Aspektovo riadená konfigurácia radu softvérových výrobkov

Jakub Perdek

Inštitút informatiky, informačných systémov a softvérového inžinierstva
Fakulta informatiky a informačných technológií
Slovenská technická univerzita v Bratislave

20. Október, 2021

Abstrakt

Generovanie variantov produktov pomáha vytvoriť produkty z danej domény efektívnejšie a lacnejšie. Nevzniká tak iba jeden produkt, ale ľahko adaptovateľné produkty pre rôzne prostredia, jazyky a ďalšie požiadavky (Beuche a Dalgarno 2006). Derivovať produkty prispôsobené požadovanej variabilite možno za pomoci aspektovo orientovaného prístupu, hlavne jazyka AspectJ (Young a Math, 1999), ktorého možnosti by sme chceli pri tejto úlohe preskúmať. Vyberáme použitie jazyka AspectJ, pretože umožňuje oddeliť záležitosti ako je logovanie, kešovanie, poolovanie, ale hlavne je možné vyradiť z vykonávania vybrané aspekty, respektíve isté vlastnosti aplikácie, prípadne aj celkovú závislosť na tomto jazyku (Laddad, 2003).

Zaujíma nás preto uplatnenie jazyka pri umožnení konfigurovateľnosti konkrétnych vlastností v kóde, a samotné generovanie konkrétneho variantu produktu s prípadnou požiadavkou nezávislosti implementácie od tohto použitého jazyka. Zhodnotíme prínosy a kvality aspektovo orientovaného prístupu na konkrétnej implementácii, a porovnáme ich s identifikovanými problémami a benefitmi spomenutými v rôznych reimplementáciách už existujúcich softvérových systémov ako je vedecká kalkulačka (Botterweck, 2009) alebo databázový systém (Kastner, 2007). Samotné analýzy naznačujú význam voľby domény pri zostrojovaní radu softvérových produktov, preto by sme v neposlednom rade chceli porovna vplyv domény pri ich implementácii.

Plánujeme prispôsobiť existujúcu hru Battleship tak, aby umožňovala manažovať vlastnosti na základe už vytvoreného modelu vlastností, väčšina ktorých nie je vôbec implementovaná. Cieľom je aj analyzovať spôsoby odvodenia konkrétneho produktu, a mieru jeho závislostí od jazyka AspectJ.

Ďalšou aplikáciou pre analýzu je odvodzovanie rôznych fraktálov (ukážky typov nájdete v Pelánek, 2012) z pôvodného algoritmu, ktoré

by sme realizovali zásahom aspektov do jeho vykonávania. Predpokladáme, že úloha samotného návrhu vlastností pre uvedenú aplikáciu je určená hodnotou vzhľadu fraktálu, pričom zhotovenie radu produktov preň môže vyžadovať iné nároky na model vlastností, ako napríklad generovanie všetkých možných derivácií pre následné overenie estetiky a identifikovanie najvhodnejších kandidátov. Zamýšľame preto zhodnotiť význam prípadného potenciálu aspektovo orientovaného riešenia pre derivovanie konkrétnych fraktálov.

1 Úvod

1. Čo najvačší počet využití už prítomných rozhraní jednotlivých komponentov implementujúcich návrhový vzor Strategy a využívajúcich reflexiu (nie je v diagrame komponentov znázornené).

2. Čo najmenší počet závislostí medzi jednotlivými komponentmi.

3. Jednoduché využitie iného komponentu v rámci druhého komponentu bez nutnosti vytvárať rozhrania týchto komponentov.

4. Zahrnúť dôležité technológie a vhodne ich rozdeliť do komponentov.

5. Oddeliť konkrétnu implementáciu hry od herného engínu ako dôkaz znovupoužiteľnosti herného engínu.

6. Zahrnúť dôležité technológie a vhodne ich rozdeliť do komponentov.

7. Navrhnúť komponent prepájajúci podstatné časti herného engínu, aby kód v používateľovom komponente pracujúci s engínom bol čo najjednoduchší.

