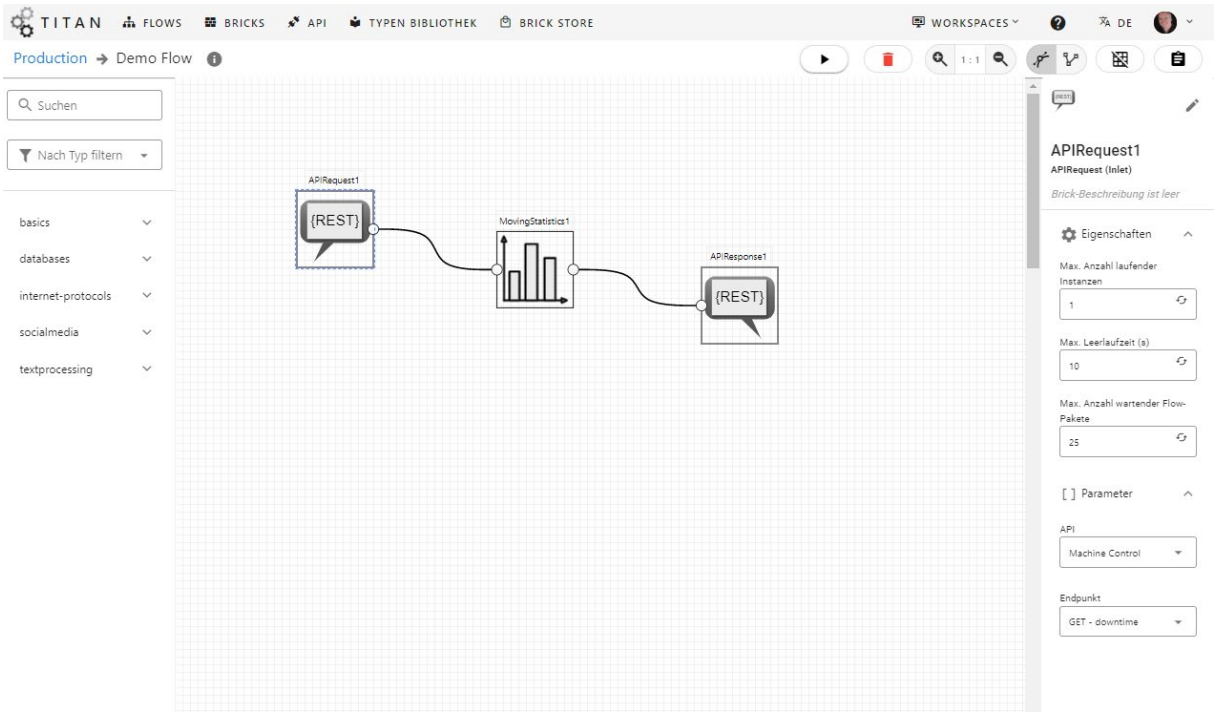

Abbildung 42: Light Theme

2.1.18 AP4-5: Datensicherheit implementieren

- SSL Verschlüsselung der Kommunikation über das Internet.
- Verschlüsselung und Passwortschutz von „Secrets“ – Parametern und Daten die vertraulich sind.
- O-Auth Zertifizierung Internet-Services: https://oauth.net/2/

2.1.19 AP4-6: Entwicklung von Tests und Werkzeugen für die Untersuchung der Grid-Architektur und des Flows

Für die Tests der Architektur und der Flow-Engine wurde das Dashboard (Abbildung 32: Dashboard Flow-Metriken) verwendet.

Für den Aufbau von Test-Flows wurde ein spezielles Brick-Paket entwickelt: https://doc.industrial-devops.org/titanPackages/testing/

2.1.20 AP5: Realisierung titan-CC

Die Realisierung des titan Control Centers wurde über das titan GitLab organisiert, indem Issue-Tracking, Continuous Integration und Versionsmanagement erfolgten: https://git.industrial-devops.org/titan/ControlCenterPrototype

Um Projektergebnisse bereits frühzeitig in Form von wissenschaftlichen Publikationen bereit stellen zu können, wurde Implementierungen außerdem auf GitHub verfügbar gemacht: https://github.com/cau-se/titan-ccp

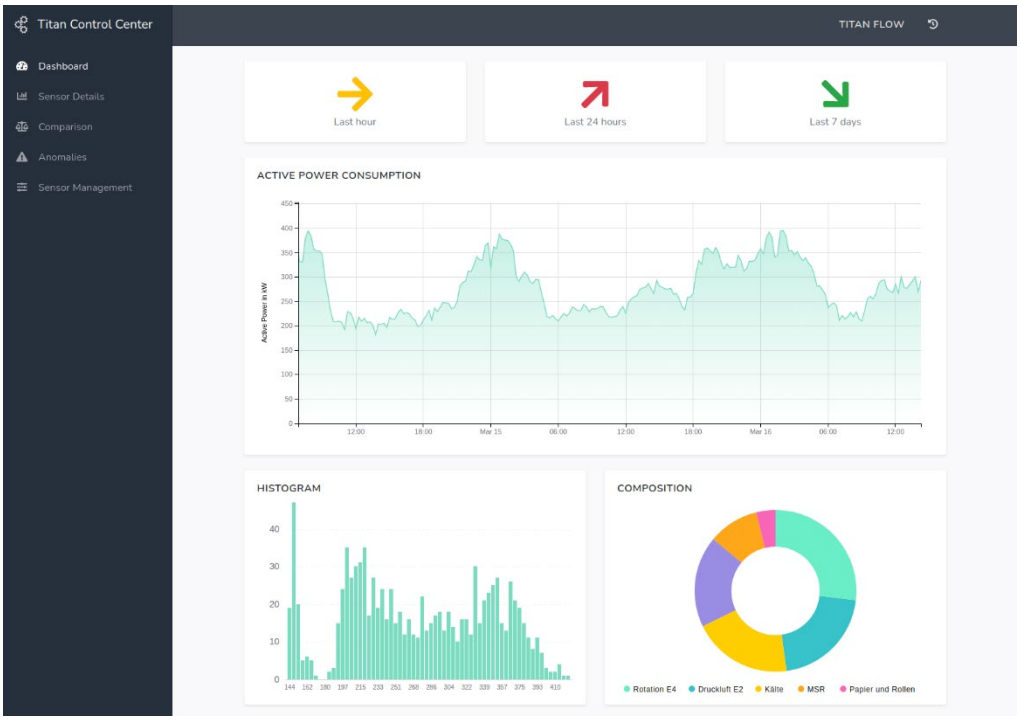

Abbildung 43: Control Center Dashboard

Die einzelnen Komponenten des titan Control Centers (z.B. die Microservices) wurden als Docker Images realisiert und veröffentlicht. Sie können über Docker Hub bezogen werden: https://hub.docker.com/u/industrialdevops

2.1.21 AP5-1: Anwendungsspezifische Komponenten entwickeln

Für den Einsatz von titan im Kontext des Energiemanagements wurden Brick-Pakete zur Integration, Transformation und Aggregation von Energiedaten entwickelt. Bereitgestellte Bricks ermöglichen die Modellierung von vollständigen Flows, die Stromverbrauchsdaten von IBAK und den Kieler Nachrichten an das Control Center übermitteln. Des Weiteren können mit den bereitgestellten Bricks auch öffentliche, industrielle Energiedatensätze angebunden,

die häufig in der Forschung zur Evaluation genutzt werden. Für die Entwicklung von Flows, die weitere Datenquellen anbinden, können bestehende Bricks wiederverwendet werden.

