

## Inhaltsverzeichnis

1. Einführung und Ziele .....	3
1.1. Aufgabenstellung .....	3
1.2. Qualitätsziele .....	5
1.3. Stakeholder .....	6
2. Randbedingungen .....	7
2.1. Technische Randbedingungen .....	7
2.2. Organisatorische Randbedingungen .....	7
2.3. Konventionen .....	8
3. Kontextabgrenzung .....	9
3.1. Fachlicher Kontext .....	9
3.2. Technischer Kontext .....	9
4. Lösungsstrategie .....	10
5. Bausteinsicht .....	11
5.1. Whitebox Gesamtsystem .....	11
5.2. Ebene 2 .....	12
5.3. Ebene 3 .....	13
6. Laufzeitsicht .....	14
7. Verteilungssicht .....	15
7.1. Infrastruktur Ebene 1 .....	15
7.2. Infrastruktur Ebene 2 .....	15
8. Querschnittliche Konzepte .....	16
8.1. <Konzept 1> .....	16
8.2. <Konzept 2> .....	16
8.3. <Konzept n> .....	16
9. Entwurfsentscheidungen .....	17
10. Qualitätsanforderungen .....	18
10.1. Qualitätsbaum .....	18
10.2. Qualitätsszenarien .....	18

11. Risiken und technische Schulden .....	19
12. Glossar .....	20
13. Anhang .....	21
13.1. Apollo .....	21
13.2. Arc42 .....	21
13.3. docToolChain .....	21

## Über arc42

arc42, das Template zur Dokumentation von Software- und Systemarchitekturen.

Erstellt von Dr. Gernot Starke, Dr. Peter Hruschka und Mitwirkenden.

Template Revision: 7.0 DE (asciidoc-based), January 2017

© We acknowledge that this document uses material from the arc42 architecture template, <http://www.arc42.de>. Created by Dr. Peter Hruschka & Dr. Gernot Starke.



Diese Version des Templates enthält Hilfen und Erläuterungen. Sie dient der Einarbeitung in arc42 sowie dem Verständnis der Konzepte. Für die Dokumentation eigener System verwenden Sie besser die *plain* Version.

# 1. Einführung und Ziele

Dieser Abschnitt führt in die Aufgabenstellung ein und skizziert die Ziele, die Apollo Auto verfolgt.

## 1.1. Aufgabenstellung

### 1.1.1. Was ist Apollo Auto?

Apollo ist eine hochleistungsfähige, flexible Architektur, die die Entwicklung, das Testen und den Einsatz von autonomen Fahrzeugen beschleunigt. Apollo Auto bietet unter anderem Lösungen für Valet Parking, V2X-Kommunikation und intelligente Lichtsignalanlagen.

### 1.1.2. Wesentliche Features:

- Valet Parking
  - Software- und Hardware-Integrationslösung. Multifusionslösung bestand aus Fahrzeug, Cloud, HD-Karte und Parkplätzen
  - Bietet hochwertige Dienstleistungen, wie automatische Parkplatzerkennung und autonomes Parken, für Kunden.
- V2X-Kommunikation
  - Interaktionslösung für intelligente Fahrzeuginfrastruktur
  - Apollo V2X umfasst ein intelligentes Transportsystem für Fahrzeug Straßendatenerfassung und intelligente Verarbeitungsanalyse, Verkehrssicherheit und -effizienz
  - Wahrnehmung aller Verkehrsteilnehmer im Sichtfeld und die bereitgestellten straßenseitigen Sensorinformationen können für die Entscheidungsfindung beim autonomen Fahren auf hohem Niveau verwendet werden
  - Wahrnehmung von Verkehrsteilnehmern ausserhalb des Sichtfeldes