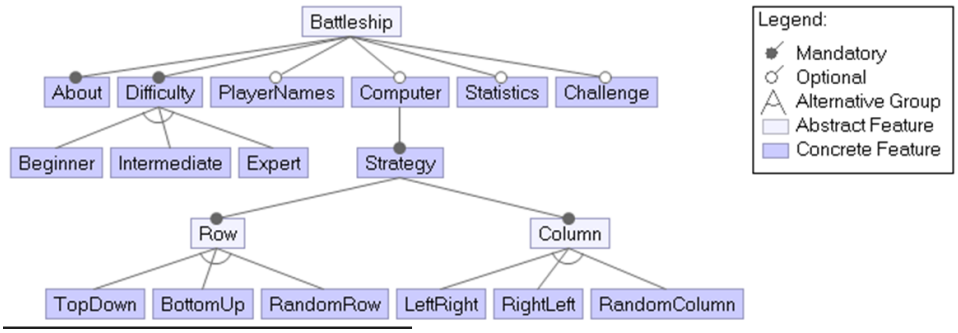
2 Prehľad obsahu sekcií

Sekcia 3 poskytuje charakteristiku jadra herného engínu a popisuje rozdiely medzi ním a samostatnou hrou. Sekcia 5 sa zameriava na analýzu architektúry herného engínu na základe kódu, ktorý obsahuje. Triedy a vzťahy medzi nimi znázorňujeme pomocou nami vytvoreného diagramu tried. Vytvorený diagram balíkov by mal znázorňovať vzťahy medzi balíkmi. V sekcii 6 pri analýze štruktúry rozdeľujeme monolyt na jednotlivé vrstvy. V tejto časti sa snažíme o realizáciu komunikácie len v rámci susedných vrstiev. Dosiahnuť komunikáciu len v jednom smere je vítané. V sekcii 7 sa snažíme oddeliť herný engín od hry už v samotnom návrhu tým, že z engínu vytvárame predpripravený subsystém, ktorý môže byť v prípade potreby nakonfigurovaný podľa potrieb hry. Tvorba hier by preto mala byť jednoduchšia a prepojenia s engínom by mali byť v menšom počte. Sekcia 8 reprezentuje návrh distribučného riešenia pre lepšiu rozšíriteľnosť a tvorbu modulov. SOA by mala

by univerzálnou voľbou kvôli princípu voľnej väzby pri skladaní služieb, ale distributívna architektúra založená na vzdialenom volaní procedúry môže byť rýchlejšia. V tejto časti sa zameriavame hlavne na zmenšenie závislostí medzi 3D matematikou. Sekcia **??** je venovaná realizácii engínu v jazyku Javascript. Snažíme sa poskytnúť informácie o použití podobnej technológie založenej na funkcionalite plátna. Zaoberáme sa komplikáciami a implikáciami preneseného riešenia v súvislosti s architektúrou. Sekcia 9 zhodnocuje vykonanú prácu spolu s opisom hlavných prínosov jednotlivých častí riešenia. Sekcia 10 sumarizuje prehľad ďalších publikácií a vývoja podobných herných engínov. Cieľom je poskytnúť aj informácie o ďalších komponentoch a technológiách pri tvorbe jadra herného engínu softvérového riešenia. Sekcia 11 obsahuje zhodnotenie práce a ďalších možností rozšírenia tejto práce.

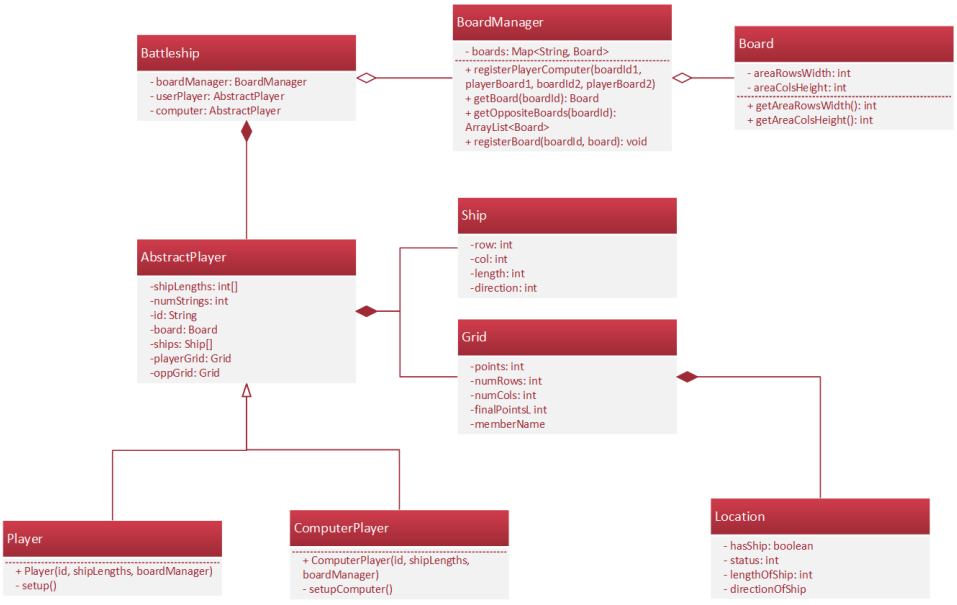
3 Rad softvérových výrobkov a analýza domény

4 Aktuálny stav a návrh rieenia pre hru Battleship

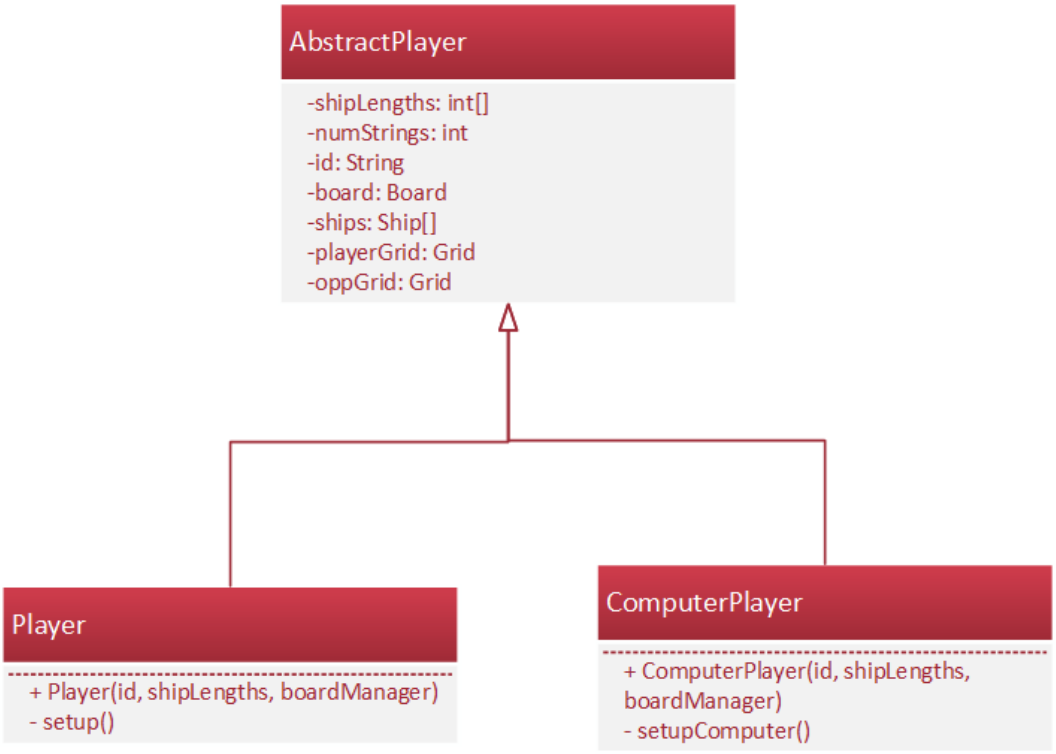


Obr. 1: Diagram vlastností pre hru Battleship

5 Úprava hry Battleship s pouitím jazyka AspectJ

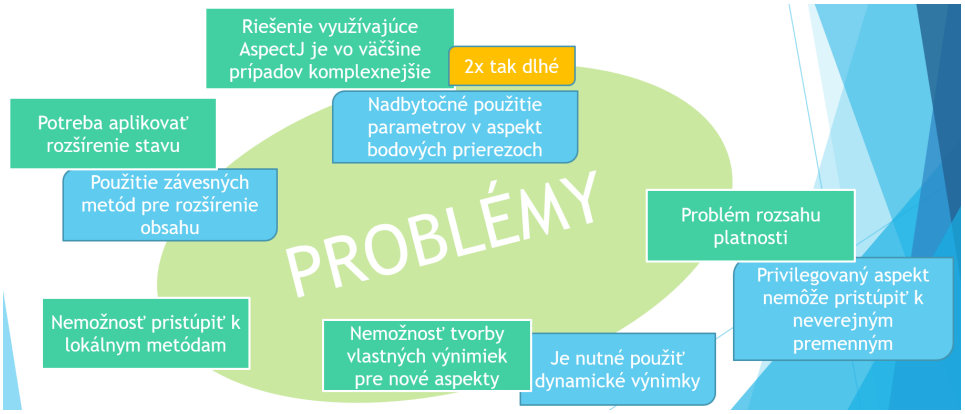


Obr. 2: Diagram tried upraveného rieenia



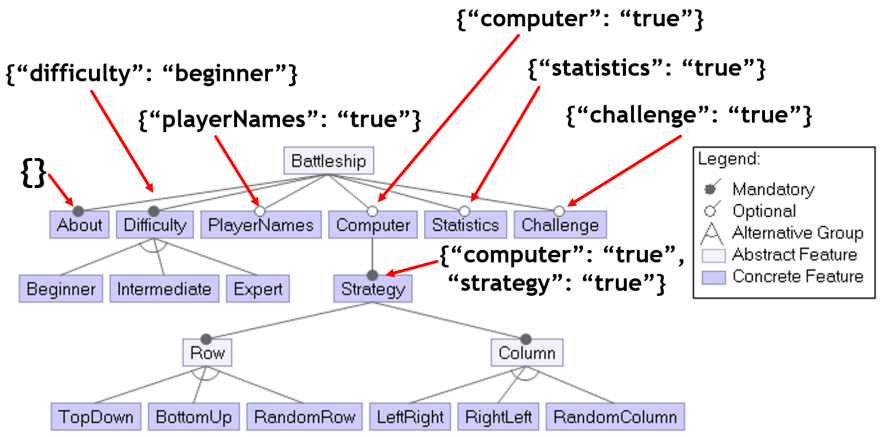
Obr. 3: Abstrakcia hráa

Po prvej iterácii je diagram balíkov zobrazený na obrázku 4.



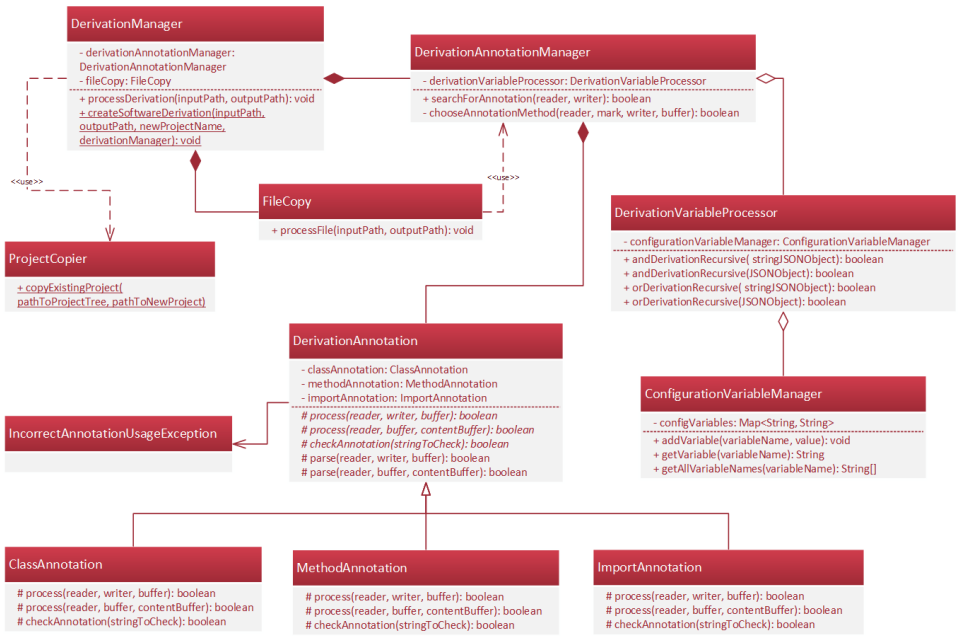
Obr. 4: Problémy pouitia jazyka AspectJ

6 Odvodenie konkrétnej Battleship hry

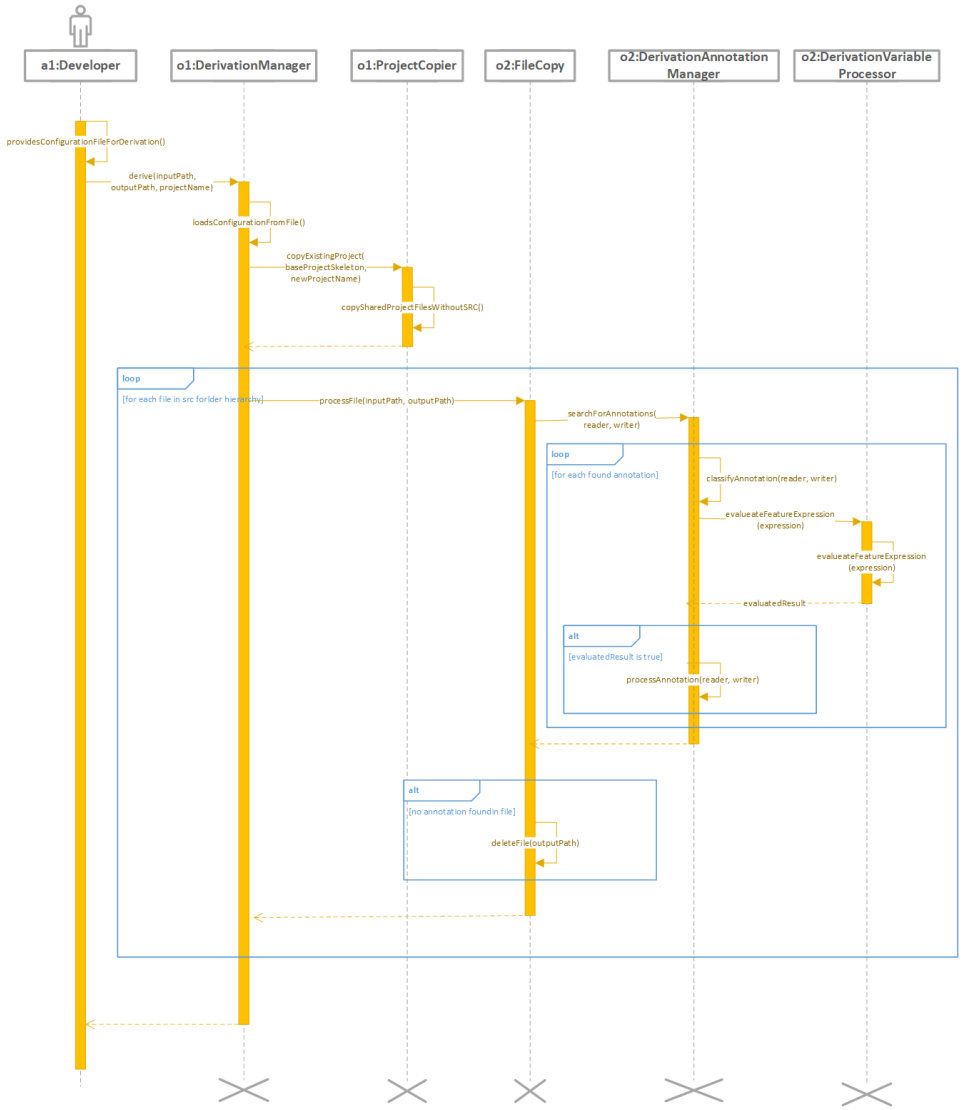


Obr. 5: Podoba derivaných pravidiel

Obr. 6: Zloený derivaný výraz



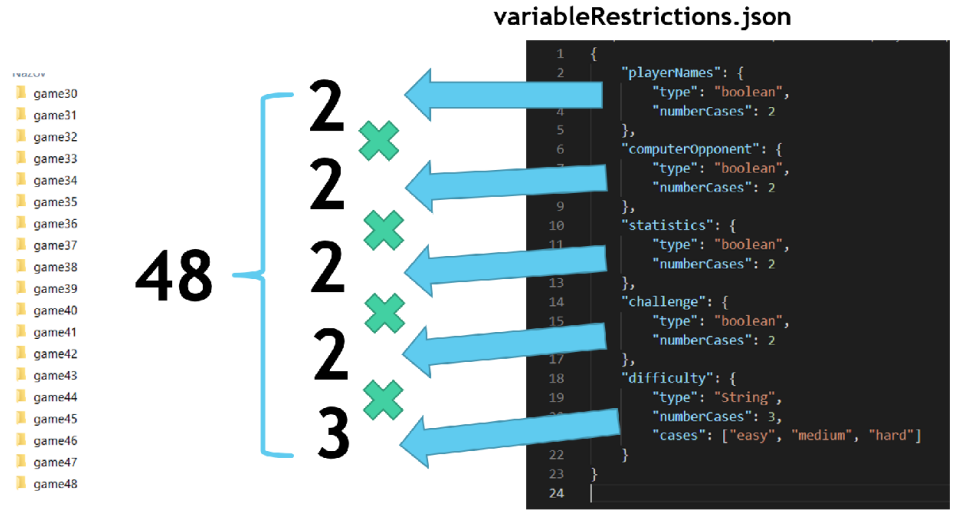
Obr. 7: Diagram tried pre deriváciu produktu



Obr. 8: Sekvenný diagram pre deriváciu produktu

7 Odvodzovanie vetkých typov hier

Aj

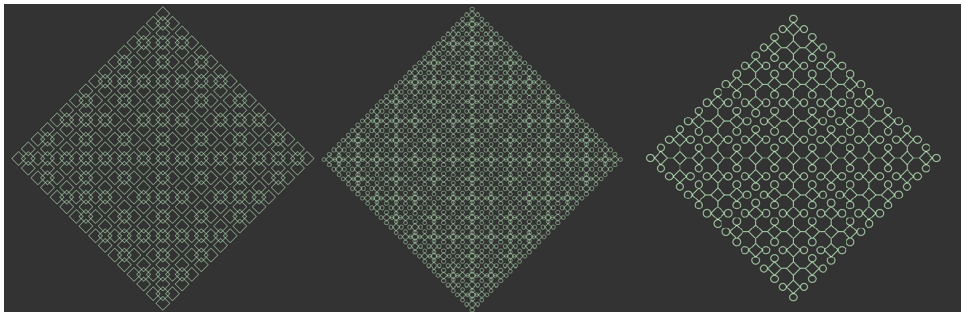


Obr. 9: Odvodenie vetkých typov hier pre hru Battleship

Kompon

8 Rad softvérových výrobkov pre fraktály

SOA vyu



Obr. 10: Rôzne podoby Anklet fraktálu

Riešenie

9 Výsledná modularita a odvodené produkty

Po postupn

9.1 Úprava funkcionality a pouitie jazyka AspectJ

V návr

9.2 Generovanie derivácií Battleship hry

Vytvorili

10 Podobná práca

Kód kt

11 Závery a ďalšie pokračovanie

Analyzo

Literatúra