

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Týmový projekt – Seminář C++
Konceptuální návrh

11. května 2025

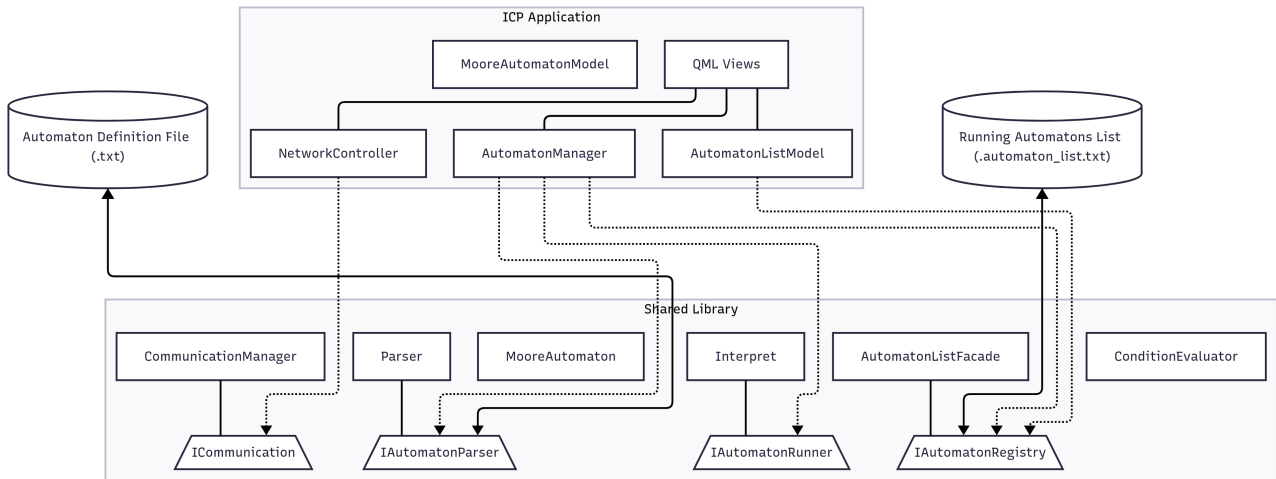
Sofia Kucherenko (xkucer0v)
Bedřich Petrášek (xpetra31)
Tomáš Boudný (xboudn05)

Obsah

1	Diagram komponent	2
2	Diagram tříd	4
3	Sekvenční diagramy	5
3.1	Spuštění automatu	5
3.2	Zpracování vstupu a aktualizace GUI	6

1 Diagram komponent

Diagram komponent (viz obrázek 1) zobrazuje vyšší úroveň abstrakce a ukazuje, jak jsou hlavní komponenty systému propojeny a jak spolu komunikují přes rozhraní. Zobrazuje také interakci s externími soubory.



Obrázek 1: Diagram komponent systému.

Systém je rozdělen na dvě hlavní komponenty:

- **Shared Library (Sdílená knihovna):** Tato komponenta zapouzdřuje veškerou základní logiku pro práci s Moorovými automaty. Obsahuje podkomponenty pro:
 - *MooreAutomaton*: Reprezentace samotného automatu, jeho stavů, přechodů, vstupů, výstupů a proměnných.
 - *Parser*: Zodpovídá za načítání definice automatu z textového souboru a ukládání automatu do souboru. Implementuje rozhraní *IAutomatonParser*.
 - *Interpret*: Komponenta pro vykonávání (simulaci) automatu. Zpracovává vstupy, vyhodnocuje podmínky přechodů, provádí akce spojené se stavy a řídí časování. Implementuje rozhraní *IAutomatonRunner*.
 - *CommunicationManager*: Poskytuje abstraktní základ a konkrétní implementace (např. *AutomatonCommunicationManager*, *GUICommunicationManager*) pro UDP komunikaci mezi interpretem automatu a grafickým uživatelským rozhraním. Implementuje rozhraní *ICommunication*.
 - *AutomatonListFacade*: Stará se o správu seznamu běžících automatů, jejich registraci (ukládání informací o portu a cestě k souboru) a deregistraci. Využívá soubor `.automaton_list.txt` pro persistenci. Implementuje rozhraní *IAutomatonRegistry*.
 - *ConditionEvaluator*: Pomocná komponenta pro vyhodnocování podmínek přechodů a provádění akcí definovaných ve stavech automatu.
- **ICP Application (Grafické uživatelské rozhraní):** Jedná se o spustitelnou aplikaci vytvořenou pomocí Qt/QML, která poskytuje uživatelské rozhraní pro návrh, správu a simulaci automatů. Využívá služby poskytované sdílenou knihovnou. Klíčové podkomponenty jsou:
 - *AutomatonManager ICP*: Centrální třída na straně GUI, která koordinuje operace jako načítání, ukládání, spouštění automatů a komunikaci s interprety. Využívá rozhraní *IAutomatonParser*, *IAutomatonRunner* a *IAutomatonRegistry* ze sdílené knihovny.

- *MooreAutomatonModel_ICP*: Qt model, který zpřístupňuje data automatu (stavy, přechody atd.) pro QML zobrazení a editaci. Slouží jako adaptér mezi *MooreAutomaton* ze sdílené knihovny a QML.
- *NetworkController_ICP*: Zajišťuje komunikaci z pohledu GUI s běžícími automaty (interprety) pomocí *GUICommunicationManager* ze sdílené knihovny. Využívá rozhraní *ICommunication*.
- *AutomatonListModel_ICP*: Qt model pro zobrazení seznamu běžících automatů v GUI. Načítá data pomocí *AutomatonListFacade* (přes *IAutomatonRegistry*).
- *QML Views*: Samotné QML soubory definující vizuální podobu a interakci uživatelského rozhraní (např. editor automatů, seznam běžících automatů).

Dále diagram znázorňuje dva klíčové externí soubory:

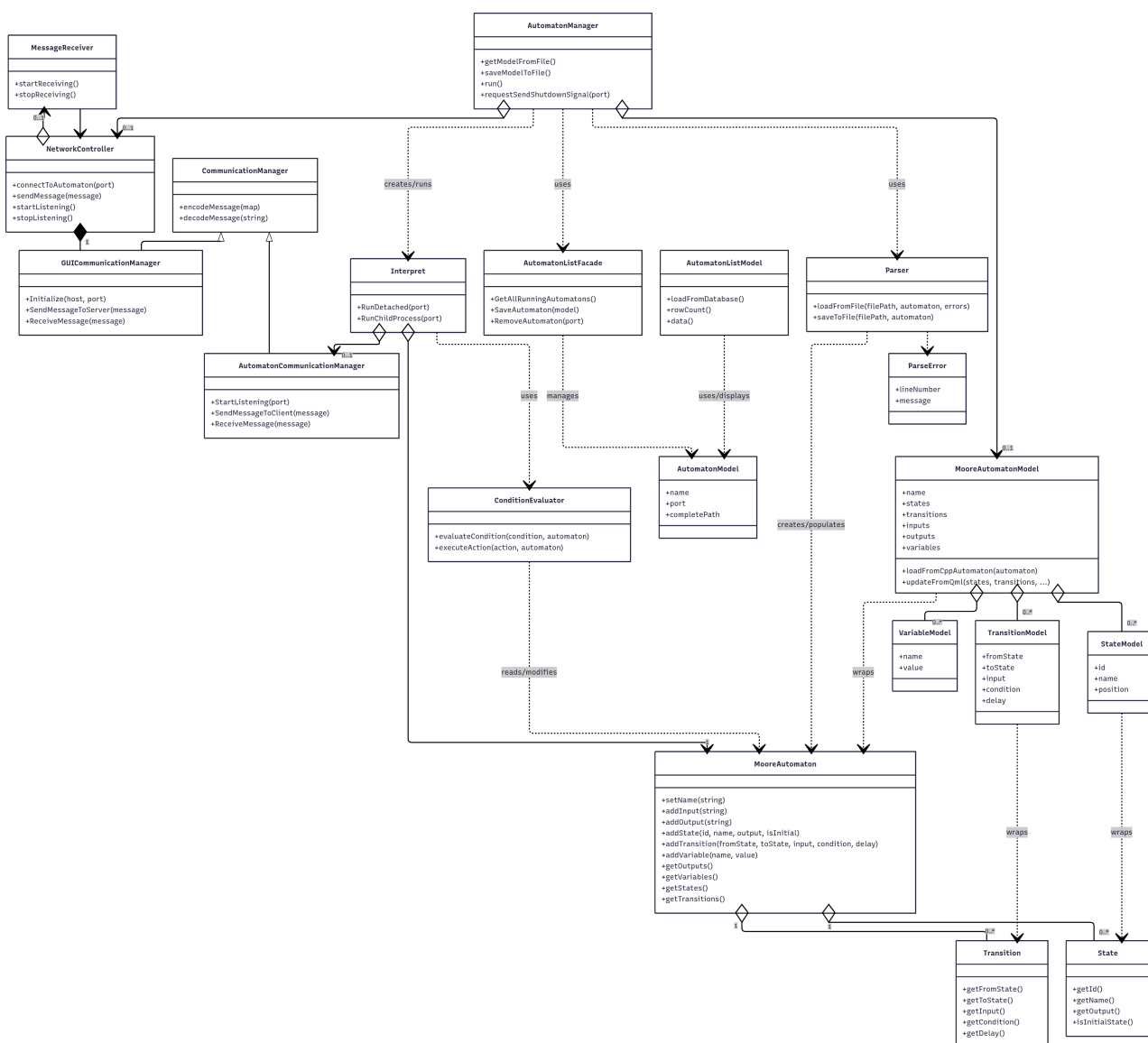
- **Automaton Definition File (.txt)**: Textový soubor obsahující definici jednoho Moorova automatu, který je zpracováván komponentou *Parser*.
- **Running Automatons List (.automaton_list.txt)**: Soubor spravovaný komponentou *AutomatonListFacade*, který slouží jako jednoduchá databáze (registr) informací o aktuálně běžících instancích automatů (jejich název, port a cesta k definičnímu souboru).

Komponenta *ICP Application* je závislá na *Shared Library*, což znamená, že využívá její funkcionalitu pro realizaci svých úloh. Komunikace mezi nimi probíhá přes definovaná rozhraní, což podporuje modularitu a oddělení zodpovědností.

2 Diagram tříd

Diagram tříd (viz obrázek 2) ukazuje klíčové třídy projektu rozdělené do dvou hlavních částí:

- Sdílená knihovna (Shared) - obsahuje základní třídy pro práci s automaty (MooreAutomaton, State, Transition), jejich analýzu (Parser, ConditionEvaluator) a komunikaci (CommunicationManager, AutomatonListFacade).
- Aplikační část (ICP Frontend) - obsahuje třídy GUI aplikace, která využívá sdílenou knihovnu (AutomatonManager, MooreAutomatonModel, NetworkController, AutomatonListModel).



Obrázek 2: Hlavní diagram tříd.

