მოდიფიცირების მექანიზმების მოდელირება ობიექტ-ორიენტირებული დაპროექტებისას

თეიმურაზ სუხიაშვილი
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

მართვის ორგანიზაციული სისტემების დინამიკურობა განაპირობებს დროთა განმავლობაში დამუშავებული საავტომატიზაციო პროგრამული სისტემის შეცვლის აუცილებლობას. სტატიაში განიხილება მოდიფიცირების მექანიზმების მოდელირება ობიექტ-ორიენტირებულ სისტემებში პაკეტების დიაგრამის აგებით და მოდელებს შორის ტრასირების მიმართებების დადგენით.

საკვანძო სიტყვები: პაკეტი. კლასი. მოდელი. მოთხოვნა. კომპონენტი. დიაგრამა. კვანძი.

1. შესავალი

ორგანიზაციული სისტემების არსებითი მხარეა მათი დინამიკურობა, ევოლუციური განვითარება. დროთა განმავლობაში იცვლება თვით ობიექტის სტრუქტურა, მისი შემცველი ელემენტების ფუნქციები. ევოლუციური განვითარება ერთის მხრივ დაკავშირებულია სისტემის გარეთ მომხდარ ცვლილებებთან, ხოლო მეორეს მხრივ - შინაგანი თვითორგანიზაციით და სრულყოფით. საპრობლემო სფეროს ცვლილება მოითხოვს დამუშავებული პროგრამული სისტემის შეცვლას. იმისათვის, რომ თავიდან ავიცილოთ ხელმეორედ შრომითი დანახარჯი, დაკავშირებული უკვე მოქმედი სისტემის სტრუქტურისა და ქცევის შესწავლაზე, მათ ამსახველ მოდელებს შორის კავშირებისა და განლაგების განსაზღვრაზე საჭიროა გათვალისწინებულ იქნას მოდელების შეცვლის შესაძლებლობა დაპროექტებისა და რეალიზების ეტაპებზე.

სისტემის არქიტექტურის კონტექსტში საპრობლემო გარემოს ცვლილება იწვევს სისტემისადმი მოთხოვნების შეცვლას. შესაბამისად, საჭირო ხდება ყოველი შეცვლილი მოთხოვნის რეალიზების მექანიზმის მოდიფიცირება. რეალიზების მექანიზმები წარმოიდგინება კოოპერაციებით [2], რომელსაც აქვს სტრუქტურული (კლასები) და ქცევითი (ურთიერთქმედება კლასებს შორის) შემადგენელი. მოთხოვნები, ისევე როგორც მათი რეალიზების მექანიზმები, სემანტიკურად დაკავშირებულია. გამორიცხული არ არის საპრობლემო გარემოს რომელიმე ელემენტი მონაწილეობდეს რამდენიმე რეალიზების მექანიზმში – კოოპერაციაში. შესაბამისად, ერთი რეალიზების მექანიზმის ცვლილებამ შესაძლებელია გავლენა იქონიოს სხვაზე.

2. ძირითადი ნაწილი

თანამედროვე მიდგომით რთული ორგანიზაციული სისტემების ავტომატიზაცია ხორციელდება კომპიუტერული ქსელის გამოყენებით, განაწილებული სამუშაო ადგილებით, მონაცემთა კლიენტ-სერვერული არქიტექტურით. შესაბამისად, სისტემის კორექტირებისათვის უნდა დადგინდეს ის მოთხოვნები, რომლებიც ექვემდებარება შეცვლას და ამ მოთხოვნების რეალიზების მექანიზმები ლოგიკურ დონეზე, ხოლო შემდეგ ეტაპზე დადგინდეს მათი ფიზიკური რეალიზება – კომპონენტები და მათი განლაგება სერვერებზე.

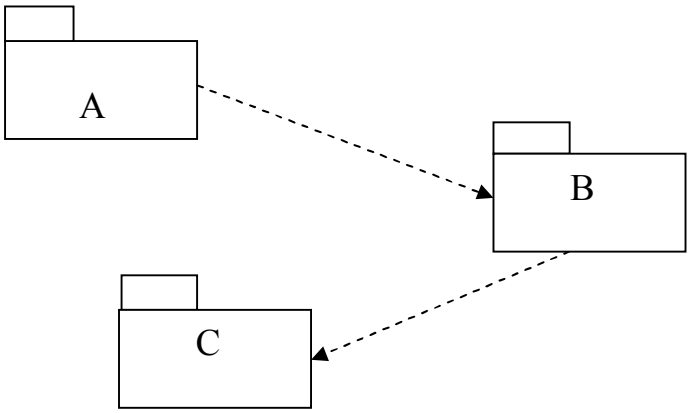
ყოველი მოთხოვნის რეალიზებაში მოწილეობს კლასების გარკვეული ჯგუფი. მოვახდინოთ ერთ კოოპერაციაში მონაწილე კლასების დაჯგუფება პაკეტში [1]. ყოველი პაკეტისათვის განვსაზღვროთ, რომელი ელემენტები უნდა იყოს გარედან მისაწვდომნი. მოვნიშნოთ ისინი როგორც გახსნილი, ხოლო დანარჩენი – როგორც დაცულები ან დახურული. შევაერთოდ პაკეტები იმპორტის მიმართებით იმ პაკეტებთან, რომლებზეც არის ისინი დამოკიდებული.

