Dipl. Ing. Matthias Kalkgruber BSc

Entwicklung einer Software für ein Lokomotionsgerät

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

Masterstudium Biomedical Engineering

eingereicht an der

Technischen Universität Graz

Betreuer

Assoc. Prof. Dr.techn. Jörg Schröttner

Insitut für Health Care Engineering

Graz, April 2016

EIDESSTATTLICHE ERKLÄRUNG

AFFIDAVIT

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit/Diplomarbeit/Dissertation identisch.

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis/diploma thesis/doctoral dissertation.

Datum / Date	Unterschrift / Signature

Danksagung

Lieber Papa, liebe Mama, liebe Michaela und lieber Joachim, ihr habt mich während meines Studiums stets mit viel Geduld unterstützt und dafür möchte mich bei euch bedanken.

Großer Dank gilt auch den Mitarbeitern des Institutes für Health Care Engineering der TU Graz für die interessante Zeit als Studienassistent. Speziell möchte ich mich hierbei bei Andi und Alex bedanken, für unzählige fachliche Anregungen während der Betreuung meiner Arbeit.

Seitens der Klink Judendorf-Straßengel, möchte ich mich bei Sigrid Ranner und Prim. Peter Grieshofer für die gute Zusammenarbeit, sowie für die Möglichkeit zur Durchführung dieser Masterarbeit, bedanken.

Ich möchte mich auch bei Joachim, Thomas, Gerald, Alex und Andi bedanken, die mir sehr hilfreiche Tipps und Verbesserungsvorschläge im Bezug zu meiner Arbeit gegeben haben. Zahlreiche unverständlichen Sätze, Satzstellungen und Rechtschreibfehler flogen dank ihrer Hilfe hinaus. ☺

Abschließend möchte ich mich bei meinen Freunden bedanken, die mich während meines Studiums motiviert und begleitet haben. Dank euch wurde die Studienzeit mit wundervollen Erinnerungen und Erlebnissen gesät.

Dank u Emma voor uw geduld en steun. Je hebt mijn tijd aanzienlijk verfraaid tijdens mijn harde werk.

Entwicklung einer Software für ein Lokomotionsgerät

Kinder mit infantiler Zerebralparese leiden an körperlichen und geistigen Einschränkungen. Um Folgeprobleme zu minimieren, sowie eine lebenslange Pflege zu vermeiden, benötigen Betroffene während der körperlichen und geistigen Entwicklung spezielle Therapien. Als vielversprechende Therapieform gilt die automatisierte Lokomotionstherapie, diese ist jedoch mit bestehenden Geräten erst ab einem Alter von zirka vier Jahren durchführbar. Da eine möglichst frühe Anwendung ein verbessertes Therapieergebnis erwarten lässt, wird an der TU Graz in Kooperation mit der Klinik Judendorf-Straßengel ein Lokomotionsgerät für Kleinkinder entwickelt.

Diese Arbeit umfasst die Entwicklung einer Software für einen Prototypen des Lokomotionsgerätes für Kleinkinder. Diese beinhaltet neben der Ansteuerung der Antriebe zur Generierung physiologischer Bewegungsverläufe, auch eine Patienten- und Anwenderverwaltung. Als Zielvorgabe wurde eine Entwicklung und Dokumentation der Software auf Basis der Norm EN 62304 angestrebt. Anhand einer ausführlichen Risikoanalyse wurden mögliche Risiken erkannt und entsprechende Abhilfemaßnahmen erarbeitet. Gemäß eines V-Diagramms wurden alle Software-Komponenten, sowie das Software-System erfolgreich verifiziert und die Anforderungen an die Funktionalität und Sicherheit können somit als erfüllt betrachtet werden.

Schlüsselwörter: Lokomotionstherapie, ICP, Risikoanalyse, EN 62304, Softwareentwicklung

Software development of an automatic locomotor therapy device

Children with infantile cerebral palsy suffer from physical and mental disabilities. In order to reduce problems at a later stage and to avoid lifelong care, special types of therapies during physical and mental development are required. The locomotion therapy represents a promising approach. However, existing locomotion devices do not support treatment of infants under the age of four years. Furthermore it is assumed that treatment at the earliest possible age can drastically improve therapeutic outcome. For that reason an automatic locomotor therapy device for infants is being developed at the University of Technology Graz in cooperation with the clinic Judendorf-Straßengel.

This work comprises the software development for a prototype of the locomotor therapy device for infants. The software requirement includes the control of driven ortheses, in order to generate physiological movement-patterns and a user and patient management system. It was the aim to carry out the development in compliance with the standard EN 62304. In an extensive risk analysis possible risks have been identified and appropriate safety measures have been deducted. According to a V-Model all software-components, as well as the complete software-system, were successfully verified. Thus, the requirements to the functionality and safety are met.

Keywords: locomotion therapy, ICP, risk analysis, EN 62304, software development

Inhaltsverzeichnis

Allgemeine Orientierungshinweise

Aus Gründen der besseren Lesbarkeit dieser Masterarbeit wurde auf eine genderkonforme Schreibweise verzichtet. Die männlichen Formen sind als geschlechtsneutral gemeint und sollen damit Frauen und Männer gleichermaßen berücksichtigen.

Um den Umfang der schriftlichen Arbeit in einem beschränkten Rahmen zu halten, werden nicht alle erstellten Dokumente der schriftlichen Version beigefügt, sondern als digitaler Anhang in Form einer CD zur Verfügung gestellt.

1 Einleitung

In Europa erkranken im Mittel zirka 2,08 von 1000 Neugeborene an infantiler Zerebralparese (ICP) [1]. Die ICP äußert sich in einer Haltungs- und Bewegungsstörung und wird durch Hirnschädigungen ausgelöst, welche in der Schwangerschaft, während der Geburt oder in den ersten Lebensjahren verursacht wurden. Diese Hirnschädigungen treten häufig bei Frühgeburten, durch Sauerstoffmangel, sowie durch Infektionen, Blutungen und Fehlbildungen des Gehirns auf [2].

Eine gebräuchliche Definition der infantilen Zerebralparese beschreibt das Krankheitsbild folgendermaßen:

> *„Der Begriff Zerebralparesen umfasst eine Gruppe von Krankheitsbildern, die häufig sind und zu einer meist schweren Behinderung führen, die bei den Betroffenen ähnliche und spezielle medizinische, sozialmedizinische und therapeutische Unterstützungen notwendig macht."*
> Zitat aus [3].

Patienten mit infantiler Zerebralparese haben Schädigungen am zentralen Nervensystem, welche sich unter anderem durch Verlust der selektiven Muskelkontrolle, unnatürlichem Muskeltonus, sowie einem Ungleichgewicht zwischen Antagonisten und Agonisten äußern. Für ein physiologisches Wachstum der Knochen, Gelenke und Muskeln eines Kleinkindes sind die wirkenden Kräfte und Dehnungen auf diese maßgebend. Bei Kleinkindern mit ICP wirken auf den Bewegungsapparat jedoch verzerrte Kräfte beziehungsweise Dehnungen. Daraus resultiert ein unphysiologisches Wachstum und es entwickeln sich im weiteren Verlauf Abnormalitäten. [4]
Häufig treten Begleitsymptome wie Sprach-, Seh- und Gehörstörungen sowie Lern- und Intelligenzdefizite auf [2].

Behandlungsmöglichkeiten von ICP

Zur Behandlung von ICP ist ein breites Spektrum an Therapieansätzen verfügbar. Diese umfassen Bereiche der Ergotherapie, Logotherapie, Krankengymnastik, wahrnehmungszentrierte Methoden, orthopädische Therapien und auch tiergestützte Therapien [5], [6].

Eine vielversprechende Therapieform stellt die Lokomotionstherapie dar und basiert auf dem Konzept der neuronalen Plastizität. Durch das Feedback der Kräfte während des Gehens, wird das Rückenmark stimuliert. Dies fördert eine Neubildung beziehungsweise ein Umstrukturierung von Nervenzellen. Durch wiederholtes und korrektes Durchführen von extern geführten beziehungsweise unterstützen Gehbewegungen werden sogenannte Lokomotionszentren stimuliert. Diese Lokomotionszentren sollen durch die Therapie soweit entwickelt werden, dass diese in der Lage sind

Gehbewegungen ohne externe Hilfe vornehmen zu können und somit ein eigenständiges Gehen des Patienten zu ermöglichen. [7]

Eine Form dieses Therapiekonzepts stellt das teilentlastete Laufbandtraining dar. Dabei geht der Patient auf einem Laufband und ein Teil seines Gewichts wird durch ein Gurtsystem getragen. Der Physiotherapeut führt dabei die unteren Extremitäten des Patienten, um einen möglichst physiologischen Bewegungsablauf nachzuahmen. Essentielle Elemente für die Stimulation der Lokomotionszentren stellen dabei der initiale Bodenkontakt, sowie die Gewichtsbelastung während der Standphase[1] dar. Ein wesentliches Problem dieser Therapieform besteht im hohen Kraftaufwand und der Anstrengung für den Therapeuten. Aufgrund der körperlichen Voraussetzungen ist nicht jeder Therapeut in der Lage diese Therapie durchzuführen beziehungsweise ist die Dauer und Häufigkeit der Therapien limitiert. [7]

Robotergestützte Lokomotionstherapie

Aus den genannten Gründen wurden robotergestützte Lokomotionsgeräte entwickelt. Diese ermöglichen im Vergleich zur manuellen Lokomotionstherapie einerseits eine deutlich präzisere Vorgabe der Bewegungsmuster und andererseits können die Bewegungsmuster unbegrenzt oft wiederholt werden, ohne den Therapeuten körperlich zu belasten. Ebenso kann über Kraftmessungen an den Orthesen ein Feedback über die Eigenleistung des Patienten gegeben werden. [7]

Die folgenden zwei Lokomotionsgeräte sind kommerziell verfügbar: Gait Trainer 1® (Reha-Stim, Deutschland) und Lokomat® (Hocoma AG, Schweiz). Beide Geräte sind vorwiegend für die Anwendung an Erwachsenen konzipiert und nur bedingt für Kinder einsetzbar. Der Lokomat® ermöglicht eine Therapie ab einer Körpergröße von 90 cm beziehungsweise ab einem Alter von zirka 4 Jahren. [8] Der Gait Trainer 1® kann ebenfalls erst ab einem Alter von 5 Jahren eingesetzt werden [9].

Entwicklung eines Lokomotionsgerätes für Kleinkinder

Wenn man nun die Entwicklungskurven der motorischen Fähigkeiten von Kindern mit ICP betrachtet, zeigt sich, dass 90 % der individuellen motorischen Fähigkeit in einem Alter von 2,7 bis 4,8 Jahren erreicht werden. Die Verläufe der motorischen Entwicklungskurven, gemessen anhand des „Gross Motor Function Measures 66“ (GMFM-66)[2], sind in Abbildung 1-1 dargestellt. Bei stärkerer

[1] Für eine detaillierte Beschreibung des Gangzyklus wird auf folgende Literatur verwiesen: [32].

[2] GMFM-66 ist ein Test zur Beurteilung der motorischen Fähigkeiten von Kindern mit infantiler. Eine ausführliche Beschreibung zu GMFM ist in [10] zu finden.

Ausprägung der Erkrankung (z.B. Level IV) verläuft die Entwicklungskurve flacher und erreicht früher ein bedeutend niedrigeres Maximum. [9]

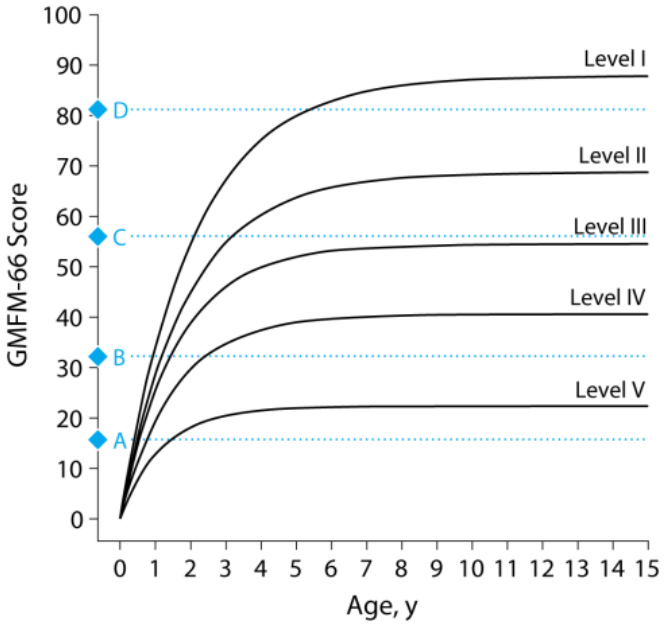


Abbildung 1-1: Entwicklungskurven der motorischen Fähigkeiten bei unterschiedlicher Ausprägung von ICP [11].

Daraus lässt sich folgende Schlussfolgerung ableiten:

Therapien in einer möglichst frühen Phase können einen wesentlichen Beitrag zur besseren Entwicklung des Bewegungsapparates leisten und folglich zu einer Verbesserung der motorischen Fähigkeiten führen. Somit können Folgeprobleme verringert und die Lebensqualität langfristig erhöht werden.

Basierend auf dieser Überlegung entstand die Motivation dieser Arbeit, mit dem Ziel, ein Lokomotionsgerät für Kleinkinder zu entwickeln, um auch Kindern in einem Alter zwischen 1 und 4 Jahren eine robotergestützte Lokomotionstherapie zu ermöglichen.

Die Entwicklung des Lokomotionsgerätes erfolgt in Kooperation mit der Forschungsabteilung der Klinik Judendorf-Straßengel „Auch ich will gehen“.

Ausgangspunkt dieser Arbeit

Im Vorfeld dieser Arbeit wurde bereits das Konzept für die technische Umsetzung eines Lokomotionsgerätes für Kleinkinder im Rahmen der Masterarbeit „Entwicklung eines Lokomotionsgerätes für Kleinkinder“ von A. Tilp [8] erarbeitet.

Das Konzept verwendet robotergesteuerte Orthesen, welche am Oberschenkel beziehungsweise Unterschenkel montiert sind und einen physiologischen Gang vorgeben. Ein passives Laufband sorgt für die Fortbewegung des Patienten während der Standphase, um eine möglichst optimale Stimulation der Lokomotionszentren während der Gewichtsbelastung, sowie durch den Initialkontakt beim Auftreten zu erzielen.

Ziel dieser Arbeit ist die Entwicklung einer Software zur Ansteuerung der robotergesteuerten Orthesen. Zusätzlich zur Erzeugung der Bewegungsmuster soll in dieser Software eine Patienten- und Anwenderverwaltung umgesetzt werden.

Das zu entwickelnde Lokomotionsgerät dient zur Behandlung von Krankheiten und ist somit per Definition nach Richtlinie 93/42/EWG ein aktives therapeutisches Medizinprodukt [12]. Die zu entwickelnde Software fällt ebenso unter die Richtlinie 93/42/EWG und ist als Medizinprodukt zu sehen, da diese ein aktives therapeutisches Medizinprodukt steuert [12]. Für die Herstellung und den Betrieb von Medizinprodukten müssen gesetzliche Rahmenbedingungen eingehalten werden. Eine dieser Rahmenbedingungen für Medizinprodukte-Software ist die Norm *EN 62304 („Medizingeräte-Software Software-Lebenszyklus-Prozesse“),* daher wird die Software nach EN 62304 entwickelt.

2 Aufgabenstellung

Ziel dieser Arbeit ist die Entwicklung einer Software zur Steuerung eines Lokomotionsgerätes für Kleinkinder.

Folgende Anforderung an die Software sollen umgesetzt werden:

- Vorgabe einer zyklischen Bewegung, um ein physiologisches Gangbild zu erzeugen.
- Umsetzung einer variablen Geschwindigkeit von bis zu 120 Schritte pro Minute.
- Graphische Darstellung von spezifischen Therapiegrößen für jedes Gelenk während der Therapie:
    - Feedback in welchem Maß der Patient die Bewegung selbst erzeugt
    - Aktuelles Drehmoment an den Orthesen
- Darstellung und Speicherung von Therapieparametern:
    - Maximale Kadenz
    - Anzahl durchgeführter Schritte
    - Dauer der Therapie
    - Mittlerer Kraftaufwand pro Gangzyklus
    - Therapeut, Datum, Uhrzeit, …
- Bedienung der Software über ein graphisches Benutzerinterface
- Anwenderverwaltung
- Patientenverwaltung
- Ableiten und Umsetzen von Sicherheitsmechanismen um eine sichere Therapie zu ermöglichen.
- Fehlerverarbeitung (Dokumentation, Behandlung und Darstellung)

Ein wesentlicher Punkt der Arbeit ist die Entwicklung und dessen Dokumentation unter Berücksichtigung der Vorgaben aus Normen und Richtlinien. Dies betrifft vor allem die Entwicklung der Software nach EN 62304, wobei im Rahmen dieser Arbeit nicht erwartet wird die EN 62304 vollständig umzusetzen. Vielmehr sollen die wesentlichen Anforderungen umgesetzt werden, um die nötige Basis für eine Weiterentwicklung der Software zur Verfügung zu stellen.

Abschließend soll noch ein Entwurf für eine Bedienungsanleitung der entwickelten Software erstellt werden.

3 Methoden

3.1 Mechanischer Aufbau

Das Konzept des Lokomotionsgerätes wurde im Zuge der Masterarbeit von A. Tilp [8] entwickelt und wird im Folgenden kurz erklärt. Der Aufbau mit Benennung der wichtigsten mechanischen Komponenten mit Ausnahme des Antriebsstranges ist in Abbildung 3-1 dargestellt. Der Antriebsstrang wird in einem Unterabschnitt in diesem Kapitel separat erklärt.

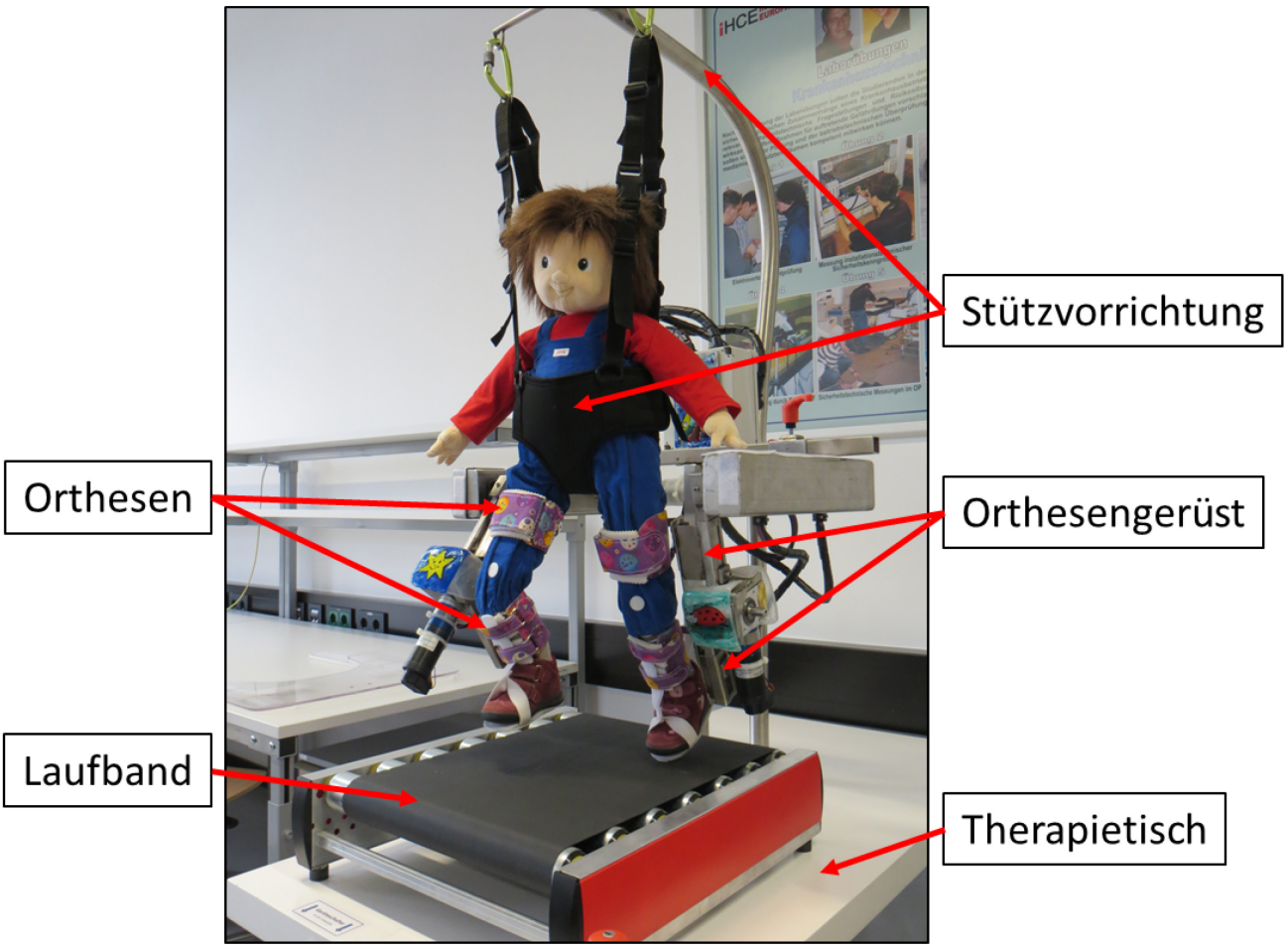


Abbildung 3-1: Gesamtübersicht des mechanischen Aufbaus des Lokomotionsgerätes.

Mit Hilfe der Stützvorrichtung, bestehend aus dem Gurtsystem und der Aufhängung, wird der Patient im Lokomotionsgerät fixiert und sorgt für eine aufrechte Haltung des Patienten. Die Ankopplung der unteren Extremitäten erfolgt durch Orthesen am Oberschenkel beziehungsweise am Unterschenkel. Diese sind wiederum am Orthesengerüst montiert. Mittels Vorgabe der Winkelverläufe an den Orthesengerüsten werden physiologische Gehbewegungen vorgegeben. Dabei unterstützt das Laufband die Fortbewegung des Patienten während der Bodenkontaktphase.

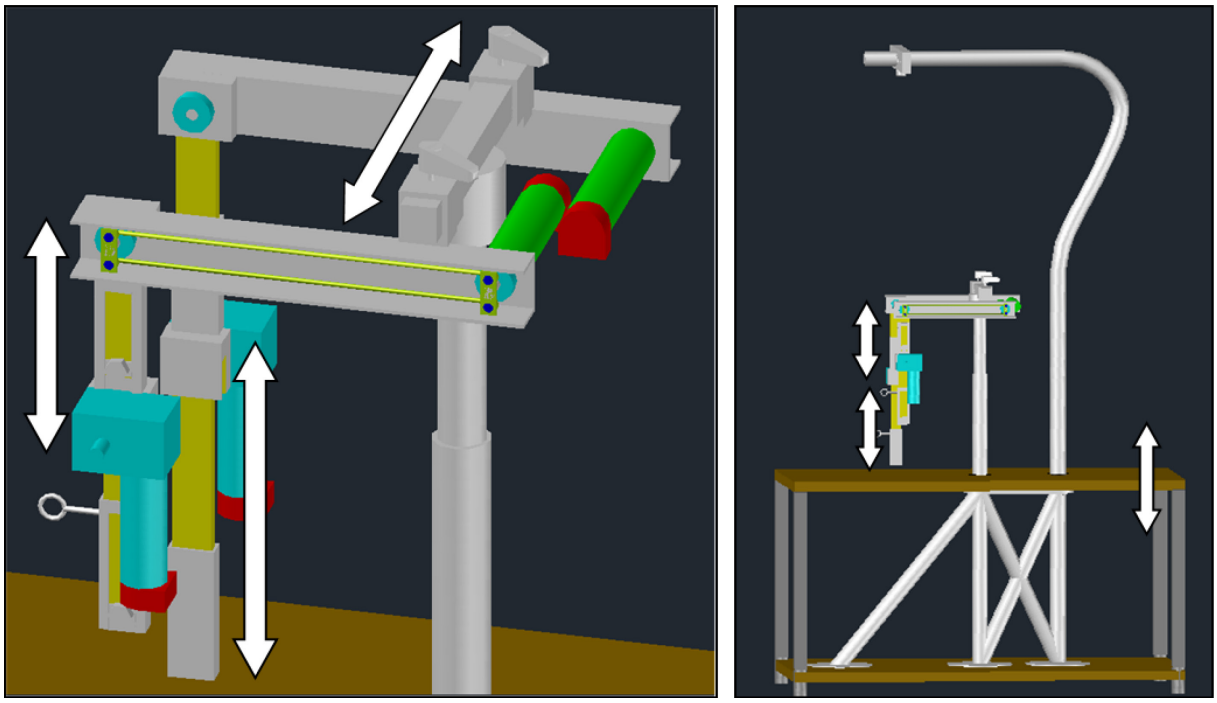


Abbildung 3-2: Funktionsschema für die optimale Anpassung des Lokomotionsgerätes an den Patienten.

Für die optimale Anpassung des Lokomotionsgerätes an die Körperproportionen des Patienten können die Längen der Orthesengerüste, die Hüftbreite zwischen den Orthesengerüsten, sowie die Höhe des Therapietisches variabel angepasst werden. Das Funktionsschema ist in Abbildung 3-2 dargestellt.

Antriebsstrang

Für die Bewegungserstellung werden vier permanent erregte Gleichstrommotoren „DC-Kleinstmotor 3863 024CR“ (Dr. Fritz Faulhaber GmbH, Deutschland) [13] verwendet. Eine Zusammenfassung der wichtigsten Kenndaten ist im Anhang in Tabelle 18 zu finden.

Um die Motormomente und Motordrehzahlen in den gewünschten Arbeitsbereich zu transformieren, werden diese mit einem Getriebe untersetzt. Beim Antrieb für das Hüftgelenk kommt ein Planetengetriebe [14] mit einem Übersetzungsverhältnis $i_{\text{Hüfte}} = 120$ zum Einsatz. Bei den Kniegelenken wird eine Kombination aus Planetengetriebe [15] und Schneckengetriebe [16] mit einem kombinierten Übersetzungsverhältnis $i_{\text{Knie}} = 98$ verwendet. Der Vollständigkeit halber und zum besseren Verständnis der Anforderungen an die Antriebe werden deren Kenndaten, sowie die benötigten Drehmomente und Drehzahlen für die Bewegungserstellung in Tabelle 1 angeführt.

Tabelle 1: Kenndaten der Gleichstrommotoren DC-Kleinstmotor 3863 024CR ; benötigte Drehmomente und Drehzahlen für die Bewegungserstellung bei 120 Schritte/min [8], [14]–[16].

Kenndaten	Einheit		Hüftgelenk	Kniegelenk
Wirkungsgrad der Getriebe	%		90	64
Übersetzungsverhältnis	a. u.		120	98
Dauerdrehmoment	mNm	Motor	157	157
Benötigtes maximales Drehmoment	mNm		110	101
Leerlaufdrehzahl	U/min		5800	5800
Benötigte maximale Drehzahl	U/min		4539	5672
Dauerdrehmoment	Nm	Orthese	16,96	9,85
Benötigtes maximales Drehmoment	Nm		11,92	6,33
Leerlaufdrehzahl	U/min		48,33	59,18
Benötigte maximale Drehzahl	U/min		37,83	57,87

Die Kraftübertragung vom Motor zum Orthesengerüst des Oberschenkels wird durch ein Parallelogramm realisiert. Für die Bewegungserzeugung des Unterschenkels wird der Motor am Oberschenkel-Orthesengerüst montiert und das Orthesengerüst des Unterschenkels direkt mit der Motorwelle verbunden. Der Antriebsstrang zur Kraftübertragung an die Orthesengerüste ist in Abbildung 3-3 dargestellt.

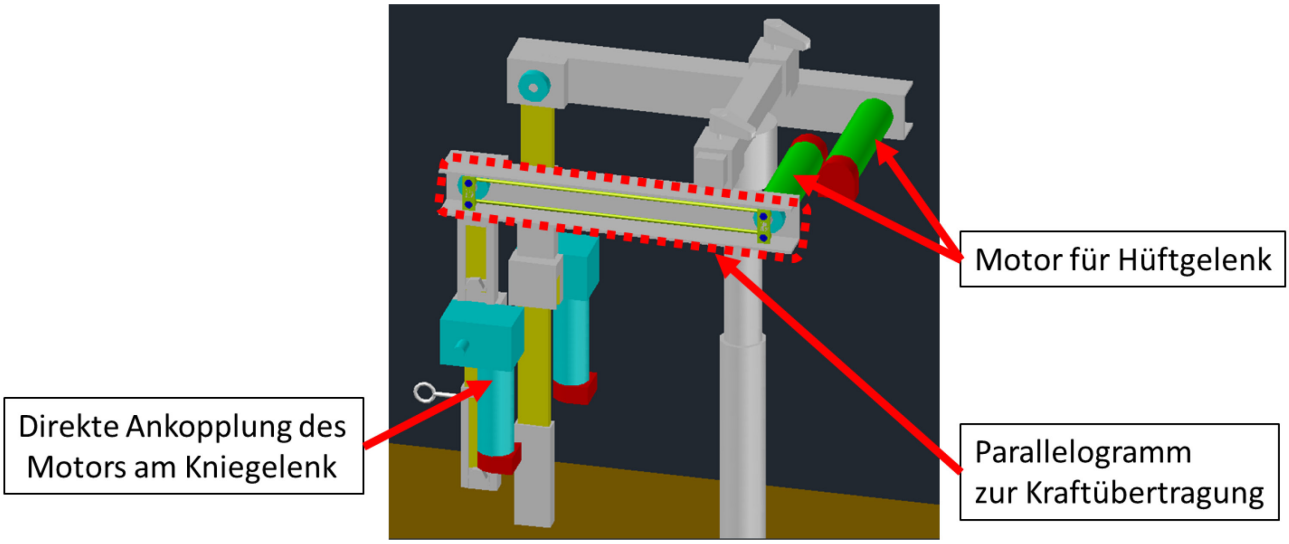


Abbildung 3-3: Antriebsstrang zur Kraftübertragung an die Orthesengerüste.

3.2 Steuerungskonzept

Eine schematische Darstellung des Steuerungskonzepts ist in Abbildung 3-4 dargestellt und wird im Folgenden genauer erläutert:

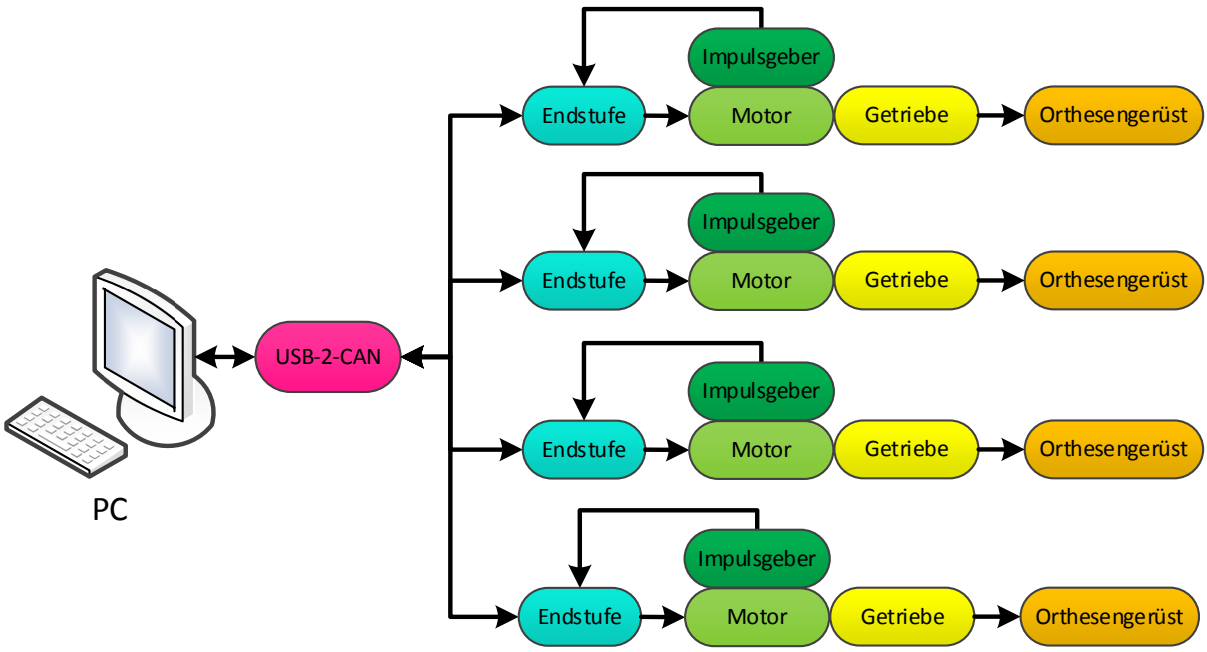


Abbildung 3-4: Steuerungskonzept des Lokomotionsgerätes.

Endstufen

Für die Regelung der Antriebe werden vier Endstufen *„MCDC 3006 S CF“* (Dr. Fritz Faulhaber GmbH, Deutschland) verwendet. Eine detailliertere Beschreibung dieser Endstufen erfolgt in Kapitel 3.2.1.

Sensorik

Als Sensoreinheit wird der optische Impulsgeber „HEDS 5500 A-12“ (Dr. Fritz Faulhaber GmbH, Deutschland) [17] zur Positionserfassung verwendet. Dieser ist direkt an die Motorwelle gekoppelt und gibt über zwei Kanäle je 500 Impulse pro Umdrehung aus. Die Auswertung der relativen Änderung erfolgt direkt durch die Endstufen mit einer 4-Flanken Auswertung. Dadurch ergibt sich eine relative Winkelbestimmung mit einer Winkelauflösung von 2000 Ink./U.. [17], [18]
Der Vollständigkeit halber sind die wichtigsten Kennwerte im Anhang in Tabelle 19 aufgelistet.

Übergeordnete Steuerung

Die drei Komponenten (Endstufe, Gleichstrommotor und Impulsgeber) bilden somit den Regelkreis zur Bewegungsgenerierung. Die einzelnen Endstufen werden dabei von einer übergeordneten Steuerungssoftware verwaltet.

Die Steuerungssoftware stellt die eigentliche Entwicklungsaufgabe dieser Arbeit dar. Die Software wird in der Entwicklungsumgebung LabVIEW (National Instruments AG, USA) implementiert und auf einem handelsüblichen Computer ausgeführt. Mit Hilfe einer grafischen Benutzerschnittstelle (GUI) werden die Benutzerinteraktionen durch die Software interpretiert und die entsprechenden Steuerbefehle an die Endstufen gesendet. Zusätzlich werden antriebsspezifische Kenngrößen von den Endstufen an die übergeordnete Steuerung übermittelt und dort zur Sicherheitsüberwachung ausgewertet. Die wichtigsten Aufgaben bezüglich der Ansteuerung der Antriebe:

- Initialisierung der Endstufen
- Konfiguration der Endstufen
- Referenzierung der Inkrementalgeber
- Vorgabe von Bewegungsparametern (Position, Geschwindigkeit und Beschleunigung)
- Überwachung der Antriebszustände (Momente, Position, Ströme, Temperatur, usw.)

Kommunikation

Für die Kommunikation zwischen den Endstufen und der übergeordneten Steuerung wurde ein High-Speed-Controller Area Network (CAN) gewählt. Im Vergleich zur Alternative (Serielle-Schnittstelle RS-232) wird eine deutlich höhere Datenrate und Störfestigkeit erreicht. Die CAN-Schnittstelle erreicht eine Datenübertragungsrate von bis zu 1 Mbit/s bei einer Bus-Länge von 40 m [19].
Zusätzlich ist der CAN als Bus-Topologie (siehe Abbildung 3-5) ausgeführt und somit lassen sich weitere Teilnehmer im Kommunikationssystem mit einem geringen Aufwand hinzufügen. Diese müssen lediglich mit einer Stichleitung verbunden werden. [19]

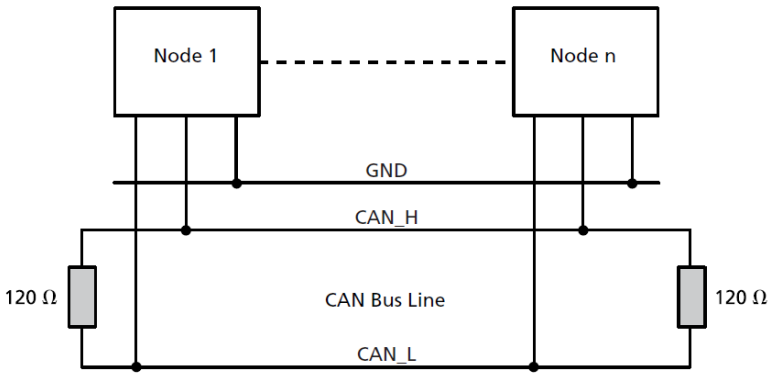


Abbildung 3-5: Bus-Topologie eines CAN-BUS. Entnommen aus [20].

Nach [21] bietet eine CAN-Schnittstelle folgende Vorteile:

- Einfacher Busaufbau
- Multi Master System
- Nachrichtenorientiertes Protokoll
- Priorisierung von Nachrichten
- Hohe Störfestigkeit des Protokolls
- Optimiert für sehr kleine Datenmengen
- Kurze Latenzzeit für hoch priorisierte Nachrichten

Die verwendeten Endstufen kommunizieren mit dem CANOpen-Protokoll nach dem Geräteprofil CiA 402 für elektrische Antriebe [20]. Die Kommunikation mit dem beschriebenen Protokoll stellt eine wesentliche Komponente dieser Arbeit dar. Auf eine ausführliche Beschreibung wird jedoch in diesem Teil verzichtet und der interessierte Leser auf Kapitel F im Anhang verwiesen. Zusätzlich ist eine ausführliche Beschreibung in [19]–[24] zu finden.

USB-to-CAN Schnittstelle

Als Schnittstelle zwischen PC und CAN wird die „USB-To-CAN compact“ Schnittstelle [25] (IXXAT Automation GmbH, Deutschland) verwendet (siehe Abbildung 3-6). Im Vergleich zu fix verbauten Schnittstellen erlaubt diese eine einfache und flexible Anbindung eines Computers zum CAN-Bus, da lediglich eine USB-Schnittstelle benötigt wird.

Abbildung 3-6: USB-to-CAN Schnittstelle der Firma IXXAT Automation. Adaptiert von [25].

