Animácie modelov softvérových architektúr

# Úvod

## Dôvody vzniku tohto dokumentu

Katalóg požiadaviek vznikol ako záväzný dokument obsahujúci požiadavky a ich jednoznačný opis pred a počas tvorby projektu Animácie modelov softvérových architektúr od zadávateľa vývojárom.

## Rozsah využita systému

Rozširujeme funkcie systému AnimArch, ktorý dokáže vizualizovať vzťahy medzi jednotlivými triedami v triednom diagrame. V prvom rade systém slúži ako pomôcka pri výučbe softvérového inžinierstva, presnejšie architektúry softvérových systémov. V druhom rade je systém nástrojom pre výskum fúzie statického a dynamického modelu class diagramu. Môže to byť aj nástroj pri analýze a návrhu architektúry softvérového systému.

## Definície, pojmy a skratky

**Class diagram** - model, ktorý popisuje štruktúru systému zobrazením tried a vzťahov medzi nimi.

**Fúzia** - spojenie, prepojenie, spojiť sa

**xUML** -​ executable UML​; jazyk pre tvorbu vykonateľných modelov softvéru

**OAL (Object Action Language)** - platformovo nezávislý jazyk kompilovateľný na platformovo závislé jazyky

**EA** ​- ​Enterprise Architect​; softvér pre modelovanie softvéru v UML

**GUI** - ​Graphical User Interface​; používateľské rozhranie pozostávajúce z interaktívnych vizuálnych prvkov

**AST** ​- ​Abstract Syntax Tree​; abstraktná reprezentácia výrazov v danom jazyku

**API**-​Application Programming Interface; rozhranie poskytované aplikáciou pre komunikáciu s inými aplikáciami

**VS**​ - ​Visual Studio;​ vývojové prostredie pre jazyk C#

## Odkazy

<https://xtuml.org/> - edukácia executive UML, know how OAL

## Obsah dokumentu

Druhá kapitola obsahuje všeobecný opis systému rozdelený do nasledovných oblastí:

2.1. všeobecná perspektíva systému

2.2. Funkcie systému

2.3. charakteristiku používateľa

2.4. všeobecné obmedzenia systému

\v tretej kapitole sa nachádza zoznam požiadaviek, ktoré sú rozdelené do lblokov, ktorých sa dané požiadavky týkajú

# Všeobecný popis

## Perspektíva systému

Aplikácia AnimArch poskytuje samostatnú metódu animácie softvérových architektúr a scenárov. Operuje nad diagramom tried, ktorý animuje na základe príkazov zapísaných v modifikovanom jazyku OAL. Jazyk umožňuje vytvárať jednoduché scenáre (postupnosť animačných krokov), alebo sofistikované scenáre pomocou podmienok, cyklov, paralelizmu, objektov, premenných a výpočtov nad výrazmi. Aplikácia umožňuje pracovať s diagramami vytvorenými v EA priamo pomocou kliknutia v EA.

## Funkcie systému

Systém vizualizuje vzťahy medzi jednotlivými triedami v class diagrame. Pracuje s modelmi class diagramov, ktoré boli vytvorené a následne exportované z externých aplikácií (napr. Enterprise Architect). Systém k danému class diagramu vytvorí object diagram v pomere 1:1 tak, že vytvorí inštanciu každej triedy a následne s týmito inštanciami pracuje. Na tvorbu vizualizácie/animácie ponúka systém skript napísaný v jazyku OAL. Skript môže používateľ napísať sám alebo ho systém vygeneruje interaktívnou formou (užívateľ klikne na triedy a metódy, ktoré chce vizualizovať). Po spustení skriptu prebehne animácia, ktorej dĺžku a výzor(farebná schéma) si volí užívateľ. Animácia prebieha naraz v samotnom class diagrame zvýraznením tried, metód a vzťahov medzi triedami a taktiež aj v skripte zvýraznením jednotlivých príkazov. Systém dokáže skript exportovať do textového súboru(.txt).

## Charakteristiky používateľa

Systém rozpoznáva 2 roly používateľov:

**vývojár animácie** - človek, ktorý vytvára skript v jazyku OAL, ktorý animuje class diagram pre používateľa (kolega, študent)

**používateľ animácie** - človek, ktorý systém využíva na učebné účely, alebo analýzu navrhovaného, prípadne existujúceho class diagramu

## Všeobecné obmedzenia

Jediné obmedzenie, ktoré stanovil zadávateľ je, že systém musí bežať na operačnom systéme, ktorý podporuje C#, Unity, Enterprise Architect (alebo ekvivalentný softvér na tvorbu diagramov) a funkčnosť vo virtuálnej realite.

# Požiadavky

## Implementácia debug módu pre vývojára

* + 1. Developer môže importovať class diagram z externého zdroja (EA).
    2. Developer môže nad otvoreným diagramom spúšťať vykonanie animácií.
    3. Developer môže vytvoriť animáciu vyklikaním v používateľskom rozhraní (UI) prototypu (skript sa vygeneruje automaticky).
    4. V používateľskom rozhraní je prítomný text OAL skriptu.
    5. Developer môže vytvoriť animáciu napísaním konkrétneho skriptu v jazyku OAL.
    6. Skript je spustený stlačením tlačidla “Play” v užívateľskom rozhraní.
    7. Animácia je znázornená vysvietením tried, ich metód a vzťahov, resp. asociácií medzi nimi.
    8. Animácia je znázornené zvýraznením práve vykonávaných príkazov v OAL skripte.
    9. Vykonávaný príkaz je zvýraznený zmenou farby fontu alebo zmenou farby pozadia.

## Implementácia debug módu pre analytika / používateľa

* + 1. Doplniť možnosť definovať metódu a jej telo v OAL skripte.
    2. Na definovanie tiel metód je potrebné upraviť jazyk OAL, aby toto povoľoval.
    3. Pokúsiť sa o možnosť definovať telá metód v samostatných jednotkách (GUI okná, textové súbory, ...) - vyplynie z pokusov pri implementácii.
    4. Vytvorenie debug okna stavov, kde sa bude zobrazovať názov aktuálne animovanej triedy, obsah jej jednotlivých atribútov a názov spúšťanej metódy
    5. Jednotlivé triedy (resp. ich inštancie) majú atribúty a ich hodnoty.
    6. Pri vyvolaní metód sa môžu meniť ich atribúty, preto je užitočné vizualizovať tieto zmeny.
    7. Zobrazenie inštancie triedy v GUI ako tabuľku atribútov a ich hodnôt pre danú triedu, resp. jej inštanciu.
    8. Vizualizovať názov práve spúšťanej metódy, resp. príslušnej triedy.

# Prílohy

<https://github.com/TIS2020-FMFI/uml-3d/blob/master/prilohy/OAL_manual.pdf> - OAL manual

<https://github.com/TIS2020-FMFI/uml-3d/blob/master/prilohy/dokumentacia_prototypu.pdf> - dokumentácia prototypu