



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



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

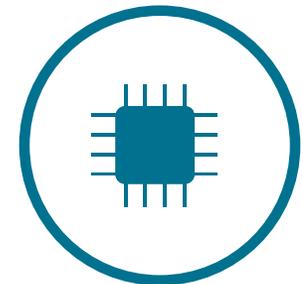
Und: Wir bieten Top Karrierechancen für IT-Spezialisten





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





01 Baukastenprinzip



02 Regulatorik



03 Bodenplatte



04 IaC Ansätze



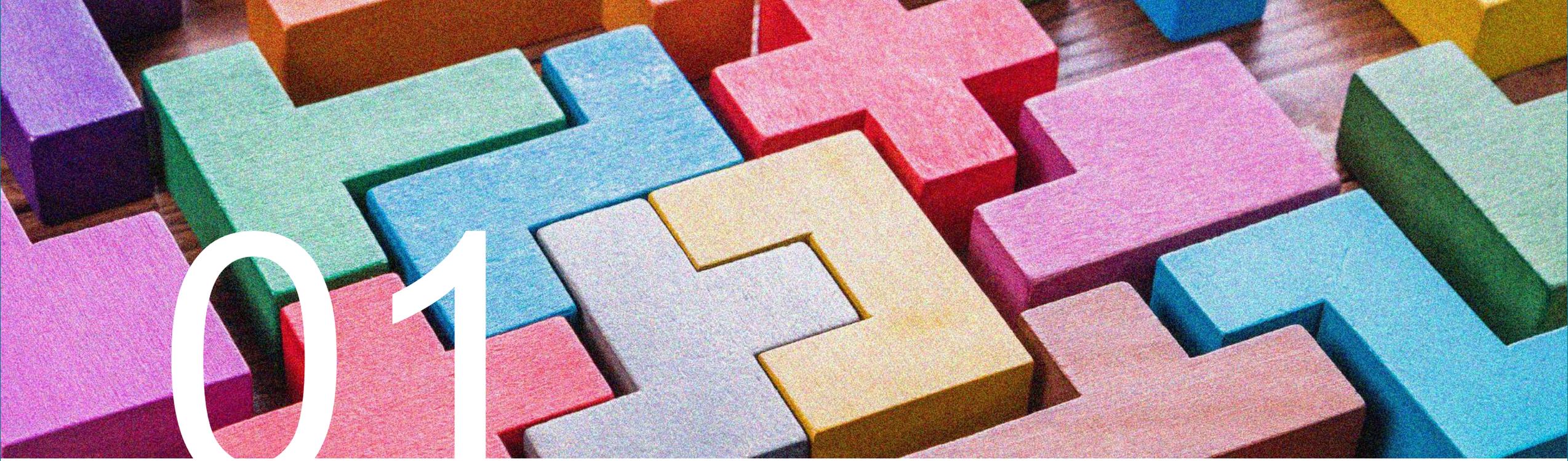
05 Klassische Ansätze



06 Metadaten & Tags

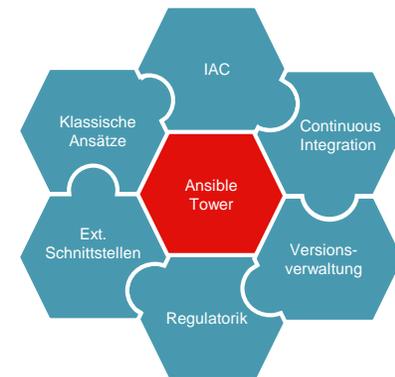


07 Big Picture

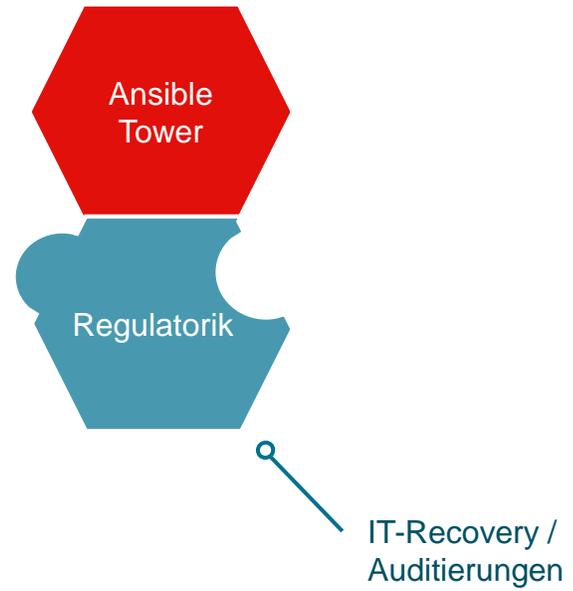


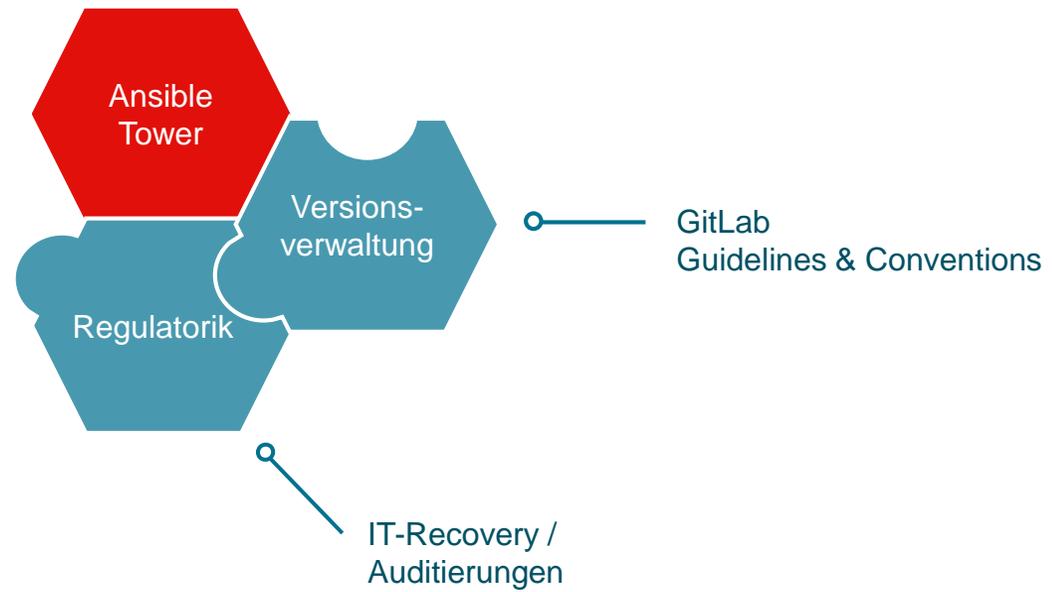
01

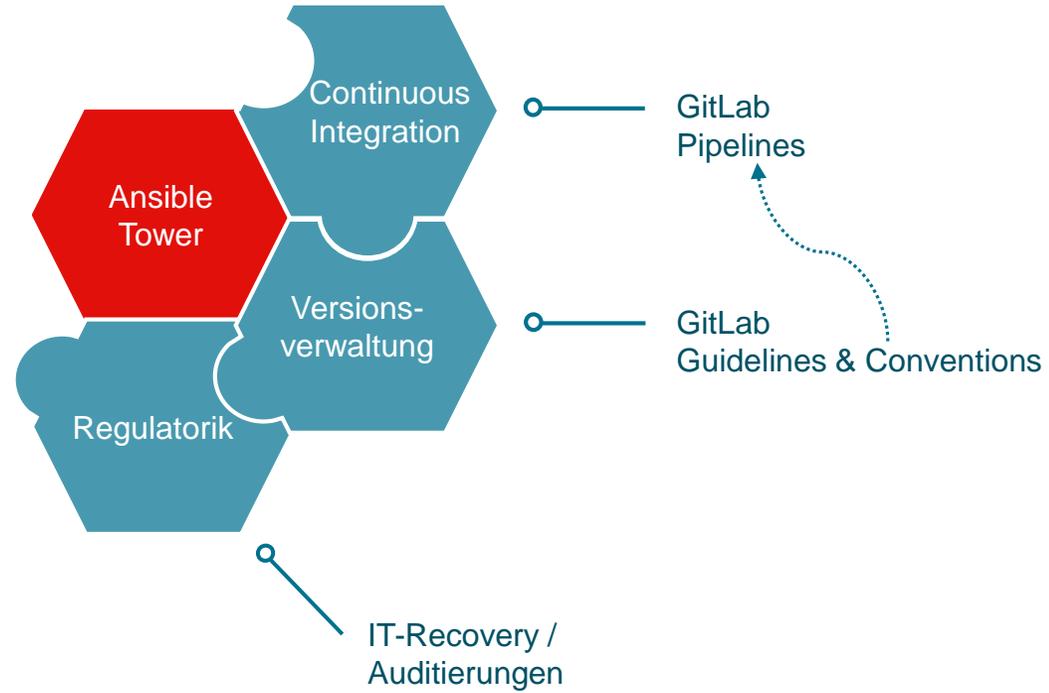
Baukastenprinzip

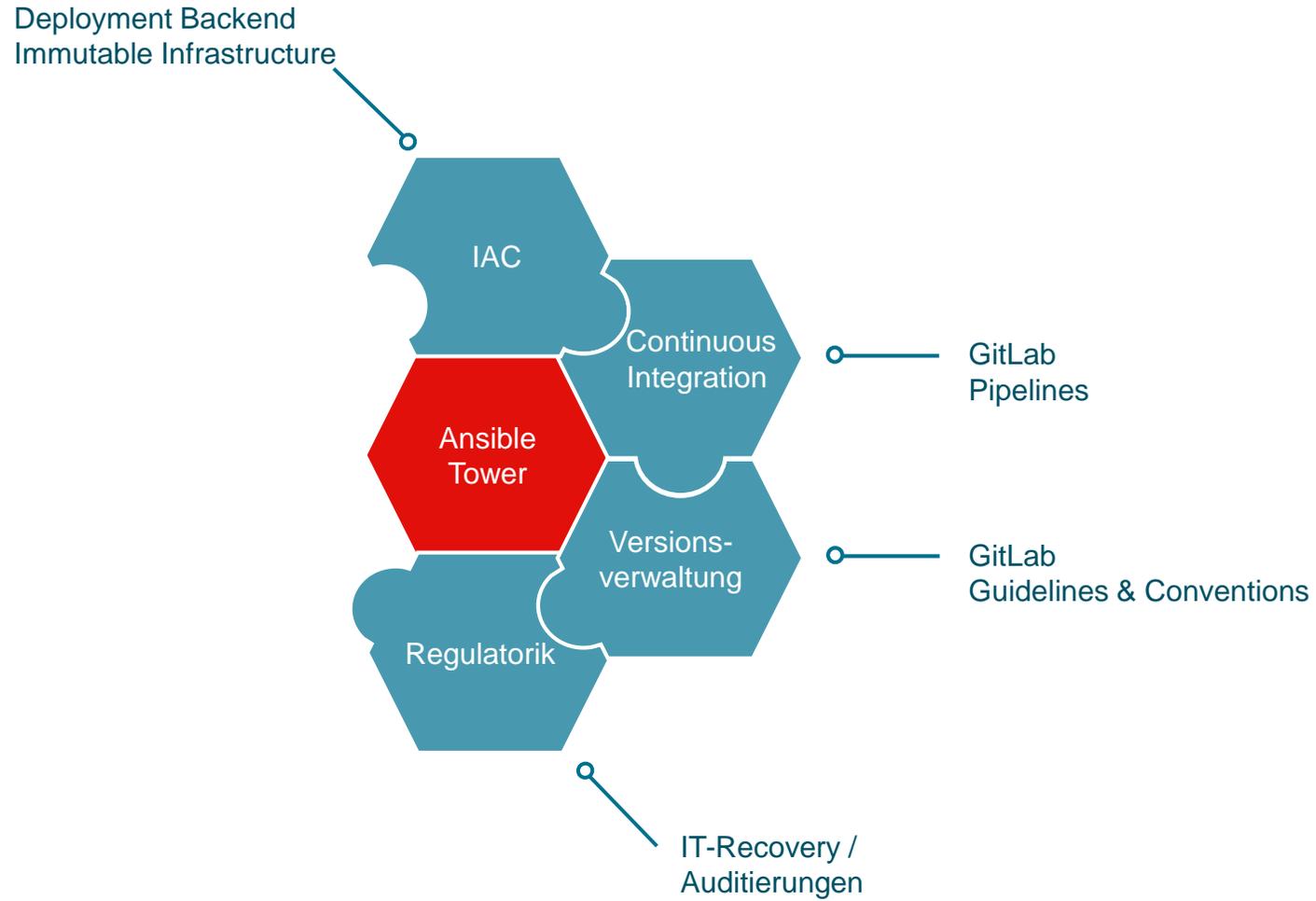


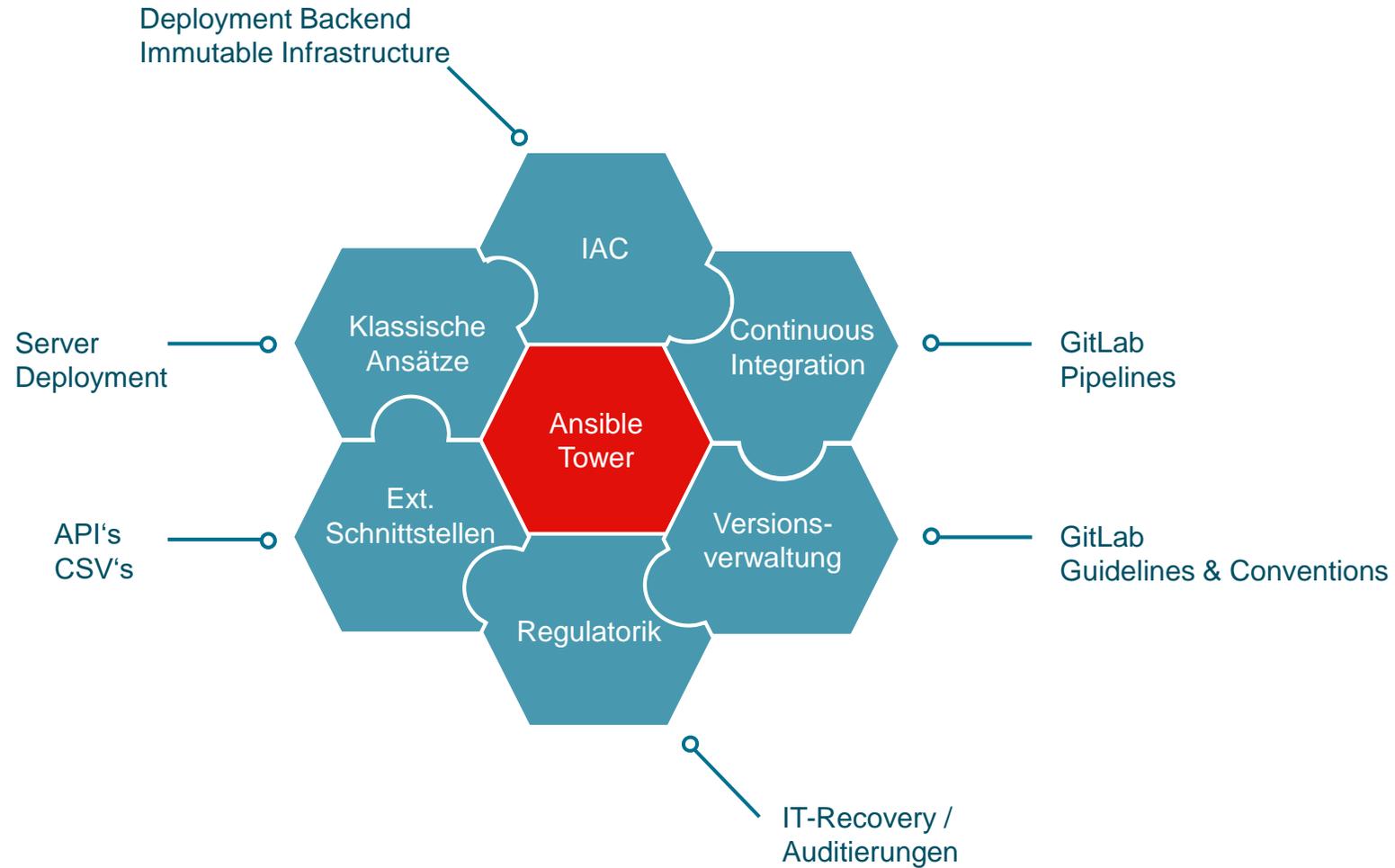












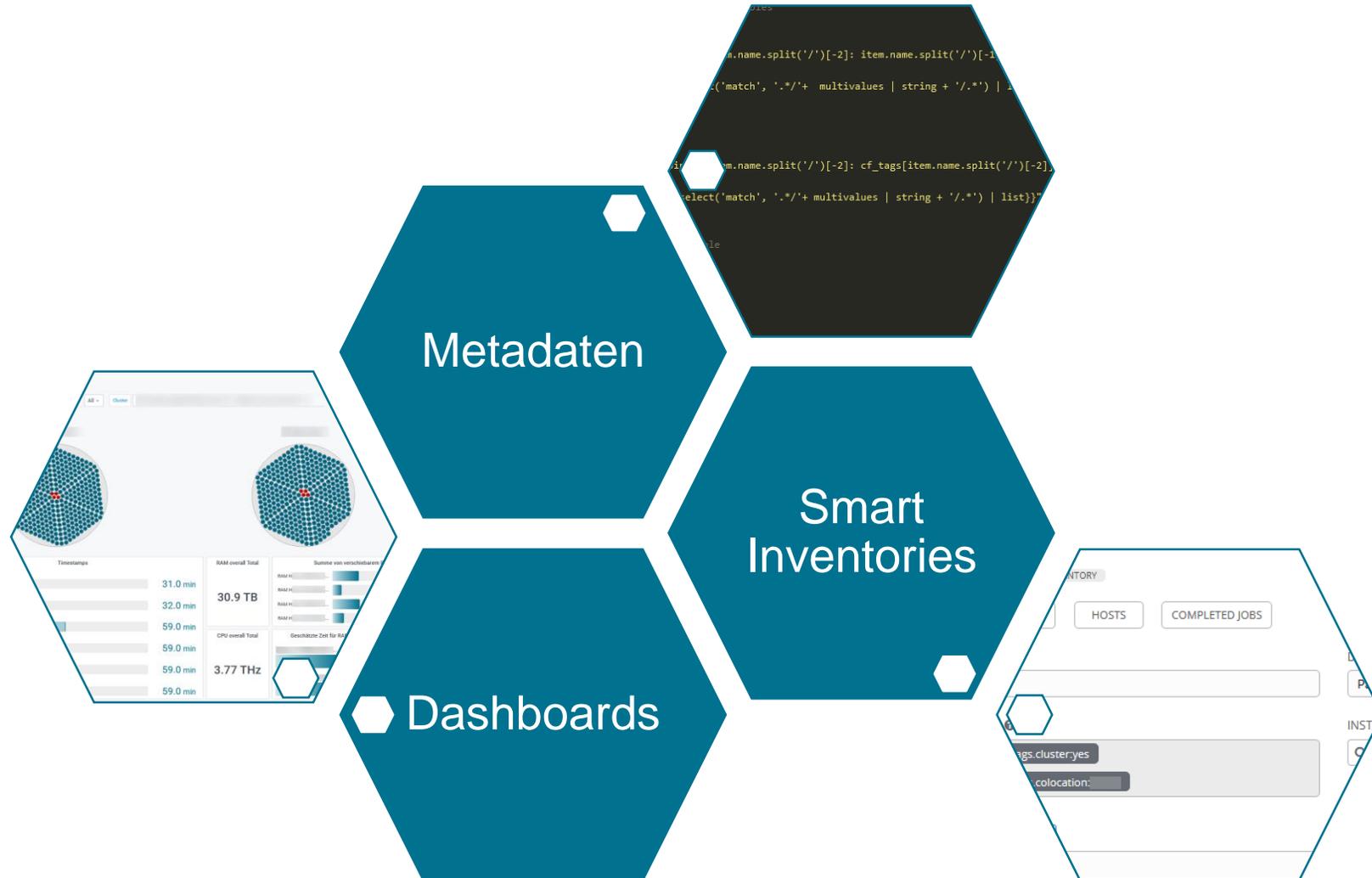


02

Regulatorik



Gothaer