3.2.1 Endstufen

Als Endstufen werden vier Motion-Controller *MCDC 3006 S CF* (Dr. Fritz Faulhaber GmbH, Deutschland) verwendet (siehe Abbildung 3-7). Unter Motion-Controller versteht man eine Endstufe, welche im Stande ist eine Positionier- und eine Drehzahlregelung durchführen zu können. Durch Vorgabe der Bewegungsparamter wie Position, Geschwindigkeit, Beschleunigung und Bremsbeschleunigung können komplexe Bewegungsprofile erstellt werden und ermöglichen somit die Vorgabe der Bewegungsabläufe für die Lokomotionstherapie. Zur Vollständigkeit und besserem Verständnis der Zusammenhänge sind die wichtigsten Parameter der Motion-Controller in Tabelle 2 angeführt.

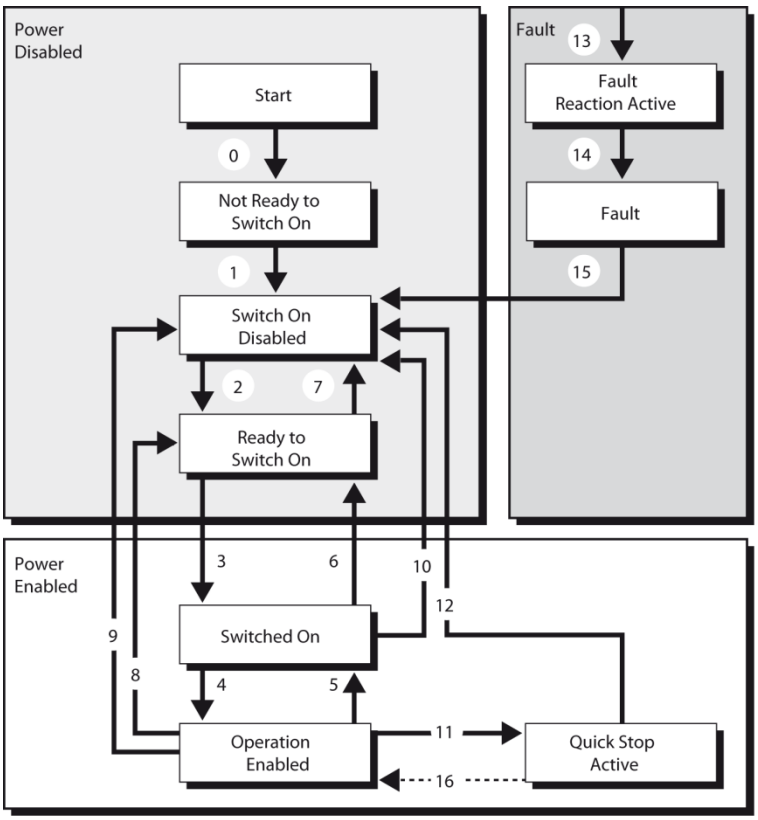
Abbildung 3-7: Motion-Controller MCDC 3006 S CF. Entnommen aus [26].

Tabelle 2: Kenndaten der Endstufe MCDC 3006 S CF. Entnommen aus [18].

Bezeichnung	Motion-Controller MCDC 3006 S CF
Betriebsbereich	4-Quadranten PWM
PWM-Schaltfrequenz in kHz	78,12
Versorgungsspannung in V DC	12 … 30
Wirkungsgrad in %	95
Max. Dauer-Ausgangsstrom in A	6
Max. Spitzen-Ausgangsstrom in A	10
Drehzahlbereich in U/min	5 … 30 000
Regler Abtastrate in µs	100
Auflösung mit externem Encoder in Ink./U	≤ 65 535
Ein- / Ausgänge	5
Schnittstelle	CAN
Protokoll	CANOpen
Max. Übertragungsgeschwindigkeit in Mbit/s	1

3.2.1.1 Zustandsmaschine

Das Verhalten der Endstufen wird durch eine Zustandsmaschine beschrieben. Diese wird durch die übergeordnete Steuerung kontrolliert, indem sogenannte Prozessdatenobjekte (PDOs) gesendet werden [20]. Der Aufbau dieser Zustandsmaschine mit den zugehörigen Kommandos ist in Abbildung 3-8 dargestellt.



Kommando	Übergänge
Shutdown	2, 6, 8
Switch on	3
Disable voltage	7, 9, 10, 12
Quick stop	7, 10, 11
Disable operation	5
Enable operation	4, 16
Fault reset	15
Autonomer Übergang	0, 1, 13, 14

Abbildung 3-8: Zustandsmaschine der FAULHABER-Antriebe. Adaptiert von [20].

Im Zustandsbereich ***Fault*** oder ***Power Disabled*** sind die Antriebe spannungsfrei und die Regler sind abgeschaltet. Durch Übermittlung der entsprechenden Kommandos werden die Antriebe in den Zustand ***Switched On*** gesetzt. In diesem Zustand wird die Spannungsversorgung freigegeben und die Regler aktiviert. Bei weiterer Übermittlung des Kommandos *Enable Operation,* wechselt die Zustandsmaschine in den Zustand ***Operation Enabled.*** Positionierbefehle können erst in diesem Zustand durchgeführt werden. [20]

Der Aufbau dieser Zustandsmaschine stellt ein wesentliches Sicherheitsmerkmal dar. Bei Erkennung eines Fehlers (Spannungsausfall, Drehmomentenüberschreitung, usw.) gehen die Antriebe sofort in einen sicheren spannungslosen Zustand über und eine entsprechende Fehlermeldung wird an die übergeordnete Steuerung versandt. Des Weiteren können keine irrtümlichen Positionierbefehle nach Spannungswiederkehr oder beim Einschalten erfolgen, da sich die Motion-Controller im Zustand ***Switch On Disabled*** befinden. [20]

3.2.1.2 Betriebsarten

Zur Regelung der Antriebe stehen zwei Modi zur Verfügung. Einerseits können die Antriebe über den *Profile Velocity Mode* und andererseits über den *Profile Position Mode* geregelt werden. Im *Profile Velocity Mode* werden zeitlich die gewünschten Zieldrehzahlen vorgegeben und es ergibt sich der Verlauf der Position.

Im Gegensatz dazu erfolgt beim *Profile Position Mode* die Vorgabe der gewünschten Zielposition und das zugehörige Drehzahlprofil wird durch die Endstufe berechnet.
Die Berechnung der entsprechenden Solldrehzahlen für den Drehzahlregler erfolgt dabei durch den Rampengenerator. Bei Vorgabe der Beschleunigung, der Bremsbeschleunigung, sowie der maximalen Drehzahl erstellt dieser ein stückweise lineares Drehzahlprofil und ergibt somit einen stückweisen quadratischen Positionsverlauf. [20]

Der Verlauf der Bewegungsprofile für die beiden Modi ist in Abbildung 3-9 dargestellt. Durch kontinuierliche Vorgabe entsprechender Bewegungsparameter können komplexe Bewegungen erstellt werden (siehe Kapitel 3.3). [20]
Bei Verwendung des *Profile Position Modus* ergeben sich für die vorliegende Anwendung folgende Vorteile:

- Einfache Berechnung der Positionstrajektorien
- Leichte Synchronisation der 4 Bewegungsachsen
- Begrenzen des Wegdriftens der Positionen aufgrund von Sende-Zyklus-Verzögerungen
- Beschränkter Winkelbereich im Vergleich zum *Profile Velocity Mode*

Aufgrund dieser Vorteile wird für die Bewegungserstellung der *Profile Position Mode* verwendet.

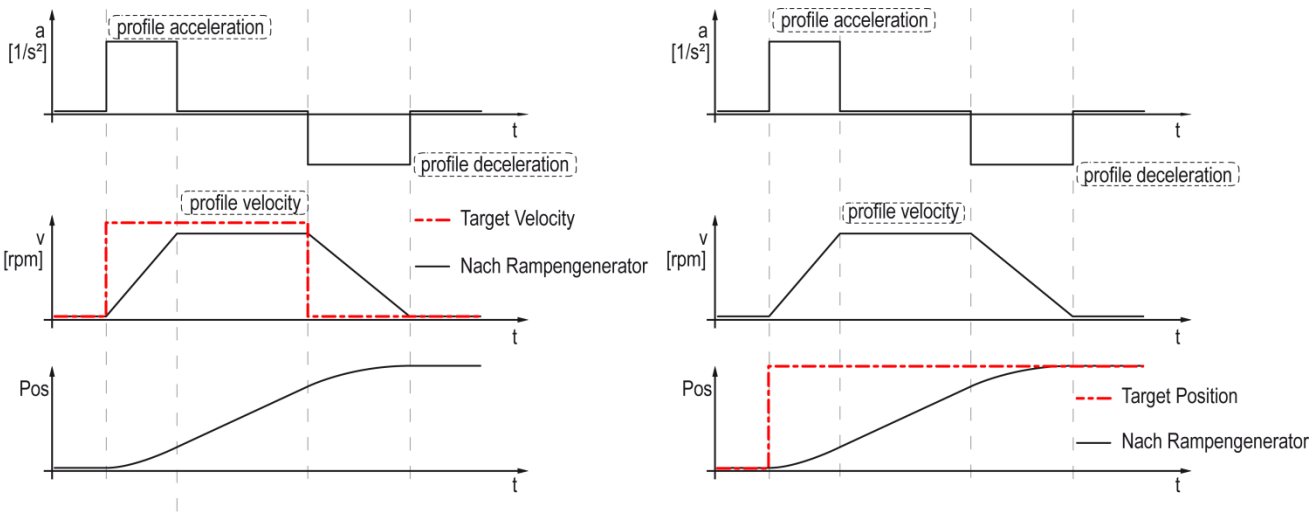


Abbildung 3-9: Bewegungsverlauf im Profile Velocity Mode links und Profile Position Mode rechts. Entnommen aus [20].

3.2.1.3 Regler-Struktur

Die Regelung im *Profile Position Mode* wird durch eine kaskadierte Reglerstruktur realisiert (siehe Abbildung 3-10). Dabei wir die Reglung durch mehrere in sich verschachtelte Regelkreise realisiert. Der Vorteil ist eine Erhöhung der Reglerperformance im Vergleich zu einem direkt wirkenden Regler [27].

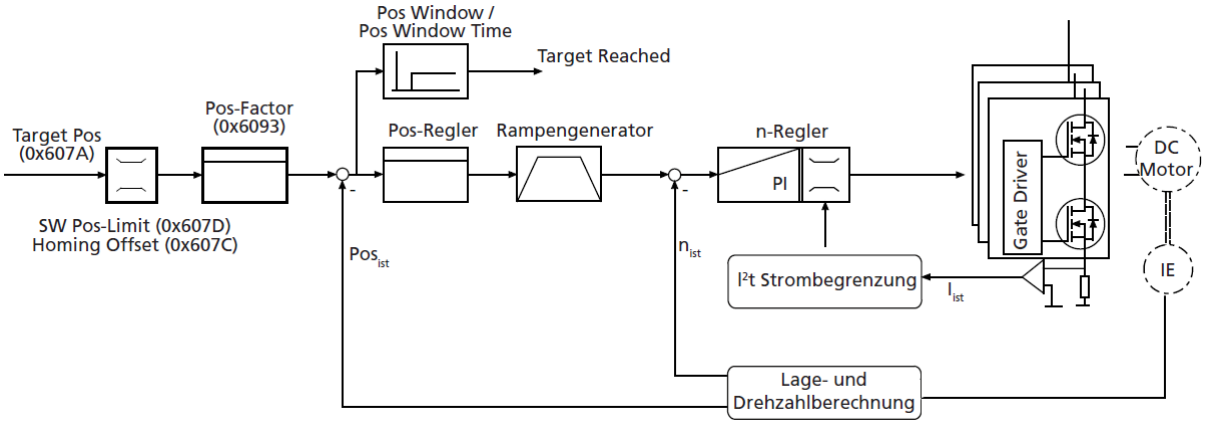


Abbildung 3-10: Reglerstruktur im Profile Position Mode. Entnommen aus [20].

Die Lageregelung stellt die äußerste Regelschleife dar. Diese wird durch einen Proportional-Differential–Regler (PD-Regler) realisiert, welcher je nach Lagefehler eine Soll-Drehzahl für die untergeordnete Regelschleife vorgibt. Zu beachten ist hierbei, dass der differentielle Anteil der Regelung erst beim Eintritt in den Zielkorridor aktiviert wird. Dadurch wird ein schnelleres Erreichen der Zielposition ermöglicht. [20]

Dem Lageregler ist ein Rampengenerator nachgestellt, um die entsprechende Solldrehzahl in Abhängigkeit der eingestellten Bewegungsparameter (maximale Beschleunigung, maximale Bremsbeschleunigung und maximale Geschwindigkeit) zu begrenzen. Somit ist die Solldrehzahl am Ausgang des Rampengenerators die Führungsgröße für den unterlagerten Drehzahlregler. Dieser ist als Proportional-Integral-Regler (PI-Regler) ausgeführt. [20]

Um die Antriebe vor thermischer Zerstörung zu schützen oder eine Momentenbegrenzung durchzuführen, wird der Ausgang des Drehzahlreglers durch einen Strombegrenzungsregler überwacht und begrenzt. [20]

3.2.1.4 Regler-Auslegung

Für die Auslegung von Standardreglern wie PI und PD gibt es grundsätzlich drei verschiedene Dimensionierungsansätze, welche im Folgenden kurz erläutert werden.

Modellbasierter Reglerentwurf

Die Grundlage des modellbasierten Reglerentwurfs basiert auf einem mathematischen Modell der Regelstrecke, welches die reale Strecke näherungsweise nachbildet [28]. Der Vorteil dieser Methode liegt darin, dass auf Basis des Streckenmodells eine optimale Auslegung der Reglerparameter erfolgen kann. Jedoch ist eine adäquate Modellbildung sowie Parameterbestimmung sehr aufwendig.

Heuristischer Reglerentwurf

Im Gegensatz zu den modellbasierten Methoden wird für die Dimensionierung mittels heuristischer Methoden kein mathematisches Modell der Strecke benötigt. Stattdessen werden Experimente an der Anlage vorgenommen, wodurch sich die nötigen Reglerparameter ableiten lassen. Die bekanntesten heuristischen Verfahren sind die Einstellregeln nach Ziegler-Nichols [29], Chien et al. [30] oder die T-Summen-Regel [31]. Diese Verfahren basieren auf der Analyse der Sprungantwort des offenen Regelkreises oder auf dem Verhalten des Systems an der Stabilitätsgrenze. Eine gute Zusammenfassung dieser Verfahren kann in [32] gefunden werden.
Aufgrund der fixen Reglerstruktur der Motion-Controller kann keine Auftrennung der Rückführung erfolgen. Aus diesem Grund kann keine Sprungantwort des offenen Regelkreises durchgeführt werden. Bei Analysen an der Stabilitätsgrenze wird eine konstante Drehzahl im Zieldrehzahlbereich vorgegeben und die Proportionalverstärkung solange erhöht, bis das System instabil wird und zu schwingen beginnt. Aufgrund des mechanischen Aufbaus ist der zur Verfügung stehende Winkelbereich jedoch kleiner als eine Umdrehung und somit können keine konstanten Drehzahlen für eine längere Zeitdauer vorgegeben werden. Dadurch können ebenfalls keine Versuche an der Stabilitätsgrenze durchgeführt werden.

Empirische Dimensionierung

Die letzte Methode besteht in der empirischen Dimensionierung der Regler. Die Voraussetzung für dieses Verfahren ist, dass die Reglerparameter auf Basis von praktischen Erfahrungswerten bereits näherungsweise richtig ausgelegt sind. Durch Analyse der Sprungantwort des Systems werden dann die Reglerparameter sukzessive variiert, bis das gewünschte Verhalten erreicht wird. Für eine ausführliche Erklärung der empirischen Dimensionierung wird auf [33] verwiesen.

Einerseits übersteigt der Aufwand der modellbasierten Dimensionierung bei weitem den Aufwand der empirischen Dimensionierung und andererseits werden durch den Antriebskonfigurator der Endstufen*, Motion Manager* (Dr. Fritz Faulhaber GmbH, Deutschland), bereits Reglerparameter für einen sicheren Betrieb zu Verfügung gestellt [20]. Aus diesem Grund wurde für die Auslegung der Reglerparameter die empirische Reglerdimensionierung gewählt.

Ein Überschwingen von 5-10 % wurde als Vorgabe der Regleroptimierung angestrebt um einen guten Kompromiss zwischen Führungsverhalten und Störverhalten zu erreichen.
In Tabelle 3 sind die ermittelten Reglerparameter für die Motoren des Hüft- und Kniegelenks aufgelistet. Zusätzlich ist die Sprungantwort der Regelung für die beiden Gelenke bei einem Führungsgrößensprung von 10 000 Inkrementen (≙ 500 Grad motorseitig) in Abbildung 3-11 dargestellt.

Tabelle 3: Empirisch ermittelte Reglerparameter.

Gelenk	Lageregler		Drehzahlregler	
	P-Anteil	D-Anteil	P-Anteil	I-Anteil
Kniegelenk	50	2	2	15
Hüftgelenk	70	2	4	14

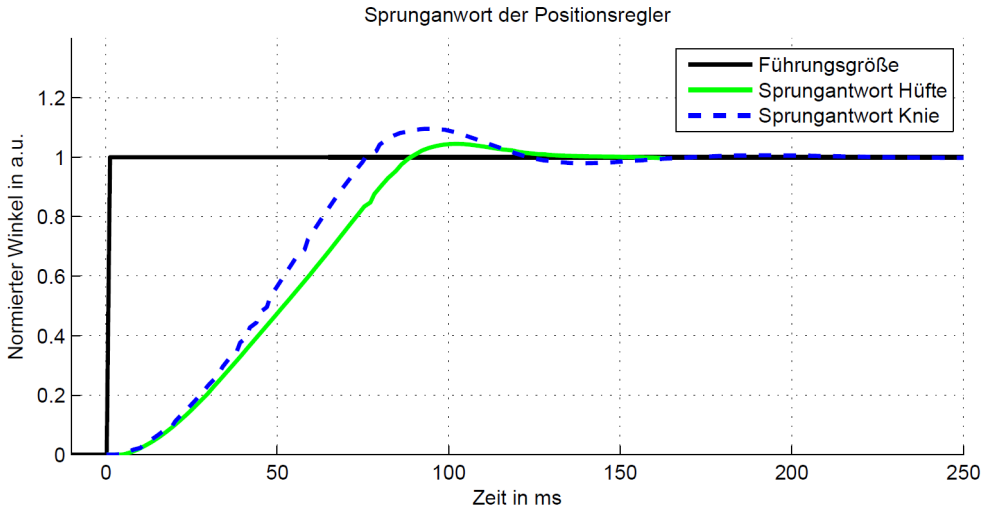


Abbildung 3-11: Normierte Sprungantworten der Lageregler für das Hüft-und Kniegelenk. Ein Führungsgrößensprung von 10 000 Inkrementen (≙ 500 Grad motorseitig) wurde vorgegeben.

Es sollte angemerkt werden, dass deutlich schnellere Regler dimensioniert werden könnten. Jedoch zeigten Testläufe zur Generierung der Bewegungsmuster, dass zu schnell eingestellte Regler zu ruckhaften Positionsverläufen führen. Die Ursache liegt an geringen Sendeverzögerungen bei der Vorgabe der Bewegungsparameter, welche von Zeit zu Zeit auftreten. Diese sind wiederum auf das nicht echtzeitfähige Verhalten des Betriebssystems zurück zu führen.
Für eine detailliertere Erklärung dieser Ursache siehe Kapitel 3.3 Erstellung der Bewegungsmuster.

3.3 Erstellung der Bewegungsmuster

In diesem Kapitel werden die einzelnen Schritte zur Generierung der Winkelverläufe, bis hin zur praktischen Umsetzung der Bewegungsmuster durch die Motion-Controller, detailliert erläutert.

Beschaffung / Korrektur der Winkelverläufe

Der Winkelverlauf zur Generierung des Bewegungsmusters, gesammelt von Svehlik [34], wurde von der Klinik Judendorf-Straßengel zur Verfügung gestellt. Die Daten wurden durch eine Ganganalyse mit 12 Kindern und Jugendlichen im Alter von 8 bis 15 Jahren erhoben und wurden in Form eines gemittelten Winkelverlaufes mit 100 Messwerten für einen Gangzyklus bereits gestellt. Für eine detaillierte Beschreibung des Gangzyklus wird auf [35] verwiesen.
Die Winkeldefinitionen der einzelnen Gelenke ist in Abbildung 3-12 dargestellt. Es sollte angemerkt werden, dass die Flexion eines Gelenks als positiver Winkel definiert ist.

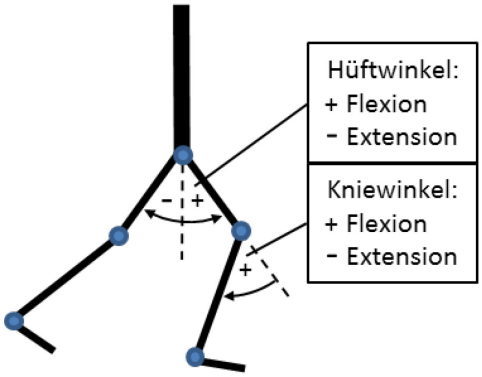


Abbildung 3-12: Definition der Winkel für die Hüft- und Kniegelenke.

Bei der Analyse der Daten wurde beobachtet, dass der Winkelverlauf der Hüfte nicht periodisch ist, beziehungsweise beim Übergang zwischen zwei Schrittzyklen, von Periode 1 zu Periode 2, eine Unstetigkeit aufweist. Die Perioden sind in Abbildung 3-13 dargestellt. Die Winkelverläufe beim Gehen variieren von Schritt zu Schritt und sind nicht exakt gleich. Durch herausschneiden und mitteln der einzelne Gangzyklen entsteht dadurch eine Unstetigkeit beim Übergang. Der Winkelverlauf ist in Abbildung 3-13 dargestellt.
Um den Winkelverlauf der Hüfte zu korrigieren wurden folgenden Modifikationen durchgeführt:

- **Bereichsauswahl:** Beginn des Gangzyklus [0 %] bis zum Nulldurchgang des Winkel-verlaufes [36 %]. (Siehe Abbildung 3-13: Rot unterlegt)
- **Skalierung:** Skalierung der Auswahl, sodass ein stetiger Übergang zwischen den einzelnen Perioden entsteht. ($\text{Skalierungsfaktor} = 0{,}975$)

Durch diese Korrektur ergab sich eine maximale Änderung der originalen Winkeldaten von $0{,}85\ \text{Grad}$. Der Vergleich zwischen korrigierten und unkorrigierten Winkelverlauf der Hüfte ist in Abbildung 3-14, sowie der Winkelverlauf am Kniegelenk in Abbildung 3-15 abgebildet.

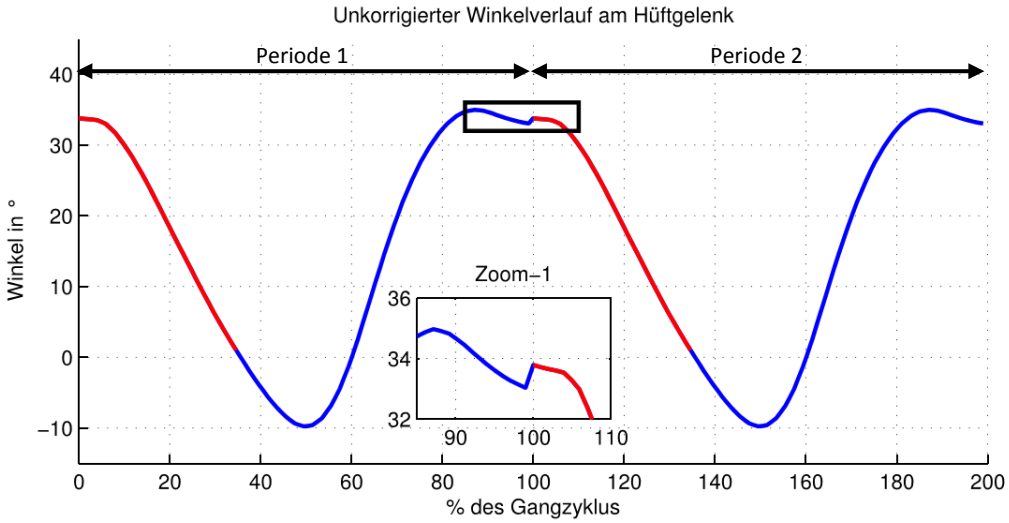


Abbildung 3-13: Unkorrigierter Winkelverlauf des Hüftgelenks für 2 Gangzyklen (blau/rot) Auswahl für die Korrektur (rot); Zoom-1: Detailansicht beim Übergang zwischen zwei Perioden.

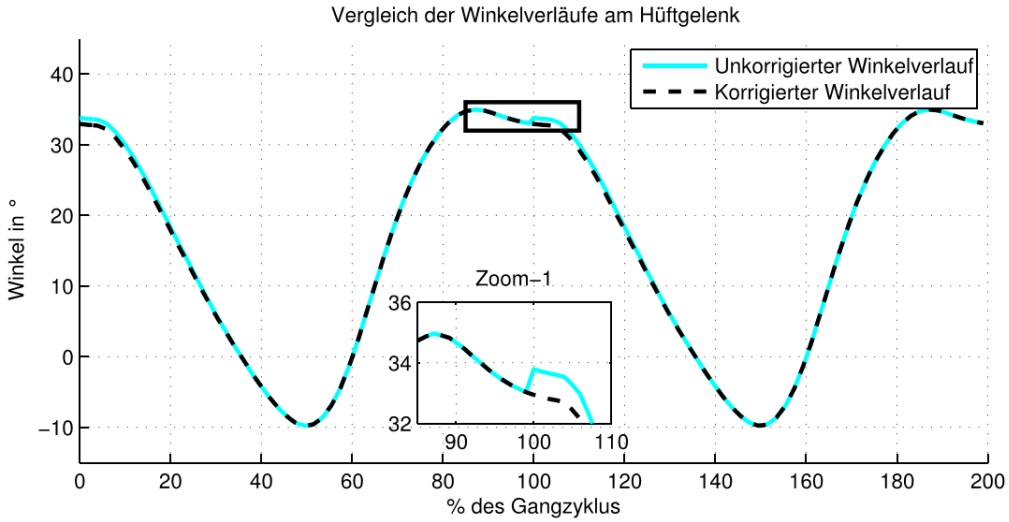


Abbildung 3-14: Vergleich des korrigierten und unkorrigierten Winkelverlaufes am Hüftgelenk für 2 Gangzyklen Zoom-1: Detailansicht beim Übergang zwischen zwei Perioden.

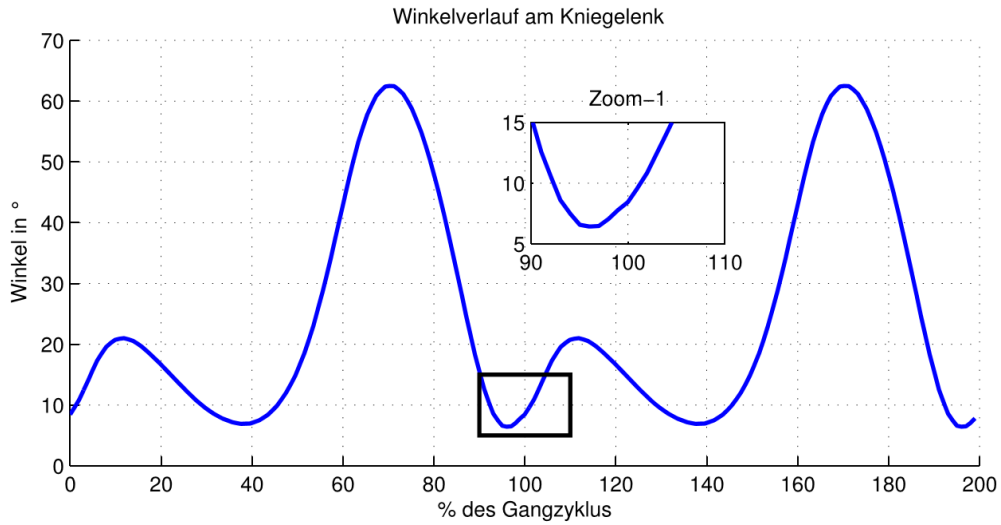


Abbildung 3-15: Winkelverlauf am Kniegelenk für 2 Gangzyklen Zoom-1: Detailansicht beim Übergang zwischen zwei Perioden.

Konzept der Bewegungserstellung

Bei der Erstellung des Antriebskonzepts beziehungsweise bei der Beschaffung der nötigen Hardware wurde davon ausgegangen, dass der Motion-Controller die vorgegebenen Positionswerte kontinuierlich abfährt, wenn die Bewegungsparameter (Position, maximale Geschwindigkeit und maximale Beschleunigung) für den darauffolgenden Zeitschritt zyklisch gesendet werden. Bei Vorversuchen stellte sich jedoch heraus, dass zwar durch Vorgabe der Bewegungsparameter ein komplexer Bewegungsverlauf vorgegeben werden kann, aber dies nur als Punkt-zu-Punkt Positionierung erfolgt. Dieses Problem soll am folgenden Beispiel veranschaulicht werden.
Der Motion-Controller empfängt von der übergeordneten Steuerung eine Zielposition, sowie die Bewegungsparameter. Der Antrieb beschleunigt nun die Motorwelle mit der vorgegebenen Beschleunigung auf die vorgegebene Maximalgeschwindigkeit. Der Controller hält diese Geschwindigkeit solange konstant, bis mit der vorgegebenen Bremsbeschleunigung die Zielposition im Stillstand exakt erreicht wird.

Nun können zwar mehrere Positionierbefehle nacheinander vorgegeben werden, jedoch bleibt der Antrieb bei jeder vorgegebenen Zielposition stehen und führt erst danach die neue Positionierung durch. Dadurch ergibt sich einerseits ein sehr unruhiger Positionsverlauf und andererseits werden die vorgegebenen Positionen nicht in der vorgegebenen Zeit erreicht. Zur Veranschaulichung ist ein exemplarischer Positionsverlauf in Abbildung 3-16 dargestellt.

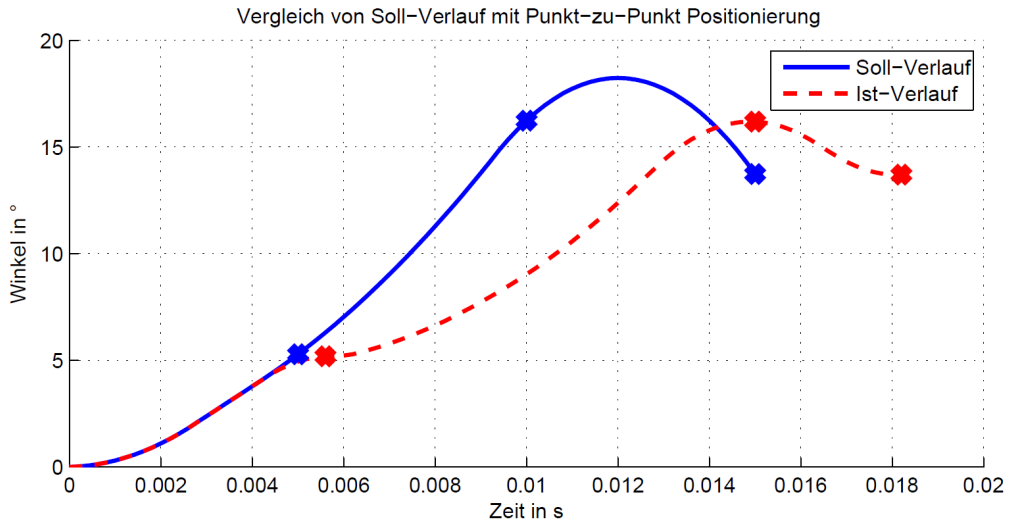


Abbildung 3-16: Vergleich zwischen Soll-Verlauf und Ist-Verlauf bei einer Punkt-zu-Punkt Positionierung; Soll-Verlauf (blau durchgehend); Zielpositionen (blaue Kreuze); Ist-Verlauf (rot gestrichelt); Zeitpunkt der tatsächlich erreichten Zielpositionen (rote Kreuze).

Um diesen Effekt der Punkt-zu-Punkt Positionierung zu umgehen, wurden folgende Adaptionen durchgeführt:

- Deaktivierung des Rampengenerators für positive und negative Beschleunigungen, durch Vorgabe des Maximalwertes ($30\,000\ U/s^2$) für die Verzögerung [20]
- Zyklisches Überschreiben der aktuellen Zielposition. Vor Erreichen der vorgegebenen Zielposition, wird die aktuelle Zielposition durch die neue Zielposition ersetzt.
- Aufgrund des nicht echtzeitfähigen Verhaltens des Betriebssystems, erfolgt die Vorgabe der neuen Bewegungsparameter nicht immer zeitlich exakt. Deshalb wird die Zielposition gestreckt, indem diese um einen konstanten Wert weiter weg gesetzt wird

Das Konzept der adaptierten Bewegungserstellung ist in Abbildung 3-17 dargestellt.

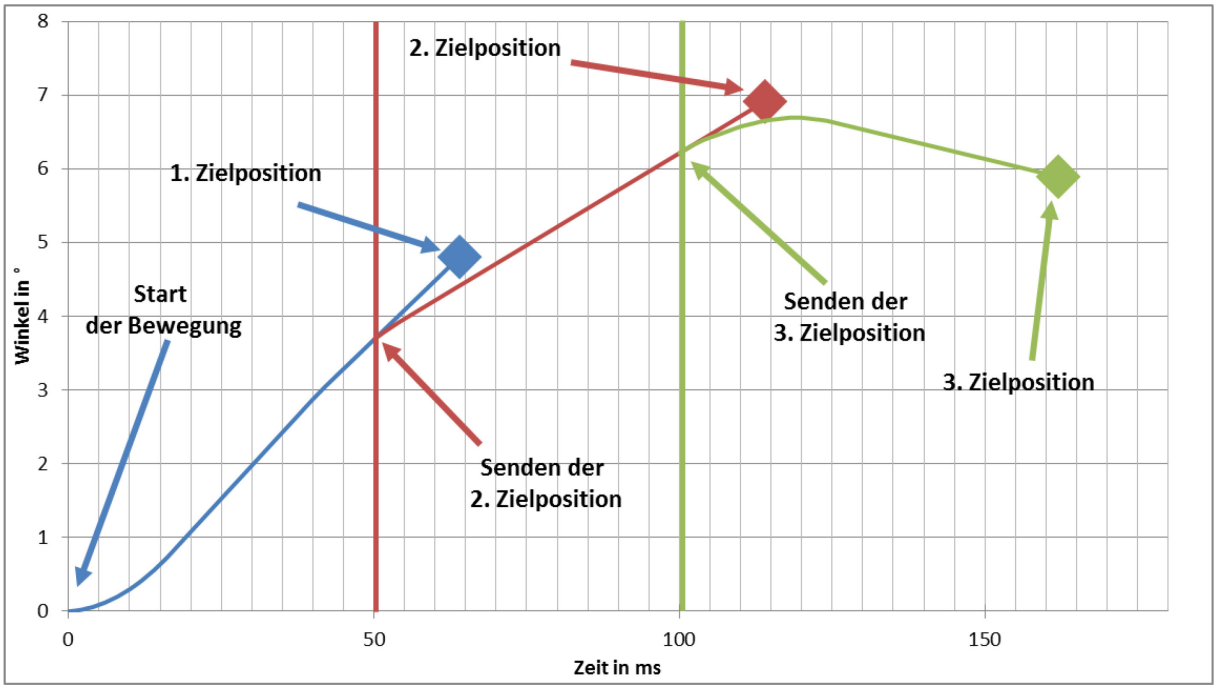


Abbildung 3-17: Erstellung eines komplexen Winkelverlaufes durch zyklisches Senden neuer Zielpositionen. Jede Farbe entspricht dem Verlauf der Winkelposition durch das Senden einer neuen Zielposition.

Durch die Deaktivierung des Rampengenerators ist die Vorgabe von Winkelverläufen mit stückweise quadratischen Verläufen nicht mehr möglich. Es können nur noch lineare Winkelverläufe vorgegeben werden. Um dennoch einen sanften Übergang zwischen den einzelnen Bewegungsbefehlen zu erhalten, wurden die Regler der Motion-Controller relativ langsam eingestellt (siehe Kap 3.2.1.4).

Generierung der Bewegungsparameter

Die Vorgabe der Bewegungsparameter an die Motion-Controller erfolgt durch die übergeordnete Steuerung. Aufgrund der limitierten Datenübertragungsrate und Prozessorgeschwindigkeit, kann der Winkelverlauf nicht mit beliebig hoher zeitlicher Auflösung vorgegeben werden, weshalb dieser abgetastet wird.

Als unterste Schranke für die minimale Sendeperiode (Zeitabstand zwischen zwei Sendevorgängen) wurde ein Wert von 50 ms bestimmt. Bei der vorgegebenen maximalen Kadenz von 120 Schritte/min ergeben sich somit 20 Positionswerte für einen Gangzyklus.

Um den Fehler durch die Abtastung möglichst gering zu halten, werden die Abtastzeitpunkte so gelegt, dass die Maxima und Minima der Originalkurve in den abgetasteten Werten enthalten sind. Zwischen den gewählten Punkten wird dann stückweise linear interpoliert. Der Verlauf der Originalkurve, der Linearinterpolation sowie der Fehler sind in Abbildung 3-18 für das Kniegelenk und in Abbildung 3-19 für das Hüftgelenk dargestellt.

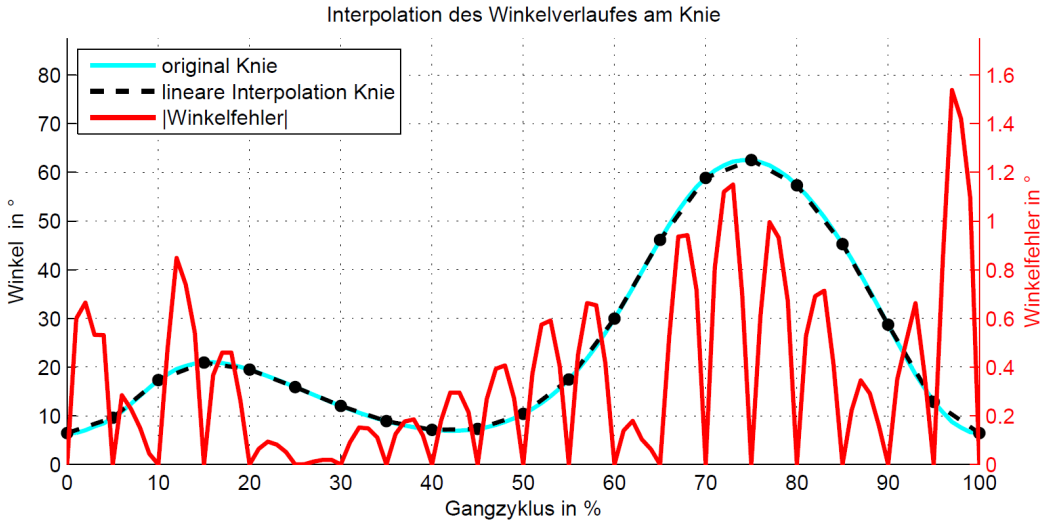


Abbildung 3-18: Interpolation des Kniewinkelverlaufes.

