Plattform Automations- und Integrationsarchitektur mit Ansible (Tower)

ein Vortrag der Gothaer IT

18.05.2021 | Köln | Alexander Prinz

Die Gothaer
Starke Marke mit positivem Image

Breite Aufstellung für eine ganzheitliche Sicht auf den Kunden

4,1 Mio. Kunden

4,4 Mrd. € gebuchte Beitrags-einnahmen

Platz 12 im deutschen Versicherungs-markt

4.744 Mitarbeiter

Marktführer bei innovativen Themen wie Erneuerbare Energien

Einer der fünf größten Versicherungsvereine in Deutschland

Gothaer feiert 200-jähriges Jubiläum! 200 Jahre Gothaer: #wirsindgothaer

Stand: 2020

Die Gothaer Systems GmbH
Wir sind die IT des Gothaer Konzerns

Wir treiben den digitalen Wandel im Konzern

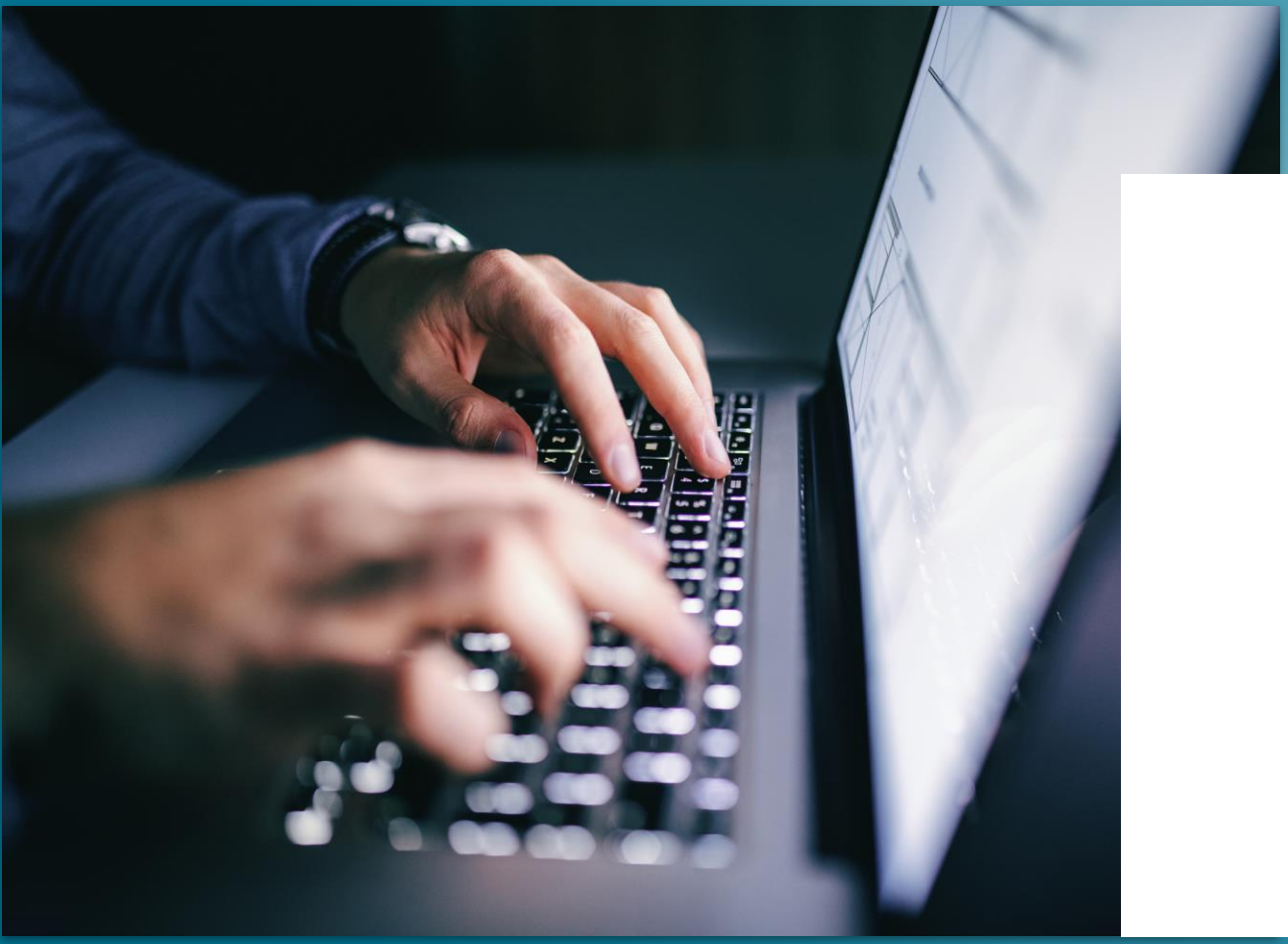
Digitale Transformation – wir machen das!

- 660 Mitarbeitende | 29 dual Studierende
- Wir stellen die gesamte IT-Infrastruktur- und Services im Konzern bereit: OnPremise und in der Cloud
- Wir entwickeln und integrieren neue und bestehende Anwendungssysteme und managen diese im laufenden Betrieb
- Wir beraten und betreuen die Gothaer Fachbereiche

Und: Wir bieten Top Karrierechancen für IT-Spezialisten

TOP KARRIERE CHANCEN
Focus Money

Die Gothaer Systems GmbH
Linux Platform Services



Ein paar Fakten …

zur Linux Plattform

- 12 Mitarbeiter
- Virtualisierungsgrad von 100%
- 800 Server
- 2.500 Cores
- 23 TB RAM
- 1.2 PB AllFlash Storage
- SelfService im Warenkorbmodell

Plattform Automations- und Integrationsarchitektur mit Ansible (Tower)