ორ ელემენტს შორის არსებობს დამოკიდებულება, თუ ცვლილებამ მის ერთ ელემენტში შესაძლებელია გამოიწვიოს ცვლილება მეორეში. რაც შეეხება კლასებს, დამოკიდებულების ხასიათი შეიძლება იყოს სხვადასხვა: ერთი კლასი უგზავნის შეტყობინებას მეორეს; ერთი კლასი მოიცავს მეორე კლასს, როგორც თავისი მონაცემების ნაწილს; ერთი კლასი მიმართავს მეორეს, როგორც რომელიმე ოპერაციის პარამეტრს. თუ კლასი იცვლის ინტერფეისს, მაშინ ნებისმიერი შეტყობინება, რომელსაც ის აგზავნის, შეიძლება აღმოჩნდეს მცდარი. იდეალურ შემთხვევაში მხოლოდ ცვლილებები კლასის

ინტერფეისში უნდა ახდენდეს გავლენას სხვა კლასებზე. დიდი სისტემების დაპროექტების ხელოვნება მოიცავს თავისში დამოკიდებულებების მინიმიზაციას; ასეთ შემთხვევაში ცვლილებების გავლენა მცირდება, ხოლო სისტემის ცვლილებისათვის საჭირო იქნება ნაკლები გარჯა.

ორ პაკეტს შორის არსებობს დამოკიდებულება, თუ არსებობს გარკვეული დამოკიდებულება მასში არსებულ ნებისმიერ ორ კლასს შორის. მაგრამ, კლასებისაგან განსხვავებით პაკეტებს შორის დამოკიდებულებას არ გააჩნია ტრანზიტულობა ე.ი. თუ A პაკეტი დამოკიდებულია B-ზე, ხოლო B პაკეტი თავის მხრივ დამოკიდებულია C-ზე, ეს არ ნიშნავს, რომ A პაკეტი დამოკიდებულია C-ზე.

როგორც 1-ელი ნახაზიდან ჩანს, თუ იცვლება რომელიმე კლასი C-პაკეტში, ეს არ ნიშნავს, რომ უნდა შევიტანოთ ცვლილებები A-პაკეტში. ეს მხოლოდ აღნიშნავს, რომ უნდა გაისინჯოს, შეიცვალა თუ არა B-პაკეტი. A-პაკეტმა შეიძლება მოითხოვოს ცვლილება მხოლოდ იმ შემთხვევაში, თუ შეიცვლება V-პაკეტის ინტერფეისი. მოცემულ შემთხვევაში B-პაკეტი იცავს A-პაკეტს ცვლილებებისაგან.

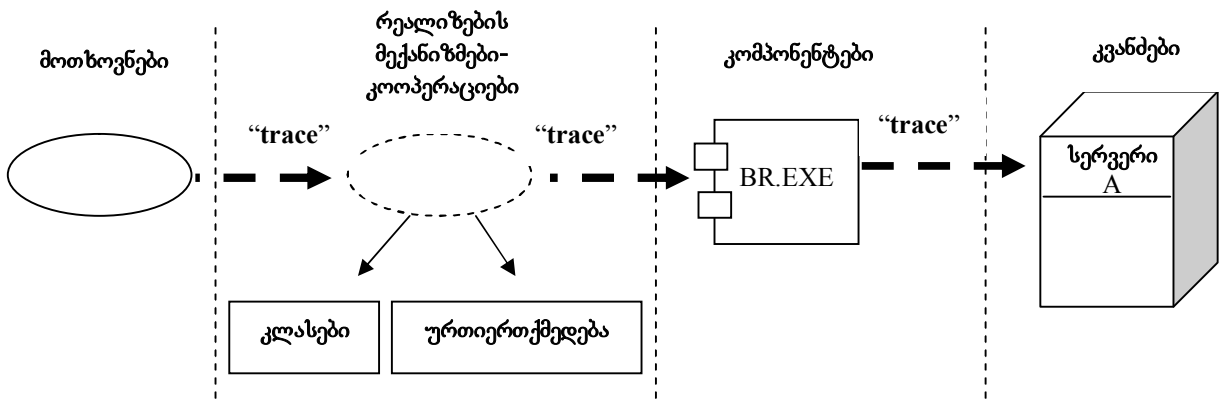


ნახ.1.