- Bietet einen vollständigen, kontinuierlichen, multimodalen Datendienst mit niedriger Latenz für L4-Autopilot-Fahrzeuge, die in mehreren Szenarien getestet wurden
- Durch die permanente dynamische Erfassung von Verkehrsinformationen und die Cloud-Integration, wird eine weltweite optimale kollaborative Steuerungsfunktionen für Verkehrsteilnehmer und Verkehrsmanagement erreicht
  - Smart Traffic Signals
- Holographisches Wahrnehmen und Verstehen, basierend auf dem holografischen Wahrnehmungs- und Erkennungssystem
- Status von Fußgängern und Fahrzeugen auf jeder Fahrspur genau erkennen und die Leistung des aktuellen Verkehrsflusses wie Volumen, Warteschlangenlänge, Verspätungen usw.
- Vollständige raum-zeitliche Ableitung und Entscheidungsfindung
- Echtzeitsteuerung der gesamten Szene
- Reduzierung der durchschnittlichen Wartezeit um 20-30% während der Rush Hour
  - Robotaxi
- Die Robotaxis, die aus Chinas erstem werkseitig installierten L4-Passagier-Fahrzeug sind zur Zeit auf öffentlichen Straßen im Testbetrieb
- Sie werden in Kooperation von Baido und FAW an einer gemeinsamen Produktionsline hergestellt
  - Minibus
- Die Minibusse ermöglichen ebenfalls autonomes Fahren der Stufe 4
- Funktionen sind unter Anderem Hinderniserkennung und -vermeidung, zu einem Zielort Fahren und Kreuzungen überqueren

## 1.2. Qualitätsziele

Die folgende Tabelle beschreibt die zentralen Qualitätsziele von DokChess, wobei die Reihenfolge eine grobe Orientierung bezüglich der Wichtigkeit vorgibt.

Qualitätsziel	Motivation und Erläuterung
<i>Zugängliches Beispiel (Analysierbarkeit)</i>	Apollo Auto ist eine offene Plattform, daher ist es wichtig, dass sich neue Entwickler möglichst schnell in die Architektur, Entwurf und Implementierung einarbeiten können
<i>Echzeitsteuerung von einzelnen Fahrzeugen und Verkehrströmen</i>	Apollo Auto übernimmt zuverlässig und sicher die autonome Steuerung von Fashrzeugen auf Level 4
<i>Echtzeit Umfelderkennung</i>	Für die Steuerung von Fahrzeugen wird ein Modell des Umfelds benötigt. Aus den Sensoprdaten wird ein digitales Abbild des Fahrweges, von beweglichen und unbeweglichen Hindernissen und von Signalen geschaffen
<i>Prediction</i>	Um die Fahrsicherheit weiter zu erhöhen, wird auf die Sensor- und Zustandsdaten von anderen Verkehrsteilnehmern in Echtzeit zugegriffen

### 1.3. Stakeholder

Die folgende Tabelle stellt die Stakeholder von Apollo Auto und ihre jeweilige Intention dar.

<b>Rolle</b>	<b>Interesse, Bezug</b>
<i>Softwarearchitekten</i>	Wollen ein Gefühl bekommen, wie Architekturdokumentation für ein konkretes System aussehen kann. Möchten sich Dinge (z.B. Form, Notation) für Ihre tägliche Arbeit abgucken. Gewinnen Sicherheit für Ihre eigenen Projekte.
<i>Entwickler</i>	Nehmen Architekturaufgaben im Team wahr. Brauchen ein generelles Verständnis für die Architektur.
<i>OEM &amp; Lieferanten</i>	Entwickeln neue Produkte auf Grundlage von Apollo Auto. Wollen Anregungen für eigene Produkte finden.
<i>Gesetzgeber &amp; Genehmigungsbehörden</i>	Entwickeln einen gesetzlichen Rahmen zur Zulassung von fahrerlosen Fahrzeugen im öffentlichen Straßenverkehr. Etablieren Prüfvorschriften und Tests für Genehmigungsverfahren.
<i>Universitäten</i>	Entwickeln eigene Forschungsprojekte auf Grundlage von Apollo Auto. Wollen Anregungen für weitere Forschungsprojekte und studentische Arbeiten finden.
<i>Studenten</i>	Interessieren sich aufgrund ihres Studiums für die verschiedenen Aspekte einer Architekturdokumentation. Setzen eigene Projekte (z.B. Masterarbeit) zum Thema autonomes Fahren mit Apollo Auto um. Schreiben eine Architekturdokumentation zu Apollo Auto.

## 2. Randbedingungen

Beim Einsatz von Apollo sind verschiedene Randbedingungen zu beachten. Dieser Abschnitt stellt sie dar und erklärt auch – wo nötig – deren Motivation.

