# Dokumentation [Kopf]

BÃ¶rse Stuttgart [Autor]

IT-Softwareentwicklung [Titel]

* [1. Einleitung](#_einleitung)
  + [1.1. Ziel des Dokuments](#_ziel_des_dokuments)
  + [1.2. Abgrenzung](#_abgrenzung)
  + [1.3. Konventionen](#_konventionen)
  + [1.4. Entwurfsentscheidungen](#_entwurfsentscheidungen)
* [2. Komponentenbeschreibung](#_komponentenbeschreibung)
  + [2.1. Allgemeine Beschreibung](#_allgemeine_beschreibung)
  + [2.2. Motivation](#_motivation)
  + [2.3. Form](#_form)
* [3. Baustein-Sicht und Laufzeit-Sicht](#_baustein_sicht_und_laufzeit_sicht)
  + [3.1. Baustein-Sicht](#_baustein_sicht)
  + [3.2. Laufzeit-Sicht](#_laufzeit_sicht)
  + [3.3. Returncodes und Nachrichten](#_returncodes_und_nachrichten)
* [4. User Interface](#_user_interface)
  + [4.1. MaskenÃ¼bersicht](#_maskenÃ¼bersicht)
  + [4.2. Beschreibung [Maske 1]](#_beschreibung_maske_1)
* [5. Betriebs-Sicht](#_betriebs_sicht)
  + [5.1. Konfigurationsparameter](#_konfigurationsparameter_2)
  + [5.2. Start- / Stoppverhalten](#_start_stoppverhalten)
* [6. Technische Konzepte](#_technische_konzepte)
  + [6.1. Persistenz](#_persistenz)
  + [6.2. Transaktionsbehandlung](#_transaktionsbehandlung)
  + [6.3. Skalierung und Clustering](#_skalierung_und_clustering)
  + [6.4. Parallelisierung und Threading](#_parallelisierung_und_threading)
  + [6.5. Ablaufsteuerung](#_ablaufsteuerung)
  + [6.6. Sicherheit](#_sicherheit)
  + [6.7. Konfigurierbarkeit](#_konfigurierbarkeit)
  + [6.8. Logging und Monitoring](#_logging_und_monitoring)
  + [6.9. Failover und HochverfÃ¼gbarkeitskonzept](#_failover_und_hochverfÃ¼gbarkeitskonzep)
  + [6.10. Testbarkeit](#_testbarkeit)
* [7. Entwurfsentscheidungen](#_entwurfsentscheidungen_2)
  + [7.1. Entwurfsentscheidung 1](#_entwurfsentscheidung_1)

**Komponentenname**  
System

Dokumenteninformation

|  |  |
| --- | --- |
| Dokumentart | Feinkonzept |
| Verfasser / Ansprechpartner |  |
| Zielgruppen |  |
| Status | <Auswahl Status> |
| Informationsklassifizierung | intern |
| Ablageort | <TFS-Pfad einfÃ¼gen $/> |

Ã„nderungshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Ã„nderung |
|  |  |  |  |

Referenzdokumente

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Dokumententitel | Quelle | Ã„nderung |
|  |  |  |  |

**Inhaltsverzeichnis**

## 1. Einleitung

**Gelb markierte Textstellen in dieser Vorlage sind vom Entwickler zu auszufÃ¼llen und dÃ¼rfen in der endgÃ¼ltigen Fassung nicht mehr vorhanden sein.**  
Wenn die im Dokument aufgefÃ¼hrten Kapitel nicht relevant sind, dann sind diese zu entfernen.  
Als Autor des Dokuments gebt ihr das Dokument nach Fertigstellung frei (Status â€žFreigegeben). Ein Dokument mit Status â€žFreigegebenâ€œ hat mindestens eine Version 1.0.

### 1.1. Ziel des Dokuments

Dieses Dokument beschreibt die Umsetzung der Komponente [Komponentenname] im Sinne einer Feinkonzeption.  
Mit diesem Dokument sollen Mitarbeiter in die Lage versetzt werden, die Aufgabe der Komponente im System zu verstehen.