Metadaten

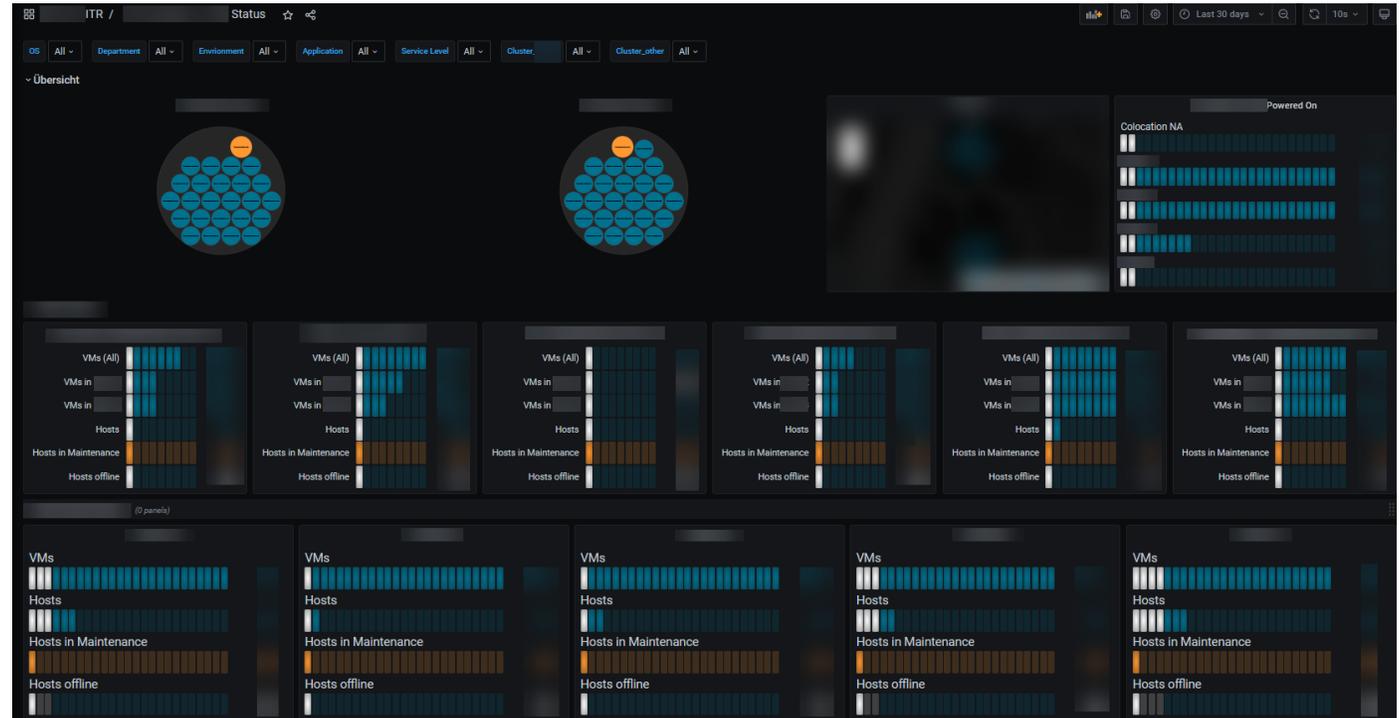
- Zentrale Datenbank
- Eigene Hoheit
- Maximale Flexibilität
- Single Point of Truth
- Skalierendes Konzept auf Basis von Tags

```
30     ### Convert only the tag section of the cloudforms variables
31     - name: Convert Custom Tags - Single Tags
32       set_fact:
33         cf_tags: "{{ cf_tags|default({}) | combine( {item.name.split('/')[-2]: item.name.split('/')[-1]}) }}"
34         cacheable: yes
35       with_list: "{{ cloudforms.tags | unique | reject('match', '.*'+ multivalued | string + '/*.*') | list }}"
36       when: cloudforms.tags is defined
37
38     - name: Convert Custom Tags - Mutli Tags
39       set_fact:
40         cf_tags: "{{ cf_tags|default({}) | combine( {item.name.split('/')[-2]: cf_tags[item.name.split('/')[-2]]|default([]) + [item.name.split('/')[-1]]}) }}"
41         cacheable: yes
42       with_list: "{{ cloudforms.tags | unique | select('match', '.*'+ multivalued | string + '/*.*') | list }}"
43       when: cloudforms.tags is defined
44
45     ### In addition convert a single preferred variable
46     - name: Convert Power State
47       set_fact:
48         power_state: "{{ cloudforms.power_state }}"
49         cacheable: yes
50       when: cloudforms.power_state is defined
```



Dashboards

- 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



Smart Inventories



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

[ITR] Cluster SMART INVENTORY

DETAILS PERMISSIONS HOSTS COMPLETED JOBS

* NAME [ITR] Cluster DESCRIPTION

* SMART HOST FILTER ? ansible_facts.cf_tags.cluster=yes
ansible_facts.cf_tags.colocation: INSTANCE GROUPS ?

