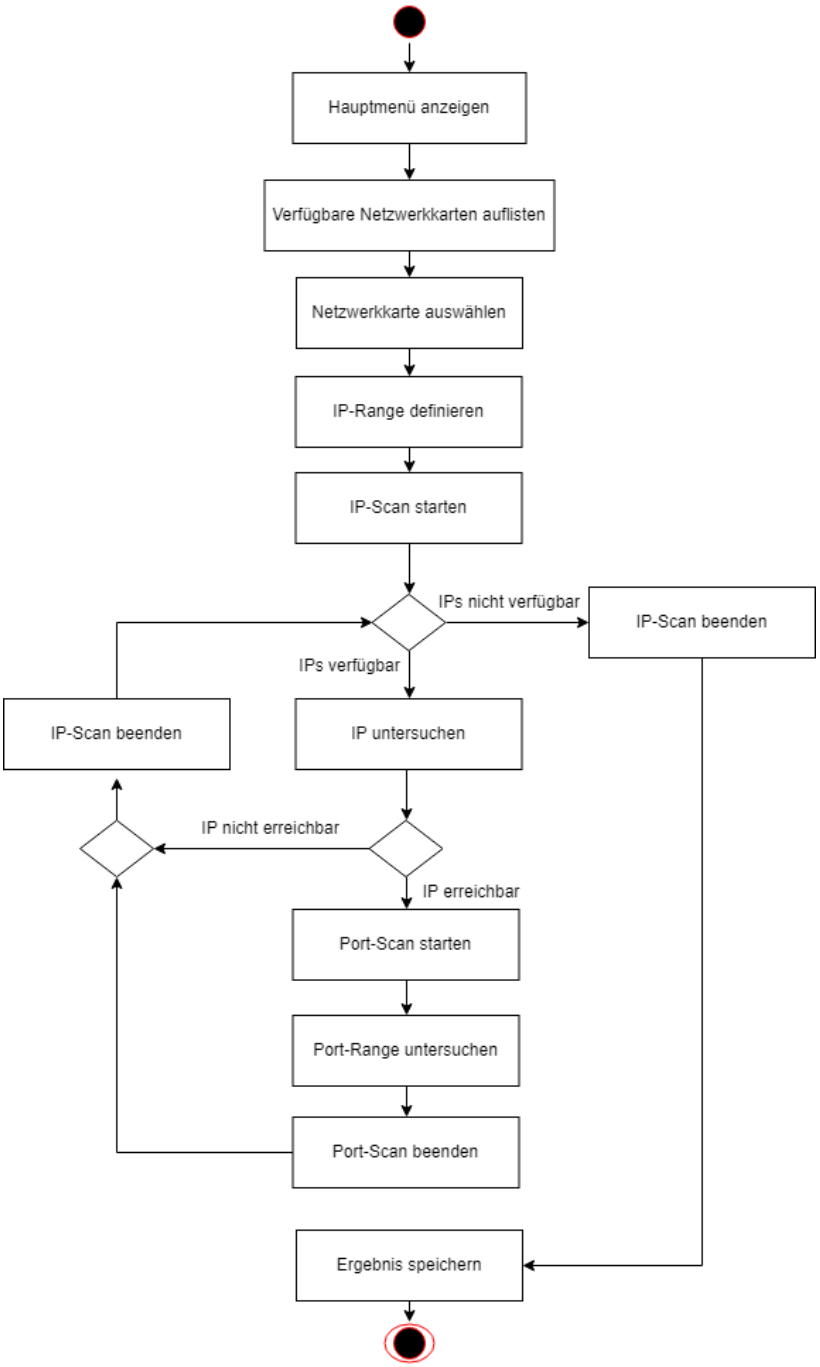
NetworkScan+
Softwaredokumentation
Einführung in die verteilten und parallelen Systeme

