VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DIAGNOSTIKA SYSTÉMŮ ZALOŽENÝCH NA GNU/-LINUX

DIAGNOSTICS OF GNU/LINUX-BASED SYSTEMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MARTIN HOFBAUER**
AUTHOR

VEDOUCÍ PRÁCE **doc. RNDr. PAVEL SMRŽ, Ph.D.**
SUPERVISOR

BRNO 2022

Abstrakt

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v českém (slovenském) jazyce.

Abstract

Do tohoto odstavce bude zapsán výtah (abstrakt) práce v anglickém jazyce.

Klíčová slova

Linux, GNU, logování, diagnostika, sběr informací, C++, CMake, Asio, modulog

Keywords

Linux, GNU, logging, diagnostics, collecting informations, C++, CMake, Asio, modulog

Citace

HOFBAUER, Martin. *Diagnostika systémů založených na GNU/Linux*. Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Pavel Smrž, Ph.D.

Diagnostika systémů založených na GNU/Linux

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. RNDr. Pavla Smrže, Ph.D. Další informace mi poskytla partnerská firma BringAuto. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Martin Hofbauer
12. února 2022

Poděkování

Děkuji vedoucímu bakalářské práce doc. RNDr. Pavlu Smrži, Ph.D. za rady, trpělivost a ochotu při konzultacích. Poděkování patří také firmě BringAuto za poskytnutí podpory a možnosti testování aplikace v reálném prostředí.

Obsah

Kapitola 1

Úvod

TODO

Kapitola 2

Výběr technologií

TODO

2.1 Programovací jazyk

Jednou z nejdůležitějších částí při výběru použitých technologií je právě programovací jazyk. V našem případě je nutné zvolit takový, ve kterém aplikace poběží velmi rychle. Také je důležité volit jazyk kompilovaný, aby na cílových zařízeních nemusel být přítomný interpret (kvůli zpomalenému vykonávání kódu a také rychlosti nasazování aplikace bez nutnosti instalovat dodatečné programy). Po zvážení všech vyjmenovaných požadavků jsme si vybrali programovací jazyk C++. Mohl by se také použít programovací jazyk C, ale C++ nabízí objektové programování a v kombinaci se systémem CMake je možné vytvářet hezky čitelný kód s jednoduchou správou knihoven třetích stran.

2.2 Meziprocesní komunikace

TODO(k čemu)

2.2.1 Roury

2.2.2 TCP/IP

TODO(co jsem vybral)

Kapitola 3

Návrh systému

Aplikaci chceme mít rozdělenou do dvou konceptuálních celků - agenti, kteří se budou starat o samostatné sbírání informací ze systému a jádrem, kterému budou tyto informace zasílány a následně je bude zpracovávat. Důležité je, aby selhání agenta neovlivnilo zbytek aplikace, takže jako nejlepší řešení se nabízí, že agenti budou samostatně běžící procesy, které svým selháním nemůžou ukončit celou aplikaci.

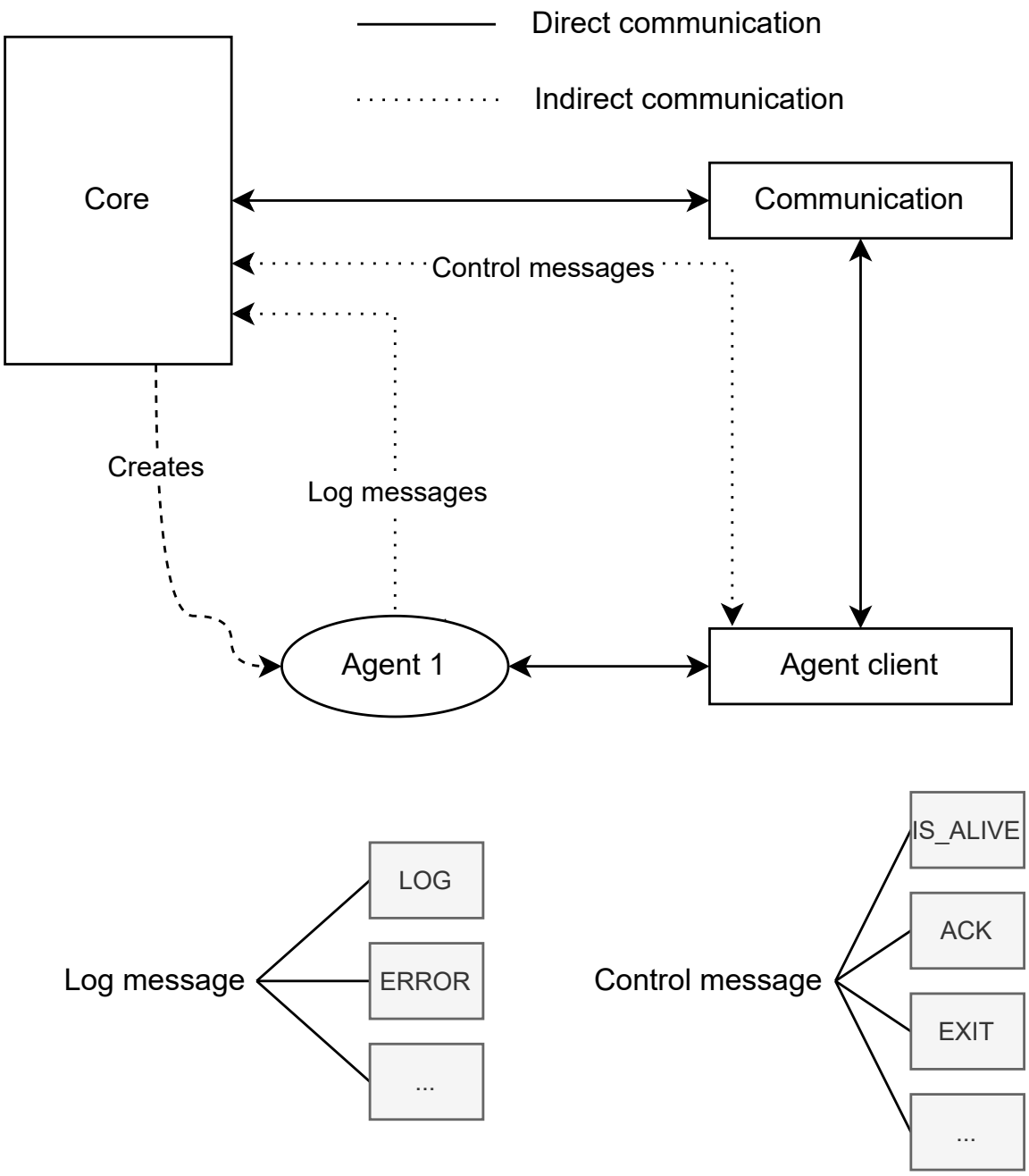
Rozdělíme tedy systém do tří částí: Core, Communication, AgentClient.