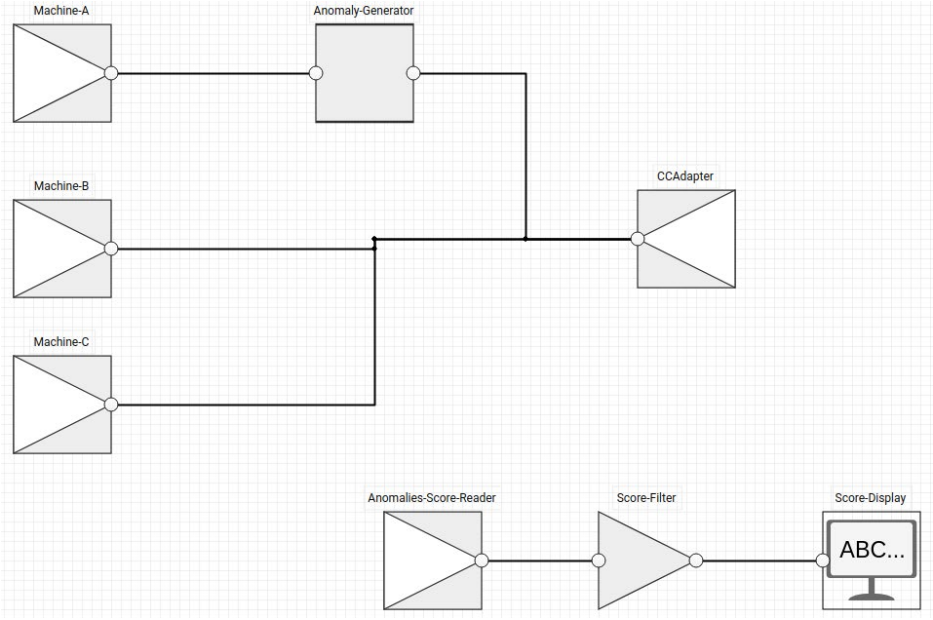

Abbildung 44:Control Center Flow

Vom Control Center bereitgestellte Analyseergebnisse können in der titan Plattform weiterverarbeitet werden. Hierzu stehen Bricks zur Verfügung, die beispielsweise erkannte Anomalien integrieren oder filtern können. Zusammen mit generischen Bricks (z.B. zum Senden von E-Mails) können so Unternehmen-spezifische Flows modelliert werden, die auf den Analysen des Control Centers aufbauen.

Abbildung 45:Anomalieerkennung

2.1.22 AP5-2: Testsuite für die titan Qualitätssicherung erstellen

Packages mit integrierter Testsuite wurden erstellt.

2.1.23 AP5-3: Anwendungsspezifisches UI entwickeln

Das anwendungsspezifische UI des Control Centers visualisiert verschiedene Analyseergebnisse des Control Centers. Hierzu stellt es verschiedene interaktive Visualisierungskomponenten zur Verfügung. Im Rahmen des Anwendungsfalls „Energiemanagement“ umgesetzte Komponenten sind unter anderem:

- Darstellungen von Stromverbrauchsdaten in Form von Zeitreihendiagrammen mit interaktivem Zoomen und Verschieben des Darstellungszeitraums und automatischem Nachladen von Daten
- Linien-, Torten- und Balkendiagramme sowie Heatmaps zur Hervorhebung energieintensiver Maschinen und Zeiträume
- Ein interaktives Werkzeug zum Vergleich der Stromverbrauchszeitreihen verschiedener Maschinen und Geräte
- Anzeige und interaktive Filterung von Anomalien nach Zeitraum und Schweregrat
- Verwaltung und Gruppierung von Sensoren

Ein öffentlicher Showcase des titan Control Centers ist einsehbar über: http://samoa.se.informatik.uni-kiel.de:8185/

2.1.24 AP6: Integration

Neben verschiedenen lokalen Instanzen, die das titan Control-Center und die titan Plattform für die Auswertung von Energiedaten, wurden während der Integrationsphase die Instanzen in der Cloud miteinander verbunden. Dabei wurde das Control-Center im Rechenzentrum der Uni-Kiel installiert. Die titan Plattform wurde bei einem Cloud-Anbieter in Betrieb genommen (https://www.netways.de/en/).

Aus Sicherheitsgründen konnten die Daten in dieser Konstellation nicht direkt aus der Produktionsumgebung übertragen werden. Stattdessen wurden basierend auf den Auswertungen der realen Daten simulierte Daten generiert.

2.1.25 AP7: Evaluierung

Die Ergebnisse der Evaluierung wurden in zahlreichen Veröffentlichungen dokumentiert, die in Kapitel 2.6 aufgeführt sind.

2.1.26 AP8: Koordination und Projektmanagement

- Code of Conduct

Mit Unterstützung von Sabine Wojcieszak von der Firma getNext IT (https://www.getnext-it.com) erstellten wir einen Code of Conduct für die Projektwebseite der regelt, wie wir im Projekt miteinander umgehen wollen: https://www.industrial-devops.org/code-of-conduct/

Wir haben die Anzahl der Regeln bewusst niedrig gehalten, jedoch im Team abgesprochen, dass sie in jeder Situation gelten.

- Kommunikationsinfrastruktur

Für die Kommunikation wurde aus Datenschutzgründen ein Team-Chat gewählt, das auf eigenen Server betrieben wird: https://mattermost.com

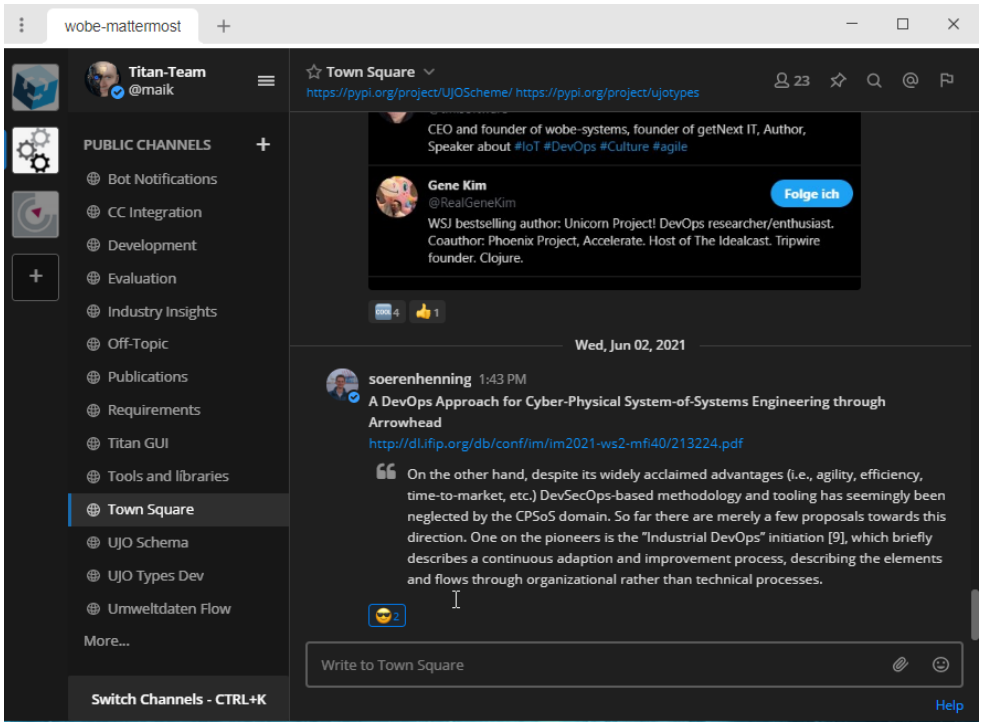

Abbildung 46: Team-Chat Mattermost

- Visualisierung von Projektfortschritt und Status

Für die Visualisierung des Fortschritts im Einzelnen werden die Reports von Atlassian Jira eingesetzt.

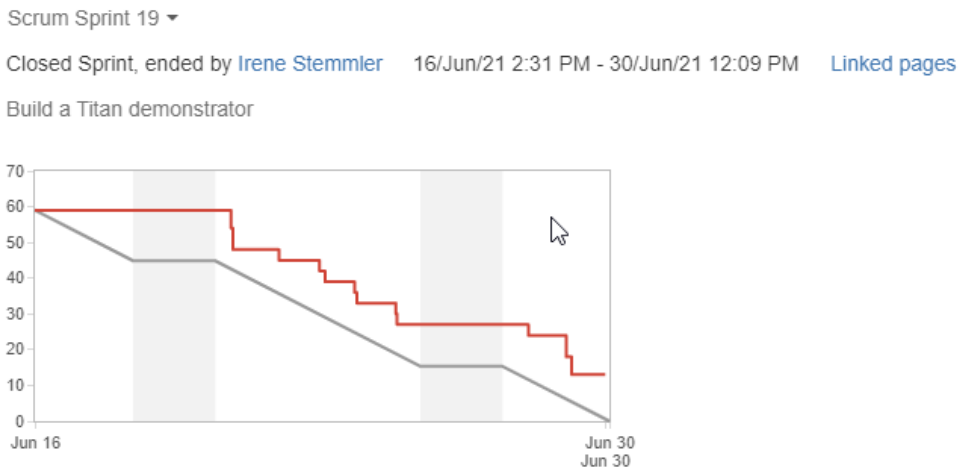

Abbildung 47: Sprint Report

- Retrospektiven

  Retrospektiven waren und sind ein fester Bestandteil im Projekt. Mit Unterstützung von getNext IT haben wir mit unterschiedlichen Methoden immer wieder geprüft, ob wir mit unseren Entscheidungen auf dem richtigen Weg sind.

