Team 3: OPC UA Server Farm

Software Engineering

D. Zichler, N. Fischer, N. Hörber, N. Huber, P. Förster, ~~K. Knöpfle~~

Gliederung

- Vorstellung des Teams
- Was ist OPC UA?
- Ziel unseres Projekts (Master Use Case)
- Funktionsumfang und Anforderungen
- Produktübersicht (Black-Box)
- Architekturübersicht & Module
- Vorgehensweise beim Testen
- Live Demo
- Fazit / Ausblick

Team Vorstellung

Name	Rolle	Inf-Nr.
Daniel Zichler	Testmanager	19055
Niclas Hörber	Projektleiter	19046
Nico Fischer	Systemarchitekt	19034
Niklas Huber	Tech. Redakteur	19216
Philipp Förster	Produktmanager	19091
~~Kay Knöpfle~~	~~Produktmanager~~	~~19067~~

D. Zichler N. Hörber N. Fischer N. Huber P. Förster

Was ist OPC UA?

- Kommunikationsprotokoll für Industrie 4.0 und IoT
- Standardisierter Zugriff auf Maschinen, Geräte und andere Systeme (industrielles Umfeld)
- Herstellerunabhängigen und gleichartigen Datenaustausch

OPC UA Server

- Basis der OPC UA Kommunikation
- Stellt OPC Schnittstelle nach außen bereit

OPC UA Client

- Gegenstück zu Server
- Verbindet sich mit Server -> Daten von Server auslesen
- OPC Standard: Jeder OPC UA Client auf jeden OPC UA Server zugreifen

OPC UA Test Client

- Testet Konfiguration und Funktion eines OPC UA Servers

Ziel unseres Projektes

- **Master Use Case:** OPC UA Clients mithilfe einer Serverfarm testen

- Serverfarm soll nur auf einen Computer sein
  - Mehrere virtuelle OPC UA Servers (via Netzwerk bereitgestellt)
- 10 verschiedene Serverprofile bereitstellen
  - Mit AML Konfigurationsdatei (CAEX 3.0) parametrisierbar

- **Zielgruppe:** Entwickler und Tester von Anwendungen mit OPC UA Client-Interface

Funktionsumfang und Anforderungen

- **Identifizierender Geschäftsprozess:**

  „Simplified testing of OPC UA Clients“ (Master Use Case)
  - Also: User will mehrere OPC UA Clients testen
  - Dazu: Serverfarm starten
- Daraus ergeben sich 3 Use Cases:
  - „**Set server profiles**“
  - „**Starting server farm**“
  - „**Testing OPC UA Client(s)**“

Funktionsumfang und Anforderungen

- Command Line Interface
- File validation
- Server configuration
- Server startup
- Server shutdown
- Logging

2. Product Requirements

The following functionalities shall be supported by the system.

2.1. /LF10/Command Line Interface

The User can interact with the tool via command line for ease of use and automatability. The implementation of a graphical user interface (GUI) is optional. The user can specify a configuration file.

2.2. /LF20/File validation

The specified configuration file is checked for validity. If errors occur, they will be logged.

2.3. /LF30/Server configuration

The server shall be configured according to the specified configuration file ~~and port~~ *(The ports are set to a default value for an easier usability of the project).*

2.4. /LF40/Server startup

The server starts after the configuration finished successfully.

