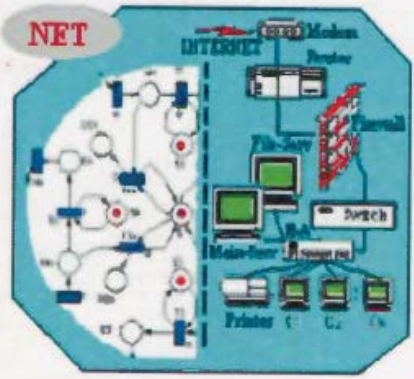
გ. გოგიჩაიშვილი, გ. ბოლხი,
გ. სურგულაძე, ლ. პეტრიაშვილი

მართვის ავტომატიზებული სისტემების ობიექტ-ორიენტირებული დაპროექტების და მოდელირების ინსტრუმენტები

(MsVisio, WinPepsy, PetNet, CPN)



„ტექნიკური უნივერსიტეტი“

საქართველოს ტექნიკური უნივერსიტეტი

გ. გოგიჩაიშვილი, გ. ბოლხი,
გ. სურგულაძე, ლ. პეტრიაშვილი

მართვის ავტომატიზებული სისტემების ობიექტ-ორიენტირებული დაპროექტების და მოდელირების ინსტრუმენტები

(MsVisio, WinPepsy, PetNet, CPN)

დამტკიცებულია სახელმძღვანელოდ:
სტუ-ს სარედაქციო-საგამომცემლო
საბჭოს მიერ 17.05.2006

თბილისი – 2013

უაკ 004.5

გადმოცემულია მას-ის ობიექტ-ორიენტირებული ანალიზის, დაპროექტებისა და მოდელირების მეთოდები და ინსტრუმენტული საშუალებანი UML-ტექნოლოგიის, .NET-პლატფორმის, MsVisio/ Enterprise_Architect, WinPepsy და PetNet/CPN პაკეტების გამოყენებით. სახელმძღვანელოში თანამიმდევრულად განიხილება განაწილებული ორგანიზაციული მართვის სისტემების კომპიუტერული დაპროექტებისა და მოდელირების თეორიული ასპექტები და პრაქტიკული რეალიზაციის საკითხები.

შემოთავაზებულია თანამედროვე ინჟინრული მეთოდები და ინსტრუმენტული საშუალებანი უნიფიცირებული მოდელირების ენის, რიგების თეორიის და პეტრის სისტემური ქსელების კომპლექსურად გამოყენების საკითხების შესახებ.

განკუთვნილია თეორიული და პრაქტიკული ინფორმატიკის და სხვა დარგების სპეციალისტებისათვის, დოქტორანტების, მაგისტრანტებისა და სტუდენტებისათვის.

რეცენზენტები:

- საქ.მეცნ.აკადემიის წევრ-კორესპონდენტი, ტ.მ.დ., პროფესორი **ა. ფრანგიშვილი**

- ტ.მ.კ., პროფესორი **თ. სუხიაშვილი**

პროფ. გ. სურგულაძის რედაქციით

ISBN 99940-56-77-8

Georgian Technical University

**GUNTER BOLCH, GEORG GOGICHAISHVILI,
GIA SURGULADZE, LILY PETRIASHVILI**

TOOLS OF OBJECT-ORIENTED DESIGN AND MODELLING OF AUTOMATED CONTROL SYSTEMS (MsVisio, WinPepsy, PetNet, CPN)

Supported by DAAD (Germany)

Methods and tools of object-oriented analysis, design and modeliing of MIS are discussed using packages of UML technology, .NET platform, MsVisio/Enterprise Architect, WinPepsy and PetNet/CPN. The textbook provides a sequential review of computer design and modeling theoretical aspects as well as their application issues for distributed systems of organizational management. Modern engineering methods and tools are presented about complex use of unified modeling languages, theories of Queuing and Petri Networks. The present textbook is destined for practitioners of theory and applied informatics and also specialists from other fields, doctorands, masters and undergraduate students.

