**Lastenheft**

**Schaltsysteme-Arbeitsblätter im Netz: SANE**

**Version 0.3**

Historie der Dokumentversionen

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Änderungsgrund / Bemerkungen |
| 0.1 | 03.04.2018 | David Sukiennik | Ersterstellung |
| 0.2 | 04.04.2018 | David Sukiennik  René Hutschenreuter | 1. Überarbeitung |
| 0.3 | 12.04.2018 | David Sukiennik  René Hutschenreuter | 2. Überarbeitung |
|  |  |  |  |

Inhaltsverzeichnis

[Historie der Dokumentversionen 2](#_Toc511301190)

[Inhaltsverzeichnis 3](#_Toc511301191)

[1 Einleitung 5](#_Toc511301192)

[1.1 Allgemeines 5](#_Toc511301193)

[1.1.1 Zweck und Ziel dieses Dokuments 5](#_Toc511301194)

[2 Konzept und Rahmenbedingungen 6](#_Toc511301195)

[2.1 Ziele des Anbieters 6](#_Toc511301196)

[2.2 Benutzer / Zielgruppe 6](#_Toc511301197)

[2.3 Übersicht der Meilensteine 6](#_Toc511301198)

[3 Beschreibung der funktionalen Anforderungen 7](#_Toc511301199)

[3.1 Anforderung 1: Erweiterung von und Einbettung in SANE 7](#_Toc511301200)

[3.1.1 Beschreibung 7](#_Toc511301201)

[3.2 Anforderung 2: Wertetabelle 7](#_Toc511301202)

[3.2.1 Beschreibung 7](#_Toc511301203)

[3.3 Anforderung 3: Karnaugh-Veitch-Diagramm 7](#_Toc511301204)

[3.3.1 Beschreibung 7](#_Toc511301205)

[3.4 Anforderung 4: Boolesches Mengendiagramm 7](#_Toc511301206)

[3.4.1 Beschreibung 7](#_Toc511301207)

[3.5 Anforderung 5: Boolesche Ausdrucksalgebra 7](#_Toc511301208)

[3.5.1 Beschreibung 8](#_Toc511301209)

[3.6 Anforderung 6: Funktionsindizes 8](#_Toc511301210)

[3.6.1 Beschreibung 8](#_Toc511301211)

[3.7 Anforderung 7: Quine-McCluskey 8](#_Toc511301212)

[3.7.1 Beschreibung 8](#_Toc511301213)

[3.8 Anforderung 8: Programmierbare Strukturen 8](#_Toc511301214)

[3.8.1 Beschreibung 8](#_Toc511301215)

[3.9 Anforderung 9: Funktions-Hasards 8](#_Toc511301216)

[3.9.1 Beschreibung 8](#_Toc511301217)

[3.10 Anforderung 10: Schaltungen im BEAST-Format 8](#_Toc511301218)

[3.10.1 Beschreibung 8](#_Toc511301219)

[3.11 Anforderung 11: Struktur-Hasards 9](#_Toc511301220)

[3.11.1 Beschreibung 9](#_Toc511301221)

[3.12 Anforderung 12: Separierung einer Event-Klasse 9](#_Toc511301222)

[3.12.1 Beschreibung 9](#_Toc511301223)

[3.13 Anforderung 13: Modul für boolesche Operationen 9](#_Toc511301224)

[3.13.1 Beschreibung 9](#_Toc511301225)

[3.14 Anforderung 14: Export und Import des Anwendungszustandes 9](#_Toc511301226)

[3.14.1 Beschreibung 9](#_Toc511301227)

[3.15 Anforderung 15: Dual-Pane-Modus 9](#_Toc511301228)

[3.15.1 Beschreibung 9](#_Toc511301229)

[3.16 Anforderung 16: Nutzung von importHelpers 10](#_Toc511301230)

[3.16.1 Beschreibung 10](#_Toc511301231)

[3.17 Anforderung 17: Modulare Integrierbarkeit in moodle 10](#_Toc511301232)

[3.17.1 Beschreibung 10](#_Toc511301233)

[3.18 Anforderung 18: Support von g-Parametern 10](#_Toc511301234)

[3.18.1 Beschreibung 10](#_Toc511301235)

[3.19 Anforderung 19: Progressive Web Applikation 10](#_Toc511301236)

[3.19.1 Beschreibung 10](#_Toc511301237)

[4 Beschreibung der nicht-funktionalen Anforderungen 11](#_Toc511301238)

[4.1 Anforderung 1: Polymer 11](#_Toc511301239)