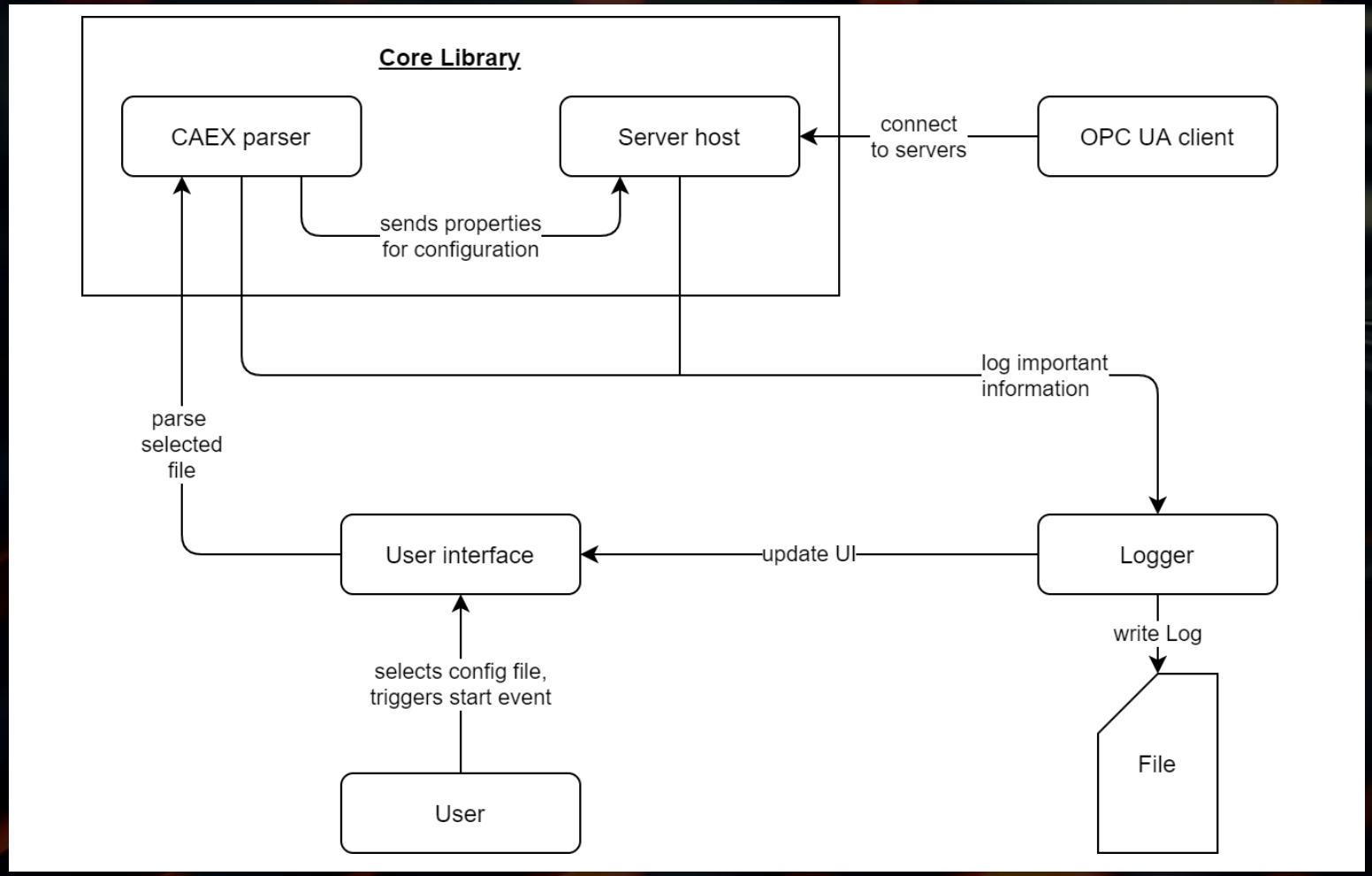
2.5. /LF50/Server shutdown

The server stops after external interrupt.

2.6. /LF60/Logging

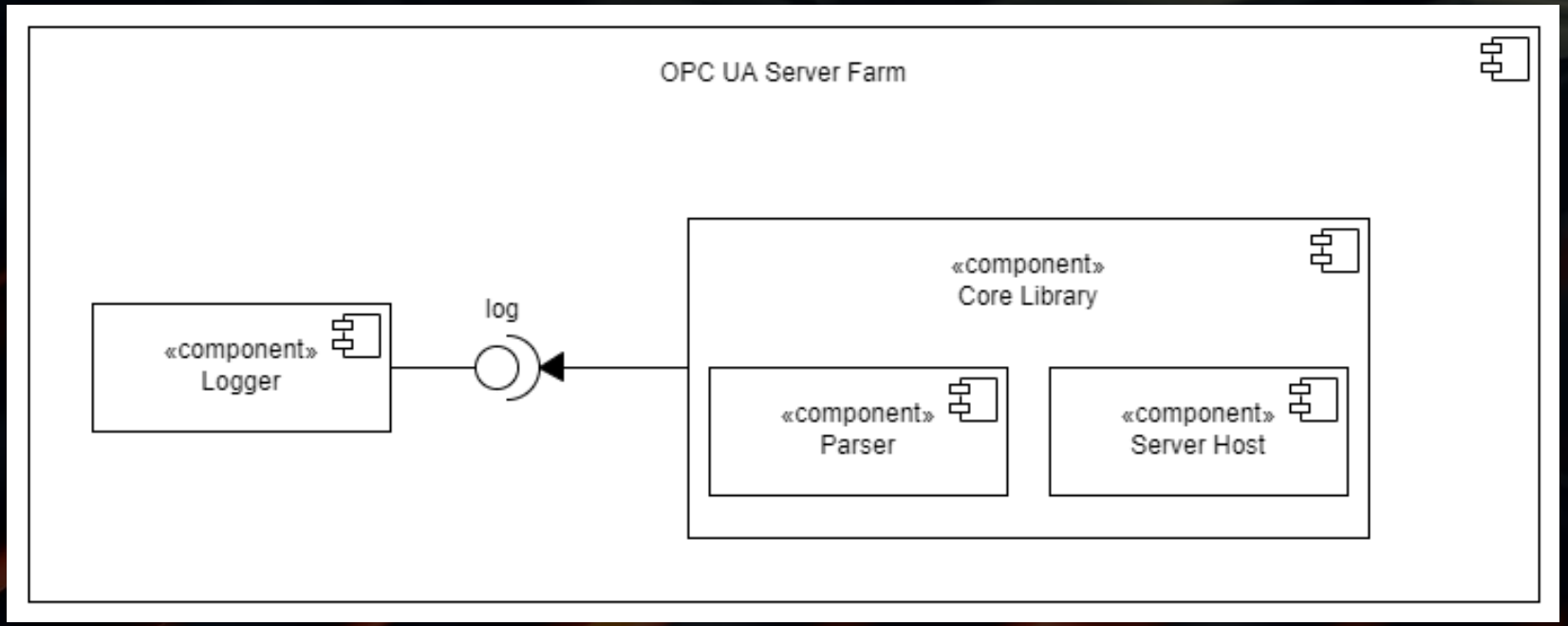
Errors and Events shall be logged to the command line interface or the GUI.