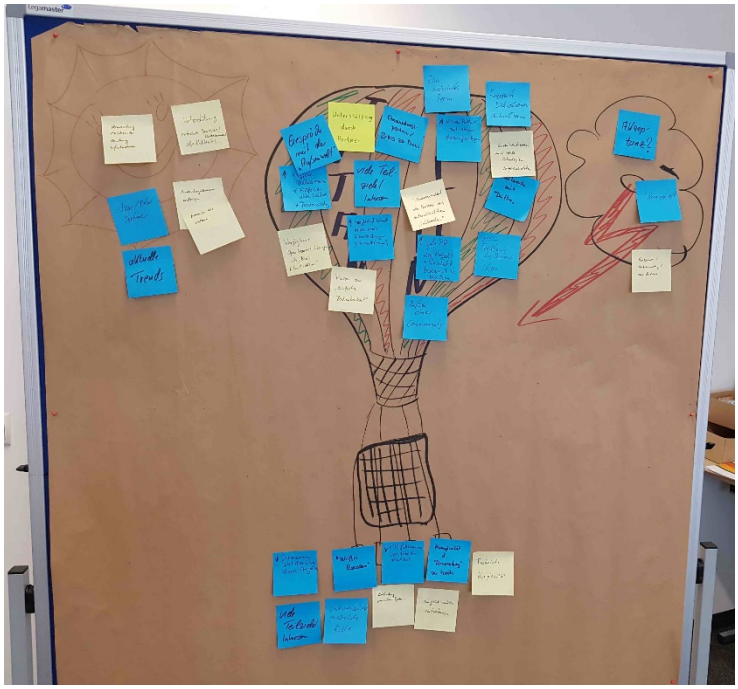

Abbildung 48: titan Heißluftballon

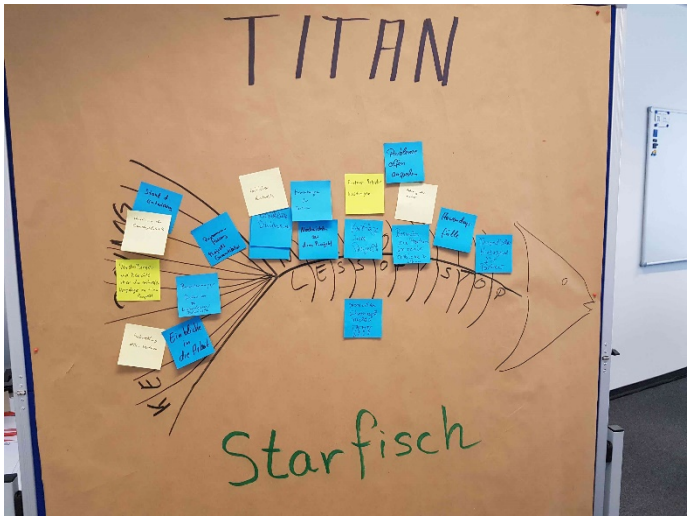

Abbildung 49: Der titan-Starfisch

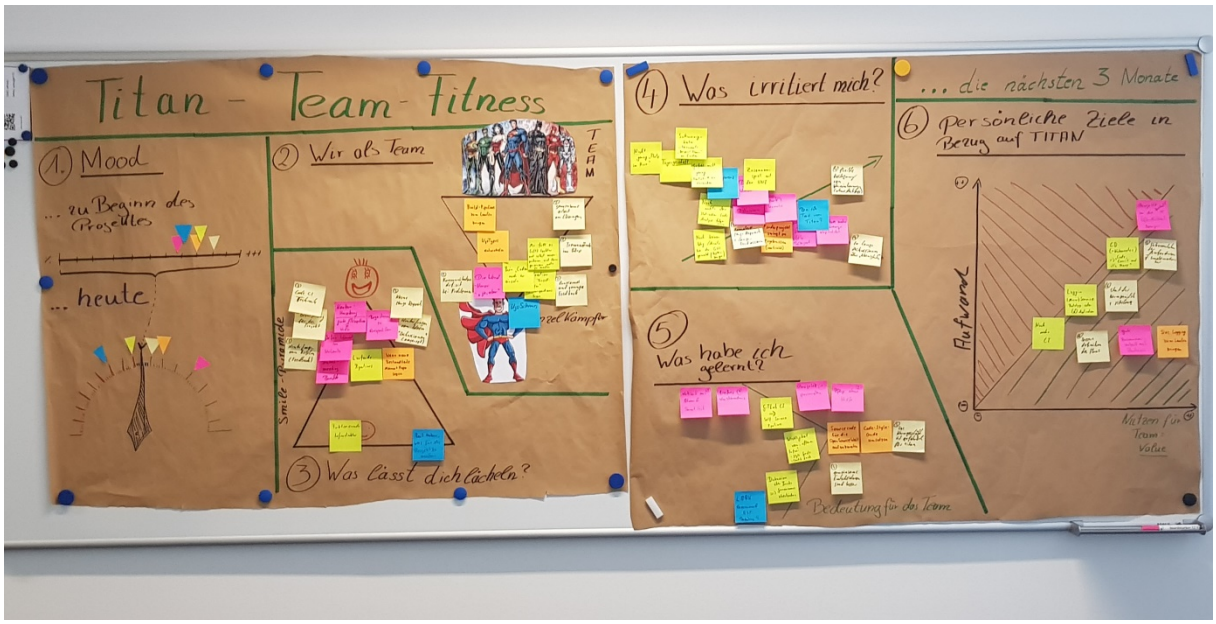

Abbildung 50: Ergebnisse einer Retrospektive

- Statusmeetings
  Besonders hat uns gefreut, dass die assoziierten Partner sich bei den Statusmeetings während der gesamten Projektlaufzeit engagiert haben. Für die Steuerung des Projekts waren das Feedback und die Diskussionen eine wertvolle Informationsquelle.

Abbildung 51: Statusmeeting bei den Kieler Nachrichten

2.2 Die wichtigsten Positionen des zahlenmäßigen Nachweises

Für wobe ergeben sich entsprechend dem Schlussverwendungsnachweis mit 50%-Förderung:

- Personalaufwendungen inkl. Gemeinkosten : 858.122,98 €
- Gesamt : 858.122,98 €

Für die CAU ergeben sich entsprechend dem Schlussverwendungsnachweis mit 100%-Förderung:

- Personalaufwendungen: 195.311,80 €
- Sachmittel (Dienstreisen): 4.868,35 €
- Gesamt: 200.180,15 €

2.3 Notwendigkeit und Angemessenheit der geleisteten Arbeit

Aus unserer Sicht hatten sich die Aussichten, die gesetzten Ziele zu erreichen, nicht prinzipiell geändert. Gespräche auf Konferenzen aber auch mit aktuellen Kunden aus dem Kerngeschäft hatten die bereits im Vorfeld getroffenen Annahmen darüber, wie Systemintegration derzeit in Unternehmen behandelt wird, bestätigt.

Weiterhin lernten wir verstärkt, dass neben der Datensammlung aus einer IT- und Produktionsinfrastruktur im Rahmen des titan Control Center die Möglichkeit einer flexiblen Datenintegration eine zunehmend wichtige Rolle in Unternehmen spielt.

Seit Beginn des Projektes stellten wir fest, dass das öffentliche Interesse auch seitens der Regierung an der Thematik „Digitalisierung“ immer weiter Fahrt aufnimmt und mit konkreten Zielen hinterlegt wird. So hat zum Beispiel der Ministerpräsident des Landes Schleswig-Holstein das Digitalisierungsprogramm des Landes vorgestellt.