### 2.1. Technische Randbedingungen

- Für den Einsatz von Apollo Auto wird eine anspruchsvolle Hardwareausstattung benötigt. Eine Umsetzung mit einem marktüblichen Standard-Notebook allein ist nicht möglich.
- Es wird ein Fahrzeug benötigt, dass mit By-Wire-Systemen ausgestattet ist, zum Beispiel Brake-by-Wire, Steering-by-Wire, Throttle-by-Wire oder Shift-by-Wire (Apollo wird derzeit auf Lincoln MKZ getestet).
- Ein Rechner mit einem 4-Kern-Prozessor und mindestens 8 GB Speicher (16 GB für Apollo 3.5 und höher)
- Ubuntu 18.04
- Zusätzlich wird eine umfangreiche Sensorik benötigt die Bild- und Abstandsinformationen aus dem Umfeld aufnehmen
- Arbeitskenntnisse über Docker

### 2.2. Organisatorische Randbedingungen

Randbedingung	Erläuterungen, Hintergrund
<i>github</i>	<i>Quellcode ist über github verfügbar.</i>
<i>Bereitstellung von Daten</i>	<i>Alle Daten müssen in einem Format hochgeladen werden, das den Apollo-Datenspezifikationen entspricht.</i>
<i>Speicherung von Daten</i>	<i>Daten, die in China gesammelt wurden, dürfen nur auf Servern in China gespeichert werden. Daten, die in anderen Ländern und Regionen erhoben werden, unterliegen den Beschränkungen der Datenspeicherung, die durch die Gesetze der jeweiligen Länder festgelegt sind.</i>

<b>Randbedingung</b>	<b>Erläuterungen, Hintergrund</b>
<i>Bereitstellung von Daten</i>	<i>Als Initiator dieser Plattform stellt Baidu die Ausgangsdaten für diese Plattform bereit. Die Daten stehen allen Partnern dieser Plattform offen. Das Prinzip der fairen Daten stellt sicher, dass Partner mit größeren eigenen Beiträgen mehr Daten und Dienste von dieser Plattform erhalten.</i>
<i>Datenschutz</i>	<i>Jeder Partner kann seine eigenen Daten anzeigen und die Datenschutzigenschaften der Daten als privat oder öffentlich festlegen. Die von Partnern hochgeladenen Daten gelten standardmäßig als privat.</i>
<i>Entscheidungsführung</i>	<i>Grundsätzlich handelt es sich bei Apollo Auto um eine offene Plattform. Allerdings wurde Baidu um die architektonische Integrität, die Systemzuverlässigkeit und die schnelle Entwicklung von Apollo zu gewährleisten, bei Bedarf wichtige Entscheidungen treffen, während die aktive Beteiligung der breiteren Gemeinschaft erhalten bleibt.</i>

## 2.3. Konventionen

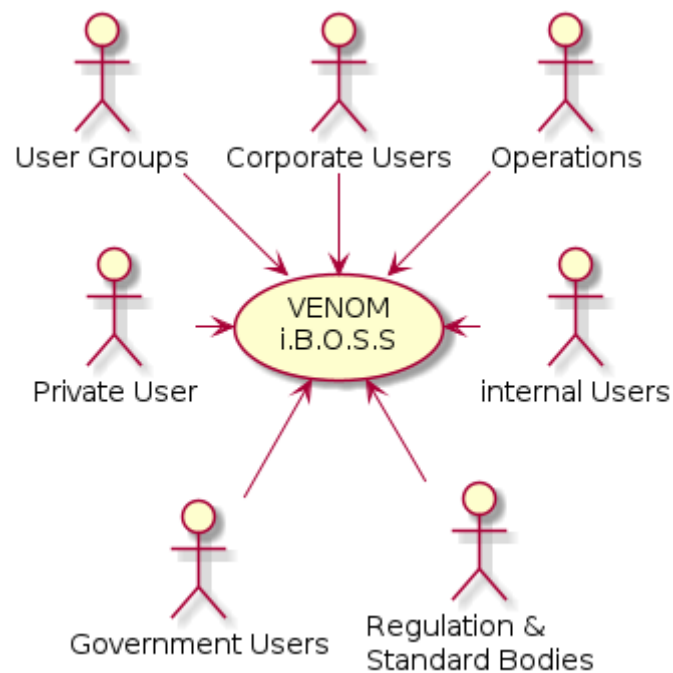
<b>Konvention</b>	<b>Erläuterungen, Hintergrund</b>
<i>Dokumentation</i>	<i>Terminologie und Gliederung nach dem deutschen arc42-Template in der Version 6.0</i>
<i>Kodierrichtlinien für C++</i>	<i>C++ Coding Conventions von Sun/Oracle, geprüft mit Hilfe von CheckStyle</i>
<i>Kodierrichtlinien für Python</i>	<i>Python Coding Conventions von Sun/Oracle, geprüft mit Hilfe von CheckStyle</i>
<i>Spezifische Datenformate und Frameworks für autonomes Fahren</i>	<i>Verwendung etablierter Standards für autonomes Fahren. zum Beispiel sind alle Softwaremodule als ROS(Robot Operating System)-Knoten zu behandeln.</i>