Produktübersicht



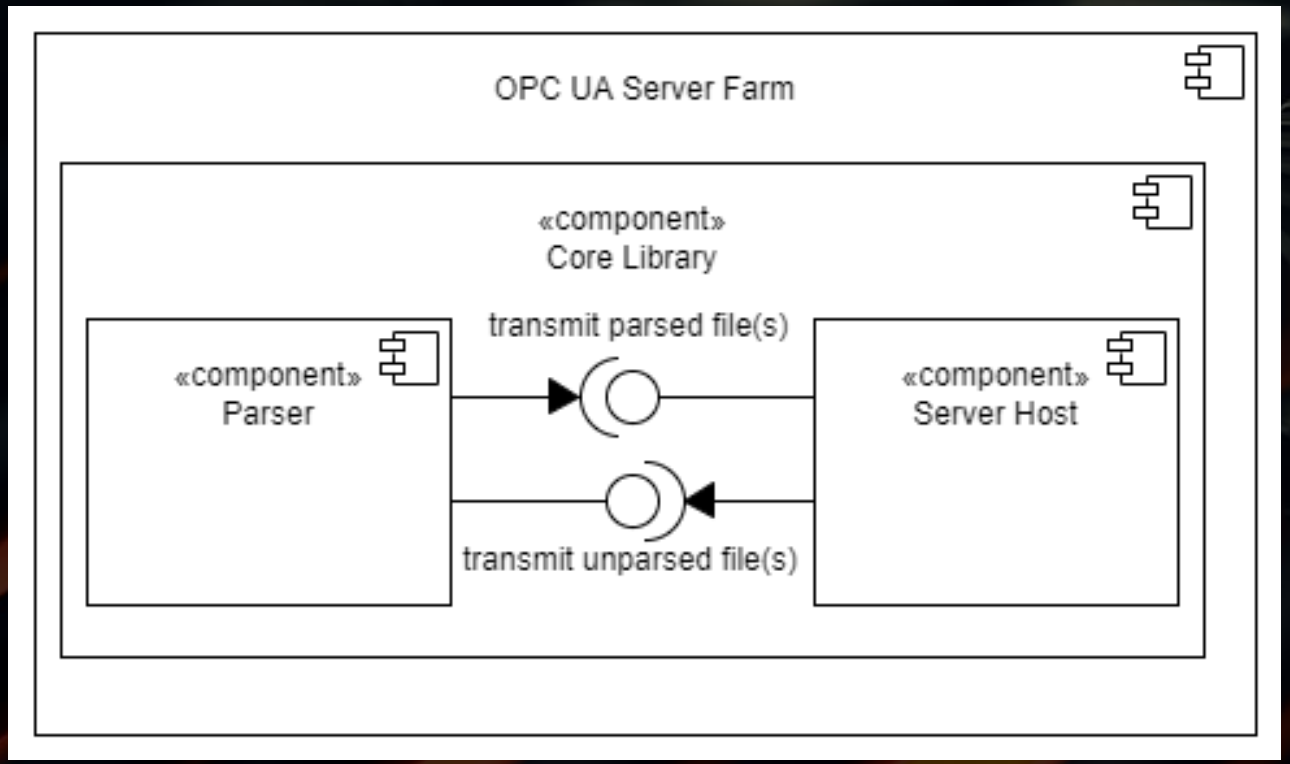
Modul: Core Library

- Modul für komplette Logik
- Enthält Parser und Server Host als Submodule



Submodul: Parser

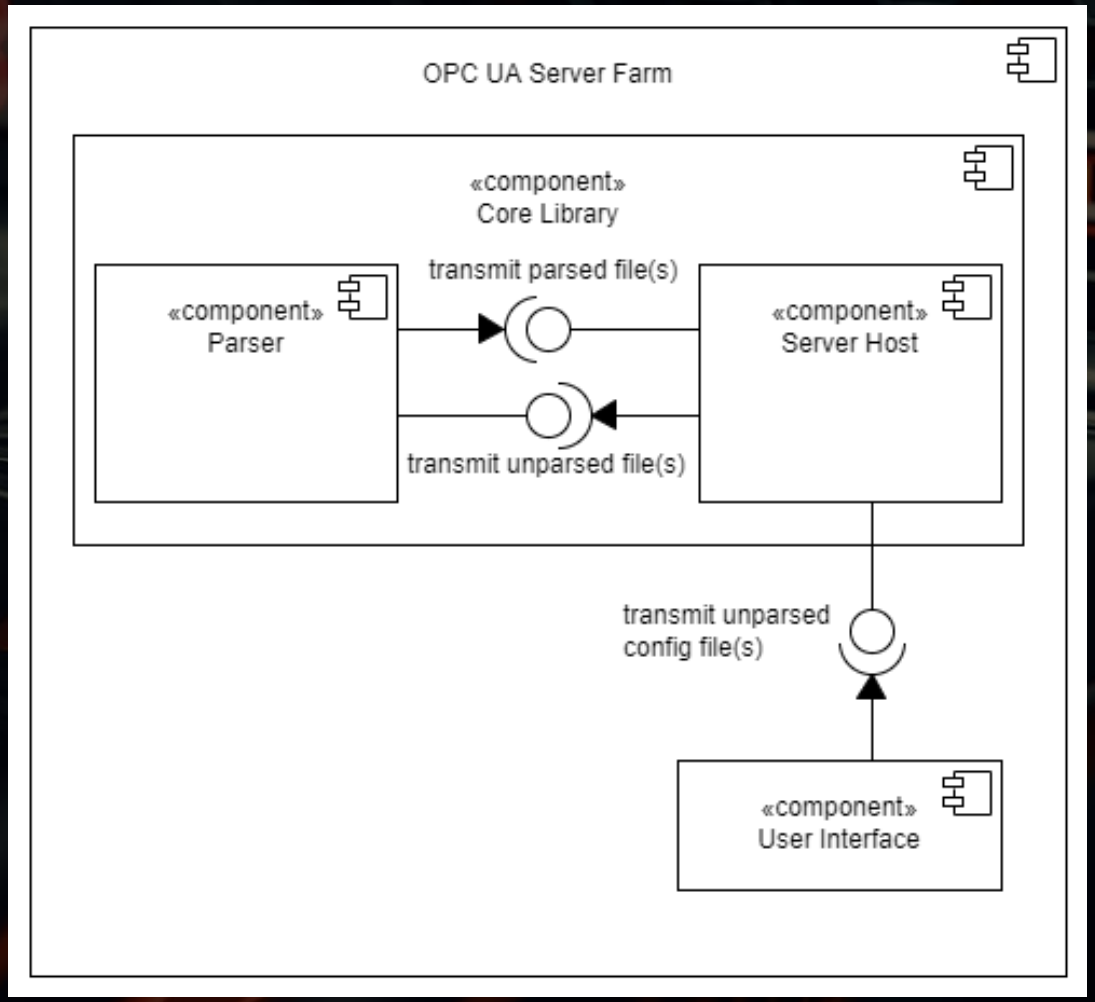
Parsed AML Konfigurations Datei in ein Objekt mit Childobjekte (in C++ mit structs)