Gothaer

Baukastenprinzip

Regulatorik

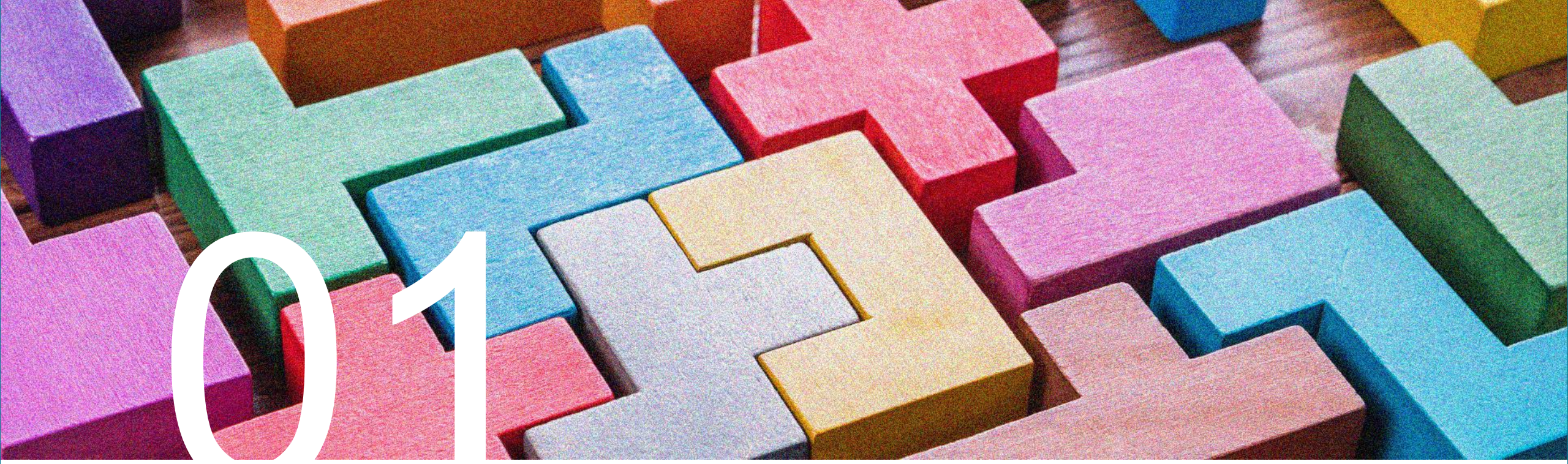
Bodenplatte

IaC Ansätze

Klassische Ansätze

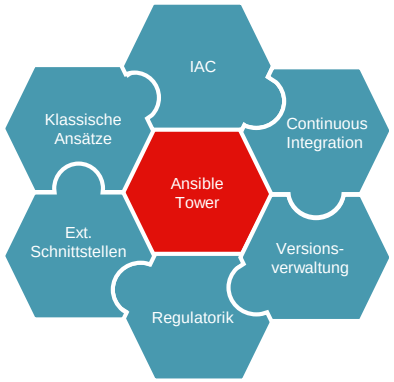
Metadaten & Tags

Big Picture



01

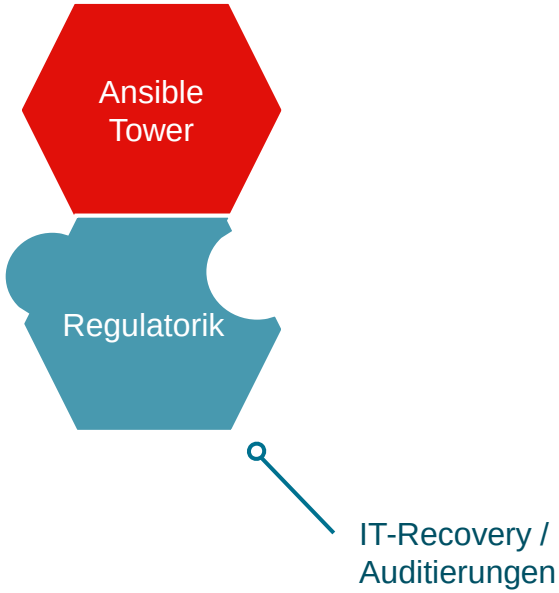
Baukastenprinzip



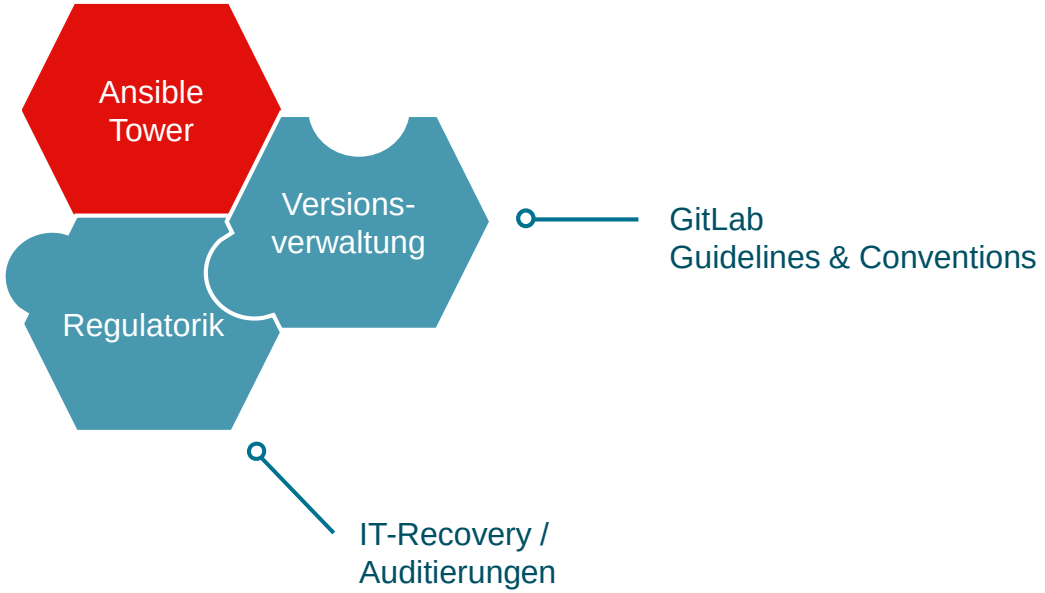
Gothaer

01 – Baukastenprinzip

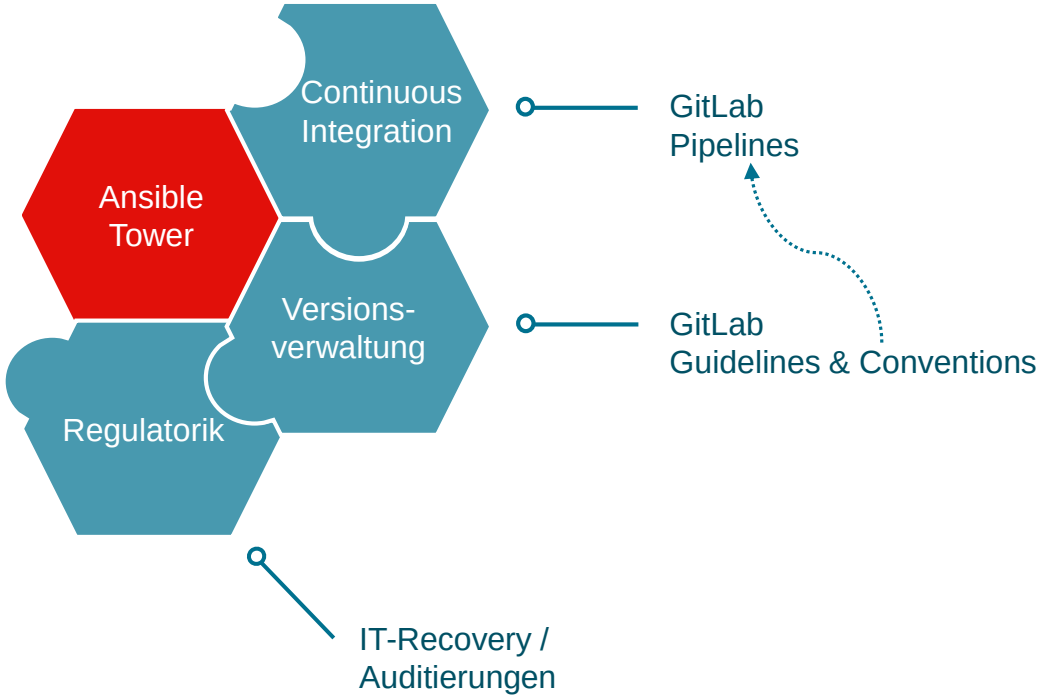
01 – Baukastenprinzip



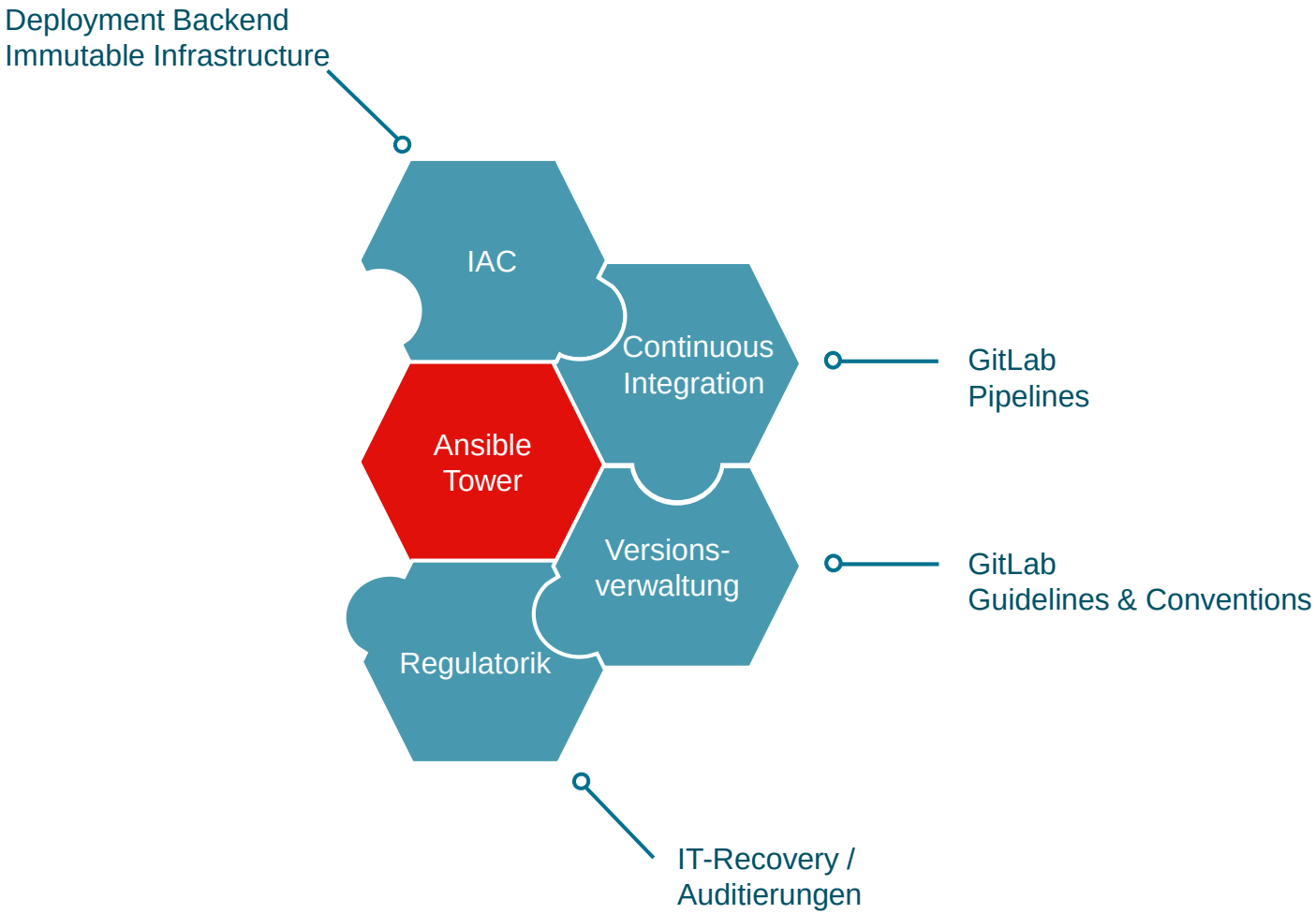
01 – Baukastenprinzip



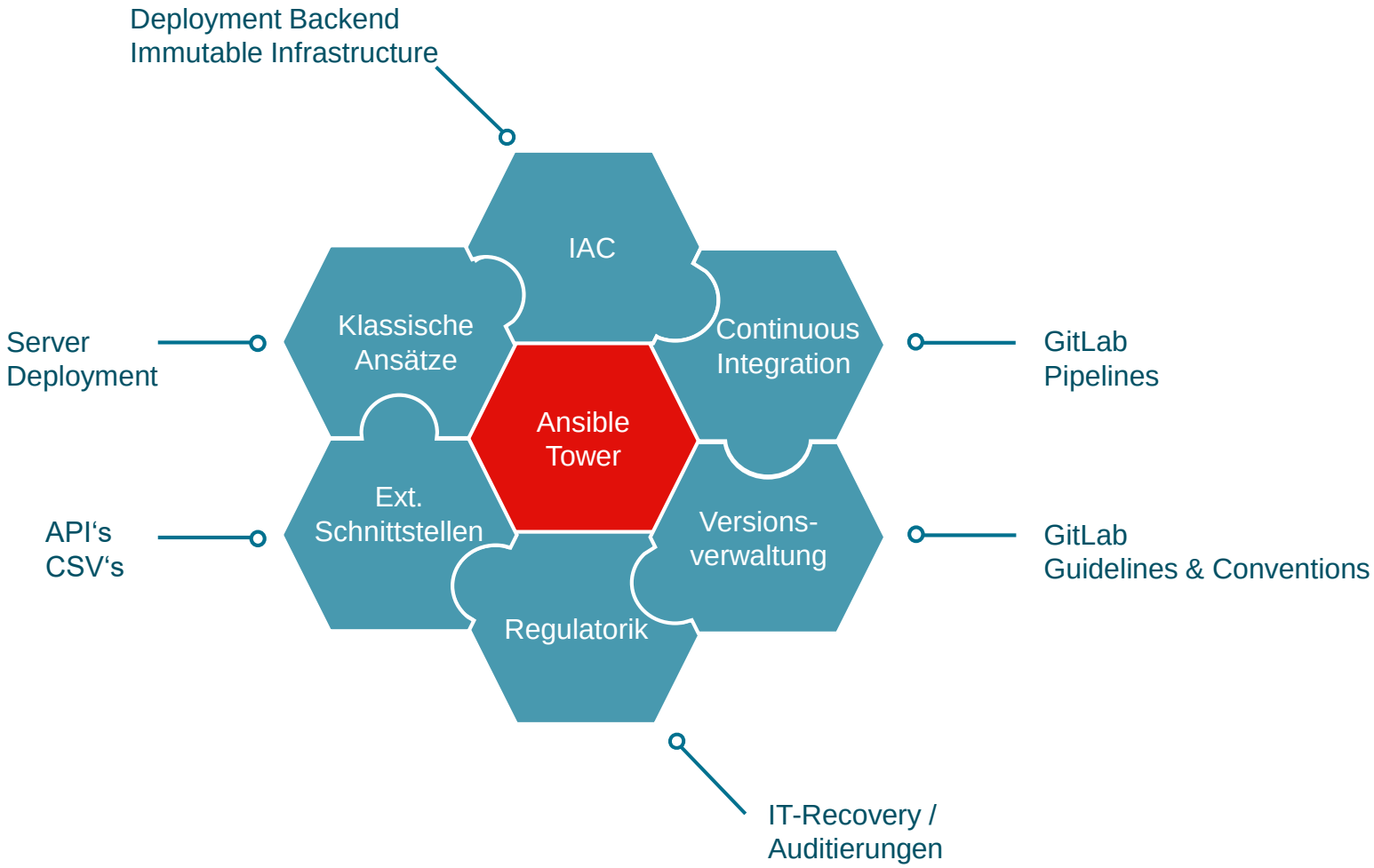
01 – Baukastenprinzip



01 – Baukastenprinzip

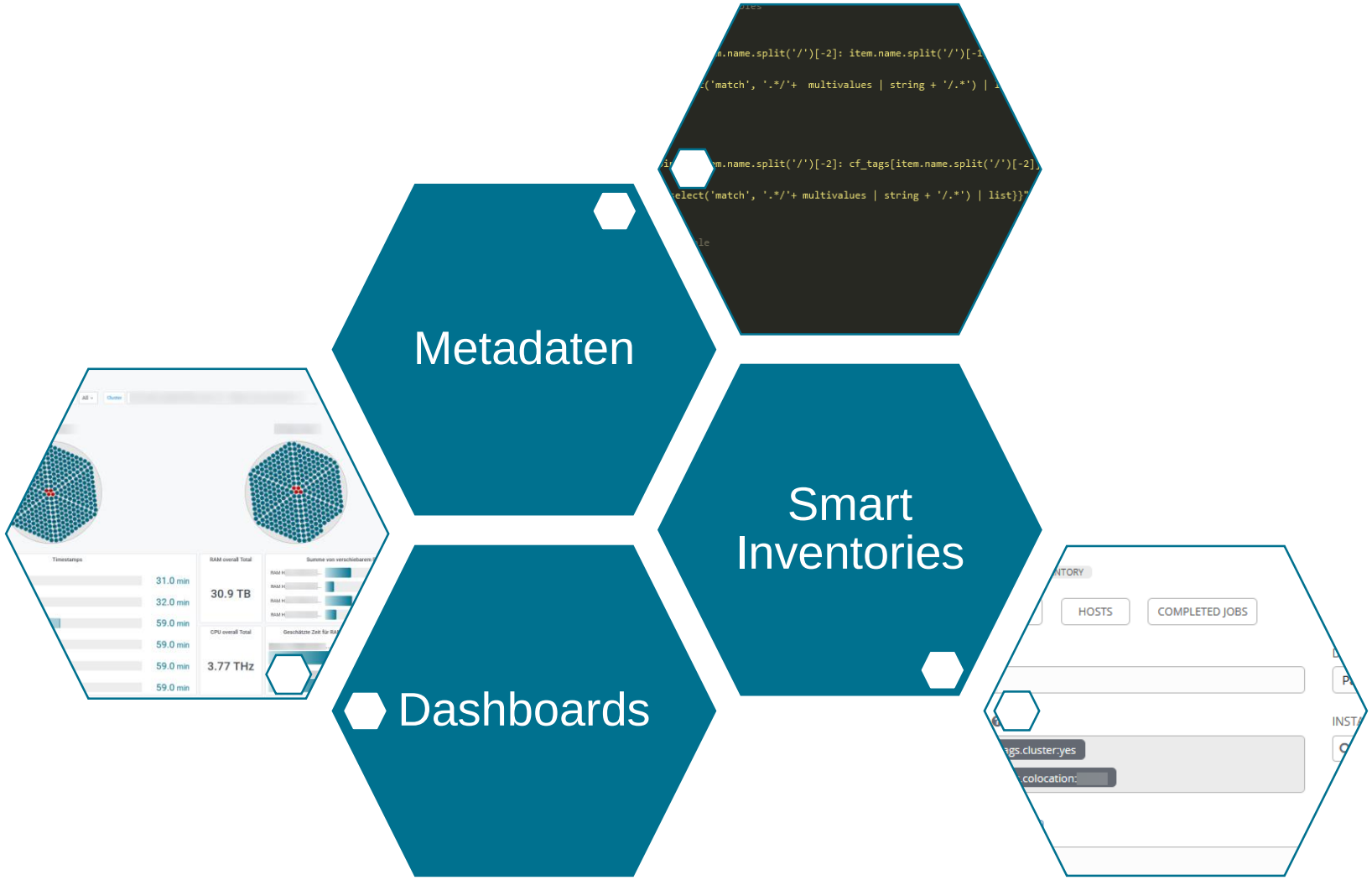


01 – Baukastenprinzip



02

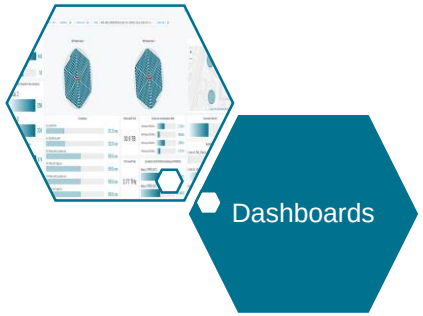
Regulatorik


Metadaten
Smart Inventories
Dashboards
HOSTS
COMPLETED JOBS
30.9 TB
3.77 THz

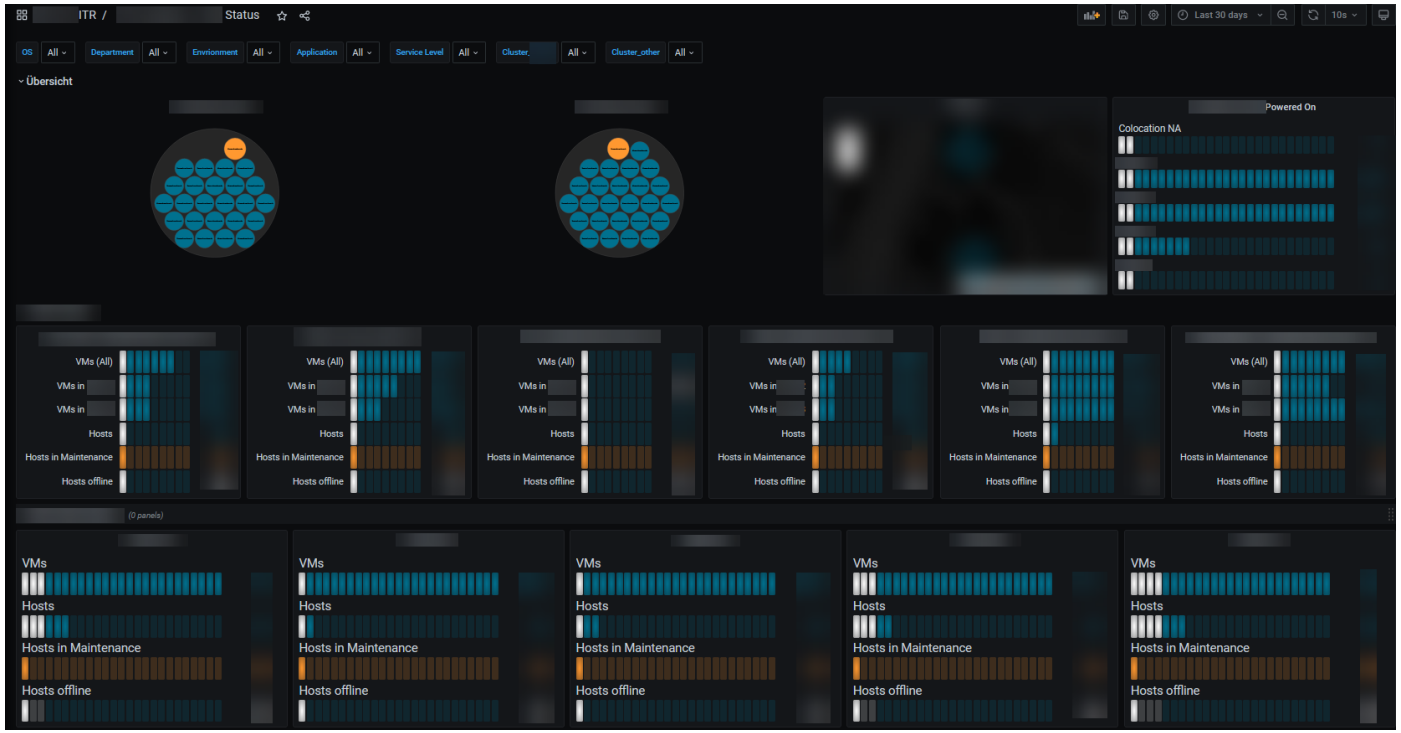
- Zentrale Datenbank
- Eigene Hoheit
- Maximale Flexibilität
- Single Point of Truth
- Skalierendes Konzept auf Basis von Tags

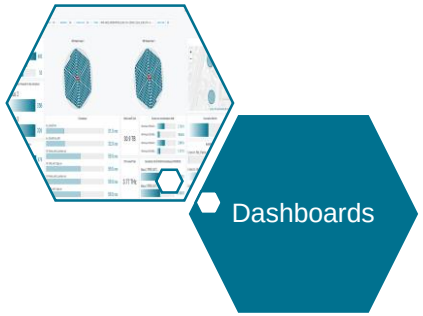
```
    ### Convert only the tag section of the cloudforms variables
    - name: Convert Custom Tags - Single Tags
      set_fact:
        cf_tags: "{{ cf_tags|default({}) | combine( {item.name.split('/')[-2]: item.name.split('/')[-1]}) }}"
        cacheable: yes
      with_list: "{{ cloudforms.tags | unique | reject('match', '.*/'+  multivalues | string + '/.*') | list}}"
      when: cloudforms.tags is defined

    - name: Convert Custom Tags - Mutli Tags
      set_fact:
        cf_tags: "{{ cf_tags|default({}) | combine( {item.name.split('/')[-2]: cf_tags[item.name.split('/')[-2]]|default([]) + [item.name.split('/')[-1]]}) }}"
        cacheable: yes
      with_list: "{{ cloudforms.tags | unique | select('match', '.*/'+ multivalues | string + '/.*') | list}}"
      when: cloudforms.tags is defined

    ### In addation convert a single preferred variable
    - name: Convert Power State
      set_fact:
        power_state: "{{ cloudforms.power_state }}"
        cacheable: yes
      when: cloudforms.power_state is defined
```

02 – Regulatorik

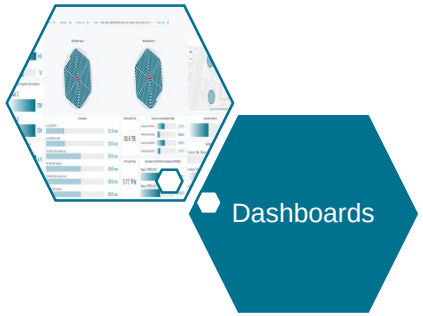


- Visualisierung anhand von Dashboards
- Daten in Echtzeit
- Kriterien auf Basis von
  - Metadaten (Tags)
  - Facts
  - SQL Selects
  - SQL Views