კლასები პაკეტში შეიძლება იყოს საერთოდ გახსნილი, დახურული და დაცულები. შესაბამისად, პაკეტი B დამოკიდებულია გახსნილი მეთოდებისაგან გახსნილი კლასისა პაკეტში C. თუ შეიცვლება C პაკეტის ნებისმიერ კლასში რომელიმე დახურული მეთოდი ან C პაკეტის რომელიმე დახურულ კლასში გახსნილი მეთოდი, ეს ცვლილებები არ შეეხება არც ერთ კლასს პაკეტში B.

თუმცა პაკეტები არ იძლევიან პასუხს კითხვაზე, როგორ შევამციროთ დამოკიდებულებების რაოდენობა ჩვენს სისტემაში, მაგრამ ისინი გვეხმარებიან გამოვყოთ ეს დამოკიდებულებები. როგორც კი ისინი გამოჩნდებიან, საჭირო არის მუშაობა მათ შესამცირებლად. ამიტომ თვლიან, რომ პაკეტების დიაგრამა არის ძირითადი საშუალება სისტემის ზოგადი სტრუქტურის მართვისა.

პაკეტების დიაგრამა საშუალებას გვაძლევს მოვახდინოთ მოდელის შეცვლა ლოგიკურ დონეზე. გამოვავლინოთ ის კლასები და მათ შორის ურთიერთქმედება, რომლებიც უნდა შეიცვალოს. იმისათვის, რომ მოვახდინოთ ფიზიკურად შესაბამისი კომპონენტების შეცვლა უნდა ვიცოდეთ გამოვლენილი კლასების მარეალიზებელი კომპონენტების განლაგება კვანძებზე. ამისათვის უნდა ვიცოდეთ კავშირი სისტემის ცალკეულ მოდელებს შორის დაწყებული მოთხოვნიდან, დამთავრებული მისი რეალიზებით.



ნახ.2

სისტემის დაპროექტება ობიექტ-ორიენტირებული მიდგომით ხორციელდება სხვადასხვა დანიშნულების მოდელების გამოყენებით. კონცეპტუალური კავშირების მოდელირება ელემენტებს შორის, რომლებიც სხვადასხვა მოდელებში არიან, შეიძლება განხორციელდეს ტრასირების დამოკიდებულებით. ტრასირება წარმოიდგინება სტერეოტიპული დამოკიდებულებით. ხშირად ყურადღებას არ აქცევენ ასეთი დამოკიდებულების მიმართულებას, თუმცა ჩვეულებრივ ისარი მიუთითებს უფრო ადრეულ ობიექტს. უფრო ხშირად ტრასირების მიმართება გამოიყენება იმისათვის, რომ უჩვენონ გზა მოთხოვნიდან რეალიზაციამდე(იხ.ნახ.2.).

როგორც ნახაზიდან ჩანს მოდელში პრეცედენტები ტრასირდებიან მოდელის კოოპერაციებთან. კოოპერაციები ტრასირდებიან კლასებთან, რომლებიც ერთობლივად ფუნქციონირებენ მოცემული კოოპერაციის განსახორციელებლად. მოდელის კომპონენტები ტრასირდებიან პროექტირების მოდელის ელემენტებთან. კვანძები განლაგების მოდელიდან ტრასირდებიან კომპონენტებთან რეალიზაციის მოდელიდან.

3. დასკვნა

სისტემის მოდიფიცირების მოყვანილი მიდგომა საშუალებას მოგვცემს საგრძნობლად შევამციროთ შესაცვლელი მოდელებისა და მათი რეალიზებადი კომპონენტების გამოვლენის პროცესი. დადგენილი კომპონენტების შეცვლით აღვადგინოთ მოდელისა და საპრობლემო გარემოს შესაბამისობა.

ლიტერატურა:

1. Фаулер М., Скотт К. UML. Основы. Пер. с англ., СПб: Символ-Плюс. 2002
2. სუხიაშვილი თ. რეალიზების მექანიზმების მოდელირება ობიექტ-ორიენტირებული პროექტირებისას. სტუ-ს შრომები, №1(2), 2007. 115-118 გვ..

МОДЕЛИРОВАНИЕ МЕХАНИЗМОВ МОДИФИКАЦИИ ПРИ ОБЪЕКТНО-ОРИЕНТИРОВАННОМ ПРОЕКТИРОВАНИИ

Сухиашвили Т.А.
Грузинский Технический Университет

Резюме

Динамичность организационных систем управления обуславливает необходимость с течением времени изменить разработанную для автоматизации програмную систему. В статье рассматривается моделирование механизмов модификации системы при объектно-ориентированном проектировании построением диаграмы пакетов и установлением отношения трасировки между моделей разных уровней.

MODELING MODIFICATION MECHANISMS IN THE OBJECT-ORIENTED PROJECTING

Sukhiashvili Teimuraz
Georgian Technical University

Summary

Dynamic of organizational management systems stipulates the necessity of alteration of the processed automated program systems in due course. The article discusses modelling modification mechanisms by building package diagrams in object-oriented systems and establishing tracing relations between models.