Submodul: Server Host

Verwaltet komplettes Serververhalten

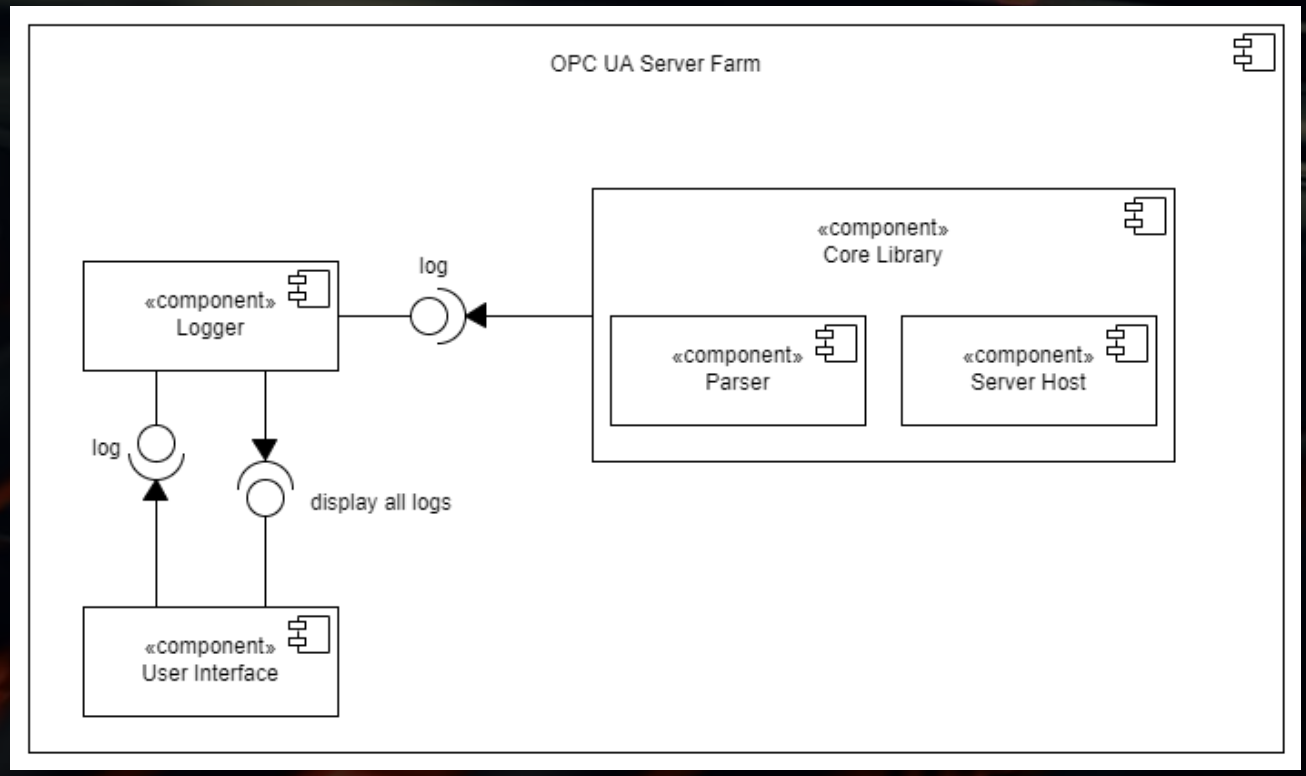
- Gibt Parser AML Konfigurationsdateien
- Startet Server Farm