### 3. Kontextabgrenzung

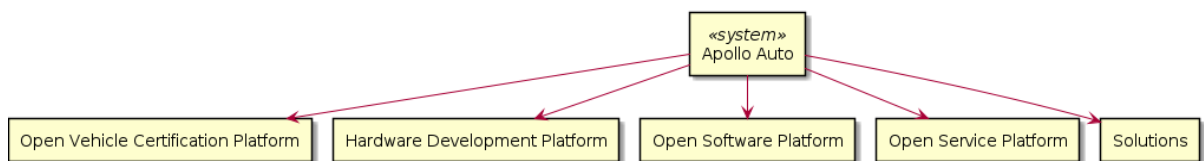
Dieser Abschnitt beschreibt das Umfeld von Apollo Auto. Für welche Benutzer ist es da, und mit welchen Fremdsystemen interagiert es?

#### 3.1. Fachlicher Kontext



*Abbildung 1. Benutzer und Benutzergruppen von VENOM*

#### 3.2. Technischer Kontext



*Abbildung 2. Technischer Kontext von Apollo Auto*

## 4. Lösungsstrategie

Dieser Abschnitt enthält einen stark verdichteten Architekturüberblick. Eine Gegenüberstellung der wichtigsten Ziele und Lösungsansätze.



## Enthaltene Bausteine

*<Beschreibung der enthaltenen Bausteine (Blackboxen)>*

## Wichtige Schnittstellen

*<Beschreibung wichtiger Schnittstellen>*

### 5.1.1. <Name Blackbox 1>

*<Zweck/Verantwortung>*

*<Schnittstelle(n)>*

*<(Optional) Qualitäts-/Leistungsmerkmale>*

*<(Optional) Ablageort/Datei(en)>*

*<(Optional) Erfüllte Anforderungen>*

*<(optional) Offene Punkte/Probleme/Risiken>*

### 5.1.2. <Name Blackbox 2>

*<Blackbox-Template>*

### 5.1.3. <Name Blackbox n>

*<Blackbox-Template>*

### 5.1.4. <Name Schnittstelle 1>

...

### 5.1.5. <Name Schnittstelle m>

## 5.2. Ebene 2

### 5.2.1. Whitebox <Baustein 1>

<Whitebox-Template>

### 5.2.2. Whitebox <Baustein 2>

<Whitebox-Template>

...

### 5.2.3. Whitebox <Baustein m>

<Whitebox-Template>

## 5.3. Ebene 3

### 5.3.1. Whitebox <\_Baustein x.1\_>

<Whitebox-Template>

### 5.3.2. Whitebox <\_Baustein x.2\_>

<Whitebox-Template>

### 5.3.3. Whitebox <\_Baustein y.1\_>

<Whitebox-Template>

## 6. Laufzeitsicht

Diese Sicht visualisiert im Gegensatz zur statischen Bausteinsicht dynamische Aspekte. Wie spielen die Teile zusammen?

## 7. Verteilungssicht

### 7.1. Infrastruktur Ebene 1

*<Übersichtsdiagramm>*

**Begründung**

*<Erläuternder Text>*

**Qualitäts- und/oder Leistungsmerkmale**

*<Erläuternder Text>*

**Zuordnung von Bausteinen zu Infrastruktur**

*<Beschreibung der Zuordnung>*

### 7.2. Infrastruktur Ebene 2

#### 7.2.1. *<Infrastrukturelement 1>*

*<Diagramm + Erläuterungen>*

#### 7.2.2. *<Infrastrukturelement 2>*

*<Diagramm + Erläuterungen>*

...