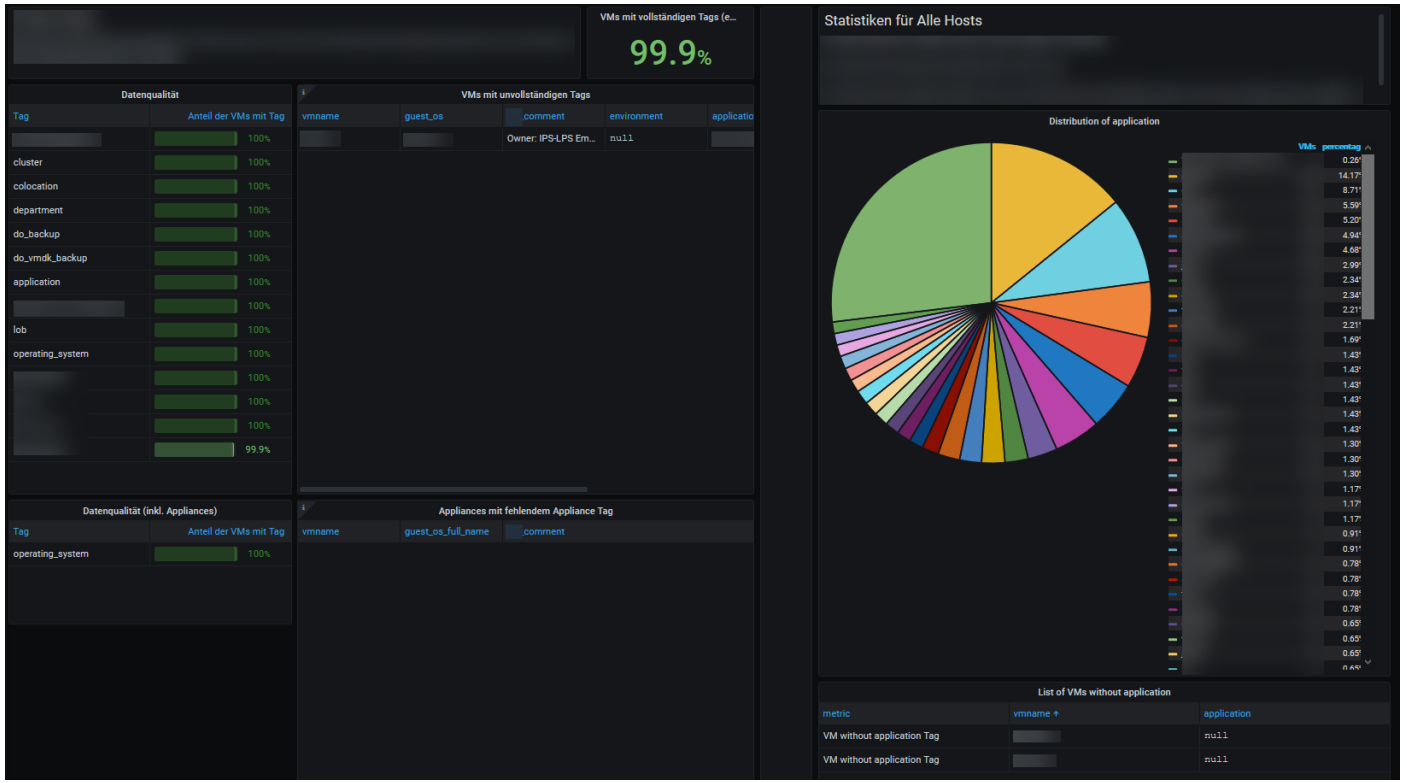




- Visualisierung anhand von Dashboards
- Daten in Echtzeit
- Kriterien auf Basis von
  - Metadaten (Tags)
  - Facts
  - SQL Selects
  - SQL Views



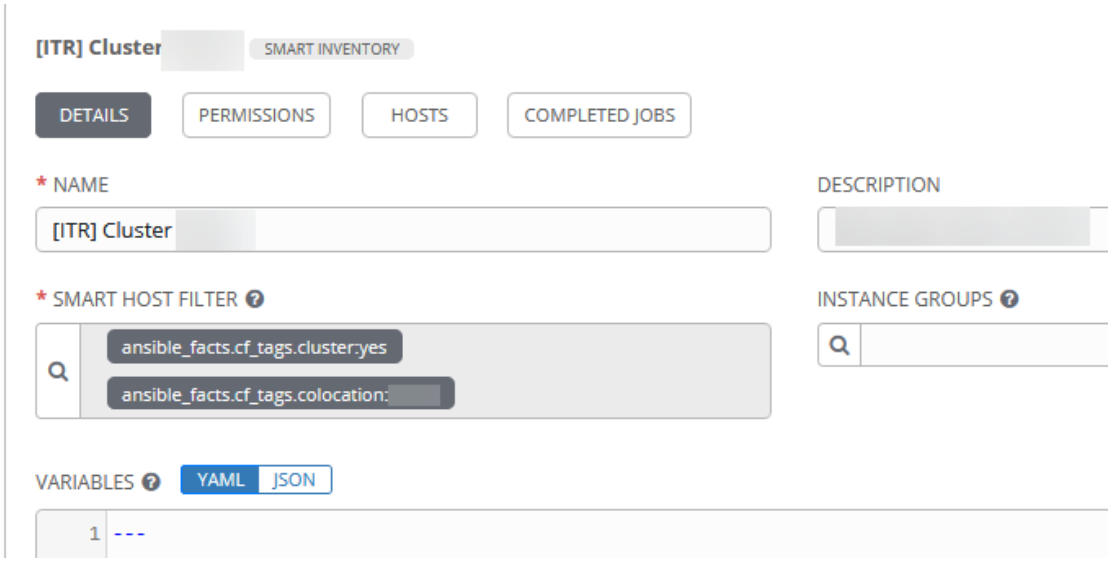
- Visualisierung anhand von Dashboards
- Daten in Echtzeit
- Kriterien auf Basis von
  - Metadaten (Tags)
  - Facts
  - SQL Selects
  - SQL Views



Gothaer

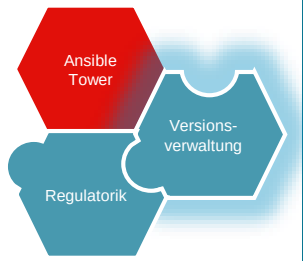
Smart Inventories

- Facts basierend auf Tags
- Dyn. Inventories auf Basis von Facts
- Max. Flexibilität auf Basis des Tagging Konzepts