ISBN 99940-56-77-8

ავტორთა შესახებ:

გიორგი გოგიჩაიშვილი: საქართველოს მეცნიერებათა ეროვნული აკადემიის წევრ-კორესპონდენტი, სტუ-ს ინფორმატიკის ფაკულტეტის „ორგანიზაციული მართვის“ დეპარტამენტის უფროსი, მართვის ავტომატიზებული სისტემების მიმართულების სრული პროფესორი, ტექნიკის მეცნიერებათა დოქტორი. მრავალი სამეცნიერო ნაშრომის და წიგნის ავტორი სისტემების მოდელირების, სიტუაციათა ანალიზის, ოპერაციათა კვლევის და ობიექტ-ორიენტირებული ანალიზის სფეროში.

გუნტერ ბოლხი: გერმანიის ნიურნბერგ-ერლანგენის უნივერსიტეტის ინფორმატიკის ფაკულტეტის პროფესორი და აკადემიური დირექტორი. ტექნიკის მეცნიერებათა დოქტორი. მრავალი სამეცნიერო ნაშრომის და წიგნის (მათ შორის გამოცემული აშშ-ში) ავტორი სისტემების მოდელირების და რიგების თეორიის სფეროში. 2001 წლიდან სისტემატურად მონაწილეობდა სტუ-ს მას კათედრის საიუბილეო კონფერენციებში. იყო სტუ-ს საქართველო-გერმანიის ერთობლივი სასწავლო-სამეცნიერო GeoGer-ცენტრის თანადამაარსებელი და ქართველი სტუდენტების მენეჯერი ერლანგენის უნივერსიტეტში.

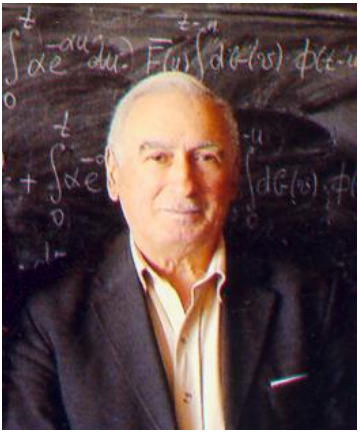
გია სურგულაძე: სტუ-ს ინფორმატიკის ფაკულტეტის „მართვის ავტომატიზებული სისტემების“ მიმართულების ხელმძღვანელი, სრული პროფესორი, ტექნიკის მეცნიერებათა დოქტორი. 55 წიგნის და 200-ზე მეტი სამეცნიერო ნაშრომის ავტორი ინფორმაციულ ტექნოლოგიათა სფეროში. სტუ-ს საქართველო-გერმანიის ერთობლივი სასწავლო-სამეცნიერო GeoGer-ცენტრის დამაარსებელი და აკადემიური დირექტორი. ბერლინის ჰუმბოლდტის და ნიურნბერგ-ერლანგენის უნივერსიტეტების მიწვეული პროფესორი 1991-2013 წლებში.

ლილი პეტრიაშვილი: სტუ-ს „მართვის ავტომატიზებული სისტემების“ კათედრის ასოც.პროფესორი, ტექნიკის მეცნიერებათა კანდიდატი. 30-ზე მეტი სამეცნიერო ნაშრომის და წიგნის ავტორი მართვის საინფორმაციო სისტემების დაპროგარმება-მოდელირების, პეტრის-ქსელების, მონაცემთა საცავების, ელექტრონული კომერციის და ინტერნეტ ბიზნესის სფეროში.

სარჩევი

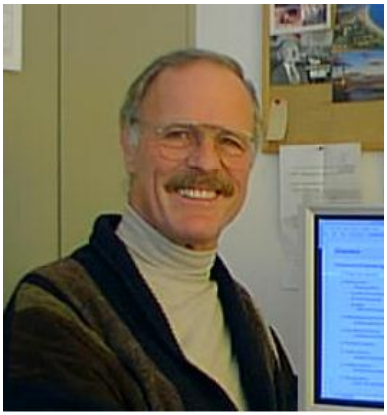
სახელოვანი მეცნიერები:
ილია მიქაძე და გუნტერ ბოლხი

