Nástroj pro podporu vývoje softwarových systémů

V rámci mojí bakalářské práce mám zpracovat nástroj pro podporu vývoje softwarových systémů.
Začal jsem prohlédnutím si čím vynikají ostatní nástroje například Umbrello, zjistil jsem že jednoduchostí a intuitivním ovládáním.
Nejříve jsem tedy začal tím že jsem si rozvrhl rozlozeni programu, jak bude graficky vypadat, jak bude jeho UI a kde budou jednotlivé prvky. Zjistil jsem že nejjednodušši a zároveň nejintuitivnější bude mít pro každý otevřený projekt záložky. V těchto záložkách bude nejlepší mít záložky pro UseCase, pro Diagram tříd a pro OOPN.
Pustil jsem se tedy hned do práce. Naprogramoval jsem záložky pro jednotlivé otevřené projekty, zde jsem zjistil že v rámci jednoduchosti a intuitivnosti bude dobré použít klávesové zkratky pro otevírání a zavíání projektů. Zvolil jsem CTR+T pro otevření nového projektu a CTRL+W pro zavření aktuálního projektu.
Dále mne napadlo, ze bych mohl po vzoru prohlížečů, použít tlačítka pro nový panel a zavření panelu křížkem v pravo. Tento úkol byl poněkud význa, nakonec se mi povedlo přidat jak tlačítko pro přidání nového tabu (projektu) tak i tlačítko pro uzavření projektu. Zavírání jsem musel vylepšit o zapamatování, ktetrý tab se zavřel, ale to již byl detail.

Po provedení tohoto úkolu, jsem opět sedl nad papír a zamyslel jsem se nad designem hlavní části, hlavního obsahu. Napadlo mne že bude nejjednodušší mít stejné rozložení prvků pro každou část aplikace. Zvolil jsem tedy rozdělení takové, že doprostřed jsem umístnil panel na který budu vykreslovat objekty a jejich hrany. Ve spodní části aplikace budou informační věci. A v levé části aplikace se nacházejí tlačítka pro výběr prvku a pro výběr spojení.

Toto jsem naprogramoval pomocí splitPane a tudíž je možnost změny velikosti jednotlivých polí. S tím že levá část aplikace je rozdělena na 2 panely, tlačítka pro přidávání prvků a tlačítka pro aktivaci spojování míst (hrany).

Tlačítka jsem použil zaklikávací a ty jsem přidal do skupin. Tudíž př kliknutí na přidání například aktora v UseCase modelu, je vidět že uživatel přidává aktora a při kliknutí na další tlačítko se odkline aktor. Toto se mi zdálo jako brilantní řešení.

Pro ukládání jednotlivých prvků jsem zvolil arrayList, v tomto arrayListu uchovávám jednotlivé objekty. Tento objekt co je uložen je otec pro další objekty. Tyto objekty mají společné některé vlastnosti, například souřadnice, barvy a další proměnné. Tudíž nejjednodušší bylo vytvořit třídu, která bude toto obsahovat. Hodilo se mi to také díky tomu, že jsem tuto třídu mohl rovnou ukládat do arrayListu a kontrolovat jenom typ objektu později, při různých akcích s objekty.

Udělal jsem tedy 2 tyto pole, jedno obsahující objekty na ploše a druhé obsahující spojení mezi těmito objekty. Toto řešení je velice elegantní v rámci s použitím grafiky, jednoduše projedu tyto pole a vykreslím daný objekt na souřadnicích. Zde jsem zjistil že budu potřebovat kreslit různé typy objektů. To jsem tedy vyřešil, tak že volám metodu v každém objektu o kreslení a tato metoda se již postará o vykreslení daného objektu, tvaru.

V momentě, kdy jsem začal kreslit objekty, jsem začal řešit jejch překrývání. Toto jsem vyřešil, tak že kontroluji zda se na místě, kde uživatel klikne nenachází objekt a pokud ne, uložím ho do pole a poté vykreslím.

Dále jsem přidal možnost přesunu objektů po ploše a jejich mazání. Mazání jsem provedl, tak že vyberu objekt a zmáčknu tlačitko DEL na klávesnici. V tomto momentě odstraním daný objekt z pole a již ho nevykresluji.