### 1.2. Abgrenzung

Beschreibt hier, was das Dokument nicht leisten kann.

### 1.3. Konventionen

Folgende Konventionen werden im Dokument verwendet und sind besonders gekennzeichnet:

* Hinweis
* Achtung
* @todo - â€¦
  + Todos sind entsprechend gekennzeichnet und in der Regel gelb hinterlegt. In der finalen Version sollten keineTodos mehr vorhanden sein.

Beispiele:

| **Hinweis:** |
| --- |
| Der Datenbank-Zugriff erfolgt mittels Integrated Security. |

| **Achtung:** |
| --- |
| Der technische User muss betriebsseitig unter Security-Logins und Datenbank-Security-User mit entsprechenden Rechten vorgesehen sein. |

| **@todo:** |
| --- |
| Aktuelle Darstellung wird eingefÃ¼gt, wenn endgÃ¼ltige Fassung festgelegt ist. |

### 1.4. Entwurfsentscheidungen

Entwurfsentscheidungen werden im Rahmen der Umsetzung in Absprache zwischen dem (der) jeweiligen Entwickler(in) und dem (der) Architekt(in) getroffen und sind in der Architekturdokumentation hinterlegt.  
Werden Entscheidungen getroffen, welche nicht die Architektur betreffen, so sind diese im Kapitel 7 zu dokumentieren.

## 2. Komponentenbeschreibung

### 2.1. Allgemeine Beschreibung

Beschreibe hier in eigenen Worten was die Ausgangssituation ist, und wie eure Komponente zur LÃ¶sung der Aufgabenstellung beitrÃ¤gt bzw. was deren Funktion ist.  
Beispiel: Bloomberg-GeschÃ¤fte werden vom Handel manuell verbucht. Dies fÃ¼hrt hÃ¤ufig zu fehlerhaften Buchungen und hohem Zeitaufwand. Die Komponente [Komponentenname] soll diese GeschÃ¤fte entgegennehmen und automatisiert verbuchen.

ZusÃ¤tzlich kann auf das Gesamtsystem kurz eingegangen werden, damit der Kontext, in welchem die Komponente eingesetzt wird, etwas klarer wird.

### 2.2. Motivation

Gibt es besondere BeweggrÃ¼nde fÃ¼r die Umsetzung dieser Komponente. LÃ¶st z.B. diese Komponente eine Ã¤ltere ab (falls ja, warum)?

### 2.3. Form

Beschreibe welche zentralen AblÃ¤ufe im Backend bzw. Front-end abgebildet werden. Welche AusprÃ¤gungen (z. B. Front-end) werden bereitgestellt?

## 3. Baustein-Sicht und Laufzeit-Sicht

Hier sollen die jeweiligen internen Module (Schnittstellen / FunktionsblÃ¶cke) der Komponente beschrieben werden. Dabei wird von der Blackbox-Sicht zur Whitebox-Sicht vorgegangen.

FÃ¼r die Baustein-Sicht kÃ¶nnen die folgenden UML2-Diagrammtypen verwendet werden:

1. Komponentendiagramm
2. Abstraktes Klassendiagramm (nur ganz High-Level)
3. Paketdiagramm

Weiterhin sollen hier fÃ¼r die Komponente Laufzeit-Sichten beschrieben werden.  
In der Laufzeit-Sicht werden AblÃ¤ufe, die zur Laufzeit der Komponente stattfinden, beschrieben werden.  
Zur Beschreibung bieten sich die folgenden UML2-Diagrammtypen an

1. AktivitÃ¤tsdiagramm
2. Sequenzdiagramm
3. Statusdiagramm

### 3.1. Baustein-Sicht

Die folgende Abbildung zeigt die Hauptbausteine und Schnittstellen der Komponente und deren AbhÃ¤ngigkeiten.  
**Beispiel (<hier Ãœberblicksdiagramm einfÃ¼gen>):**

**Error! Filename not specified.**

Figure 1. Hauptbausteine und Schnittstellen der Komponente und deren AbhÃ¤ngigkeiten

Nachfolgend werden die einzelnen Kommunikationsschnittstellen und Bausteine dann detailliert beschrieben.