#### 7.2.3. *<Infrastrukturelement n>*

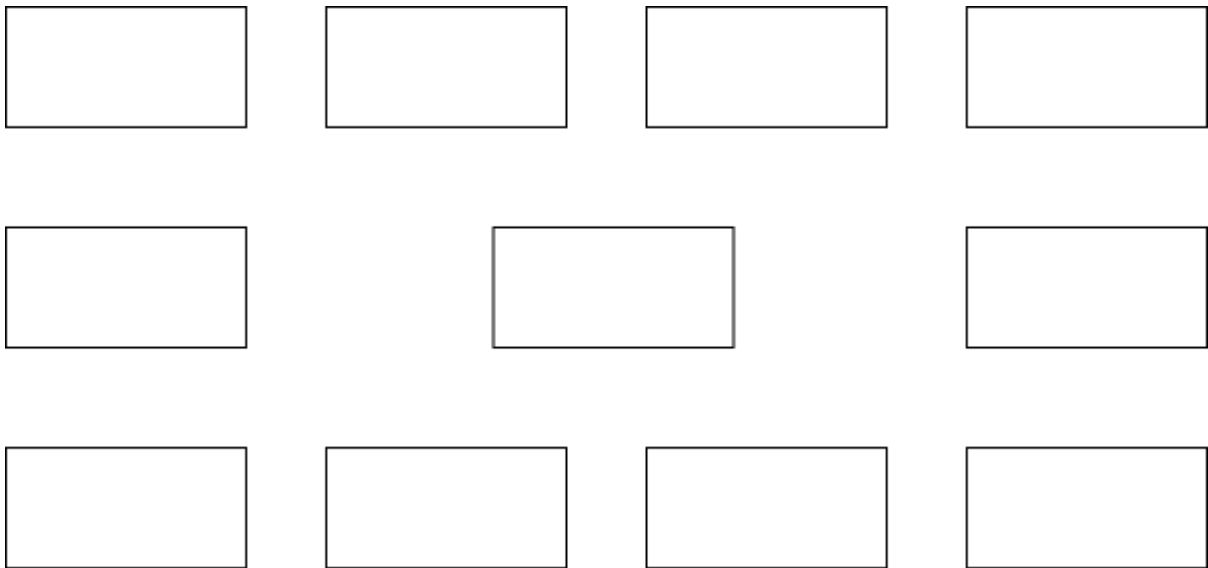
*<Diagramm + Erläuterungen>*

## 8. Querschnittliche Konzepte

Dieser Abschnitt beschreibt allgemeine Strukturen und Aspekte, die systemweit gelten. Darüber hinaus stellt er verschiedene technische Lösungskonzepte vor.

### 8.1. <Konzept 1>

<Erklärung>



### 8.2. <Konzept 2>

<Erklärung>

...

### 8.3. <Konzept n>

<Erklärung>



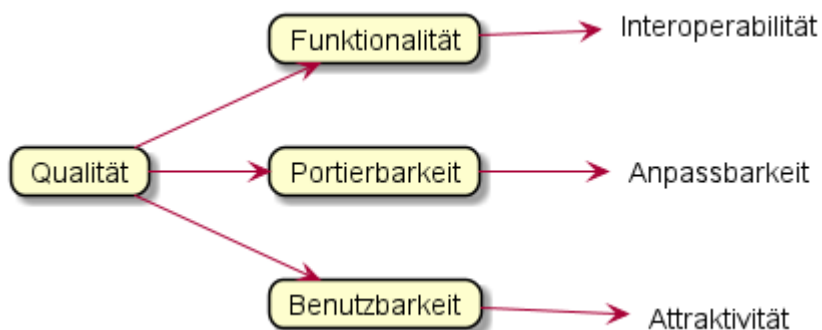
## 9. Entwurfsentscheidungen

## 10. Qualitätsanforderungen

Dieser Abschnitt beinhaltet konkrete Qualitätsszenarien, welche die zentralen Qualitätsziele, aber auch andere geforderte Qualitätseigenschaften besser fassen. Sie ermöglichen es, Entscheidungsoptionen zu bewerten.