- Schaltet Server Farm aus



Modul: Logger

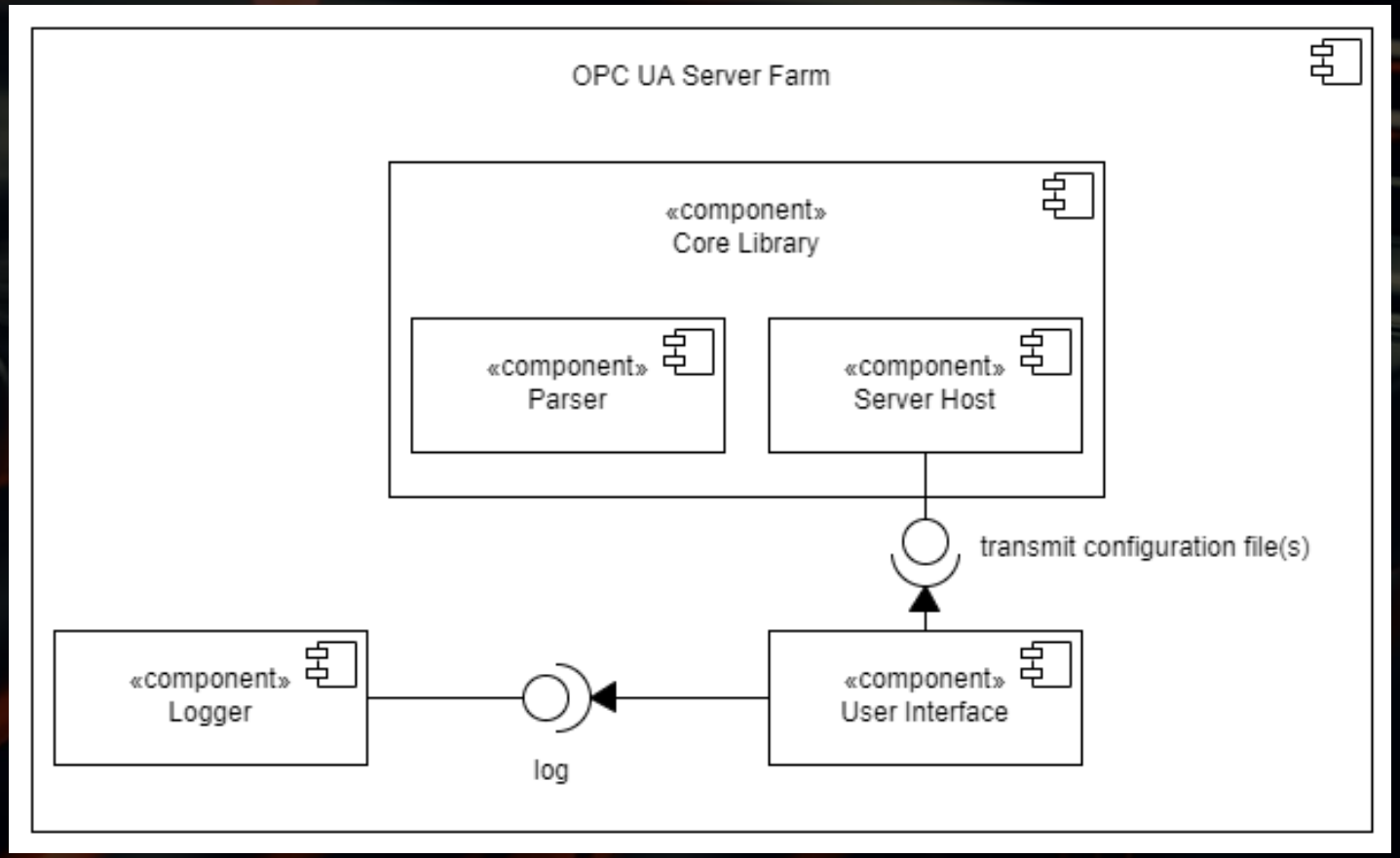
- Loggt die wichtigsten Fehler und Ereignisse im UI und Logfile