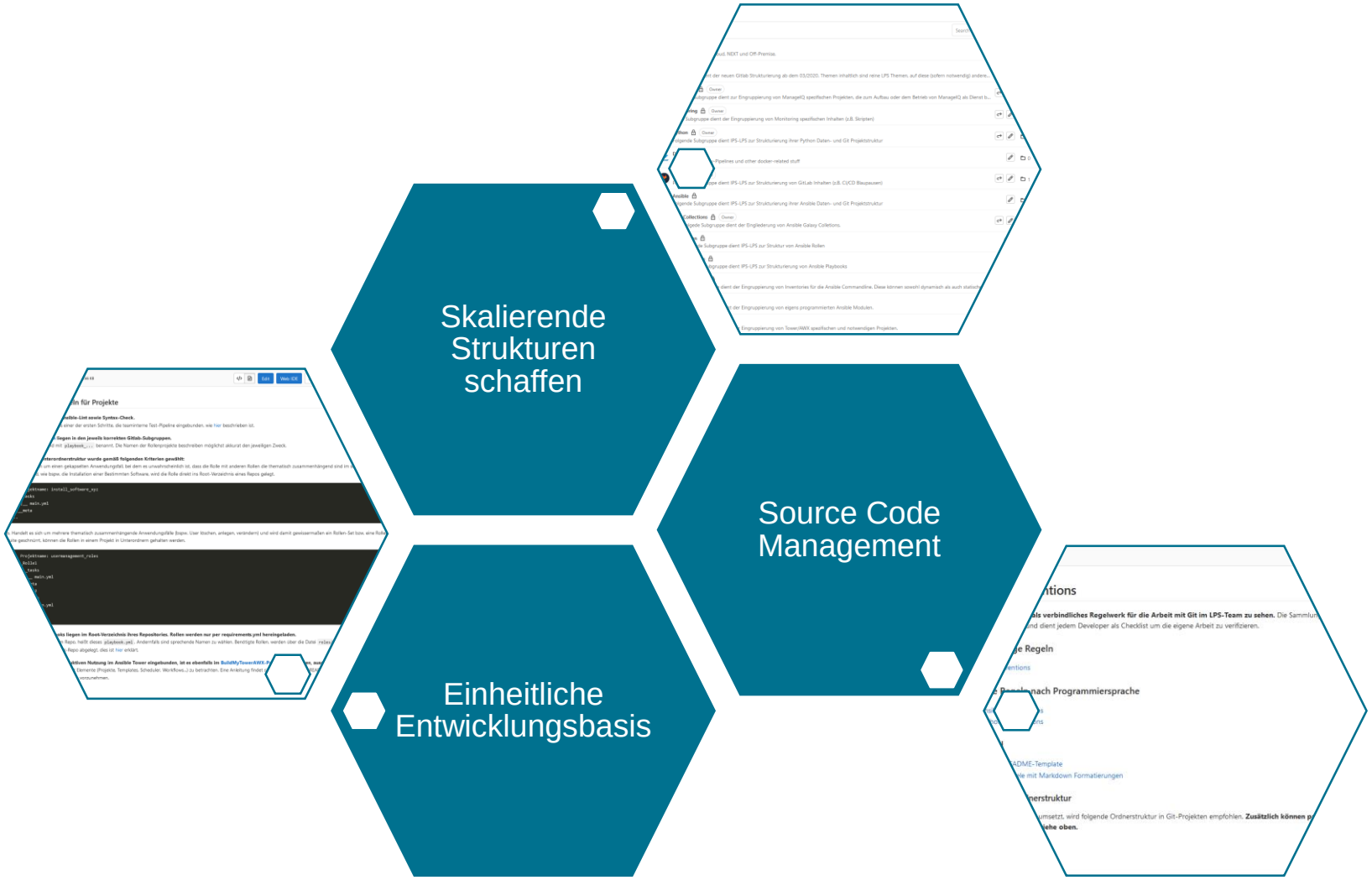
03

Bodenplatte



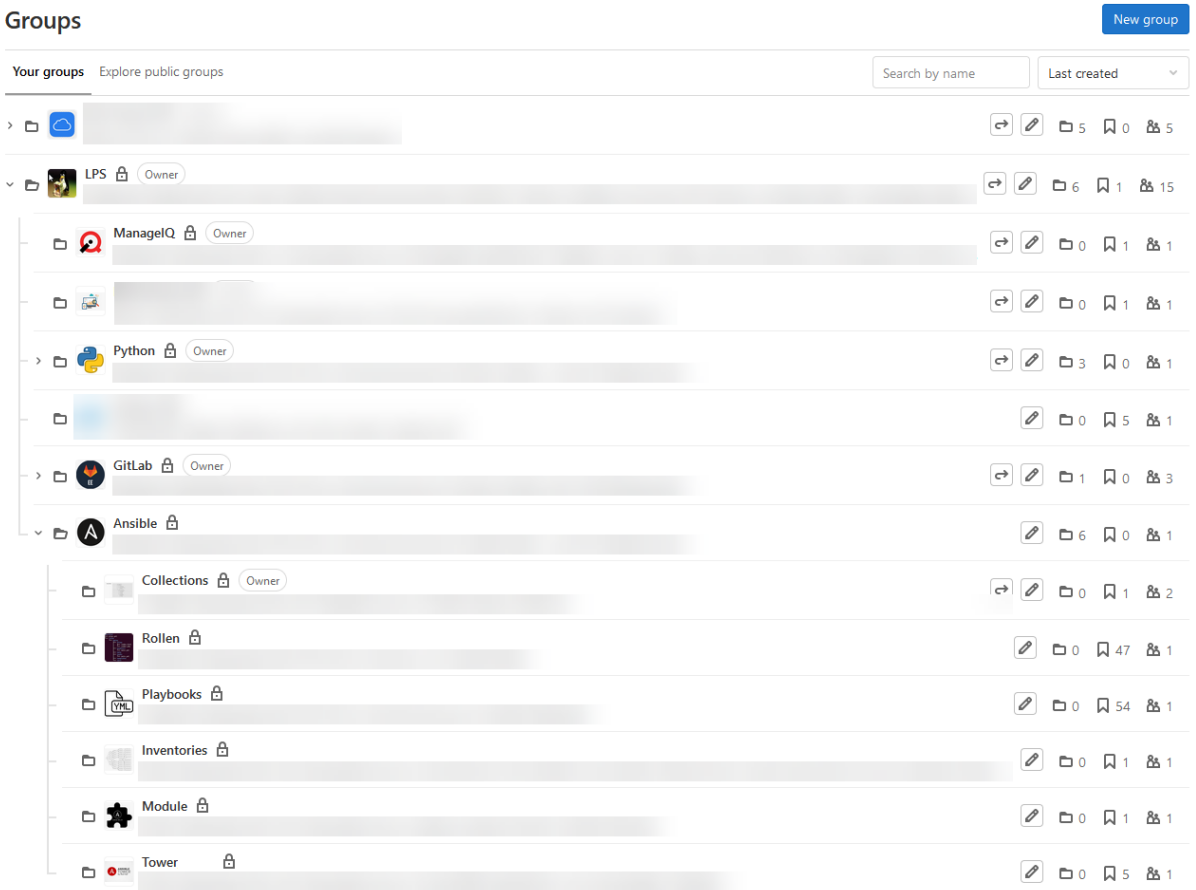
Gothaer

03 – Bodenplatte

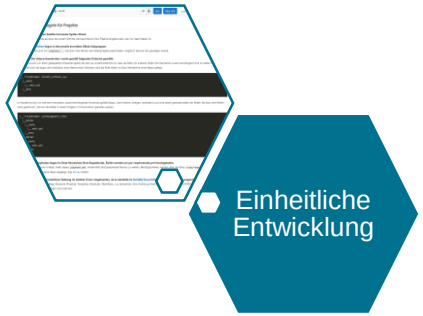


03 – Bodenplatte

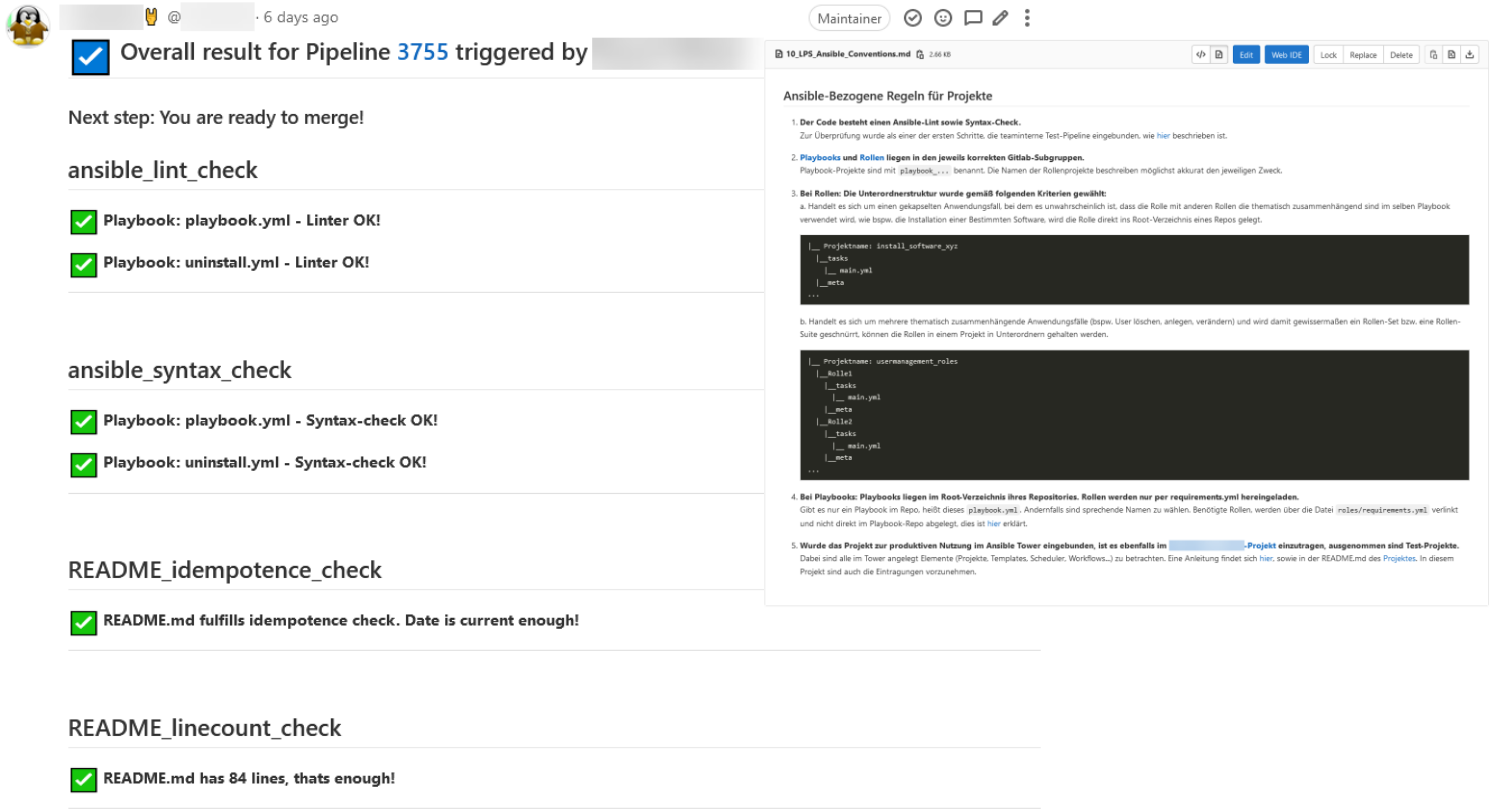
- Klar definierte Struktur
- Regelungen bzgl.
  - Ablage
  - Informationen
  - Dokumentation
- Ausgelegt auf Wiederverwendbarkeit
- Indizierte Codesuche



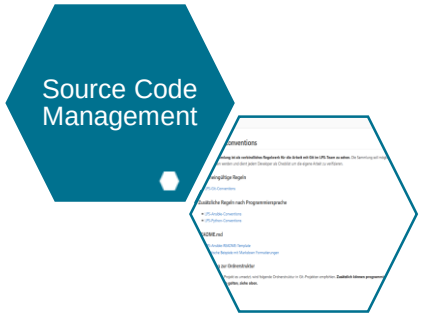
03 – Bodenplatte



- Definiertes Regelwerk
- Einheitliches Vorgehen
- Abgeleitete, automatische Checks
- Jederzeit erweiterbar
- Zentrale Ansätze (greift für alle Projekte)
  - Regelset‘s
  - Pipelines



03 – Bodenplatte



- Definiertes Regelwerk
- Einheitliches Vorgehen
- Nachvollziehbarkeit auf Basis von Commits
- Übergeordnete Policies (zentr. Ansätze)
  - 4 Augen Prinzip (Approver)
  - Änderungen lediglich durch Merges

README.md

LPS Code Conventions

Allgemeingültige Regeln

- LPS-Git-Conventions

Zusätzliche Regeln nach Programmiersprache

- LPS-Ansible-Conventions
- LPS-Python-Conventions

README.md

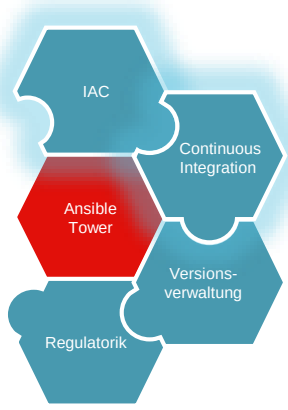
- LPS-Ansible-README-Template
- Praktische Beispiele mit Markdown Formatierungen

Empfehlung zur Ordnerstruktur

▸ Ordnerstruktur

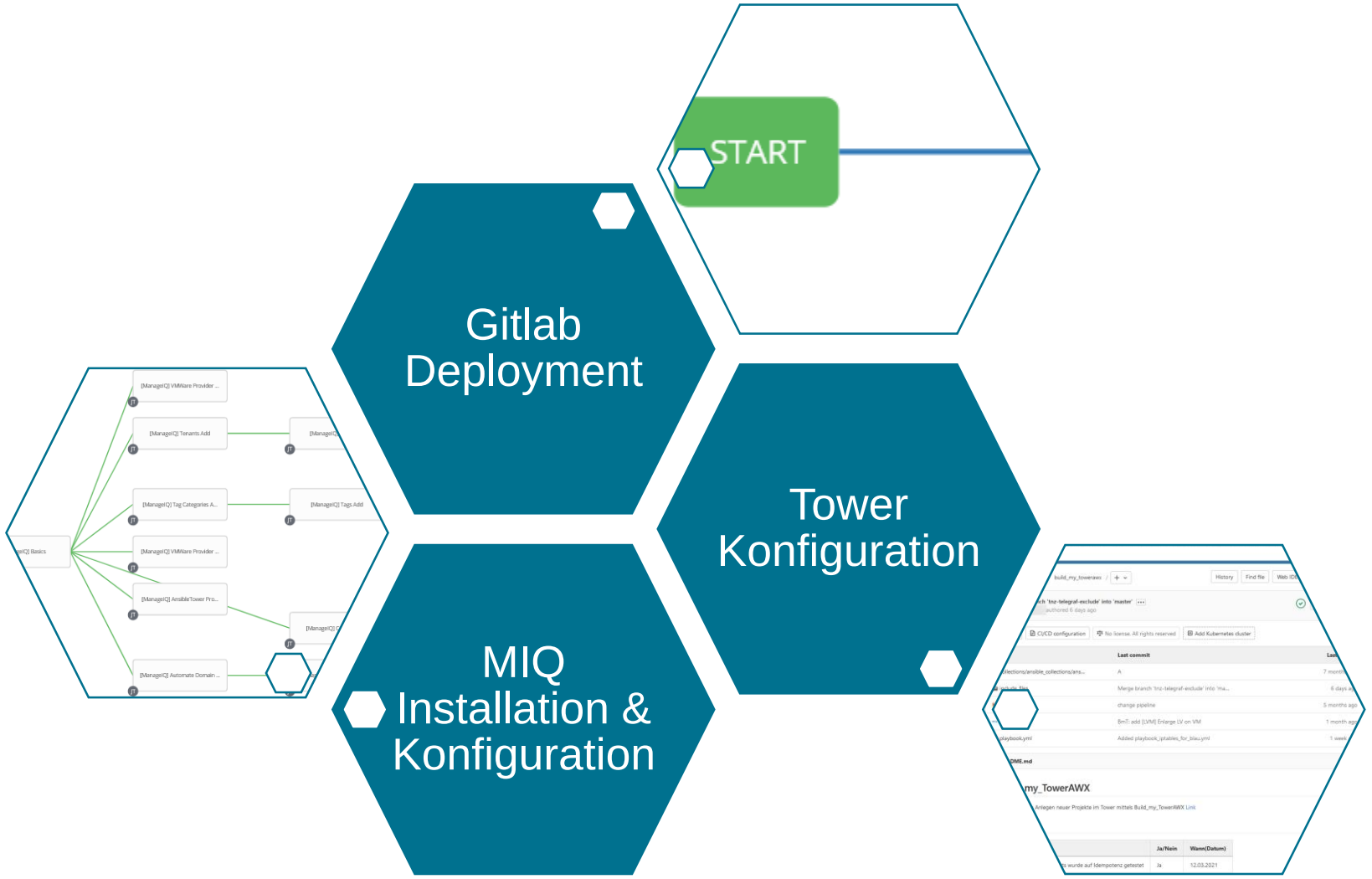
04

IaC – Infrastructe as Code & Immutable Infrastructure

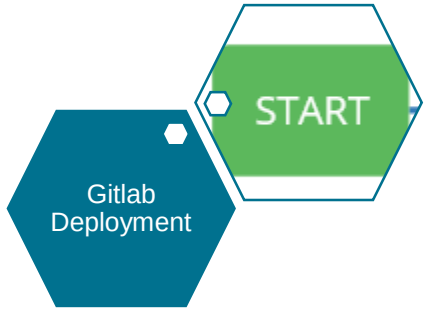


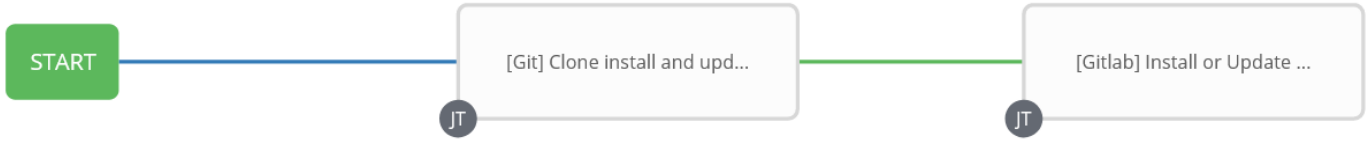
Gothaer

04 – IaC & Immutable Infrastructure



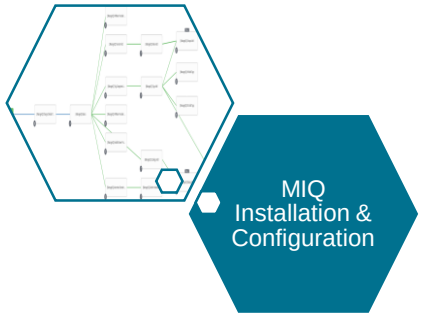
04 – IaC & Immutable Infrastructure





- Henne Ei- Problem
- Aufbau von Gitlab auf Basis von Ansible
- Konfiguration auf Basis von YaML
- Simples Beispiel (o. Abhängigkeiten)

