System Test Plan

(Systemtest Plan)

(TINF17C, SWE I Praxisprojekt 2018/2019)

Project: *Profinet DCP Client*

Customer: *Rentschler & Ewertz*
Rotebühlplatz 41
70178 Stuttgart

Supplier: Team 2 (Rene Scholz, Sinan Yurttadur, Nicolas Breuninger, Noah Broß, Jannik Schwarz, Marvin Sonntag)
Rotebühlplatz 41
70178 Stuttgart

Version	Date	Author	Comment
0.1	10.05.2020	Jannik Schwarz	Created document, filled out the general information
1.0	14.05.2020	Rene Scholz	Added all test cases
1.1	14.05.2020	Rene Scholz	Added information about test cases

1. Contents

2. Scope

The STP (System Test Plan) specifies the test strategy and test planning. It references tests to be performed to verify the accordance of the demanded features given by the SRS (System Requirements Specification) to the implemented features. The document derived from the STP is the STR (System Test Report) where additionally the results are given.

Der Systemtestplan spezifiziert die Teststrategie und den Testumfang zur Verifikation des Pflichtenheftes (SRS). Testfälle werden referenziert. Er bildet die Basis für den Systemtestbericht (STR) der zusätzlich noch die Testergebnisse auflistet.

3. Definitions

TC Testcase (Testfall)
TS Testsuite (Gruppierung von Testfällen)

4. Product Names and Attributes

The following test objects must be verified:

Ref.-Id.	Product Number	Product Name	Product Description
1	Build 1.0	Profinet DCP Client	The application we developed over this year
2	Version 1909 x64	Windows 10	The operating system

5. Features

Die aufgelisteten Anforderungen müssen verifiziert werden, sofern sie nicht als "not to be tested" klassifiziert sind. Die Tabelle bildet zusätzlich die Testabdeckung zwischen Funktionalitäten und Testsuiten bzw. Testfällen ab.

Req. - ID	Functionality	Prio	Testsuite ID
UC-001: Search devices	Suchen der Geräte im Netzwerk	A	TS-001: frontend
UC-002: Display detailed data of Device	Anzeigen der Geräteinformationen	A	TS-001: frontend
UC-003: Save changes to device configuration	Speichern der Konfiguration	B	TS-002: backend

6. Test Preparation Strategy

Es bietet sich an, die Testfallerstellung anwendungsfallbasiert durchzuführen. Für jeden Basis-Anwendungsfall wird eine Testsuite mit der notwendigen Anzahl von Testfällen erstellt bis der Anwendungsfall aus Black-Box-Sicht vollständig abgedeckt ist. Dann wird die Abdeckung aller anderen im Pflichtenheft aufgeführten Anforderungen durch diese Testsuiten geprüft. Für dann immer noch nichtabgedeckte Anforderungen müssen weitere Testsuiten/Testfälle entworfen werden, bis eine vollständige Anforderungsabdeckung in der Tabelle im Kapitel 5 nachgewiesen werden kann.

Durch die Vielzahl der Eingabeparameter des Testobjekts und der daraus entstehenden kombinatorischen Explosion möglicher Testdatensätze ist der Einsatz von Äquivalenzklassenmethode, Grenzwertanalyse sowie Klassifikationsbaummethode sinnvoll.

7. Test Execution Strategy

Da es sich um eine Software-Neuentwicklung handelt, ist ein vollständiger Test zwingend notwendig. Die Testdurchführung soll in folgende Phasen gegliedert werden:

1) Frontend tests wurden zuerst durchgeführt. Dabei sollte die Funktionalität für den Benutzer überprüft werden.
2) Es wird die Kommunikation zwischen Front- und Backend über die API getestet.
3) Es wird die Kommunikation des Backends und der Profinet-Geräte getestet.

8. Test Schedule and Budget

Die Frontend Tests wurden zeitgleich mit der Entwicklung durchgeführt. Beim Backend wurde nach der Implementierung der Kommunikation mit Frontend und lokalem Netzwerk getestet. Ein Budget musste nicht berechnet werden.

9. Test Planning

Die folgende Tabelle dient der Ressourcenplanung für die Testvorbereitungs- und Testdurchführungsphase.