- z.B.
  - Erfolgreiches/Fehlgeschlagenes parsen der Konfigurationsdateien **(Logfile)**
  - Erfolgreiches/Fehlgeschlagenes starten der Server **(UI)**
  - Erfolgreiches/Fehlgeschlagenes ausschalten der Server **(UI)**

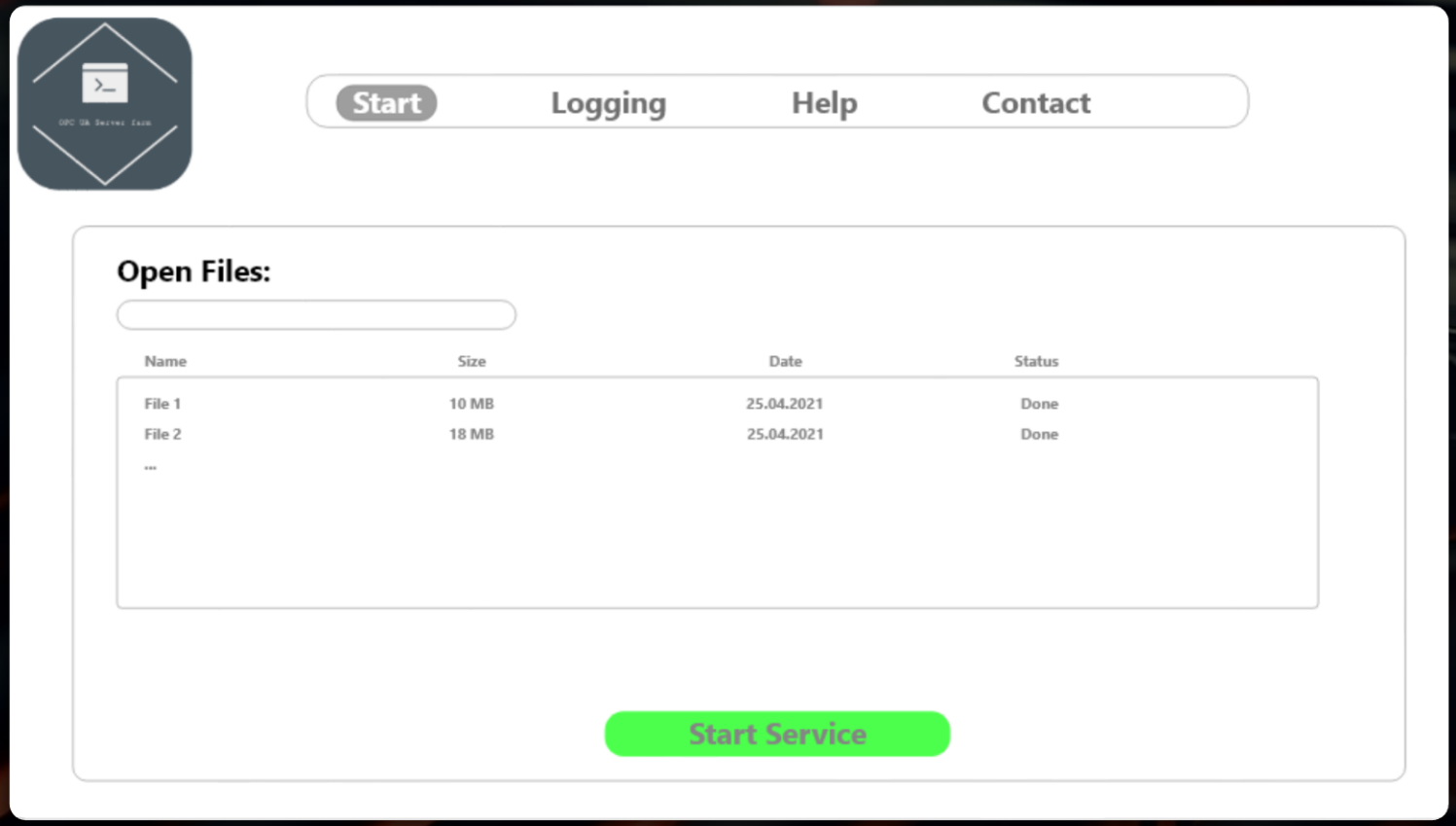


Modul: User Interface

- Keine hardcodierte Limitierung für Serveranzahl

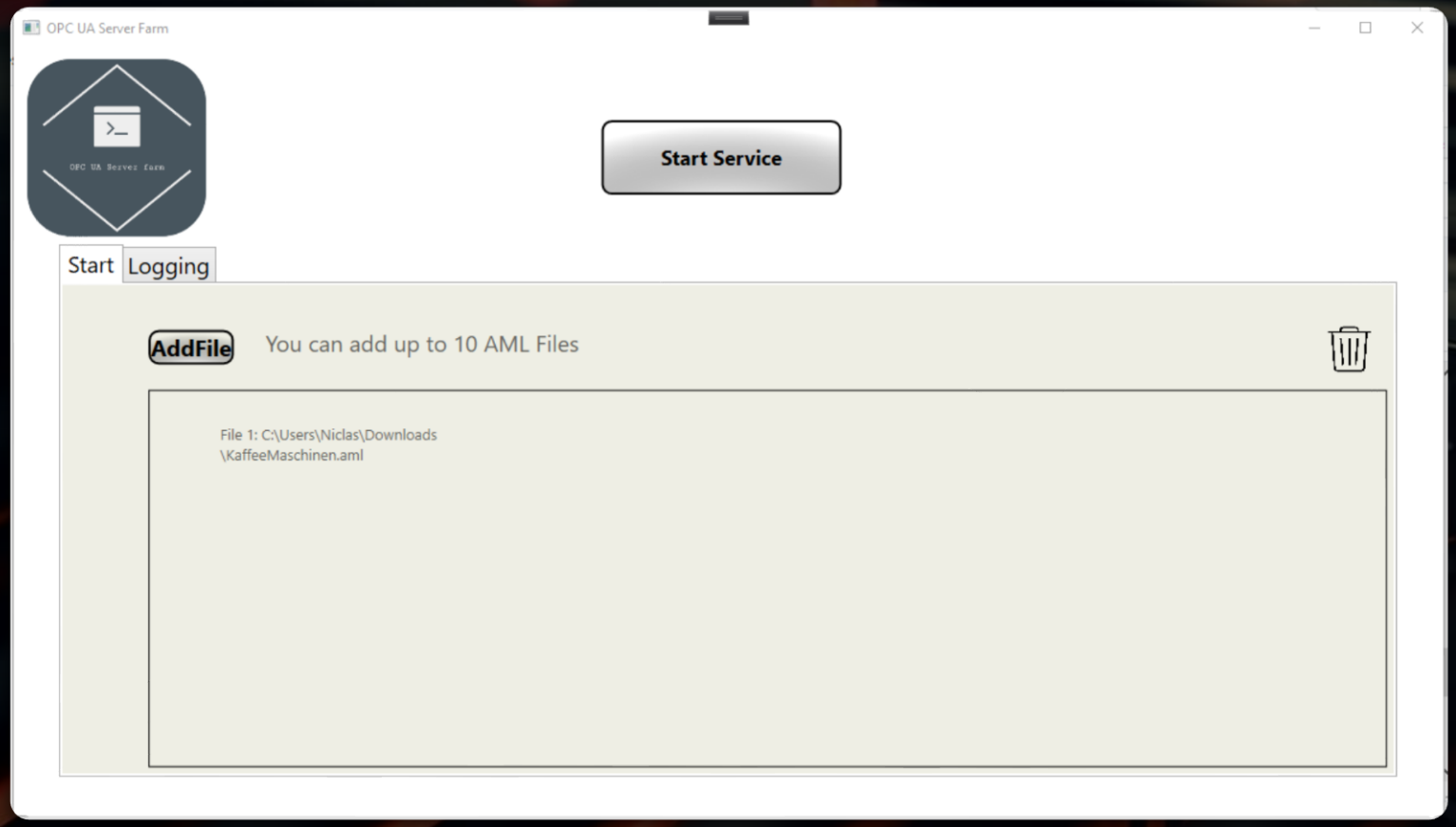


Modul: GUI



Adobe XD Prototyp

Modul: GUI



GUI Prototyp in C#

Vorgehensweise beim Testen

Test Preparation Strategy

- Die Tests sollen Fallabhängig ausgeführt werden
- Es gibt Zwei Hauptfälle, das GUI und die Core Library
- GUI wird nicht getestet, da es eine Sonderanforderung ist

Test Execution Strategy

- Weiterentwicklung einer bereits vorhandenen Software
- Da GUI nicht getestet werden soll, wird nur Core Library getestet
- Core Library wird mithilfe des Test Clients überprüft

Vorgehensweise beim Testen

Test Equipment

- Computer mit Linux
- OPC UA Demo Client
- 10 verschiedene Automation ML Dateien

Vorgehensweise beim Testen

Testergebnisse (Ausschnitt)

a. <TC-001-001> File validation of valid input file

Testcase ID:	TC-001-001
Testcase Name:	File validation of valid input file
Req.-ID:	LF10, LF20, LF30, LF60
Descrip-tion	The test case verifies that errors are detected during the validation of the input file. A corresponding log file will be created containing information of the error.

Test Steps

Step	Action	Expected Result	Actual Result
