# Dokumentace softwarové architektury pro kompresi stromových struktur

Verze 1.0

30. března 2025

# Obsah

# 1 Úvod

## 1.1 Účel

Tento dokument poskytuje přehled o softwarové architektuře systému navrženého pro kompresi stromových struktur na základě analýzy vět. Cílem projektu je identifikovat opakující se vzorce v textových větách, zpracovat tyto vzorce do stromové struktury a navrhnout metody pro jejich kompresi. Architektura systému je navržena podle architektonického vzoru **Pipes and Filters**, což znamená, že systém bude složen z několika nezávislých filtrů, které zpracovávají data v průběhu několika fází, přičemž každý filtr vykonává specifickou operaci na datech a předává je dalšímu filtru v řetězci.

## 1.2 Rozsah

Dokument se zaměřuje na návrh a implementaci algoritmů pro kompresi stromových struktur. Systém bude zahrnovat moduly pro analýzu textových vět, detekci opakujících se vzorců, jejich převod na stromové struktury a aplikaci metod pro jejich kompresi. Každý z těchto kroků bude reprezentován samostatným filtrem v rámci architektury Pipes and Filters, čímž se umožní efektivní modulární přístup k vývoji, testování a optimalizaci.

## 1.3 Definice a zkratky

- **Stromová struktura**: Reprezentace vět s hierarchickými vztahy mezi slovy a frázemi.
- **Kompresní algoritmus**: Algoritmus sloužící k redukci velikosti stromové struktury při zachování všech relevantních informací.
- **Opakující se vzory**: Vzory, které se v textu vyskytují vícekrát a mohou být efektivně komprimovány.
- **Syntaktická analýza**: Proces analýzy textu, který identifikuje gramatické struktury v textových větách.

# 2 Přehled

# 3 Architektonické znázornění

## 3.1 Architektonické cíle a omezení

Cílem architektury je vytvořit systém založený na principu Pipes and Filters, kde každý filtr bude vykonávat specifickou operaci na datech, a výsledky budou postupně procházet celým systémem. Mezi hlavní požadavky na systém patří:

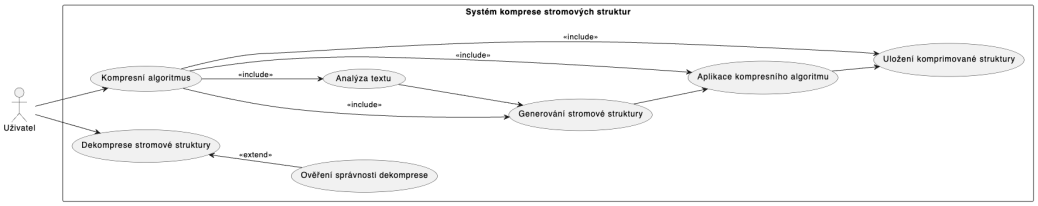
- **Modularita**: Systém bude navržen tak, že každý krok (analýza, identifikace vzorců, komprese) bude reprezentován samostatným filtrem.
- **Výkon**: Optimalizace pro zpracování velkých textových dat a složitých stromových struktur.
- **Flexibilita**: Schopnost přidávat nové filtry pro analýzu textu a kompresní metody.

## 3.2 Use-Case pohled

Hlavními use-case scénáři pro tento systém jsou:

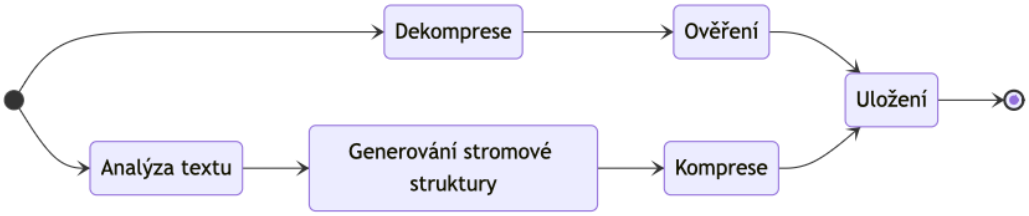
- **Kompresní algoritmus**: Tento scénář zahrnuje řetězec filtrů, které zahrnují analýzu textu, identifikaci opakujících se vzorců, generování stromové struktury, aplikaci kompresního algoritmu a uložení komprimované struktury.
- **Dekomprese stromové struktury**: Tento proces zahrnuje řetězec filtrů pro dekompresi a ověření správnosti dekomprimované struktury.
- **Analýza textu**: Tento filtr analyzuje text a generuje stromovou strukturu, která bude následně použita pro identifikaci vzorců.
- **Identifikace opakujících se vzorců**: Tento filtr provádí detekci vzorců v textu.
- **Generování stromové struktury**: Tento filtr generuje stromovou strukturu na základě analýzy textu a identifikovaných vzorců.