- https://digitalisierung.schleswig-holstein.de/pdf/Digitalisierungsprogramm_Schleswig-Holstein.pdf

- https://www.schleswig-holstein.de/DE/Schwerpunkte/Digitalisierung/Digitalisierungsprogramm/digitalisierungsprogramm_node.html

Hieraus geht hervor, dass inzwischen sowohl die Notwendigkeit, der Nutzen als auch der Bedarf für eine ressortübergreifende Zusammenarbeit in der digitalen Transformation erkannt ist. Dies trifft nicht nur auf die von bislang adressierten Unternehmen in der produzierenden Industrie, sondern auch auf die Bereiche öffentliche Verwaltung, Bildung und Gesundheit zu. Auch hier herrscht die Situation vor, dass viele verschiedene Systeme genutzt werden, die noch nicht in ein übergeordnetes System integriert werden können. Daher ist auch hier ein umfassender Nutzen der vorhandenen Daten und ein entsprechendes Monitoring kaum oder nur erschwert möglich.

Die Zielsetzung stieß, so wie sie war, auf großes Interesse. Die Umsetzung bleibt blieb unserer Sicht ebenfalls genau wie geplant sinnvoll und machbar, wenn auch mit Verzögerungen aufgrund verspäteten Personaleinsatzes zum Beginn des Projekts. Des Weiteren ergaben sich zeitliche Verschiebungen innerhalb der Arbeitspakete im Vergleich zur Planung, da es sich sinnvoll und praktikabler erwies, sich zunächst um andere – laut Planung später anstehende - Themen zu kümmern und diese vorzuziehen. Dadurch wurden Themenpakete etwas nach hinten verschoben. Dies waren aber lediglich Verschiebungen innerhalb des gesetzten Zeitrahmens.

Der bereits erwähnte verspätete Personaleinsatz zum Beginn des Projekts verursachte weiterhin leichte Verzögerungen, die bis kurz vor Ende des Projekts nicht kompensiert werden konnten.
Während der Integration der verschiedenen Module zeigte sich, dass für einen erfolgreichen Einsatz der titan Plattform zur Evaluierung des Projekts nachgebessert werden musste. Damit der erfolgreiche Abschluss und belastbare Ergebnisse während der Evaluierung nicht gefährdet wurden, beantragten die Konsortialpartner eine kostenneutrale Verlängerung der Projektlaufzeit.

Die beantragte kostenneutrale Laufzeitverlängerung wurde für die Konsortialpartner genehmigt. Die Laufzeit des Projekts verlängerte sich somit um 6 Monate.

Die grundsätzliche Arbeitsplanung wurde beibehalten und lediglich um 6 Monate verschoben, d.h. in dieser Zeit wurde die Evaluierungshase abgeschlossen.

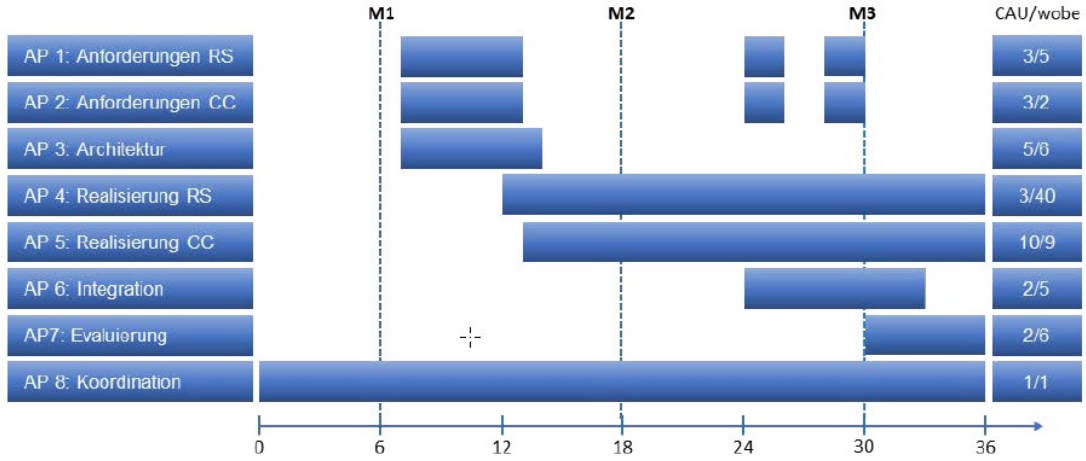

Abbildung 52: Neue Planung der Restlaufzeit

Ergebnisse und Tätigkeiten aus dieser Phase umfassen:

- Zur Evaluation der Titan Flow Engine und des Titan Control Centers wurden weitere Bricks entwickelt, die es z.B. ermöglichen, vom Control Center detektierte Anomalien über die Flow Engine weiter zu verarbeiten. In Ergänzung zu der von wobe öffentlich bereitgestellten Titan Instanz, wurde von der CAU eine öffentliche Instanz des Control Centers bereitgestellt, die Daten aus der Flow Engine verarbeiten kann. Interessierte Nutzer können auf diese Art eigene Daten über die Flow Engine verarbeiten und diese anschließend vom bereitgestellten Control Center analysieren und visualisieren lassen.
- Basierend auf den Rückmeldungen der Statustreffen, wurde das Titan Control Center um zusätzliche Funktionen erweitert. So wurden zusätzliche Visualisierungen entwickelt, die beispielsweise „Heatmaps“ abbilden und es ermöglichen, saisonale Muster in den analysierten Daten zu erkennen. Vom KN Druckzentrum wurden reale Stromverbrauchsdaten für einen Zeitraum von 4,5 Jahren zur Verfügung gestellt. Um solche, großen Datenmengen visualisieren zu können, wurde ein Verfahren entwickelt, bei dem Daten in verschiedenen Auflösungen gespeichert werden, die die

Visualisierung-Software-Komponente automatisch und nur nach Bedarf lädt. Des Weiteren wurde die grafische Benutzungsoberfläche des Control Centers intuitiver gestaltet.
- Für das Journal „Software Impacts“ wurde ein Artikel eingereicht und zur Veröffentlichung akzeptiert, der darstellt, wie das Titan Control Center die Forschung im Kontext von Industrial DevOps Analytics fördert.
- Im Rahmen einer Masterarbeit wurde Titan für die Analyse von Umweltdaten (z.B. Feinstaub oder Stickstoffdioxid) eingesetzt. Hierzu wurden Bricks entwickelt, die von verschiedenen Diensten aktuelle und historische Daten zur Feinstaub- und Stickstoffdioxidbelastung sowie Wetterdaten abfragen. In eigens entwickelten Flows wurden diese Daten miteinander korreliert und über Dashboards visualisiert.
- Die Methode zur Evaluation der Skalierbarkeit von Datenfluss-Architekturen wurde weiterentwickelt. Es wurden die aus dem Titan Control Center abgeleiteten Anwendungsfälle für weitere Stream Processing Systeme implementiert, um diese Systeme in Bezug auf Skalierbarkeit vergleichen zu können. Weiterhin wurde an einer effizienteren Ausführungsmethode geforscht, die auf dem 11th Symposium on Software Performance vorgestellt wurde, sowie eine Metrik für Skalierbarkeit entwickelt, die zur Begutachtung beim 9th International Workshop on Load Testing and Benchmarking eingereicht wurde.

2.4 Voraussichtlicher Nutzen, insbesondere der Verwertbarkeit des Ergebnisses im Sinne des fortgeschriebenen Verwertungsplans

Verwertungsplan

Die Förderperiode wurde zum 31.01.2021 abgeschlossen. Die Ziele für den Prototypen wurden erreicht und teilweise übertroffen. Aktuell wird die Plattform zu einem Produkt weiterentwickelt und Anwendern für den produktiven Einsatz für kleinere Projekte angeboten.

Neben Anwendern aus der Industrie, die über Partnerschaften mit Lösungsanbietern erreicht werden, soll unter dem Stichwort „Citizen-Developer“ die Plattform in der Cloud für jeden kreativen Menschen zur Verfügung stehen.

Die Evaluierungsphase brachte überdies interessante Aspekte auf, die halfen, die Verwertung der Projektergebnisse anzugehen. Entstanden ist das Projekt aus der Idee, Fachkräfte mit umfangreichen Möglichkeiten auszustatten, digitale Systeme selbst zu gestalten und eigene
Lösungen zu schaffen. Im Einzelnen sind folgende Aspekte zu nennen:

- Einordnung der Plattform im Bereich Industrial Internet of Things (IIoT)
- Einordnung über Low-Code zu No-Code Programmierung.
- Industrial-DevOps als geeigneter Methodenpool für organisatorische Maßnahmen, um die kontinuierliche Entwicklung an IT-Systemen durch Fachkräfte sinnvoll zu steuern.