Testsuite	Test objective	Testplan Creator	Testplan Reviewer	Tester
TS-001	GUI functionality. Consisting mostly of checks for displaying the correct values	Rene Scholz	Nicolas Breuninger	Rene Scholz
TS-002	Communication, Backend functionality. Consists of API route testing and result checking.	Rene Scholz	Noah Broß	Rene Scholz

10. References / Standards

[1] SRS TINF18C Profinet DCP Client

Appendix: Testcases

10.1. Testsuite <TS-001 frontend>

10.1.1. <TC-001-001> (should create)

Testcase ID:	**1**
Testcase Name:	*should create*
Req.-ID:	*LF20*
Test Setup:	*This test is used to make sure that the application creates successfully. This is usually the first basic test to make.*

Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**component**	**true**	**true**

10.1.2. <TC-001-002> (should have a title)

Testcase ID:	**2**
Testcase Name:	*Should have a title*
Req.-ID:	*LF20*
Test Setup:	*This test is used to make sure that the frontend title is set.*

Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**Component**	**Not null**	**Not null**

10.1.3. <TC-001-001> (should refresh)

Testcase ID:	**3**
Testcase Name:	*Should refresh*
Req.-ID:	*LF10, LF20,*
Test Setup:	*This test is used to make sure that the refresh button works like it should. After clicking on the refresh button the application should refresh the data from the backend and display it to the frontend.*

Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**click**		
2	**detectChanges**		
3	**refresh**		

10.1.4. <TC-001-004> (should open device info after click)

Testcase ID:	4		
Testcase Name:	*Should open device info after click*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that the slide "device information" after clicking on any device in the slide "devices".*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.5. <TC-001-005> (should have the correct title in device info)

Testcase ID:	5		
Testcase Name:	*Should have the correct title in device info*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that the title in the device information is set correctly.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.6. <TC-001-006> (should have device name)

Testcase ID:	6		
Testcase Name:	*Should have a device name*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a name.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.7. <TC-001-007> (should have device ip)

Testcase ID:	7		
Testcase Name:	*Should have a device ip*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a ip address.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.8. <TC-001-008> (should have a device mac adress)

Testcase ID:	8		
Testcase Name:	*Should have a decice mac adress*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a mac adress*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.9. <TC-001-009> (should have a device subnet mask)

Testcase ID:	9		
Testcase Name:	*Should have a device subnet mask*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a subnet mask*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChange		

10.1.10. <TC-001-010> (should have a vendor value)

Testcase ID:	10		
Testcase Name:	*Should have a vendor value*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a vendor value.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.1.11. <TC-001-011> (should have a device role)

Testcase ID:	11		
Testcase Name:	*Should have a device role*		
Req.-ID:	*LF20*		
Test Setup:	*This test makes sure that every device has a device role.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	click		
2	detectChanges		

10.2. Testsuite <TS-002 backend>

10.2.1. <TC-002-001> (should return 404 cause route not found)

Testcase ID:	12		
Testcase Name:	*Should return 404 cause route not found*		
Req.-ID:	*LF30*		
Test Setup:	*This is a GET Routing test for the backend routing.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**GET**	**404**	**404**

10.2.2. <TC-002-002> (should return 200 if refreshed)

Testcase ID:	13		
Testcase Name:	*Should return 200 if refreshed*		
Req.-ID:	*LF30*		
Test Setup:	*This is a GET request, it should return 200 after refreshing.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**GET**	**200**	**200**

10.2.3. <TC-002-003> (get device overview)

Testcase ID:	14		
Testcase Name:	*Get device overview*		
Req.-ID:	*LF30*		
Test Setup:	*This test makes sure that 200 is returned if a route works. Also it tests if the content is returned.*		
Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**GET**	**200**	**200**
2	**GET**	**true**	**true**

10.2.4. <TC-002-004> (should return 404 cause route not found)

Testcase ID:	**15**
Testcase Name:	*Detail view*
Req.-ID:	*LF30*
Test Setup:	*This test should return 404 if the device is not in the list. Also it should return 200 if the device is in list. After all that it should return the content if the device is in the list.*

Test Steps			
Step	**Action**	**Expected result**	**Actual Result**
1	**GET**	**404**	**404**
2	**GET**	**200**	**200**
3	**GET**	**true**	**true**