Nejnáročnější část byla pravděpodobně vyřešit kreslení spojů. Pokud je aktivováno spojování objektů a kliknu na objekt si ukládám do objektu pro hrany jaké objekty jsou spojeny. Respektive první objekt, objekt ze kterého hrana vychází. Druhý objekt, objekt na který hrana ukazuje. Toto

řešení nabízí možnost kontroly, zda se daná hrana dá vykonat a zda je validní. Například při spojování UseCase a aktora je validní pouze směr z UseCase, dále nelze spojit Asociací 2 UseCase. A mnoho dalších pravidel. Všechny tyto věci mohu kontrolovat rovnou při ukládání daného přechodu. A pokud narazím na chybu upozorním uživatele. Upozornění na chybu jsem zatím ještě neudělal.

Kontrolu validity hran řeším v průběhu aplikace. Momentálně v části UseCase nemám vyřešenu validitu hran. Ale implementace dané funkčnosti by neměla být naročná.

V momentě, kdy jsem byl schopen spojit 2 objekty jsem začal řešit, jak odstranit hranu. Jak zvýraznit hranu pod myší a jak ji přesunout k jinému objektu. Nejdříve jsem tedy vyřešil zjištění hrany pod myší. Použil jsem tedy obecný vzorec úsečky, odvozený od obecného vzorce přímky.

Pravděpodobně nejnáročnější částeditoru. Hledání nejlepšího vzorce a následná implementace byla opravdu náročná. Nicméně nepochybuji o tom, že toto jistě přepracuji na mnohem elegantnější řešení.

Jakmile jsem dokázal zjistit hranu pod myší, tak jsem přidal aktivaci dané hrany. Toto bylo velice jednoduché. Smazání aktivované hrany a přesun jsem implementoval v zápětí. Přesun je implementován kliknutím a držením levého tlačítka myši. Přetáhnutím hrany a puštěním pod objektem, na který potřebuji aby hrana ukazovala.

Mazání hrany jsem opět provedl, tak že jsem si vyhledal aktivovanou hranu a tu jsem odstranil z pole. Při kreslení se již tato hrana nekreslila.

Následně jsem implementoval při mazání objektu také smazání hran, které ukazují z něj na další objekty.

Také při kliknutí na objekt jsem zvýraznil hrany, které ukazují z něj na další objekty.

V momentě, kdy jsem měl tedy mazání základních hran a kompletní editaci jsem udělal také šipky. Nejnáročnější na tomto úkolu bylo, aby šipka „obíhala“ kolem objektu. Chtěl jsem, aby šipka opisovala tvar objektu, pokud budu měnit její směr. Zde jsem opět využil obecný vzorec pro úsečku.

Tímto byla část pro UseCase hotová. Implementace dalších částí nebude náročná, jelikož veškerou logiku mám již hotovou.

Budu muset vyřešit překreslování hlavního panelu, pokud umístním objekt mimo aktivní plochu. Nyní pouze z části, nepovažuji to za nutnou část editoru. Převážně kvůli tomu, že editor je pouze v ranném stádiu. Nicméně ve finální aplikaci bude jistě potřeba změny velikosti pracovní plochy.

Momentálně se tedy moje práce nachází ve stádiu, kdy mám hotový UseCase a Diagram tříd rozpracovaný.

Další práce se bude skládat z dokončení Diagramu tříd a udělání OOPN. Nejdříve udělám tyto části odděleně, tudíž nebudou spolu nijak komunikovat. Jakmile budou hotovy jednotlivé editory, tak je prováži mezi sebou. Toto docílím tím že objekty mám uloženy v poli a jejich spoje v dalším poli. Provázání tedy nebude příliš náročné.

V momentě, kdy budu mít hotovy editory, které se budou navzájem upravovat a budou navzájem provázány provedu nastavení. Rád bych měl intuitivní aplikaci a hlavně použitelnou, tudíž bude jistě potřeba nastavení, ať již hran, tvaru objektů a barev. Tak celkový vzhled aplikace.

Během práce jsem několikrát narazil na problém se špatným návrhem a musel jsem nekolikrát předělat aplikaci a přemístit metody. Při dalším postupu budu tedy postupně předělávat většinu tříd a jejich implementaci.