|  |  |
| --- | --- |
|  | |
| **Datově orientovaný přístup při vývoji software** | |
|  | |
| Bc. Tomáš Janečka | |
|  | |
|  |  |
| 2023 | Popis: fai_logo_cz |
|  |  |
|  | |

Text, letter

Description automatically generated

Text, letter

Description automatically generated**Prohlašuji, že**

* beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
* beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
* byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
* beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
* beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
* beru na vědomí, že pokud bylo k vypracování diplomové práce  
  využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými  
  subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu  
  využití), nelze výsledky diplomové práce využít ke komerčním  
  účelům;
* beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

* + že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
  + že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne TODO Tomáš Janečka, v.r.

ABSTRAKT

Diplomová práce se bude zabývat tématem datově orientovaného vývoje software. V teoretické části se student bude zabývat popisem jednotlivých technik návrhu software, charakteristikou používaného hardware a popisem vybraných aspektů ovlivňujících výkonnost implementovaných algoritmů. V praktické části budou srovnány jednotlivé způsoby implementace algoritmů včetně praktických příkladů doplněné o výstupy nástrojů pro výkonnostní testy a profilování.

Klíčová slova: datová orientace, C++, optimalizace, výkon, benchmarking, profilování, vyrovnávací paměť, operační paměť

ABSTRACT

This diploma thesis is concerned with the topic of data-oriented software design. In the theoretical part, the student is tasked with describing software design techniques, the characteristics of the hardware used and pointing out selected aspects that determine the performance of the implemented algorithms. The practical part shows the comparison of the different ways of algorithm implementations, along with examples including outputs from performance measuring tools and profilers.

Keywords: data-oriented, C++, optimization, performance, benchmarking, profiling, cache, memoryTODO

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

[Úvod 7](#_Toc120888191)

1. [TEORETICKÁ ČÁST 8](#_Toc120888192)

[1 Nadpis 9](#_Toc120888193)

[1.1 Podnadpis 9](#_Toc120888194)

[1.2 Podnadpis 9](#_Toc120888195)

[1.2.1 Podpodnadpis 9](#_Toc120888196)

[2 Nadpis 10](#_Toc120888197)

[2.1 Podnadpis 10](#_Toc120888198)

[2.1.1 Podpodnadpis 10](#_Toc120888199)

1. [Praktická část 11](#_Toc120888200)

[3 Nadpis 12](#_Toc120888201)

[3.1 Podnadpis 12](#_Toc120888202)

[3.2 Podnadpis 12](#_Toc120888203)

[4 Nadpis 13](#_Toc120888204)

[4.1 Podnadpis 13](#_Toc120888205)

[Závěr 14](#_Toc120888206)

[Seznam použité literatury 15](#_Toc120888207)

[Seznam použitých symbolů a zkratek 16](#_Toc120888208)

[Seznam obrázků 17](#_Toc120888209)

[Seznam tabulek 18](#_Toc120888210)

[Seznam Příloh 19](#_Toc120888211)

TODO – obsah poskládat manuálně

Úvod

Objektově orientované programování je jedno z nejrozšířenějších paradigmat mezi programátory. Mnoho z nich, včetně mě, se s ním setkali na úplném začátku své programátorské kariéry. Jedná se o velmi užitečný nástroj. Každý nástroj má ale svůj účel, a není určen k řešení každého problému. Jedním ze zásadních rozdílů mezi OOP a DOP (datově orientované programování) je ta věc, kterou tyto způsoby programování považují jako hlavní. Objektově orientovaný návrh se soustředí na vytvoření abstraktního, idealizovaného a také co nejobecnějšího modelu reálného problému. Naproti tomu datově orientovaný návrh považuje data za to nejvýznamnější. Pokud porozumíme datům, porozumíme problému. Přeci jenom, programy ve své podstatě slouží k transformaci dat z jedné podoby do druhé. [TODO acton2014] Zároveň se toto paradigma soustředí na charakteristiky hardware, na kterém náš software běží a dbá na efektivní využívání zdrojů. Z tohoto důvodu je tento způsob tvorby programů mimo jiné využíván v herním průmyslu. Právě herní vývojáři tvoří naučné podklady o tomto tématu, ve kterých často poukazují na podstatné nuance při souhře hardware a software. Tyto zdánlivé detaily však často mají zásadní vliv na rychlost běhu programu a také na využití operační paměti. Osobně jejich nápady považuji za velmi zajímavé, a proto by tato diplomová práce měla být shrnutím nejzásadnějších myšlenek tohoto paradigmata. V teoretické části si představíme definici DOP a popíšeme hlavní myšlenky. Rovněž je třeba se zabývat tématem mikroarchitektury počítače, jelikož jeho znalost je pro využití v této oblasti kritická. V praktické části bude představeno množství praktik, které lze aplikovat na tvorbu programu a budou porovnána běžná řešení často řešených problémů a také zhodnoceny výstupy výkonnostních testů a profilovacích nástrojů.

TODO

|  |  |
| --- | --- |
|  | TEORETICKÁ ČÁST |

# Datově orientované programování

V této sekci si přiblížíme pojem datově orientované programování a datově orientovaný návrh. Kromě dále uvedených definic je možné o tomto paradigmatu říci, že se jedná o způsob, jakým vyvíjet software. Zároveň ale může koexistovat s kódem, který byl napsán způsobem jiným ve stejném projektu. Tento nástroj se neváže ke konkrétní oblasti problémů či programovacích jazyků. V žádném případě se nejedná o něco, co by nebylo použito v minulosti, byť například pod jiným jménem.