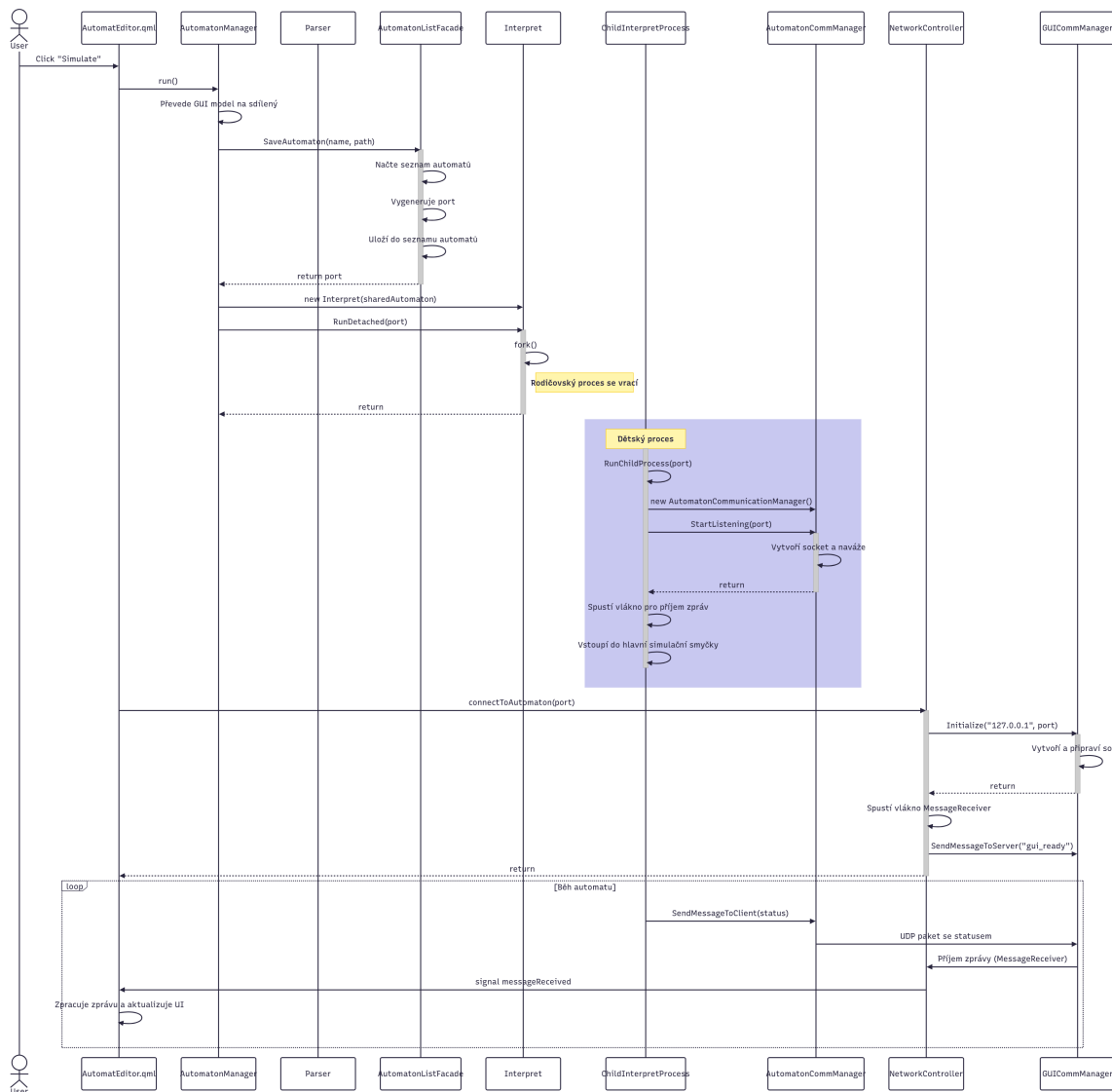
Diagram znázorňuje vztahy mezi třídami, jejich klíčové atributy a metody. Pro přehlednost jsou některé implementační detaily, jako konkrétní typy všech privátních proměnných nebo úplné signatury všech metod, vynechány.