პროფ. ილია მიქაძე (1928-2010) **Prof. Dr. Ilia Mikadze**



დამსახურებული მეცნიერი მასობრივი მომსახურების თეორიაში. გამოთვლითი ტექნიკის და პროგრამული სისტემების დამუშავების, წარმოების და დანერგვის ერთ-ერთი თვალსაჩინო ფუძემდებელი საქართველოში და საზღვარგარეთ. მის მიერ შექმნილი მეთოდები სისტემური ანალიზის, მასობრივი მომსახურების, საიმედოობის თეორიის მნიშვნელოვანი მეცნირული მონაპოვარია. მან ძვირფასი მეცნიერული მემკვიდრეობა დაუტოვა ქართულ საინჟინრო სკოლას ამ მიმართულებით. მისი უშუალო ხელმძღვანელობით მრავალი წარმატებული სადოქტორო და საკანდიდატო დისერტაცია იქნა დაცული. იყო სტუდენტების დიდი ქომაგი და ღირსეული პროფესორი აუდიტორიაში.

Prof. Dr. Gunter Bolch



პროფ. გუნტერ ბოლხი (1940-2008), გერმანიის ნიურნბერგ-ერლანგენის უნივერსიტეტის „ქსელური და ოპერაციული სისტემების" კათედრის აკადემიური დირექტორი, იყო სტუ-ს საერთაშორისო სამეცნიერო ჟურნალ „მართვის ავტომატიზებული სისტემები" რედკოლეგიის წევრი, „გერმანია-საქართველოს ერთობლივი სასწავლო-სამეცნიერო ცენტრის დამაარსებელი, 1991 წლიდან ახალგაზრდა ქართველ მეცნიერ-სტაჟიორთა გულისხმიერი კურატორი ერლანგენის უნივერსიტეტში. მისმა წიგნებმა და კვლევებმა რიგების თეორის, ქსელური სისტემების მოდელირების სფეროში წარმატებული აღიარება მოუტანა. იგი იყო მიწვეული პროფესორი აშშ-ის, ბრაზილიის, კანადის, უნგრეთის და სხვა ქვეყნების უნივერსიტეტებში.

„ ვუძღვნით გამოჩენილი მეცნიერების
ილია მიქაძის და გუნტერ ბოლჰის ნათელ ხსოვნას ”

შესავალი

ორგანიზაციული მართვის საინფორმაციო სისტემების ობიექტ-ორიენტირებული დაპროექტება, მათი ვიზუალური მოდელირება და ანალიზი შემდგომი დაპროგრამებით, თანამედროვე ვიზუალური კომპიუტერული ტექნოლოგიების საშუალებით მეტად მნიშვნელოვანი და აქტუალურია, რამეთუ საგრძნობლად უმჯობესდება პროექტის რეალიზაციის ხარისხი და მცირდება მისი დამუშავების დრო და ხარჯები.

განსაკუთრებით საყურადღებოა დღეს ფირმა მაიკროსოფთის მიერ შემოთავაზებული დაპროგრამების ახალი პლატფორმა დოტ-NET ტექნოლოგიის სახით, რომელიც Windows- და Web-დანართების ასაგებადაა გამიზნული თავისი ახალი ვიზუალურ-ობიექტ-ორიენტირებული დაპროგრამების ინსტრუმენტებით: VB.NET, C#.NET, C++.NET, ADO.NET, ASP.NET, XML, MS VISIO და ა.შ. [1,2,3].

მეორეს მხრივ, მნიშვნელოვანია პროგრამული ინჟინერიის (Software Engineering) ისეთი ინსტრუმენტის ათვისება, როგორიცაა უნიფიცირებული მოდელირების ენა (UML - Unified Modeling Language), ვინაიდან იგი ითვლება კომპიუტერული პროგრამული პაკეტების შექმნის მეთოდოლოგიურ საფუძვლად.

ესაა დაპროგრამების ობიექტ-ორიენტირებულ მეთოდზე შექმნილი თანამედროვე ინფორმაციული ტექნოლოგია, რომელიც არის მოდულების სპეციფიკაციის, კონსტრუირების, ვიზუალიზებისა და დოკუმენტირების ენა და აღნიშვნათა სისტემა. დღეს ამ სტანდარტს იყენებს Microsoft, Oracle, Hewlet-Packard და სხვა ცნობილი ფირმები.