Argumentativ entsteht nun ein Gesamtbild, das die im Internet und in den Fachmedien diskutierten Herausforderungen adressiert. Basierend darauf wurde die Internetseite des Projekts entsprechend angepasst.

Abbildung 53: Internetauftritt des Industrial DevOps Projekts

Die Kooperationspartner planen die Projektergebnisse der Öffentlichkeit vorzustellen und auch nach Abschluss des Projekts die Betreuung einer interessierten Community aktiv weiterzuführen. Der Kern der titan-Plattform wird als Open-Source veröffentlicht und so anderen Unternehmen für eigene Lösungen zugänglich gemacht. Die Projektpartner erwarten, dass so das Potential der Idee und der Technologie voll ausgeschöpft werden kann.

Der Projektpartner wobe plant, basierend auf dem Open-Source-Kern der titan-Plattform, Komponentenpakete für die grafische und verarbeitende Industrie zu entwickeln. Im Rahmen ihres bewährten Geschäftsmodells soll so ein Angebot für KMUs für iterative, individuelle Software- und Integrations-Projekte entstehen. Für Unternehmen, die domänenspezifische Komponenten für titan entwickeln möchten, plant wobe ein Schulungs- und Beratungsangebot zu entwickeln.

Die Kernidee, Entwicklung und Betrieb von Software, auch in der Industrieautomatisierung, technologisch und organisatorisch enger miteinander zu verbinden, bieten die Projektpartner in Veröffentlichungen und Vorträgen an. Wir erwarten damit eine Diskussion über einen wichtigen Aspekt der Digitalisierung und Automatisierung, sowie Industrie 4.0 anzuregen. Die sich daraus ergebenen Nutzungsmöglichkeiten und Produktideen sind kaum absehbar.

Durch die Vernetzung der Projektpartner im Cluster „Digitale Wirtschaft Schleswig-Holstein“ (DiWiSH) und die assoziierte Partnerschaft mit dem „Maritimen Cluster Norddeutschland“ (MCN) ermöglicht es, schon während der Projektlaufzeit, zahlreiche Unternehmen aus unterschiedlichen Branchen zu erreichen. Mit dem Cluster Management DiWiSH (Digitale Wirtschaft Schleswig-Holstein, http://diwish.de) und dem Kompetenzverbund KoSSE (Kompetenzverbund Software Systems Engineering, http://kosse-sh.de) ist eine Zusammenarbeit in der Öffentlichkeitsarbeit für das titan-Projekt geplant, um die Projektergebnisse unter anderem den zurzeit 190 Mitgliedsunternehmen dieses Clusters bekannt zu machen.

Mittlerweile ist wobe auch Mitglied in der Open Industry 4.0 Alliance, https://openindustry4.com/de/, ein Netzwerk von über 80 führender Hersteller, die offene Standards nutzen, um Interoperabilität zu fördern. Das gemeinsame Ziel: Die Industrie 4.0 entscheidend voranzubringen.

Wirtschaftliche Erfolgsaussichten

Projektpartner und assoziierte Partner bewerten die wirtschaftlichen Erfolgsaussichten als sehr vielversprechend. Mit fast zwei Jahrzehnten Erfahrung in Automatisierungsprojekten der industriellen Zeitungsproduktion geht wobe davon aus, dass titan geeignet ist, die Wettbewerbsfähigkeit des Unternehmens nachhaltig zu sichern und den Umsatz innerhalb weniger Jahre deutlich zu erhöhen. Diese Annahme basiert auf der Beobachtung, dass sich im Bereich des Akzidenzdrucks ein Trend zu mehr Integration abzeichnet. Der Akzidenzdruck ist bezogen auf Digitalisierung und Automatisierung in etwa vergleichbar mit der fertigenden Industrie. Das Know-how im Bereich Print-Medien und die Erfahrungen aus

dem Zeitungsdruck, kombiniert mit einer flexiblen Integrations-Plattform, wie titan, sind eine hervorragende Basis für die Gewinnung neuer Kunden.

wobe bietet seit mehreren Jahren seine Erfahrungen im Bereich der fertigenden und verarbeitenden Industrie für individuelle Projekte an. Der Schwerpunkt liegt dabei auf Industrie 4.0 und dem Internet der Dinge. Dieser Markt ist gerade erst dabei, sich zu entwickeln und wobe geht davon aus, dass die Ergebnisse des titan-Projekts genutzt werden können, sich in diesem stark umkämpften Markt zu behaupten. Der modulare Aufbau von titan und die Möglichkeit, domänenspezifische Komponenten zu verwenden, würde wobe ermöglichen, auch weiterhin wettbewerbsfähige Projekte anzubieten.

Sollte die Technologie von anderen Softwareunternehmen aufgegriffen werden, so ergeben sich Möglichkeiten für weitere Umsätze im Bereich Beratung und Schulung.

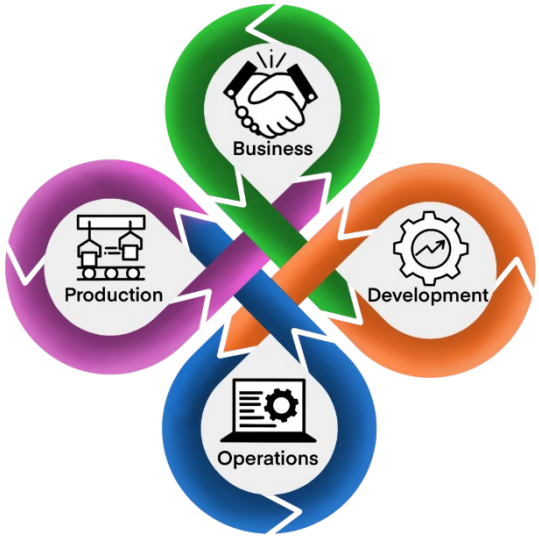

Abbildung 54: Industrial DevOps im Unternehmen

wobe hat eigene Produkte bereits am Markt eingeführt und in hochdynamischen, variablen Produktionsumgebungen Projekte erfolgreich umgesetzt. Damit ist das grundlegende Wissen, wie auch titan erfolgreich genutzt werden kann, vorhanden und praktisch erprobt.

Als Forschungspartner verfolgt die Christian-Albrechts-Universität zu Kiel keine direkten wirtschaftlichen Verwertungsabsichten. Indirekte wirtschaftliche Verwertungsperspektiven ergeben sich jedoch durch positiven Synergieeffekte aus der geplanten Open-Source-Softwareentwicklung. Die Arbeitsgruppe Software Engineering der Christian-Albrechts-Universität zu Kiel hat bereits in der Vergangenheit sehr gute Erfahrungen beispielsweise mit der langjährigen Entwicklung (inzwischen über zehn Jahre) des Monitoring-Frameworks Kieker für die Akquise neuer Forschungsprojekte gemacht, wie es in dokumentiert wird. Wir

erwarten ähnliche Verwertungsmöglichkeiten durch die Open-Source-Softwareentwicklung in titan.

Wissenschaftlich-technische Erfolgsaussichten

Christopher Little, einer der ersten DevOps Chronisten sagte: „Jedes Unternehmen ist ein Technologieunternehmen, egal was sie denken in welchem Geschäft sie sind. Eine Bank ist ein IT-Unternehmen mit einer Banklizenz“. Man sagt, es sei nahezu unmöglich, dass eine Geschäftsentscheidung nicht mindestens eine IT-Änderung zur Folge hat. Die Kooperationspartner sehen Industrie 4.0 als die Entwicklung der Industrie zu einem Netzwerk von IT-Unternehmen mit Produktionsmaschinen. Diese Entwicklung macht Unternehmen anfällig für eine Abwärtsspirale aus Zeitmangel, technischer Schuld, Workarounds und Burnout von Mitarbeitern. Am Ende stehen unvorhersehbare, hohe Kosten und schlimmstenfalls ein Unternehmen, das unfähig ist, Geschäftsentscheidungen zeitnah umzusetzen.

Seit Patrick Debois und Andrew Schafer 2008 auf der Agile Conference in Toronto vorgeschlagen haben, die Anwendung agiler Methoden von der Softwareentwicklung auf IT-Infrastruktur auszuweiten, ist DevOps eine Erfolgsgeschichte. Die Kooperationspartner erwarten, dass DevOps Methoden auf industrielle Automatisierung und Industrie 4.0 anzuwenden ähnliches Potential hat.