04 – IaC & Immutable Infrastructure



- Installation & Konfiguration des Backends
- Auf Basis von Infrastructure as Code
- Konfiguration via YaML oder anhand von API Call‘s
- Aufbau & Initialisierung 10min
- Konfiguration & Synchronisation 45min

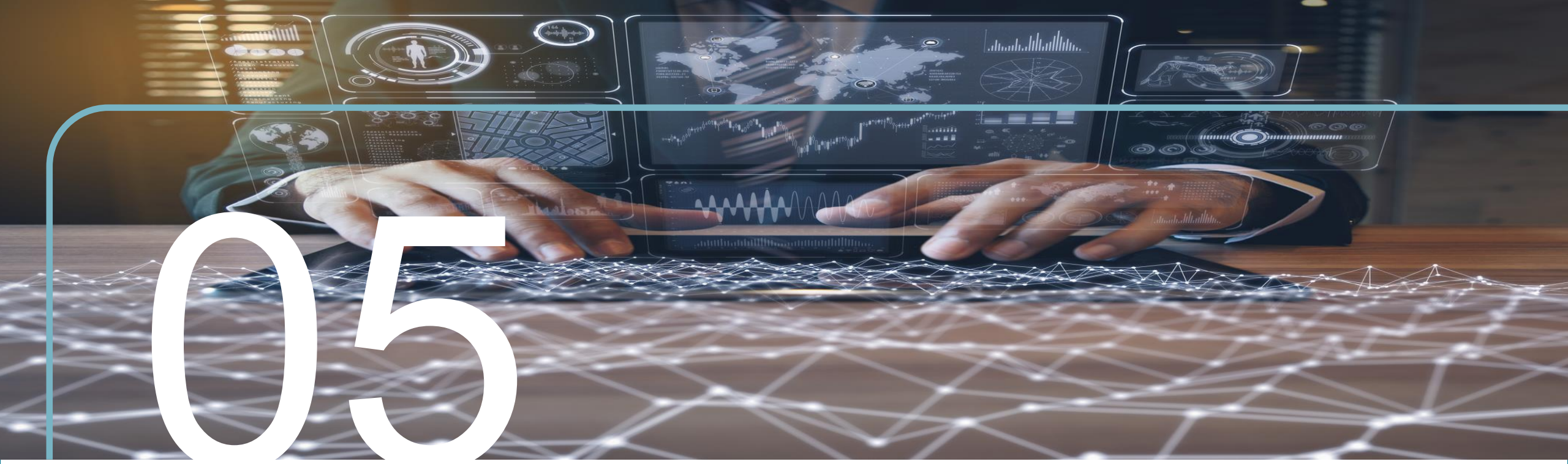
04 – IaC & Immutable Infrastructure

Gothaer

Tower Configuration

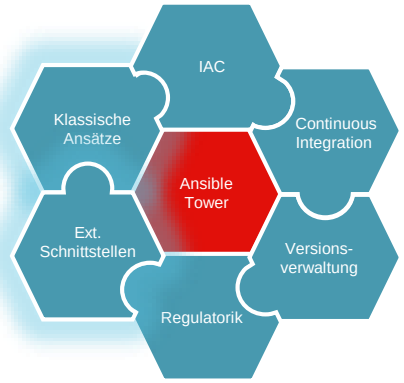
- Konfiguration via Ansible
- IaC Pipeline als Wrapper
- Historische Daten nicht zwingend relevant
  - Optional: Backup
- Ausblick: Version 2.0 mit zugrunde liegenden Informationen in Diskussion

```
TASK [Create workflow nodes 7] *****************************************************
ok: [        ]
TASK [Create workflow nodes 6] *****************************************************
ok: [        ]
TASK [Create workflow nodes 5] *****************************************************
ok: [        ]
TASK [Create workflow nodes 4] *****************************************************
ok: [        ]
TASK [Create workflow nodes 3] *****************************************************
ok: [        ]
TASK [Create workflow nodes 2] *****************************************************
ok: [        ]
TASK [Create workflow nodes 1] *****************************************************
ok: [        ]
TASK [include_tasks] ***************************************************************
included:              /builds/NsTYxzPz/0/lps/ansible/tower                    'include_files/Finalize_Workflow/apicalls_finalize.yml for [      
TASK [Add Tower Project APICalls Finalize Deployment] ******************************
ok: [        ]
TASK [Add Tower Job template APICalls Finalize Deployment] *************************
ok: [        ]
TASK [Add Tower Job template APICalls Finalize Deployment] *************************
ok: [        ]
TASK [Give Executer Access] ********************************************************
ok: [        ]
TASK [include_tasks] ***************************************************************
included:              /builds/NsTYxzPz/0/lps/ansible/tower                    'include_files/Finalize_Workflow/workflow_finalize.yml for [      
TASK [Create Finalize Workflow] ****************************************************
[WARNING]: The value 63 (type int) in a string field was converted to u'63'
(type string). If this does not look like what you expect, quote the entire
value to ensure it does not change.
ok: [        ]
TASK [Give Admin Access] ***********************************************************
ok: [        ]
TASK [Create workflow nodes] *******************************************************
ok: [        ] => (item=[
ok: [        ] => (item=[
ok: [        ] => (item=[
ok: [        ] => (item=[
ok: [        ] => (item=[
ok: [        ] => (item=[
ok: [        ] => (item=[
```