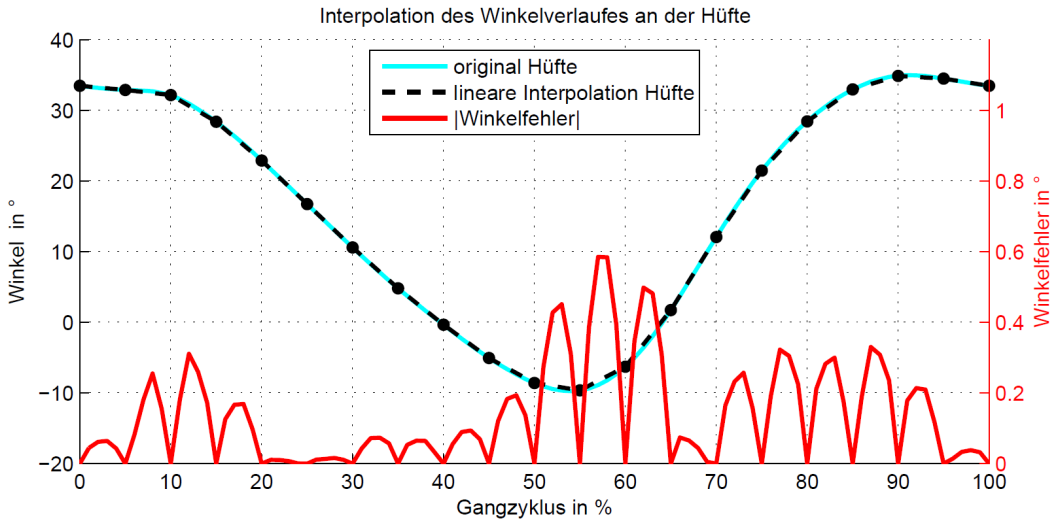


Abbildung 3-19: Interpolation des Hüftwinkelverlaufes.

Durch die Interpolation ergeben sich maximale Fehler von zirka 1,6 Grad für das Kniegelenk und zirka 0,6 Grad für das Hüftgelenk.

Wenn man die Winkelverläufe eines physiologischen Gangbildes betrachtet, erkennt man, dass die Verläufe der einzelnen Gangzyklen nicht exakt gleich sind, sondern variieren.

Bei der wiederholten Durchführung von mehreren Gangzyklen an einem Stück (Inter-Trial), variiert der Winkelverlauf mit einer Standardabweichung von bis zu $\sigma \approx 2\,\mathrm{Grad}$ beim Hüftgelenk und $\sigma \approx 3\,\mathrm{Grad}$ beim Kniegelenk [36]. Der Verlauf der Standardabweichung für das Hüft- und Kniegelenk ist in Abbildung 3-20 dargestellt.

Die Abweichungen durch die Interpolation befinden sich somit innerhalb des physiologischen Bereiches.

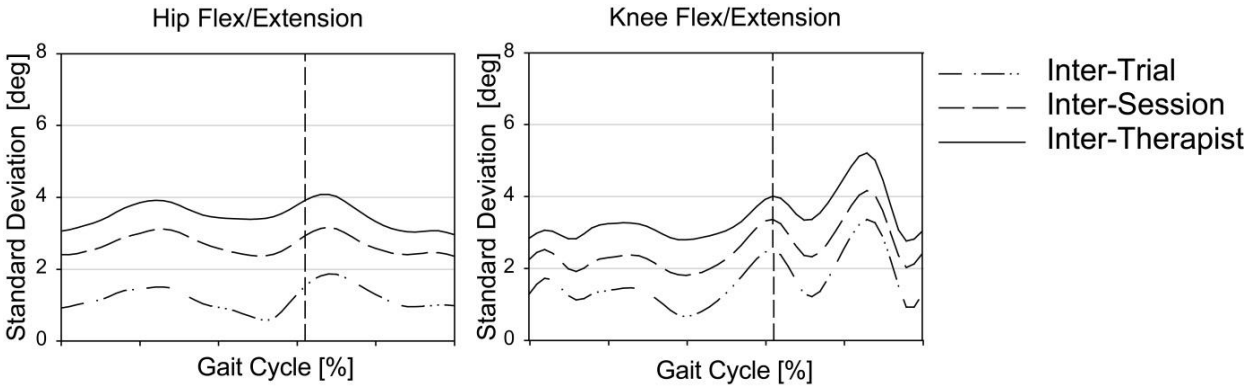


Abbildung 3-20: Standardabweichung der Winkelverläufe der Hüfte (links) und des Knies (rechts), adaptiert von [36].

Für die Berechnung der Geschwindigkeiten wird der Vorwärts-Differentialquotient nach Formel (1) mit den abgetasteten Positionen verwendet. Die berechnete Geschwindigkeit v_i entspricht somit jener Geschwindigkeit die benötigt wird, um von der aktuellen Position s_i zur nächsten Position s_{i+1} in der Zeit Δt zu gelangen. Dabei entspricht Δt der Zeit zwischen zwei Sendevorgängen bei maximaler Kadenz (120 Schritte/min) und beträgt $\Delta t = 50$ ms. Das berechnete Geschwindigkeits-Profil für eine Kadenz von 120 Schritte/min ist in Abbildung 3-21 dargestellt.

$$\mathrm{v_i} = \frac{\Delta s}{\Delta t} = \frac{s_{i+1} - s_i}{\Delta t} \tag{1}$$

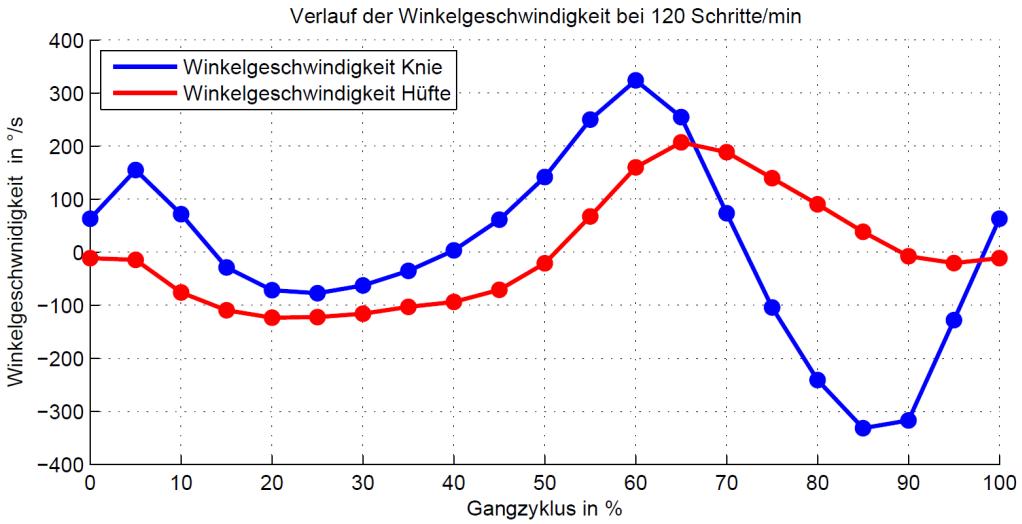


Abbildung 3-21: Verlauf der Winkelgeschwindigkeit bei 120 Schritte/min.

Zur Vorgabe einer variablen Geschwindigkeit muss das Geschwindigkeitsprofil und die Sendeperiode entsprechend der Zielkadenz skaliert werden. Die Sendeperiode T_{Ziel} verhält sich indirekt proportional zur Zielkadenz K_{Ziel} und wird nach Formel (2), ausgehend von der Sendeperiode $\mathrm{T_{120}} = 50\ ms$ bei der Kadenz $K_{120} = 120$ Schritte/min berechnet.

$$\mathrm{T_{Ziel}} = \mathrm{T_{120}} \cdot \frac{K_{120}}{K_{Ziel}} = 0{,}05 \cdot \frac{120}{K_{Ziel}} \tag{2}$$

Das Geschwindigkeitsprofil $v_{i,ziel}$ ist direkt-proportional zur Zielkadenz K_{Ziel} und kann nach Formel (3), ausgehend von der Geschwindigkeit $v_{i,120}$ bei der Kadenz $K_{120} = 120$ Schritte/min berechnet werden.

$$\mathrm{v}_{i,Ziel} = \mathrm{v_{i,120}} \cdot \frac{K_{Ziel}}{K_{120}} = \mathrm{v_{i,120}} \cdot \frac{K_{Ziel}}{120} \tag{3}$$

3.4 Entwicklungsumgebung LabVIEW

Als Entwicklungsumgebung für die Software wurde LabVIEW 2012 in Version 12.0 (32 Bit) (National Instruments Corp., Austin, USA) gewählt. Die Eigenschaften der Entwicklungsumgebung haben einen wesentlichen Einfluss auf den Software-Lebenszyklus-Prozess, sowie auf Design- und Konzeptentscheidungen.

LabVIEW ist eine graphische Programmiersprache (G-Sprache) und wurde vorwiegend für die Anwendung in der Messtechnik sowie Regel- beziehungsweise Automatisierungstechnik entwickelt. Erstellte Programme werden als „Virtuelle Instrumente“ (VI) bezeichnet und bestehen aus zwei Komponenten. Die erste Komponente ist das Frontpanel und repräsentiert die grafische Benutzerschnittstelle (GUI). Der zweite Teil wird Blockdiagramm genannt und stellt den graphischen Programmcode dar. LabVIEW arbeitet nach dem Datenflussmodell, welches hierarchisch und modular aufgebaut ist. Das heißt, die Abarbeitungsreihenfolge der einzelnen Funktionsblöcke wird durch ihre Datenabhängigkeit bestimmt. [37], [38]
Die Eigenschaften von LabVIEW können wie folgt beschrieben werden [37]–[40]:

Vorteile:

- Gute Unterstützung für Entwicklung von Parallelitäten und dadurch optimale Ausnutzung der System-Ressourcen (z.B. Multi-Threading)
- Vergleichbare Performance mit anderen Hochsprachen (z.B.: C/C++)
- Überprüfung und direkte Anzeige während der Programmierung, ob die Übersetzung in Maschinen-Code erfolgreich möglich ist. Somit werden Fehler während der Programmierung früher erkannt und minimiert
- Breite Unterstützung von Soft und Hardwareschnittstellen
- Implementierungsdetails wie unter anderem Speicherverwaltung, Zeigerhandling werden durch LabVIEW selbst verwaltet. Dadurch ergibt sich eine Minimierung der Fehlereinflüsse

Nachteile:

- Objektorientierte Programmierung wird nur schlecht unterstützt. Deshalb ist fast nur ein Top-Down-Design beziehungsweise eine Bottom-Up Integration möglich.
- Kleine Änderungen können einen großen Programmieraufwand nach sich ziehen. Oftmals ist eine Neustrukturierung notwendig.

3.5 EN 62304: Medizingeräte-Software Software-Lebenszyklus-Prozesse

3.5.1 Einleitung

Die *EN 62304: Medizingeräte-Software Software-Lebenszyklus-Prozesse* stellt einen Rahmen von Lebenszyklus-Prozessen zur Verfügung, um eine sichere Entwicklung und Wartung einer Medizinprodukte-Software zu gewährleisten [41]. Die Einhaltung dieser Norm basiert auf der verbindlichen Forderung in der Medizinprodukterichtlinie 93/42/EWG:

> *„Bei Produkten, die Software enthalten oder bei denen es sich um medizinische Software an sich handelt, die Software entsprechend dem Stand der Technik validiert werden muss, wobei die Grundsätze des Software-Lebenszyklus, des Risikomanagements, der Validierung und Verifizierung zu berücksichtigen sind.“*
> Zitat aus [12].

Dabei deckt die EN 62304 die Entwicklung und Wartung von Medizinprodukte-Software ab, wenn die Software ein eigenständiges Medizinprodukt oder ein Teil eines Medizinprodukts ist [41].
Eine Medizinprodukte-Software wird folgendermaßen definiert:

> *„Ein SOFTWARE-SYSTEM, das entwickelt wurde, um in das in der Entwicklung befindliche MEDIZINPRODUKT integriert zu werden, oder das für die Benutzung als selbständiges MEDIZINPRODUKT vorgesehen ist.“*
> Zitat aus *[41]*.

Die in dieser Arbeit zu entwickelnde Software steuert ein aktives therapeutisches Medizinprodukt und ist per Definition nach Medizinprodukterichtlinie 93/42/EW selbst ein Medizinprodukt und somit eine Medizinprodukte-Software. Die Entwicklung und Dokumentation der Software erfolgt nach EN 62304. Wobei die Umsetzung der EN 62304 hierbei nicht ausreicht, um die Konformität der Anforderungen durch die Medizinprodukterichtlinie 93/42/EW nachzuweisen, sondern deckt nur einen Teil der grundlegenden Anforderungen ab.

Die Einhaltung der EN 62304 setzt die Entwicklung der Software im Rahmen eines Qualitätsmanagement-Systems (z.B. EN ISO 13485 [42]), sowie die Anwendung des Risikomanagement-Prozesses nach EN ISO 14971 [43] voraus. Als allgemeine Anforderung wird eine Software-Sicherheitsklassifizierung (siehe Kapitel 3.5.3) verlangt. Die Detailtiefe und somit der Umfang des Aufwandes für die Umsetzung der im Folgenden beschriebenen Prozesse richtet sich nach der Software-Sicherheitsklasse.

Die EN 62304 definiert folgende Software-Lebenszyklus-Prozesse [41]:

- Software-Entwicklungs-Prozess:

  Dieser beschreibt den Prozess der Software-Entwicklung, beginnend mit der Analyse der Software-Anforderungen bis hin zur Verifizierung und Freigabe der Software.
- Software-Wartungs-Prozess:

  Dieser beschreibt die Prozesse der Wartung von freigegebener Software.
- Software-Risikomanagement-Prozess:

  Dient zur Identifikation, Analyse und Dokumentation der Software-Komponenten, die zur Gefährdungssituation beitragen könnten, sowie zum Ableiten von Risiko-Kontrollmaßnahmen und der Verifikation dieser.
- Software-Konfigurationsmanagement-Prozess

  Dient zur Identifizierung der Konfiguration der Software und der Dokumentation dieser. Zusätzlich werden Anforderungen bezüglich der Änderung dieser Konfigurationselemente gestellt.
- Problemlösungs-Prozess von Software

  Dieser Prozess beschreibt, auf welche Weise Probleme analysiert und gelöst werden, die während Entwicklung, Wartung oder in anderen Prozessen auftreten.

Im Rahmen dieser Arbeit wurde kein Qualitätsmanagement-System erstellt, um den Umfang der Arbeit zu begrenzen. Der Software-Wartungs-Prozess beschreibt die Wartung der Software erst nach Freigabe dieser. Der Fokus dieser Arbeit liegt in der Entwicklung der Software und umfasst keine nachgelagerten Phasen. Somit wurde der Software-Wartungs-Prozess ebenfalls nicht behandelt.

Im Folgenden werden die einzelnen Prozesse, welche für die Implementierung der Software eingesetzt wurden, genauer beschrieben.

3.5.2 Software-Risikomanagement-Prozess

Die EN 62304 [41] fordert die Umsetzung eines Risikomanagementprozesses nach EN ISO 14971 [43] und führt weitere Anforderungen an die Softwareentwicklung, durch den Software-Risiko-Managementprozess, ein. In diesem Kapitel wird der generelle Ablauf des Risikomanagement-prozesses nach EN ISO 14971 erläutert und um die zusätzlichen Anforderungen der EN 62304 erweitert.

Als Risiko wird die Kombination des Schweregrad eines Schadens mit der Wahrscheinlichkeit des Auftretens dieses Schadens bezeichnet [43].

Die Schritte des Risikomanagement-Prozesses sind in Abbildung 3-22 dargestellt und werden in den folgenden Kapiteln erklärt.

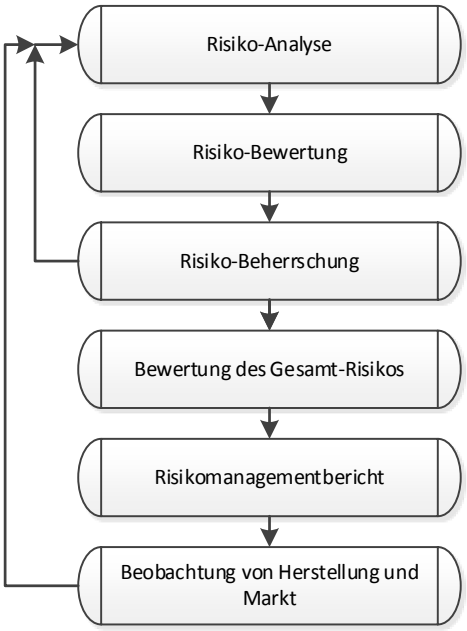


Abbildung 3-22: Ablaufdiagramm des Risikomanagement-Prozesses, nach [43].

3.5.2.1 Risiko-Analyse

Zweckbestimmung festlegen

Bevor mit der eigentlichen Aktivität der Risikoanalyse begonnen werden kann, muss die Zweckbestimmung definiert werden. Dazu zählen unter anderem auch die Beschreibung der Anwender, der Patienten sowie der Gebrauchsumgebung. [44]

Für die vollständige Beschreibung der Zweckbestimmung wird auf das Dokument „Spezifikation der Gebrauchstauglichkeit.docx“ [45] verwiesen.

Im Folgenden wird ein Auszug des medizinischen Zweckes als Anhaltspunkt präsentiert:

> *„Förderung der Bewegungsentwicklung von Kleinkindern mit infantiler Zerebralparese und Hirnschäden um eine bestmögliche Funktionalität des Bewegungsapparats zu erreichen. Linderung von Beschwerden durch abnorme Haltungs- und Bewegungsmuster und Erhöhung einer eventuellen Gehzeit, Schrittabfolge wie auch Stehdauer[…].“*
> Zitiert nach [45]

Gefährdungen identifizieren:

Der nächste Schritt ist die Identifizierung von Gefährdungen durch das Medizinprodukt, wobei dies neben den Normalbetrieb (NC) auch den ersten Fehlerfall (SFC) und den vorhersehbaren Missbrauch umfasst [43]. Die Software selbst stellt dabei keine Gefährdung dar, sondern trägt nur zur Gefährdungssituation bei. Aus Sicht der EN 62304 müssen alle Software-Komponenten, welche zu einer Gefährdungssituation beitragen, identifiziert werden. Es müssen nicht nur die möglichen Ursachen, sondern auch die Folge von Ereignissen, bis hin zum Schaden identifiziert werden. [46]

Falls Software unbekannter Herkunft (SOUP)[3] für die Medizin-Produkte-Software verwendet wird, so müssen veröffentlichte Listen von Anomalien dieser SOUPs auf einen möglichen Beitrag zu einer Gefährdungssituation evaluiert werden. [41]

Durchführung

Als Bewertungsgrundlage für die Risikoanalyse wurde die Software-Architektur verwendet. Anhand der in der Architektur festgelegten Anforderungen und Funktionen wurden für jede Software-Komponente mögliche Fehler, beziehungsweise Ursachen dieser Fehler identifiziert.
Die Aktivitäten der Risikoanalyse wurden als Team in einem Brainstorming-Prozess durchgeführt, wobei als methodischer Ansatz die Fehlermöglichkeits- und Einflussanalyse (FMEA) verwendet wurde.

Die FMEA ist eine Bottom-Up-Methode, welche von einem Ereignis oder einer einzelnen Komponente (Tätigkeit eines Anwenders oder einer Software-Komponente) ausgeht und untersucht, welche Folgen bei einem Fehler auftreten könnten. Man stellt sich die Frage: „Was geschieht, wenn ein Fehler eintritt?“ Dieser Vorgang wird solange wiederholt, bis die Ursachenkette bis zum Schadensereignis verfolgt wurde. In Abbildung 3-23 ist der methodische Ansatz der FMEA dargestellt. [46], [47]

[3] Die EN 62304 definiert SOUP folgendermaßen [41]:
„Software-Komponente, die bereits entwickelt und allgemein verfügbar ist und die nicht entwickelt wurde, um in das Medizinprodukt integriert zu werden (auch bekannt als ‘Off-The-Shelf Software‘), oder bereits entwickelte Software, für die angemessene Aufzeichnungen zum Entwicklungs-Prozess nicht verfügbar sind.“

Bei der Identifizierung der Folge von Ereignissen sollte angemerkt werden, dass für einen Fehler beziehungsweise für eine Ursache unterschiedliche Folgen auftreten können. Analog dazu können bei einer bestehenden Gefährdungssituation unterschiedliche Schäden mit unterschiedlicher Eintrittswahrscheinlichkeit auftreten. Somit sollten nicht nur der größte potentielle Schaden einer Gefährdung, sondern auch mögliche geringere Schäden analysiert werden.

Abbildung 3-23: Methodischer Ansatz der FMEA. Entnommen aus [47].

Risiken abschätzen:

Nachdem die Gefährdungen identifiziert wurden, müssen deren Risiken abgeschätzt werden. Anhand der Folge von Ereignissen, sowie der Beschreibung des Schadens wird untersucht, welcher potentielle Schaden mit welcher Wahrscheinlichkeit auftreten könnte. Die Schadensausmaße und Wahrscheinlichkeiten werden anschließend einer qualitativen Einteilung unterzogen und mit Hilfe der Risikobewertungsmatrix einer Risikostufe zugeordnet. Die Einteilungen der Schadensausmaße und der Wahrscheinlichkeiten sind in Tabelle 4 und Tabelle 5 beschrieben. In Tabelle 6 ist die verwendete Risikobewertungsmatrix dargestellt. Für eine genauere Erklärung wird auf das folgende Kapitel verwiesen.

Tabelle 4: Beschreibung der qualitativen Schadensfolgen zur Risikobewertung.

Nummer	Bezeichnung	Bedeutung
1	gering	Keine Verletzung
2	mittel	Leichte Verletzung
3	schwer	Schwere Verletzung oder Tod
4	katastrophal	Schwere Verletzung oder Tod für mehrere Personen

Tabelle 5: Beschreibung der qualitativen Eintrittswahrscheinlichkeiten zur Risikobewertung.

Nummer	Bedeutung
1	Unglaublich
2	Unwahrscheinlich
3	Selten
4	Gelegentlich
5	Manchmal
6	Häufig

Tabelle 6: Risikobewertungs-Matrix.

Eintrittswahrscheinlichkeit								Risikostufe
	6	häufig					I	
	5	manchmal						
	4	gelegentlich						
	3	selten					II	
	2	unwahrscheinlich					III	
	1	unglaublich					IV	
			gering	mittel	schwer	katastrophal		
			1	2	3	4		
			Schadensfolge					

3.5.2.2 Risiko-Bewertung

Für jede identifizierte Gefährdungssituation muss anhand festgelegter Akzeptanzkriterien entschieden werden, ob eine Minimierung des Risikos zu erfolgen hat. Dies erfolgt mit Hilfe der Risikobewertungsmatrix. Die verwendete Risikobewertungsmatrix definiert vier Risikostufen, welche in Tabelle 7 beschrieben sind. Dabei wird das Konzept von "As low as reasonable achievable" (ALARA) und "As low as reasonable practicable" (ALARP) verwendet. Ein Risiko fällt in die ALARA –Stufe wenn keine Reduzierung vernünftigerweise erreichbar ist. Im Vergleich dazu fällt ein Risiko in den ALARP Bereich wenn eine Minimierung nicht mehr vernünftigerweise praktikabel ist.

Tabelle 7: Beschreibung der Risikostufen.

Risikostufe	Bedeutung
I	Nicht akzeptierbares Risiko
II	ALARA: kann nur toleriert werden, wenn eine Vermeidung nicht vernünftigerweise erreichbar ist
III	ALARP: kann nur toleriert werden, wenn eine Vermeidung nicht vernünftigerweise praktikabel ist
IV	Tolerierbar: vernachlässigbares Risiko

Für die Festlegung der Akzeptanz wurden folgende Kriterien definiert:

> *„Aufgrund des zu erwartenden medizinischen Nutzens sind keine Risiken akzeptierbar, die der Risikostufe **I** oder **II** zugeordnet werden. Alle Risiken der Stufe **IV** werden toleriert. Risiken der Stufe **III** werden akzeptiert, wenn eine Reduzierung nur mit einem unproportional hohen Aufwand im Vergleich zum Nutzen erreichbar ist. Jedoch sollte unter der Berücksichtigung des Alters der Patienten, das Ziel eine Minimierung der Risiken dieses Bereichs sein."*
>
> Zitiert nach [48].

3.5.2.3 Risiko-Beherrschung

In dieser Phase werden für Gefährdungen mit nicht akzeptierbarem Risiko, geeignete Risiko-Kontrollmaßnahmen zur Reduzierung der Risiken erarbeitet. Bei der Wahl der Risiko-Kontrollmaßnahmen sollte zuerst eine unmittelbare Maßnahme (z.B. konstruktive Einschränkung des Winkelbereichs) angestrebt werden. Falls dies nicht möglich ist, sollte das Risiko durch eine mittelbare Maßnahme (z.B. Überwachung des maximalen Drehmoments) erfolgen. Falls auch dies nicht möglich ist, können Warnhinweise am Gerät beziehungsweise in der Gebrauchsanweisung auf das vorhandene Restrisiko hinweisen. [46]

Für die umgesetzten Maßnahmen wird anschließend das Restrisiko bewertet und untersucht, ob Rückwirkungen neue Risiken hervorrufen. Entdeckte neue Risiken müssen den beschriebenen Prozess nochmals durchlaufen.

Um die bisher beschriebenen Schritte (Risiko-Analyse, Risiko-Bewertung und Risiko-Beherrschung) zu dokumentieren, sowie die Anforderungen der EN 62304 in Bezug auf die Rückverfolgbarkeit[4] zu erfüllen, wird ein Risikoanalyse-Protokoll verwendet. Das Protokoll mit einem Beispieldatensatz ist in Tabelle 9 dargestellt. Die Erklärungen der verwendeten Abkürzungen sind in Tabelle 8 zu finden.

Tabelle 8: Erklärung der Abkürzungen im Risikoanalyse-Protokoll.

Abkürzung	Bedeutung
Komp.	**Komponente**
Fkt. / Aufg.	**Funktion / Aufgabe**
NC	**Normal Condition (Normalfall)**
SFC	**Single Fault Condition (erster Fehlerfall)**
E	**Eintrittswahrscheinlichkeit**
S	**Schaden**
R	**Risiko**
RR	**Restrisiko**
ID	**Identifizierung der Risiko-Kontrollmaßnahme**
RW	**Rückwirkung**
SS	**Sicherheitsstufe: 1…konstruktiv, 2…mittelbar, 3…hinweisend 4…Einschränkung des bestimmungsgemäßen Gebrauchs)**
DM	**Durchgeführte Maßnahme**
J	**Ja**
N	**Nein**

[4] Die EN 62304 fordert, dass für jede identifizierte Software-Gefährdung die Rückverfolgbarkeit folgendermaßen dokumentiert wird [41]:

a) Von der Gefährdungssituation zur Software-Komponente
b) Von der Software-Komponente zur spezifischen Software-Ursache
c) Von der Software-Ursache zur Risiko-Kontrollmaßnahme und
d) Von der Risiko-Kontrollmaßnahme zur Verifizierung der Risiko-Kontrollmaßnahme

Tabelle 9: Risikoanalyse-Protokoll des Software-Risikomanagement-Prozesses.

3. Therapiemodus																		
Komp.		**Fkt. / Aufg.**	**Ursache**	**NC / SFC**	**Folge von Ereignissen zur Gefährdungssituation**	**Schaden**	**E**	**S**	**R**	**ID**	**Abhilfe**	**RW**	**SS**	**DM**	**E**	**S**	**RR**
3. Therapiemodus	3.1 Daten laden	Laden der Maximalstromwerte zur Drehmomentenbegrenzung	Laden korrupter Daten	SFC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos: Drehmoment-begrenzung → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen. Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
										RA-V2-14	Minimierung der maximalen Stromabgabe durch interne Strombegrenzung der Endstufe.	N	1	J			

3.5.2.4 Bewertung des Gesamt-Risikos

In diesem Schritt fällt die Entscheidung, ob das Gesamt-Risiko der identifizierten Gefährdungen in Bezug auf den vorgesehen Einsatz und Gebrauch des Produktes akzeptabel ist. Dabei wird nicht nur überprüft, ob alle Einzelrisiken im akzeptablen Risikobereich liegen, sondern es wird die Gesamtheit aller Risiken beurteilt. Dabei ist zu beachten, dass das gleichzeitige Auftreten oder die Kombination von akzeptierbaren Einzelrisiken zu einer Erhöhung des Gesamtrisikos führen können. [47]

3.5.2.5 Risikomanagementbericht

Der Risikomanagementbericht dient dazu, die Ergebnisse des Risikomanagement-Prozesses zusammenzufassen und darzustellen. Er dient also zur Sicherstellung, dass die geforderten Ziele erreicht wurden.

3.5.2.6 Beobachtung von Herstellung und Markt

Durch einmaliges Durchführen der oben beschriebenen Schritte ist der Risikomanagement-Prozess noch nicht abgeschlossen. Einerseits können neue Gefährdungen während der Entwicklung, Herstellung oder durch Beobachtung des Marktes identifiziert werden und andererseits handelt es sich vorerst nur um eine grobe Schätzung der Wahrscheinlichkeiten und Schadensfolgen. Aus diesem Grund müssen Informationen während der Herstellung und in den nachgelagerten Phasen gesammelt werden, um die Einschätzung von Gefährdungen und Risiken anzupassen und gegebenenfalls zu korrigieren.
Gegebenenfalls kann die Weiterentwicklung vom Stand der Technik dazu führen, dass bisher akzeptierte Risiken, nicht mehr als akzeptierbar bewertet werden können.

Für Gefährdungen und Probleme die während der Entwicklung, Herstellung und Beobachtung vom Markt auftreten, muss der Problemlösungs-Prozess angewandt werden. Eine ausführliche Beschreibung ist in Kapitel 3.5.5 Problemlösungs-Prozess für Software zu finden.

3.5.3 Software-Sicherheitsklassifizierung

Die Software-Sicherheitsklassifizierung stellt einen wesentlichen Punkt der EN 62304 dar und ist üblicherweise einer der ersten Schritte der Software-Entwicklung. Je nach Software-Sicherheitsklasse wird die Detailtiefe und somit der Aufwand der Dokumentation für die Entwicklung von medizinischer Software definiert [46].

Die EN 62304 fordert, dass jedes Software-System einer Software-Sicherheitsklasse zugeordnet ist. Wird ein Software-System in Software-Komponenten unterteilt, so muss die Software-Sicherheitsklasse der ursprünglichen Software-Komponente übernommen werden oder eine ausreichende Begründung für die Abweichung angegeben werden.
Die Software kann in die folgenden drei Software-Sicherheitsklassen[5] nach [41] unterteilt werden:

- **A:** Das Software-System kann nicht zu einer Gefährdungssituation beitragen oder es trägt zwar zu einer Gefährdungssituation bei, aber diese führt zu keinem inakzeptablen Risiko. Es werden nur Risiko-Kontrollmaßnahmen außerhalb des Software-Systems berücksichtigt.

- **B:** Das Software-System kann zu einer Gefährdungssituation beitragen, welche zu einem inakzeptablen Risiko führen. Der potentielle Schaden führt jedoch nicht in zu einer schweren Verletzung. Es werden wieder nur Risiko-Kontrollmaßnahmen außerhalb des Software-Systems berücksichtigt.

- **C:** Das Software-System kann zu einer Gefährdungssituation beitragen, welche zu einem inakzeptablen Risiko, mit dem potentiellen Schaden einer schweren Verletzung, führt. Es werden wieder nur Risiko-Kontrollmaßnahmen außerhalb des Software-Systems berücksichtigt.

Als Risiko-Kontrollmaßnahmen außerhalb des Software-Systems werden jene bezeichnet, die nicht durch das Software-System selbst umgesetzt werden. Diese können einerseits Hardwaremaßnahmen (Winkelbeschränkung), aber auch Risiko-Kontrollmaßnahmen eines anderen Software-Systems sein.

Für die Software-Sicherheitsklassifizierung wurde die Risikoanalyse der Masterarbeit von A. Tilp [8] als Ausgangspunkt verwendet. Es wurden alle Risiken durch die Software sowie Risiko-Kontrollmaßnahmen, welche von der Software umgesetzt werden, extrahiert und Fehlfunktionen durch die Software mit einer Wahrscheinlichkeit von 100 % angenommen.

[5] Durch das Amendment 1 der EN 62304 im Jahr 2015 wurde die Sicherheitsklassifizierung überarbeitet. Bei der EN 62304:2007 wurde lediglich der Schweregrad des möglichen Schadens berücksichtigt. Dadurch war es immer möglich einen beliebigen Fall mit dem Resultat eines potentiellen schweren Schaden zu konstruieren. [46]

Demzufolge entfallen ebenso alle Risiko-Kontrollmaßnahmen durch das Software-System selbst. Auf Basis dieser Annahmen erfolgt dann eine Neubewertung der Risiken. Die Software-Sicherheitsklassifizierung wird anschließend auf Basis des potentiellen Schadens der inakzeptablen Risiken durchgeführt. In Tabelle 10 ist ein Beispiel der adaptierten Risikoanalyse dargestellt.

Anmerkung: Die Abkürzungen sind ident zum Risikoprotokoll in Kapitel 3.5.2. Der Aufbau des Protokolls unterscheidet sich vom Protokoll der Software-Risikoanalyse geringfügig, da in der ursprünglichen Version von A. Tilp [8] die Anforderungen der EN 62304 nicht berücksichtig wurden.

Tabelle 10: Risikoprotokoll der Risikoanalyse zur Software-Sicherheitsklassifizierung; Risiko-Kontrollmaßnahmen umgesetzt durch die Software selbst werden weggestrichen und bei der Neubewertung nicht berücksichtigt. Ausfälle oder Fehler mit einer Wahrscheinlichkeit von 100 % sind rot hervorgehoben.

Gefahren-quelle		Gefährdung	NC	SFC	Folgen	E	S	R	Abhilfe	RW	SS	DM	E	S	RR
Funktion	Zuführung von kinetischer Energie	Zu hohe Drehmomentabgabe		x	Verletzung des Patienten - Muskelverletzungen und Frakturen der unteren Extremitäten	4	2	III	~~indirekte Drehmomentbegrenzung durch Begrenzung des Motorstroms (Endstufe intern)~~	N	1	J	3	2	III
									Kraftmessung	N	1	N			
									Drehmomentbegrenzung (Rutschkupplung)	J	1	N			
									Not-Aus	J	2	J			
techn. Lösung	Softwareumgebung	Absturz des Betriebssystems oder der Steuerungssoftware → keine Übertragung von Befehlen über die Schnittstelle		x	Erzeugung von undefinierten Ausgangswerten (Position, Geschwindigkeit, Drehmoment)	4	3	II	~~Bewegungsabschnitt kann nur mittels Befehl von der Steuerung gestartet werden~~	N	1	J	3	2	III
									Konstruktive Eingrenzung des Winkelbereichs	N	1	J			
									Not-Aus	J	2	J			

3.5.4 Software-Entwicklungs-Prozess

Der Software-Entwicklungs-Prozess stellt den wichtigsten und umfangreichsten Prozess der EN 62304 dar. Er legt den Rahmen der Software-Entwicklung fest beziehungsweise enthält Forderungen über die Art und Weise wie Aktivitäten und Aufgaben umgesetzt werden müssen.

Die EN 62304 setzt folgende Aktivitäten voraus, wobei eine separate Aufteilung der Aktivitäten, wie hier aufgelistet, nicht gefordert wird, sondern lediglich deren Inhalt umgesetzt werden muss [41]:

- Planung der Software-Entwicklung
- Analyse der Software-Anforderungen
- Design der Software-Architektur
- Detailliertes Software-Design
- Implementierung und Verifizierung der Software-Einheiten
- Software-Integration und Integrationsprüfung
- Prüfung des Software-Systems
- Software-Freigabe

Anmerkung: Wie bereits in Kapitel 3.5.3 erwähnt, richtet sich die Detailtiefe der geforderten Aktivitäten nach der Software-Sicherheitsklasse. Aus diesem Grund wird das Ergebnis der Software-Sicherheitsklassifizierung bereits vorweggenommen. Die Software wurde in die Sicherheitsklasse B eingestuft.

3.5.4.1 Planung der Software-Entwicklung

Der erste Schritt des Software-Entwicklungs-Prozesses ist die Planung der Software-Entwicklung. Es muss ein Software-Entwicklungsplan erstellt werden, welcher das Vorgehen während der Entwicklung definiert und somit als eine Art Handbuch für die Entwicklung zu sehen ist.

Als Vorgehensmodell wurde das V-Modell[6] gewählt. Dieses stellt ein Gerüst für die Entwicklung der Software zur Verfügung. Um die Vorgehensweise optimal an die Eigenschaften der Entwicklungsumgebung anzupassen, wurden einzelne Teilschritte des V-Modells modifiziert. (siehe Kapitel 3.5.4.5)

Das verwendete Vorgehensmodell ist in Abbildung 3-24 dargestellt. Es beschreibt die einzelnen Aktivitäten des Software-Entwicklungs-Prozesses, sowie die Interaktionen zu den Prozessen des Risikomanagements, des Konfigurationsmanagements und des Problemlösungs-Prozesses. Zum Beispiel wird für die Durchführung der Risikoanalyse das Ergebnis des Entwurfs der Software-Architektur benötigt. Umgekehrt müssen wiederum die abgeleiteten Risiko-Kontrollmaßnahmen als Software-Anforderung übernommen werden. Die einzelnen Aktivitäten des Software-Entwicklungs-Prozesses und deren Schnittstellen werden in den folgenden Kapiteln näher beschrieben.

[6] Ein ausführlicher Vergleich möglicher Vorgehensmodelle ist unter [49] zu finden.

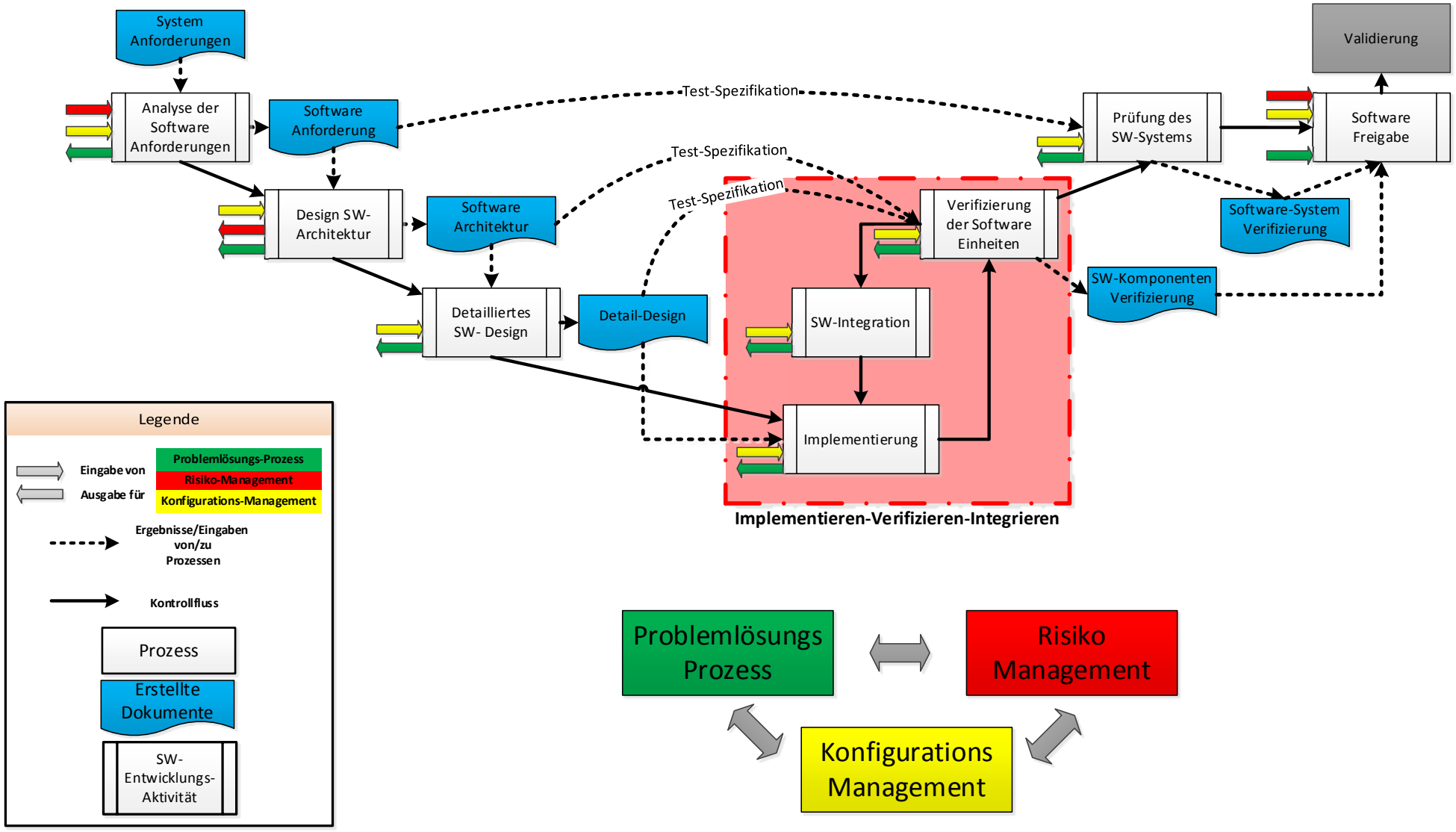


Abbildung 3-24: Darstellung des modifizierten V-Modells des Software-Entwicklungs-Prozesses, sowie die Interaktionen zwischen Problemlösungs-Prozess, Management-Prozess und Konfigurations-Management-Prozess.

Das V-Modell kann in den absteigenden linken Ast und den aufsteigenden rechten Ast unterteilt werden. Der linke Ast entspricht der konstruktiven Phase und der rechte Ast der Testphase.

In der konstruktiven Phase werden die eher allgemeinen Anforderungen sukzessive verfeinert und genauer spezifiziert, um schließlich in der feinsten Ebene, im Detail-Design, für jede einzelne Software-Einheit sehr detaillierte Anforderungen zu erhalten. Die einzelnen Schritte werden in absteigender Reihenfolge durchgeführt, wobei die Ergebnisse des vorherigen Schrittes als Eingaben des Folgeschrittes dienen. Das bedeutet, dass die Software-Architektur aus den Dokumenten der Software-Anforderung entwickelt wird und im weiteren Verlauf, auf Basis dieser Architektur das Detail-Design abgeleitet wird. Die sukzessive Verfeinerung ermöglicht somit die Rückverfolgbarkeit zwischen den unspezifischen System-Anforderungen, bis hin zur detailliert ausgearbeiteten Lösung beziehungsweise Umsetzung in den Software-Einheiten.

Die Implementierung der einzelnen Software-Einheiten erfolgt dann anhand des ausführlich beschriebenen Detail-Designs.

Jeder konstruktiven Phase ist eine Testphase gegenübergestellt. Dies bedeutet, dass dem detaillierten Software-Design die Überprüfung der einzelnen Software-Einheiten gegenübersteht. Es wird überprüft, ob die Anforderungen, welche im Detail-Design spezifiziert wurden, korrekt umgesetzt worden sind. In der nächsten Phase wird die Integration der Software-Komponenten überprüft, das heißt ob die Unterprogramme korrekt laut Software-Architektur integriert wurden. Abschließend wird das Software-System gegen die Software-Anforderungen geprüft. Analog zur konstruktiven Phase wird die Testphase ebenfalls schrittweise durchgeführt, wobei die Reihenfolge umgekehrt ist.

Aufgrund der Eigenschaften der Entwicklungsumgebung LabVIEW, wurden die Aktivitäten ***Implementierung und Verifizierung der Software-Einheiten,*** sowie ***Software-Integration und Integrationsprüfung*** zu einer iterativen Aktivität ***Implementieren-Verifizieren-Integrieren*** zusammengefasst. Somit stehen dieser Aktivität das Software-Detail-Design und auch die Software-Architektur gegenüber.

Der Software-Lebenszyklus-Prozess schließt mit der Freigabe des Software-Systems ab, wobei dies nicht mit der Marktfreigabe des Medizinproduktes verwechselt werden darf. Dafür sind noch einige übergeordnete Schritte notwendig, wie z.B. die Validierung des Lokomotionsgerätes. Es soll angemerkt werden, dass die Validierung kein Teil des Software-Lebenszyklus-Prozesses ist, sondern im Software-Entwicklungsplan nur dargestellt ist, um auf die weiteren Schritte hinzuweisen.

3.5.4.2 Analyse der Software-Anforderungen

Die Analyse der Software-Anforderungen besteht aus drei wesentlichen Schritten, deren Ablauf in Abbildung 3-25 dargestellt ist.

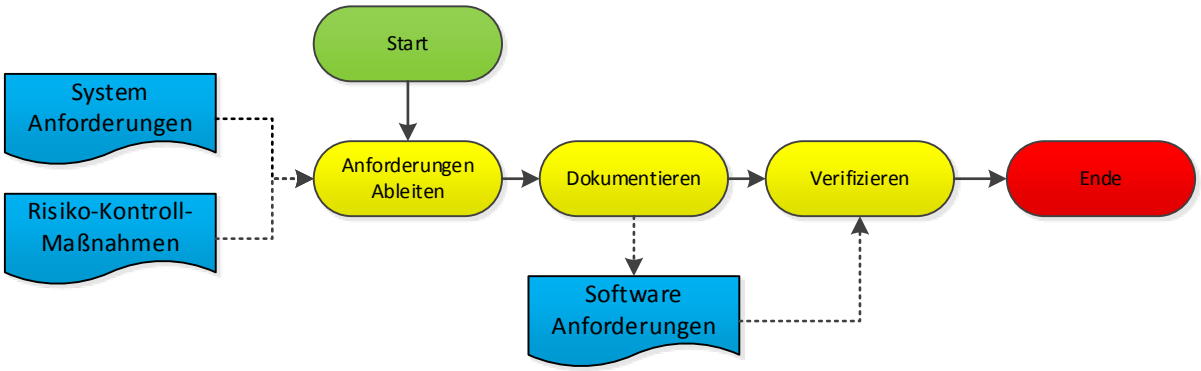


Abbildung 3-25: Ablauf der Analyse der Software-Anforderungen.

Für jedes Software-System müssen die Software-Anforderungen aus den System-Anforderungen abgeleitet werden. Zusätzlich zu den abgeleiteten Anforderungen müssen bereits vorhandene Risiko-Kontroll-Maßnahmen, welche durch die Software umgesetzt werden, als Software-Anforderungen übernommen werden.

Die abgeleiteten Anforderungen werden anschließend im Traceability Dokument [50] eingetragen, um die geforderte Rückverfolgbarkeit der EN 62304 zu erfüllen. Folgende Anforderungen der EN 62304 werden durch das Traceability Dokument erfüllt:

- Rückverfolgbarkeit von den System-Anforderungen beziehungsweise von Risiko-Kontrollmaßnahmen zu den Software-Anforderungen
- Rückverfolgbarkeit von den Software-Anforderungen zur Umsetzung in der Software-Architektur
- Rückverfolgbarkeit von Software-Architektur zum Detail-Design
- Rückverfolgbarkeit von den Software-Systemtests zu den Software-Anforderungen

Bei späteren Änderungen können somit die betroffenen Komponenten für die Überarbeitung eindeutig und schnell identifiziert werden.

Ein Auszug aus dem Traceability-Dokument „Traceability-V-0.2.docx“ [50] ist in Tabelle 11 dargestellt. Erklärungen der Abkürzungen sind in Tabelle 17 im Anhang zu finden. Jeder System-Anforderung steht mindestens eine Software-Anforderung gegenüber und jeder Software-Anforderung eine Software-Komponente. Es können Software-Anforderungen in unterschiedlichen Komponenten umgesetzt werden beziehungsweise durch das Zusammenspiel mehrerer Software-Komponenten realisiert werden (siehe Tabelle 11 rote Box). Jede Software-Anforderung wird durch mindestens einen System-Test verifiziert und das Ergebnis ist in die Tabelle eingetragen.

Tabelle 11: Auszug aus dem Traceability-Dokument: „Traceability-V-0.2.docx“ [50]. Erklärungen der Abkürzungen sind in Tabelle 17 im Anhang zu finden.

System-Anforderungen / Risiko-Kontrollmaßnahmen			Software Anforderungen			Software Komponente	Detail Design	Test		
Ident.		Beschreibung	Ident.		Beschreibung			Ident.		E
SysA	7	Die Patientendaten sollen angezeigt und bearbeitet werden können.	SWA	14	Die Patientendaten sollen angezeigt werden können.	2.2	2.2	SST	25	**P**
			SWA	15	Die Patientendaten sollen bearbeitet werden können.	2.2	2.2.1	SST	27	**P**
RA-V2	13	Monatliche Abfrage zur Aktualisierung der Patientendaten. Therapie kann nur mit aktuellen Daten durchgeführt werden	SWA	59	Wenn das Aktualisierungsdatum länger als 1 Monat zurück liegt, wird der Anwender aufgefordert die Patientendaten zu aktualisieren	2.2 3.1	2.2+ 2.2.5 3.1.2 + 2.2.5	SST	30 31 32	**P**
			SWA	60	Hinzufügen eines Attributes • Aktualisierungsdatum • Daten aktuell zu den **Patientendaten**	2.1	2.1	SST	30 31 32	**P**
			SWA	61	Wenn die Patientendaten nicht aktuell sind darf eine Therapie nicht durchführbar sein und es soll eine Fehlermeldung erscheinen.	3.1	3.1.2	SST	32 31	**P**

Der letzte Schritt der Analyse der Software-Anforderungen ist die Verifizierung der abgeleiteten Software-Anforderungen. Dabei wird das Traceability–Dokument durch ein formales Review auf die in Abbildung 3-26 aufgelisteten Kriterien geprüft.

Verifizierung der Software-Anforderungen:

Verifizierung durch: **Vorname Nachname**
Datum: **TT.MM.JJJJ**

Die Software-Anforderungen erfüllen folgende Kriterien:

Kriterium	Pass
Die Software-Anforderung implementiert die System-Anforderung einschließlich der Anforderung für die Risikobeherrschung (siehe Traceability-Dokument).	
Die Software-Anforderungen widersprechen sich nicht gegenseitig.	
Die Software-Anforderungen sind so formuliert, dass Mehrdeutigkeit vermieden wird.	
Die Software-Anforderungen sind so formuliert, dass sie die Festlegung von Prüfkriterien und die Durchführung von Prüfungen, die entscheiden, ob die Prüfkriterien erfüllt sind, ermöglichen.	
Die Software-Anforderungen sind eindeutig identifiziert (keine doppelten Nummern).	
Die Software-Anforderungen sind auf die System-Anforderung \ Risikoanalyse oder auf einen Änderungsantrag zurückzuführen (siehe Traceability-Dokument).	

Abbildung 3-26: Checkliste zur Verifizierung der Software-Anforderungen.

Anmerkung:

Für das zu entwickelnde Medizinprodukt muss ebenfalls die „Richtlinie 2006/42/EG des Europäischen Parlaments und des Rates vom 17.Mai 2006 über Maschinen und zur Änderung der Richtlinie 95/16/EG“ [51], oder kurzerhand Maschinenrichtlinie genannt, eingehalten werden. Im Folgenden sind die Anforderungen der Maschinenrichtlinie aufgelistet und werden ebenfalls als System-Anforderungen übernommen:

1. Die Maschine darf nicht unbeabsichtigt in Gang gesetzt werden können.
2. Das Stillsetzen der Maschine darf nicht verhindert werden können, wenn der Befehl zum Stillsetzen bereits erteilt wurde.
3. Automatisches oder manuelles Stillsetzen von beweglichen Teilen jeglicher Art darf nicht verhindert werden.
4. Ein Ausfall der Energieversorgung der Maschine, eine Wiederherstellung der Energieversorgung nach einem Ausfall oder eine Änderung der Energieversorgung darf nicht zu gefährlichen Situationen führen.
5. Nichttrennende Schutzeinrichtungen müssen uneingeschränkt funktionsfähig bleiben oder aber einen Befehl zum Stillsetzen auslösen.
6. Nichttrennende Schutzeinrichtungen müssen so konstruiert und in die Steuerung der Maschine integriert sein, dass beim Fehlen oder Störung eines ihrer Bestandteile das Ingangsetzen der beweglichen Teile verhindert wird oder die beweglichen Teile stillgesetzt werden.

3.5.4.3 Design der Software-Architektur

Das Design der Software Architektur kann in drei Schritte unterteilt werden. Der Ablauf ist in Abbildung 3-27 dargestellt.

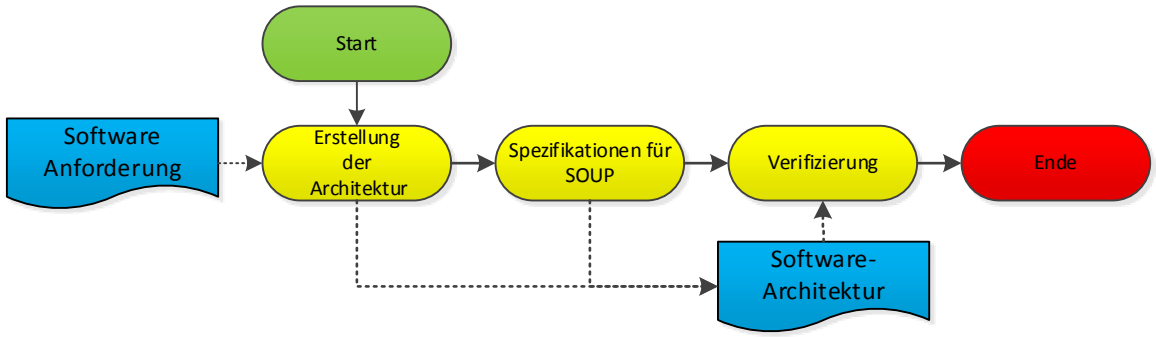


Abbildung 3-27: Ablauf zum Design der Software-Architektur.

Erstellung der Architektur

Der erste Schritt beschreibt die Umsetzung der Software-Anforderungen in die Software-Architektur. Die Software-Architektur soll einen groben Überblick über die zu erstellende Software geben und umfasst die statische und die dynamische Sicht.

Statische Sicht:

Die statische Sicht beschreibt die Struktur der Software und stellt dar, aus welchen Software-Komponenten das Software-System, in welcher Weise, zusammengesetzt ist. Ebenso werden die Schnittstellen zwischen den Software-Komponenten selbst und zu den Komponenten außerhalb des Software-Systems (Hard- und Software) beschrieben.

Aufgrund des hierarchischen Aufbaus von LabVIEW-Programmen (vgl. Kapitel 3.4), eignet sich ein einfaches Hierarchiediagramm hervorragend, um die Software-Struktur zu beschreiben. Schnittstellen zwischen den Software-Komponenten treten nur zwischen direkt übergeordneten oder untergeordneten Software-Komponenten auf. Die Beschreibung des Hierarchiediagramms ist in Abbildung 3-28 dargestellt.

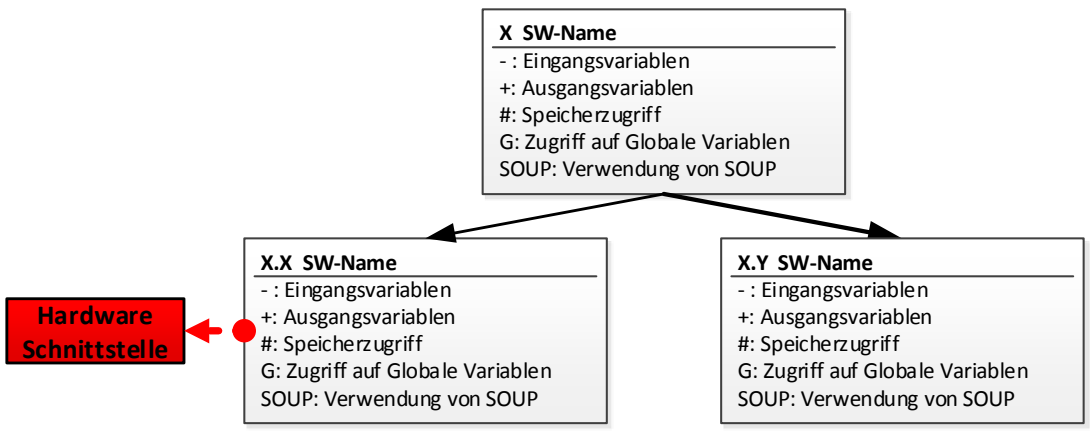


Abbildung 3-28: Template zur Darstellung der Software-Architektur.

Die Software-Komponenten **X.X** und **X.Y** sind die Teile der übergeordneten Komponente **X**. Die rote Box beschreibt die Schnittstelle zwischen einer Komponente zur Hardware. Folgende Bezeichnungen werden als Kennzeichnung der Schnittstellen verwendet:

„-“ Kennzeichnet Eingangsvariablen.

„+“ Kennzeichnet Ausgangsvariablen.

„#“ Kennzeichnet die Verwendung beziehungsweise den Zugriff von Datenspeicher.

„G“ Kennzeichnet die Verwendung beziehungsweise den Zugriff von globalen Variablen.

„SOUP“ Gibt die jeweilige verwendete SOUP an.

Dynamische Sicht:

Der zweite Teil der Software-Architektur wird durch die dynamische Sicht beschrieben. Diese stellt den Ablauf beziehungsweise die Funktionalität des Software-Systems dar. Zur graphischen Darstellung der dynamischen Sicht werden Flussdiagramme verwendet.

Zusätzlich zur graphischen Darstellung soll für jede Software-Komponente ein Dokument erzeugt werden, in welchem die folgenden Informationen enthalten sind:

- Bezeichnung
- Autor der Architektur
- Datum der Erstellung
- Software-Sicherheitsklasse
- Beschreibung
- Software-Anforderungen
- Schnittstellen
    - Eingangsvariablen
    - Ausgangsvariablen
    - Datenzugriff
    - Verwendetet globale Variablen
    - Verwendete SOUPs
    - Zugriff auf externe Hardware

Nachdem die Architektur erstellt wurde, muss wiederum das Traceability-Dokument (siehe Tabelle 11) aktualisiert werden.

Spezifikation von SOUPs

Jede verwendete SOUP muss eindeutig identifiziert und die Voraussetzungen für den bestimmungsgemäßen Gebrauch angegeben werden.

Folgende Angaben werden für jede SOUP dokumentiert:

- Beschreibung
- Bezeichnung
- Version
- Verweis zur Installationsdatei
- Herausgeber
- Benötigte Hardware
- Software-Anforderungen
- Leistungs-Voraussetzungen

Verifizierung der Software-Architektur

Die abschließende des Designs der Software-Architektur besteht in der Verifizierung der erstellten Architektur durch ein formales Review der Software-Architektur und dem Traceability-Dokument mit der folgenden Checkliste (siehe Abbildung 3-29).

Verifizierung der Software-Architektur:

Verifizierung durch: **Vorname Nachname**
Datum: **TT.MM.JJJJ**

Die Software-Architektur erfüllen folgende Kriterien:

Kriterium	Pass
Die Software-Architektur implementiert die System- und Software-Anforderung einschließlich der Anforderung für die Risikobeherrschung.	
Die Software-Architektur ist in der Lage, die Schnittstellen zwischen den einzelnen Software-Komponenten und zwischen Software-Komponenten und Hardware zu unterstützen.	
Die Architektur ist in der Lage den ordnungsgemäßen Betrieb aller SOUP-Komponenten zu unterstützen.	

Abbildung 3-29: Checkliste für die Verifizierung der Software-Architektur.

3.5.4.4 Detailliertes Software-Design

Für die vorliegende Software (Sicherheitsklasse B) fordert die EN 62304 lediglich eine strukturelle Feinaufteilung der Software-Architektur in Software-Einheiten. Eine Software-Einheit ist eine nicht mehr unterteilte Software-Komponente und stellt somit die unterste Ebene im Hierarchiediagramm dar. [41]

Eine Erstellung eines Detail-Designs der Software-Einheiten sowie der Schnittstellen wird dezidiert nur für Software mit der Sicherheitsklasse C verlangt. Um jedoch die Implementierung, Wartung und das Testen möglichst gut zu unterstützen, wurde ein detailliertes Design jeder einzelnen Einheit und Schnittstelle erstellt. Das Design soll dabei so detailliert sein, dass vom Programmierer keine Ad-hoc-Design-Entscheidungen getroffen werden müssen. [41]

Zur Darstellung des strukturellen Aufbaus des Detail-Designs wurde dasselbe Darstellungsverfahren wie zur Darstellung der Software-Architektur (siehe Kapitel 3.5.4.3) verwendet. Zur besseren Übersichtlichkeit wird für jede Software-Komponente, die in der Software-Architektur definiert wurde, ein eigenes Hierarchiediagramm erstellt. Für Software-Komponenten mit einer hohen Hierarchietiefe wird gegebenenfalls eine weitere Aufteilung durchgeführt.

Für jede Software-Komponente wird ein Detail-Design-Dokument angelegt. Folgende Informationen müssen angegeben werden:

- Autor der Dokuments
- Datum der Erstellung
- Version
- Änderungen zur vorherigen Version
- Software-Sicherheitsklasse
- Kurzbeschreibung der Software-Komponente
- Anforderung an die grafische Benutzeroberfläche
- Beschreibung der Funktionalität
- Verweis auf die umgesetzte Software-Anforderung
- Definition der Schnittstelle in Bezug auf:
    - Speicherzugriff
    - Verwendete Unterprogramme
    - Verwendete SOUPs
    - externe Schnittstellen
    - Beschreibung der Variablen
        - Eingangsvariablen
        - Ausgangsvariablen
        - Variablen für Unterprogramme
        - Zugriff auf globale Variablen

Zusätzlich wird das Detail-Design durch folgende Hilfs-Dokumente unterstützt:

- Ablaufdiagramme
    - Programmsicht
    - Benutzersicht
- Kommunikationswege
- Zustandstabellen
- Sammlungen von
    - Verwendeten Dateiformaten und deren Aufbau
    - Speicherstrukturen
    - Variablendefinitionen
    - Typ-Definitionen
    - Fehlerdefinitionen

Nachdem die Erstellung des Detail-Designs abgeschlossen ist, muss das Traceability-Dokument (siehe Tabelle 11) aktualisiert werden.

In Abbildung 3-30 ist ein exemplarisches Beispiel für das Detail-Design-Dokument der Softwarekomponente ***1.1.3 Anwender löschen-V-0*** dargestellt.

DD-1.1.3 Anwender löschen-V-0.docx

Detaildesign:DD-1.1.3 Anwender löschen-V-0.docx

Autor: Matthias Kalkgruber
Datum: 25.03.2014
Version: V-0
Änderungen zur vorherigen Version:
Erstversion
Software-Sicherheitsklasse : **B**
Beschreibung:
In diesem VI soll das Element an der Position **Liste** im **Anwender**-**Array** gelöscht werden.
Benutzerinteraktion:
Diese VI besitzt keine eigene GUI (Frontpanel), es werden lediglich Dialogfelder angezeigt.

Dialogfeld:

- Dialogabfrage: „Wollen Sie den Anwender: **Anwendername** wirklich löschen?“,
  - Schaltfläche: „Ja, löschen!“
  - Schaltfläche: “Abbrechen!“

Funktionalität:

1. Beim Aufruf dieses VI soll das Dialogfeld erscheinen.
2. Bei: Schaltfläche: „Ja, löschen!“, soll das Element an der Position **Liste** aus dem **Anwender**-**Array** gelöscht werden, das **Anwender**-**Array** aktualisiert und anschließend das **Anwender**-**Array** mittels „1.0.2 Anwender Daten schreiben“ gespeichert werden.
3. Bei Schaltfläche: “Abbrechen!“, soll keine Änderung von **Anwender**-**Array** erfolgen.

Umgesetzte Anforderungen:

SWA	5	Anwender sollen gelöscht werden können
SWA	42	Beim Löschvorgang des Anwenders, wird der Anwendername des zu löschenden Anwender nochmals dargestellt und erst bei erneuter Bestätigung gelöscht.

Schnittstellen:

- Speicherzugriff: kein Speicherzugriff
- SubVI:
  - 1.0.2 Anwender Daten schreiben-V-0
- SOUP: keine Verwendung von SOUP
- Extern: keine externe Schnittstellen
- Variablen:

Name	Schnittstellen-Typ	Daten-Typ	Bemerkung
Error	Eingang / Ausgang / Intern	Error-Cluster	Cluster mit Fehlerinformationen
Anwender-Pfad	Eingang / Intern	Datei-Pfad	Pfad zur Anwender-Datei, „AnwenderDaten.xml“
Anwender-Array	Eingang / Ausgang / Intern	Array: TypDef: Anwender	Enthält alle Anwender
Liste	Eingang	Long	Position der Auswahl im Listenfeld

Seite **1** von **1**

Abbildung 3-30: Exemplarisches Detail-Design-Dokument am Beispiel von 1.1.3 Anwender löschen-V-0.docx.

3.5.4.5 Implementieren–Verifizieren-Integrieren

Der Ablauf der Implementieren-Verifizieren-Integrieren-Aktivität ist in Abbildung 3-31 dargestellt.

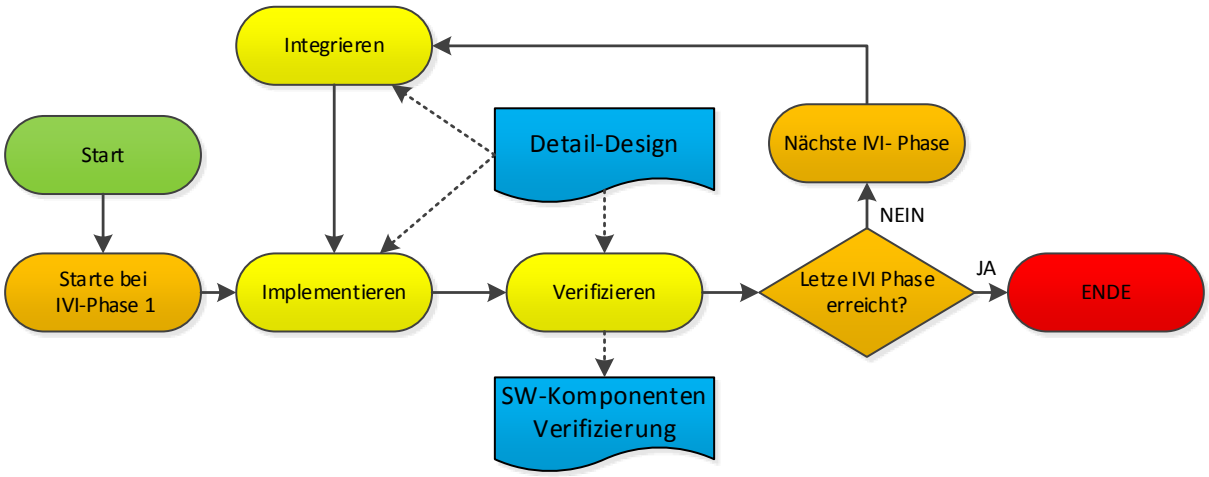


Abbildung 3-31: Ablauf der Aktivität: Implementieren-Verifizieren-Integrieren.

Aufgrund der hierarchischen Struktureigenschaften von LabVIEW-Programmen wurden die Aktivitäten der EN 62304 ***Implementierung und Verifizierung der Software-Einheiten*** sowie ***Software-Integration und Integrationsprüfung*** zu einer iterativen Aktivität ***Implementieren-Verifizieren-Integrieren*** zusammengefasst.

Die Implementieren-Verifizieren-Integrieren-Aktivität (IVI) erfolgt anhand der strukturorientierten Bottom-Up-Integrationsstrategie. Eine Darstellung der Bottom-Up-Integration ist in Abbildung 3-32 illustriert. Die Reihenfolge in der die einzelnen Komponenten implementiert, verifiziert und integriert werden, richtet sich vorwiegend nach der strukturellen Abhängigkeit der einzelnen Komponenten. [52]

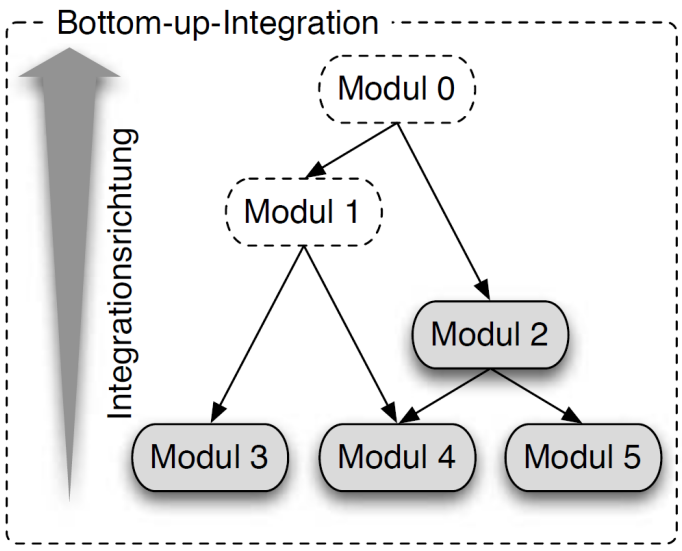


Abbildung 3-32: Bottom-Up-Integrationsstrategie. Entnommen aus [52].

Die erste IVI-Phase beginnt mit der Implementierung der Software-Komponenten, die sich in der untersten Schicht der Software-Struktur befinden. Erst nach der vollständigen Umsetzung aller Software-Komponenten in dieser Schicht werden die Komponenten getestet.
Mit der Implementierung der nächsthöheren Schicht und somit der Integration der bereits verifizierten Sub-Software-Komponenten darf erst begonnen werden, wenn alle Komponenten der vorherigen Phase erfolgreich verifiziert wurden. Das bedeutet, alle verwendeten Software-Komponenten die in einer übergeordneten Komponente zur Implementierung verwendet werden, müssen bereits erfolgreich getestet sein. Die einzelnen Komponenten werden somit phasenweise implementiert, bis die oberste Software-Komponente erfolgreich implementiert und verifiziert wurde.

Zur Abarbeitung der einzelnen IVI-Phasen wurde zuerst ein Plan erstellt, der die einzelnen Software-Komponenten in IVI-Phasen einteilt. Wie bereits erwähnt, basiert dieser Plan auf den strukturellen Abhängigkeiten der Software-Komponenten. Zusätzlich werden aber auch die funktionellen Abhängigkeiten in Hinblick auf die Testphase berücksichtigt. Es kann zum Beispiel vorkommen, dass zum Testen einer Software-Komponente andere Software-Komponenten benötigt werden, welche keine strukturelle Abhängigkeit aufweisen. Zum Bespiel können keine Bewegungsbefehle an die Endstufen gesendet werden, bevor nicht die Kommunikation initialisiert wurde. Aus diesem Grund sollten zuerst die Software-Komponenten für die Initialisierung der Kommunikation implementiert werden und erst danach die Software-Komponenten für die Erstellung der Bewegung.

Verifizieren

Die Verifizierung der einzelnen Software-Komponenten wird anhand von statischen und dynamischen Tests durchgeführt. Bei statischen Tests wird mittels Codereview überprüft, ob alle Akzeptanzkriterien erfüllt wurden. Diese umfassen einerseits Codierungsvorschriften und andererseits die Kontrolle der korrekten Umsetzung der Schnittstellen. Zusätzlich werden die Anforderungen an die grafische Benutzerschnittstelle laut Detail-Design kontrolliert. Die einzelnen Akzeptanzkriterien sind in Abbildung 3-33 unter Code-Review dargestellt.

Der zweite Teil der Verifizierung wird durch dynamische Tests abgedeckt, indem überprüft wird, ob die Software-Komponenten die geforderten Funktionalitäten, wie im Detail-Design gefordert, aufweisen. Im Zuge des dynamischen Tests wird die Software-Komponente ausgeführt und definierte Abläufe beziehungsweise Eingabedaten vorgegeben. Weiters werden die erwarteten Ausgaben beziehungsweise Ereignisse des Programms vorgegeben und der Tester muss überprüfen, ob diese Ereignisse eintreten.
Ein Beispiel für ein Test-Protokoll ist in Abbildung 3-33 und Abbildung 3-34 dargestellt.

Testprotokoll: TP-1.1.3 Anwender löschen-V-0-1.docx

Testprotokoll: TP-1.1.3 Anwender löschen-V-0-1.docx

Test Autor: **Matthias Kalkgruber** Datum: 19.04.2014
Freigabe: **Matthias Kalkgruber** Datum: 26.04.2014
Software-Tester: **Matthias Kalkgruber** Datum: 28.04.2014
Testergebnis: **PASS**
Detail-Design: **DD-1.1.3 Anwender löschen-V-0**
Struktur: **Detailstrukturplan 1.1 Anwender bearbeiten-V-0**

Code -Review:

Nr.	Beschreibung	Ja/Nein	Anmerkung:
CR-1.1.3-1	Der Informationsfluss erfolgt von links nach rechts.	**JA**	
CR-1.1.3-2	Alle Funktionsblöcke sind sichtbar.	**JA**	
CR-1.1.3-3	Die Schnittstellen der Software entsprechen den Vorgaben im Detail-Design (Name/Daten-Typ).	**JA**	
CR-1.1.3-4	Am linken oberen Rand des Blockdiagramms befindet sich eine kurze Beschreibung der Software als Kommentar.	**JA**	
CR-1.1.3-5	In der Beschreibung ist das Datum der Implementierung angegeben	**JA**	
CR-1.1.3-6	In der Beschreibung ist der Autor der Implementierung angegeben	**JA**	
CR-1.1.3-7	Am linken oberen Rand des Blockdiagramms befindet sich eine Beschreibung der Teilaufgaben dieser Software.	**JA**	
CR-1.1.3-8	Jedem Funktionsblock ist mittels Kommentar einer Teilaufgabe zugeordnet.	**JA**	
CR-1.1.3-9	Die im Detail-Design geforderten Elemente sind im Frontpanel vorhanden und eindeutig identifizierbar.	**JA**	
CR-1.1.3-10	Die Subsoftwarekomponenten/-einheiten wurden laut Detail-Design bzw. Software-Architektur eingefügt.	**JA**	

Funktionstest:

Die Funktion wird überprüft indem ein Anwender aus der Anwenderdatei gelöscht wird. Zusätzlich werden das Verhalten der Buttons und Anzeige Elemente überprüft.

Testumgebung:
- TC-1.1.3-1
- "~\Daten SW-Test\1.1.3 Anwender ändern\Daten\AnwenderDaten.xml"

Ablauf:
1. Laden Sie das VI „TC-1.1.3-1"
2. Geben Sie im Frontpanel unter **Anwender-Pfad** folgenden Pfad an:
   - "~\Daten SW-Test\1.1.3 Anwender ändern\"
3. Geben Sie in **Liste** den Wert 1 ein
4. Starten Sie das VI

Nr.	Erwartetes Ereignis	Ja/Nein	Anmerkung
FT-1.1.3-1	Es erscheint folgender Dialog: "Wollen Sie den Anwender: „Testperson2" wirklich löschen?"	**JA**	

Seite **1** von **2**

Abbildung 3-33: Seite 1 des exemplarischen Testprotokolls für die Software-Komponente 1.1.3 Anwender löschen V-0.

Testprotokoll: TP-1.1.3 Anwender löschen-V-0-1.docx

FT-1.1.3-2	Folgende Buttons stehen im Dialog zur Verfügung: ☒ Ja, löschen! ☒ Abbrechen!	**JA**	

5. Bestätigen Sie den Löschvorgang mit dem Button **Ja, löschen!**

FT-1.1.3-3	Im Frontpanel von TC-1.1.3-1 beinhaltet das Anzeigeelement **Anwender-Array Ausgabe** und **Anwender-Array Kontrolle** folgendes: o **Anwendername**: „Testperson1“ o **Passwort**: „Test“ o **Adminrechte**: *False* o **Anwendername**: „Testperson3“ o **Passwort**: „Test“ o **Adminrechte**: *False* o **Anwendername**: „Testperson4“ o **Passwort**: „Test“ o **Adminrechte**: *False*	**JA**	

6. Starten Sie das VI nochmals

FT-1.1.3-4	Es erscheint folgender Dialog: “Wollen Sie den Anwender: „Testperson2“ wirklich löschen?“	**JA**	

7. Drücken Sie den Button **Abbrechen**

FT-1.1.3-5	Im Frontpanel von TC-1.1.3-1 beinhaltet das Anzeigeelement **Anwender-Array Ausgabe** und **Anwender-Array Kontrolle** folgendes: o **Anwendername**: „Testperson1“ o **Passwort**: „Test“ o **Adminrechte**: *False* o **Anwendername**: „Testperson3“ o **Passwort**: „Test“ o **Adminrechte**: *False* o **Anwendername**: „Testperson4“ o **Passwort**: „Test“ o **Adminrechte**: *False*	**JA**	

Sonstige Anmerkungen:

Seite **2** von **2**

Abbildung 3-34: Seite 2 des exemplarischen Testprotokolls für die Software-Komponente 1.1.3 Anwender löschen V-0.

3.5.4.6 Prüfung des Software-Systems

Nachdem das komplette Software-System implementiert wurde, muss es durch eine Software-System-Prüfung verifiziert werden. Dabei wird mittels Funktionstest überprüft, ob das kompilierte Exekutiv die Software-Anforderungen erfüllt. Der Ablauf ist dem Funktionstest in der Aktivität Implementieren-Verifizieren-Integrieren sehr ähnlich. Der Unterschied besteht darin, dass bei der Aktivität Implementieren-Verifizieren-Integrieren überprüft wird, ob das Detail-Design umgesetzt wurde.
Bei der Software-System-Prüfung wird direkt die Umsetzung und Funktionalität der Software-Anforderungen überprüft.
Nach der Software-System-Verifizierung muss das Traceability-Dokument (siehe Tabelle 11) wieder aktualisiert werden.

Die Begriffe Validieren und Verifizieren werden sehr oft verwechselt oder falsch verwendet. Aus diesem Grund möchte der Autor diese zwei Begriffe nochmals erläutern.
Die EN 62304 definiert Verifizierung folgendermaßen:

> *„Bestätigen durch Bereitstellen eines objektiven Nachweises, dass festgelegte Anforderungen erfüllt worden sind.“*
> Zitiert nach [41].

Im Vergleich definiert die EN ISO 13485 Validierung folgendermaßen:

> *„Bestätigung durch Bereitstellen eines objektiven Nachweises, dass die Anforderungen an eine spezielle Zweckbestimmung oder Anwendung erfüllt worden sind.“*
> Zitiert nach [42].

In Bezug auf das Lokomotionsgerät versteht man unter Verifizieren den objektiven Nachweis, dass die Software-Anforderungen korrekt umgesetzt wurden, beziehungsweise dass die Software ordnungsgemäß funktioniert.

Unter Validieren versteht man den objektiven Nachweis, dass das Lokomotionsgerät so entwickelt wurde, dass die Zweckbestimmung erfüllt ist. Man stellt sich also die Frage ob mit dem entwickeltem Gerät eine:

> *„[…]Förderung der Bewegungsentwicklung von Kleinkindern mit infantiler Cerebralparese[…]“*
> Zitiert nach [45].

erzielt wird. Diese Forderung kann jedoch nicht durch reine Software-Tests überprüft werden. Hierbei wird auf die klinische Bewertung und auf die Prüfung der Gebrauchstauglichkeit verwiesen.

3.5.4.7 Freigabe der Software

Bevor die Software freigegeben werden darf, müssen noch formale Aktivitäten durchgeführt werden. Zuerst muss sichergestellt werden, dass die Verifizierung der Software vollständig abgeschlossen ist und die vorhandenen Anomalien dokumentiert und bewertet wurden. Die Software darf nur freigegeben werden, wenn sichergestellt ist, dass die vorhandenen Anomalien nicht zu einem unvertretbaren Risiko beitragen. [41]

Die freigegebene Software, inklusive ihrer zugehörigen Konfigurationselemente und Dokumentation muss archiviert werden [41].

3.5.5 Problemlösungs-Prozess für Software

Der Problemlösungs-Prozess für Software muss für alle Probleme angewandt werden, die in einem Software-Produkt entdeckt werden. Probleme können in allen Phasen der Software-Entwicklung und auch nach Freigabe der Software entdeckt werden. Im Folgenden wird der Problemlösungs-Prozess für Software erklärt und ist in Abbildung 3-35 dargestellt.

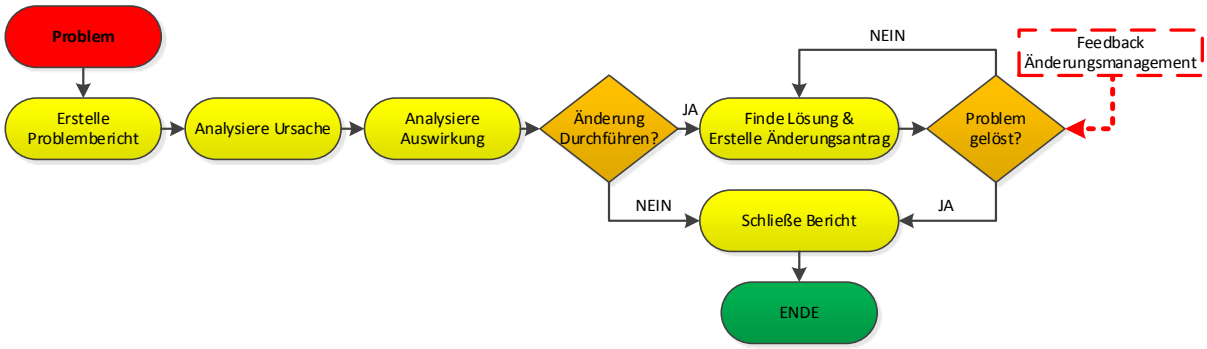


Abbildung 3-35: Problemlösungs-Prozess für Software.

Der erste Schritt zur Lösung ist die Erstellung eines Problemberichts. Dieser soll das Problem so ausführlich wie möglich beschreiben, um ausreichend Informationen für die weiteren Schritte zur Verfügung zu stellen.

Anhand dieser Informationen wird die Ursache des Problems analysiert und dokumentiert. Weiters müssen die Auswirkungen auf die Sicherheit, unter Verwendung des Risiko-Management-Prozesses, geprüft werden. Je nach Schweregrad der Auswirkungen soll jedem Problem eine Kritikalitäts-Stufe zugeordnet werden.

Folgende Kritikalitäts-Stufen wurden definiert:

- **Stufe 5:** Schwerer Fehler; Muss im nächsten Release gelöst werden.
- **Stufe 4:** Mittlerer Fehler; Soll im nächsten Release gelöst werden.
- **Stufe 3:** Leichter Fehler; Lösung des Fehlers wenn nötige Ressourcen vorhanden sind.
- **Stufe 2:** Änderungswunsch; Kein Fehler sondern nur Änderungswunsch.
- **Stufe 1:** Idee; Kein Fehler sondern Verbesserungsvorschlag für das Produkt.

Nachdem Ursache und Auswirkung erfasst wurden, muss entschieden werden, ob eine Änderung durchgeführt werden soll. Falls eine Änderung nicht genehmigt wird, bleibt das Problem ungelöst und der Problembericht wird geschlossen.

Falls eine Änderung genehmigt wird, muss eine Lösung gefunden und ein Änderungsantrag verfasst werden. Für die Handhabung von Änderungen wird auf das Kapitel 3.5.6 verwiesen.

Nachdem die Änderung durchgeführt wurde, wird überprüft, ob durch die Änderung das Problem gelöst wurde. Falls dies nicht der Fall ist, muss ein neuer Lösungsansatz gefunden werden. Bei einer erfolgreichen Lösung wird der Problembericht geschlossen. Ein Beispiel für einen exemplarischen Problembericht ist in Abbildung 3-36 dargestellt.

Problembericht : PB_2.docx

Problembericht:

Aktueller Status: **Bericht geschlossen**
(Bericht erstellt / Ursache identifiziert / Auswirkung analysiert/ Änderungsantrag gestellt /Bericht geschlossen

Identifikation: **PB-2**
Autor: **Sonja Langthaler** Datum: **10.6.2015**
Beschreibung:

Beim Erstellen (2.1 Patient hinzufügen) und Bearbeiten (2.2.1 Patienten Daten ändern) von Patientendatensätzen können die Orthesenlängen nicht eingeben oder bearbeitet werden.

Ursache des Problems:
Autor: **Matthias Kalkgruber** Datum: **10.6.2015**

Funktionalität wird nicht unterstützt.

Auswirkung auf die Sicherheit:
Autor: **Matthias Kalkgruber** Datum: **10.6.2015**
Kritikalität(niedrig 1 – 5 hoch) : **2**

Keine Auswirkung auf die Sicherheit. Die Orthesenlängen können bei der ersten Therapie im Therapiemodus vorgegeben werden.

Falls Auswirkung auf Sicherheit. Welche Risikokontrollmaßnahmen wurden abgeleitet:

- ☒ **keine**

Soll eine Änderung durchgeführt werden: **JA**
Autor: **Matthias Kalkgruber** Datum: **12.06.2015**

a) Falls keine Änderung durchgeführt wird. Beschreibung der Begründung:

b) Falls Änderung durchgeführt werden soll. Verweis auf Änderungsantrag:
   - ☒ **AEA-2**

Bericht geschlossen am**: 15.06.2015**
Autor: **Matthias Kalkgruber**
Problem gelöst : **JA**

Seite **1** von **1**

Abbildung 3-36: Exemplarischer Problembericht.

3.5.6 Software-Konfigurationsmanagement

Das Konfigurationsmanagement hat die Aufgabe Konfigurationselemente (Software-Komponenten und ihre Dokumentation) zu identifizieren, sowie Änderungen und Freigaben dieser zu kontrollieren. Anders ausgedrückt, das Software-Konfigurationsmanagement ist dafür verantwortlich, die Zusammensetzung des Produktes sowie Änderungen klar ersichtlich zu machen. Der Verlauf der Entwicklung soll nachvollzogen werden können, um gegebenenfalls eine vergangene Konfiguration wiederherstellen zu können. Das Konfigurationsmanagement kann in „Identifizierung der Konfiguration“ und „Änderungskontrolle“ unterteilt werden.

Identifizierung der Konfiguration

Bei der Identifizierung wird zuerst festgelegt, welche Dokumente unter das Konfigurationsmanagement gestellt werden. Hierzu gehören gewissermaßen alle Dokumente und Software-Komponenten, die während der Entwicklung erstellt werden. Dazu zählen unter anderem:

- Source-Code
- Test-Code
- Dokumente der System-Anforderungen
- Dokumente der Software-Anforderungen
- Dokumente der Software-Architektur
- Dokumente des Detail-Designs
- Dokumente der Software-Tests
- Dokumente der Software-System-Verifizierung
- Dokumente des Risikomanagements

Für jedes Konfigurationselement muss ein eindeutiges Schema zur Identifizierung, inklusive ihrer Version, vorgegeben werden. Um den Umfang dieser Arbeit in einem überschaubaren Rahmen zu halten, wird der Autor nicht die vollständige Benennung der einzelnen Konfigurationselemente präsentieren, sondern nur ein exemplarisches Beispiel anhand des Testprotokolls erklären. Der interessierte Leser wird auf das Dokument **Konfigurationsmanagement.docx** [53] verwiesen.

Testprotokolle werden folgendermaßen benannt:

- TP-Softwarekomponente-Version-Testnummer
- TP-1.1.2 Anwender ändern-V-0.1-1

„TP-1.1.2 Anwender ändern-V-0.1-1“ steht für das Testprotokoll (**TP**) der Softwarekomponente **1.1.2 Anwender ändern** mit der Version **V-0.1.** Der letzte Eintrag steht für die erste Testdurchführung. Falls derselbe Test noch einmal durchgeführt wird, wird die Testnummer inkrementiert.

Änderungskontrolle

Der zweite Teil des Software-Konfigurationsmanagements stellt die Änderungskontrolle dar. Dessen Ziel ist, dass Änderungen an der Konfiguration der Software kontrolliert durchgeführt werden. Es ist zu gewährleisten, dass keine unbeabsichtigten oder unautorisierten Änderungen durchgeführt werden und dass Änderungsanträge vollständig umgesetzt und verifiziert werden. Nur Änderungen auf Basis eines Änderungsantrages dürfen durchgeführt werden. In Abbildung 3-37 ist der Ablauf der Änderungskontrolle dargestellt.

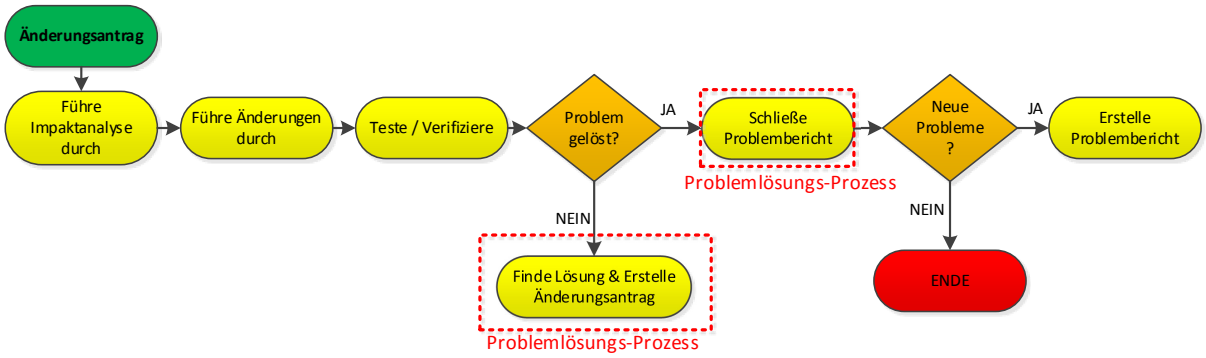


Abbildung 3-37: Ablauf der Änderungskontrolle.

Der erste Schritt ist die Erstellung eines Änderungsantrages durch Angabe folgender Informationen:

- Eindeutige Identifizierung des Änderungsantrages
- Verweis auf den Problembericht
- Beschreibung der Änderung
- Kritikalität des Problems
- Autor und Datum

Als nächster Schritt wird die Impaktanalyse durchgeführt, um zu erkennen, welche Konfigurationselemente, in welcher Weise, geändert werden müssen. Dies umfasst nicht nur die Software selbst, sondern auch Dokumente der konstruktiven Phase, zum Beispiel eine Änderung des Detail-Designs. Zusätzlich muss erfasst werden, welche Komponenten aufgrund der Änderungen erneut getestet werden müssen und ob der Softwaretest (Testprotokoll), gegebenenfalls angepasst werden muss.

Selbstverständlich müssen alle Änderungen vollständig umgesetzt und anschließend getestet werden, ob das Problem gelöst wurde. Wenn das Problem nach den Änderungen noch immer vorhanden ist, muss ein neuer Lösungsansatz gefunden werden (siehe Problemlösungs-Prozess). Bei erfolgreicher Elimination des Problems wird der zugehörige Problembericht geschlossen.

Falls durch die Änderungen neue Probleme verursacht oder erkannt wurden, muss ein entsprechender Problembericht erstellt werden. In Abbildung 3-38 und Abbildung 3-39 ist ein exemplarischer Änderungsantrag dargestellt.

Änderungsantrag : AEA-2.docx

Änderungsantrag:

Autor: **Matthias Kalkgruber** Datum: **12.06.2015**
Verweis auf Problembericht: **PB-2**
Identifizierung: **AEA-2**
Kritikalität: **2**
Kurze Beschreibung der Änderung?

Beim Erstellen (2.1 Patient hinzufügen) und Bearbeiten (2.2.1 Patienten Daten ändern) von Patientendaten sollen die Orthesenlängen eingeben und bearbeitet werden können.

Impaktanalyse:

Autor: **Matthias Kalkgruber** Datum: **12.06.2015**

Software-Anforderung:

Müssen **Software-Anforderungen** geändert werden? **NEIN**

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
----------	Keine Änderung	----------	----------	----------

Software- Architektur:

Muss die **Software-Architektur** geändert werden? **NEIN**

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
----------	Keine Änderung	----------	----------	----------

Software-Detail-Designs:

Muss das **Detail-Design** geändert werden? **JA**

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
2.2.1 Patienten Daten ändern	Im Frontpanel sollen die Orthesenlängen angezeigt und geändert werden können.	0.1	12.06.2015	Matthias Kalkgruber
2.2 Patientenakte anzeigen_bearbeiten	Geänderte SubVIs : 2.2.1 Patienten Daten ändern	0.1	12.06.2015	Matthias Kalkgruber
2.1 Patient hinzufügen	Im Frontpanel sollen die Orthesenlängen angezeigt und geändert werden können.	0.1	12.06.2015	Matthias Kalkgruber
2. Patientenverwaltung	Geänderte SubVIs : 2.1 Patient hinzufügen 2.2 Patientenakte anzeigen_bearbeiten	0.1	12.06.2015	Matthias Kalkgruber
0. Lokomotionstherapie	Geändertes SubVi: 2.Patienteverwaltung	0.1	12.06.2015	Matthias Kalkgruber

Software-Implementierung:

Welche **Software** muss direkt geändert werden?

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
2.2.1 Patienten Daten ändern	Im Frontpanel sollen die Orthesenlängen angezeigt und geändert werden können.	0.1	12.06.2015	Matthias Kalkgruber
2.1 Patient hinzufügen	Im Frontpanel sollen die Orthesenlängen angezeigt und geändert werden können.	0.1	12.06.2015	Matthias Kalkgruber

Seite **1** von **2**

Abbildung 3-38: Exemplarischer Änderungsantrag; Seite 1 von 2.

Änderungsantrag : AEA-2.docx

Welche **Software** muss indirekt geändert werden? (Änderung der Konfiguration /SubVIs)

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
2. Patientenverwaltung	Geänderte SubVIs : ☒ 2.1 Patient hinzufügen-V-0.1 ☒ 2.2 Patient anzeigen_bearbeiten-V-0.1	0.1	12.06.2015	Matthias Kalkgruber
2.2 Patient anzeigen_bearbeiten	Geänderte SubVIs : ☒ 2.2.1 Patienten Daten ändern -V-0.1	0.1	12.06.2015	Matthias Kalkgruber
0. Lokomotionstherapie	Geändertes SubVI: ☒ 2.Patienteverwaltung-V-0.1	0.1	12.06.2015	Matthias Kalkgruber

Software –Test:

Welche **Testprotokolle** müssen **aktualisiert** werden?

Element	aktuelle Version	Datum d. Änderung	Autor
TP - 2. Patientenverwaltung-V-0	V-0.1	13.06.2015	Matthias Kalkgruber
TP - 2.2 Patient anzeigen_bearbeiten-V-0	V-0.1	13.06.2015	Matthias Kalkgruber
TP - 2.2.1 Patienten Daten ändern -V-0	V-0.1	13.06.2015	Matthias Kalkgruber
TP - 2.1 Patient hinzufügen -V -0	V-0.1	13.06.2015	Matthias Kalkgruber
TP - 0. Lokomotionstherapie -V -0	V-0.1	13.06.2015	Matthias Kalkgruber

Welche **Testcases** müssen **geändert** / **erstellt** werden?

Element	Beschreibung:	aktuelle Version	Datum d. Änderung	Autor
----------	Keine Änderung	----------	----------	----------

Welche **Tests** müssen **durchgeführt / wiederholt** werden? Wann wurde der Test **durchgeführt**?

Testprotokoll	Ergebnis	Datum d. Test	Tester
TP - 2. Patientenverwaltung-V-0.1-1	PASS	14.6.2015	Matthias Kalkgruber
TP - 2.2 Patient anzeigen_bearbeiten-V-0.1-1	PASS	14.6.2015	Matthias Kalkgruber
TP - 2.2.1 Patienten Daten ändern -V-0.1-1	PASS	14.6.2015	Matthias Kalkgruber
TP - 2.1 Patient hinzufügen -V -0.1-1	PASS	14.6.2015	Matthias Kalkgruber
TP - 0. Lokomotionstherapie -V -0.1-1	PASS	14.6.2015	Matthias Kalkgruber

Ergebnis:

Name: **Matthias Kalkgruber** Datum: 15.06.2015

Wurde das Problem gelöst? **JA**

Problembericht geschlossen? **JA**

Wurden zusätzliche Probleme verursacht? **NEIN**

☒ Verweis auf Problembericht: **keine**

Seite **2** von **2**

Abbildung 3-39: Exemplarischer Änderungsantrag; Seite 2 von 2.

4 Ergebnisse

4.1 Software-Sicherheitsklassifizierung

Die vollständige Risiko-Analyse zur Software-Sicherheitsklassifizierung ist in der schriftlichen Arbeit nicht enthalten und es werden im Folgenden nur die Ergebnisse präsentiert. Für die vollständige Risikoanalyse wird auf das Dokument „RA-V1.docx“ [54] im digitalen Anhang verwiesen.

In Tabelle 12 ist die Zusammenfassung der Risikoanalyse in Form einer Risikomatrix dargestellt. Es wurden 15 Risiken mit Bezug zur Software identifiziert. Davon wurden 7 Risiken in die Risikostufe **IV** eingestuft. Diese sind vernachlässigbar und werden akzeptiert. Die übrigen 8 Risiken fallen in die Risikostufe **III** (ALARP: nur tolerierbar wenn eine Vermeidung nicht praktikabel ist). Der höchste Schaden für ein nicht akzeptierbares Risiko entspricht somit einer mittleren Schadensfolge beziehungsweise einer leichten Verletzung (siehe Definition der Schadensfolge in Tabelle 4).
Auf Basis der höchsten potentiellen Schadensfolge der inakzeptablen Risiken wurde das Software-System in die **Software-Sicherheits-Klasse: B** eingestuft.

Tabelle 12: Risikomatrix für die Software-Sicherheitsklassifizierung.

Eintrittswahrscheinlichkeit							Risikostufe
	6	häufig	0	0	0	0	
	5	manchmal	0	0	0	0	I
	4	gelegentlich	0	3	0	0	
	3	selten	1	4	0	0	II
	2	unwahrscheinlich	0	6	0	0	III
	1	unglaublich	0	0	1	0	IV
			gering	mittel	schwer	katastrophal	
			1	2	3	4	
			Schadensfolge				

4.2 Software-Risiko-Management

Die Ergebnisse der Risiko-Analyse, Risiko-Bewertung sowie Risiko-Beherrschung sind in Form eines Risikoprotokolls im **Anhang E** angefügt. Die Zusammenfassung der bewerteten Gefährdungen, vor der Durchführung der Abhilfemaßnahmen, ist in Tabelle 13 als Risikomatrix dargestellt. Im Vergleich ist das Ergebnis der neu bewerteten Gefährdungen, nach der Durchführung der Risiko-Kontrollmaßnahmen, in Tabelle 14 ersichtlich.

Tabelle 13: Gesamtrisikomatrix vor der Durchführung von Abhilfemaßnahmen.

Eintrittswahrscheinlichkeit							Risikostufe
	6	häufig	0	0	0	0	
	5	manchmal	1	6	0	0	I
	4	gelegentlich	2	19	2	0	
	3	selten	12	22	9	0	II
	2	unwahrscheinlich	0	1	3	0	III
	1	unglaublich	0	0	0	0	IV
			gering	mittel	schwer	katastrophal	
			1	2	3	4	
			Schadensfolge				

Tabelle 14: Gesamtrisikomatrix nach der Durchführung von Abhilfemaßnahmen.

Eintrittswahrscheinlichkeit							Risikostufe
	6	häufig	0	0	0	0	
	5	manchmal	0	0	0	0	I
	4	gelegentlich	0	0	0	0	
	3	selten	0	1	0	0	II
	2	unwahrscheinlich	6	28	2	0	III
	1	unglaublich	9	26	5	0	IV
			gering	mittel	schwer	katastrophal	
			1	2	3	4	
			Schadensfolge				

Es wurden 77 Risiken identifiziert, wobei vor der Durchführung der Abhilfemaßnahmen 8 Risiken in die Risikostufe **II**, 68 in die Risikostufe **III** und nur ein Risiko in die Risikostufe **IV** eingeteilt wurden. Mit Hilfe der 34 abgeleiteten Risiko-Kontrollmaßnahmen wurde der Großteil der Risiken auf die Risikostufe **IV** reduziert. Von den 77 Risiken befinden sich nur mehr drei Risiken in der Risikostufe **III**.
Die in Risikostufe **III** verbliebenen Risiken wurden analysiert und als akzeptierbare Risiken eingestuft. Eine Risikominimierung kann nur noch mit einem zum Nutzen unproportionalem Aufwand (ALARP-Bereich) erreicht werden.
Die starke Minimierung der Risiken rührt daher, dass für die Gefährdungen mit hohem Risiko, mehrere Risiko-Kontrollmaßnahmen kombiniert wurden, um diese in einen akzeptierbaren Risikobereich zu verschieben.

Gesamtbewertung

Durch Berücksichtigung von eventuellen Abhängigkeiten zwischen den einzelnen Risiken, sowie der Möglichkeit eines zeitgleichen Eintritts mehrerer Einzelrisiken, konnten keine Risiken identifiziert werden, die zu einem nicht akzeptierbaren Gesamtrisiko führen könnten. [55]

Risiko-Nutzen-Bewertung

Zur Abschätzung des Risiko-Nutzen Verhältnis wurde ein Vergleich mit den bisherigen Behandlungsmöglichkeiten durchgeführt. Diese können einerseits in eine risikoarme Therapieform mittels konventioneller Physiotherapie und andererseits in eine risikoreichere Behandlung durch Denervierung und orthopädische Chirurgie unterteilt werden. [8]

Im Vergleich zur konventionellen Physiotherapie wird ein höherer medizinischer Nutzen durch die Therapie mit dem entwickelten Lokomotionsgerätes für Kleinkinder erwartet. Die Begründung basiert auf den erzielten Ergebnissen [56] vom Lokomat® (Hocoma AG, Schweiz). Mit Hilfe des entwickelten Therapiegeräts wird ebenfalls eine erhebliche Verbesserung der Lebensqualität des Patienten erwartet. Im Vergleich zur chirurgischen Therapie tritt bei der Lokomotionstherapie ein deutlich geringeres Risiko auf. [1]

Aufgrund des zu erwartenden medizinischen Nutzens, sowie dem Vergleich mit bestehenden Therapieformen, wird das vorhandene Gesamt-Risiko des entwickelten Gerätes zur Lokomotionstherapie als akzeptabel eingestuft.

4.3 Software-Anforderungen

Als Ergebnis der Analyse der Software-Anforderungen wurden in Summe 86 Software-Anforderungen abgeleitet. Davon beziehen sich 39 Anforderungen auf System-Anforderungen, 46 auf die Risiko-Kontrollmaßnahmen und eine weitere auf die Umsetzung der Maschinendirektive. Es wird angemerkt, dass einige Anforderungen der Maschinendirektive bereits durch vorhandene Risiko-Kontrollmaßnahmen abgedeckt wurden.
Die Software-Anforderungen können den folgenden Software-Hauptkomponenten zugeordnet werden:

- **Anwenderverwaltung**: 17 Software-Anforderungen
- **Patientenverwaltung**: 23 Software-Anforderungen
- **Therapiemodus**: 45 Software-Anforderungen
- **Fehlerbehandlung**: 1 Software-Anforderung

Eine vollständiger Liste der Software-Anforderungen ist im digitalen Anhang unter „Software-Anforderungen-V-0.2.docx“ [57] beziehungsweise „Traceability-V-0.2.docx“ [50] zu finden.

4.4 Software–Architektur

In diesem Kapitel wird der grobe Aufbau des Software-Systems beschrieben. Zuerst werden die Struktur der Software und die Funktionen der einzelnen Komponenten skizzenhaft erläutert. Anschließend wird der Ablauf der Lokomotionstherapie anhand eines Ablaufdiagrammes beschrieben.

4.4.1 Struktur

Die Struktur des Software-Systems ist in Abbildung 4-1 dargestellt. Nachstehend werden die einzelnen Funktionen der Software-Komponenten im Überblick erläutert. Für eine detailliertere Beschreibung wird auf die entsprechende Dokumentation der Software-Architektur **„Software-Architektur-Hauptdokument-V-0.docx“** [58] und des Detail-Designs **„Detail-Design-Hauptdokument-V-0.2.docx“** [59] im digitalen Anhang verwiesen. Einen guten Einblick in die Funktionalität der Software gibt die „**Bedienungsanleitung-V-0.2.docx“** [60] im digitalen Anhang**.**

In der Software-Struktur wurde auf die Darstellung der Schnittstellen hinsichtlich einer besseren Übersichtlichkeit verzichtet. Für interessierte Leser ist das ausführliche Hierarchiediagramm in Abbildung 8-6 im Anhang zu finden.

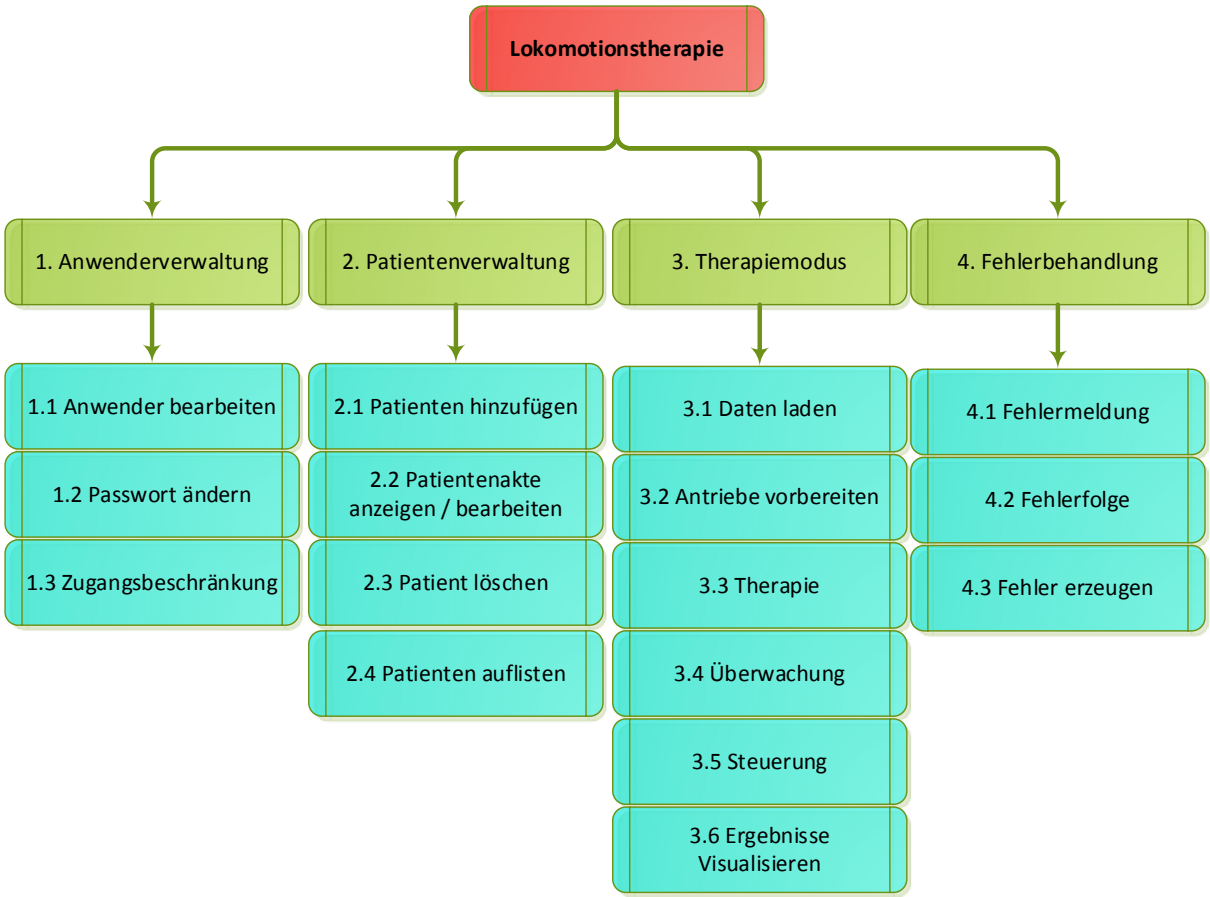


Abbildung 4-1: Vereinfachtes Hierarchiediagramm der Software zur Lokomotionstherapie.

Lokomotionstherapie

Die Software-Komponente Lokomotionstherapie stellt das Grundgerüst, sowie die GUI des Hauptmenüs zur Verfügung. Sie koordiniert den Aufruf der folgenden Software-Komponenten:

1. Anwenderverwaltung
2. Patientenverwaltung
3. Therapiemodus
4. Fehlerbehandlung

1. Anwenderverwaltung

Diese Software-Komponente dient zur Verwaltung von Anwendern. Je nach Benutzerinteraktion im Hauptmenü wird die untergeordnete Software-Komponente, welche die geforderte Funktionalität umsetzt, aufgerufen.

1.1 Anwender bearbeiten

Diese Software-Komponente dient dazu, um neue Anwender hinzuzufügen, Anwender zu löschen oder deren Daten (Anmeldename, Passwort und Adminrechte) bearbeiten zu können. Diese Funktionalität steht nur einem Anwender mit Adminrechten zur Verfügung.

1.2 Passwort ändern

Diese Software-Komponente ermöglicht, dass der Anwender sein eigenes Passwort ändern kann.

1.3 Zugangsbeschränkung

Diese Software-Komponente realisiert die Zugangsbeschränkung der Lokomotionstherapie indem überprüft wird, ob die Login-Daten in der verschlüsselten Datei „AnwenderDaten.xml“ vorhanden sind.

2. Patientenverwaltung

Diese Software-Komponente stellt die GUI zur Verfügung, in der alle vorhandenen Patienten angezeigt werden. Je nach Benutzerinteraktionen können Patientendaten angezeigt, bearbeitet, gelöscht oder neue Patienten zum Verwaltungssystem hinzugefügt werden.
Die einzelnen Funktionen werden durch Aufruf der untergeordneten Software-Komponenten realisiert.

2.1 Patienten hinzufügen

Diese Software-Komponente stellt die GUI und Funktionalität zur Verfügung, um einen neuen Patienten zum Verwaltungssystem hinzufügen zu können. Für jeden Patienten müssen die sicherheitsrelevanten Daten angegeben werden, ansonsten wird ein Fehler ausgegeben und der Patient kann im System nicht registriert werden.

2.2 Patientenakte anzeigen / bearbeiten

Diese Software-Komponente dient dazu, dass der Anwender die Patientenakte eines Patienten einsehen, analysieren und bearbeiten kann. Die Patientenakte bezeichnet die Gesamtheit der gespeicherten Daten eines Patienten und umfasst Patienten- sowie Therapiedaten.

2.3 Patient löschen

Mit Hilfe dieser Software-Komponenten wird der in „2. Patientenverwaltung“ ausgewählte Patient, inklusive allen gespeicherten Daten, aus dem Verwaltungssystem gelöscht.

2.4 Patienten auflisten

Diese Software-Komponente dient dazu, alle vorhandenen Patientenakten in der übergeordneten Software-Komponente „2. Patientenverwaltung“ mit *Vorname / Name / Geburtsdatum* in einem Listenelement aufzulisten.

3. Therapiemodus

Diese Software-Komponente stellt den umfangreichsten und wichtigsten Teil des Software-Systems dar. Sie stellt die GUI zur Durchführung der Therapie zur Verfügung. In dieser Komponente werden alle untergeordneten Komponenten parallel ausgeführt, mit der Ausnahme von „3.1 Daten laden“ und „3.6 Ergebnisse visualisieren“. Die einzelnen Software-Komponenten kommunizieren untereinander mittels Nachrichten-Queues, Meldungen und Referenzen auf gemeinsamen Variablen[7]. Je nach Benutzerinteraktionen werden entsprechende Nachrichten an die Software-Komponente „3.5 Steuerung“ gesendet und dort weiterverarbeitet.

3.1 Daten laden

Diese Komponente dient dazu, die Daten des ausgewählten Patienten, sowie alle Daten die zur Durchführung der Therapie benötigt werden, zu laden.

[7] Eine gute Einführung zur Programmierung von parallelen Prozessen und mögliche Kommunikationskonzepte ist in [37] zu finden.

3.2 Antriebe vorbereiten

Diese Software-Komponente dient zur Vorbereitung der Antriebe. Dies umfasst folgende Punkte:

- Initialisierung der USB2CAN Schnittstelle
- Initialisierung der Motion-Controller
- Konfiguration der Motion-Controller
- Referenzierung der Antriebe

3.3 Therapie

Diese Software-Komponente ist zuständig für die Steuerung der Antriebe, sobald der Patient vorbereitet wurde. Es werden die Bewegungsbefehle für die Erzeugung der Gangzyklen an die Motion-Controller gesendet. Diese Software-Komponente führt folgende Arbeitsschritte aus:

- Starten der Antriebe (Antriebe werden mit Spannung versorgt und die Positionsregler werden aktiviert)
- Anfahren der Startpositionen
- Starten der Therapie und zyklische Vorgabe der Bewegungsbefehle

3.4 Überwachung

Diese Software-Komponente dient zur Überwachung der Sicherheit. Folgende Aufgaben werden umgesetzt:

- Kontinuierliches Abfragen folgender Parameter von den Motion-Controllern:
    - Position
    - Stromaufnahme
    - Antriebszustand
- Überwachung der Antriebszustände
- Drehmomenten-Überwachung
- Encoder-Überwachung
- Node-Guarding
- Aufmerksamkeitsüberwachung

3.5 Steuerung

Diese Software-Komponente dient zur Steuerung des Programmablaufes im Therapiemodus und ist als Zustandsautomat aufgebaut. Sie verarbeitet die Nachrichten beziehungsweise Meldungen der Benutzerschnittstelle und der Software-Komponenten im Therapiemodus. Je nach Nachricht und aktuellem Zustand teilt sie den einzelnen Software-Komponenten Aufgaben zu. Zusätzlich werden die Anzeige- und Eingabe-Elemente der GUI des Therapiemodus verwaltet.

3.6 Ergebnisse visualisieren

Mit Hilfe dieser Software-Komponente werden die Ergebnisse der durchgeführten Therapie grafisch dargestellt. Zusätzlich kann der Therapeut Bemerkungen zur Therapie eintragen. Nach der Visualisierung werden die Therapiedetails in die Patientenakte gespeichert.

4. Fehlerbehandlung

Diese Software-Komponente dient zum Darstellen und Speichern von Fehlern, welche während der Ausführung des Programms auftreten. Je nach Art des Fehlers wird entschieden, ob das Programm aufgrund des Fehlers beendet werden muss.
Die einzelnen Funktionen werden durch Aufruf der untergeordneten Software-Komponenten realisiert und folgend beschrieben.

4.1 Fehlermeldung

Diese Software-Komponente präsentiert dem Anwender die Fehlermeldung.

4.2 Fehlerfolge

Diese Software-Komponente realisiert das Verhalten im Fehlerfall. Anhand einer definierten Entscheidungstabelle wird bestimmt, ob das Programm aufgrund des Fehlers beendet werden muss oder weiterlaufen kann.