Wird das Risiko der Abwärtsspirale auf dem Weg zum vernetzten, digitalen Unternehmen minimiert, würde die Hemmschwelle sinken für dringend notwendige Investitionen in die Wettbewerbsfähigkeit der Zukunft.

Das Ergebnis der wissenschaftlichen Prüfung einer Reihe von Hypothesen hat sich im Projektverlauf bestätigt:

- In-Situ Development: Softwaresysteme können sinnvoll und sicher in Produktionsumgebungen entwickelt und getestet werden. Dieses Vorgehen ist geeignet, um die Kosten für derartige Systeme erheblich zu senken.
- Die Methoden und Vorgehensmuster aus DevOps, die in großen Internetsystemen zu hoher Flexibilität und Ausfallsicherheit führen, lassen sich auf große, heterogene, stark integrierte Systeme übertragen.
- Flow-basierte Automatisierung ist geeignet, von Personen ohne IT-Hintergrund sicher und effizient eingesetzt zu werden

Für die wobe-systems GmbH war das Projekt die Chance, grundlegende Erkenntnisse zu erlangen, die ihre Produktentwicklung und den wirtschaftlichen Erfolg nachhaltig beeinflussen können.

Die Arbeitsgruppe Software Engineering der Christian-Albrechts-Universität zu Kiel hatte bereits Erfahrungen mit DevOps als innovative, agile Softwareentwicklungsmethode zur Integration von Entwicklung und Betrieb komplexer Softwaredienste. Der neue Ansatz Industrial DevOps bietet die Chance, sich über methodische und technische Alleinstellungsmerkmale zu differenzieren und so Innovation und Wettbewerb im Markt der DevOps-Lösungen zu fördern.

Auf dieser Grundlage ergibt sich für die Christian-Albrechts-Universität zu Kiel ein großes Potential zur Einwerbung weiterer Drittmittel. Ein wesentliches Erfolgskriterium wird es sein, ob im KoSSE-Kontext weitere Drittmittel-Projekte im Bereich des Industrial DevOps eingeworben werden können. Die Projektergebnisse von titan werden darüber hinaus auch in Forschung und Lehre der Arbeitsgruppe Software Engineering, Christian-Albrechts-Universität zu Kiel, Verwendung finden.

Dies schlägt sich auch in den Lehrinhalten der Arbeitsgruppe nieder. Industrial DevOps und somit auch die durch titan zu entwickelnden Konzepten bilden für die Studierenden attraktive, aktuelle Themenfelder, da bei potenziellen Arbeitgebern eine große Nachfrage nach diesbezüglichem Know-how besteht. Des Weiteren stellen die Projektergebnisse auch eine Basis für studentische Abschlussarbeiten und Promotionen dar. In Form von Beiträgen für Konferenzen und Fachzeitschriften sollen die Ergebnisse zudem einer breiteren Fachöffentlichkeit zugänglich gemacht werden.

Für die Arbeitsgruppe Software Engineering der Christian-Albrechts-Universität zu Kiel haben wir insbesondere auf Basis der Entwicklung und Evaluierung der titan-CC-Komponente neue wissenschaftlich-technische Forschungsergebnisse erzielt, die geeignet publiziert und in weiteren Forschungsprojekten verwertet werden können. Das Thema Industrial DevOps wird zukünftig kontinuierlich wissenschaftlich weiterbearbeitet. Durch das titan-Projekt wurde es uns nun ermöglicht, neben der wirtschaftlichen Verwertung einen wissenschaftlichen Impact mit dem Thema Industrial DevOps zu erreichen.

Verwertung der Projektergebnisse nach Projektende

Im Rahmen einer Community wird die titan-Open-Source-Plattform weiterentwickelt. Die während des Projekts entstandenen Innovationen werden so verfeinert und in verschiedenen

Bereichen angewendet. Die Erfahrungen aus Projekten fließen in die Software ein und werden innerhalb der Community verbreitet.

Die Projektpartner planen weiterhin die Ergebnisse auf Konferenzen und Fachtagungen vorzustellen. Ziel dabei ist es, weitere Strategie- und Technologie-Partner aus Wissenschaft und Wirtschaft zu finden, die aus ihrem Gebiet Wissen und Erfahrungen beitragen können.

Die wobe-systems GmbH entwickelt nach Abschluss des Projekts basierend auf den Ergebnissen der Open-Source-Plattform aktuell Produkte und Services für ihren Geschäftsbereich. Es werden dabei folgende Gesichtspunkte vorrangig betrachtet.
Gemeinsam mit unseren Partnern werden wir Einstiegslösungen für Industrieanwender vermarkten. Zum Beispiel ist eine Installation der Plattform auf einem einzelnen Industrie-PC geplant, der es erlaubt, beliebige Sensoren anzuschließen und auszuwerten.
Jede installierte Instanz der titan No-Code Plattform kann an den sogenannten „Brick-Store“ angeschlossen werden. So können Anwender neue und aktualisierte Funktionspakete nutzen.
Geeignete Geschäftsmodelle sind in der Umsetzungsphase. wobe-systems wird zum Beispiel für die Plattform ein geeignetes Programm für die Unterstützung von professionellen Anwendern und Entwicklern aufbauen:

- Installation On-Premises oder in der Cloud
- Professioneller Service und Support
- Integration von beliebigen Daten aus verschiedenen Abteilungen eines Industriebetriebs in einem System.
- Auswertung der Daten für frühe Fehlererkennung und flexible Maschinenwartung
- Projektumsetzung und Brick-Entwicklung (Domänen-spezifische Komponentenentwicklung für die titan-Plattform)
- Schulung, Qualifizierung und Zertifizierung von Brick-Entwicklern und Flow-Managern
- Prozessautomatisierung durch nicht IT-Personal beim Anwender
- Beratung und Training (Schrittweise Entwicklung zu einem digitalen Industrie 4.0 Unternehmen
- Qualitätssicherung und Zertifizierung von Brick-Paketen
- Betrieb der IT-Infrastruktur (Zusammenarbeit mit Partnern für IT-Systeme)

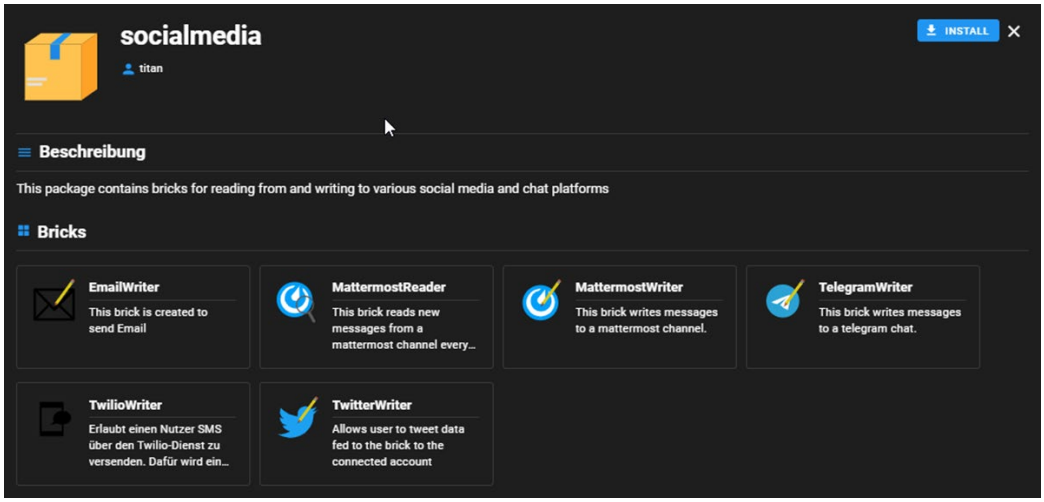

Abbildung 55: Brick-Paket Social Media

Nachhaltigkeit des Verwertungsplans

In seinem Artikel „Looking Into The Future: Five Characteristics That Will Define The Future Of Work" beschreibt Ben Wharfe wie Unternehmen zukünftig ihre Innovationsfähigkeit steigern werden und dabei IT-Technologie einsetzen. Ein Aspekt dabei ist der Einsatz von Open Source und die Unterstützung des Innovationsprozesses durch eine Community. Was passiert, wenn die Innovationsgeschwindigkeit schneller ist, als der Patentierungsprozess? Auch KMUs müssen sich zukünftig diesen Fragen stellen. Die titan-Plattform ist ein Werkzeug, um sich auf den Weg zu machen.

