

Arbeitspackete

Studienarbeit FS-2020

31. März 2020

Autoren:

 $\begin{array}{l} Mike \; SCHMID \\ \text{mike.schmid@hsr.ch} \end{array}$

Janik SCHLATTER janik.schlatter@hsr.ch

Supervisors:

Prof. Stettler BEAT beat.stettler@hsr.ch

Baumann URS urs.baumann@hsr.ch

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.



Zweck

Dieses Dokument beschreibt die Arbeitspackete und deren Aufwandschätzung und Priorisierung.

Änderungsgeschichte

| Datum | Version | Änderung | Autor |
|------------|---------|--------------------------|-----------------|
| 31.03.2018 | 1.0 | Initial Setup | Janik Schlatter |
| 31.03.2020 | 1.0 | Erstellen Arbeitspackete | Mike Schmid |



Inhaltsverzeichnis

| 1 | Arbeitspackete | 1 |
|-------|--|---|
| 1.1 | Kategorie Ausführung der Tests | 1 |
| 1.1.1 | AP: Implementation Testrunner | 1 |
| 1.1.2 | AP: Implementation Evaluator | 1 |
| 1.1.3 | AP:Implementation Reporter | 2 |
| 1.1.4 | AP: Implementation Testresultat-Mapper | 2 |
| 1.1.5 | AP: Implementation Testcontroller | 2 |
| 1.2 | Kategorie Erstellen der Tests | 3 |
| 1.2.1 | AP: Implementation Testbuilder | 3 |
| 1.2.2 | AP: Auswahl der Testreihenfolge | 3 |
| 1.2.3 | AP: Inventar Management Klassen | 3 |
| 1.2.4 | AP:Testdefinition Klassen | 4 |
| 1.2.5 | AP: Connection-Strategy-Factory | 4 |
| 1.2.6 | AP: Testerstellung-Strategy-Factory | 4 |
| 1.2.7 | AP: Automatische Geräteerfassung | 5 |
| 1.3 | Kategorie Hilfsklassen | 6 |
| 1.3.1 | AP: FileHandler | 6 |
| 1.3.2 | AP: Logging | 6 |
| 1.3.3 | AP: Integration Tests | 6 |
| 1.3.4 | AP: Kommandozeilen-GUI | 7 |
| 1.4 | Kategorie Test-Integration | 8 |



1 Arbeitspackete

Die Arbeitspackete sind in fünf Kategorien eingeteilt, um eine Übersicht zu bieten.

- Ausführung der Tests Alles was mit der Ausführung und Evaluation der Netzwerktests zu tun hat.
- Erstellen der Tests Alle Arbeitspackete, die sich mit der Erstellung der Testsuite auseinandersetzen.
- Hilfsklassen Filehandler, Logging, Integration Tests und Kommandozeileninteraktion
- Tests Integration einzelner Tests in das System
- Connections Integration weiterer Verbindungsschnittstellen in das System

1.1 Kategorie Ausführung der Tests

1.1.1 AP: Implementation Testrunner

| Beschreibung | Die Testrunner-Klasse führt die ihm vom Controller übergebenen Tests gegen das Netzwerk aus und gibt ein Resultat in einem beliebigen Format zurück. |
|--------------------|---|
| Akzeptanzkriterien | Testrunner ist in der Lage, Tests gegen ein Netzwerk auszuführen und reagiert auf Fehler, indem fehlgeschlagene Tests als solche annotiert werden. Es existieren Unittests für die Testrunner-Klasse und bekannte Bugs sind als weitere Arbeitspackete erfasst. |
| Aufwandschätzung | 6 Stunden |
| Priorität | Mittel |

1.1.2 AP: Implementation Evaluator

| Beschreibung | Der Evaluator vergleicht die Resultate der Netzwerktests mit den aus den Testdefinitionen beschriebenen Soll-Werten. |
|--------------------|--|
| Akzeptanzkriterien | Der Evaluator ist in der Lage, die normalisierten Testresultate mit den Test- expectations zu vergleichen und ein verständliches Evaluationsresultat zu er- zeugen. Unittests sind erstellt und bekannte Bugs sind dokumentiert. |
| Aufwandschätzung | 10 Stunden |
| Priorität | Mittel |