#### 3.1.1. [Schnittstelle 1]

##### Zweck / Verantwortlichkeit

Welchem Zweck dient die Schnittstelle  
Wer verwendet die Schnittstelle bzw. wer ist an der Schnittstelle verbunden

##### ErfÃ¼llte Anforderungen

Verweis auf Anforderungen aus dem Anforderungskatalog / Fachkonzept

##### Eigenschaften

Handelt es sich um eine Kommandoschnittstelle (Request / Response), eine Subscribe- / Publish-Schnittstelle (z.B. AIS) oder eine reine Pushschnittstelle.

##### Konfigurationsparameter

Sofern sie von den Standardparametern des CSF abweichen bzw. erweiterte Parameter verwendet werden.  
Hier auch die vom Standard abweichenden Parameter fÃ¼r Buffersize, Queuesize, etc. des CSF Frameworks eintragen. Die Parameter sind so zu definieren, dass sie Anforderungen an die Schnittstelle genÃ¼gen.  
Name der TP Datei.

##### Formatbeschreibung

|  |
| --- |
| **Hinweis** Hier sind IDS feldorientierte, oder komplexe Datenstrukturen (z.B. XML) Schnittstellen exakt zu beschrieben. Beispiele fÃ¼r DatensÃ¤tze sind hier anzugeben. |

Bei feldorientierten Schnittstellen

1. Feldbezeichnung
2. Verwendungszweck
3. Wertebereich

Die Darstellung kann in tabellarischer Form erfolgen:

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1. Felder Schnittstelle abc | | | |
| Feld-Name | Datentyp | Wertebereich | Verwendungszweck |
|  |  |  |  |
|  |  |  |  |

Bei Schnittstellen mit komplexen Datenstrukturen (z.B. Kommandoschnittstelle)

1. Name des Kommandos
2. Beschreibung derRequestdaten
   1. Datenstruktur mit allen Eigenschaften
3. Beschreibung der Responsedaten
   1. Returncodes
   2. Datenstruktur mit allen Eigenschaften
4. Verweis auf Quellcodedokumentation

##### Protokollbeschreibung

Welches fachliche Protokoll wird verwendet (IDS, Command,â€¦)  
Welche Art der Serialisierung wird verwendet (XML, IDS, â€¦ )  
Welches technische Protokoll wird verwendet (TCP/IP, UDP, Named Pipe, Unicast, Multicast)

##### Leistungsmerkmale

Anzahl Pakete pro Sekunde maximal  
Maximale DatensatzlÃ¤nge

#### 3.1.2. [Bausteinname 1]

< Hier Ãœberblicksdiagramm fÃ¼r Innenleben von Baustein 1 einfÃ¼gen>

1. zeigt das Innenleben des Bausteines in Diagrammform mit den lokalen Bausteinen 1 - n, sowie deren ZusammenhÃ¤nge und AbhÃ¤ngigkeiten.

Wenn das Verhalten des Bausteins Ã¼ber Konfigurationsparameter gesteuert werden kann, Verweis auf Kapitel 5.1 (Konfigurationsparameter) einfÃ¼gen

##### Zweck / Verantwortlichkeit

##### ErfÃ¼llte Anforderungen

##### Leistungsmerkmale

### 3.2. Laufzeit-Sicht

Hier kann z.B. der Nachrichtenfluss Ã¼ber mehrere Schnittstellen und Bausteine hinweg beschrieben werden.  
Zur Beschreibung bieten sich Sequenzdiagramme, AktivitÃ¤ts- oder Statusdiagramme an. Es ist hilfreich, zu einem Diagramm ein paar erklÃ¤rende Worte zu schreiben.

