**System Architecture Specification  
(Architekturspezifikation)**

**(TIT10AID, SWE I Praxisprojekt 2011/2012)**

**Projekt:** *Hier den Namen des Projektes eingeben!*

**Auftraggeber:** *Name des Auftraggebers*

*Adresse*

*Telefon / Fax*

*Ansprechpartner*

**Auftragnehmer:** *Name des Auftragnehmers*

*Adresse*

*Telefon / Fax*

*Ansprechpartner*

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Autor** | **Kommentar** |
| 0.1 | 01.01.1970 |  | Dokument angelegt |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

***Allgemeine Hinweise:***

*Alles, was in dieser Schriftart gesetzt ist, dient nur zur Erläuterung und sollte im fertigen Pflichtenheft nicht mehr auftauchen!*

*Eine Architekturspezifikation (SAS) enthält die Beschreibung, wie die fachlichen Anforderungen, die das zu entwickelnde Produkt laut Pflichtenheft (SRS) erfüllen muss, realisiert werden sollen. Die Inhalte der SAS dienen als Grundlage für die Subsytem- bzw. Modulspezifikationen (MOD)s. Im Gegensatz zum SRS –welches die Black-Box-Sicht des Systems beschreibt- ist im SAS die White-Box-Sicht auf übergeordneter struktureller Ebene aufzuführen.*

*Es soll dabei ein Dekompositionsansatz der folgenden Art verwendet werden (Quelle: http://arc42.de):*

*Wichtig ist dabei die weitgehende Beibehaltung der Anforderungs-Traceability SRS –> SAS –> MOD, d.h. es muss identifierbar sein, welche MODs welche Anforderungen aus der SRS realisieren sollen.*

*In der SAS können (und sollen) Diagramme und verschiedene Abbildungen verwendet werden. Diagramme und Abbildungen ohne erläuternden Text sind jedoch wertlos. Bitte achten Sie daher darauf, dass jede Abbildung im Text referenziert und auch erklärt wird. Das Verhältnis zwischen erläuternden Texten und Abbildungen sollte möglichst ausgewogen sein, d.h. eine SAS mit 30 Seiten darf maximal 15 Seiten Abbildungen enthalten.*

[*http://arc42.de/*](http://arc42.de/)

[*http://v-modell.iabg.de/v-modell-xt-html/15443f771c1e066.html*](http://v-modell.iabg.de/v-modell-xt-html/15443f771c1e066.html)

*http://www.bredemeyer.com/papers.htm*

[Inhalt](http://inhalt)

**1.** **Einführung 5**

1.1. Glossar 5

**2.** **Systemübersicht 6**

2.1. Systemumgebung 6

2.2. Hardwareumgebung 6

2.3. Softwareumgebung 6

**3.** **Architekturkonzept 7**

3.1. Architekturziele 7

3.2. Architekturmodell 7

**4.** **Systemdesign 8**

**5.** **Subsystemspezifikation 9**

5.1. <MOD-ID 1>: <Name des Moduls> 9

**6.** **Technische Konzepte 10**

6.1. Persistenz 10

6.2. Benutzungsoberfläche 10

6.3. Ergonomie 10

6.4. Ablaufsteuerung 10

6.5. Transaktionsbehandlung 10

6.6. Sessionbehandlung 11

6.7. Kommunikation und Integration mit anderen IT-Systemen 11

6.8. Verteilung 11

6.9. Plausibilisierung und Validierung 11

6.10. Ausnahme-/Fehlerbehandlung 11

6.11. Logging, Protokollierung, Tracing 11

6.12. Konfigurierbarkeit 12

6.13. Parallelisierung und Threading 12

6.14. Internationalisierung 12

6.15. Migration 12

6.16. Testbarkeit 12

6.17. Skalierung, Clustering 12

6.18. Hochverfügbarkeit 13

**Referenzen 14**

**Anhang 15**

# Einführung

*Dieser Abschnitt hat die Aufgabe, als Einleitung zu dienen. Informelle Beschreibung des Systems und seine technische Umgebung, die wesentlichen Einfluss auf die konkrete Lösung haben wird.*