4.3 Fehler erzeugen

Diese Software-Komponente wird aufgerufen, wenn ein selbst definierter Fehler erzeugt wird. Wichtige Daten für die spätere Speicherung und Darstellung des Fehlers werden in definierter Form in den Error-Cluster eingefügt.
Selbst definierte Fehler sind alle Fehler, die nicht von LabVIEW selbst erkannt werden. Dazu zählen unter anderem:

- Prüfsummenfehler beim Laden korrupter Daten (Patientenakten, Winkelverläufe,…)
- Fehler bezüglich des Betriebs des Lokomotionsgerätes
    - Kommunikationsfehler
    - Encoderfehler
    - Drehmomentenüberschreitung
    - Endstufenfehler
    - usw.

Eine vollständige Liste der Fehler ist in „**DD-Fehlerdefinitionen-V-0.1.docx“** [61] im digitalen Anhang zu finden.

4.4.2 Ablauf

Nachdem nun die wesentlichen Software-Komponenten beschrieben wurden, wird im Folgenden der Ablauf am Beispiel des Ablaufdiagramms in Abbildung 4-2 erklärt.

Die Software startet mit der Anmeldung am System. Der Anwender wird solange aufgefordert seinen Anwendername und Passwort anzugeben, bis entweder seine Daten korrekt eingegeben wurden oder bis dieser das Programm beendet.

Nach erfolgreicher Anmeldung erscheint das Hauptmenü mit Anzeige des Anwendernamens, sowie des aktuellen Datums und der Zeit.
Nun hat der Anwender die folgende Wahl:

- **Beenden** des Programms.
- **Passwort ändern**: Änderung seines eigenen Passwortes.
- **Anwenderwechsel**: Einen Anwenderwechsel durchführen.
- **Anwenderverwaltung:** Andere Anwender bearbeiten (nur für Administratoren).
- **Patientenverwaltung**: Patienten verwalten.
- **Therapiemodus**: In den Therapiemodus wechseln.

Passwort ändern

In dieser GUI kann der Anwender sein Passwort ändern.

Anwenderwechsel

Der Anwender wird abgemeldet und die Anmeldemaske erscheint.

Anwenderverwaltung (nur für Administratoren)

Beim Aufruf werden alle im System registrierten Anwender aufgelistet.
Anschließend können Anwender gelöscht, bearbeitet oder ein neuer Anwender zum System hinzufügt werden. Für jede Aufgabe erscheint eine eigene GUI.
Nachdem die Aufgabe durchgeführt wurde, erscheint wieder die Liste mit den aktualisierten Anwendern.

Patientenverwaltung

Beim Aufruf werden alle im System registrierten Patienten angezeigt. Der Anwender kann nun Patienten löschen oder zum System hinzufügen. Zusätzlich kann die Patientenakte eines Patienten eingesehen werden, um den Therapieverlauf zu analysieren oder Patientendaten zu ändern. Für jede Aufgabe erscheint eine eigene GUI.

Nachdem die Aufgabe durchgeführt wurde, erscheint wieder die Liste mit den aktualisierten Patienten.

Therapiemodus

Im Therapiemodus erscheint das Therapiemenü zur Durchführung der Therapie. Anschließend wird der Anwender aufgefordert, den Patienten, mit dem die Therapie durchgeführt wird, auszuwählen. Danach werden die entsprechenden Patientendaten sowie die notwendigen Daten zur Durchführung der Therapie geladen.

Als nächster Schritt werden die Antriebe vorbereitet, indem die folgenden Vorgänge schrittweise durchgeführt werden:

- Initialisierung der USB2CAN Schnittstelle
- Initialisierung der Motion-Controller
- Konfiguration der Motion-Controller
- Referenzierung der Antriebe

Bevor nun mit der Therapie begonnen werden kann, muss zuerst der Therapeut den Patienten für die Lokomotionstherapie vorbereiten und dem Programm die erfolgreiche Vorbereitung bestätigen. Anschließend werden die Antriebe aktiviert, die Startposition angefahren und mit der Therapiedurchführung begonnen.

Nach Abschluss der Therapie werden noch die Ergebnisse der durchgeführten Therapie dargestellt, um dem Therapeuten eine Analyse zu ermöglichen. Zusätzlich können Anmerkungen zur der Therapie eingetragen werden.

Nach dem Schließen der Visualisierungsmaske erscheint wieder das Hauptmenü.

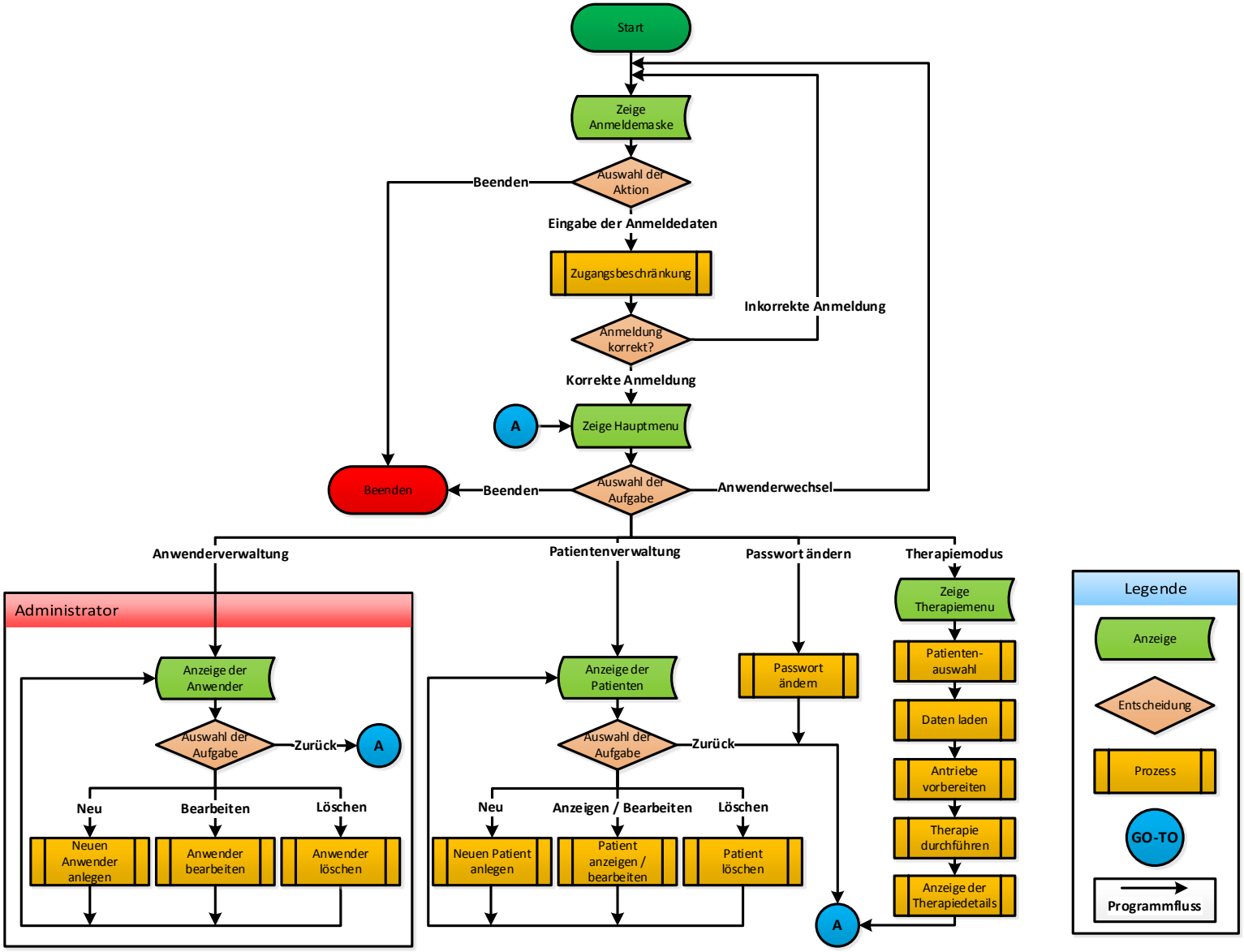


Abbildung 4-2: Programmablauf der Lokomotionstherapie.

4.5 Software-Detail-Design

Im Zuge des Software-Detail-Designs wurden 79 Software-Komponenten entwickelt und in 135 Design-Dokumenten ausführlich dokumentiert. Um den schriftlichen Teil dieser Arbeit in einem begrenzten Rahmen zu halten, werden ausgewählte Beispiele näher erläutert. Der interessierte Leser wird auf das „Detail-Design-Hauptdokument-V-0.2.docx“ [59] und die zugehörigen Dokumente im digitalen Anhang verwiesen. Eine Übersicht der entwickelten Software-Komponenten ist in Tabelle 15 dargestellt.

Tabelle 15: Liste der entwickelten Software-Komponenten.

Software-Komponente	Software-Komponente
0. Lokomotionstherapie-V-0.2	3.1.2 Patient wählen-V-0
1. Anwenderverwaltung-V-0.1	3.2 Antriebe vorbereiten-V-0
1.0.1 Anwenderdaten laden-V-0	3.2.1 Init USB2CAN-V-0
1.0.2 Anwenderdaten schreiben-V-0	3.2.1.1 Reset USB-CAN-V-0
1.1 Anwender bearbeiten-V-0.1	3.2.2 Init Endstufe-V-0
1.1.1 Neuen Anwender erzeugen-V-0	3.2.3 Referenzierung-V-0
1.1.2 Anwender ändern-V-0.1	3.2.3.1 Automatisch Referenzieren-V-0
1.1.3 Anwender löschen-V-0	3.2.3.2 Manuell Referenzieren-V-0
1.1.4 Anwender darstellen-V-0	3.2.3.0.1 Nullpunkt verschieben-V-0
1.2 Passwort ändern-V-0.1	3.2.4 Setting Antriebe-V-0
1.3 Zugangsbeschränkung-V-0.1	3.3 Therapie-V-0
2. Patientenverwaltung-V-0.2	3.3.1 Startposition anfahren-V-0
2.1 Patient hinzufügen-V-0.1	3.3.2 zyklische Bewegung-V-0
2.2 Patientenakte anzeigen_bearbeiten-V-0.2	3.4 Überwachung-V-0
2.2.1 Patientendaten ändern-V-0.2	3.4.1 Leseschleife-V-0
2.2.2 Therapiedetail anzeigen-V-0	3.4.1.1 CAN-Nachrichten Verwalten-V-0
2.2.3 Übersicht der Therapien darstellen-V-0	3.4.1.1.1 Abfrage Verwaltung-V-0
2.2.4 Therapieinfos2Grafik-V-0	3.4.1.1.2 Antriebsstatus Verwalten-V-0
2.2.5 Aktualisierungsdatum prüfen-V-0	3.4.1.1.3 Weitergabe undefinierte Nachrichten-V-0
2.2.6 Patient identifizieren-V-0	3.4.1.1.4 Node Guarding Verwaltung-V-0
2.3 Patient löschen-V-0	3.4.2 Zustandsüberwachung-V-0
2.4 Patienten auflisten-V-0	3.4.3 Drehmomentenüberwachung-V-0
2.0.1 Alter berechnen-V-0	3.4.4 Missing Code Überwachung-V-0
2.0.2 Patientenakte schreiben-V-0	3.4.5 Node Guarding-V-0
2.0.3 Foto kopieren-V-0	3.4.6 Abfrage Schleife-V-0
2.0.4 Patientenakte laden-V-0	3.4.6.1 Abfrage-Konfiguration-V-0
2.0.5 Foto laden-V-0	3.4.6.2 Abfrage anfordern-V-0
3. TherapiemodusV-0.1	3.4.7 Aufmerksamkeitsüberwachung-V-0
3.0.1 Antriebe starten-V-0	3.5 SteuerungV-0.1
3.0.2 Kraftlevel einstellen-V-0	3.5.1 Orthesen einstellen-V-0
3.0.3 NodeID2Antrieb-V-0	3.5.2 Therapieinfos live darstellen-V-0
3.0.4 Bewegung vorbereiten-V-0	3.5.3 Online FehlerbehandlungV-0.1
3.0.5 Bewegung starten-V-0	3.6 Ergebnisse visualisieren-V-0
3.0.6 Antriebe abschalten-V-0	3.6.1 Speichern de Therapieinformationen-V-0
3.0.7 Grad zu Inkremente-V-0	4. FehlerbehandlungV-0.1
3.0.8 Inkremente zu Grad-V-0	4.1 FehlermeldungV-0.1
3.0.9 Timer-V-0	4.2 Fehlerfolge-V-0
3.0.10 Byte2Nmbr-V-0	4.3 Fehler erzeugen-V-0
3.1 Daten laden-V-0	0.1 Prüfsumme-V-0
3.1.1 xml-String laden-V-0	

4.5.1 Referenzierung der Antriebe

Zur Positionsbestimmung wird der optische Impulsgeber HEDS 5500 A-12 (Dr. Fritz Faulhaber GmbH, Deutschland) [17] verwendet und stellt Relativpositionen zur Verfügung. Aus diesem Grund muss vor jeder Verwendung des Gerätes, eine Referenzierung aller Antriebsachsen durchgeführt werden.
Zur Referenzierung stehen keine Endlagenschalter zur Verfügung und somit muss die Referenzierung mit Hilfe der Stromaufnahme der Endstufen realisiert werden. Folgendes Prinzip wird für die Referenzierung angewandt:

- Wahl eines vorläufigen Koordinatensystems
- Die Orthesengerüste werden langsam in Richtung der Winkelbeschränkung bewegt
- Beim Erreichen der Winkelbeschränkung steigt der Motorstrom sprunghaft an, um das erhöhte externe Moment zu kompensieren
- Der Strom jedes einzelnen Motors wird kontinuierlich mit einer Abtastperiode von 10 ms ausgelesen
- Falls der durchschnittliche Motorstrom über eine Dauer von 40 ms einen definierten Schwellwert überschreitet, wird die aktuelle Position als Endlage detektiert
- Beim Erkennen der Endlage wird das Koordinatensystem referenziert
- Nach der Erkennung der Endlage werden die Orthesengerüste in die Nulllage bewegt
- Jede Achse wird zeitgleich referenziert. Die Detektion der Endlage und die Rückführung in die Nulllage werden jedoch für jeden Antrieb individuell durchgeführt
- Während der Referenzierung darf sich kein Patient und keine Person im Bewegungsbereich der Orthesen befinden

Die Schwellwerte wurden auf einen möglichst niedrigen Wert eingestellt, um den mechanischen Aufbau des Gerätes zu schonen und die Wahrscheinlichkeit einer Verletzung von Personen zu minimieren.
Folgende Stromgrenzen für die Erkennung der Endlagen wurden definiert:

- **Hüfte**: 1800 mA
- **Knie**: 600 mA

Für das Hüftgelenk muss eine höhere Stromgrenze verwendet werden, da diese Achse den Motor des Kniegelenks mit anheben muss und deshalb mehr Strom benötigt.

Als redundante Sicherheitsvorrichtung wird während Referenzierung eine zusätzliche Strombegrenzung durch die Endstufen realisiert. Diese greift ein, falls die softwaremäßige Begrenzung ausfallen sollte. Falls der Motorstrom die Grenzwerte der endstufeninternen Begrenzung überschreitet, gehen die Antriebe in den Fehlerzustand über, schalten sich ab und senden eine Fehlernachricht an die übergeordnete Steuerungssoftware.

Folgende Stromgrenzen für die endstufeninterne Begrenzung wurden eingestellt:

- **Hüfte**: 3000 mA
- **Knie**: 1000 mA

In Abbildung 4-3 ist der Winkelverlauf (links) und Stromverlauf (rechts) während der Referenzierung dargestellt. Die sprunghafte Änderung des Winkelverlaufes (links) entsteht durch die Änderung des Bezugskoordinatensystem bei Referenzierung.

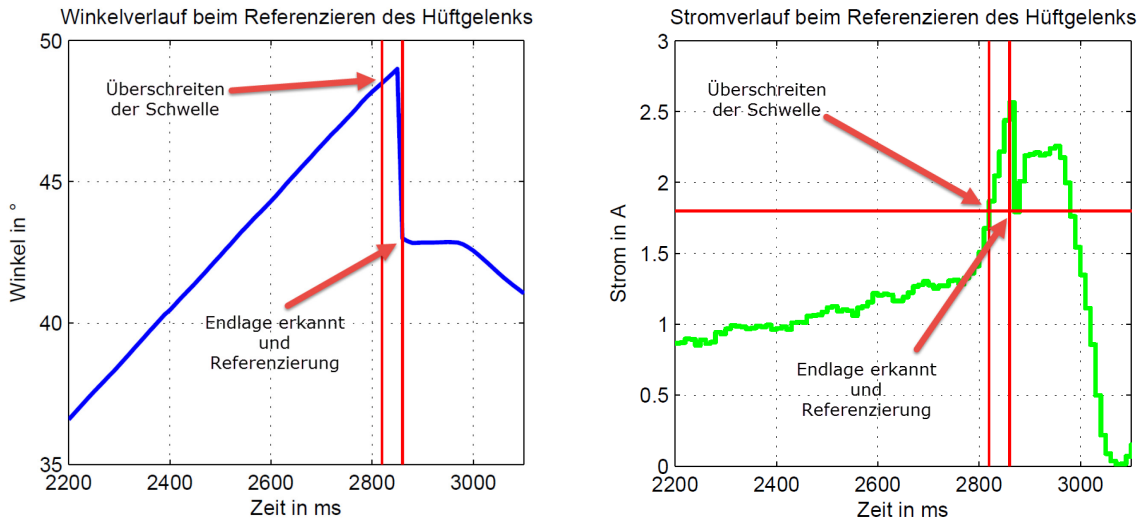


Abbildung 4-3: Winkelverlauf (links) und Stromverlauf (rechts) beim Referenzieren des Hüftgelenks; rote Hilfslinien zur Darstellung der Überschreitung der Schwelle sowie des Zeitpunktes der Referenzierung.

4.5.2 Feedback zur Therapie

Eine Wunschanforderung der Physiotherapeuten ist, dass ein Feedback zur Therapie visualisiert werden soll. Es wird erfasst, in welchem Maß der Patient die Bewegung selbst erzeugt, beziehungsweise wie viel Kraft durch die Antriebe zugeführt werden muss, um die Bewegung durchzuführen.

Mit der vorhandenen Hardware kann die Patientenkraft nicht direkt gemessen werden, jedoch kann über den Motorstrom den die Motion-Controller zur Verfügung stellen, auf das aktuelle Drehmoment der einzelnen Orthesen zurückgerechnet werden.
Der Zusammenhang zwischen dem erzeugten Drehmoment M und dem Motorstrom I, ist für einen permanent erregten Gleichstrommotor, bei Vernachlässigung von Verlusten, durch die Drehmomentkonstante k_M und dem Übersetzungsverhältnis i wie folgt gegeben:

$$\mathrm{M} = \mathrm{k_M} \cdot I \cdot i \tag{4}$$

Somit besteht ein direkt proportionaler Zusammenhang zwischen Motormoment und Motorstrom.

Das berechnete Drehmoment setzt sich aus den folgenden Teilen zusammen:

- Drehmoment zur Erzeugung der Bewegung der Orthesengerüste ($\stackrel{\text{def}}{=}$ Orthesendrehmoment)
- Drehmoment zur Erzeugung der Bewegung der unteren Extremitäten des Patienten ($\stackrel{\text{def}}{=}$ Patientendrehmoment)

Die Orthesendrehmomente können durch Versuche am leeren Lokomotionsgerät bestimmt werden, indem für jedes Gelenk der Motorstrom bei jeder einstellbaren Schrittgeschwindigkeit aufgezeichnet wird. Der Motorstrom (Leer-Stromaufnahme) wird über einen Gangzyklus gemittelt, geglättet und als Look-Up-Table gespeichert.

Für die Darstellung des Therapiefeedbacks wurde festgelegt, dass das Patientendrehmoment im Verhältnis zum Orthesendrehmoment angegeben wird und als Aufwand bezeichnet wird. Somit kürzen sich die Drehmomentkonstante und die Übersetzung aus der Gleichung. Der Aufwand in Prozent kann wie folgt angegeben werden:

$$\text{Aufwand} = \left(\frac{\bar{I}_{\text{ist}} - \bar{I}_0(Kadenz)}{\bar{I}_o(Kadenz)} \right) \cdot 100\ \% \tag{5}$$

$\bar{I}_{ist}$ entspricht dem gemessenen Motorstrom, gemittelt über den letzten Gangzyklus, und $\bar{I}_o(Kadenz)$ entspricht der Leer-Stromaufnahme, gemittelt über einen Gangzyklus, in Abhängigkeit der eingestellten Kadenz. In Abbildung 4-4 ist der mittlere Leer-Stromverlauf für das Hüft- und Kniegelenk dargestellt.

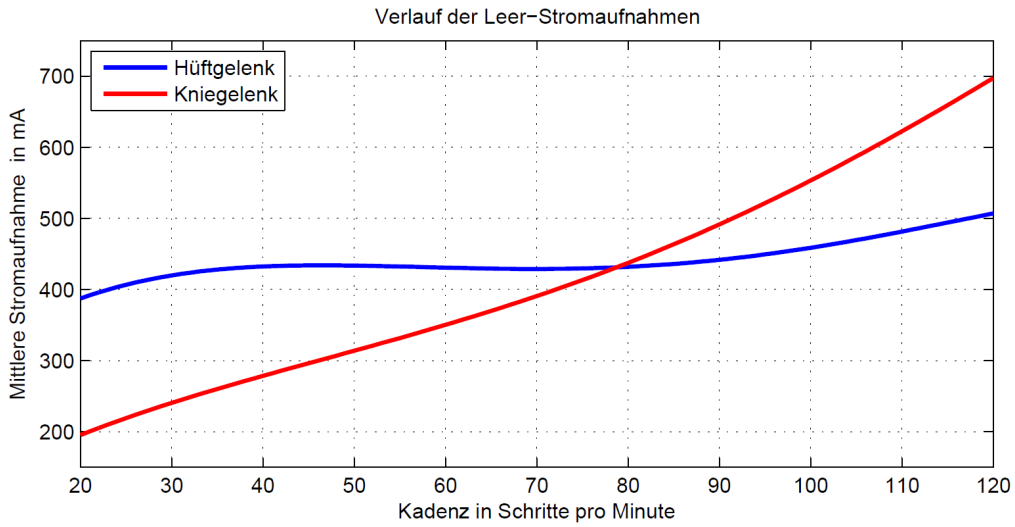


Abbildung 4-4: Verlauf der Leer-Stromaufnahmen zur Berechnung des Aufwandes.

4.5.3 Sicherheitsmechanismen

Um einen Einblick in die Sicherheitsmechanismen zu geben, wird zuerst kurz das allgemeine Verhalten der Software im Fehlerfall beschrieben und anschließend werden die einzelnen Fehlererkennungsmechanismen genauer erläutert.
Falls durch die Fehlererkennungsmechanismen der Software ein Fehler erkannt wird oder im Programm selbst ein Fehler auftritt, wird eine „Shut-Down“ CAN-Nachricht an jede Endstufe gesendet. Diese schalten die Antriebe sofort ab und die Orthesen sind frei beweglich. Das Programm gibt eine entsprechende Fehlermeldung aus und wechselt in einen sicheren Zustand.
Unter den Umständen, dass durch einen Programmfehler doch noch Bewegungsbefehle an die Endstufen gesendet werden würden, lehnt die Endstufe diese Befehle ab. Bevor die Endstufe wieder Bewegungsbefehle annimmt, muss eine definierte Abfolge von Befehlen an die Endstufe gesendet werden (vgl. Kapitel 3.2.1.1 Zustandsmaschine).

Als Informationsquelle der Fehlererkennung dienen die Stromaufnahmewerte, sowie die Positionswerte der einzelnen Antriebe. Diese werden mit einer Abtastrate von 100 Hz zur Verfügung gestellt. Bei einer Zustandsänderung oder bei Auftreten eines internen Fehlers der Motion-Controller, werden ereignisgesteuerte Status- und Fehlernachrichten an die Software gesendet.

4.5.3.1 Zustands-Überwachung

Diese Software-Komponente dient zur Überwachung der Zustände der einzelnen Endstufen, um im Fehlerfall ein synchrones Abschalten aller Endstufen zu gewährleisten. Die Endstufen werden so konfiguriert, dass die folgenden internen Endstufenfehler erkannt werden:

- Stromgrenze überschritten
- Überspannung detektiert
- Temperatur überschritten
- Speicherfehler
- CAN-Überwachungsfehler
- Interner Softwarefehler
- Positionsabweichung überschritten

Wenn eine Endstufe einen internen Fehler erkennt, wechselt diese in den Fehler-Zustand und versendet eine Fehler- und eine Zustandsänderungsnachricht. Die Zustands-Überwachung empfängt diese Nachrichten, schaltet alle Antriebe ab und geht in einen sicheren Zustand über.

Zusätzlich sind die Endstufen so konfiguriert, dass diese im Fehlerzustand den Fehlerausgang schalten. In Kombination mit einer Relaisschaltung wird dadurch die Stromversorgung zu allen

Motoren unterbrochen. Hierbei soll angemerkt werden, dass es irrelevant ist, welche Endstufe ihren Fehlerausgang schaltet. Es werden immer alle Antriebe deaktiviert. Für eine genauere Beschreibung bezüglich der Dimensionierung dieser Schaltung wird auf die Bachelorarbeit von A. Zangl [62] verwiesen.

4.5.3.2 Node-Guarding

Fast alle Sicherheitsmechanismen sind auf eine zuverlässige Kommunikation zwischen den Endstufen und der übergeordneten Steuerung angewiesen. Ein Ausfall der Kommunikation würde zum unbemerkten Ausfall weiterer Sicherheitsmechanismen führen. Deshalb ist die Überwachung der Kommunikation mittels Node-Guarding einer der wichtigsten Sicherheitsmechanismen.

Für die Erklärung des Node-Guarding Prinzips wird vereinfacht nur eine Endstufe betrachtet. (Siehe Abbildung 4-5)

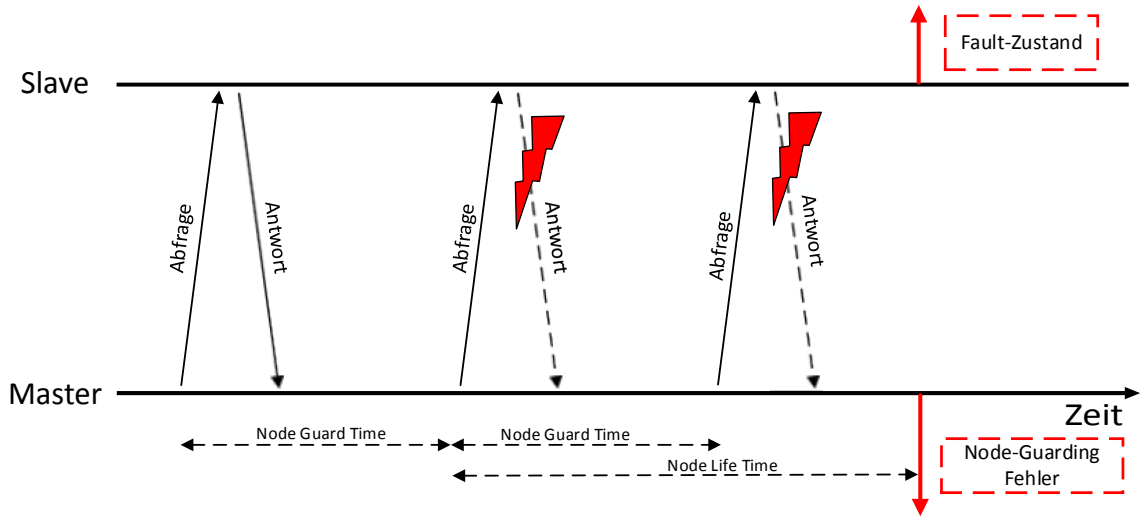


Abbildung 4-5: Prinzip des Node-Guardings.

Die übergeordnete Steuerung (PC) fungiert dabei als Master und die Endstufe als Slave. Der Master sendet zyklisch eine Node-Abfrage an den Slave und dieser antwortet darauf. Die Abfrageperiode wird durch die Node Guard Time beschrieben.

Falls der Slave nicht innerhalb der Node Life Time auf eine Anfrage reagiert, wird ein Node-Guarding Fehler erzeugt und an die Fehlerverarbeitung weiter gegeben. Der Slave wechselt ebenfalls in den Fault-Zustand falls keine Node-Abfrage des Masters innerhalb der Node Life Time eintrifft. Somit überwachen sich die beiden Systeme gegenseitig und wechseln in einen sicheren Zustand im Fehlerfall.

Folgende Werte werden verwendet:

- Node Guard Time: 50 ms
- Node Life Time: 200 ms

4.5.3.3 Missing-Code-Überwachung

Die Aufgabe der Missing-Code Überwachung ist die Erkennung möglicher Encoderfehler der optischen Impulsgeber HEDS 5500 A-12 (Dr. Fritz Faulhaber GmbH, Deutschland) [17]. Ursachen für Encoderfehler könnten zum Beispiel Kabelbrüche oder Kontaktfehler sein. Um das Prinzip des Sicherheitsmechanismus besser zu verstehen, wird zuerst das Verhalten des Reglers beim Auftreten eines Encoderfehlers beschrieben.
Es wird angenommen, dass eine Positionierung durchgeführt wird und ein Encoderfehler auftritt. Durch diesen Fehler werden keine Drehgeber-Impulse an die Regler gesendet und dieser errechnet eine konstante Position mit der Geschwindigkeit null. Dadurch steigt der Drehzahlfehler sprunghaft an und der Regler versucht die aufgetretene Störung durch sprunghafte Erhöhung des Motorstroms zu kompensieren. Im weiteren Verlauf wird durch den anwachsenden Drehzahlfehler, sowie durch den Integralanteil des PI-Reglers, der Motorstrom weiter erhöht.

Aus der Beschreibung der Fehlerfolge konnten folgende Kriterien für die Erkennung eines Encoderfehlers abgeleitet werden:

- Sprunghafte Änderung des Motorstroms:
    - Steigung des Motostroms muss einen Grenzwert für 40 ms überschreiten.
    - $\frac{\Delta I}{\Delta t} > 5\ \frac{A}{s}$
- Messung gleicher Positionswerte:
    - 5 aufeinanderfolgende Positionswerte sind identisch.

Um Fehldetektionen zu minimieren wurden noch folgende Ausschlusskriterien definiert:

- Motoren müssen in Bewegung sein:
    - Die Antriebe dürfen in den letzten 50 ms keine Endposition erreicht haben.
    - Erreichte Endpositionen werden durch eine Statusänderung „Target-Reached“ der Endstufen signalisiert.
- Eine Gefährdung tritt erst bei vorhandenem Motormoment auf. Der Motorstrom muss einen Minimalwert für 20 ms überschreiten.
    - Hüfte > 1200 mA
    - Knie > 650 mA

Die eingestellten Motorstromgrenzen entsprechen ungefähr dem Motormoment, welches benötigt wir, um das Bewegungsprofil im Leerlauf bei einer Kadenz von 20 Schritten pro Minute gerade noch durchführen zu können.
Wenn alle oben genannten Kriterien gleichzeitig erfüllt sind, wird ein Encoderfehler erzeugt und an die Fehlerverarbeitung weiter gegeben.

4.5.3.4 Drehmomenten-Überwachung

Um das Verletzungsrisiko durch zu hohe Drehmomentabgabe zu verringern, wird eine indirekte Drehmomenten-Überwachung durchgeführt. Wie bereits in Kapitel 4.5.2 Feedback zur Therapie erwähnt, steht für die direkte Messung des Drehmoments keine Hardware zur Verfügung. Jedoch kann das Drehmoment durch die Messung des Motorstromes ermittelt werden. Die Drehmomentenbegrenzung wird durch eine Strombegrenzung realisiert. Wenn der Motorstrom die Stromgrenze für 50 ms überschreitet, wird eine Drehmomentenüberschreitung ausgelöst und an die Fehlerverarbeitung weitergeleitet.

Der Strombedarf zur Generierung des Bewegungsmusters der leeren Orthesengerüste legt den untersten Strombedarf fest und ist von der Schrittgeschwindigkeit abhängig. Die obere Stromgrenze ist durch den maximal zulässigen Dauerstrom der Gleichstrommotoren begrenzt. Durch Versuche am leeren Lokomotionsgerät kann der Strombedarf für die Bewegung der Orthesen bestimmt werden. Für jedes Gelenk wurde der maximale Strombedarf eines Gangzyklus in Abhängigkeit der Schrittgeschwindigkeit aufgenommen. Die aufgenommenen Kurven wurden anschließend geglättet und als Look-Up-Table gespeichert.
Um den Patienten einem möglichst geringen Risiko auszusetzen, kann für jeden Patienten eine individuelle Strombegrenzung in Form eines Kraftlevels eingestellt werden. Die Strombegrenzung $i_{Grenze}(K)$ berechnet sich in Abhängigkeit des Kraftlevels und der Kadenz *K* wie folgt:

$$i_{Grenze}(K) = i_{Bedarf,K} + \left(\frac{Kraftlevel}{8} \cdot i_{Bedarf,30}\right). \tag{6}$$

$i_{Bedarf,K}$ bezeichnet den Strombedarf bei der Kadenz *K* und $i_{Bedarf,30}$ den Strombedarf bei einer Kadenz von 30 Schritten pro Minute.
Die berechnete Stromgrenze wird zusätzlich auf 95 % der internen Strombegrenzung der Endstufen ($0{,}95 \cdot 4800$ mA) begrenzt.
Die Kraftlevels können für das Hüftgelenk und das Kniegelenk separat in eingestellt werden. Um eine willkürliche Erhöhung der Kraftlevels zu unterbinden, kann ein Kraftlevel immer nur um eine Stufe erhöht werden, wenn die Drehmomentgrenze überschritten wurde. Der Stromunterschied zwischen den einzelnen Levels, beziehungsweise der Ausdruck $\frac{Kraftlevel}{8} \cdot i_{Bedarf,30}$ wurde heuristisch durch Experimente ermittelt.
In Abbildung 4-6 ist der Verlauf des Strombedarfs der Motoren für das Knie und Hüftgelenk, in Abhängigkeit der Kadenz, dargestellt. Zusätzlich sind noch die Kraftlevels 1 bis 10 angeben.

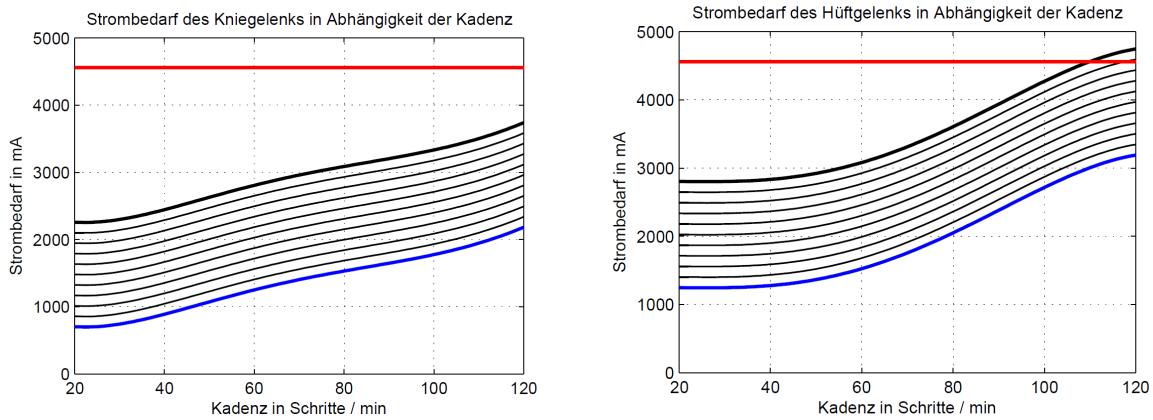


Abbildung 4-6: Verlauf des Strombedarfs (blau) der Motoren des Kniegelenks (links) und des Hüftgelenks (rechts) in Abhängigkeit der Kadenz; Stromgrenzen der ersten 10 Kraftlevels (schwarz); oberste Strombegrenzung (rot).

Falls der unwahrscheinliche Fall eintreten sollte, dass die Drehmomentenbegrenzung nicht ordnungsgemäß funktionieren sollte, greift die interne Strombegrenzung der Endstufen und die Endstufe geht beim Überschreiten der Stromgrenze in den Fehler-Zustand über. Die Stromgrenze wurde auf 4800 mA eingestellt.

4.5.3.5 Aufmerksamkeit-Überwachung

Um das Risiko bei Verletzung der Aufsichtspflicht durch den Therapeuten zu minimieren, wurde eine Aufmerksamkeitsüberwachung implementiert. Der Therapeut muss innerhalb einer definierten Zeit seine Anwesenheit bestätigen, ansonsten wird die Therapie angehalten und die Antriebe werden abgeschalten. Die Überwachung wurde folgendermaßen umgesetzt.

Beginnend bei einem definierten Startwert dekrementiert ein Timer in Sekundenschritten. Wenn der Timer die erste Schwelle unterschreitet, wird ein kurzer Signalton ausgegeben um den Therapeuten auf die Bestätigung seiner Anwesenheit hinzuweisen. Die Anwesenheit kann durch Bewegung der Maus bestätigt werden, wodurch der Timer wieder auf den Startwert zurückgesetzt wird.
Falls der Therapeut seine Anwesenheit bis zum Erreichen der zweiten Schwelle nicht bestätigt, wird ein hoher kurzer Signalton in einem Abstand von zirka einer Sekunde wiederholt ausgegeben.
Falls beim Ablauf des Timers immer noch keine Bestätigung des Therapeuten registriert wurde, wird ein langer hoher Signalton ausgeben, die Therapie angehalten und die Antriebe werden abgeschalten.

Die Parameter können über die verschlüsselte Datei „Init.xml“ vorgegeben werden. Folgende Werte sind implementiert:

- **Startzeit**: 40 Sekunden
- **1. Schwelle**: 10 Sekunden
- **2. Schwelle:** 5 Sekunden

4.6 Implementieren-Verifizieren-Integrieren

Alle 79 Software-Komponenten der ersten Version **Lokomotionstherapie-V-0** wurden in 30 IVI-Phasen implementiert und erfolgreich getestet. Nach der ersten Phase der Gebrauchstauglichkeitstests [63], wurden in vier weiteren IVI-Phasen 11 Software-Komponenten geändert und anschließend wieder erfolgreich verifiziert.

Ein Teil der Änderungsanträge der zweiten Phase der Gebrauchstauglichkeitstests wurde in einer dritten Iteration, in vier weiteren IVI-Phasen, durchgeführt. Es wurden neun weitere Software-Komponenten geändert und erfolgreich geprüft.

Alle Software-Komponenten des aktuellen Software-Systems **Lokomotionstherapie-V-0.2** wurden implementiert und erfolgreich getestet. Der „Implementieren-Verifizieren-Integrieren-Plan“ sowie alle 99 Testprotokolle sind im digitalen Anhang zu finden.