- **Aplikace kompresního algoritmu**: Tento filtr aplikuje kompresní algoritmus na vytvořenou stromovou strukturu.
- **Uložení komprimované struktury**: Tento filtr uloží komprimovanou strukturu pro pozdější použití.
- **Ověření správnosti dekomprese**: Tento filtr ověří, zda dekomprimovaná stromová struktura odpovídá původnímu textu.




Obrázek 1: Use-Case diagram pro kompresi a dekompresi textu

## 3.3 Logický pohled



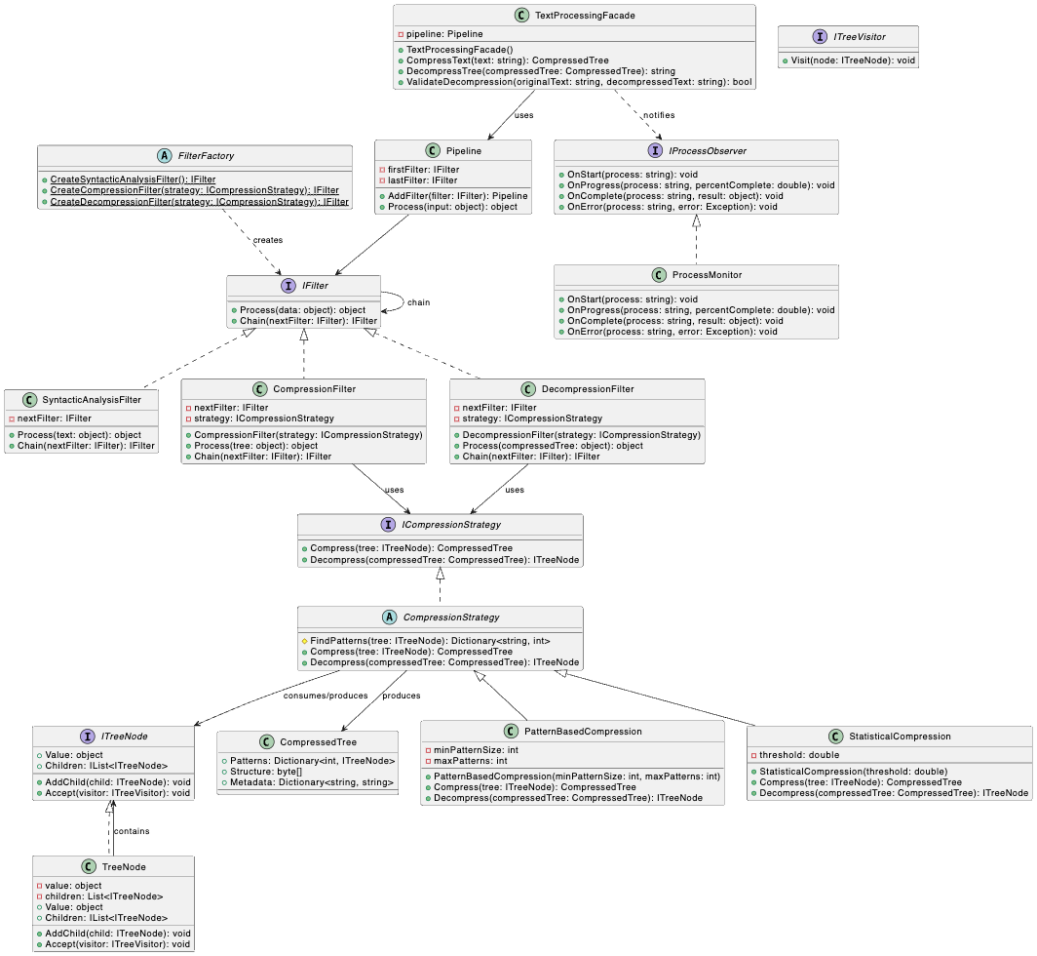

Obrázek 2: Diagram pipeline pro kompresi a dekompresi textu

Systém bude rozdělen do několika filtrů:

- **Filtr pro syntaktickou analýzu**: Tento filtr bude zodpovědný za rozpoznání syntaktických struktur ve větách a jejich převod na stromovou strukturu.
- **Filtr pro kompresi**: Tento filtr bude implementovat různé kompresní algoritmy pro optimalizaci velikosti stromových struktur.
- **Filtr pro dekompresi**: Po kompresi bude tento filtr zodpovědný za dekompresi a obnovu původních dat.

