Inhaltsverzeichnis

LS 3 – Infos – Programmierrichtlinien

Programmierrichtlinien – Warum, Ziele?

Programme werden öfters gelesen als geschrieben.

- Einheitliche Struktur, Layout, Namensgebung
  - ✓ Bessere Übersicht
  - ✓ Einfacher zu lesen
  - ✓ Einfacher zu ändern und zu erweitern

- Gängelung des Programmierers, Freiheitsbeschränkung

Python Programmierrichtlinien BK-GuT

Python Programmierrichtlinien

PEP8

- Einrückung 4 Leerzeichen; keine Tabs
- Zeilenlänge soll max. 80 Zeichen
- Leerzeilen zur Strukturierung nutzen; max. 2 Leerzeilen hintereinander

- Einheitlicher Kopfkommentar am Anfang jeder Datei mit Docstrings
- Funktionen (und Klassenbeschreibungen) mit Docstrings
- Kommentare deutsch
- Kommentare nur da, wo sie hilfreich sind
- Kommentare müssen immer zum Programm passen

- Bezeichner (Funktionsnamen, Variablennamen, …) nur Standard-ASCII-Zeichen (Buchstaben, Ziffern, _) benutzen; keine Umlaute und Sonderzeichen benutzen
- Für Bezeichner beschreibende Namen benutzen
- Bei zusammengesetzten Namen CamelCase-Schreibweise benutzen
- Konstanten in GROSSBUCHSTABEN

- Strings in “Gänsfüßchen“
- Importe am Dateianfang; jeder Import in eine eigene Zeile
- Encoding: UTF-8
- Ein- und Ausgaben nicht mit Berechnungen / Verarbeitung mischen, sondern jeweils in eine eigene Zeile.

Python IDE

Autoformatierer

Code Analyzer (Linter)

LS 3 – Infos – Versionsmanagements

Quelle:
www.wikipedia.de

Eine Versionsverwaltung (Versionsmanagement) ist ein System, dass zur Erfassung von Änderungen an Dokumenten und Dateien verwendet wird.

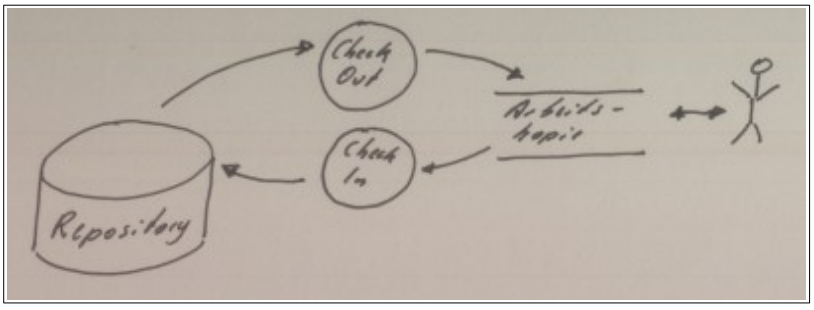
Aufgaben eines Versionsmanagementsystems

- Protokollierung von Änderungen; wer hat was wann geändert
- Wiederherstellung alter Stände
- Archivierung spezieller Stände
- Koordinierung gemeinsamer Zugriffe mehrerer Entwickler
- Gleichzeitige Entwicklung mehrerer Entwicklungszweige
- Zusammenfassung mehrerer Elemente zu einer Konfiguration
- Build-Unterstützung; automatische Generierung des Systems aus dem Repository

Ziele des Versionsmanagements

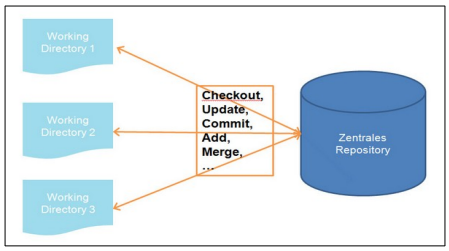
- Änderungen kontrollieren
- Qualität sicherstellen
- Produktivität steigern
- Transparenz verbessern

