Bajnógel Tamás
C4KNIO

Mintamodellek aspektus-orientált modellezéshez

Összegzés

Rövid elméleti összefoglaló

Léteznek olyan programozási feladatok, amelyek a rendszer egészére vonatkoznak és átívelik az objektum-orientált programegységek határait. Az ilyen típusú problémák megoldására született meg az aspektus-orientált programozás. Az AOP manapság egyre népszerűbb, ezt bizonyítja, hogy több konkrét nyelv és keretrendszer is megjelent. Az AOP-t rendkívül széles körben lehet alkalmazni: naplózás, hitelesítés, hibakezelés, hibatűrés stb.
A feladat fő célja rendszerek szolgáltatásbiztonság analízisének segítése, oly módon, hogy az alaprendszerhez könnyen hozzáilleszthessünk különböző hibatűrő sémákat, aspektusokat. Ezzel a megoldással a rendszer hibatűrő változatainak analízise könnyen végrehajtható válik.

Félév során elvégzett munka

Először megismerkedtem az AO paradigmával, majd egy konkrét AO nyelvvel, az AspectJ-vel. Ezután az AOP-t hibatűrés megvalósítása szempontjából vizsgáltam. A hibatűrést külön csomagokban érdemes megvalósítani a nagyfokú újrafelhasználhatóság érdekében. Ezekben a csomagokban a jól ismert hibatűrő struktúrákat célszerű létrehozni: NVP, RB, NSCP.
Olyan feladatokat kellett kitalálnom, ahol a hibatűrés megvalósítása észszerű. Ezután a kitalált rendszerek modelljét UML osztály-diagrammon ábrázoltam. A feladatoknak két modellje készült el: egy eredeti modell és egy hibatűrő modell, amelyet az eredeti modell és a hibatűrő csomagok összeszövésével kaphatunk. A modellrajzoláshoz Rational Rose-t használtam. A félév során két modell készült el:

1. Kliens-szerver üzenetküldő rendszert
A kliens egész számokat küld azonosítóval a szervernek egy csatornán keresztül. A szerver a kapott számokat négyzetre emeli, majd a kliensnek visszaküldi a kapott eredményt. Az elkészült modellt az alábbi módon lehet hibatűrővé tenni: az üzenetküldést kliens oldalon RB használatával, a számítást szerver oldalon NVP használatával. Az elkészült eredeti modellt Java nyelven implementáltam. Fejlesztő eszközként Borland JBuilder-t használtam. A hibatűrés aspektusát AspectJ nyelven valósítottam meg.

2. Gépkocsi ABS és légzsák vezérlése
Ebben a példában a légzsákot egy gyorsulásmérő alapján, a fékeket pedig a gyorsulásmérő és a kerekek forgását monitorozó érzékelők alapján vezéreljük. Az egyes modulokat busz köti össze. Az elkészült modell az alábbi módon tehető hibatűrővé: a buszokat NSCP-vel, az érzékelőket és a gyorsulásmérőt pedig NVP segítségével replikáljuk.

Az elkészült eredeti modellek egy modellszövő tesztbemeneteiként szolgáltak. Ez a szövő automatikusan végzi el a hibatűrő csomagok beszövését a modellbe. A kapott eredményeket ezután a kézi szövés eredményeivel hasonlítottam össze. Az eredmények azonosak voltak, a modellszövő jól működött.
Az automatikus szövés előnye, hogy nagymértékben leegyszerűsödik a különböző hibatűrő modellek szolgáltatásbiztonság analízise, tehát könnyebbé válik az optimális modell kiválasztása.