### 3.3.1 Popis diagramu tříd

Diagram tříd znázorňuje architekturu systému pro kompresi stromových struktur pomocí několika návrhových vzorů. Hlavní komponenty a jejich vztahy jsou následující:




Obrázek 3: Třídní diagram nejdůležitějších komponent

#### Hlavní rozhraní

- **IFilter**: Definuje metody pro zpracování dat a řetězení filtrů.

- **ITreeNode**: Reprezentuje uzel stromu s hodnotou a seznamem potomků.
- **ITreeVisitor**: Definuje metodu pro návštěvu uzlů stromu.
- **ICompressionStrategy**: Definuje metody pro kompresi a dekompresi stromových struktur.

**Abstraktní továrna pro filtry**

- **FilterFactory**: Abstraktní třída pro vytváření různých typů filtrů.

**Konkrétní filtry**

- **SyntacticAnalysisFilter**
- **CompressionFilter**
- **DecompressionFilter**

**Stromová struktura (Composite pattern)**

- **TreeNode**: Implementuje uzel stromu s hodnotou a seznamem potomků.
- **CompressedTree**: Reprezentuje komprimovanou stromovou strukturu s uloženými vzory a metadaty.

**Strategie pro kompresi (Strategy pattern)**

- **CompressionStrategy**: Abstraktní třída pro kompresní strategie.
- **PatternBasedCompression**
- **StatisticalCompression** Třídy sloužící k demonstraci implementace strategií komprese

**Pipeline (Pipes and Filters implementation)**

- **Pipeline**: Třída pro řetězení a zpracování filtrů.
- **TextProcessingFacade**: Fasáda pro zjednodušení interakce s klientem.

**Observer Pattern pro monitorování**

- **IProcessObserver**: Rozhraní pro sledování průběhu zpracování.
- **ProcessMonitor**: Implementace sledování průběhu zpracování.

**Vztahy mezi třídami**

- **Pipeline** používá **IFilter**.
- **IFilter** může být řetězeno s jiným **IFilter**.
- **FilterFactory** vytváří instance **IFilter**.
- **CompressionFilter** a **DecompressionFilter** používají **ICompressionStrategy**.
- **TextProcessingFacade** používá **Pipeline**.
- **TreeNode** obsahuje **ITreeNode**.
- **CompressionStrategy** produkuje **CompressedTree** a zpracovává **ITreeNode**.
- **TextProcessingFacade** notifikují **IProcessObserver**.

### 3.3.2 Implementační pohled

Systém bude využívat jazyk C# a binding knihovny pro jazykové modely od *Digital Research Infrastructure for the Language Technologies, Arts and Humanities*, jako UDPipe a MorphoDiTa. Tyto knihovny jsou doimplementovány a přijímají potřebné modely.

## 3.4 Procesní pohled

Všechny filtry poběží v jednom hlavním vlákně. U kompresního algoritmu se pokusím zamyslet, jakým způsobem by mohla vhodně využívat paralelizaci, případně vhodnou implementaci.

### 3.5 Nasazení

Systém bude nasazen jako konzolová aplikace. Tato aplikace bude implementována v jazyce C# a poběží na platformě .NET. Konzolová aplikace bude umožňovat spouštění jednotlivých filtrů v rámci architektury Pipes and Filters přímo z příkazové řádky. Uživatelé budou moci zadávat vstupní data, specifikovat parametry pro různé filtry a sledovat průběh zpracování v reálném čase. Konzolová aplikace poskytne flexibilní a snadno použitelný způsob interakce se systémem, což usnadní testování, ladění a nasazení v různých prostředích.

## 4 Velikost a výkon

Systém bude optimalizován pro analýzu a kompresi stromových struktur. K dosažení co nejlepšího výkonu bude systém využívat paralelní zpracování pro efektivní analýzu textů a kompresi vzorců.

## 5 Kvalita

Systém bude vysoce kvalitní díky:

- **Rozšiřitelnosti**: Možnost přidávat nové filtry pro kompresi a analýzu.
- **Přesnosti**: Systém bude zajišťovat vysokou přesnost při analýze a kompresi.
- **Modularitě**: Struktura systému umožňuje snadnou údržbu a rozšiřování.