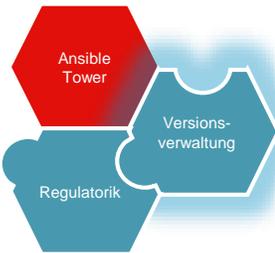
VARIABLES ? YAML JSON

1 ---

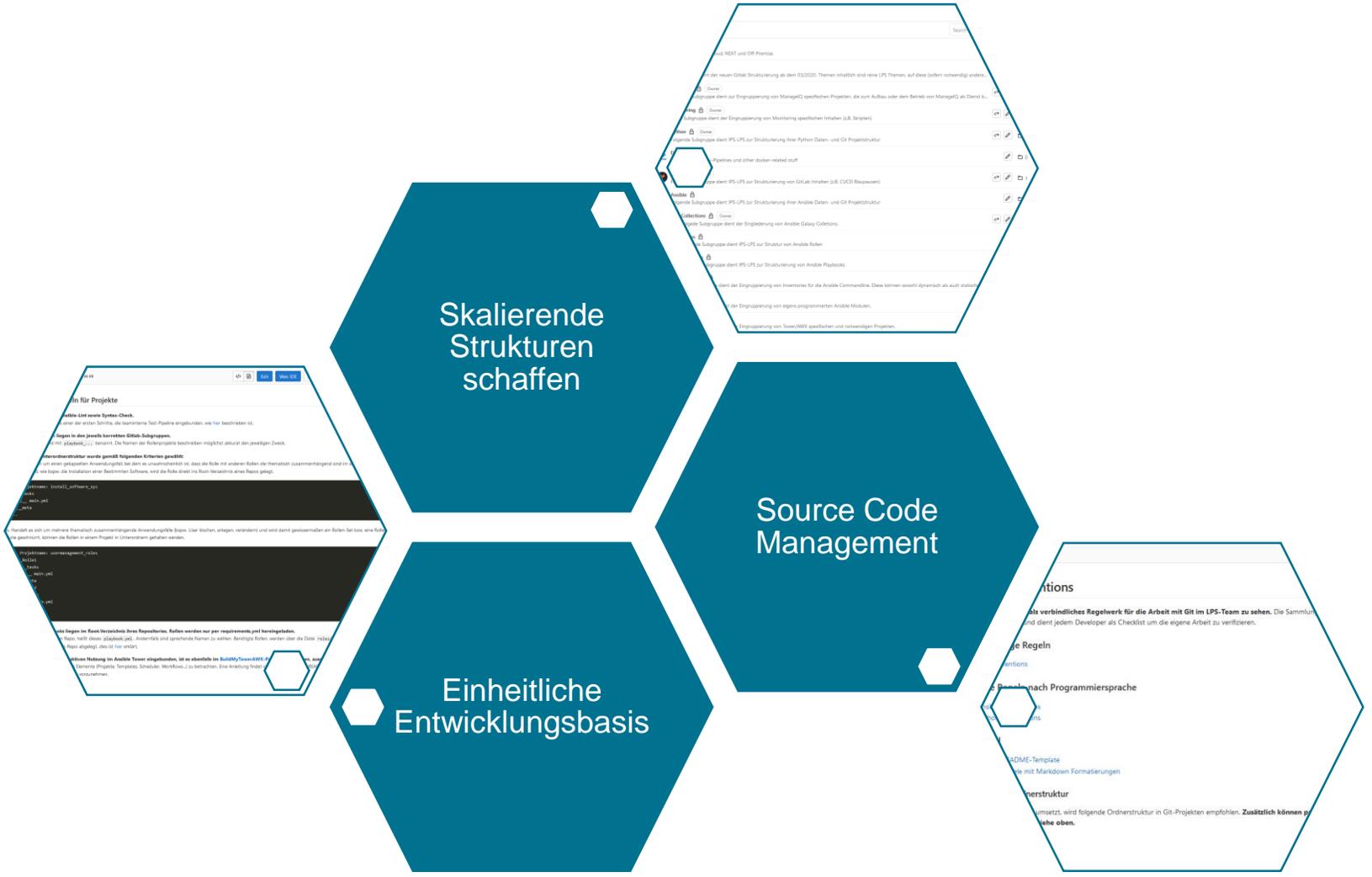
03



Bodenplatte



Gothaer



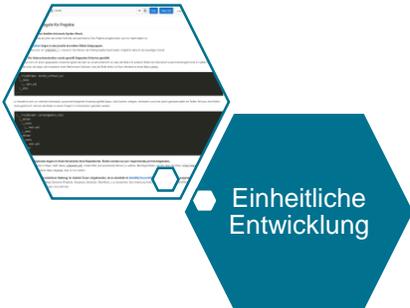


- Klar definierte Struktur
- Regelungen bzgl.
 - Ablage
 - Informationen
 - Dokumentation
- Ausgelegt auf Wiederverwendbarkeit
- Indizierte Codesuche

Groups

Search by name | Last created

- Cloud
- LPS (Owner) - 6 folders, 1 bookmark, 15 users
- ManagelQ (Owner) - 0 folders, 0 bookmarks, 1 user
- Python (Owner) - 3 folders, 0 bookmarks, 1 user
- GitLab (Owner) - 1 folder, 0 bookmarks, 3 users
- Ansible (Owner) - 6 folders, 0 bookmarks, 1 user
 - Collections (Owner) - 0 folders, 0 bookmarks, 2 users
 - Rollen - 0 folders, 0 bookmarks, 47 users
 - Playbooks - 0 folders, 0 bookmarks, 54 users
 - Inventories - 0 folders, 0 bookmarks, 1 user
 - Module - 0 folders, 0 bookmarks, 1 user
 - Tower - 0 folders, 0 bookmarks, 5 users



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

Overall result for Pipeline 3755 triggered by Maintainer · 6 days ago

Next step: You are ready to merge!

ansible_lint_check

- ✅ Playbook: `playbook.yml` - Linter OK!
- ✅ Playbook: `uninstall.yml` - Linter OK!

ansible_syntax_check

- ✅ Playbook: `playbook.yml` - Syntax-check OK!
- ✅ Playbook: `uninstall.yml` - Syntax-check OK!

README_idempotence_check

- ✅ README.md fulfills idempotence check. Date is current enough!

README_linecount_check

- ✅ README.md has 84 lines, thats enough!

10_LPS_Anible_Conventions.md · 2.46 KB

Ansible-Bezogene Regeln für Projekte

- Der Code besteht einen Ansible-List sowie Syntax-Check.**
Zur Überprüfung wurde als einer der ersten Schritte, die teaminterne Test-Pipeline eingebunden, wie hier beschrieben ist.
- Playbooks und Rollen liegen in den jeweils korrekten Gitlab-Subgruppen.**
Playbook-Projekte sind mit `playbook_...` benannt. Die Namen der Rollenprojekte beschreiben möglichst akkurat den jeweiligen Zweck.
- Bei Rollen: Die Unterordnerstruktur wurde gemäß folgenden Kriterien gewählt:**
 - Handelt es sich um einen gekapselten Anwendungsfall, bei dem es unwahrscheinlich ist, dass die Rolle mit anderen Rollen die thematisch zusammenhängend sind in selben Playbook verwendet wird, wie bspw. die Installation einer Bestimmten Software, wird die Rolle direkt ins Root-Verzeichnis eines Repos gelegt.


```

__projekte: install_software_xyz
|__tasks
|__main.yml
|__meta
...
```
 - Handelt es sich um mehrere thematisch zusammenhängende Anwendungsfälle (bspw. User löschen, anlegen, verändern) und wird damit gewissermaßen ein Rollen-Set bzw. eine Rollen-Suite geschürt, können die Rollen in einem Projekt in Unterordnern gehalten werden.


```

__projekte: usermanagement_roles
|__roles
|__tasks
|__main.yml
|__meta
|__roles
|__tasks
|__main.yml
|__meta
...
```
- Bei Playbooks: Playbooks liegen im Root-Verzeichnis ihres Repositories. Rollen werden nur per requirements.yml hereingeladen.**
Gibt es nur ein Playbook im Repo, heißt dieses `playbook.yml`. Andernfalls sind sprechende Namen zu wählen. Benötigte Rollen, werden über die Datei `roles/requirements.yml` verlinkt und nicht direkt im Playbook-Repo abgelegt, dies ist hier erklärt.
- Würde das Projekt zur produktiven Nutzung im Ansible Tower eingebunden, ist es ebenfalls im `roles/requirements.yml` Projekt einzutragen, ausgenommen sind Test-Projekte.**
Dabei sind alle im Tower angelegt Elemente (Projekte, Templates, Scheduler, Workflows...) zu betrachten. Eine Anleitung findet sich hier, sowie in der README.md des Projektes. In diesem Projekt sind auch die Eintragungen vorzunehmen.

Source Code Management



- 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](#)