1.1.3 AP:Implementation Reporter

| Beschreibung | Der Reporter nimmt die Gesamtergebnisse entgegen und gibt sie auf der Konsole in einem einfach verständlichen Format aus. Zudem schreibt er die Ergebnisse in ein Report-File, welches die Testresultate und allfällige Fehlermeldungen beinhaltet. |
|--------------------|--|
| Akzeptanzkriterien | Die Reporter-Klasse ist in der Lage, detaillierte Berichte zu den Ergebnissen zu verfassen und abzuspeichern. Unittests sind erstellt und laufen alle durch. Bekannte Bugs sind im Issue-Tracker erfasst. |
| Aufwandschätzung | 10 Stunden |
| Priorität | Niedrig |

1.1.4 AP: Implementation Testresultat-Mapper

| Beschreibung | Der Testresultatmapper normalisiert die Rückgabewerte der einzelnen Netzwerktests in ein einheitliches Format, so dass der Evaluator damit arbeiten kann. |
|--------------------|---|
| Akzeptanzkriterien | Die Normalform der Resultate ist festgelegt und implementiert. Unittests beenden mit Status "Grününd bekannte Bugs sind erfasst. |
| Aufwandschätzung | 12 Stunden |
| Priorität | Mittel |

1.1.5 AP: Implementation Testcontroller

| Beschreibung | Der Testcontroller ist der Einstiegspunkt in das Programm. Er instanziert die benötigten Klassen und steuert den Programmfluss. |
|--------------------|---|
| Akzeptanzkriterien | Testcontroller ist implementiert und das Programm lässt sich erfolgreich ausführen. Bekannte Bugs sind erfasst und Unittests sind erstellt und bestanden. |
| Aufwandschätzung | 4 Stunden |
| Priorität | Niedrig |



1.2 Kategorie Erstellen der Tests

1.2.1 AP: Implementation Testbuilder

| Beschreibung | Der Testbuilder übernimmt die Tests aus der Testdefinition und erstellt ein Testbundle mit den angegebenen Parameter. |
|--------------------|---|
| Akzeptanzkriterien | Der Testbuilder ist implementiert und erstellt ausführbare Tests. Unittests sind implementiert und bekannte Bugs sind dokumentiert. |
| Aufwandschätzung | 6 Stunden |
| Priorität | Mittel |

1.2.2 AP: Auswahl der Testreihenfolge

| Beschreibung | Erweiterung des Testbuilder um die Möglichkeit, die Reihenfolge der Tests zu bestimmen. |
|--------------------|--|
| Akzeptanzkriterien | Testdurchführungsreihenfolge lässt sich ändern, ohne dass die Unittests dabei durchfallen. |
| Aufwandschätzung | 6 Stunden |
| Priorität | Niedrig |

1.2.3 AP: Inventar Management Klassen

| Beschreibung | Zu diesem Arbeitspacket gehören die Klassen Inventar, Device und Device- Connection. Diese Klassen beschäftigen sich damit, die Informationen über die Physischen Geräte in einem Netzwerk festzuhalten. |
|--------------------|--|
| Akzeptanzkriterien | Klassen sind implementiert und die Zusammenarbeit funktioniert. Es lassen sich Tests mit den Gerätedaten vom TestBuilder erstellen und durchführen. |
| Aufwandschätzung | 10 Stunden |
| Priorität | Hoch |



1.2.4 AP:Testdefinition Klassen

| Beschreibung | Die Testdefinitionen beschreiben die auszuführenden Tests gegen das Netzwerk. Der Loader instanziert die Definitionen der einzelnen Tests und gibt diese als Collection an den Testbuilder weiter, der daraus konkrete Tests instanziert. |
|--------------------|---|
| Akzeptanzkriterien | Testdefinitionen lassen sich erstellen und ausführen. Unittests sind implementiert und laufen erfolgreich durch. Allfällige Bugs sind im Issue-Tracking dokumentiert. |
| Aufwandschätzung | 8 Stunden |
| Priorität | Hoch |

1.2.5 AP: Connection-Strategy-Factory

| Beschreibung | Dieser Teil der Software kümmert sich um die Auswahl und Instanzierung der Kommunikationsschnittstelle. |
|--------------------|---|
| Akzeptanzkriterien | Alle Klassen sind gemäss der Architektur implementieren und es lässt sich mindestens eine Schnittstelle auswählen und instanzieren. Tests sind erfolgreich und allfällige Bugs sind dokumentiert. |
| Aufwandschätzung | 20 Stunden |
| Priorität | Hoch |