1	Install the OPC UA Server Farm and open the CLI.	The OPC UA Server Farm is installed on the system and the CLI is open.	The OPC UA Server Farm is installed on the system and the CLI is open.
2	Select a valid input file to con-figure the Server with.	The validation is executed successfully and the server will start configur-ated	The validation is exe-cuted successfully and the server starts con-figurating
3	Then open the logs which can be found under: <install-direc-tory>/Serverfarm/LogFile.txt	The log file is open and contains information about the start up	The log file is open and contains information about the start up
4	Find the message that the server has been started cor-rectly.	A Log message should be found within in the first few lines containing infor-mation about the start up	The Log file contains information about the start up and further shows on which port the server is running.

Tester:	Daniel Zichler
Date:	20.05.2021
Testcase Result:	PASS

b. <TC-001-002> File validation of invalid input file

Testcase ID:	TC-001-002
Testcase Name:	File validation of valid input file
Req.-ID:	LF10, LF20, LF30, LF60
Descrip-tion	The test case verifies that errors are detected during the validation of the input file. A corresponding log file will be created containing information of the error.

Test Steps

Step	Action	Expected Result	Actual Result
1	Install the OPC UA Server Farm and open the CLI.	The OPC UA Server Farm is installed on the system and the CLI is open.	The OPC UA Server Farm is installed on the system and the CLI is open.