Prinzipielle Arbeitsweise

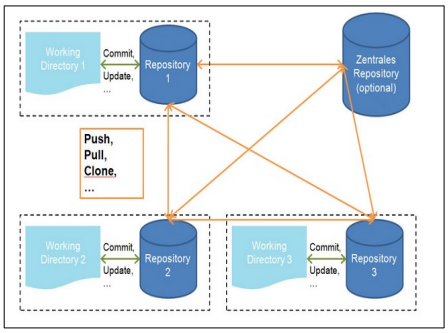


Typen von Versionsmanagementsystemen

1. **Lokale Versionsverwaltungssysteme** (z.B.: SCCS)
2. **Zentrale Versionsverwaltungssysteme** (z.B.: CVS, SVN(Subversion))



3. **Verteilte Versionsverwaltungssysteme** (z.B.: GIT)

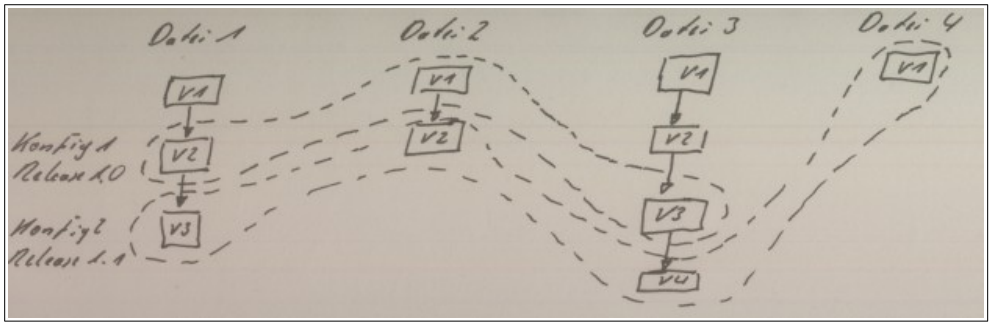


Zugriffskonzepte

1. **Lock-Modify-Write/Unlock (Pessimistische Versionsverwaltung)**
   - Sequentielles Arbeiten
   - kein Zusammenmischen (Merge) notwendig
   - Warten auf Zugriff
2. **Copy-Modify-Merge (Optimistische Versionsverwaltung)**
   - Gleichzeitiger Zugriff möglich
   - kein Warten auf Zugriffe
   - Automatisches oder manuelles Zusammenmischen notwendig

Versionsmanagement vs. Konfigurationsmanagement

- Oft als Synonyme gebraucht
- Versionsmanagement eher Verwaltung eines einzelnen Elements
- Konfigurationsmanagement eher Zusammenspiel vieler Elemente zu einem Gesamtsystem