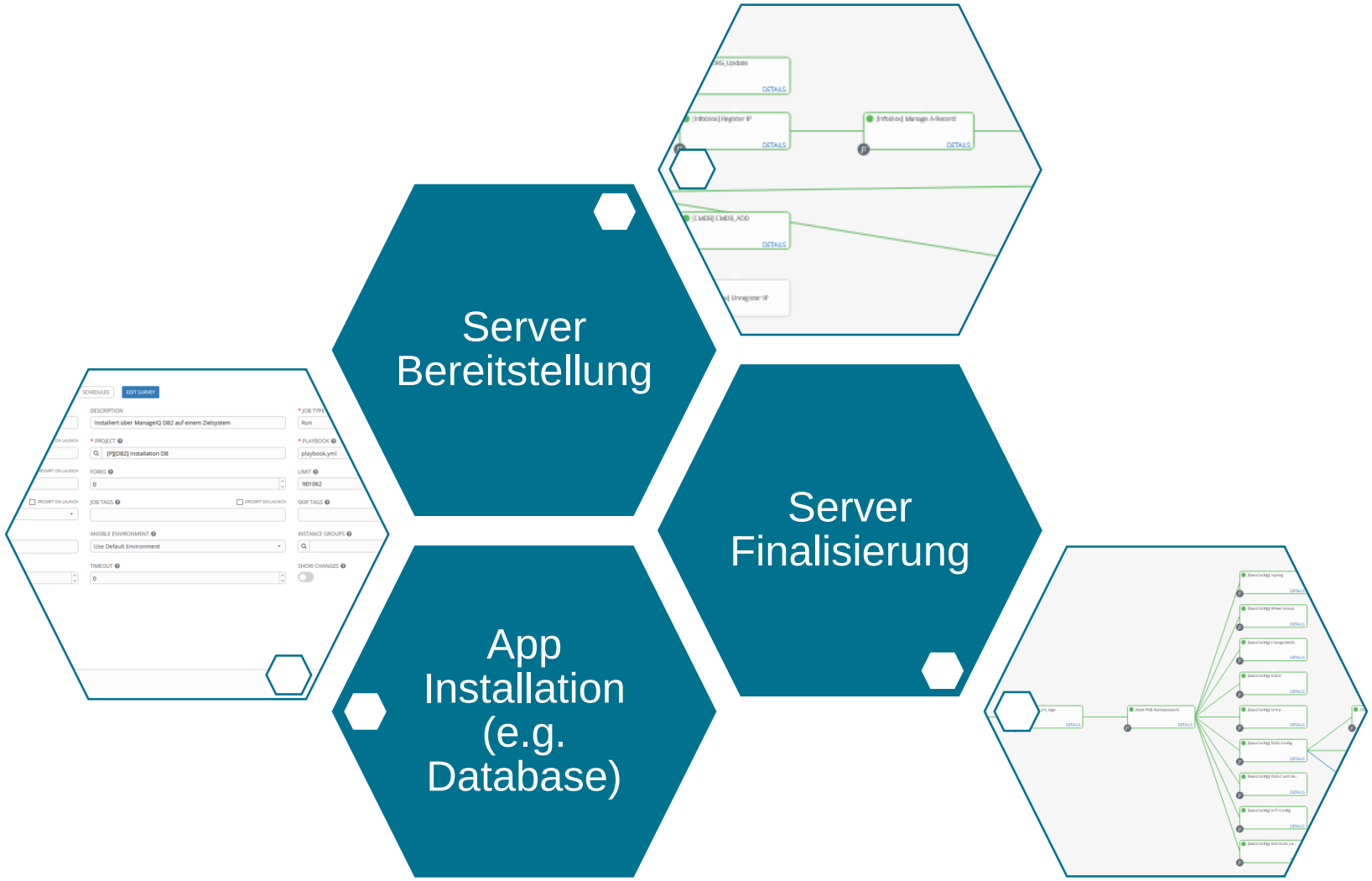
05

Klassische Ansätze

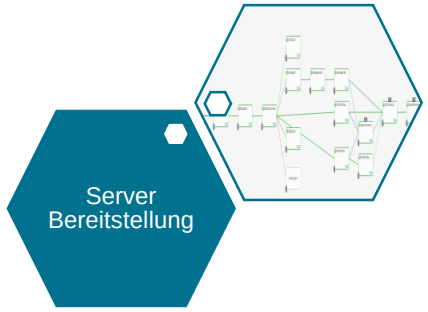


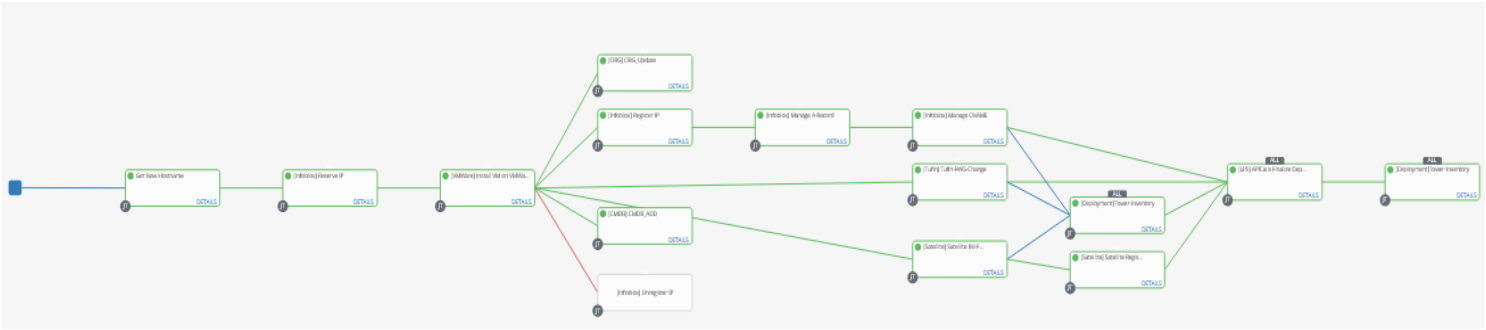
Gothaer

05 – Klassische Ansätze



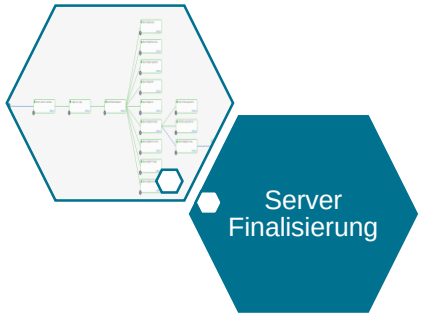
05 – Klassische Ansätze



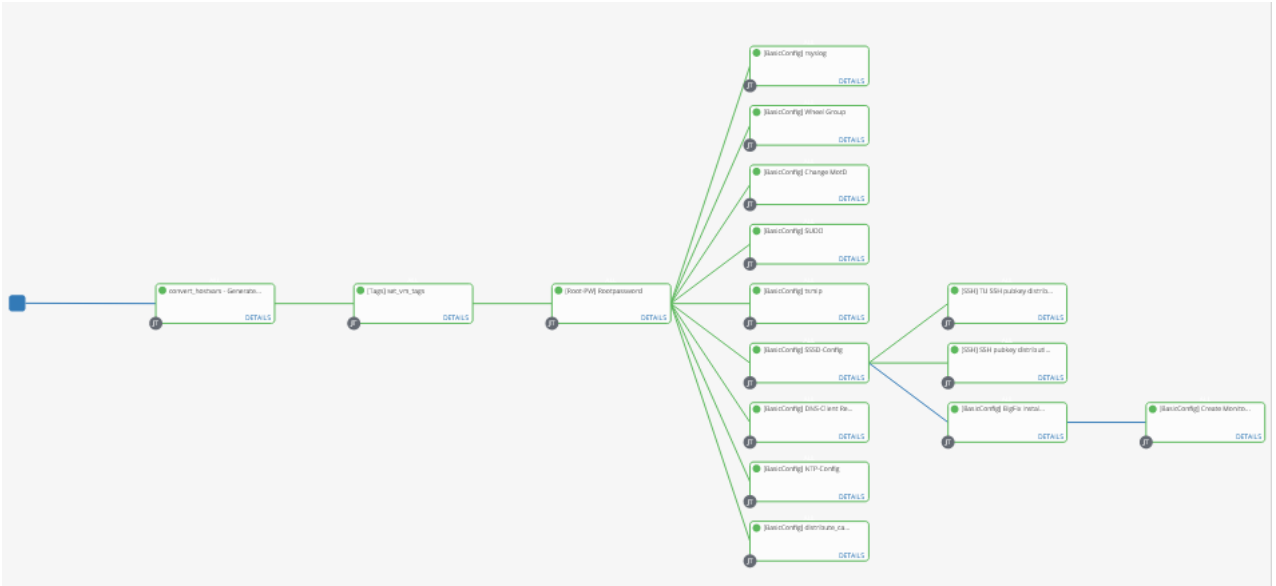


- Deployment auf Basis von Infrastructure as Code
- Nutzung von Workflows
- Abbildung von Abhängigkeiten & Logiken
- Hochgradig modular / jederzeit erweiterbar
- Qualitätssteigerung durch autom. Checks

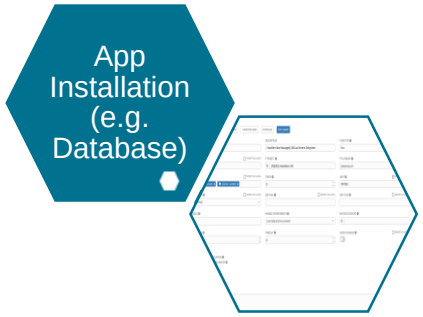
05 – Klassische Ansätze



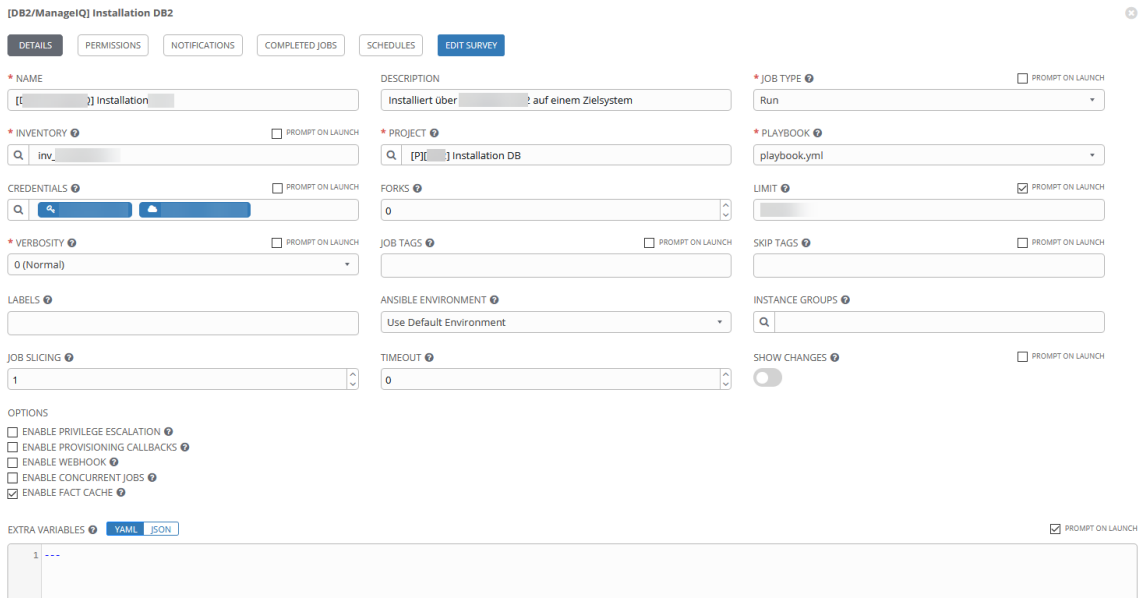
- Finalisierung auf Basis von Infrastructure as Code
- Nutzung von Workflows
- Abbildung von Abhängigkeiten & Logiken
- Hochgradig modular / jederzeit erweiterbar
- Qualitätssteigerung durch autom. Checks

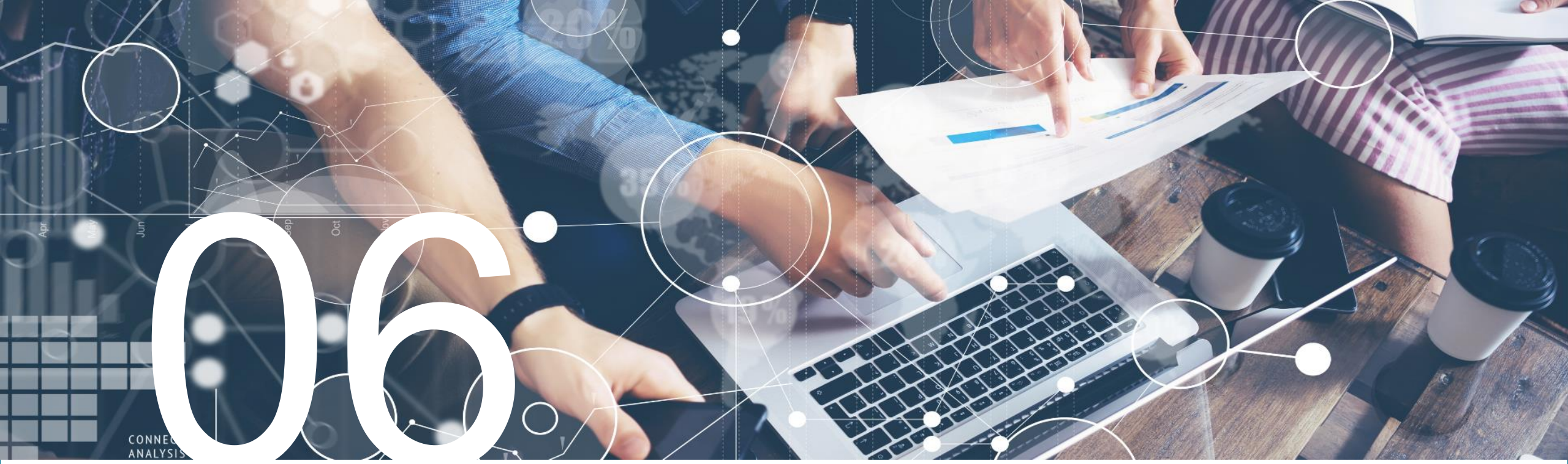


05 – Klassische Ansätze



- Installation & Konfiguration via Ansible
- Trigger auf Basis von API Calls
- SelfSerivce
  - Stellt Oberfläche zur Verfügung
  - Sammelt Übergabeparameter
  - Bietet Warenkorbposition an
  - Triggert Workflow(s) (/Templates)