4.7 Verifizierung des Software-Systems

Die Software-System Verifizierung wurde mit der Version **Lokomotionstherapie-V-0.2** durchgeführt. 82 der 86 Software-Anforderungen wurden durch einen umfangreichen Funktionstest geprüft. Dieser umfasste 146 Eingabewerte beziehungsweise Ablaufschritte sowie 74 Überprüfungsschritte.

Vier Software-Anforderungen konnten nicht durch diesen Funktionstest geprüft werden, jedoch konnten diese Software-Anforderungen bereits durch Funktionstests in der Aktivität „Implementieren-Integrieren-Verifizieren“ abgedeckt werden.

Für die Verifizierung der Bewegungsmuster wird auf das folgende Unterkapitel 4.7.1 verwiesen.

Alle Testergebnisse fielen positiv aus und das Software-System wurde erfolgreich verifiziert. Der vollständige Bericht „SoftwareSystem-Verifizierung-V-0.docx“ [64] ist im digitalen Anhang zu finden.

4.7.1 Verifizierung der Bewegungsmuster

Im Zuge der Verifizierung der Bewegungsmuster wurde der Ist-Verlauf der Orthesen mit Hilfe der Motion-Controller bei einer Kadenz von 20, 60 und 120 Schritten pro Minute aufgezeichnet. Der Vergleich zwischen den Soll- und Ist-Verläufen, sowie der Fehler ist in Abbildung 4-7 dargestellt.

Der maximale Fehler der Kniewinkelverläufe liegt für alle 3 Geschwindigkeiten unter 1,7 Grad. Der Fehler für den Hüftwinkelverlauf beträgt zirka 0,8 Grad, wobei dieser bei einer Kadenz von 20 beziehungsweise 60 Schritte pro Minute geringer ist als bei 120 Schritten pro Minute.

Die Variabilität der Winkelverläufe eines physiologischen Gangzyklus, dargestellt als Standardabweichung, beträgt zirka $\sigma \approx 2$ Grad für das Hüftgelenk und $\sigma \approx 3$ Grad für das Kniegelenk [36]. Die vorhandenen Positionsfehler liegen innerhalb der physiologischen Abweichung und die Forderung eines physiologischen Bewegungsmusters ist somit erfüllt.

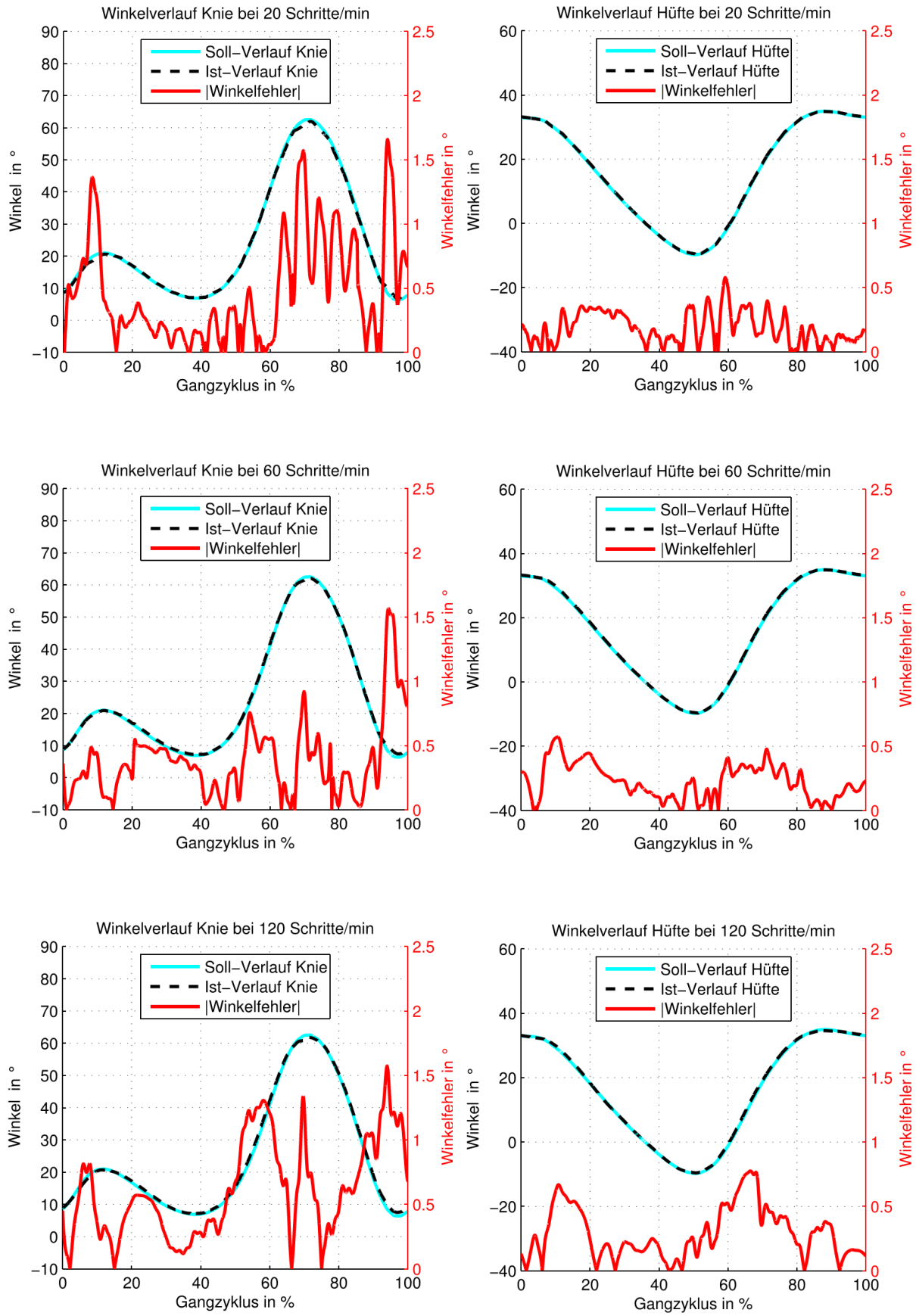


Abbildung 4-7: Soll-Ist-Vergleich der Winkelverläufe bei 20, 60 und 120 Schritte/min.

4.8 Änderungsanträge und Problemberichte

Die aktuelle Software Version durchlief zwei Revisionszyklen, in der die Ergebnisse der ersten und zweiten Phase der Gebrauchstauglichkeitstests [63] eingearbeitet wurden. In Summe wurden 25 Problemberichte erstellt. Der Großteil der Probleme sind Verbesserungsvorschläge (7 Berichte) und Änderungswünsche (11 Berichte). Fünf Probleme wurden als mittlere Fehler und kein einziges Problem als schwerer Fehler eingestuft. Bei zwei Problemberichten konnte der Fehlerfall nicht nachgestellt und die Ursache, bis zum Zeitpunkt der Fertigstellung dieser Arbeit, nicht noch nicht identifiziert werden. Eine vorläufige Analyse der Auswirkungen stufte die zwei Probleme als leichte Fehler ein. Zur Behebung der Fehler werden jedoch noch weitere Analysen und Informationen benötigt.

Auf Basis dieser Problemberichte wurden 12 Änderungsanträge genehmigt. Diese enthalten alle bestätigten Probleme der Kritikalitätsstufe größer-gleich drei (5 Änderungsanträge) und 7 Probleme mit der Kritikalitätsstufe zwei.

Die Hälfte der Änderungsanträge wurde vollständig umgesetzt und deren Probleme gelöst. Die restlichen Änderungsanträge beinhalten nur noch Änderungswünsche (Kritikalitätsstufe 2) und müssen erst in zukünftigen Versionen umgesetzt werden.

4.9 Freigabe

Im Zuge der Software Freigabe wurde die Verifizierung der Software **Lokomotionstherapie-V-0.2** überprüft und als vollständig abgeschlossen bewertet.
Die bestehenden Restanomalien sind in Tabelle 16 dargestellt. Aufgrund der unbekannten Fehlerursachen konnten diese noch nicht korrigiert werden. Nach Analyse der Auswirkungen in Bezug auf die Sicherheit wurden die Risiken der Restanomalien als akzeptierbar bewertet. Somit kann die Software **Lokomotionstherapie-V-0.2** freigegeben werden.

Tabelle 16: Vorhandene Restanomalien des Software-Systems: Lokomotionstherapie-V-0.2.

Problem-bericht	Beschreibung	Auswirkung auf die Sicherheit
PB-5	Erneutes Öffnen des Therapiemodus nach Beenden des Therapiemodus.	• Programmteil muss wieder geschlossen werden. **Keine Auswirkung auf die Sicherheit.**
PB-24	Programmabsturz nach Drücken des Not-Stopp Buttons.	• Patient muss aus dem Gerät entfernt werden. • Software muss neu gestartet werden. • Erneutes Durchführen der Referenzierung. • Erneutes Vorbereiten des Patienten. • Verlust von Therapiezeit. **Keine Auswirkung auf die Sicherheit.**

5 Diskussion

Ziel dieser Arbeit war es eine Software für den Prototyp des Lokomotionsgerätes für Kleinkinder zu entwickeln. Die entwickelte Software umfasst neben der Generierung der Bewegungsverläufe, auch eine Patienten- und Anwenderverwaltung, sowie zahlreiche Sicherheitsmechanismen um eine sichere Therapie zu gewährleisten.

Die wesentlichen Herausforderungen dieser Arbeit lagen einerseits in der Umsetzung der Steuerung des Gerätes, sodass eine sichere Therapie gewährleistet ist und andererseits in der korrekten Umsetzung der Anforderungen für den Entwicklungsprozess nach EN 62304. Eine Herausforderung war die Frage wie und auf welche Weise die Anforderungen der EN 62304 umgesetzt werden könnten. Die einzelnen Schritte und Vorgehensweisen sowie die Dokumente, in Bezug auf die Umsetzung der EN 62304, wurden sehr ausführlich und mit zahlreichen Beispielen beschrieben. Bis auf die Umsetzung eines QM-Systems, sowie des Software-Wartungsprozesses, wurden alle Anforderungen der EN 62304, in einem vernünftigen Rahmen für eine Software der Software-Sicherheitsklasse B, erfüllt.

Bei der Planung der Software-Entwicklung wurde von einem überschaubaren Entwicklungsaufwand ausgegangen. Auf Basis dieser Einschätzung wurde somit eine manuelle Verwaltung der Versionen gewählt. Mit der fortschreitenden Entwicklung wuchs jedoch der Dokumentationsaufwand stark an. Bei der Umsetzung der Revisionen stellte sich heraus, dass aufgrund der hierarchischen Software-Struktur, bereits bei kleinen Änderungen der Software mit einem beträchtlichen Aufwand zu rechnen ist. Dies betrifft einerseits die Software selbst, aber auch die Änderungen der Dokumentation. Zusätzlich mussten noch zahlreiche Software-Tests wiederholt werden. Aus diesem Grund wird für eine weiterführende Entwicklung, die Einführung eines automatischen Versionskontrollsystems sowohl für die Dokumentation, als auch für den Quellcode empfohlen. Zusätzlich sollten automatisierte Komponenten-Tests, sowie statische Code-Analyse-Methoden eingeführt werden, um den Zeitaufwand zur Umsetzung von Änderungen zu minimieren.

Für die Erstellung, Freigabe und Durchführung von Software-Tests beziehungsweise der Software-System-Verifizierung, wird nach EN 62304 keine explizite Unabhängigkeit des Personals zwischen den Implementierungs- und Testschritten gefordert [41]. Wenn jedoch nur der Entwickler seinen Code testet, wird der Testfokus vorwiegend auf die schwierigen Aufgaben gelegt und triviale Fehler werden häufig übersehen. Um die Fehlerentdeckungsrate zu erhöhen, sollte das Entwicklungsteam erweitert werden, um zumindest Tests nach dem 4-Augen-Prinzip durchführen zu können. Dies wird beispielsweise in der EN 60601-1 gefordert, indem klar dargelegt wird, dass für die Verifizierung und

Validierung des kompletten Medizinproduktes eine explizite Unabhängigkeit des Personals zwischen Entwicklung und Testung gefordert wird [65].

Bei der durchgeführten Risikoanalyse für die Software wurden nicht nur Risiken in Bezug auf die Steuerung und Bewegungsgenerierung aufgedeckt, sondern es wurden auch zahlreiche Risiken in Bezug zur Anwenderverwaltung und Patientenverwaltung identifiziert. Sie umfassen mehr als ein Drittel aller Risiko-Kontrollmaßnahmen. Die Ursachen sind vorwiegend auf Bedienfehler, vorhersehbaren Missbrauch sowie auf die Datensicherheit zurückzuführen.

Infolge der zahlreichen Sicherheitsmechanismen wurde die vorerst geplante lineare Programmstruktur (Programmteile werden nacheinander abgearbeitet), in eine komplexere parallele Programmstruktur geändert. Zahlreiche Software-Komponenten arbeiten zeitgleich, kommunizieren miteinander und müssen untereinander synchronisiert werden. Dadurch erhöhten sich einerseits die Anforderungen an die Programmierkenntnisse des Entwicklerteams und andererseits der zeitliche Aufwand für die Entwicklung. Als Resultat wurde jedoch ein effizienteres, sichereres und fehlertoleranteres System entwickelt. Durch die Parallelitäten und Redundanzen kann trotz Ausfall einzelner Sicherheitsmechanismen, ein sicheres Stillsetzen des Lokomotionsgerätes gewährleistet werden.

Aufgrund der zahlreichen Risiko-Kontrollmaßnahmen verblieben nur wenige Risiken in der Risikostufe **III.** Eines dieser Risiken resultiert beispielsweise aus der Notwendigkeit der Referenzierung. Aufgrund der relativen Positionsmessung, muss eine Referenzierung durchgeführt werden. Das Risiko besteht darin, dass während der Referenzierung die softwaremäßige Drehomentenüberwachung nicht durchgeführt werden kann. Hinsichtlich der Minimierung der Risiken, sollte deshalb bei einer Weiterentwicklung des Prototyps das Positionsmesssystem zumindest um einen Sensor pro Motor mit absoluter Winkelmessung der Orthesenwinkel erweitert werden. Somit könnten alle Risiken durch die Referenzierung vollständig vermieden werden.

Im Zuge der Weiterentwicklung könnte das Steuerungskonzept ebenfalls verbessert werden. Mit dem aktuellen Konzept der Bewegungsgenerierung wird zwar eine ausreichend genaue Vorgabe der Winkelverläufe erzielt, jedoch stellt das aktuelle System keine optimale Lösung für eine kontinuierliche Trajektorienvorgabe dar. Die Vorgabe der Positionsbefehle kann nicht exakt zu den festgelegten Zeitpunkten erfolgen, sondern kann je nach Prozessorauslastung verzögert sein. Die Ursache liegt in dem nicht echtzeitfähigen Verhalten des Betriebssystems. Um ruckhafte Positionsverläufe durch die Sendeverzögerungen zu vermeiden, mussten die Regler sehr träge dimensioniert, sowie Modifikationen am Konzept der Bewegungserstellung durchgeführt werden.

Durch diese Modifikationen können jedoch nur stückweise lineare Winkelverläufe vorgegeben werden. Bei langsamen Schrittgeschwindigkeiten wurde dadurch ein etwas ruckartiger Übergang zwischen den einzelnen Teilkurven beobachtet.

Eine mögliche Verbesserung wäre die Verwendung eines Motion-Controllers mit integrierter Trajektoriengenerierung oder der Einbau eines echtzeitfähigen Micro-Controllers als Zwischenelement zwischen PC und Motion-Controller. Dieser könnte die Interpolation der Positionen durchführen und die Bewegungsbefehle zu exakten Zeitpunkten vorgeben.

6 Schlussfolgerung

In dieser Arbeit wurde eine Software zur sicheren Steuerung eines Lokomotionsgerätes inklusive Patientenverwaltung und Anwenderverwaltung implementiert. Eine wesentliche Forderung war es, die EN 62304 für die Entwicklung der Software so gut wie möglich umzusetzen. Bis auf die Erstellung eines QM-Systems, sowie der Umsetzung des Software-Wartungsprozesses, wurden alle Anforderungen an die EN 62304 in einem vernünftigen Rahmen erfüllt.

Während der Entwicklung wurden zwei Revisionszyklen, auf der Grundlage der ersten beiden Phasen der Gebrauchstauglichkeitstests, durchgeführt. Alle kritischen Probleme wurden erfolgreich gelöst, sowie das Software-System und alle Software-Komponenten wurden ausführlich und erfolgreich getestet.

Bei der Durchführung der Risikoanalyse wurden 77 Gefährdungen auf Basis der Software-Architektur identifiziert. Durch die Wahl von geeigneten Abhilfemaßnahmen wurden die Restrisiken soweit reduziert, dass 74 der 77 Risiken in die Risikostufe **IV** eingeteilt werden konnten und somit ein vernachlässigbares Risiko darstellen. Die übrigen drei Risiken wurden der Risikostufe **III** zugeordnet und unter Berücksichtigung der möglichen Risiko-Minimierungsmaßnahmen ebenfalls akzeptiert. Aufgrund des zu erwartenden medizinischen Nutzens, sowie durch den Vergleich mit bestehenden Therapieformen, wird das vorhandene Gesamt-Risiko des entwickelten Gerätes zur Lokomotionstherapie akzeptiert.

Die Risiken durch die vorhandenen Restanomalien wurden ebenfalls als akzeptierbar bewertet und somit wurde die Software **Lokomotionstherapie (**in Version V-0.2) entsprechend der EN 62304 auch freigegeben.

Die Entwicklung der Software für ein Lokomotionsgerät konnte erfolgreich abgeschlossen werden und der Autor dieser Arbeit hofft, dass mit diesem Prototyp ein wichtiger Schritt in Richtung einer verbesserten Therapierung von Kleinkindern mit infantiler Zerebralparese erfolgt.

7 Literatur

[1] A. Johnson, „Prevalence and characteristics of children with cerebral palsy in Europe“, *Dev. Med. Child Neurol.*, Bd. 44, Nr. 9, S. 633–640, 2002.

[2] K. Heimann, *Neurogene Ursachen kindlicher Verhaltensstörungen - Entwicklung eines Screening-Verfahrens*. München: Grin Verlag Gmbh, 2013.

[3] I. Krägeloh-Mann, „Klassifikation, Epidemiologie, Pathogenese und Klinik“, in *Das Kind und die Spastik. Erkenntnisse der evidence based medicine zur Cerebralparese*, Bern: Huber, Hans, 2001, S. 37–48.

[4] J. R. Gage und T. F. Novacheck, „An update on the treatment of gait problems in cerebral palsy.“, *J. Pediatr. Orthop. B*, Bd. 10, Nr. 4, S. 265–274, 2001.

[5] A. I. Van Saanen, „Infantile Cerebralparese-Behandlungsmöglichkeiten und unterstützende Ressourcen für Eltern und das behinderte Kind“, Diplomarbeit, uniwien, 2011.

[6] V. Erkinger, M. Haidvogl, K. Janes, H. Rerrares, P. Brenneis, G. Weninger, und S. Krael, „Vernetzte Therapieangebote für Kinder mit Cerebralparese (0 – 15 Jahre) in der Steiermark“, 2006.

[7] A. Kumar, „Effektivität und Sicherheit der Roboter-gestützten Gangorthese Lokomat bei Kindern und Jugendlichen mit zentralen Gangstörungen“, Dissertation, lmu, 2009.

[8] A. Tilp, „Entwickelung eines Lokomotionsgerätes für Kleinkinder“, Diplomarbeit, 2013.

[9] Reha Stim, „Rehabilitation Kinder“. [Online]. Verfügbar unter: http://www.reha-stim.de/cms/index.php?id=105. [Zugegriffen: 21-März-2016].

[10] D. J. Russell, P. L. Rosenbaum, L. M. Avery, und M. Lane, *Gross Motor Function Measure (GMFM - 66 and GMFM - 88) User's Manual*. London: Mac Keith Press, 2002.

[11] P. L. Rosenbaum, S. D. Walter, S. E. Hanna, R. J. Palisano, D. J. Russell, P. Raina, E. Wood, D. J. Bartlett, und B. E. Galuppi, „Prognosis for gross motor function in cerebral palsy: creation of motor development curves“, *Jama*, Bd. 288, Nr. 11, S. 1357–1363, 2002.

[12] *Richtlinie 93/42/EWG des Europäischen Parlaments und des Rates vom 14. Juni 1993 über Medizinprodukte,*. 1993.

[13] Faulhaber, „Datenblatt 3863 CR (DC-Motor)“. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/resources/img/DE_3863_CR_DFF.PDF. [Zugegriffen: 04-März-2016].

[14] Faulhaber, „Datenblatt Planetengetriebe 38A“. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/resources/img/DE_38A_DFF.PDF. [Zugegriffen: 04-März-2016].

[15] Faulhaber, „Datenblatt Planetengetriebe 38/2“. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/resources/img/DE_38-2S_FMM.PDF. [Zugegriffen: 04-März-2016].

[16] ELRA Antriebstechnik, „Datenblatt Schneckengetriebe SR25“. [Online]. Verfügbar unter: http://www.elra.at/images/produkte/getriebe/datenblaetter/SR25.pdf. [Zugegriffen: 04-März-2016].

[17] Faulhaber, „Datenblatt Optische Encoder HEDS 550“. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/resources/img/DE_HEDS_HEDM_DFF.PDF. [Zugegriffen: 04-März-2016].

[18] Faulhaber, „Datenblatt Motion Controller MCDC 3006“. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/resources/img/DE_MCDC3006_V2_5_DFF.PDF. [Zugegriffen: 04-März-2016].

[19] W. Lawrenz, Hrsg., *CAN System Engineering*. London: Springer London, 2013.

[20] Faulhaber, „Motion Controller Kommunikations-/Funktionshandbuch“. [Online]. Verfügbar unter: http://www.faulhaber.com/manuals/pdf/schnittstelle/canopen_fau_bl_dc/DE_7000_00030.pdf. [Zugegriffen: 04-März-2016].

[21] S. G. Helmut, „CAN Bus Lehrsystem“, Diplomarbeit.

[22] H. Engels, *CAN-Bus: Feldbusse im Überblick, CAN-Bus-Protokolle, CAN-Bus-Meßtechnik, Anwendungen ; [neu: mit TTCAN]*. Franzis, 2002.

[23] G. Schnell und B. Wiedemann, *Bussysteme in der Automatisierungs- und Prozesstechnik*. Vieweg+Teubner Verlag, 2012.
[24] M. Bräutigam und C. Wilmes, „Entwicklung einer Programmierschnittstelle und eines Treibers für eine CANopen-Schnittstellenkarte“, Diplomarbeit, 2004.
[25] IXXAT Automation GmbH, „USB-to-CAN compact- Handbuch“.
[26] Faulhaber, „MCDC3006S_P.JPG“, *Motion Controller Serie MCDC 3006*. [Online]. Verfügbar unter: https://fmcc.faulhaber.com/details/overview/PGR_17309_13833/PGR_13833_13803/de/DE/. [Zugegriffen: 05-März-2016].
[27] W. Schneider, *Praktische Regelungstechnik: ein Lehr- und Übungsbuch für Nicht-Elektrotechniker ; mit 72 Tabellen*. Wiesbaden: Vieweg + Teubner, 2008.
[28] M. Horn und N. Dourdoumas, *Regelungstechnik: Rechnerunterstützter Entwurf zeitkontinuierlicher und zeitdiskreter Regelkreise*, 1 edition. Pearson Studium, 2003.
[29] J. G. Ziegler und N. B. Nichols, „Optimum settings for automatic controllers“, *J. Dyn. Syst. Meas. Control*, Bd. 115, Nr. 2B, S. 220–222, 1993.
[30] K. L. Chien, J. A. Hrones, und J. B. Reswick, „On the Automatic Control of Generalized Passive Systems., In: Transactions of the American Society of Mechanical Engineers., Bd. 74, Cambridge (Mass.), USA, Feb. 1952, S. 175-185“.
[31] U. Kuhn, „Eine praxisnahe Einstellregel für PID-Regler: Die T-Summen-Regel“, *Autom. Prax.*, Bd. 37, Nr. 5, S. 10–16, 1995.
[32] D. S. Christen, *Praxiswissen der chemischen Verfahrenstechnik: Handbuch für Chemiker und Verfahrensingenieure*. Berlin: Springer, 2005.
[33] J. Müller, *Regeln mit SIMATIC: Praxisbuch für Regelungen mit SIMATIC S7 und SIMATIC PCS7: Praxisbuch Fur Regelungen Mit SIMATIC S7 Und SIMATIC PCS7*, 3. überarb. u. erw. Auflage edition. Publicis Corporate Publishing, 2004.
[34] M. Svehlik, „Ganganalysedaten- Ganglabor Graz“.
[35] K. Götz-Neumann, *Gehen verstehen: Ganganalyse in der Physiotherapie*. Georg Thieme Verlag, 2006.
[36] M. H. Schwartz, J. P. Trost, und R. A. Wervey, „Measurement and management of errors in quantitative gait data“, *Gait Posture*, Bd. 20, Nr. 2, S. 196–203, Okt. 2004.
[37] W. Georgi und E. Metin, *Einführung in LabVIEW*, 5. Aufl. München: Carl Hanser Verlag GmbH & Co. KG, 2012.
[38] National Instruments, *LabVIEW Benutzerhandbuch*. 1998.
[39] T. Spranz-Fogasy, *Entwicklung einer automatisierten Dokumentation von LabVIEW Quellcode für das Rahmenwerk CS*. Sonderforschungsbereich 245, 1992.
[40] U. Möschel, „LabVIEW kann auch C“, 2010.
[41] *BS EN 62304:2006 + A1:2015 Medical device software- Software life-cycle processes*. 2015.
[42] *DIN EN ISO 13485:2015-05 Medizinprodukte - Qualitätsmanagementsysteme - Anforderungen für regulatorische Zwecke*. 2015.
[43] *DIN EN ISO 14971:2013-04 Medizinprodukte - Anwendung des Risikomanagements auf Medizinprodukte*. 2013.
[44] *IEC 62366-1:2015-02, Medical devices - Part 1: Application of usability engineering to medical devices*. 2015.
[45] S. Langthaler, „Spezifikation der Gebrauchstauglichkeit“. 2016.
[46] C. Johner, M. Hölzer-Klüpfel, und S. Wittorf, *Basiswissen Medizinische Software: Aus-und Weiterbildung zum Certified Professional for Medical Software*. dpunkt. verlag, 2015.
[47] N. Leitgeb, *Sicherheit von Medizingeräten*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.
[48] M. Kalkgruber, „Software-Sicherheitsklassifizierung“. 2016.
[49] J. Noack und B. Schienmann, „Objektorientierte Vorgehensmodelle im Vergleich“, *Inform.-Spektrum*, Bd. 22, Nr. 3, S. 166–180, 1999.
[50] M. Kalkgruber, „Traceability-V-0.2“. 2016.

[51] *Richtlinie 2006/42/EG des Europäischen Parlaments und des Rates vom 17. Mai 2006 über Maschinen und zur Änderung der Richtlinie 95/16/EG (Neufassung)*. 2006.
[52] D. W. Hoffmann, *Software-Qualität*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.
[53] M. Kalkgruber, „Konfigurationsmanagement.docx“. 2016.
[54] M. Kalkgruber, „RA-V1.docx“. 2016.
[55] M. Kalkgruber, „RA-V2.docx“. 2016.
[56] I. Borggraefe, L. Kiwull, J. S. Schaefer, I. Koerte, A. Blaschek, A. Meyer-Heim, und F. Heinen, „Sustainability of motor performance after robotic-assisted treadmill therapy in children: an open, non-randomized baseline-treatment study“, *Eur. J. Phys. Rehabil. Med.*, Bd. 46, Nr. 2, S. 125–131, Juni 2010.
[57] M. Kalkgruber, „Software-Anforderungen-V-0.2.docx“. 2016.
[58] M. Kalkgruber, „Software-Architektur-Hauptdokument-V-0.docx“. 2016.
[59] M. Kalkgruber, „Detail-Design-Hauptdokument-V-0.2.docx“. 2016.
[60] M. Kalkgruber, „Bedienungsanleitung-V-0.2.docx“. 2016.
[61] M. Kalkgruber, „DD-Fehlerdefinitionen-V-0.1.docx“. 2016.
[62] A. Zangl, „Elektrische Versorgung eines Lokomotionsgerätes“, 2014.
[63] S. Langthaler, „Beurteilung der Gebrauchstauglichkeit eines Lokomotionsgerätes für Kleinkinder“, Masterarbeit, 2016.
[64] M. Kalkgruber, „SoftwareSystem-Verifizierung-V-0.docx“. 2016.
[65] *DIN EN 60601-1:2013-12 : Medizinische elektrische Geräte - Teil 1: Allgemeine Festlegungen für die Sicherheit einschließlich der wesentlichen Leistungsmerkmale*. 2013.

8 Anhang

A. Abkürzungsverzeichnis

ALARA ..... As low as reasonable achievable

ALARP ..... As low as reasonable practicable

a.u. ..... arbitrary unit

CAN ..... Controller Area Network

CiA ..... CAN in Automation

CSMA/CA ..... Carrier Sense Multiple Access with Collision Avoidance

DLC ..... Data Length Code

EMCY ..... Emergencyobjekt

FMEA ..... Fehlermöglichkeits- und Einflussanalyse

GMFM ..... Gross motor function measure

GUI ..... Grafische Benutzeroberfläche

G-Sprache ..... Graphische Programmiersprache

IVI ..... Implementieren-Verifizieren-Integrieren

NC ..... Normal Condition (Normalfall)

NMT ..... Netzwerkmanagement

NRZ ..... Non-Return-To-Zero

PDO ..... Prozessdatenobjekt

PD-Regler ..... Proportional-Differential-Regler

PI-Regler ..... Proportional-Integral-Regler

PWM ..... Pulsweitenmoduliert

QM-Systems ..... Qualitätsmanagement-System

SDO ..... Servicedatenobjekt

SF ..... Single Fault Condition (Erster Fehlerfall)

SOUP ..... Software of unknown provenance (Software unbekannter Herkunft)

SW ..... Software

SYNC ..... Synchronisationsobjekt

VI ..... Virtuelles Instrument

B. Formelzeichen:

Δs .....Positionsunterschied zwischen zwei Teilschritte

Δt .....Zeitbasis entspricht der Zeit zwischen zwei Sendevorgängen

s_i .....Position bei Teilschritt i

s_{i+1}.....Position bei Teilschritt $i + 1$

T_{Ziel} .....Sendeperiode beim der Zielkadenz

T_{120}.....Sendeperiode bei 120 Schritten pro Minute

K_{120} .....Kadenz von 120 Schritten pro Minute

K_{Ziel} .....Zielkadenz

$v_{i,Ziel}$ .....Wert des Geschwindigkeit bei Teilschritt i und Zielkadenz

$\mathrm{v}_{\mathrm{i},120}$.....Wert des Geschwindigkeit bei Teilschritt i und 120 Schritte pro Minute

v_i .....Geschwindigkeit bei Teilschritt i

σ .....Standardabweichung

K.....Kadenz K

$i_{Bedarf,K}$ .....Strombedarf bei der Kadenz K

$\bar{I}_{ist}$ .....Ist-Stromaufnahme gemittelt über einen Gangzyklus

I .....Stromaufnahme

$\bar{I}_o(Kadenz)$ .....Leer-Stromaufnahme gemittelt über einen Gangzyklus als Funktion der Kadenz

$i_{Grenze}(K)$ .....Strombegrenzung bei der Kadenz K

M .....Motormoment

k_M.....Drehmomentkonstante

i .....Übersetzungsverhältnis

C. Abbildungsverzeichnis

D. Tabellenverzeichnis

E. Software-Risikoanalyse

1.Anwenderverwaltung

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
1. Anwender-Verwaltung	1.1 Anwender bearbeiten	hinzufügen von Anwendern	Hinzufügen nichtberechtigter Anwender	NC	Zugriff durch unberechtigten Anwender → Inbetriebnahme durch ungeschulten Anwender → Falsche Handhabung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-1	Einschränkung der Berechtigungen durch: Vergabe von Administrator-rechten für ausgewählte Anwender nur Administratoren dürfen neue Anwender hinzufügen	N	2	J	1	2	IV
						Verletzung des Anwenders oder Dritter	3	2	III						1	2	IV
					Zugriff durch unberechtigten Anwender → Zugriff auf Patientenakte	Datenschutz-verletzung	3	1	III						1	1	IV
		Anwender löschen	Irrtümliches Löschen von Anwenderprofilen	NC	Anwenderprofil ist nicht mehr vorhanden →Anwender kann sich nicht Anmelden →Ausfall der Therapie	keine Linderung / Heilung des Patienten	4	1	III	RA-V2-2	Beim Löschvorgang soll eine zusätzliche Abfrage durchgeführt werden, ob der ausgewählte Anwender gelöscht werden soll.	N	2	J	1	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
1. Anwender-Verwaltung	1.1 Anwender bearbeiten	Speichern der „AnwenderDaten.xml	Ungeschützte "AnwenderDaten .xml"	NC	Zugriff auf AnwenderDaten.xml" → Auslesen von Anwendername + Passwort →Inbetriebnahme durch ungeschulten Anwender → Falsche Handhabung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-3	Verschlüsselung der Daten	N	2	J	1	2	IV
						Verletzung des Anwenders oder Dritter	3	2	III						1	2	IV
					Zugriff auf AnwenderDaten.xml" von Dritten → Auslesen von Anwendername + Passwort → Anmeldung im System → Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III						1	1	IV
	1.2 Passwort ändern	Passwort ändern	Tippfehler bei der Passworteingabe	NC	Speichern des falschen Passwortes → Anwender kann sich nicht Anmelden → Ausfall der Therapie	keine Linderung / Heilung	5	1	III	RA-V2-4	Ein neues Passwort darf nur zur übernommen werden, wenn das Passwort zweifach identisch eingegeben wurde.	N	2	J	2	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
1. Anwender-Verwaltung	1.2 Passwort ändern	Speichern der „AnwenderDaten.xml“	Ungeschützte "AnwenderDaten .xml"	NC	Zugriff auf AnwenderDaten.xml" → Auslesen von Anwendername + Passwort →Inbetriebnahme durch ungeschulten Anwender → Falsche Handhabung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-3	Verschlüsselung der Daten	N	2	J	1	2	IV
						Verletzung des Anwenders oder Dritter	3	2	III						1	2	IV
					Zugriff auf AnwenderDaten.xml" → Auslesen von Anwendername + Passwort → Anmeldung im System → Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III						1	1	IV
	1.3 Zugangs-beschränkung	Passwort-Eingabe	Gedächtnis-schwäche des Anwenders: Passwort und Anwendername sind nicht korrekt	NC	Anwender kann sich nicht anmelden → Ausfall der Therapie	Keine Linderung oder Heilung	4	1	III	RA-V2-5	Administrator darf: Vergabe von Adminrechte Password neu vergeben Anwendername ändern	N	2	J	1	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
1. Anwender-Verwaltung	1.3 Zugangs-beschränkung	Passwort-Eingabe	Ablesen des Passwortes	NC	Anmeldung im System →Inbetriebnahme durch ungeschulten Anwender → Falsche Handhabung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-7	Bei der Eingabe des Passwortes: Passwort Anzeige als „******“	N	2	J	1	2	IV
						Verletzung des Anwenders oder Dritter	3	2	III						1	2	IV
					Anmeldung im System → Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III						1	1	IV
		Laden der "AnwenderDaten.xml"	korrupte "AnwenderDaten. xml"	SFC	Anwender kann sich nicht Anmelden → Ausfall der Therapie	keine Linderung / Heilung	3	1	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden.	N	2	J	2	1	IV

2. Patientenverwaltung

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
2. Patientenverwaltung	2.1 neuen Patienten anlegen	Eingabe der neuen Patientendaten	Überschreitung des zulässigen Gewichts	NC	Beschädigung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-8	Überprüfung des zulässiges Gewichtsbereich bei der Eingabe der Daten keine Therapie möglich, falls Gewicht nicht zulässig	N	2	J	2	2	IV
			Überschreitung des zulässigen Größenbereichs	NC	Orthesen können nicht an den Patienten angepasst werden -> nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	4	2	III	RA-V2-9	Überprüfung des zulässigen Größenbereichs bei der Eingabe der Daten keine Therapie möglich, falls Größe nicht zulässig	N	2	J	2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
2. Patientenverwaltung	2.1 neuen Patienten anlegen	Eingabe der neuen Patientendaten	Fehlerhafte Eingabe von Patientendaten	NC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-10	Erneute Überprüfung der Patientendaten und Bestätigung durch den Anwender	N	3	J	2	2	IV
					Falsche Orthesen-montage → nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	4	2	III						2	2	IV
		Speichern des neuen Patientendatensatzes	Ungeschützte Patientendaten	NC	Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III	RA-V2-3	Verschlüsselung der Daten	N	2	J	1	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
2. Patientenverwaltung	2.2 Patientenakte anzeigen / bearbeiten	Bearbeiten von Patientendaten	Überschreitung des zulässigen Gewichts	NC	Beschädigung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-8	Überprüfung des zulässiges Gewichtsbereich bei der Eingabe der Daten keine Therapie möglich, falls Gewicht nicht zulässig	N	2	J	2	2	IV
			Überschreitung des zulässigen Größenbereichs	NC	Orthesen können nicht an den Patienten angepasst werden -> nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	4	2	III	RA-V2-9	Überprüfung des zulässigen Größenbereichs bei der Eingabe der Daten keine Therapie möglich, falls Größe nicht zulässig	N	2	J	2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
2. Patientenverwaltung	2.2 Patientenakte anzeigen / bearbeiten	Bearbeiten von Patientendaten	Fehlerhafte Eingabe von Patientendaten	NC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-10	Erneute Überprüfung der Patientendaten und Bestätigung durch den Anwender	N	3	J	2	2	IV
				NC	Falsche Orthesen-montage → nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	4	2	III						2	2	IV
		Speichern des aktualisierten Patientendatensatzes	Ungeschützte Patientendaten	NC	Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III	RA-V2-3	Verschlüsselung der Daten	N	2	J	1	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
2. Patientenverwaltung	2.2 Patientenakte anzeigen / bearbeiten	Darstellen von Therapieergebnissen	Auswahl eines falschen Patienten	NC	laden der falschen Patientendaten ->Darstellung falscher Therapieverläufe →fehlerhafte Vorgabe von Therapieparameter	Verletzung / Überforderung des Patienten	4	2	III	RA-V2-11	Zusätzliche Überprüfung der Patientenauswahl durch erneute Bestätigung durch Foto und Patientenname	N	3	J	2	2	IV
		Patientenakte laden	korrupte Patientendaten	SFC	Darstellung falscher Therapieverläufe →Fehlerhafte Vorgabe von Therapieparametern	Verletzung / Überforderung des Patienten	3	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
	2.3 Patient löschen	Patient löschen	Irrtümliches Löschen des falschen Patienten	NC	Verlust von Therapie Informationen → keine optimale Vorgabe der Therapieparameter	geringere Linderung / Heilung	3	1	III	RA-V2-12	Beim Löschvorgang soll eine erneute Abfrage durchgeführt werden, ob der ausgewählte Patient gelöscht werden soll.	N	3	J	2	1	IV