Steuerung der Komponente  
Bei Komponenten, die Ã¼ber eine Kommandoschnittstelle gesteuert werden, muss hier erkennbar sein, was durch das Kommando passiert (EVA Prinzip).  
Jedes Kommando muss beschrieben werden (Request / Response).  
Alle Returncodes mÃ¼ssen beschrieben sein.  
Werden innerhalb der Komponente Ã¼ber die Schnittstelle Nachrichten verschickt sind diese ebenfalls zu beschreiben. Es muss erkennbar sein, in welchen FÃ¤llen die Nachricht verschickt wird (was lÃ¶st die Nachricht aus).  
Zu jedem beschriebenen Kommando und zu jeder beschriebenen Nachricht muss ein Link auf das entsprechende Schnittstellenkapitel eingefÃ¼gt werden.

Interne AblÃ¤ufe / Workflows  
Jeder interne Ablauf, der nicht direkt durch ein Kommando ausgelÃ¶st wird, muss beschrieben werden. Die jeweiligen StatusÃ¼bergÃ¤nge fÃ¼r jeden Workflowschritt sind zu beschreiben.  
Die Beschreibung kann mit Hilfe des UML2 Activity Diagramms erfolgen. Die Abbruchbedingungen und die daraus resultierenden Aktionen (mit Returncodes) mÃ¼ssen erkennbar sein.

### 3.3. Returncodes und Nachrichten

Alle in der Komponente verwendeten Returncodes und Nachrichten, die Ã¼ber eine Schnittstelle nach auÃŸen gegeben werden, sind als Ãœbersicht zu dokumentieren

|  |  |  |
| --- | --- | --- |
| Table 2. Messages und Codes | | |
| Returncode | Nachricht (Text) | AusgelÃ¶ste Aktion(en) |
| 999 | Interner Fehler in Modul 0815 aufgetreten | Eintrag im Logfile als ErrorVersenden einer Nachricht vom Type ErrorNotification Ã¼ber die Schnittstelle xyz |
|  |  |  |

## 4. User Interface

### 4.1. MaskenÃ¼bersicht

Hier eine Struktursicht der GUI einfÃ¼gen, auf der die verwendeten Masken namentlich erwÃ¤hnt werden und der grobe Zusammenhang sichtbar wird.

### 4.2. Beschreibung [Maske 1]

Screenshot der Maske mit allen Steuerelementen.  
Jedes Steuerelement benennen und beschreiben

1. Eingabefelder
2. Verwendungszweck
3. Erlaubter Wertebereich
4. Verhalten bei Fehleingabe
5. Buttons
6. Verwendungszweck
7. Aktion die durch Button ausgelÃ¶st wird
   1. Beschreibung Ã¼ber Ablaufdiagramm oder Sequenzdiagramm

Wenn das Verhalten der Maske Ã¼ber Konfigurationsparameter gesteuert werden kann, Verweis auf Kapitel 5.1 (Konfigurationsparameter) einfÃ¼gen

## 5. Betriebs-Sicht

In der Betriebs-Sicht sollten Themen wie generelle KonfigurationsmÃ¶glichkeiten und Parameter, StartabhÃ¤ngigkeiten, LaufzeitabhÃ¤ngigkeiten, sowie sonstige betriebsrelevanten Informationen beschreiben werden.  
FÃ¼r Start / Stopp des Systems kÃ¶nnen Sequenzdiagramme verwendet werden.

### 5.1. Konfigurationsparameter

Hier werden alle Konfigurationsparameter Ã¼ber die Standard C/S-Frameworkparameter hinaus beschrieben.  
Dies gilt auch fÃ¼r Parameter fÃ¼r â€žhidden functionalityâ€œ, also Parameter, die fÃ¼r Test- oder Debugzwecke verwendet werden und die das Verhalten der Komponente beeinflussen.  
Werden weitere Konfigurationsdateien verwendet, wie die app.config, so sind diese vollumfÃ¤nglich zu dokumentieren.  
Werden Softwarekomponenten eingesetzt, die eine entsprechende Konfiguration mitbringen, mÃ¼ssen diese hier ebenfalls fÃ¼r die Komponente beschrieben werden.