## Definice

„Datově orientovaný návrh je dovednost navrhnout software pomocí vývoje transformací pro data v řádné formě, kde řádná forma je řízena cílovým hardwarem a transformacemi, které na něm běží.“ [TODO dodmain]

## Historický výskyt

## Hlavní myšlenky

### Je to o datech

„Data jsou vše, co máme“. Všechny aplikace, co kdy byly napsány, slouží k poskytnutí výstupu v závislosti na vstupních datech. Grafické aplikace pracují s obrázky. Textové editory pracují s textem. Každá z nich očekává určitý formát dat. Ten může být velmi složitý, nebo velmi jednoduchý. Programátoři si také často neuvědomují, že instrukce jsou také data, protože se rovněž nachází v operační paměti. Za všech okolností je potřeba myslet na to, že data nikdy neexistují jen tak v éteru, ale pokaždé se nachází na nějakém hardware, ať už na virtuálním stroji, nebo konkrétním procesoru. [TODO dodmain]

### Data nejsou problémová doména

„Datově orientovaný návrh nezabudovává problém z reálného světa do kódu“. Na rozdíl od objektově orientovaného přístupu, datově orientovaný přístup obětovává čitelnost kódu pro lidi, což nám umožňuje nezatěžovat počítač lidskými koncepty. Umístěním dat do třídy nám umožní dát těmto datům kontext, což ale může mít následek existence velkého množství dat, které spolu nesouvisí. Proto v tomto paradigmatu uvažujeme o datech jako o „faktech, o kterých můžeme uvažovat tak, jak potřebujeme pro získání výstupních dat v požadovaném formátu.“ [TODO dodmain]

### Statistika

„Data jsou typ, frekvence, množství, tvar a pravděpodobnost.“ [TODO dodmain] Nejenom vstupní data programu jsou zahrnuta do pojmu „data“. Data o datech mohou být stejně nebo i více významná. Ta mohou mít zásadní vliv na to, jak píšeme kód.

### Data se mění

### Forma dat

## Využití

Text

### Podpodnadpis



Obrázek . Ukázkový obrázek

Tabulka . Ukázková tabulka

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# jaká je vaše orientace?

text

## OOP

Text

## FP

## Porovnání

„Objektově orientovaný návrh je soustředěn na problém a jeho řešení. Objekty, abstraktní reprezentace věcí, které tvoří návrh řešení problému představeného v návrhovém dokumentu aplikace. Objekty manipulují pouze s těmi daty, které jsou potřeba pro jejich reprezentaci bez jakéhokoliv ohledu na hardware nebo na data z reálného světa nebo jejich množství. Z tohoto důvodu nám objektově orientovaný návrh umožní rychle sestavit první verze aplikací a tím pádem také první podobu kódu. Datově orientovaný návrh se k problému staví jinak. Na rozdíl od předpokladu, že nevíme nic o hardware, usuzujeme, že nevíme nic o řešeném problému. “ [TODO dodmain]

„Datově orientovaný návrh se vyhýbá mrhání prostředků tím, že nikdy nepředpokládá, že návrh musí existovat jinde než v dokumentu zadání, zatímco poskytuje řešení aktuálního problému.“ [TODO dodmain]

„Datově orientovaný návrh si nechává napovědět daty, která jsou pozorovatelná nebo očekávaná. Na rozdíl od uvažování všech možných scénářů nebo plánování adaptability, využíváme nejpravděpodobnější vstupy pro nasměrování algoritmu. Na rozdíl od plánování rozšiřitelnosti je jednoduchý a má za cíl splnit svůj úkol.“ [TODO dodmain]

# hardware jako platforma

## Vyrovnávací paměť

## Operační paměť

## CPU pipelining

## Out-of-order vykonávání instrukcí

## Prefetching instrukcí

## Assembler

## Kompilátory

## SIMD

## Paralelizace

# benchmarking

# profilování

## Typy profilerů

|  |  |
| --- | --- |
|  | Praktická část |

# příklady optimalizací

## Iterace polem

## Předvídatelnost operací

## Hot vs cold data

## SoA vs AoS

## Využití pipeliningu

# ověření výkonu pomocí nástrojů

# souhrn optimalizací

Závěr

text

Seznam použité literatury

1. Text

TODO – očíslovat a smazat

CppCon 2014: Mike Acton "Data-Oriented Design and C++". *Youtube* [online]. 30.9.2014 [cit. 2022-12-21]. Dostupné z: <https://www.youtube.com/watch?v=rX0ItVEVjHc>. Kanál uživatele CppCon

TODO – zkontrolovat na <https://odevzdej.cz/>

Seznam použitých symbolů a zkratek

|  |  |  |
| --- | --- | --- |
| DOD |  | Data-oriented design |
| DOP |  | Datově orientované programování |
| FP |  | Funkcionální programování |
| OOP |  | Objektově orientované programování |

Seznam obrázků

[Obrázek 1. Ukázkový obrázek 9](#_Toc56699409)

Seznam tabulek

[Tabulka 1. Ukázková tabulka 9](#_Toc56699398)

Seznam Příloh

Příloha P I: Název přílohy