Die konsequente Nutzung von Open Source, der Aufbau einer Community und die Verbreitung der damit verbundenen Ideen bei Anwendern aus der Industrie, kann die Richtung der Digitalisierung verändern. Weg von klassischen „Aus einer Hand" – Systemen hin zu flexiblen und offenen Lösungen, die dem Anwender maximale Autonomie bieten.

Wissenschaft und Technologie

No-Code kann eine Möglichkeit sein, zukünftig auch ohne ständige Anpassungen von Software in Unternehmen flexibler auf neue Anforderungen zu reagieren. Was die Anforderungen an die Plattform selbst angeht, steht diese Entwicklung noch am Anfang. Während im titan-Projekt bereits organisatorische Abläufe untersucht wurden und eine wichtige Rolle spielte, sind andere Plattformen auf die funktionale Entwicklung fokussiert.

Wir sehen für die Entwicklung von No-Code Plattformen folgende Herausforderungen:

- Einfache Erstellung und Integration von Daten und Datenmodellen
- Unterstützung von Kollaboration und Team Topologien (https://teamtopologies.com)

- Unterstützung von Domain Driven Design (https://de.wikipedia.org/wiki/Domain-driven_Design)
- Hyperskalierbarkeit ohne Anpassung der Lösung (https://en.wikipedia.org/wiki/Hyperscale_computing)
- Einsatz von Low-Code für die Entwicklung von Bricks
- Verteilte Knoten mit Software ausstatten und dynamisch aktualisieren

Der professionelle Einsatz von No-Code jenseits einer lediglich flexibleren Konfiguration als echte Programmierung in Unternehmen steht ganz am Anfang. IT-Analysten wie Gartner oder Forester Research bestätigen aber in ihren Prognosen für die nächsten Jahre einen starken Aufwärtstrend in dieser Software-Technologie.

2.5 Während der Durchführung des Vorhabens dem ZE bekannt gewordene Fortschritte auf dem Gebiet des Vorhabens bei anderen Stellen

Während der Durchführung des Projekts wurden unseres Wissens folgende Ergebnisse zum Thema „Industrial DevOps“ bekannt.

- *Industrial DevOps*. IT Revolution. 2018.: „Applying DevOps and Continuous Delivery to Significant Cyber-Physical Systems“
- *Applied Industrial DevOps*. IT Revolution. 2019.: „Practical Guidance for the Enterprise“
- *Continuous Delivery and DevOps in industrial environments*. Siemens. 2019.: „Changing product development for a changing world“
- *Gartner Forecasts Worldwide Low-Code Development Technologies Market to Grow 23% in 2021*. Gartner. 2021.: „Digital Business Acceleration Drives Application Delivery“

Die grundlegende Idee Facharbeiter No-Code/Low-Code Plattformen zu befähigen IT-Lösungen zu schaffen wird zum Ende des Projekts in der Öffentlichkeit auf sozialen Medien breit und engagiert diskutiert. Gartner sieht Low-Code als Technologie der Zukunft.

2.6 Erfolgte oder geplanten Veröffentlichungen der Ergebnisse

Gespräche auf Konferenzen, aber auch mit aktuellen Kunden aus dem Kerngeschäft, haben die bereits im Vorfeld getroffenen Annahmen darüber, wie Systemintegration derzeit in Unternehmen behandelt wird, bestätigt.

Um neue, interessante Netzwerke ausfindig zu machen, betreiben wir auch weiterhin eine intensive Suche in den Sozialen Medien. Interessante Artikel und Blogs werden geteilt.

Artikel und Blogs

Alle Daten unter einem Dach. (2018). idw Informationsdienst Wissenschaft. https://idw-online.de/de/news690173

Forschung, B. F. B. U. (2019, 4. Januar). *Daten bündeln, um Energie zu sparen*. Wissenschaftsjahr 2018: Arbeitswelten der Zukunft. https://www.wissenschaftsjahr.de/2018/neues-aus-den-arbeitswelten/alle-aktuellen-meldungen/maerz-2018/titan-will-analoge-produktionsprozesse-und-moderne-it-verbinden/

Glockmann, U. (2020). *Die Smarte Fabrik und das Internet der Dinge – Ein Industrie 4.0 Fallbeispiel*. Industrial DevOps. https://www.industrial-devops.org/die-smarte-fabrik-und-das-internet-der-dinge-ein-industrie-4-0-fallbeispiel/

Henning, S. (2019). *Titan auf der IEEE Big Data 2019*. Industrial DevOps. https://www.industrial-devops.org/ieee-big-data-2019/

Latte, B. (2020). *titan mit der IoT Welt verbinden*. Industrial DevOps. https://www.industrial-devops.org/titan-mit-der-iot-welt-verbinden/

Tofik, L. (2020). *Mobile App für das Smart Home mit No-Code entwickeln*. Industrial DevOps. https://www.industrial-devops.org/mobile-app-fuer-das-smart-home-mit-no-code-entwickeln/

Uni Kiel | Alle Daten unter einem Dach. (2018, 2. März). CAU - Christian-Albrechts-Universität zu Kiel. http://www.uni-kiel.de/pressemeldungen/index.php?pmid=2018-053-titan

Wetzel, M. (2019, 19. Mai). *No-Code Process Mining*. Industrial DevOps. https://www.industrial-devops.org/no-code-process-mining/

Wojcieszak, M. (2020a). *Daten als Schlüssel zur No-Code Programmierung*. Industrial DevOps. https://www.industrial-devops.org/daten-als-schluessel-zu-nocode/

Wojcieszak, M. (2020b). *Datenstrukturen dokumentieren*. Industrial DevOps. https://www.industrial-devops.org/datenstrukturen-dokumentieren/

Wojcieszak, M. (2020c). *Digitale Transformation in der Produktion*. Industrial DevOps. https://www.industrial-devops.org/industrial-devops-digitale-transformation-in-der-produktion/

Wojcieszak, M. (2020d, Januar 4). *Grenzen setzen*. Industrial DevOps. https://www.industrial-devops.org/grenzen-setzen/

Wojcieszak, M. (2020e, September 23). *Typen gibt's!* Industrial DevOps. https://www.industrial-devops.org/typen-gibts/

Wojcieszak, M. (2021). *No-Code Automatisierung von Prozessen mit Industrial DevOps skalieren*. wobe-systems GmbH. https://www.wobe-systems.com/no-code-automatisierung-von-prozessen-mit-industrial-devops-skalieren/

Messen und Konferenzen

- KoSSE-Tag, Kiel, Vortrag, „Industrial DevOps", 2018.
- „Agile, Testing and DevOps Showcase", Zürich, Vortrag, "Industrial DevOps"
- 6. Fachtagung zum Förderschwerpunkt „KMU-innovativ": Mittelstandskonferenz Berlin, Wobe und CAU präsentieren das titan-Projekt und Industrial DevOps als Aussteller, 19./20.11.2018
- Meetup zum Thema Industrial DevOps, Kiel, Vorstellung des titan Projekts
- Präsentation und Diskussion zu den Erfahrungen mit Clean Code im titan Projekt und der generellen Vorteile sauberen Codes für die Software Entwicklung, 17.06.2019
- Hesch Expertentag, Hamburg, Präsentation des titan Projekts, 18.06.2019
- Gastvortrag im Rahmen der Vorlesung: Softwaretechnik (Inf-ST) an der CAU Kiel, 18.06.2019
- Meilensteintreffen M2 im Fleet 7, Kiel, Vorstellung von "Flow-Thinking" mit Hilfe der Abbildung der Anforderungen des Partners Kieler Nachrichten als Flow, 12.08.2019
- 6. GI-Fachtagung "DevOps & Microsorvices" in Kiel, Präsentation der Konzepte der titan Plattform, Vorstellung der Softwareplattform und des Stands der Ergebnisse, 24.09.2019
- Workshop mit Prof. Bermbach von der TU Berlin (TUB) und dem Einstein Center Digital Future (ECDF) (Fog Computing Platforms, Cloud Service
- Benchmarking, Interdisciplinary Research), Positionierung von titan als "Edge-Computing" Komponente, 07.10.2019
- Messe IFRA 2019 Berlin, Präsentation titan Projekt, 08./09.10.2019