2	Select a valid input file to con-figure the Server with.	The validation is executed successfully and the server will start configur-ated	The validation is exe-cuted successfully and the server starts con-figurating
3	Then open the logs which can be found under: <install-direc-tory>/Serverfarm/LogFile.txt	The log file is open and contains information about the start up	The log file is open and contains information about the start up
4	Find the message that the server has been started cor-rectly.	A Log message should be found within in the first few lines containing infor-mation about the start up	The log file shows that the file is not readable.

Tester:	Daniel Zichler
Date:	20.05.2021
Testcase Result:	PASS

Live Demo

Lessions Learned

- Fähigkeiten in C++ erweitert
- Schnittstelle C# & C++
- Projektstrukturierung/management
- Dokumentation von Software
- Gruppenaufteilung ist sinnvoll (Technischer Redakteur, Entwickler, …)
- Mit wegfallenden Teammitglied rechnen -> neue Aufteilung
- Regelmäßige Meetings mit Kunden -> Verbesserung des Projektes + eingehen auf Wünsche & Anregungen

Fazit & Ausblick

- Fazit:
  - Ungewohntes Themengebiet
  - Hohe Einarbeitungszeit

- Ausblick:
  - Open-source -> weiterentwickeln (GUI einbinden, …)
  - Produktiveinsatz um Clients zu testen

Team 3: OPC UA Server Farm

Software Engineering

D. Zichler, N. Fischer, N. Hörber, N. Huber, P. Förster~~, K. Knöpfle~~