3 Sekvenční diagramy

Dva sekvenční diagramy ilustrují klíčové scénáře použití aplikace:

3.1 Spuštění automatu

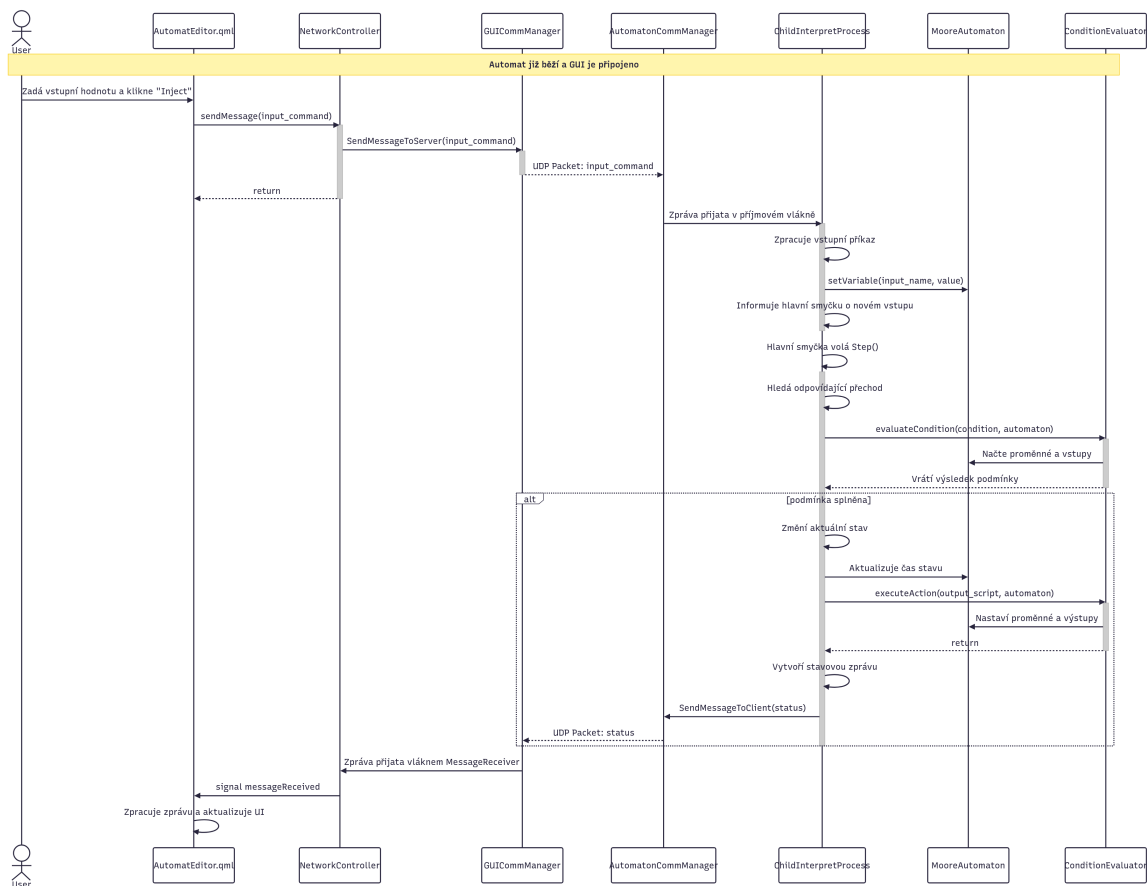
Zachycuje proces, kdy uživatel spustí simulaci automatu z GUI. To zahrnuje interakci s 'AutomatonManager', uložení informací o automatu pomocí 'AutomatonListFacade', vytvoření instance 'Interpret' a spuštění nového (dceřiného) procesu pro samotnou simulaci. Následně GUI naváže spojení s tímto novým procesem (viz obrázek 3).



Obrázek 3: Sekvenční diagram spuštění automatu.

3.2 Zpracování vstupu a aktualizace GUI

Ukazuje tok dat, když uživatel zadá vstup (např. hodnotu pro vstupní port) do běžícího automatu prostřednictvím GUI. Zpráva je odeslána interpretu automatu, který ji zpracuje, potenciálně změní svůj stav a provede akce. Následně interpret odešle aktualizovaný stav zpět do GUI, které se podle toho obnoví (viz obrázek 4).



Obrázek 4: Sekvenční diagram zpracování vstupu a aktualizace GUI.