## Glossar

*Fachbegriffe, Abkürzungen: Erläuterung mit maximal 3 kurzen Sätzen.*

# Systemübersicht

*Beschreibung des Systems aus Black-Box-Sicht.*

## Systemumgebung

*Festlegung aller Nachbarsysteme des betrachteten Systems mit Spezifikation aller fachlichen Daten, die mit diesen ausgetauscht werden. Zusätzlich evtl. Datenformate und Protokolle der Kommunikation mit Nachbarsystemen und der Umwelt (falls diese nicht erst bei den spezifischen Bausteinen präzisiert wird.*

*Darstellung durch Diagrammarten, die das System als Black Box darstellen und die Schnittstellen zu den Nachbarsystemen (mehr oder weniger ausführlich) beschreiben.*

***Durch eigenes Diagramm ersetzen.***

*Abbildung 3.1: Diagramm für die Systemumgebung*

## Hardwareumgebung

*Angaben über die existierenden oder zu erwartenden Hardwareumgebungen.*

## Softwareumgebung

*In diesem Abschnitt werden Angaben zur Softwareumgebung des zu entwickelnden Produktes gemacht. Insbesondere das Betriebssystem und zur Verfügung stehende Laufzeitumgebungen /Bibliotheken sind wichtig. Andere Systeme mit denen das zu entwickelnde Produkt kooperieren muss, sollten möglichst genau spezifiziert sein.*

# Architekturkonzept

*Bspw. statische Zerlegung des Systems in Bausteine (Module, Komponenten, Subsysteme, Teilsysteme, Klassen, Interfaces, Pakete, Bibliotheken, Frameworks, Schichten, Partitionen, Tiers, Funktionen, Makros, Operationen, Datenstrukturen...).*

## Architekturziele

*Im Folgenden werden die Qualitätsziele aufgelistet, die mit Hilfe der Architektur erreicht werden sollen, z.B. Modularität, Wiederverwendbarkeit etc.. Im DIN/ISO 9126 Standard findet sich eine umfangreiche Sammlung möglicher Qualitätsziele.*

### Architekturziel 1

*Beschreibung, wie das genannte Qualitätsziel angestrebt werden soll.*

## Architekturmodell

*Aufgabe dieses Abschnittes ist es, die verwendeten Architekturkonzepte abstrakt darzustellen (z.B. Schichtenmodell, Client-Server-Modell, Model-View-Controller etc.).*

*Begründung aufführen, warum an bestimmten Stellen eine bestimmte Art der Modellierung gewählt wurde.*

***Hier könnte eine Graphik stehen!***

*Abbildung 2.1: Graphik zur Illustration des Architekturmodells*

# Systemdesign

*Dieser Abschnitt verfeinert den vorherigen. Durch die Verwendung eines graphischen Modells (z.B. Blockdiagramm, Verteilungsdiagramm und/oder Klassendiagramm) sollen die Zusammenhänge zwischen dem abstrakten Modell und der konkreten angestrebten Realisierung präzisiert und übersichtlich dargestellt werden.*

*Dieser Abschnitt hat die Aufgabe, die Übersicht über die Architektur des zu entwickelnden oder zu erweiternden Systems zu beschreiben. Es folgt die Aufführung aller relevanten Systemmodule, deren Zuständigkeiten, Verhalten und Schnittstellen, als Basis für eine weitere Verfeinerung des Entwurfs, welche später detailliert in den Modulspezifikationen (MODs) beschrieben wird.*

***Durch eigenes Klassendiagramm ersetzen.***

*Abbildung 2.2: Klassendiagramm für das Systemdesign*

*Das Systemdesign identifiziert bereits eindeutig die nachfolgend aufzuführenden Subsysteme (Module).*