LS 3 – Infos – Technische Möglichkeiten beim Distanzlernen in Schülergruppen

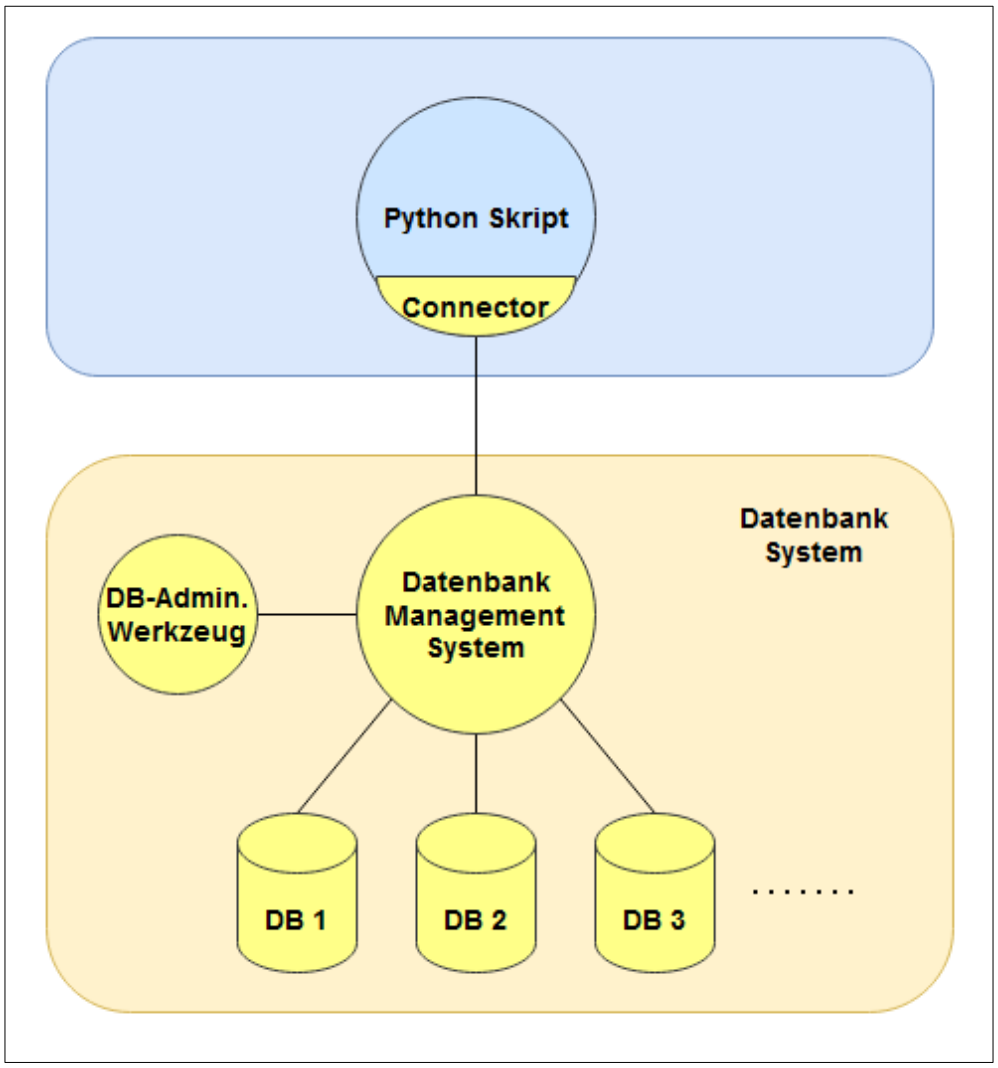
Video-Konferenzen (frei bzw. kostenlose Varianten vorhanden)
- Cisco-Webex
- MS-Teams (Whiteboard-Funktionalität)
- Zoom