Jonas Theis
Matrikelnummer: 5095269

Übersicht

1. Ziel und Funktionsumfang von NetworkScan+

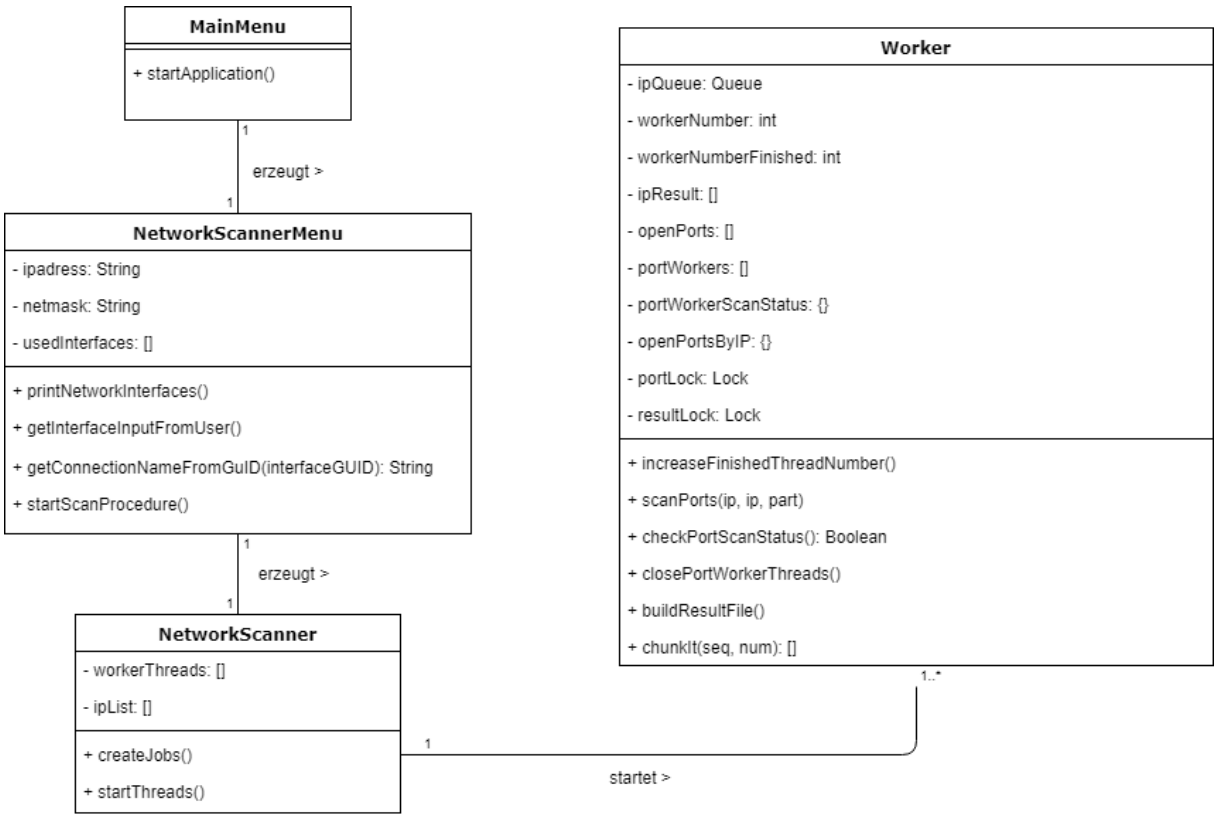
Das Ziel der in Python geschriebenen Konsolenanwendung „NetworkScan+“ für Windows Systeme ist es, dem Benutzer einen schnellen Überblick über die erreichbaren Hosts in einem vorhandenen Netzwerk zu ermöglichen. Neben der Übersicht, welche IPs erreichbar sind, wird zusätzlich ein Port-Scan für jede erreichbare IP durchgeführt. Dadurch können geöffnete Ports, welche nicht benötigt werden, schnell identifiziert werden. Bereits während des Scans wird der aktuelle Status und neue Informationen bzgl. des Scans in der Konsole ausgegeben. Nach Abschluss jedes Scans werden alle Informationen d.h. erreichbare IPs mit den offenen Ports in einer Log-Datei abgespeichert, sodass sich der Anwender nachträglich einen Überblick über das Netzwerk verschaffen kann. Das folgende Aktivitätsdiagramm beschreibt einen vereinfachten Ablauf der Software:



2. Technischer Aufbau

2.1. Klassendiagramm

Das folgende Diagramm stellt die in NetworkScan+ verwendeten Klassen und deren Beziehungen grafisch dar:



2.2. Beschreibung der Klassen

MainMenu

Die Klasse MainMenu startet das NetworkScan+ Menu und instanziiert daraufhin das NetworkScannerMenu.

NetworkScannerMenu

Die Klasse NetworkScannerMenu ist für die Ausgabe von verfügbaren Netzwerkkarten zuständig. Netzwerkkarten, welche über eine IP verfügen werden aufgelistet. Um die entsprechenden Namen der Interfaces auszulesen ist ein Zugriff auf die Windows-Registry erforderlich. Nach erfolgreicher Auswahl des Netzwerk-Interfaces wird die entsprechende IP und Subnetzmaske ausgelesen und beim Instanziieren an den Konstruktor der Klasse NetworkScanner übergeben.

NetworkScanner

Die Klasse NetworkScanner ermittelt nun auf Basis der IP und der Subnetzmaske alle Adressen, welche untersucht werden müssen. Anschließend wird die statische Queue der Klasse Worker mit den zuvor ermittelten IPs befüllt. Nach Befüllung der Queue werden 30 Objekte der Klasse Worker, welche von Thread erbt, instanziiert und gestartet.

Worker

Die Worker Klasse ist Dreh- und Angelpunkt der gesamten Applikation. Sie untersucht nun solange IPs in der Queue vorhanden sind, ob diese erreichbar ist. Die Ermittlung ob eine IP erreichbar ist oder nicht, wird anhand von Ports überprüft, welche standardmäßig auf den Betriebssystemen Windows, Linux und iOS offen sind. Die Liste dieser Ports wird mithilfe eines Sockets überprüft. Sobald eine erreichbare IP gefunden wurde, wird ein tiefgehender Port-Scan gestartet, welcher alle Ports bis 1024 untersucht. Der tiefgehende Port-Scan startet nun 11 Threads, wobei jeder dieser Threads einen Teil der Portrange 1024 bekommt. Also beispielsweise Thread 1 bekommt 1 – 100, Thread 2 bekommt 101 – 200 usw.. Sobald eine erreichbare IP oder ein offener Port gefunden wurde, wird dies in den vorhandenen Datenstrukturen abgespeichert. Nachdem alle IP und Port-Scans durchgeführt wurden, wird eine finale Log-Datei erstellt.

3. Verwendung von Zusatzbibliotheken

Folgende Zusatzbibliotheken wurden neben der Python 3.6.4 Standardbibliothek verwendet:

netifaces

- Für den Zugriff auf Netzwerkschnittstellen

Link: https://pypi.python.org/pypi/netifaces

Das Installieren dieser Bibliothek ist zwingend erforderlich anderenfalls für das Starten der Anwendung zu Fehlern.