[4.1.1 Beschreibung 11](#_Toc511301240)

[4.2 Anforderung 2: Material Design 11](#_Toc511301241)

[4.2.1 Beschreibung 11](#_Toc511301242)

[4.3 Anforderung 3: Responsives Design 11](#_Toc511301243)

[4.3.1 Beschreibung 11](#_Toc511301244)

[4.4 Anforderung 4: Datenfluss 11](#_Toc511301245)

[4.4.1 Beschreibung 11](#_Toc511301246)

[4.5 Anforderung 5: TypeScript 11](#_Toc511301247)

[4.5.1 Beschreibung 11](#_Toc511301248)

[4.6 Anforderung 6: Moderne Plattformfunktionen 11](#_Toc511301249)

[4.6.1 Beschreibung 12](#_Toc511301250)

[4.7 Anforderung 7: Linting 12](#_Toc511301251)

[4.7.1 Beschreibung 12](#_Toc511301252)

[4.8 Anforderung 8: Performanz 12](#_Toc511301253)

[4.8.1 Beschreibung 12](#_Toc511301254)

[4.9 Anforderung 9: Aktualisieren der 3rd-party-libs und Portieren auf aktuelle Versionen? 12](#_Toc511301255)

[4.9.1 Beschreibung 12](#_Toc511301256)

# Einleitung

Dieses Kapitel gibt einen Überblick über das Ziel des Projektes SANE: „Schaltsysteme-Arbeitsblätter im Netz“.

## Allgemeines

Ziel ist es die Begleitung und Unterstützung des Lehrbetriebes in den Fächern Technische Informatik, Rechnerorganisation und Schaltsysteme durch interaktive Applets. Dazu sind wichtige Grundelemente zur Darstellung boolescher Funktionen, wie Wertetabellen, Mengendiagramme, boolesche Gleichungen, Funktionsindizes, KDNF, KKNF, Karnaugh-Veitch-Diagramm, Hasards, Quine-McCluskey, ROM, PLA, GAL und evtl. g-Parameter in einem Online-Tool zur Verfügung zu stellen.

### Zweck und Ziel dieses Dokuments

Zweck dieses Dokumentes ist die Beschreibung der Anforderungen an das Softwareprojekt.

# Konzept und Rahmenbedingungen

In diesem Kapitel werden die Rahmenbedingungen und terminlichen Pläne für das Softwareprojekt beschrieben.

## Ziele des Anbieters

Ziel ist es zum Ende des Sommersemesters 2018 ein getestetes und einsatzbereites Tool zur Verfügung zu stellen.

## Benutzer / Zielgruppe

Studenten der TU Ilmenau.

## Übersicht der Meilensteine

|  |  |
| --- | --- |
| **Vorbereitungsphase** | |
| Überprüfung Lastenheft | März 2018 |
| Freigabe Lastenheft | April 2018 |
| **Grobentwurf** | |
| Erstellung Pflichtenheft / Grobentwurf | April 2018 |
| Präsentation Pflichtenheft |  |
| Freigabe Pflichtenheft |  |
| **Feinentwurf** | |
| Erstellung des Feinentwurfs |  |
| Präsentation des Feinentwurfs |  |
| Freigabe des Feinentwurfs |  |
| **Implementierung und Test** | |
| Implementierung |  |
| Test |  |
| Abschlusspräsentation | Juli 2018 |
| Freigabe und Bewertung | Juli 2018 |

# Beschreibung der funktionalen Anforderungen

Im folgenden Kapitel werden die einzelnen Anforderungen an das Projekt gestellt.

## Anforderung 1: Erweiterung von und Einbettung in SANE

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_01 | **Nichttechnischer Titel** | Erweiterung von und Einbettung in SANE |

### Beschreibung

Der bestehende Stand von SANE soll erweitert werden. Dazu werden verschiedene Darstellungen einer zentralen booleschen Funktion ermöglicht. Die Darstellung lässt sich über die Navigation auswählen. Hinzugefügte Darstellungen müssen auch in die Navigation eingebettet werden.

## Anforderung 2: Wertetabelle

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_02 | **Nichttechnischer Titel** | Wertetabelle |

### Beschreibung

Eine skalierbare Wertetabelle mit konfigurierbar vielen Eingangs- und Ausgangsvariablen sowie der Einbeziehung von H\*-Ausdrücken und toggelbaren Ausgängen soll entworfen werden. Die Zeilenindizes sollen auf der linken Seite der Wertetabelle angezeigt werden.