Folgende Form der Darstellung kann verwendet werden:

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | WertebereichAusprÃ¤gung | Verwendungszweck | Beispiel |
| MinMaxParameter | -1 bis 9999 | Wird in der [Komponente] verwendet fÃ¼r Festlegung der Unter- / Obergrenze.Das Trennzeichen zwischen Minimum und Maximum ist das Pipe Symbol â€š|â€˜ | <MinMaxParameter>1|999</MinMaxParameter> |
|  |  |  |  |

### 5.2. Start- / Stoppverhalten

Wie verhÃ¤lt sich die Komponente beim Start?  
Gibt es Kommandozeilenparameter?  
Welche Komponenten mÃ¼ssen vorher gestartet sein?  
Welche Bedingungen mÃ¼ssen erfÃ¼llt sein, damit die Komponente startet?  
Wie verhÃ¤lt sich die Komponente im Shutdown?

## 6. Technische Konzepte

### 6.1. Persistenz

Wie werden Daten persistiert (Datei, Datenbank, â€¦ )?  
Welche Art von Datenbank (relational, Objektorientiert)?  
Was wird wie persistiert?  
Was wird wann persistiert?

### 6.2. Transaktionsbehandlung

Wie werden Transaktionen durchgefÃ¼hrt und abgesichert (nicht nur Datenbanktransaktionen?  
hier auch an Kommandos im Request â€“ Response Verfahren denken)

### 6.3. Skalierung und Clustering

Kann die Komponente skaliert werden und wie?

### 6.4. Parallelisierung und Threading

Welche AblÃ¤ufe laufen parallel und welche Synchronisationsmechanismen gibt es?  
Welche Konsequenzen ergeben sich bei AsynchronitÃ¤ten ?

### 6.5. Ablaufsteuerung

Bei Verwendung einer Rule-Engine oder eines Workflow-Managements kann hier beschrieben werden, wie und wo die Ablaufsteuerung eingesetzt wird.

### 6.6. Sicherheit

Falls zutreffend: Berechtigungen werden Ã¼ber AD-Gruppen geregelt und sind im Komponentenhandbuch beschrieben.  
Gibt es weitere Sicherheitsmechanismen bzw. Zugriffsschutz?

### 6.7. Konfigurierbarkeit

Wie, wann (zur Laufzeit, Restart notwendig, â€¦ )?

### 6.8. Logging und Monitoring

Was wird wie und wann protokolliert?

1. Log / Trace
2. Technisches Log vs. Fachliches Log
3. Steuerung desLoggingmechanismus
4. Benennung der wichtigsten Logger (bei Log4Net) und kurze Beschreibung welche Informationen angezeigt werden

### 6.9. Failover und HochverfÃ¼gbarkeitskonzept

Wie verhÃ¤lt sich die Komponente im Fehlerfall?  
Welche Situationen fÃ¼hren zu einem Abbruch der Komponente?  
Gibt es Situationen in denen die Komponente eingeschrÃ¤nkt funktionsfÃ¤hig ist?  
Wie verhÃ¤lt sich die Komponente im Fehlerfall beim Umschalten auf eine Ersatzadresse?  
Wie ist das Wiederanlaufverhalten der Komponente nach einem Fehler?  
Wie verhÃ¤lt sich die Komponente bei einem Verbindungsabbruch zu den jeweiligen Schnittstellen wÃ¤hrend der Initialisierungsphase sowie danach?  
Wie verhÃ¤lt sich die Komponente bei einem Timeout (beim Warten auf den Empfang von Daten, Request Timeout, o.Ã¤.)?

### 6.10. Testbarkeit

Wie kann die Komponente getestet werden?  
Gibt es Testschnittstellen?  
Gibt es Testtreiber / Testtools?  
Gibt es spezielle Messdaten und Messpunkte?

## 7. Entwurfsentscheidungen

### 7.1. Entwurfsentscheidung 1

#### 7.1.1. Fragestellung

#### 7.1.2. Rahmenbedingungen

#### 7.1.3. Annahmen

#### 7.1.4. Betrachtete Alternativen

#### 7.1.5. Entscheidung

**Abbildungsverzeichnis**

**Tabellenverzeichnis**