# Subsystemspezifikation

*Nähere Beschreibung der relevanten –zu entwickelnden oder zu ändernden- Subsysteme bzw. Systemkomponenten aus Black-Box-Sicht. Spätere Verfeinerung und Darstellung der White-Box-Sicht erfolgt in der jeweils zugehörigen Moduldokumentation (MOD).*

## <MOD-ID 1>: <Name des Moduls>

*Informelle Beschreibung des Subsystems.*

### Übersicht

|  |  |
| --- | --- |
| ***<MOD-ID>*** | *<Identifkation des Moduls, Name des Moduls>* |
| ***Abgedeckte Systemanforderungen:*** | *<Auflistung der zugehörigen Requirement-IDs aus SRS>* |
| ***Leistung:*** | *<Was das Modul tun soll >* |
| ***Schnittstellen:*** | *<Auflistung der relevanten Kontrollschnittstellen des Moduls.>* |
| ***Externe Daten:*** | *<Auflistung der relevanten Datenschnittstellen des Moduls.>* |
| ***Ablageort:*** | *<Wo liegen die zugehörigen Moduldateien ?>* |
|  |  |
| ***Offene Punkte:*** | *<Was ist noch zu klären ?>* |
|  |  |

***Durch eigenes Diagramm ersetzen.***

*Abbildung 2.4: Illustration von <MOD-ID> z.B. durch Klassen-, Zustands-, Aktivitäten- oder Sequenzdiagramm*

# Technische Konzepte

*Die folgenden Kapitel sind Beispiele für oft benötigte querschnittliche technische Aspekte. Füllen Sie die Kapitel mit Inhalt, wenn es KEINEN Baustein gibt, der diesen Aspekt abdeckt.*

*Motivation:*

*Manche der Aspekte lassen sich nicht "zentral" als Baustein in der Architektur unterbringen. Hier ist der Platz im Template, wo Sie Konzepte zu derartigen Themen geschlossen behandeln können.*

## Persistenz

*Persistenz (Dauerhaftigkeit, Beständigkeit) bedeutet, Daten aus dem (flüchtigen) Hauptspeicher auf ein beständiges Medium (und wieder zurück) zu bringen. Einige der Daten, die ein Software-System bearbeitet, müssen dauerhaft auf einem Speichermedium gespeichert oder von solchen Medien gelesen werden:*

*• Flüchtige Speichermedien (Hauptspeicher oder Cache) sind technisch nicht für dauerhafte Speicherung ausgelegt. Daten gehen verloren, wenn die entsprechende Hardware ausgeschaltet oder heruntergefahren wird.*

*• Die Menge der von kommerziellen Software-Systemen bearbeiteten Daten übersteigt üblicherweise die Kapazität des Hauptspeichers.*

*• Auf Festplatten, optischen Speichermedien oder Bändern sind oftmals große Mengen von Unternehmensdaten vorhanden, die eine beträchtliche Investition darstellen.*

*Persistenz ist ein technisch bedingtes Thema und trägt nichts zur eigentlichen Fachlichkeit eines Systems bei. Dennoch müssen Sie sich als Architekt mit dem Thema auseinander setzen, denn ein erheblicher Teil aller Software-Systeme benötigt einen effizienten Zugriff auf persistent gespeicherte Daten. Hierzu gehören praktisch sämtliche kommerziellen und viele technischen Systeme. Eingebettete Systeme (embedded systems ) gehorchen jedoch oft anderen Regeln hinsichtlich ihrer Datenverwaltung.*

## Benutzungsoberfläche

*IT-Systeme, die von (menschlichen) Benutzern interaktiv genutzt werden, benötigen eine Benutzungsoberfläche. Das können sowohl grafische als auch textuelle Oberflächen sein.*

## Ergonomie

*Ergonomie von IT-Systemen bedeutet die Verbesserung (Optimierung) deren Benutzbarkeit aufgrund objektiver und subjektiver Faktoren. Im wesentlichen zählen zu ergonomischen Faktoren die Benutzungsoberfläche, die Reaktivität (gefühlte Performance) sowie die Verfügbarkeit und Robustheit eines Systems.*

## Ablaufsteuerung

*Ablaufsteuerung von IT-Systemen bezieht sich sowohl auf die an der (grafischen) Oberfläche sichtbaren Abläufe als auch auf die Steuerung der Hintergrundaktivitäten. Zur Ablaufsteuerung gehört daher unter anderem die Steuerung der Benutzungsoberfläche als auch die Workflow-Steuerung.*

## Transaktionsbehandlung

*Transaktionen sind Arbeitsschritte oder Abläufe, die entweder alle gemeinsam oder gar nicht durchgeführt werden. Der Begriff stammt aus den Datenbanken - wichtiges Stichwort hier sind ACID-Transaktionen (atomar, consistent, isolated, durable).*

## Sessionbehandlung

*Eine Session, auch genannt Sitzung, bezeichnet eine stehende Verbindung eines Clients mit einem Server. Den Zustand dieser Sitzung gilt es zu erhalten, was insbesondere bei der Nutzung zustandsloser Protokolle (etwa HTTP) wichtige Bedeutung hat. Sessionbehandlung stellt für Intraund Internetsysteme eine kritische Herausforderung dar und beeinflusst häufig die Performance eines Systems.*

## Kommunikation und Integration mit anderen IT-Systemen

*Kommunikation: Übertragung von Daten zwischen System-Komponenten. Bezieht sich auf Kommunikation innerhalb eines Prozesses oder Adressraumes, zwischen unterschiedlichen Prozessen oder auch zwischen unterschiedlichen Rechnersystemen.*

*Integration: Einbindung bestehender Systeme (in einen neuen Kontext). Auch bekannt als: (Legacy) Wrapper, Gateway, Enterprise Application Integration (EAI).*

## Verteilung

*Verteilung: Entwurf von Software-Systemen, deren Bestandteile auf unterschiedlichen und eventuell physikalisch getrennten Rechnersystemen ablaufen.*

*Zur Verteilung gehören Dinge wie der Aufruf entfernter Methoden (remote procedure call, RPC), die Übertragung von Daten oder Dokumenten an verteilte Kommunikationspartner, die Wahl passender Interaktionsstile oder Nachrichtenaustauschmuster (etwa: synchron / asynchron, publish- subsribe, peer-to- peer).*

## Plausibilisierung und Validierung

*Wo und wie plausibilisieren und validieren Sie (Eingabe-)daten, etwa Benutzereingaben?*

## Ausnahme-/Fehlerbehandlung

*Wie werden Programmfehler und Ausnahmen systematisch und konsistent behandelt? Wie kann das System nach einem Fehler wieder in einen konsistenten Zustand gelangen? Geschieht dies automatisch oder ist manueller Eingriff erforderlich? Dieser Aspekt hat mit Logging, Protokollierung und Tracing zu tun.*

*Welche Art Ausnahmen und Fehler behandelt ihr System? Welche Art Ausnahmen werden an welche Außenschnittstelle weitergeleitet und welche Ausnahmen behandelt das System komplett intern? Wie nutzen Sie die Exception-Handling Mechanismen ihrer Programmiersprache? Verwenden Sie checked- oder unchecked-Exceptions?*

## Logging, Protokollierung, Tracing

*Es gibt zwei Ausprägungen der Protokollierung, das Logging und das Tracing . Bei beiden werden Funktions- oder Methodenaufrufe in das Programm aufgenommen, die zur Laufzeit über den Status des Programms Auskunft geben.*

*In der Praxis gibt es zwischen Logging und Tracing allerdings sehr wohl Unterschiede:*

*- Logging kann fachliche oder technische Protokollierung sein, oder eine beliebige Kombination von beidem.*

*- Fachliche Protokolle werden gewöhnlich anwenderspezifisch aufbereitet und übersetzt. Sie dienen Endbenutzern, Administratoren oder Betreibern von Softwaresystemen und liefern Informationen über die vom Programm abgewickelten Geschäftsprozesse.*

*- Technische Protokolle sind Informationen für Betreiber oder Entwickler. Sie dienen der Fehlersuche sowie der Systemoptimierung.*

*- Tracing soll Debugging -Information für Entwickler oder Supportmitarbeiter liefern. Es dient primär zur Fehlersuche und -analyse.*

## Konfigurierbarkeit

*Die Flexibilität von IT-Systemem wird unter anderem durch ihre Konfigurierbarkeit beeinflusst, die Möglichkeit, manche Entscheidungen hinsichtlich der Systemnutzung erst spät zu treffen. Konfigurierbarkeit kann zu folgenden Zeitpunkten erfolgen:*

*- Während der Programmierung: Dabei werden beispielsweise Server-, Datei- oder Verzeichnisnamen direkt ("hart") in den Programmcode aufgenommen.*

*- Während des Deployments oder der Installation: Hier werden Konfigurationsinformationen für eine bestimmte Installation angegeben, etwa der Installationspfad.*

*- Beim Systemstart: Hier werden Informationen vor oder beim Programmstart dynamisch gelesen.*

*- Während des Programmablaufs: Konfigurationsinformation wird zur Programmlaufzeit erfragt oder gelesen.*

## Parallelisierung und Threading

*Programme können in parallelen Prozessen oder Threads ablaufen - was die Notwendigkeit von Synchronisationspunkten mit sich bringt. Die Grundlagen dieses Aspekten legt die Parallelverarbeitung. Für die Architektur und Implementierung nebenläufiger Systeme sind viele technische Detailaspekte zu berücksichtigen (Adressräume, Arten von Synchronisationsmechanismen (Guards, Wächter, Semaphore), Prozesse und Threads, Parallelität im Betriebssystem, Parallelität in virtuellen Maschinen und andere).*

## Internationalisierung

*Unterstützung für den Einsatz von Systemen in unterschiedlichen Ländern, Anpassung der Systeme an länderspezifische Merkmale. Bei der Internationalisierung (aufgrund der 18 Buchstaben zwischen I und n des englischen Internationalisation auch i18n genannt) geht es neben der Übersetzung von Aus- oder EIngabetexten auch um verwendete Zeichensätze, Orientierung von Schriften am Bildschirm und andere (äußerliche) Aspekte.*

## Migration

*Für die meisten Systeme gibt es existierende Altsysteme, die durch die neuen Systeme abgelöst werden sollen. Denken Sie als Architekt nicht nur an Ihre neue, schöne Architektur, sondern rechtzeitig auch an alle organisatorischen und technischen Aspekte, die zur Einführung oder Migration der Architektur beachtet werden müssen.*

*Beispiele*

*- Konzept, Vorgehensweise oder Werkzeuge zur Datenübernahme und initialen Befüllung mit Daten*

*- Konzept zur Systemeinführung oder zeitweiliger Parallelbetrieb von Alt- und Neusystem*

*Müssen Sie bestehende Daten migrieren? Wie führen Sie die benötigten syntaktischen oder semantischern Transformationen durch?*

## Testbarkeit

*Unterstützung für einfache (und möglichst automatische) Tests. Diese Eigenschaft bildet die Grundlage für das wichtige Erfolgsmuster "Continous Integration". In Projekten sollte mindestens täglich der gesamte Stand der Entwicklung gebaut und (automatisch) getestet werden - daher spielt Testbarkeit eine wichtige Rolle. Wichtige Stichworte hierzu sind Unit- Tests und Mock-Objekte.*

## Skalierung, Clustering

*Wie gestalten Sie Ihr System „wachstumsfähig“, so daß auch bei steigender Last oder steigenden Benutzerzahlen die Antwortzeiten und/oder Durchsatz erhalten bleiben?*

## Hochverfügbarkeit

*Wie erreichen Sie hohe Verfügbarkeit des Systems? Legen Sie Teile redundant aus? Verteilen Sie das System auf unterschiedliche Rechner oder Rechenzentren? Betreiben Sie Standby-Systeme?*

# Referenzen

[1] CRS…

[2] SRS…

[3] ….

# Anhang