Core

Tato část bude jádrem celého programu. Na začátku vykonávání se postará o přečtení požadovaných agentů. Následně tyto agenty postupně začne jako samostatné procesy pouštět, během toho s nimi naváže spojení, vymění si s nimi potřebné informace a následně začne přijímat zaznamenané informace. Pokud během vytváření alespoň u jednoho agenta nastane chyba a nenaváže se spojení nebo něco podobného, tak je celý program ukončen.

Jádro poté přijímá zasílané informace agentů do té doby, dokud program není přerušen signálem nebo dokud nejsou všichni agenti ukončeni. V případě přerušení musí všechny agenty ukončit a následně ukončit i sebe.

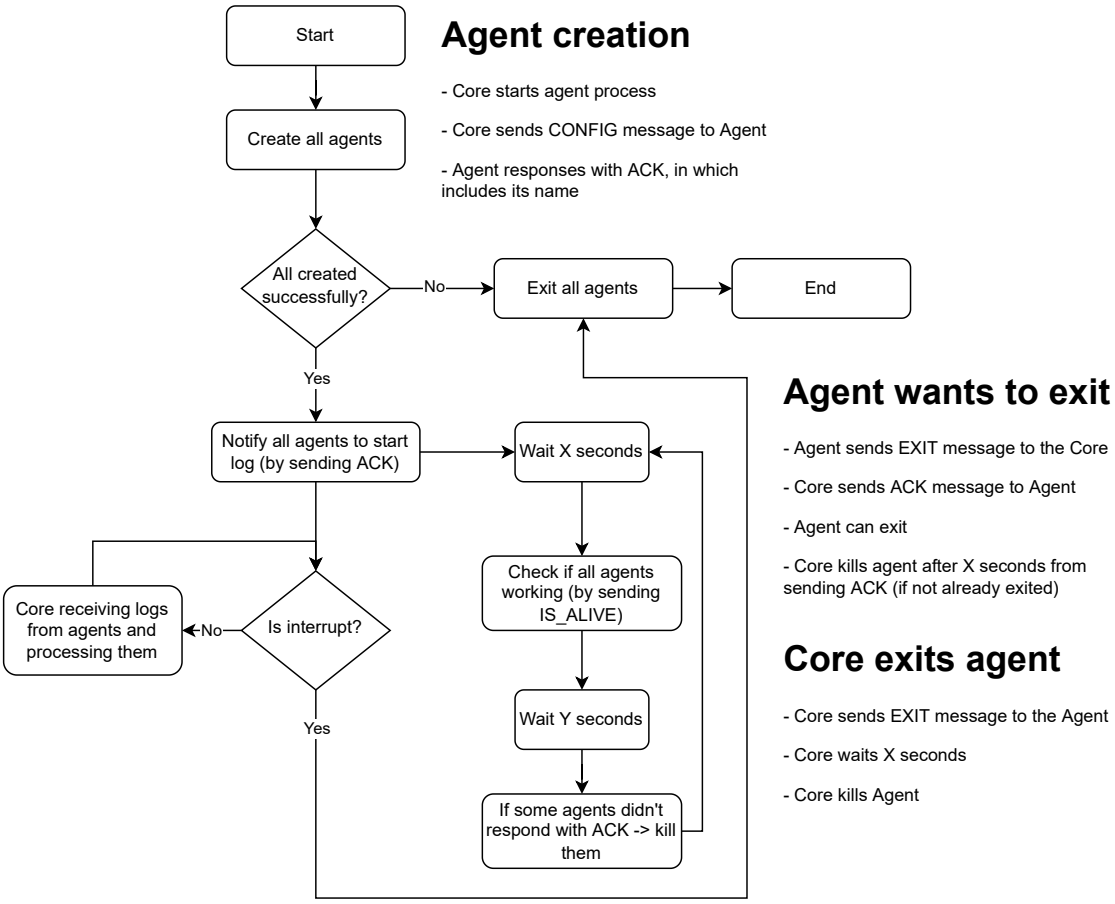
Na obrázku 3.1 vidíme.