## Anforderung 3: Karnaugh-Veitch-Diagramm

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_03 | **Nichttechnischer Titel** | Karnaugh-Veitch-Diagramm |

### Beschreibung

Die zentrale boolesche Funktion soll in einem Karnaugh-Veitch-Diagramm (KV-Diagramm) dargestellt werden. Es sollen an dieser Stelle bis zu 6 Eingangsvariablen unterstützt werden. Ein Klick oder Tap auf ein Bit togglet dieses zwischen 1, 0 und \*. Per Drag-and-drop, Shift-Klick und Long-Press Tap lassen sich Bits zur Vereinfachung des Ausdrucks gruppieren. Weiterhin soll eine automatische Bildung der maximalen Blöcke in allen Varianten zur Ermittlung **aller** Minimalterme möglich sein. Grundlage für diese Berechnung stellt die Verwendung von Quine-McCluskey dar.

## Anforderung 4: Boolesches Mengendiagramm

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_04 | **Nichttechnischer Titel** | Boolesches Mengendiamm |

### Beschreibung

Per Drag-and-drop lassen sich Werte zwischen den vorhandenen Mengen verschieben. Bei mehr als drei verschiedenen Ausgangsvariablen wählt der Nutzer die drei anzuzeigenden aus. Die Anzahl der Elemente kann anhand der Anzahl der Eingangsvariablen bestimmt und verändert werden. Für die Übersichtlichkeit sollen maximal 4 Eingangsvariablen und 16 Elemente entsprechend der Belegung verwendet werden können.

## Anforderung 5: Boolesche Ausdrucksalgebra

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_05 | **Nichttechnischer Titel** | Boolesche Ausdrucksalgebra |

### Beschreibung

Die zentrale boolesche Funktion wird in KDNF und KKNF sowie minimalen KNF und DNF dargestellt. **Alle** minimalen Ausdrücke sollen ebenfalls durch QMC berechnet werden können. Eine Änderung des angezeigten Ausdrucks mit allen Symbolen incl. Äquivalenz, XOR und Implikation ändert auch den zentralen booleschen Funktionen. Die dazu verwendeten Zeichen für Konjunktion, Disjunktion, Negation, Äquivalenz, Antivalenz und Implikation sollen konfigurierbar sein.

## Anforderung 6: Funktionsindizes

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_06 | **Nichttechnischer Titel** | Funktionsindizes |

### Beschreibung

Die Darstellung und Eingabe als Funktionsindizes soll hinzugefügt werden. Die Ausdrücke werden dabei durch Hexadezimalzahlen bestimmt.

## Anforderung 7: Quine-McCluskey

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_07 | **Nichttechnischer Titel** | Quine-McCluskey |

### Beschreibung

Für die spezielle Ansicht des QMC-Algorithmus sollen alle Implikanten und Kürzungstabellen einschließlich Markierungen während der Minimierung angezeigt werden, sowie die Liste der Primimplikanten und der ausgewählten Implikanten für die endgültigen minimierten Terme. Der QMC-Algorithmus soll in ein wiederverwendbares Modul ausgelagert werden.

## Anforderung 8: Programmierbare Strukturen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_08 | **Nichttechnischer Titel** | Programmierbare Strukturen |

### Beschreibung

Es sollen für die Darstellung ROM, PLA und GAL jeweils programmierbare und editierbare Strukturen als Ansichten der dahinterliegenden booleschen Funktion entwickelt werden. Durch Klick auf die einzelnen programmierbaren Verbindungen werden diese getoggelt.

## Anforderung 9: Funktions-Hasards

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_09 | **Nichttechnischer Titel** | Funktions-Hasards |

### Beschreibung

Funktions-Hasards sollen erkannt und im wiederverwendeten KV-Diagramm dargestellt werden.

## Anforderung 10: Schaltungen im BEAST-Format

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_10 | **Nichttechnischer Titel** | Schaltungen im BEAST-Format |

### Beschreibung

Einbindung von Teilen des letzten Softwareprojektes BEAST **ODER** Export von Dateien im BEAST-Format und automatischer Import über den „Local Storage“ des Browsers zur Anzeige der generierten Schaltung aus den eingegebenen und berechneten booleschen Funktionen soll ermöglich werden.

## Anforderung 11: Struktur-Hasards

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_11 | **Nichttechnischer Titel** | Struktur-Hasards |

### Beschreibung

Es sollen Struktur-Hasards innerhalb der gefundenen Blöcke im KV-Diagramm ermittelt werden, sowie deren Vermeidung. Für die Anzeige und Simulation der Schaltung aller Blöcke incl. der redundanten Blöcke soll die in Anforderung SANE\_FA\_10 implementierte Schnittstelle zum BEAST verwendet werden.