### 10.1. Qualitätsbaum

Das folgende Bild gibt einen Überblick über die relevanten Qualitätsmerkmale und den ihnen jeweils zugeordneten Szenarien.



### 10.2. Qualitätsszenarien

## 11. Risiken und technische Schulden

Die folgenden Risiken wurden zu Beginn des Vorhabens identifiziert. Sie beeinflussten die Planung der ersten drei Iterationen maßgeblich. Seit Abschluss der dritten Iteration werden sie beherrscht. Dieser Architekturüberblick zeigt die Risiken inklusive der damaligen Eventualfallplanung weiterhin, wegen ihres großen Einflusses auf die Lösung.

### *Inhalt*

Eine nach Prioritäten geordnete Liste der erkannten Architekturrisiken und/oder technischen Schulden.

### *Motivation*

Risikomanagement ist Projektmanagement für Erwachsene.

— Tim Lister, Atlantic Systems Guild

Unter diesem Motto sollten Sie Architekturrisiken und/oder technische Schulden gezielt ermitteln, bewerten und Ihren Management-Stakeholdern (z.B. Projektleitung, Product-Owner) transparent machen.

### *Form*

Liste oder Tabelle von Risiken und/oder technischen Schulden, eventuell mit vorgeschlagenen Maßnahmen zur Risikovermeidung, Risikominimierung oder dem Abbau der technischen Schulden.

## 12. Glossar

Das folgende Glossar erklärt Begriffe aus dem Bereich Autonomes Fahren.

Begriff	Definition
<i>By-Wire</i>	<i>Bezeichnung für (zumindest partielles) Fahren oder Steuern von Fahrzeugen ohne mechanische Kraftübertragung der Bedienelemente zu den entsprechenden Stellelementen wie etwa Drosselklappen. Das By-Wire-Konzept umfasst dabei zumindest zwei oder mehr der „X-by-Wire“- Systeme wie etwa Brake-by-Wire (Bremssteuerung) und Steer-by-Wire (Lenkung)..</i>
<i>LIDAR</i>	<i>ist eine dem Radar verwandte Methode zur optischen Abstands- und Geschwindigkeitsmessung sowie zur Fernmessung. Statt der Radiowellen wie beim Radar werden Laserstrahlen verwendet.</i>
<i>RADAR</i>	<i>ist die Bezeichnung für verschiedene Erkennungs- und Ortungsverfahren und -geräte auf der Basis elektromagnetischer Wellen im Radiofrequenzbereich (Funkwellen).</i>

## 13. Anhang

### 13.1. Apollo

### 13.2. Arc42

- die Dokumentation muss bereits zu Beginn der Arbeiten an der Architektur/am Quellcode erfolgen um stets den aktuellen Stand abbilden zu können.
- die Dokumentation soll mit dem Projekt wachsen!
- die nachträgliche Dokumentation ist eine Sysiphusarbeit
  - Beispiel:
    - Diagramme in der Laufzeitsicht sind effizienter zu gestalten während ein Modul im Entstehen ist. Änderungen können mit jeder Entwicklungsstufe eingepflegt werden.
    - Ein fertiges Modul in ein Diagramm zu überführen erfordert viel Zeit, da alle beteiligten Module und deren Funktionsweise erst identifiziert werden müssen.

### 13.3. docToolChain

- plantUML ist zwar bei sehr einfachen Diagrammen eine Erleichterung, kommt aber an beispielsweise draw.io oder tikz (in Latex) nicht heran wenn es um komplexe und detaillierte Diagramme geht
  - teilweise werden bei der inline-Erstellung bei plantUML Diagrammeinstellungen nicht übernommen
- Schriftgrößen sind nicht einstellbar, was an manchen Stellen - z.B Tabellen nötig gewesen wäre
- Ansonsten ist docToolChain eine interessante Alternative, besonders in Bezug auf den CI-Task in github.
  - damit nicht bei jedem Commit die Dokumentation neu erstellt wird,

bietet github die Möglichkeit über Commitnachrichten, z.B. "[no ci]" oder "[skip ci]" den CI-Task auszusetzen. Hier wäre es sinnvoller über Commitnachrichten explizit die Generierung zu starten.