- Messe Meerkontakte Kiel, Präsentation titan, 23.10.2109
- Webmontag in der „Starterkitchen" in Kiel, Vorstellung des titan Projekts, 18.11.2019, https://vimeo.com/374102719/f36034a3b9
- Software Quality Days Wien, Vortrag: „Clean Code schreibt sich nicht einfach so - Ein Weg zu wartbarer und langlebiger Software“; Björn Latte, 14.01 -17.01.2020
- Mittelstandskompetenz-zentrum 4.0, Online-Vorstellung titan Projekt, 29.01.2020
- IT For Business, Messe, Lübeck, Vorstellung des titan Projekts, 05.02.2020
- Erste Industrial-DevOps Konferenz, Online, 19.11.2020:
    - Was ist Industrial DevOps?; Maik Wojcieszak; https://www.industrial-devops.org/industrial-devops-online-konferenz-2020/#indevops
    - VUCA und BANI – und die Dynamik des Wandels; Uwe Glockmann; https://www.industrial-devops.org/industrial-devops-online-konferenz-2020/#vucabani
    - Low Code und No Code Programmierung für Industrielle Anwender; Prof. Dr. Wilhelm Hasselbring; https://www.industrial-devops.org/industrial-devops-online-konferenz-2020/#lowcode
    - titan UI - Design trifft Teamwork; Muhammad Adnan; https://www.industrial-devops.org/industrial-devops-online-konferenz-2020/#titanui

Geplante Veranstaltungen

Aktuell sind keine konkreten Veranstaltungen geplant. Wir warten aktuell den Verlauf der Corona-Pandemie ab und konzentrieren und auf die sozialen Medien. Folgende Veranstaltungen sind jedoch angedacht:

- Member of Open Industry 4.0 Alliance
- Messeauftritt mit unserem Partner Höntzsch
- Messeauftritt mit unserem Partner Frank Höppke – Haus & Technik
- Industrial DevOps Summit – Fachmesse für Industrieanwender mit offenem CfP
- No-Code Hackathon-Event – Digitale Woche Kiel

Studentische Abschlussarbeiten

J. R. Bensien, “Scalability Benchmarking of Stream Processing Engines with Apache Beam”, Bachelorarbeit, 2021. http://eprints.uni-kiel.de/52342/

J. Grabitzky, “A Showcase for the titan Control Center”, Bachelorarbeit, 2021. http://eprints.uni-kiel.de/52341/

C. T. Tsida, “Analyzing Environmental Data with the titan Platform”, Masterarbeit, 2020. http://eprints.uni-kiel.de/51036/

L. Tofik, „Aufbau und Entwicklung eines Messe Demonstrators für eine Low Code IoT Plattform für Systemintegration”, Bachelorarbeit, 2020. http://eprints.uni-kiel.de/51041/

T. Koch, “Scalable and Interactive Real Time Visualization of Time Series Data”, Bachelorarbeit, 2020. http://eprints.uni-kiel.de/50728/1/

N. Biernat, “Scalability Benchmarking of Apache Flink”, Bachelorarbeit, 2020.

L. Boguhn, “Forecasting Power Consumption of Manufacturing Industries Using Neural Networks”, Bachelorarbeit, 2020. http://eprints.uni-kiel.de/49521/

A. Hansen, “Exploring an Energy Status Data Set from Industrial Production”, Bachelorarbeit, 2019. http://eprints.uni-kiel.de/48018/

B. Wetzel, „Entwicklung eines Dashboards für eine Industrial DevOps Monitoring Plattform”, Bachelorarbeit, 2019. http://eprints.uni-kiel.de/48019/

S. Ehrenstein, “Distributed Sensor Management for an Industrial DevOps Monitoring Platform”, Bachelorarbeit, 2019. http://eprints.uni-kiel.de/48020/

M. Taleghani Esfahani, “Design and Evaluation of Touch Interactions in a No-Code - Programming User Interface Using HTML5 Canvas”, Masterarbeit, 2019. https://www.researchgate.net/project/Design-and-Evaluate-of-Touch-Interactions-in-a-No-Code-programming-User-Interface-Using-HTML5-Canvas

P. Murarka, “Simulating Actor Based Execution of Flow Models in Cloud Environments”, Masterarbeit, 2018.

S. Henning, “Prototype of a Scalable Monitoring Infrastructure for Industrial DevOps Masterarbeit“, Masterarbeit, 2018. http://eprints.uni-kiel.de/43910 (Die Arbeit wurde für ihre wissenschaftliche Qualität und die industriepraktische Relevanz mit dem b+m Software + Systems Engineering Preis 2018 ausgezeichnet.)

Begutachtete Veröffentlichungen und Konferenzbeiträge

[Hen18] S. Henning, “Monitoring Electrical Power Consumption with Kieker”, Softwaretechnik-Trends, 39 (3), 2019 (Symposium on Software Performance, Hildesheim, 2018). https://pi.informatik.uni-siegen.de/stt/39_3/01_Fachgruppenberichte/SSP18/Henning18.pdf

[HH19] S. Henning, W. Hasselbring, “Scalable and Reliable Multi-Dimensional Aggregation of Sensor Data Streams”, IEEE International Conference on Big Data, Los Angeles, USA, 2019. DOI: https://doi.org/10.1109/BigData47090.2019.9006452

[HH20a] S. Henning, W. Hasselbring, “Scalable and Reliable Multi-dimensional Sensor Data Aggregation in Data Streaming Architectures”, Data-Enabled Discovery and Applications, Springer, 4, 2020. DOI: https://doi.org/10.1007/s41688-020-00041-3

[HH20b] S. Henning, W. Hasselbring, “Toward Efficient Scalability Benchmarking of Event-Driven Microservice Architectures at Large Scale”, Softwaretechnik-Trends, 40 (3), 2020 (Symposium on Software Performance, online, 2020).

https://pi.informatik.uni-siegen.de/stt/40_3/01_Fachgruppenberichte/SSP2019/SSP2020_Henning.pdf

[HH21a] S. Henning, W. Hasselbring, “The titan Control Center for Industrial DevOps analytics research”, Software Impacts, Elsevier, 7, 2021. DOI: https://doi.org/10.1016/j.simpa.2020.100050

[HH21b] S. Henning, W. Hasselbring, “Theodolite: Scalability Benchmarking of Distributed Stream Processing Engines in Microservice Architectures”, Big Data Research, Elsevier, 25, 2021. DOI: https://doi.org/10.1016/j.bdr.2021.100209

[HH21c] S. Henning, W. Hasselbring, “How to Measure Scalability of Distributed Stream Processing Engines?”, ACM/SPEC International Conference on Performance Engineering, online, 2021. DOI: https://doi.org/10.1145/3447545.3451190

[HHB+21] S. Henning, W. Hasselbring, H. Burmester, A. Möbius, M. Wojcieszak Henning, “Goals and Measures for Analyzing Power Consumption Data in Manufacturing Enterprises”, Journal of Data, Information and Management, Springer, 3, 2021. DOI: https://doi.org/10.1007/s42488-021-00043-5

[HHL+19] W. Hasselbring, S. Henning, B. Latte, A. Möbius, T. Richter, S. Schalk, M. Wojcieszak, “Industrial DevOps”, IEEE International Conference on Software Architecture, Hamburg, 2019. DOI: https://doi.org/10.1109/ICSA-C.2019.00029

[HHM19] S. Henning, W. Hasselbring, A. Möbius, “A Scalable Architecture for Power Consumption Monitoring in Industrial Production Environments”, IEEE International Conference on Fog Computing, Prag, Tschechien, 2019. DOI: https://doi.org/10.1109/ICFC.2019.00024

[HWD21] W. Hasselbring, M. Wojcieszak, S. Dustdar, “Control Flow vs. Data Flow in Distributed Systems Integration”, IEEE Internet Computing, July/August 2021. DOI: http://dx.doi.org/10.1109/MIC.2021.3053712

[LHW19] B. Latte, S. Henning, M. Wojcieszak, “Clean Code: On the Use of Practices and Tools to Produce Maintainable Code for Long- Living Software”, Collaborative Workshop on Evolution and Maintenance of Long-Living Systems, Stuttgart, 2019. http://ceur-ws.org/Vol-2308/emls2019paper01.pdf