3. Therapiemodus

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.1 Daten laden	Patientenakte laden	Auswahl eines falschen Patienten	NC	Laden der falschen Patientendaten ->Darstellung falscher Therapieverläufe →fehlerhafte Vorgabe von Therapieparameter	Verletzung / Überforderung des Patienten	4	2	III	RA-V2-11	Zusätzliche Überprüfung der Patientenauswahl durch erneute Bestätigung durch Foto und Patientenname	N	3	J	2	2	IV
					Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III						2	2	IV
					Falsche Orthesenmontage → nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	4	2	III						2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.1 Daten laden	Patientenakte laden	korrupte Patientendaten	SFC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
					Falsche Orthesenmontage → nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	3	2	III						1	2	IV
					->Darstellung falscher Therapieverläufe →fehlerhafte Vorgabe von Therapieparameter	Verletzung / Überforderung des Patienten	3	2	III						1	2	IV
			Laden veralteter Daten	NC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-13	Monatliche Abfrage zur Aktualisierung der Patientendaten, Therapie kann nur mit aktuellen Daten durchgeführt werden	N	2	J	2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.1 Daten laden	Patientenakte laden	Laden veralteter Daten	NC	Wachstum des Patienten →Überschreitung des zulässigen Gewichts →Beschädigung des Gerätes	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-13	Monatliche Abfrage zur Aktualisierung der Patientendaten, Therapie kann nur mit aktuellen Daten durchgeführt werden	N	2	J	2	2	IV
					Wachstum des Patienten →Überschreitung des zulässigen Größenbereichs → Orthesen können nicht an den Patienten angepasst werden -> nicht physiologische Bewegung	Verletzung des Patienten: Muskel-verletzung.	3	2	III						2	2	IV
		Laden der Maximalstromwerte zur Drehmomentenbegrenzung	Laden korrupter Daten	SFC	Berechnung falscher Sicherheitsgrenzen → Sicherheitsmaßnahme nutzlos (indirekte Drehmomentbegrenzung) → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
										RA-V2-14	Minimierung der maximalen Stromabgabe durch interne Strombegrenzung der Endstufe	N	1	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.1 Daten laden	Bewegungsverlauf laden	Laden korrupter Daten	SFC	Berechnung falscher Winkelwerte -> Vorgabe eines nicht physiologischen Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J	1	2	IV
										RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J			
			Laden korrupter Daten	SFC	Berechnung falscher Winkelwerte -> Vorgabe eines falschen Verlaufes	Verletzung des Patienten: Muskel-verletzung	4	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
		Grundbewegungs-Energie laden zur Berechnung des Aufwandes	Laden korrupter Daten	SFC	Berechnung/Darstellung falscher Therapieinformationen → fehlerhafte Vorgabe von Therapieparameter	Verletzung / Überforderung des Patienten	3	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.1 Daten laden	Laden der Referenzierungsdaten	Laden korrupter Daten	SFC	Falsche Referenzierungspunkte → Berechnung falscher Winkelwerte -> Vorgabe eines nicht physiologischen Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J	1	2	IV
										RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J			
										RA-V2-16	Bestätigung der Plausibilität der Referenzierung durch den Anwender	N	3	J			
					Falsche Referenzierungspunkte → Berechnung falscher Winkelwerte -> Vorgabe eines falschen Verlaufes	Verletzung des Patienten: Muskel-verletzung	4	2	III	RA-V2-6	Datensätze werden mit Prüfsumme versehen Nur Daten mit korrekter Prüfsumme dürfen verwendet werden	N	2	J	1	2	IV
										RA-V2-16	Bestätigung der Plausibilität der Referenzierung durch den Anwender	N	3	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Referenzieren	Bei der Referenzierung ist die softwaremäßige Drehmomenten-begrenzung nicht aktiviert	NC	Keine Begrenzung des Drehmoments	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	5	2	II	RA-V2-17	Bestätigung durch Anwender, dass sich kein Patient im Gerät und keine Person im Bewegungsbereich der Orthesen befindet	N	3	J	3	2	III
										RA-V2-18	Hinweis in der GA: Während der Referenzierung darf sich kein Patient und keine Person im Bewegungsbereich der Orthesen befinden.	N	4	J			
										RA-V2-14	Minimierung der maximalen Stromabgabe durch interne Strombegrenzung der Endstufe	N	1	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Referenzieren	Bei der Referenzierung ist die softwaremäßige Drehmomenten-begrenzung nicht aktiviert	NC	Keine Begrenzung des Drehmoments	Verletzung des Anwenders oder Dritter	3	2	III	RA-V2-17	Bestätigung durch Anwender, dass sich kein Patient im Gerät und keine Person im Bewegungsbereich der Orthesen befindet	N	3	J	2	2	IV
										RA-V2-18	Hinweis in der GA: Während der Referenzierung darf sich kein Patient und keine Person im Bewegungsbereich der Orthesen befinden.	N	4	J			
										RA-V2-14	Minimierung der maximalen Stromabgabe durch interne Strombegrenzung der Endstufe	N	1	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Referenzieren	Fehlerhafte Referenzierung	SFC	Falsche Referenzierungspunkte → Berechnung falscher Winkelwerte -> Vorgabe eines nicht physiologischen Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J	1	2	IV
										RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J			
										RA-V2-16	Bestätigung der Plausibilität der Referenzierung durch den Anwender	N	3	J			
					Falsche Referenzierungspunkte → Berechnung falscher Winkelwerte -> Vorgabe eines falschen Verlaufes	Verletzung des Patienten: Muskel-verletzung	4	2	III	RA-V2-16	Bestätigung der Plausibilität der Referenzierung durch den Anwender	N	3	J	2	2	IV
										RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Initialisierung der Drehrichtung	Fehlerhafte Kommunikation: Drehrichtung wird nicht übernommen	SFC	Falsche Vorgabe von Winkelpositionen -> Vorgabe eines nicht physiologischen Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	2	IV
										RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J			
					Falsche Vorgabe von Winkelpositionen -> Vorgabe eines falschen Verlaufes	Verletzung des Patienten: Muskel-verletzung	3	2	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Setzen des Positioniermodus	Vorgabe von Absolut-Positionen (RA-V2-23) Fehlerhafte Kommunikation: Fehlerhafte Vorgabe des Positioniermodus	SFC	Positioniermodus wird nicht übernommen →Endstufe interpretiert Positionsbefehle als Relativpositionen → Vorgabe falscher Positionswerte → nicht physiologischer Winkelbereich	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	2	3	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	3	IV
					Positioniermodus wird nicht übernommen → Endstufe interpretiert Positionsbefehle als Relativpositionen → Vorgabe falscher Positionswerte → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	2	2	IV						1	2	IV
		Fehlerregister setzen	Fehlerhafte Kommunikation: Fehlerregister für Fehlerausgang wird nicht übernommen	SFC	Ausfall der Sicherheitsmaßnahme (RA-V2-31 und RA-V2-25) → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	2	3	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	3	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.2 Antriebe vorbereiten	Fehlerregister setzen	Fehlerhafte Kommunikation: Fehlerregister für Fehlermeldung wird nicht übernommen	SFC	Ausfall der Sicherheitsmaßnahme (RA-V2-30) → Nur eine Endstufe wird abgeschaltet →Verlust der Synchronität → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	3	2	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	2	IV
			Fehlerhafte Kommunikation: Fehlerregister für die Abschaltung der Endstufe wird nicht übernommen	SFC	Ausfall der Sicherheitsmaßnahme (RA-V2-29 und RA-V2-25) → nicht physiologischer Winkelbereich	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	2	3	III	RA-V2-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	3	IV
			Fehlerhafte Kommunikation: Stromgrenzen für interne Strombegrenzung der Endstufen werden falsch / nicht übernommen	SFC	Ausfall der Sicherheits-maßnahme RA-V2-14 → Zu hohe Drehmomenten-abgabe	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	2	III	RA-V-19	Überprüfung der gesendeten Initialisierungsdaten+ Wiederholtes Senden im Fehlerfall	N	2	J	1	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Startposition anfahren	Anfahren der Startposition mit vorgegebener Zielkadenz	NC	Sprunghafte Vorgabe der Geschwindigkeit → Hohe Geschwindigkeitsänderung → Hohe Beschleunigungskräfte auf den Patienten	Verletzung des Patienten: Muskel-verletzung	5	2	II	RA-V2-20	Beschränkung der Geschwindigkeits-änderung beim Anfahren der Startposition	N	1	J	2	2	IV
		Ändern der Zielkadenz	Ändern der Zielkadenz durch den Anwender	NC	Sprunghafte Vorgabe der Geschwindigkeit → Hohe Geschwindigkeitsänderung → Hohe Beschleunigungskräfte auf den Patienten	Verletzung des Patienten: Muskel-verletzung	5	2	II	RA-V2-21	Beschränkung der Geschwindigkeits-änderung bei Zielkadenzänderung	N	1	J	1	2	IV
		Senden der Bewegungsbefehle	Sendeverzögerung der Bewegungsbefehlen durch zu geringe Rechenresourcen	SFC	Fehlerhafte Vorgabe von Winkelwerten → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-22	Überprüfen des Sendeperiode beim Sendezyklus	N	2	J	1	2	IV
										RA-V2-23	Laufender Soll / Istwert-Vergleich	N	2	N			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Senden der Bewegungsbefehle	Elektromagnetische Störeinkopplungen	SFC	fehlerhafte Daten-übertragung → Erzeugung von undefinierten Ausgangswerten (Position, Geschwindigkeit, Drehmoment) → Vorgabe eines nicht physiologischen Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-24	Verwendung eines CAN-Bus als Übertragungs-element	N	1	J	2	2	IV
										RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J			
			Verlust Kommunikation PC-Endstufe	SFC	Ausfall einer / mehrerer Endstufen →Verlust von Synchronität → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	4	2	III	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikationsv erlust und schaltet sich ab • Software erkennt Kommunikationsv erlust und deaktiviert alle Antriebe	J	2	J	2	2	IV
			Verlust Kommunikation PC-Endstufe Annahme: (relative Positionsvorgabe)	SFC	Verlust eine Positionsnachricht → Vorgabe falscher Positionswerte → nicht physiologischer Winkelbereich	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-26	Vorgabe von Absolut-Positionen	N	1	J	1	3	VI

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Senden der Bewegungsbefehle	Verlust Kommunikation PC-Endstufe Annahme: (relative Positionsvorgabe)	SFC	Verlust eine Positionsnachricht → Vorgabe falscher Positionswerte → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	4	2	III	RA-V2-26	Vorgabe von Absolut-Positionen	N	1	J	2	2	IV
			Verhalten nach Spannungs-wiederkehr der Endstufe	SFC	Undefinierter Zustand der Endstufe → Ungewollte Bewegung → Verletzung des Patienten durch unerwartete Bewegung	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	4	3	II	RA-V2-27	Endstufen befinden sich nach Spannungs-wiederkehr nicht im Operativen Modus	N	1	J	2	2	IV
										RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikationsverlust und schaltet sich ab • Software erkennt Kommunikationsverlust und deaktiviert alle Antriebe	J	2	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Erfassung der Positionswerte	Encoderstörungen	SFC	Erfassung falscher Positionswerte(Position ändert sich nicht mehr) →Erhöhung des Motorstromes → Eröhung des Antriebsmoment → Antrieb beschleunigt undefiniert → Verlassen des physiologischer Winkelbereichs	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-28	Softwaremäßige Detektion von Encoderstörung	N	2	J	2	2	IV
										RA-V2-15	Konstruktive Einschränkung des Winkelbereichs	N	1	J			
		Senden der Bewegungsbefehle	Überhitzung aufgrund von Überlastung	SFC	Motoren werden zu heiß →Wärmeenergie der Motoren	Patient erleidet Verbrennungen	3	2	III	RA-V2-29	Fehlererkennung durch die Endstufe und Abschalten der Endstufe	J	2	J	2	2	IV
			Rückwirkung (RA-V2-29): Abschalten einer Endstufe bei Fehlersituation	NC	Nur eine Endstufe wird abgeschaltet →Verlust der Synchronität → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	5	2	II	RA-V2-30	Fehlermeldung an Software und Abschaltung aller Endstufen	N	2	J	2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Senden der Bewegungsbefehle	Rückwirkung (RA-V2-29): Abschalten einer Endstufe bei Fehlersituation	NC	Nur eine Endstufe wird abgeschaltet →Verlust der Synchronität → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	5	2	II	RA-V2-31	Antriebe im Fehlerzustand schalten den Fehlerausgang, welcher über eine Relaisschaltung die Stromversorgung der Antriebe trennt.	N	2	J	2	2	IV
			Defekt einer Endstufe	SFC	-> Endstufe lässt sich nicht mehr kontrollieren	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikations-verlust und schaltet sich ab • Software erkennt Kommunikations-verlust und deaktiviert alle Antriebe	N	2	J	1	3	IV
										RA-V2-31	Antriebe im Fehlerzustand schalten den Fehlerausgang, welcher über eine Relaisschaltung die Stromversorgung der Antriebe trennt.	N	2	J			

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Senden der Bewegungsbefehle	Absturz vom Betriebssystem oder der Steuerungs-software	SFC	Wiederholte Ausgabe des gleichen Befehls → Vorgabe derselben Position → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	3	1	VI	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikations-verlust und schaltet sich ab • Software erkennt Kommunikations-verlust und deaktiviert alle Antriebe	N	2	J	2	1	IV
					Ausgabe eines undefinierten Befehl →Endstufe aktzeptiert falschen Befehl nicht → Keine Änderung der Position → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	3	1	VI	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikations-verlust und schaltet sich ab • Software erkennt Kommunikations-verlust und deaktiviert alle Antriebe	N	2	J	2	1	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.3 Therapie	Senden der Bewegungsbefehle	Absturz vom Betriebssystem oder der Steuerungs-software	SFC	Keine Ausgabe eines Befehls → Keine Änderung der Position → nicht physiologischer Bewegungsverlauf	Verletzung des Patienten: Muskel-verletzung	3	1	VI	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikations-verlust und schaltet sich ab • Software erkennt Kommunikations-verlust und deaktiviert alle Antriebe	N	2	J	2	1	IV
		Durchführung der Therapie	Anwender vernachlässigt Aufsichtspflicht	NC	Anwender vernachlässigt Aufsichtspflicht	Verletzung des Patienten: Muskel-verletzung und Frakturen der unteren Extremität	5	2	II	RA-V2-32	Regelmäßige Bestätigung der Anwesenheit des Anwenders durch die Software, Stoppen der Therapie bei Zeitüberschreitung	N	2	J	2	2	IV

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapiemodus	3.4 Überwachung	Indirekte Drehmomentenbegrenzung	Willkürliche Erhöhung der Grenze	NC	Zu hoch eingestellte Grenze → Ausfall der Sicherheitsmaßnahme → zu hohe Drehmomentenabgabe	Verletzung des Patienten: Muskelverletzung und Frakturen der unteren Extremität	4	2	III	RA-V2-33	Drehmomentgrenze Stufenweisen einstellbar: Erhöhung um maximal 1 Stufe je Drehmomenten-überschreitung	N	2	J	2	2	IV
		Abschaltung der Endstufen im Fehlerfall	Kommunikations-probleme	SFC	Deaktivierungsbefehl wird nicht übernommen → Antriebe werden im Fehlerfall nicht deaktiviert	Verletzung des Patienten: Muskelverletzung und Frakturen der unteren Extremität	4	3	II	RA-V2-25	Node-Guarding: bei Kommunikations-verlust: • Endstufe erkennt Kommunikations-verlust und schaltet sich ab • Software erkennt Kommunikations-verlust und deaktiviert alle Antriebe	N	2	J	2	3	III
		Sicherheits-maßnahme	Fehlerfall einer Softwaresicherheits maßnahme	SFC	Sicherheitsmaßnahme nicht aktiv	Verletzung des Patienten: Muskelverletzung und Frakturen der unteren Extremität	3	3	III	RA-V2-34	Deaktivierung der Antriebe bei Ausfall einer Software-Sicherheitsmaßname	N	2	J	2	3	III

Komp.		Fkt./ Aufg.	Ursache	NC / SFC	Folge von Ereignissen zur Gefährdungssituation	Schaden	E	S	R	ID	Abhilfe	RW	SS	DM	E	S	RR
3. Therapie	3.5 Visualisieren und speichern	Aktualisieren der Patientenakte	Ungeschützte Patientendaten	NC	Zugriff auf Patientendaten	Datenschutz-verletzung	3	1	III	RA-V2-3	Verschlüsselung der Daten	N	2	J	1	1	IV

F. Detaillierte Beschreibung der CAN-Schnittstelle

CAN (Controller Area Network) wurde 1983 von R. Bosch GmbH als Kommunikationsprotokoll für Bus-Systeme für den Automobilbereich entwickelt. Im weiteren Verlauf wurde es als ISO 11898 im Jahr 1993 standardisiert. Das CAN-Protokoll beschreibt die Bitübertragungsschicht (1. Schicht) und der Sicherungsschicht (2. Schicht) im OSI-Referenzmodell. Prinzipiell wird der CAN-Bus in zwei Arten unterteilt. Einerseits dem „Low-Speed-CAN“, welcher eine Datenrate von 40-250 kbit/s bei einer maximalen Bus-Länge von 1000 m ermöglicht, sowie fehlertolerant in Bezug auf Leitungsschäden ist. Die zweite Variante stellt der „High-Speed-CAN“ dar. Dieser erreicht eine Datenübertragungsrate von bis zu 1 Mbit/s bei einer Bus-Länge von 40 m. [19]

Nach [21] bietet eine CAN-Schnittstelle folgende Vorteile:

- einfacher Busaufbau
- Multi Master System
- nachrichtenorientiertes Protokoll
- Priorisierung von Nachrichten
- hohe Störfestigkeit des Protokolls
- optimiert für sehr kleine Datenmengen
- kurze Latenzzeit für hoch priorisierte Nachrichten

Für die Kommunikation zwischen den Endstufen und der übergeordneten Steuerung wird ein „High-Speed-CAN“ mit einer Datenübertragungsrate von 1 Mbit/s verwendet. Die weiteren Erklärungen beziehen sich auf den „High-Speed-CAN“.

F.1. Topologie

Die CAN Schnittstelle ist als BUS-Topologie ausgelegt. Ein Bus-System bietet den Vorteil das zusätzliche Teilnehmer im Kommunikationssystem mit einem geringen Aufwand hinzugefügt werden können. Bei einer Erweiterung muss lediglich eine Stichleitung für CAN_L, CAN_H sowie einer gemeinsamen Masseleitung GND zur Verfügung gestellt werden. Um Reflektionen an den Enden der Busleitungen zu vermeiden, muss der Bus mit dem Leitungswellenwiderstand (120 Ω) an beiden Enden abgeschlossen werden. Die Topologie des CAN-BUS ist in Abbildung 8-1 dargestellt [19].

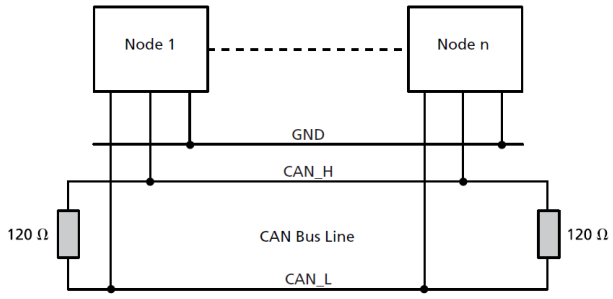


Abbildung 8-1: Bus-Topologie beim CAN-BUS. Entnommen aus [20].

F.2. Funktionsweise

CAN ist ein Multimaster System und dies bedeutet, dass jeder Teilnehmer gleichberechtigt Nachrichten versenden kann. Um Kollisionen zu vermeiden, wird die Zugriffsberechtigung mit dem CSMA/CA-Verfahren (Carrier Sense Multiple Access with Collision Avoidance) geregelt. Die erste Voraussetzung für diese Verfahren ist, dass sich beim Senden unterschiedlicher logischer Pegel durch mehrere Teilnehmer sich der dominante Zustand durchsetzt. Zum Beispiel sind zwei Knoten mit einem logischen UND verknüpft, so setzt sich immer die logische 0 gegenüber der logischen 1 durch. Ist diesem Fall sowie beim CAN-Protokoll ist die logische 0 der dominante Zustand. [23]

Die zweite Voraussetzung ist, dass jeder Teilnehmer ständig, auch während seines Sendevorgangs, die Nachrichten auf dem Bus ausliest. Versuchen nun zwei Teilnehmer eine Nachricht gleichzeitig zu versenden, so senden beide Teilnehmer so lange bis sich der logische Pegel eines Senders vom logischen Pegel des Busses unterscheidet. Der durch den dominanten Pegel überstimmte Sender erkennt den Pegelunterschied und stoppt seinen Sendevorgang. Dieses Verfahren ermöglicht somit eine verzögerungsfreie und priorisierbare Übertragung. Je geringer der numerische Wert des Arbitrierungsfeldes ist (siehe Kapitel F.3), desto höher ist die Priorität der Nachricht. Ein Beispiel dieses Verfahrens, auch Busarbitration genannt, ist in Abbildung 8-2 dargestellt. [23]

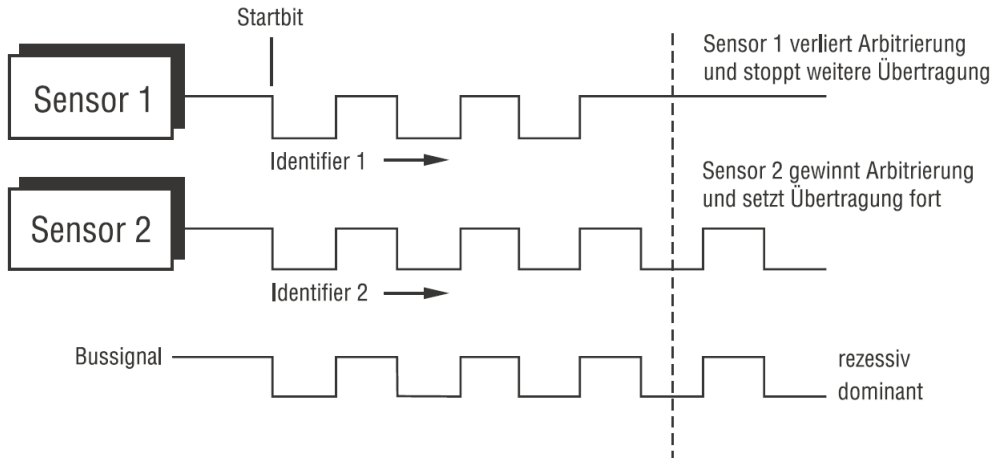


Abbildung 8-2: CAN-Busarbitration. Entnommen aus [23].

Die Signalübertragung beim CAN erfolgt mittels Differenzsignale und ist nach den NRZ-Verfahren (Non-Return-To-Zero) codiert. Somit werden Gleichtakt-Störungen besser unterdrückt und ermöglichen eine störfestere Datenübertragung. Die Pegel des CAN-BUS sind in Abbildung 8-3 dargestellt [19].

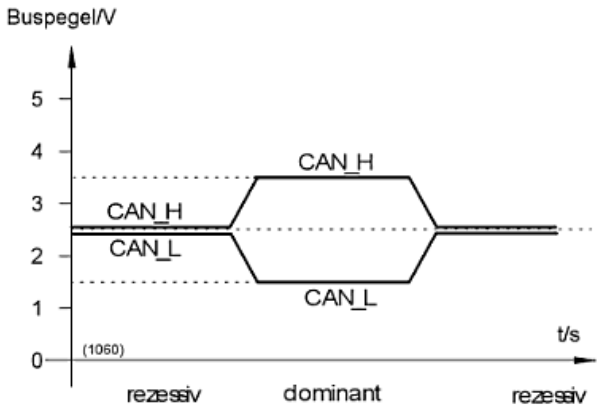


Abbildung 8-3: Spannungspegel beim High-Speed-CAN. Entnommen aus [22].

F.3. Frames

Folgende vier Frames werden beim CAN unterschieden:

- Data Frame
- Error Frame
- Remote Frame
- Overload Frame

Zur Nachrichtenübertragung wird nur der Data Frame verwendet. Die anderen Frames werden für die Fehlerbehandlung, Synchronisation und Triggerung verwendet. Jede Nachricht wird in Felder mit definierter Länge unterteilt und ist wie in Abbildung 8-4 aufgebaut [19].

Im folgendem werden die wichtigsten Felder des Data Frames genauer erläutert [19]:

Start-of-Frame: Signalisiert den Anfang der Übertragung.

Arbitrierungsfeld: Dient zur logischen Adressierung und Priorisierung von Nachrichten. Je niedriger der numerische Wert desto höher ist die Priorität.

Steuerfeld: Dient zur Angabe des Data Length Code (DLC), welche die Länge des nachfolgenden Datenfelds angibt.

Datenfeld: Beinhaltet die zu übertragende Nachricht und ist 0 - 8 Bytes lang.

CRC-Feld: Stellt die Prüfsumme dar, welche zur Fehlererkennung eingesetzt wird.

ACK-Feld: Signalisiert dem Sender, dass die Nachricht von mindestens einem Empfänger korrekt empfangen wurde. Die Nachricht wird solange wiederholt gesendet, bis die Nachricht korrekt übertragen wurde.

End-of-Frame-Feld: Signalisiert das Ende des aktuellen Frames.

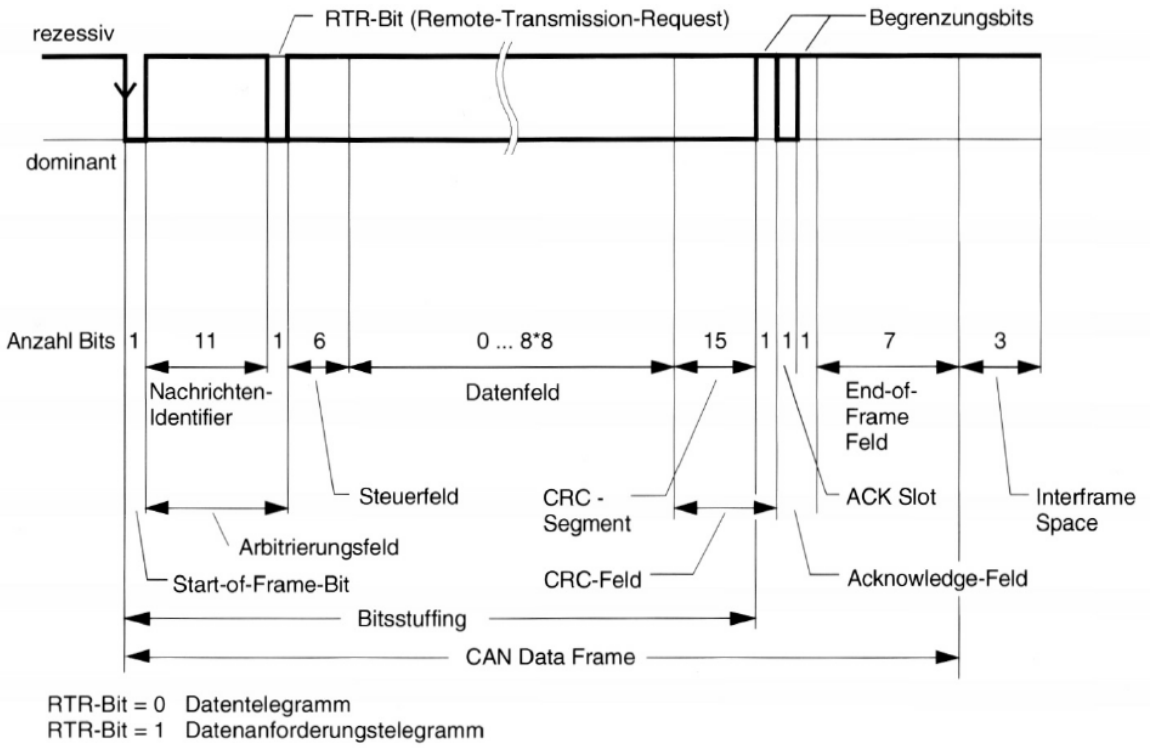


Abbildung 8-4: Standardformat des Daten-Frames. Entnommen aus [24].

F.4. CANOpen

Da CAN nur die ersten zwei Schichten im OSI-Referenzmodell spezifiziert, wurden für komplexere Aufgaben verschiedene Implementierungen der Applikationsschicht (7. Schicht) eingeführt. Eine Implementierung stellt das CANOpen Kommunikationsprotokoll dar und wird von der Organisation CiA (CAN in Automation) verwaltet [23]. Die verwendeten Motion-Controller kommunizieren mit CANOpen-Protokollen nach dem Geräteprofil CiA 402 (elektrische Antriebe).

> „Ein Geräteprofil ist eine Sammlung von Eigenschaften und Funktionen, über die das jeweilige Gerät verfügt."
>
> Zitat aus [23].

Dieses spezifiziert zum Beispiel die Bewegungserstellung mittels Profile Position Mode (siehe Kapitel 3.2.1.2)

CANOpen stellt folgende Funktionalitäten zur Verfügung:

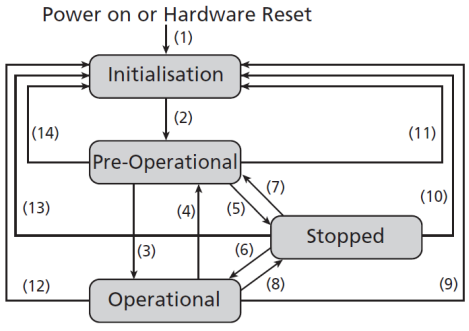
PDO (Prozessdatenobjekte): PDOs dienen zur Übertragung von Prozessdaten für die Steuerung oder Überwachung des Gerätes. Zum Beispiel kann damit der Status der Zustandsmaschine der Antriebseinheit geändert oder ausgelesen werden. Es kann auch auf die aktuellen Positions- und Stromwerte zugegriffen werden. Aufgrund der kleinen Datenmengen und des kritischen Zeitverhaltens besitzen PDOs eine hohe CAN-Botschaftspriorität beziehungsweise geringe Arbitration-IDs [23].

SDO (Servicedatenobjekt): SDOs dienen zum Lesen und Beschreiben des Objektverzeichnisses. Über das Objektverzeichnis können die Konfigurationsparameter der Motion-Controller verwaltet werden wie zum Beispiel: Drehrichtung, Fehlerverhalten, Referenzpunkverschiebungen, etc. [20]. Die Übertragungsmengen sind verhältnismäßig größer als bei PDOs und das Zeitverhalten unkritisch. Deshalb werden niedrigere CAN-Botschaftsprioritäten beziehungsweise höhere Arbitration-IDs verwendet [23].

EMCY (Emergency Objekt): Das Emergency Objekt dient dazu, anderen Busteilnehmern über einen aufgetretenen Fehler zu informieren. Des Weiteren beinhaltet das EMCY einen Fehlercode, welcher zur Identifizierung des entsprechenden Fehlerzustandes dient. Zum Beispiel werden bei Überspannung, Übertemperatur, Dauerüberschreitung des Ausgangsstromes, Kommunikationsfehler etc. Emergency Objekte gesendet. Die CAN-Botschaftspriorität ist höher als bei SDOs und PDOs. [20]

SYNC (Synchronisationsobjekt): Das SYNC Objekt dient zum Synchronisieren von PDOs. Zum Beispiel wird der zuvor mittels PDO gesendete Befehl erst nach Erhalt des entsprechenden SYNC Objekts ausgeführt, falls die entsprechenden Parameter im Objektverzeichnis eingestellt wurden. [20]

NMT (Netzwerkmanagement): NMT-Objekte dienen zur Steuerung der Kommunikation-Zustandsmaschine eines Gerätes. Je nach Zustand werden Nachrichten-Objekte angenommen oder abgelehnt. Zum Beispiel werden PDOs nur im Zustand „Operational“ und SDOs nur in den Zuständen „Operational“ und „Pre-Operational“ von den Geräten akzeptiert. Um vom Zustand „Pre-Operational“ zu „Operational“ zu wechseln, muss ein NMT-Objekt mit dem Befehl „Start Remote Node“ übermittelt werden. In Abbildung 8-5 ist die Zustandsmaschine eines CANOpen-Gerätes dargestellt. Das NMT ist auch für das Node-Guarding zuständig, welches in Kapitel 4.5.3.2 näher beschrieben wird. [20]



(1)	At Power on the initialisation state is entered autonomously
(2)	Initialisation finished – enter PRE-OPERATIONAL automatically
(3), (6)	Start_Remote_Node indication
(4), (7)	Enter PRE-OPERATIONAL_State indication
(5), (8)	Stop_Remote_Node indication
(9), (10), (11)	Reset_Node indication
(12), (13), (14)	Reset_Communication indication

Abbildung 8-5: Zustandsmaschine eines CANOpen-Gerätes. Entnommen aus [20].

G. Weitere Abbildungen

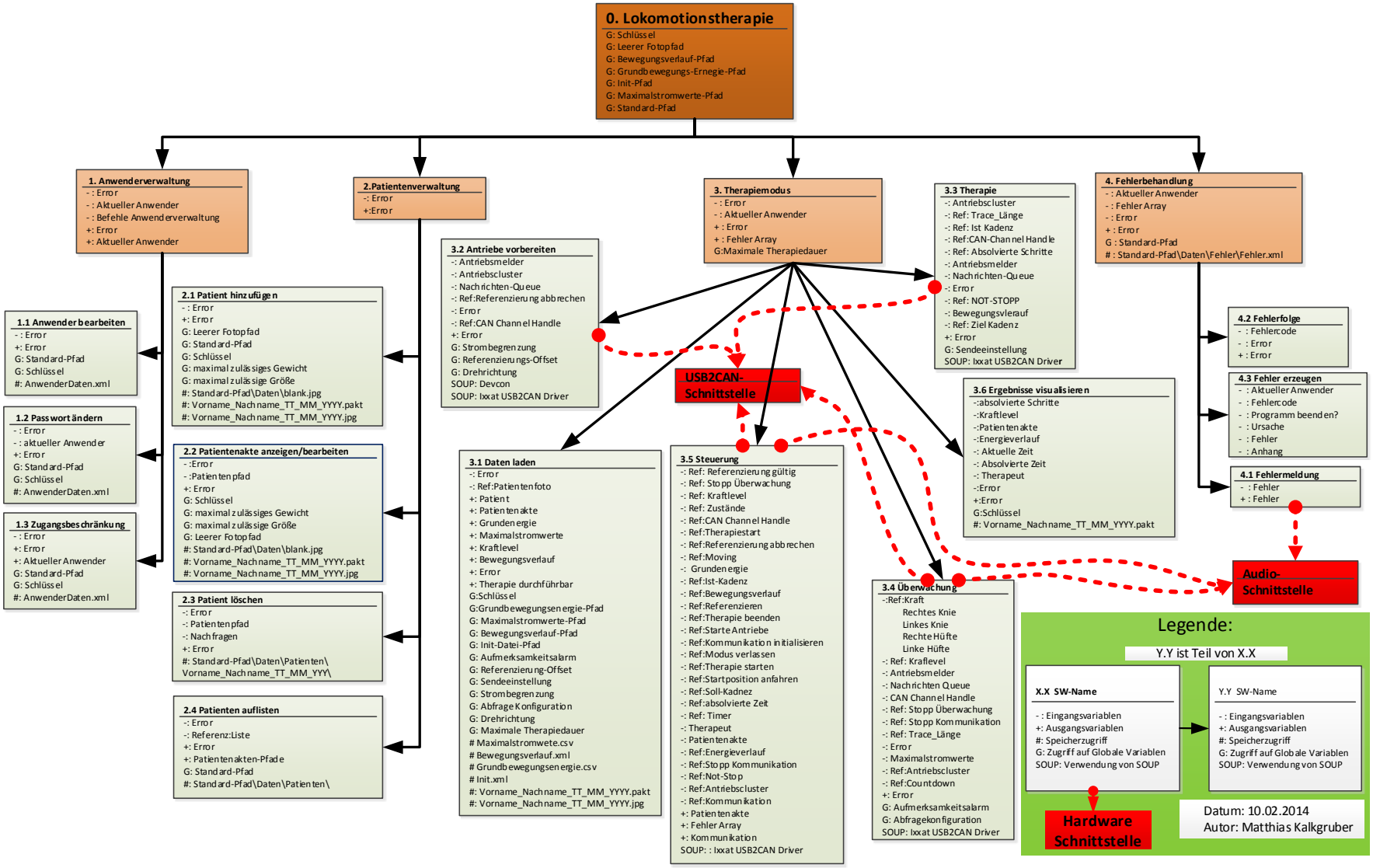


Abbildung 8-6: Hierarchiediagramm der Software-Architektur.

H. Weitere Tabellen

Tabelle 17: Abkürzungen des Traceability-Dokuments.

Abkürzung	Bedeutung
Ident.	Identifizierung
SysA	System-Anforderung
SWA	Software-Anforderung
SW-Komp.	Software-Komponente
RA-V2	Risikoanalyse-Version 2
MD	Maschinendirektive
SST	Software-System-Test
E	Ergebnis
P	Pass
F	Fail

Tabelle 18: Kenndaten des DC-Kleinstmotor 3863 024CR. Entnommen aus [13].

Bezeichnung	Faulhaber DC-Kleinstmotor 3863 024CR
Versorgungsspannung	24 V DC
Thermisch zulässiger Dauerstrom	4,8 A
Dauerdrehmoment	157 mNm
Leerlaufdrehzahl	5800 U/min
Max. Drehzahl	8000 U/min
Gewicht	390 g

Tabelle 19: Kenndaten des Impulsgebers HEDS 5500 A-12. Entnommen aus [17].

Bezeichnung	Optischer Impulsgeber HEDS 5500 A-12
Impulse pro Umdrehung	500
Kanäle	2
Betriebsspannung in V DC	4,55 … 5,5
Frequenzbereich in kHz	bis 100
Drehzahl in U/min	bis 12 000
Signalphasenverschiebung, Kanal A zu B in °	90

Tabelle 20: Grenzwerte für die Lokomotionstherapie. Entnommen aus [8].

Kenndaten	Einheit	Grenzwert
minimale Körpergröße	cm	67
maximale Körpergröße	cm	90
minimale Oberschenkellänge	cm	16
maximale Oberschenkellänge	cm	22
minimale Unterschenkellänge	cm	19
maximale Unterschenkellänge	cm	27