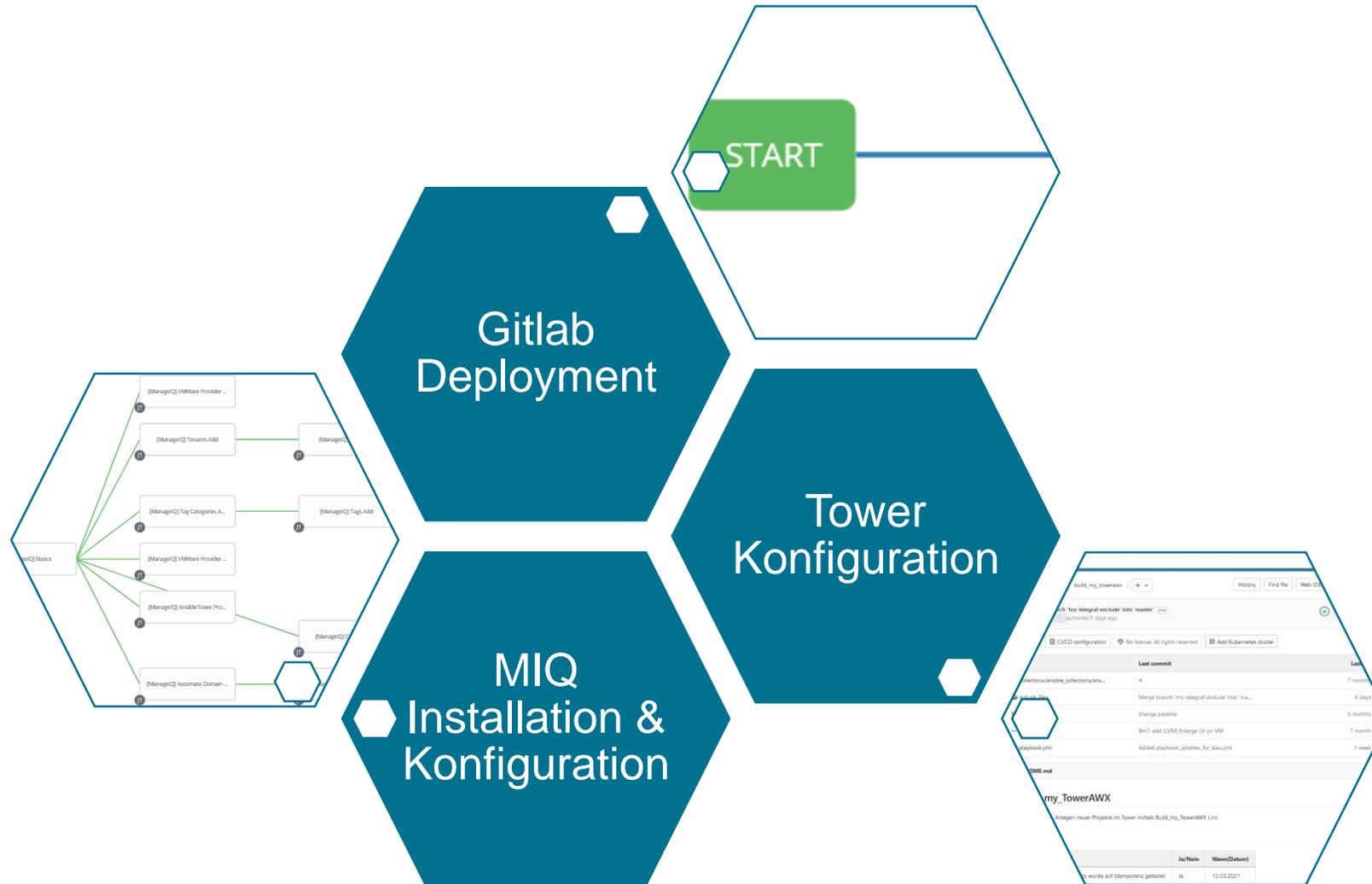
Empfehlung zur Ordnerstruktur

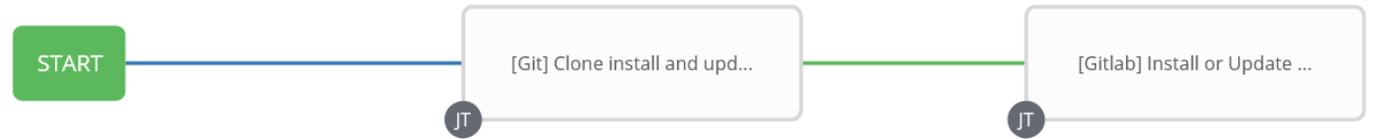
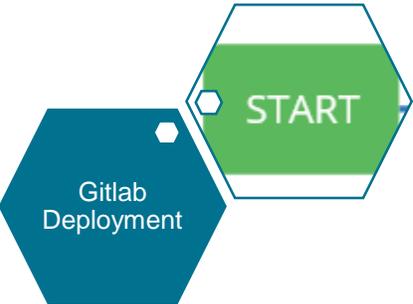
- ▶ Ordnerstruktur

04

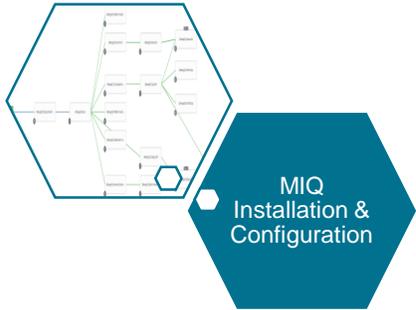
IaC – Infrastructure as Code & Immutable Infrastructure







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



- 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



Tower Configuration

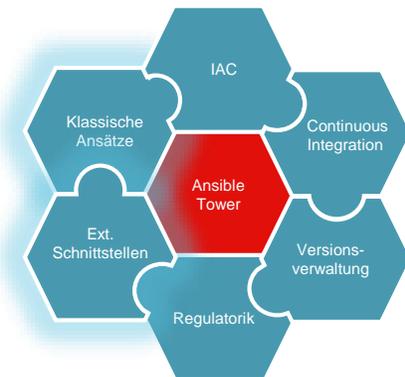


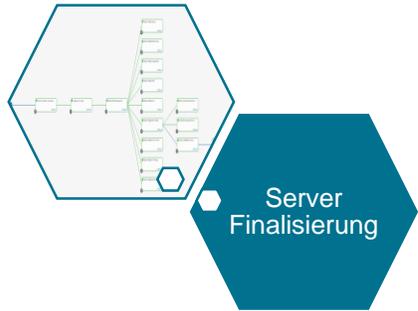
- Konfiguration via Ansible
- IaC Pipeline als Wrapper
- Historische Daten nicht zwingend relevant
 - Optional: Backup
- Ausblick: Version 2.0 mit zugrunde liegenden Informationen in Diskussion

```
409 TASK [Create workflow nodes 7] *****
410 ok: [ ]
411 TASK [Create workflow nodes 6] *****
412 ok: [ ]
413 TASK [Create workflow nodes 5] *****
414 ok: [ ]
415 TASK [Create workflow nodes 4] *****
416 ok: [ ]
417 TASK [Create workflow nodes 3] *****
418 ok: [ ]
419 TASK [Create workflow nodes 2] *****
420 ok: [ ]
421 TASK [Create workflow nodes 1] *****
422 ok: [ ]
423 TASK [include_tasks] *****
424 included: /builds/NsTYxzPz/0/lps/ansible/tower_ include_files/Finalize_workflow/apicalls_finalize.yml for [ ]
425 TASK [Add Tower Project APICalls Finalize Deployment] *****
426 ok: [ ]
427 TASK [Add Tower Job template APICalls Finalize Deployment] *****
428 ok: [ ]
429 TASK [Add Tower Job template APICalls Finalize Deployment] *****
430 ok: [ ]
431 TASK [Give Executer Access] *****
432 ok: [ ]
433 TASK [include_tasks] *****
434 included: /builds/NsTYxzPz/0/lps/ansible/tower_ include_files/Finalize_workflow/workflow_finalize.yml for [ ]
435 TASK [Create Finalize Workflow] *****
436 [WARNING]: The value 63 (type int) in a string field was converted to u'63'
437 (type string). If this does not look like what you expect, quote the entire
438 value to ensure it does not change.
439 ok: [ ]
440 TASK [Give Admin Access] *****
441 ok: [ ]
442 TASK [Create workflow nodes] *****
443 ok: [ ] => (item=[ ]
444 ok: [ ] => (item=[ ]
445 ok: [ ] => (item=[ ]
446 ok: [ ] => (item=[ ]
447 ok: [ ] => (item=[ ]
448 ok: [ ] => (item=[ ]
449 ok: [ ] => (item=[ ]
```

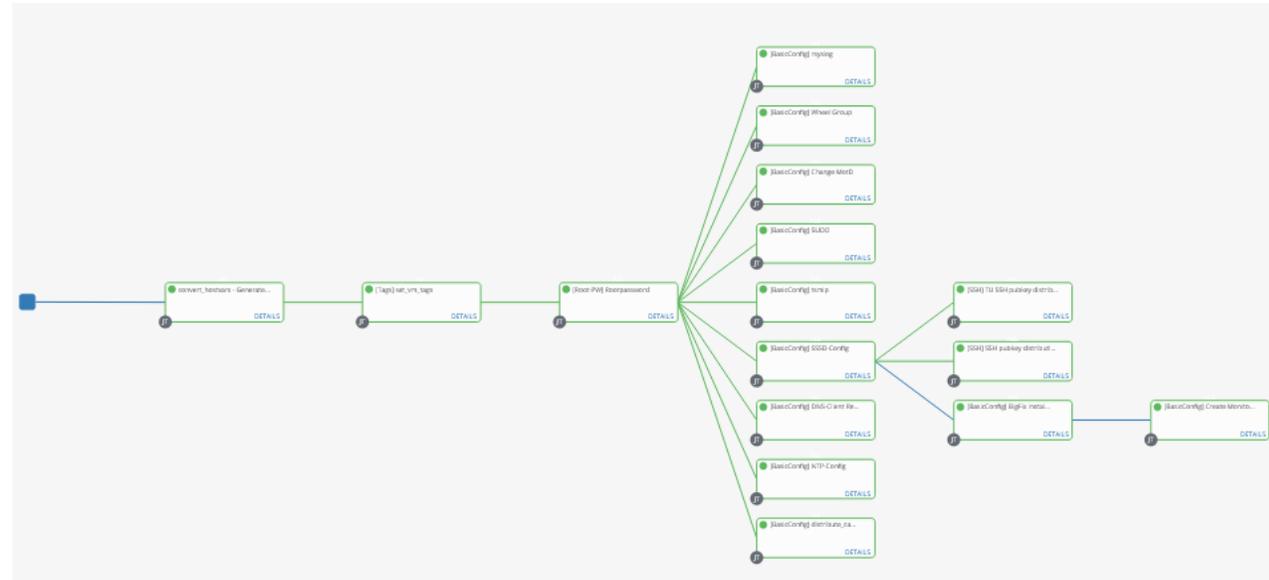
05

Klassische Ansätze





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



App
Installation
(e.g.
Database)



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

[DB2/ManageIQ] Installation DB2

DETAILS PERMISSIONS NOTIFICATIONS COMPLETED JOBS SCHEDULES EDIT SURVEY

* NAME: [{{ }} Installation] PROMPT ON LAUNCH

DESCRIPTION: [installiert über {{ }} auf einem Zielsystem]

* INVENTORY: [inv_{{ }}] PROMPT ON LAUNCH

* PROJECT: [P]{{ }} Installation DB

* PLAYBOOK: [playbook.yml]

CREDENTIALS: [{{ }}] PROMPT ON LAUNCH

FORKS: [0] PROMPT ON LAUNCH

* VERBOSITY: [0 (Normal)] PROMPT ON LAUNCH

JOB TAGS: [] PROMPT ON LAUNCH

SKIP TAGS: [] PROMPT ON LAUNCH

LABELS: []

ANSIBLE ENVIRONMENT: [Use Default Environment]

INSTANCE GROUPS: []