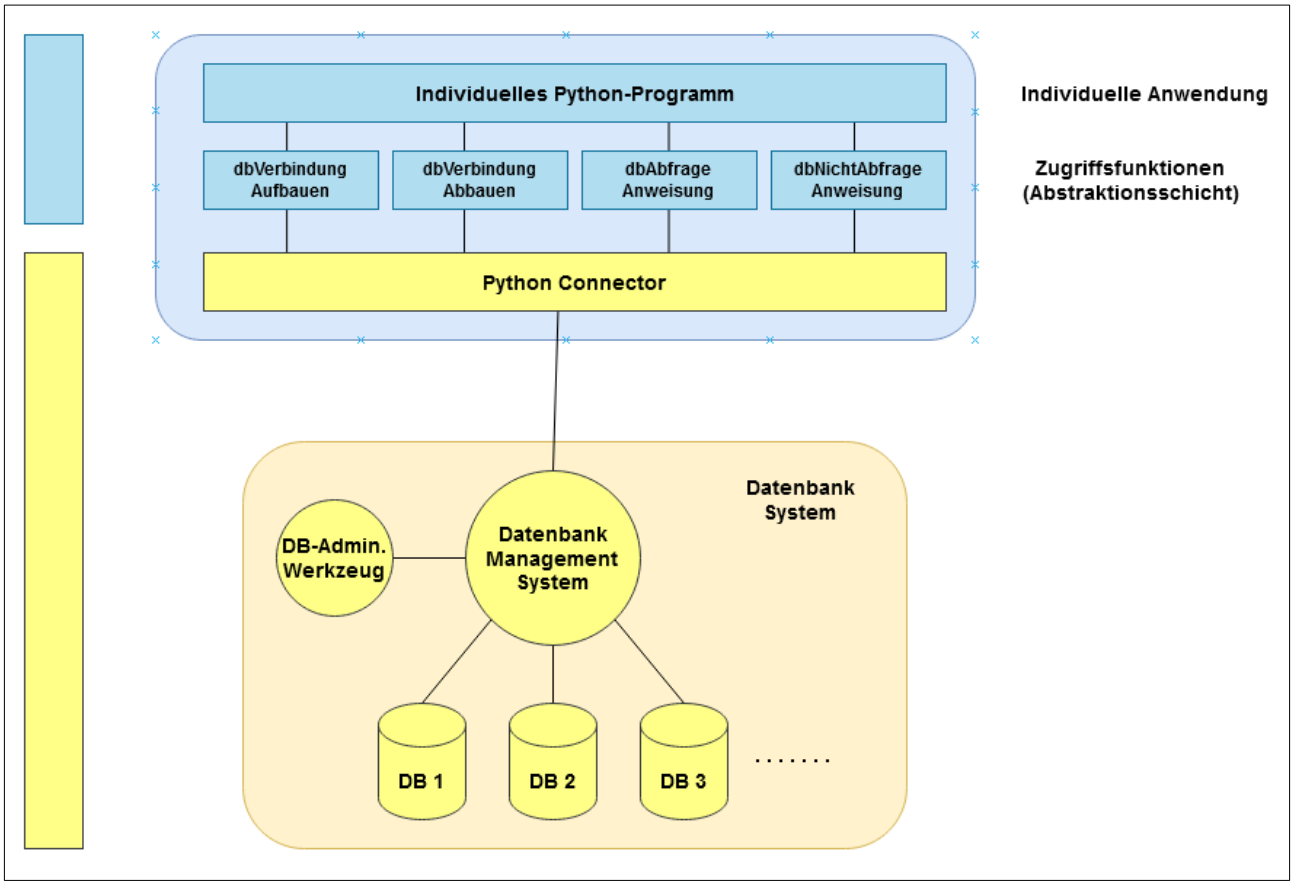
Email
- … diverse Anbieter und Programme (tlw. mit Cloud-Speicher)

Messenger
- Facebook
- WhatsApp

Cloud-Speicher / Dateienaustausch
- Dropbox, Google Drive, MS-OneDrive, ...
- div. kostenlose Anbieter (Mega, pCloud, ...)
- GitHub, Bitbucket, GitLab, ... (Versionsmanagement)