## Anforderung 12: Separierung einer Event-Klasse

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_12 | **Nichttechnischer Titel** | Separierung einer Event-Klasse |

### Beschreibung

Zur Kommunikation von Änderungen des Anwendungszustandes (sog. Actions) senden Module Events, die an das oberste Modul übergeben werden, von dem sie verarbeitet werden. Bisher sind diese Actions statisch in die sie nutzenden Klassen verwoben. Sie sollen separiert werden, damit sie in neuen Klassen wiederverwendet werden können.

## Anforderung 13: Modul für boolesche Operationen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_13 | **Nichttechnischer Titel** | Modul für boolesche Operationen |

### Beschreibung

Sämtliche Funktionen, Methoden und Operationen, die allgemeingültig und mehrfach benötigt werden, besonders für boolesche Operationen, sind in mindestens ein ES6-Modul auszulagern.

## Anforderung 14: Export und Import des Anwendungszustandes

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_14 | **Nichttechnischer Titel** | Export und Import des Anwendungszustandes |

### Beschreibung

Die zentrale boolesche Funktion soll exportiert und importiert werden können.

## Anforderung 15: Dual-Pane-Modus

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_15 | **Nichttechnischer Titel** | Dual-Pane-Modus |

### Beschreibung

Falls das Browser-Fenster groß genug ist, soll ein Dual-Pane-Modus aktiviert werden. Dabei werden links und rechts in je eigenen Teilen der Anwendungsoberfläche (»Pane«) zwei verschiedene Darstellungen derselben zentralen booleschen Funktion gezeigt. Die Änderung in einem Pane werden sofort auch in der entsprechenden Darstellung im anderen Pane umgesetzt. Mittig zwischen den Panes ist die Navigation dargestellt, die die Auswahl der Darstellung für die beide Panes ermöglicht.

## Anforderung 16: Nutzung von importHelpers

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_16 | **Nichttechnischer Titel** | Nutzung von importHelpers |

### Beschreibung

Um duplizierten Code der von TypeScript benötigten Funktionen zu vermeiden, soll TypeScript mit der Compiler-Option importHelpers ausgeführt und die Laufzeitbibliothek tslib einmalig geladen werden.

## Anforderung 17: Modulare Integrierbarkeit in moodle

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_17 | **Nichttechnischer Titel** | Modulare Integrierbarkeit in moodle |

### Beschreibung

Die SANE-Umgebung stellt ein großes Spektrum an verschiedenen Darstellungsformen der gleichen kombinatorischen booleschen Logik dar, die jeweils in Bezug auf eine oder mehrere Ausgangsgleichungen (y0, …, yn) ineinander überführt werden können. Perspektivisch gesehen soll die SANE-Umgebung als eigenständiges „kleines“ Applet in moodle in einen I-Frame innerhalb des TI/RO-Kurses an verschiedenen Stellen in verschiedener Konfiguration zusammen mit speziell dafür konzipierten Aufgaben integriert werden.

Studenten bekommen in einer View das Problem mitgeteilt und müssen es dann in die andere Darstellungsform überführen und manuell die Ergebnisse dort eintragen. D.h. für möglichst alle Views soll ein automatisches und ein manuelles Eintragen mit automatischer Überprüfung der Eingabe möglich sein. Somit soll man SANE über entsprechende Initialisierungsparameter leicht konfigurierbar für ganz spezielle Aufgabenstellungen nutzen können. Nach Lösung der Aufgaben durch die Studierenden soll eine Rückmeldung sowohl als Popup-Nachricht mit Korrekturen als auch in Form von Logs für andere Systeme generiert werden.

## Anforderung 18: Support von g-Parametern

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_18 | **Nichttechnischer Titel** | Support von g-Parametern |

### Beschreibung

G-Parameter an sich nehmen sehr starken Einfluss auf alle Views, die zugrundeliegende globale Datenstruktur und die allgemeine Funktion des SANE-Projektes. Als Ausblick soll in der aktuellen Implementierung eine spätere Erweiterung um g-Parameter als umschaltbare Option berücksichtigt werden. Tabellen sollen eine Eingabe von mehreren möglichen Ausgängen und deren direkte Auswahl als aktiver Ausgang, sowie Karnaugh-Veitch-Diagramme die Eingabe eines logische g-Ausdruckes in eines der Felder etc. ermöglichen. Dieses Feature soll in diesem Softwareprojekt an sich noch nicht umgesetzt aber vorbereitet und bedacht werden.