JOB SLICING: [1] PROMPT ON LAUNCH

TIMEOUT: [0] PROMPT ON LAUNCH

SHOW CHANGES: PROMPT ON LAUNCH

OPTIONS

ENABLE PRIVILEGE ESCALATION

ENABLE PROVISIONING CALLBACKS

ENABLE WEBHOOK

ENABLE CONCURRENT JOBS

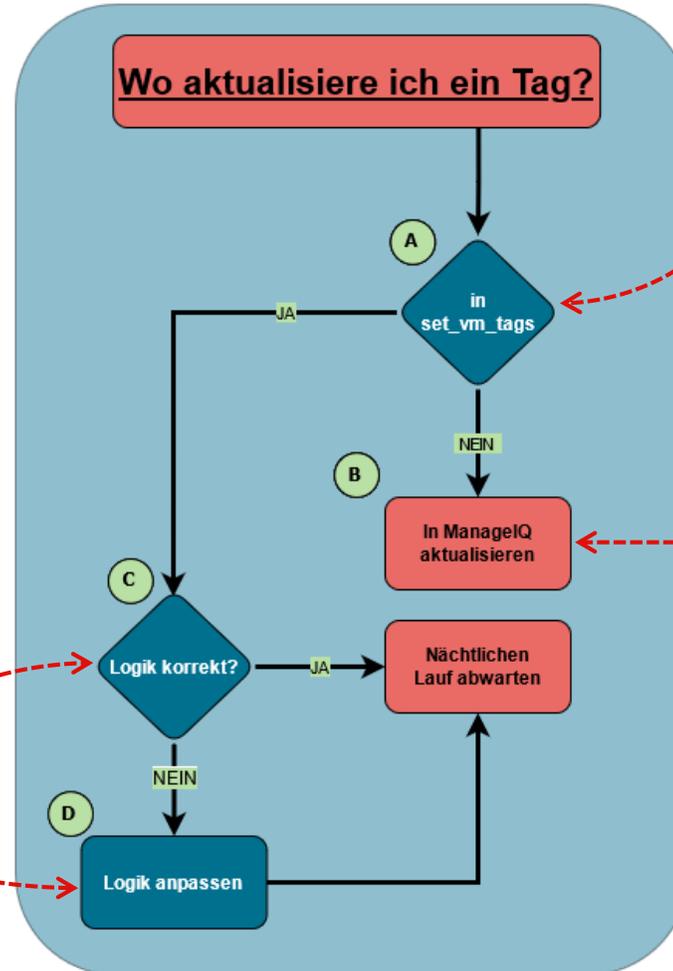
ENABLE FACT CACHE

EXTRA VARIABLES: [YAML JSON] PROMPT ON LAUNCH

1 ---



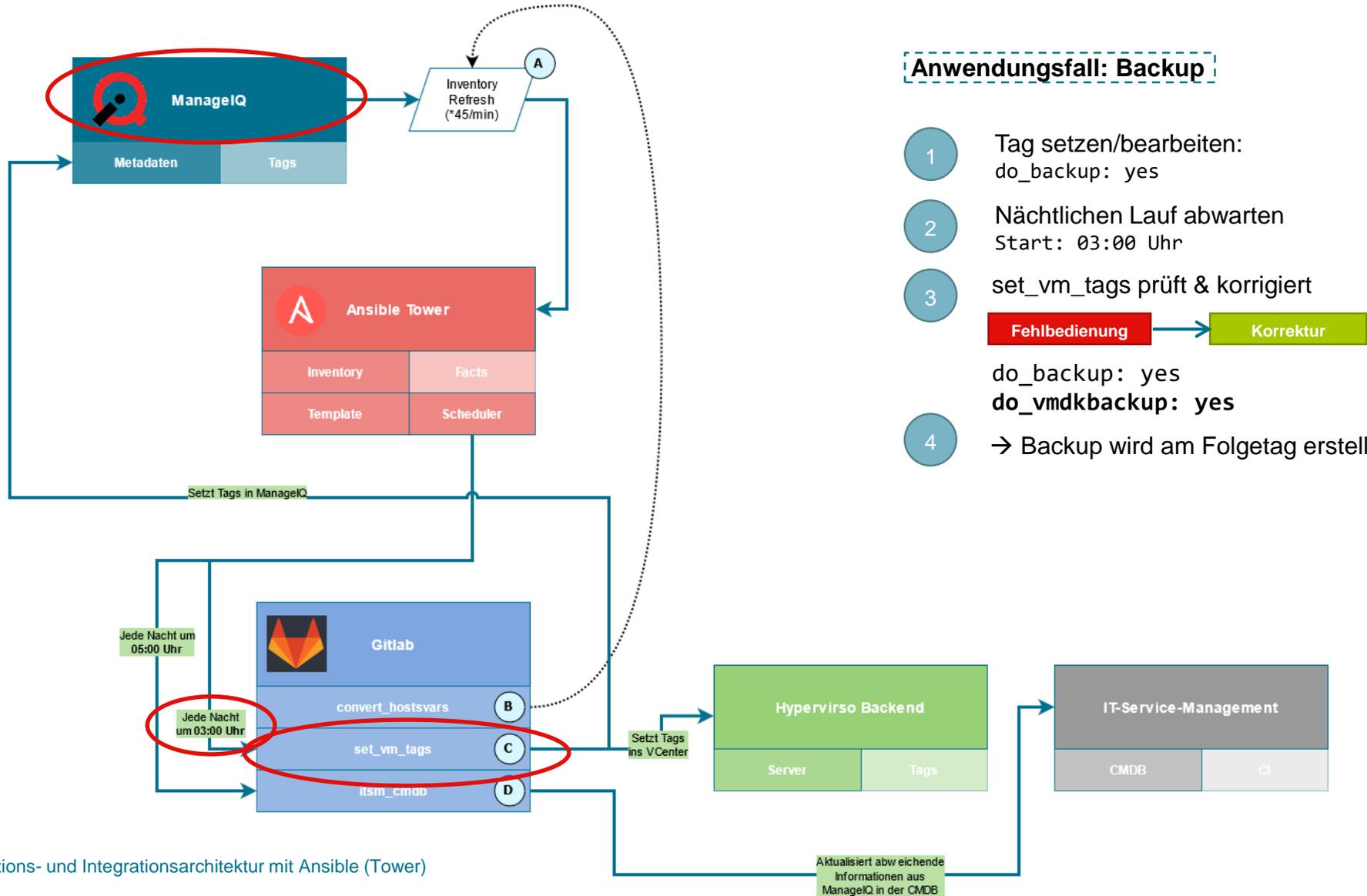
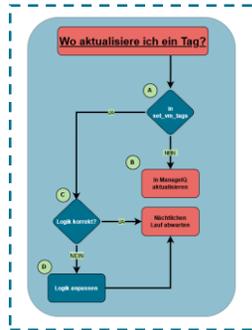
Metadaten & Tags



Einstiegspunkt: [README.md](#)
des Projekts

Einstiegspunkt: [Tagging Dokumentation](#)

Dokumentation: [README.md](#)
des Projekts



Anwendungsfall: Backup