LS 3 – Infos – Aufbau einer Datenbank-anwendung

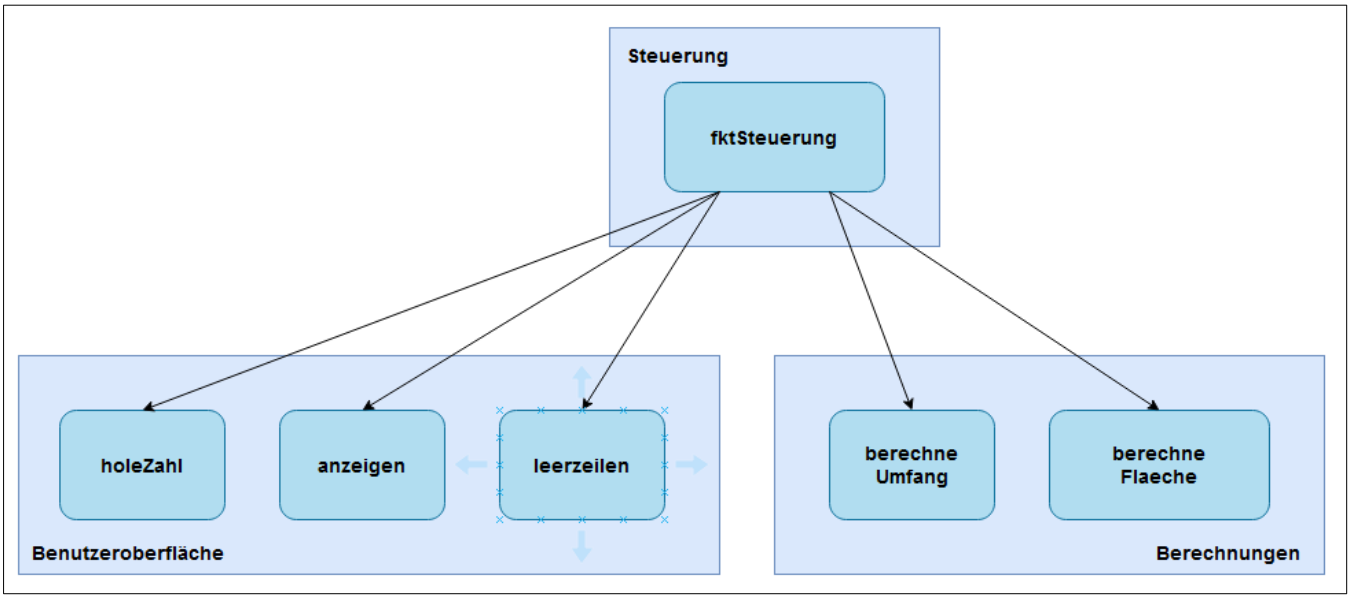



Individuelles Python-Programm
dbVerbindung Aufbauen
dbVerbindung Abbauen
dbAbfrage Anweisung
dbNichtAbfrage Anweisung
Python Connector
Individuelle Anwendung
Zugriffsfunktionen (Abstraktionsschicht)
Datenbank System
DB-Admin. Werkzeug
Datenbank Management System
DB 1
DB 2
DB 3
.....

LS 3 – Infos – Zerlegungsansätze

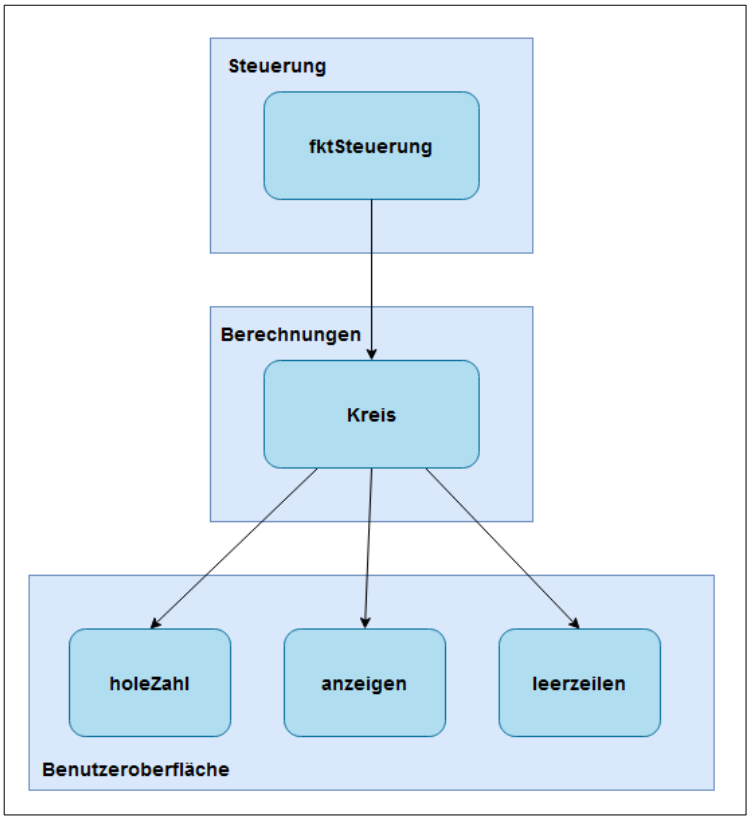
Ansatz A

`beispiel_ls3_04a.py`



Ansatz B

`beispiel_ls3_04b.py`



LS 3 – Infos – Resourcenverbrauch ermitteln

beispiel_ls3_05.py

```
import psutil as ps

def anzeigenResourcebVerbrauch():
    while True:
        cpuAuslastung = ps.cpu_percent(interval=1)
        print( "CPU-Auslastung:", cpuAuslastung, "%")
    return

anzeigenResourcebVerbrauch()
```

LS 3 – Infos – Datenbankzugriff modular isolieren