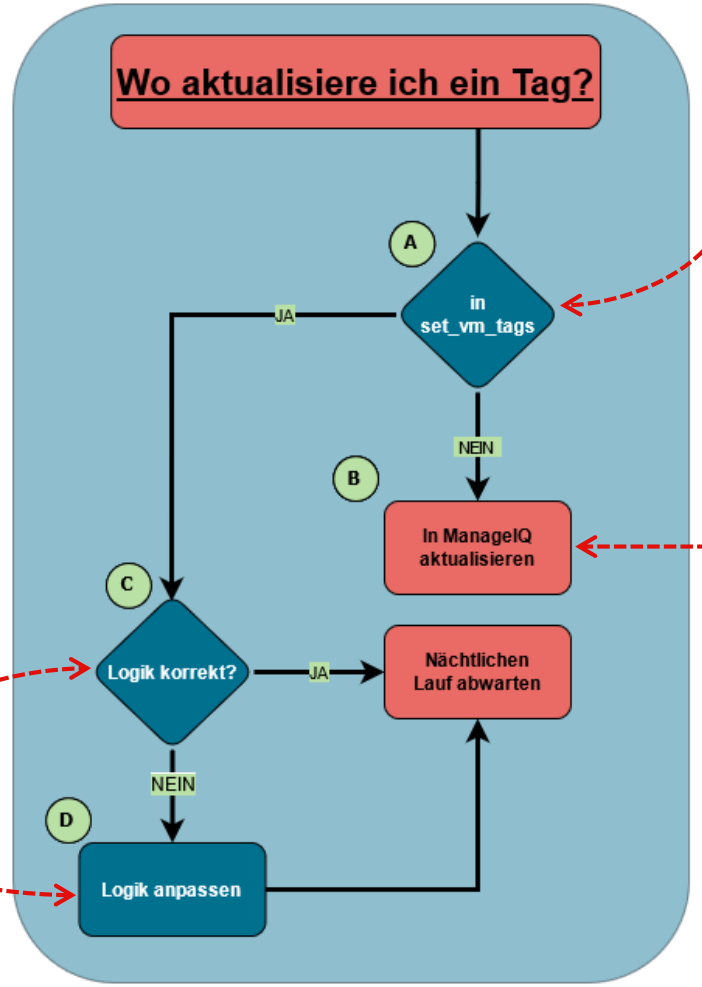




06

Metadaten & Tags

Gothaer

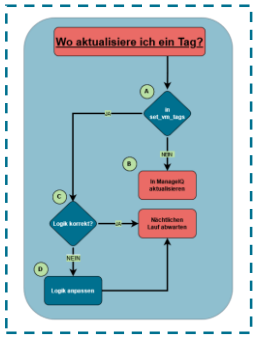


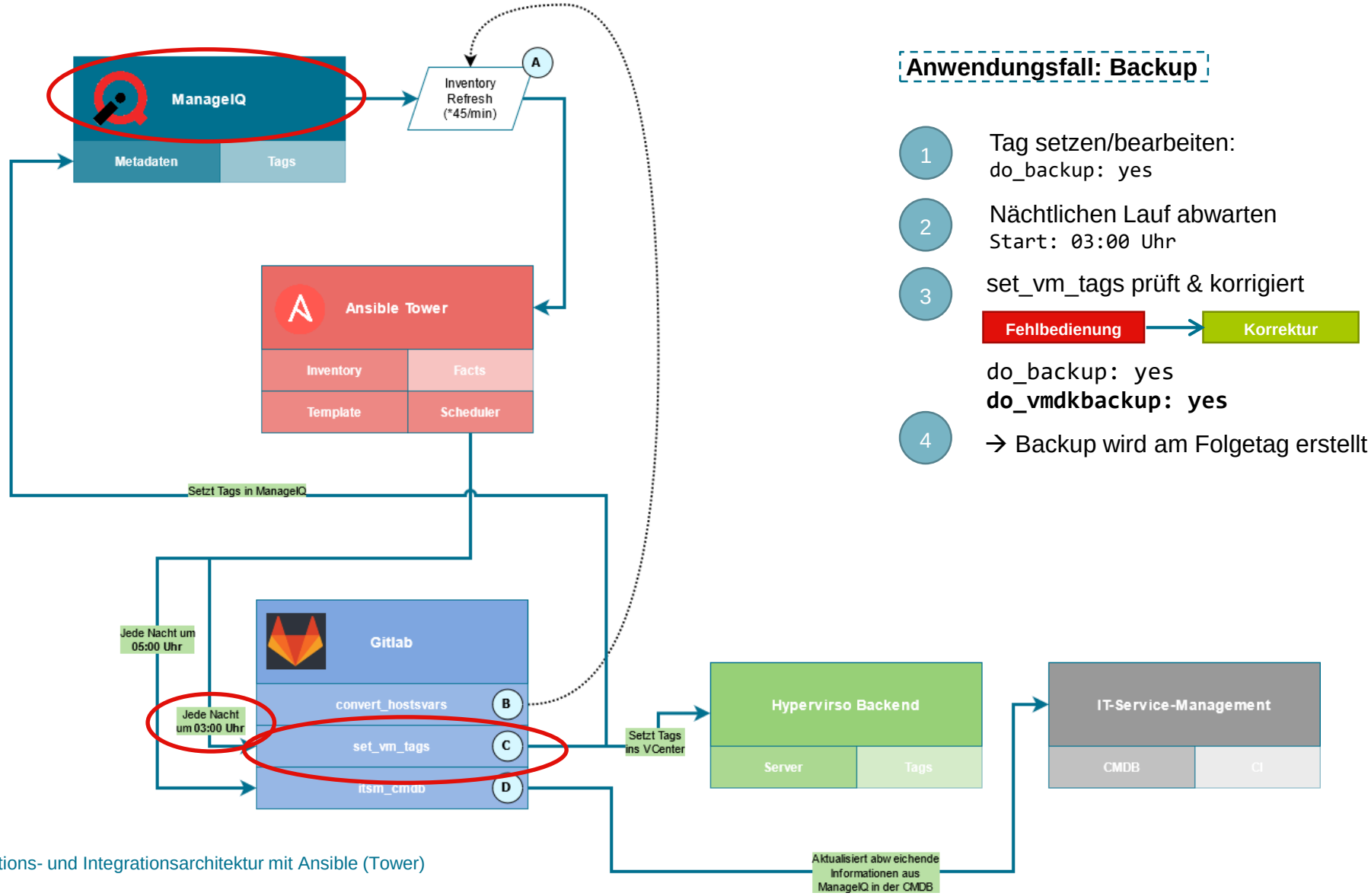
Einstiegspunkt: README.md des Projekts

Einstiegspunkt: Tagging Dokumentation

Dokumentation: README.md des Projekts

06 – Metadaten & Tags





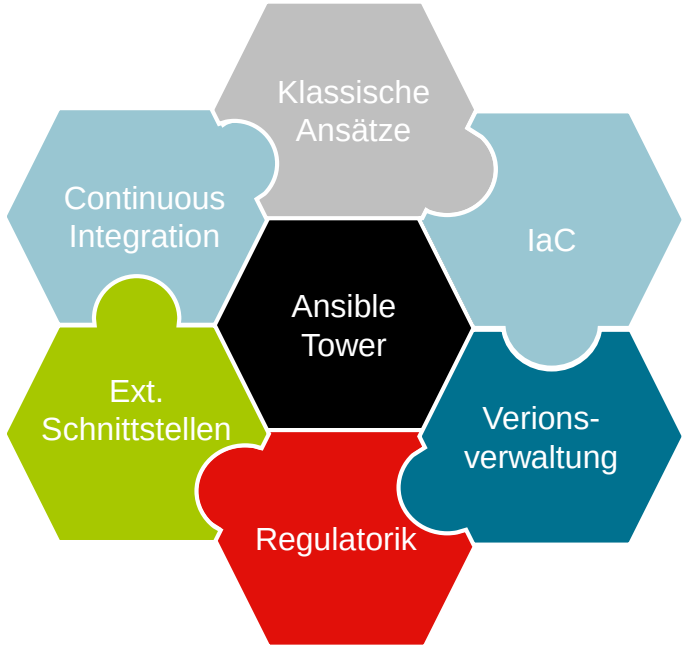
Anwendungsfall: Backup

1. Tag setzen/bearbeiten:
   `do_backup: yes`
2. Nächtlichen Lauf abwarten
   `Start: 03:00 Uhr`
3. set_vm_tags prüft & korrigiert
   Fehlbedienung → Korrektur
   `do_backup: yes`
   `do_vmdkbackup: yes`
4. → Backup wird am Folgetag erstellt

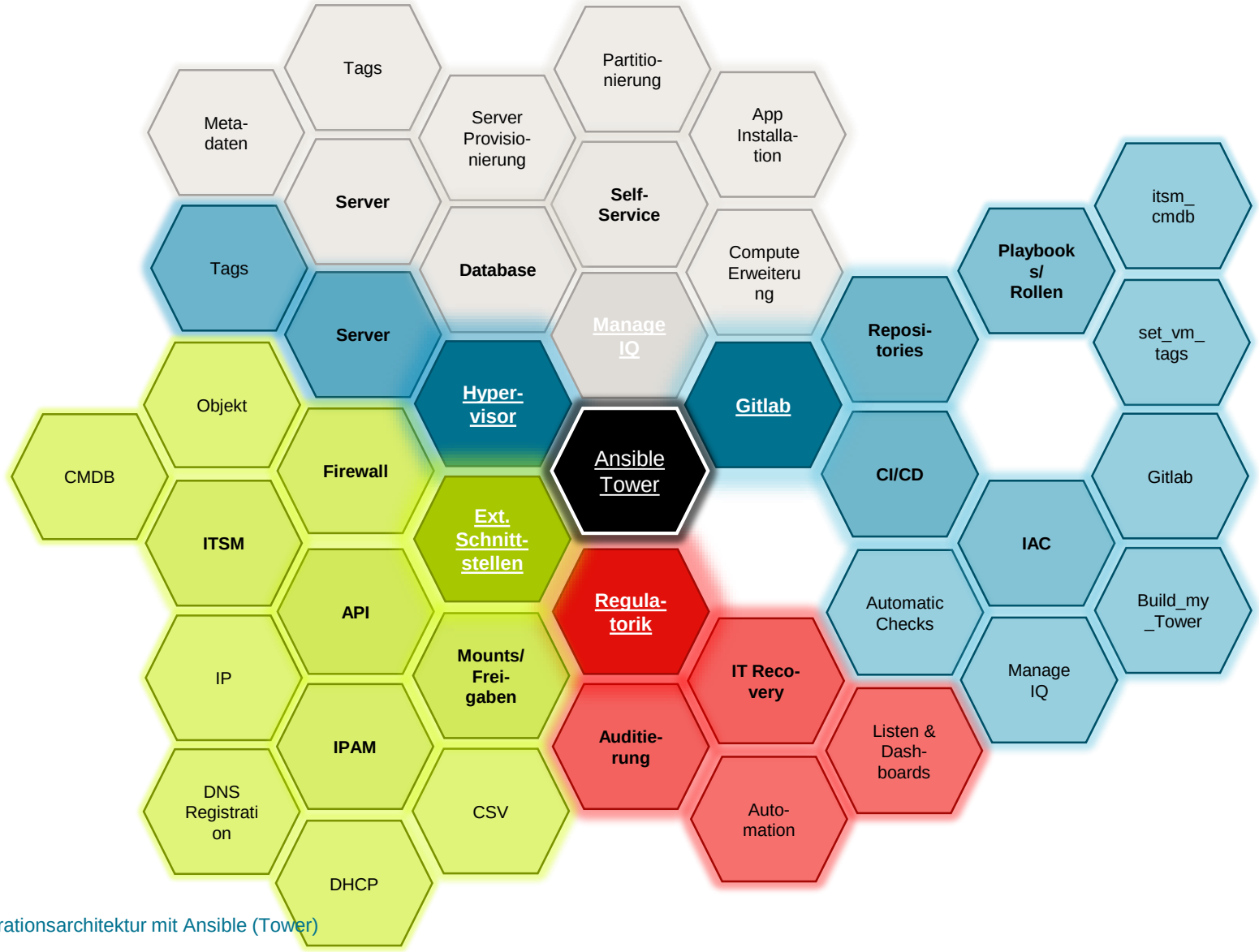
07

Big Picture

07 – Big Picture



07 – Big Picture



Follow us:

www.xing.com/companies/gothaerkonzern

www.linkedin.com/company/gothaer

www.facebook.com/gothaerkarriere

@gothaer_karriere

www.twitter.com/gothaer

Follow me:

xing.to/A-PRINZ

linkedin.com/in/a-prinz

Unser Antrieb:
In der Gemeinschaft Werte schützen.

Gothaer