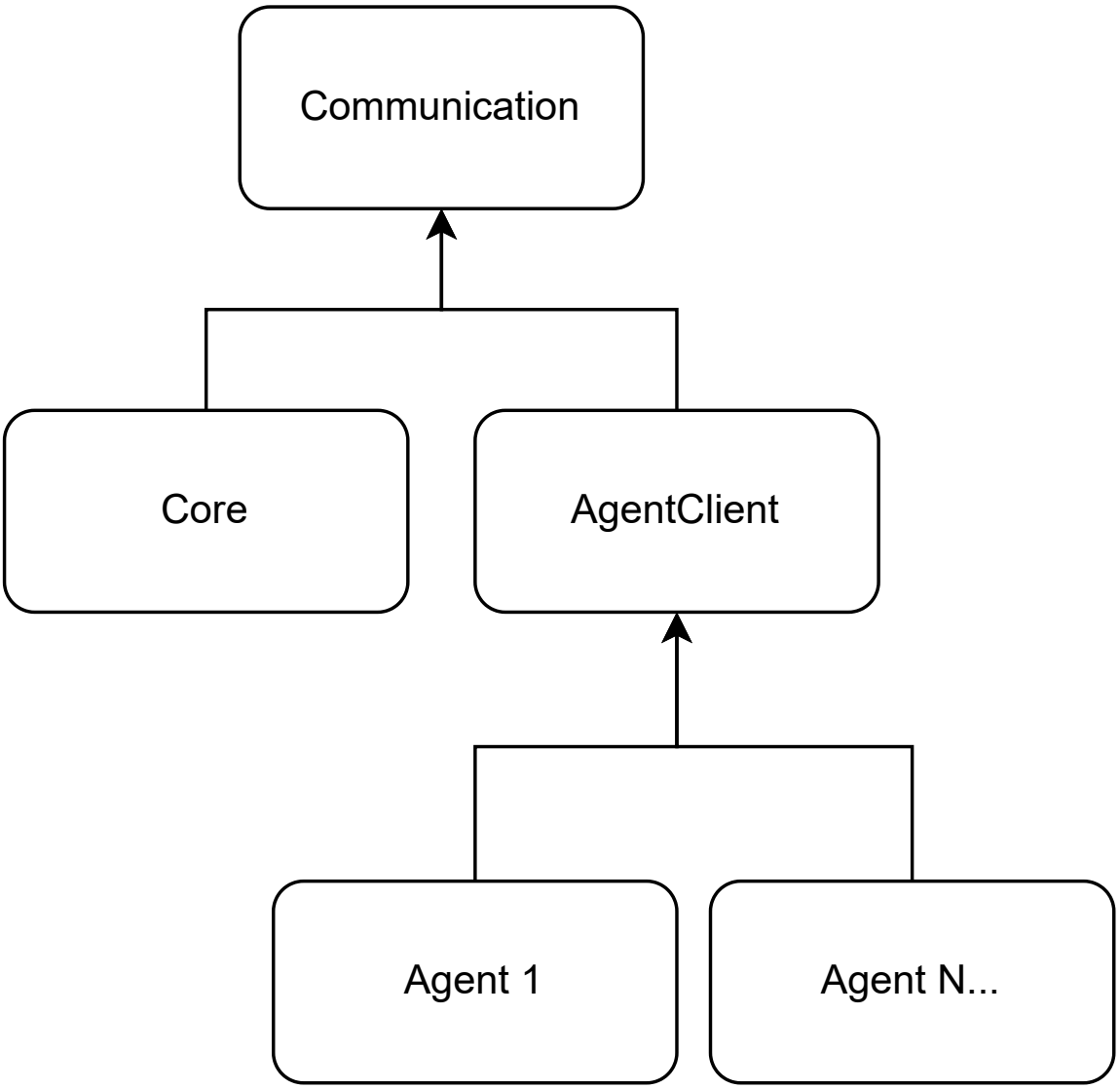
Obrázek 3.1: Architektura systému

Kapitola 4

Implementace



Obrázek 4.1: Vývojový diagram chování systému



Obrázek 4.2: Diagram závislostí

Kapitola 5

Testování

TODO

Kapitola 6

Závěr

TODO

Literatura

Příloha A

Příloha X