Posudek *oponenta* diplomové práce

Jméno a příjmení autora posudku: Filip Krijt

Jméno a příjmení autora práce: Marek Linka

Název práce: Visual Studio Refactoring and Code Style Management Toolset

---

Tato implementační diplomová práce se zaměřuje na vytvoření sady nástrojů pro podporu kvality kódu v jazyce C#. Práce představuje implementaci rozšiřitelné platformy, ale i sady základních diagnostik, oprav a refaktorovacích operací. Součástí služeb platformy je centralizovaná správa konfigurace pluginů pro platformu. Práce rovněž obsahuje relativně nezávislou část ve formě panelu "Code Map", který umožňuje rychlou vizuální navigaci v kódu aktuální třídy (resp. souboru).

Text práce je v angličtině, což považuji za pozitivum. Celkově vzato je práce z jazykového i typografického hlediska v pořádku, angličtina je na dostatečné úrovni aby čtení po většinu času nijak nenarušovala. Práce nicméně obsahuje větší množství překlepů a místy nestandardní větné stavby a konstrukce a jistě by bývala byla snesla více kontrolních přečtení.

Text práce je strukturován logicky, práce obsahuje jasně řečenou motivaci a cíle, k cílům se odvolává v závěru při hodnocení splnění. Rovněž obsahuje popis a hodnocení funkcionality konkurenčních produktů, vymezení práce vůči nim je opodstatněné a uvěřitelné. Analytická část práce obsahuje adekvátní popis a zdůvodnění řešení několika zajímavých problémů. Zvolená řešení jsou v souladu s moderními softwarově-inženýrskými praktikami a rovněž obsahují návrhové vzory. Část popisující implementaci obsahuje dostatečně detailní text a diagramy pro vhled do architektury. Práce zmiňuje i problémy a limitace řešení, nikoliv pouze pozitiva. Motivace pro část "Code Map" by mohla být formulována lépe, neboť práce odkazuje na absenci nástrojů pro navigaci v jednom souboru, ve skutečnosti však chybí *vizuální* navigační nástroj – Visual Studio obsahuje nástroj umožňující základní navigaci na deklaraci třídy, metody, atp. v jednom souboru.

Implementační část práce působí solidně a adekvátně zadání. Jak již bylo řečeno dříve, práce je stavěna na velmi moderní technologii Roslyn, která se od verze Visual Studio 2015 stane hlavním kompilátorem jazyka C# a zároveň službou pro aktivity související s analýzou zdrojového kódu a manipulací s ním. To pomáhá vymezení práce vůči konkurenčním produktům (které typicky stojí na samostatných parserech) a také to práci poskytuje dobrou pozici co se týče možnosti adaptace na budoucí verze produktu Visual Studio.

Problémem se nicméně ukázal vývoj vůči experimentálnímu API. Implementace byla vyvíjena proti Preview verzi Roslynu a Visual Studia 2015, ta již ale v době psaní tohoto posudku není k dispozici, neboť byla nahrazena verzí RC. Po instalaci verze RC se mi nepodařilo tu část implementace, která záleží na funkcionalitě Roslynu, zprovoznit. Když byl řešitel o nemožnosti zprovoznění implementace informován, rychle identifikoval příčinu a i přes ukončení platnosti Preview licencí přišel se způsobem, jak funkcionalitu na požádání demonstrovat ve verzi Preview. Implementaci bohužel není možné bez většího přepisování překompilovat proti RC verzi, neboť API Roslynu doznalo mezi verzemi větších změn, což ale není chyba na straně řešitele. Zároveň tento fakt nepředstavuje velkou překážku pro budoucí využití práce.

Co se týče architektury řešení, působí celistvým dojmem a vhodně pro řešený problém. Práce rozumně využívá externí technologie, což se projevuje relativně jednoduchou architekturou. Rozdělení do projektů a další organizační aspekty práce jsou dobré. Jak již bylo zmíněno dříve, práce obsahuje na vhodných místech návrhové vzory. Kód působí stylově jednotně a udržovaně, používá rozumné abstrakce a celkově vzato působí kvalitně. Práce obsahuje vygenerovanou dokumentaci s rozumným pokrytím, až na občasnou chybějící summary jmenného prostoru.

Práce obsahuje testy, testované komponenty se zdají být rozumně izolované. Práce připouští, že pokrytí testů není dokonalé, je nicméně adekvátní a už jejich samotnou přítomnost považuji za pozitivum. Část implementace obsahuje pomocné metody pro testování, tyto jsou postaveny na fluent API syntaxi známé např. z mockovacích frameworků. Použití fluent API se zde tedy zdá jako rozumná volba, zlepšuje přehlednost testů a usnadňuje jejich případné rozšiřování.

Celkově vzato považuji práci až na drobné detaily za kvalitní a doporučuji k obhajobě.

Doporučení k obhajobě:

Z výše uvedených důvodů práci *doporučuji* k obhajobě.

Vynikající práce vhodná pro soutěž studentských prací	ANO ☐

Seznam soutěží studentských prací, viz http://www.mff.cuni.cz/studium/bcmgr/prace/

Pokud jste výše zaškrtli ANO, zdůvodněte prosím svůj návrh, případně uveďte konkrétní soutěž, pro kterou je práce vhodná (rámeček lze nechat prázdný, pokud za dostatečné zdůvodnění považujete text posudku):

V Praze dne: 7. 6. 2015

Podpis: