Slovenská Technická Univerzita v Bratislave
Fakulta informatiky a informačných technológií

Objektovo-orientované programovanie

Správa o realizácii projektu

Názov a zámer projektu:

CompExpres

Nie každý má vedomosti v oblasti IT technológií a má schopnosť si opraviť alebo postaviť svoj vlastný PC. Preto existujú PC servisy, ktoré tieto pre iných zložité problémy riešia za nich. Zákazník môže jednoducho priniesť/odoslať svoju požiadavku/objednávku a ďalej sa nemusí o nič starať.

V servise túto požiadavku spracuje kompetentný zamestnanec. Zistí, čo zákazník v objednávke požaduje a následne vykoná potrebné kroky, na splnenie objednávky. Pokiaľ bude zákazník požadovať napríklad opravu svojho PC, bude potrebné zistiť, aký je v PC problém.

Preto musí niekto PC skontrolovať a zistiť, čo spôsobuje jeho nefunkčnosť. Po nájdení chyby, bude potrebné túto chybu odstrániť, poprípade nefunkčnú súčiastku vymeniť. Servis preto musí skontrolovať dostupnosť náhradnej súčiastky u seba na sklade, poprípade novú súčiastku objednať od svojho dodávateľa.

Keď bude mať servis novú súčiastku k dispozícií, tak ju môže nainštalovať do PC. Po osadení novej súčiastky v PC, je potrebné v systéme PC vykonať príslušné konfiguračné zmeny. Ďalším dôležitým krokom je otestovanie funkčnosti PC. Či je všetko správne zapojené a či všetky časti PC spolu komunikujú a PC riadne otestovať záťažovými testami.

Pokiaľ by niekde v procese opravy nastala chyba, tak sa celý proces opravy musí zopakovať. Ale pokiaľ PC prejde záverečným testovaním, tak bude pripravený na odovzdanie zákazníkovi. Servis môže okrem opráv PC poskytovať aj iné služby, spojené s údržbou elektronických zariadení, pri ktorých môže byť proces opravy jednoduchší alebo zložitejší (napr. odvírenie PC, skladanie PC, ...).

Opis funkcionality programu:

Program slúži na organizáciu výrobného procesu (oprava/skladanie PC) v servise. V programe je možné simulovať vytvorenie objednávky na opravu PC alebo poskladanie nového PC. Celý proces opravy/skladania PC spúšťa manager servisu po prihlásení sa do svojho profilu. Následne môže vybrať objednávku na spracovanie a konkrétneho zamestnanca. Má možnosť zvoliť proces manuálnej, alebo automatickej simulácie výrobného procesu.

Pri manuálnej simulácii výrobného procesu môže nastať situácia, že jeden zo zamestnancov nedokončí svoj úkon. Vtedy, pokiaľ nie je dostupný iný zamestnanec tej istej špecializácie, nie je možné spracovať nové objednávky ďalej, ako po výrobný úkon tohto zamestnanca. Vtedy treba manuálne uvoľniť zamestnanca tým, že sa treba prihlásiť do jeho profilu s jeho používateľským meno a heslom (*poskytnuté nižšie*) a vykonať pracovný úkon manuálne, alebo vytvoriť nového zamestnanca s takou istou špecializáciou.

Pri automatickej simulácií sa celý proces výroby vykoná automaticky. Môže však nastať situácia, že servis nebude mať dostatok zamestnancov, resp. jeden zo zamestnancov nedokončí svoj úkon a tak sa výroba zastaví u jedného z nich. Vtedy treba manuálne priať nového zamestnanca, aby bolo možné pokračovať vo výrobnom procese.

Manager má za úlohu vybrať konkrétnu objednávku a spustiť proces výroby buď ako automatickú simuláciu alebo ako manuálnu simuláciu s konkrétnym zamestnancom

Diagnostik má za úlohu označiť (diagnostikovať) chybné komponenty v PC

Konfigurátor má za úlohu vybrať/objednať nové/náhradné komponenty zo skladu

Montážnik má za úlohu demontovať všetky chybné komponenty z pc a namontovať do PC všetky nové/náhradné komponenty

Softvérový Technik má za úlohu nainštalovať všetok požadovaný softvér do PC

Tester má za úlohu otestovať funkčnosť všetkých komponentov v PC

Mená a heslá pre základných zamestnancov servisu

```
Manager( store: this,  name: "CompExpress",  surname: "Manager",  userName: "admin",  userPassword: "admin");

 servisu
 DiagnosticTechnician( store: this,  name: "Peter",  surname: "Presny",  userName: "d",  userPassword: "d"));
 ConfigurationTechnician( store: this,  name: "Roman",  surname: "Rychly",  userName: "c",  userPassword: "c"));
 AssemblyTechnician( store: this,  name: "Zdeno",  surname: "Zrucny",  userName: "a",  userPassword: "a"));
 SoftwareTechnician( store: this,  name: "Stano",  surname: "Slusny",  userName: "s",  userPassword: "s"));
 TestingTechnician( store: this,  name: "Tomas",  surname: "Tupy",  userName: "t",  userPassword: "t"));
```

Diagram tried:

- Priložený v samostatnom súbore (*MichalikL_uml.svg*)

Vysvetlenie vzťahov:

- "Hlavný" objekt typu ***Store,*** ktorý reprezentuje servis, obsahuje (vlastní) svojich zamestnancov, ktorí sú uložení v *ArrayListe* ***Employees***. Servis taktiež má svojho ***Managera***, ktorý celý servis spravuje (spúšťa proces vybavenia objednávky, prijíma nových zamestnancov). Servis má ďalej priradený sklad ***Warehouse***, v ktorom skladuje komponenty ***Component*** do počítačov rôznych typov. Pokiaľ sa v procese výroby požaduje nejaký konkrétny typ komponentu, ktorý sa na sklade nenachádza, tak sa tento komponent objedná.

- Servis prijíma objednávky, ktoré sa ukladajú do *ArrayListu* ***Orders***. Na vytvorenie objednávok slúži rozhranie *CreateOrder*, pomocou ktorého sa pri podľa typu objednávky príslušná objednávka aj s požadovanými atribútmi vytvorí a zapíše sa do *ArrayListu* ***Orders.***

- Typ objednávok sa delí na 2 typy. Prvým je objednávka na nový PC (***NewPc***), kedy sa pri tvorbe objednávky zvolí typ výkonu nového PC a pomocou rozhrania *CreateOrder sa* určia špecifikácie systému. Následne na základe týchto špecifikácií servis musí vo výrobnom procese vyskladať nový PC tak, aby im vyhovoval. Druhým typom objednávky je oprava PC (***BrokenPc***). Vtedy sa pomocou rozhrania *CreateOrder* vygeneruje náhodná konfigurácia PC s náhodnou chybovosťou komponentov. Úlohou servisu je potom tieto chybné komponenty identifikovať, odstrániť a nahradiť novými, ktoré budú vyhovovať pôvodnej konfigurácii systému.

- Samotný proces výroby/skladania PC spúšťa samotný ***Manager*** servisu pomocou metódy *startProduction(Order order)* a začína podľa typu objednávky pri konkrétnom type zamestnanca (***Employee***). Pri type objednávky na opravu PC (***BrokenPc***) proces začína u Diagnostika (***DiagnosticTechnician***) a pri type objednávky na nový PC (***NewPc***) proces začína u Konfigurátora (***ConfigurationTechnician***). Každý typ zamestnanca má v sebe implementovanú metódu *doTask(),* pomocou ktorej vykoná príslušné korky na realizáciu vybavenia objednávky, teda každý typ zamestnanca ma za úlohu niečo iné. Každý zamestnanec po vykonaní svojej práce musí objednávku (***Pc***) posunúť príslušnému kolegovi tak, aby nenarušil proces výroby. Na to slúži rozhranie *MooveOrder,* ktoré obsahuje 2 rôzne metódy *moovePc(..).* Prvá slúži na posúvanie objednávky (***Pc***) medzi jednotlivými zamestnancami pri automatickej simulácii celého procesu výroby. Druhá, ktorá má o jeden argument viac (***Employee*** *to*), slúži na posúvanie objednávky (***Pc***) medzi jednotlivými zamestnancami pri manuálnej simulácií procesu výroby, kedy je nutné, aby sa daný zamestnanec prihlásil do systému a vykonal príslušné korky výrobného procesu manuálne a následne posunul objednávku (***Pc***) ďalšiemu zamestnancovi.

- Proces opravy/skladania PC sa končí u Testera PC (***TestingTechnician***), ktorý po úspešnom otestovaní PC proces ukončí a pri neúspešnom otestovaní PC vráti naspäť príslušnému zamestnancovi.

- Proces výroby pri novom PC
    - ***Manager -> ConfigurationTechnician -> AssemblyTechnician -> SoftwareTechnician -> TestingTechnician***

- Proces výroby pri novom PC
    - ***Manager -> DiagnosticTechnician -> ConfigurationTechnician -> AssemblyTechnician -> SoftwareTechnician -> TestingTechnician***

Splnenie kritérií hodnotenia (hlavné):

- Program v súlade so zadaním a zámerom projektu, ako aj pokynmi vyučujúceho, s adekvátne použitým dedením a polymorfizmom v aspoň dvoch oddelených hierarchiách dedenia, vrátane použitia rozhraní, a s korektným zapuzdrením a vhodným použitím agregácie.
- Dôsledné oddelenie aplikačnej logiky od používateľského rozhrania
- Kód vhodne organizovaný do balíkov
- Prehľadná dokumentácia

Splnenie kritérií hodnotenia (vedľajšie):

- Ošetrenie mimoriadnych stavov prostredníctvom vlastných výnimiek
- Poskytnutie grafického používateľského rozhrania oddelene od aplikačnej logiky a s aspoň časťou spracovateľov udalostí vytvorenou manuálne (Triedy ***HomeControllerNew*** v balíku *Controllers* a ***HomeScreenNew*** v balíku *View*)
- Explicitné použitie RTTI
- Použitie lambda výrazov alebo referencií na metódy
- Použitie implicitnej implementácie metód v rozhraniach
- Použitie serializácie

Hlavné verzie programu v GitHube:

- **Pracovná verzia 1** (druhé odovzdanie)
  - Implementovaná časť základnej funkcionality
  - Z časti vypracované Gui
  - Žiadne komentáre

- **Pracovná verzia 2**
  - Implementovaná kompletná funkcionalita
  - Kompletne vypracované Gui
  - Žiadne komentáre

- **Pracovná verzia 3**
  - Zakomentovaná aplikačná časť
  - Drobné úpravy v Gui

- **Finálna verzia**
  - Lepšie zorganizované balíky
  - Dokončené ošetrenia
  - Pridané komentáre
  - Drobné upravy