## Anforderung 19: Progressive Web Applikation

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_FA\_16 | **Nichttechnischer Titel** | Nutzung von importHelpers |

### Beschreibung

Die Seite soll sich auf Smartphones und Tablet als offline-verwendbare Single-Site Applikation installieren lassen (Beispiel: <https://www.pokedex.org/>)

# Beschreibung der nicht-funktionalen Anforderungen

## Anforderung 1: Polymer

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_01 | **Nichttechnischer Titel** | Polymer |

### Beschreibung

Die zu entwickelnden Elemente werden mit Hilfe der JavaScript-Bibliothek Polymer implementiert. Die zu entwickelnden Komponenten werden als Web Components modularisiert und die endgültige Funktion über die Komposition bestehender und neu entwickelter Web Components erfüllt.

## Anforderung 2: Material Design

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_02 | **Nichttechnischer Titel** | Material Design |

### Beschreibung

Angelehnt an die wiederverwendeten Elemente aus der Polymer-Bibliothek rezipieren die neu entworfenen Elemente das Material Design.

## Anforderung 3: Responsives Design

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_03 | **Nichttechnischer Titel** | Responsives Design |

### Beschreibung

Alle Erweiterungen sollen ein responsives Design umsetzen. Das bedeutet, dass sie sich ebenso gut auf einem Desktop-Rechner wie auf einem Tablet oder Smartphone benutzen lassen. Das beinhaltet das Design ebenso wie die Reaktion auf Maus/Tastatur und Touchscreens.

## Anforderung 4: Datenfluss

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_04 | **Nichttechnischer Titel** | Datenfluss |

### Beschreibung

Der Datenfluss ist strikt unidirektional. Das bestehende Datenmodell wird genutzt und ggf. erweitert. Sämtliche Kommunikation findet von den Submodulen mittels Events statt. Änderungen des Anwendungszustandes werden mittels das Data-Bindings aus Polymer an die Submodule übergeben.

## Anforderung 5: TypeScript

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_05 | **Nichttechnischer Titel** | TypeScript |

### Beschreibung

Sämtlicher JavaScript-Code ist in TypeScript zu entwickeln, damit Erweiterungen und Refactoring erleichtert werden. TypeScript bildet vor allem eine typisierte Obermenge von JavaScript. Alle Klassen, Methoden, Attribute und sonstige Variablen sollen typisiert sein.

## Anforderung 6: Moderne Plattformfunktionen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_06 | **Nichttechnischer Titel** | Moderne Plattformfunktionen |

### Beschreibung

Der Quelltext soll in modernem JavaScript und mit aktuellem CSS entwickelt werden. Das entspricht ECMAScript 6 (auch ECMAScript 2015 oder ES6 genannt) und CSS 3. Mittels TypeScript können und sollen auch neuere JavaScript-Funktionen genutzt werden, da sie browser-kompatibel compiliert werden. Als Referenzplattform fungiert die aktuelle Version des Browsers Chrome. Bei der Entwicklung soll die Verwendung von JQuery explizit vermieden werden.

## Anforderung 7: Linting

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_07 | **Nichttechnischer Titel** | Linting |

### Beschreibung

Das in der Anwendung bestehende Linting soll genutzt werden und alle Regeln sollen erfüllt werden. Damit werden einerseits ein einheitlicher Stil und andererseits ein qualitativer Standard gewährleistet.

## Anforderung 8: Performanz

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_08 | **Nichttechnischer Titel** | Performanz |

### Beschreibung

Die Anwendung muss performant sein, um auch auf verbreiteten Smartphones schnell ausgeführt zu werden. Ergebnisse von aufwendigen Berechnungen sollen hierbei zwischengespeichert werden, um unnötige Rechenoperationen zu vermeiden.

## Anforderung 9: Aktualisieren der 3rd-party-libs und Portieren auf aktuelle Versionen?

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr. / ID** | SANE\_NFA\_09 | **Nichttechnischer Titel** | Aktualisieren der 3rd-party-libs und Portieren auf aktuelle Versionen? |

### Beschreibung

Polymer 2.6 wurde am 22. März publiziert. Polymer 3 wird in den kommenden Wochen publiziert. Der bisherige Stand muss auf den aktuellen Stand aller 3rd-party-libraries gebracht werden und ggf. auf neue APIs portiert werden.