დაპროგრამების თანამედროვე ინსტრუმენტები ინტეგრირებული პაკეტებია, რომლებიც აერთიანებს მონაცემთა აღწერისა და მანიპულირების ენებს (მონაცემთა ბაზის სახით), პროცედურების დამუშავების ხერხებს კლასთა თეორიის გამოყენებით და სტანდარტულ ბიბლიოთეკებს.

ამგვარად, მათში რეალიზებულია ობიექტ-ორიენტირებული დაპროგრამების მეთოდი და სტილი: ინკაფსულაციის, კლასთა მემკვიდრეობითობის, პოლიმორფიზმისა და აბსტრაქციის სახით.

კომპიუტერულ ქსელებში მიმდინარე დინამიკური პროცესების მოდელირებისა და ანალიზისათვის აქტუალურია პეტრის ქსელების ინსტრუმენტის შესწავლა. მისი დახმარებით აიგება მართვის სისტემის აქტიურობათა და მდგომარეობათა დიაგრამები (Activity and State Diagrams UML-ში), კომპიუტერული ქსელების პროცესების მართვის მოდელი კონფლიქტური სიტუაციების აღმოფხვრის მიზნით და ა.შ.

დიდი მნიშვნელობა ენიჭება სისტემური პეტრის ქსელებით აგებულ მოდელებში მასობრივი მომსახურების მეთოდების გამოყენებას, რათა შესაძლებელი გახდეს ასეთ განაწილებულ სისტემებში რაოდენობრივი ანალიზის ჩატარება. ასეთი სისტემების ანალიზისათვის ნაშრომში განიხილება სხვადასხვა ტიპის მათემატიკური მოდელები, კერძოდ M/M/1, M/M/m, M/G/1, G/M/1, G/G/1 და სხვა. ჩვენთვის განსაკუთრებით საყურადღებოა M/M/m ტიპის მოდელის ანალიზი, რაც ნიშნავს, რომ შემავალი ნაკადი უმარტივესია (მარკოვულია), მომსახურების დრო ექსპონენტური კანონით განაწილებული შემთხვევითი სიდიდეა (მარკოვული).

შემოთავაზებულია აღნიშნული საკითხების თეორიულ-პრაქტიკული ასპექტები. კონკრეტული საპრობლემო სფეროს მაგალითზე განიხილება მართვის საინფორმაციო სისტემის ობიექტ-ორიენტირებული ანალიზის, მოდელირების, დაპროექტებისა და პროგრამული რეალიზაციის ამოცანები.

წიგნის ბოლო თავი ეხება პეტრის ქსელების, როგორც მათემატიკური მოდელირების ინსტრუმენტის გამოყენებას სამუშაო ნაკადებისა და ბიზნეს-პროცესების მოდელირების, სიმულაციისა და ვერიფიკაციის ეტაპებზე, რაც მეტად აქტუალური და მნიშვნელოვანია პროგრამული ინჟინერიის ამოცანების გადასაწყვეტად. შემოთავაზებულია კლასიკური პეტრის ქსელების აგების და ანალიზის ინსტრუმენტი PetEdit, აგრეთვე მაღალი დონის პეტრის ფერადი ქსელების სიმულატორი CPN, რომლის გამოყენების დინამიკა განსაკუთრებით მაღალია ამერიკის, ჩინეთის და ევროპის მოწინავე უნივერსიტეტებსა და ბიზნესის სგფეროში.

I თავი

ინფორმატიკა, პროგრამული ინჟინერია და მართვის ავტომატიზებული სისტემები

გადმოცემულია ინფორმატიკის, როგორც კომპლექსური, ინტერდისციპლინარული მეცნიერების არსის, მისი სტრუქტურული კომპონენტების ანალიზის, თანამედროვე მდგომარეობისა და განვითარების ტენდენციათა საკითხები. განიხილება მართვის ავტომატიზებული სისტემების როლი და ადგილი ინფორმაციულ ტექნოლოგიებში, მათი მოდელირების, დაპროექტების და პროგრამული რეალიზაციის ამოცანები. თანამედროვე პროგრამული პლატფორმები და ენები, რომლებიც ფართოდ გამოიყენება აშშ-სა და ევროპის უნივერსიტეტებში. შემოთავაზებულია ის ძირითადი სამეცნიერო მიმართულებები და შედეგები, რომლებიც სტუ-ს ინფორმატიკის და მართვის სისტემების ფაკულტეტზე ავტორთა მიერ გერმანელ კოლეგებთან ერთად იქნა მიღებული.