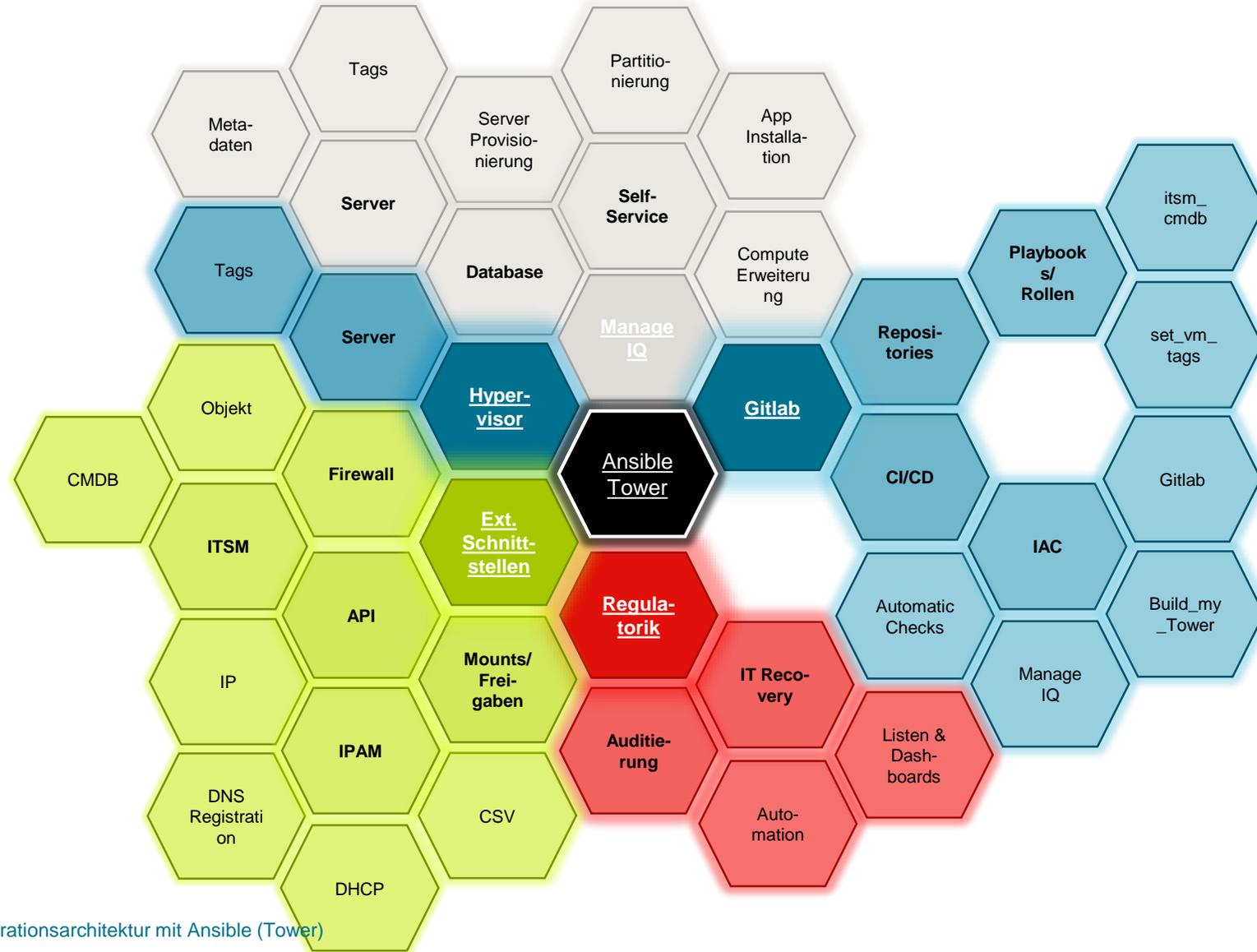
- 1 Tag setzen/bearbeiten:
do_backup: yes
- 2 Nächtlichen Lauf abwarten
Start: 03:00 Uhr
- 3 set_vm_tags prüft & korrigiert

Fehlbedingung	Korrektur
---------------	-----------
- 4 → Backup wird am Folgetag erstellt



Big Picture





Follow us:

 www.xing.com/companies/gothaerkonzern

 www.linkedin.com/company/gothaer

 www.facebook.com/gothaerkarriere

 [@gothaer_karriere](https://www.instagram.com/gothaer_karriere)

 www.twitter.com/gothaer

Follow me:

xing.to/A-PRINZ 

linkedin.com/in/a-prinz 

**Unser Antrieb:
In der Gemeinschaft Werte schützen.**