1.2.6 AP: Testerstellung-Strategy-Factory

| Beschreibung | Dieser Teil des Programms entscheidet, welche konkreten Testklassen verwendet werden und instanziert diese. |
|--------------------|---|
| Akzeptanzkriterien | Alle Klassen sind gemäss der Architektur implementiert und es gibt mindestens eine konkrete Testklasse die sich ausführen lässt. Es existieren Unittests, welche erfolgreich durchlaufen und bekannte Bugs sind erfasst. |
| Aufwandschätzung | 20 Stunden |
| Priorität | Hoch |



1.2.7 AP: Automatische Geräteerfassung

| Beschreibung | Damit das Inventar nicht von Hand geführt werden muss, lassen sich die Geräteeinstellungen automatisch erfassen. |
|--------------------|---|
| Akzeptanzkriterien | Die Erfassung lässt sich erfolgreich durchführen, sie erfasst sämtliche Geräte im Netzwerksystem und speichert die Daten im YAML-Format. Unittests sind erstellt und laufen erfolgreich durch. Allfällige Bugs sind im Issue-Tracker erfasst. |
| Aufwandschätzung | 30 Stunden |
| Priorität | Niedrig |



1.3 Kategorie Hilfsklassen

1.3.1 AP: FileHandler

| Beschreibung | Der Filehandler wird immer dann aufgerufen, wenn etwas aus dem Filesystem geladen werden muss oder etwas darin geschrieben werden soll. |
|--------------------|---|
| Akzeptanzkriterien | Der Filehandler ist implementiert und es lassen sich damit Dokumente im YAML Format schreiben und lesen. |
| Aufwandschätzung | 6 Stunden |
| Priorität | Hoch |

1.3.2 AP: Logging

| Beschreibung | Der Logger ist dafür zuständig, dass alle Events, die in der Software geschehen, beispielsweise Exceptions oder Informations-Logs, persistent gespeichert werden. |
|----------------------------|---|
| ${\bf Akzeptanzkriterien}$ | Der Logger ist implementiert. |
| Aufwandschätzung | 6 Stunden |
| Priorität | Niedrig |

1.3.3 AP: Integration Tests

| Beschreibung | Mit den Integration Tests wird überprüft, ob die Softwarekomponenten alle miteinander zusammen agieren können. Sie dienen der Überprüfung des Systems. |
|--------------------|--|
| Akzeptanzkriterien | Integrationstests wurden systematisch und geplant durchgeführt und die Resultate dokumentiert. Allfällige Fehler wurden im Issue-Tracking erfasst. |
| Aufwandschätzung | 10 Stunden |
| Priorität | Niedrig |



1.3.4 AP: Kommandozeilen-GUI

| Beschreibung | Um mit dem Programm interagieren zu können, wird ein Grafikinterface be- |
|--------------------|---|
| | nötigt. |
| | Vorerst wird dieses mit einem Kommandozeilen-GUI umgesetzt. |
| Akzeptanzkriterien | Es ist möglich, die Erfassung, Orchestrierung und Durchführung über das GUI |
| | zu steuern. |
| | Usability-Tests wurden durchgeführt und daraus entstandene Verbesserungs- |
| | punkte aufgenommen. |
| Aufwandschätzung | 14 Stunden |
| Priorität | Mittel |



1.4 Kategorie Test-Integration

Die Testintegration befasst sich mit der Implementation der einzelnen Netzwerktests. Für die meisten Tests sollte ein Aufwand von vier bis acht Stunden ausreichend sein, wobei umfangreichere Tests durchaus mehr Zeit für die Implementation benötigen können. Die Integration von weiteren Tests hat geringe Priorität, solange die Kernfunktionalität des Programms noch nicht fertiggestellt ist. Ausnahme dafür ist der Ping-Test, welcher für die Entwicklung zum Ausprobieren verwendet wird und ein bis drei weitere Tests, welche für das Integration-Testing verwendet werden.

Mögliche Tests:

- ping
- tracert
- show ip interface
- · show ip protocols
- show ip route
- show interfaces
- iperf
- nmap portscan
- show ip ospf interface
- show ip ospf neighbor
- show ip eigrp neighbors
- show tag-switching tdp discovery
- show tag-switching tdp bindings
- show tag-switching forwarding-table
- show ip bgp
- show ip route bgp
- show ip bgp neighbors
- show mpls lsp extensive
- show isis adjacency
- show isis interface