1.1. ინფორმატიკა, როგორც მეცნიერება და მისი მიმართულებები

ინფორმატიკულ მეცნიერებათა კავშირი საზოგადოების განვითარების დონესთან აშკარაა. რაც უფრო მაღალია საზოგადოების ინფორმატიზაციის დონე, მით უფრო სრულყოფილია მისი მენტალიტეტი, მით უფრო ადაპტირებადია იგი სწრაფადცვლად გარემოში. კომპიუტერული ტექნიკა და ტექნოლოგიები ამ გარემოს აქტიური კომპონენტებია. მათი ცოდნა კი ხშირად განმსაზღვრელია ახალგაზრდობის შრომითი დასაქმების სფეროში. ამიტომაც, გასაკვირი არაა ის დიდი მოთხოვნილება, რომელიც დღეისათვის არსებობს პრაქტიკული და გამოყენებითი ინფორმატიკის, კერძოდ კი კომპიუტერული სისტემებისა და ტექნოლოგიების მცოდნე საინჟინრო კადრებზე.

სტუ-ს ინფორმატიკის ფაკულტეტი (ადრე სპი, ავტომატიკა და გამოთვლითი ტექნიკა) ყოველთვის იდგა პოსტსაბჭოური უნივერსიტეტების მოწინავე ინსტიტუტების რიგში. კავკასიაში „მართვის ავტომატიზებული სისტემების“ კათედრა პირველად

თბილისში შეიქმნა 1971 წელს და ამ ხნის განმავლობაში გამოუშვა მრავალი მაღალკვალიფიციური სპეციალისტი [1].

ამჟამად ჩვენი მიზანი სტუ-ს, და კერძოდ ინფორმატიკისა და მართვის სისტემების ფაკულტეტის და მისი სპეციალობათა მიმართულებების სასწავლო-სამეცნიერო პროცესის ხარისხის სრულყოფაა, ბოლონიის კონვენციის შესაბამისად ევროპის საგანმანათლებლო სივრცეში ჩასართველად.

განსაკუთრებული მნიშვნელობა ამ პროცესში თანამედროვე ტექნიკურ ბაზას, უახლესი საინფორმაციო ტექნოლოგიების სწავლების მოდულების შექმნას, შესაბამისი სასწავლო-მეთოდური ლიტერატურის ფორმირებას, და რაც მთავარია, პროფესიონალი ლექტორების შერჩევას აქვს. გამოსაშვები საინჟინრო სპეციალისტების ხარისხი, რა თქმა უნდა, ბევრად იქნება დამოკიდებული სასკოლო განათლების და ეროვნული გამოცდების შედეგებით ჩარიცხულ სტუდენტებზე (კარგ პროდუქციას კარგი ნედლეული სჭირდება).

კომპიუტერული მეცნიერება კომპლექსური, საგანთშორისი დისციპლინაა. იგი აერთიანებს საკვლევი ობიექტის, მაგალითად, მართვის პროცესების შინაარსობრივ აღწერას (სემანტიკური მოდელირება), მათი გადაწყვეტის ალგორითმული სქემების აგებას (ლოგიკური მოდელირება) და კომპიუტერის „ენაზე“ ამ უკანასკნელთა რეალიზაციას (ლინგვისტური მოდელირება). ეს საკითხები მჭიდრო კავშირშია „კომპიუტერულ დაპროგრამებასთან“ (როგორც მეცნიერული, ასევე აკადემიური თვალსაზრისით), რაც დასმული თემატიკის კვლევის ობიექტი და საგანია.

ინფორმატიკა მეცნიერებაა, რომელიც შეისწავლის ინფორმაციის სისტემატიზებულ დამუშავებას გამოთვლითი ტექნიკის საშუალებით. ისტორიულად იგი ჩამოყალიბდა მათემატიკური მეცნიერების (გამოყენებითი მათემატიკა) განვითარების საფუძველზე, გამოთვლითი ტექნიკა კი სათავეს ელექტროტექნიკის, მიკროელექტრონიკისა და კავშირგაბმულობის ტექნიკის საფუძველზე იღებს [2].

ტერმინი „ინფორმატიკა“ ევროპაში 70-იანი წლებიდან იხმარება. იგი პირველად გერმანიაში, დრეზდენის სამეცნირო კონფერენციაზე იქნა მიღებული რუსი და ფრანგი მეცნიერების

ინიციატივით. ამერიკის და სხვა ქვეყნების ინგლისურენოვან ლიტერატურაში მას შეესაბამება Computer Science and Information Systems. გამოთვლითი ტექნიკა ინფორმატიკაში ინსტრუმენტის როლს ასრულებს. ცნობილი ჰოლანდიელი მეცნიერი ე. დიეკსტრა წერდა, რომ ,,კომპიუტერი იგივეა ინფორმატიკაში, რაც ტელესკოპი ასტრონომიაშიო".

ინფორმატიკას სამი ძირითადი განშტოება აქვს: თეორიული ინფორმატიკა, პრაქტიკული ინფორმატიკა და ტექნიკური ინფორმატიკა. მათ საფუძველზე იქმნება გამოყენებითი ინფორმატიკის მიმართულებები, მაგალითად, ეკონომიკური ინფორმატიკა, ბიოინფორმატიკა, გეოინფორმატიკა, ლინგვისტიკა და ა.შ.. ევროპის ქვეყნებში, მაგალითად გერმანიის უნივერსიტეტებშიც ინფორმატიკის ინსტიტუტები (ან დეპარტამენტები) აღნიშნულ მიმართულებათა კათედრებს აერთიანებს [3].

თეორიული ინფორმატიკა შეისწავლის ფორმალურ ენათა თეორიას. მაგალითად, სისტემური ანალიზი და რთული სისტემების თეორია, სიმრავლეთა თეორია და ლოგიკა, ავტომატებისა და გრაფთა თეორია, პეტრის ქსელები, პრედიკატების აღრიცხვა და რელაციური ალგებრა, ფორმალური სემანტიკა და კატეგორიალური ანალიზი, ოპერაციათა კვლევა, ეკონომიკურ-მათემატიკური მოდელირების მეთოდები, მასობრივი მომსახურების თეორია და ა.შ. ყოველივე ეს ინფორმატიკის ფორმალურ ხერხემალს წარმოადგენს.

პრაქტიკული ინფორმატიკა ემსახურება ინფორმატიკის სფეროს კონკრეტული პრობლემების გადაწყვეტას, განსაკუთრებით კომპიუტერული დაპროგრამების განვითარებას პროგრამული უზრუნველყოფის ტექნოლოგიებისთვის (Software Engineering). აქ მნიშვნელოვანია დაპროგრამების ენები, ოპერაციული სისტემები, მონაცემთა და ცოდნის ბაზების მართვის სისტემები [4]. იგი გამოიმუშავებს ძირითად კონცეფციებს ისეთი სტანდარტული ამოცანების გადასაწყვეტად, როგორიცაა ინფორმაციის შენახვა და მართვა მონაცემთა სტრუქტურების საშუალებით. მნიშვნელოვანი ადგილი უჭირავს აქ მანქანურ ალგორითმებს, რომლებიც რთული და ხშირადგამოყენებადი ამოცანების ავტომატიზებულ გადაწყვეტას

ემსახურება. პრაქტიკული ინფორმატიკის ცენტრალური და მუდამ აქტუალური თემაა რთული გამოყენებითი სისტემების (Windows- და Web-აპლიკაციების) აგების პროგრამული ტექნოლოგიების შექმნა და განვითარება. ესაა სტრუქტურული, ობიექტ-ორიენტირებული და ვიზუალური დაპროგრამების მეთოდები, უნიფიცირებული მოდელირების ენა (UML) და მათი ავტომატიზებული დაპროგრამების რეალიზაციის ინსტრუმენტული საშუალებანი [5].

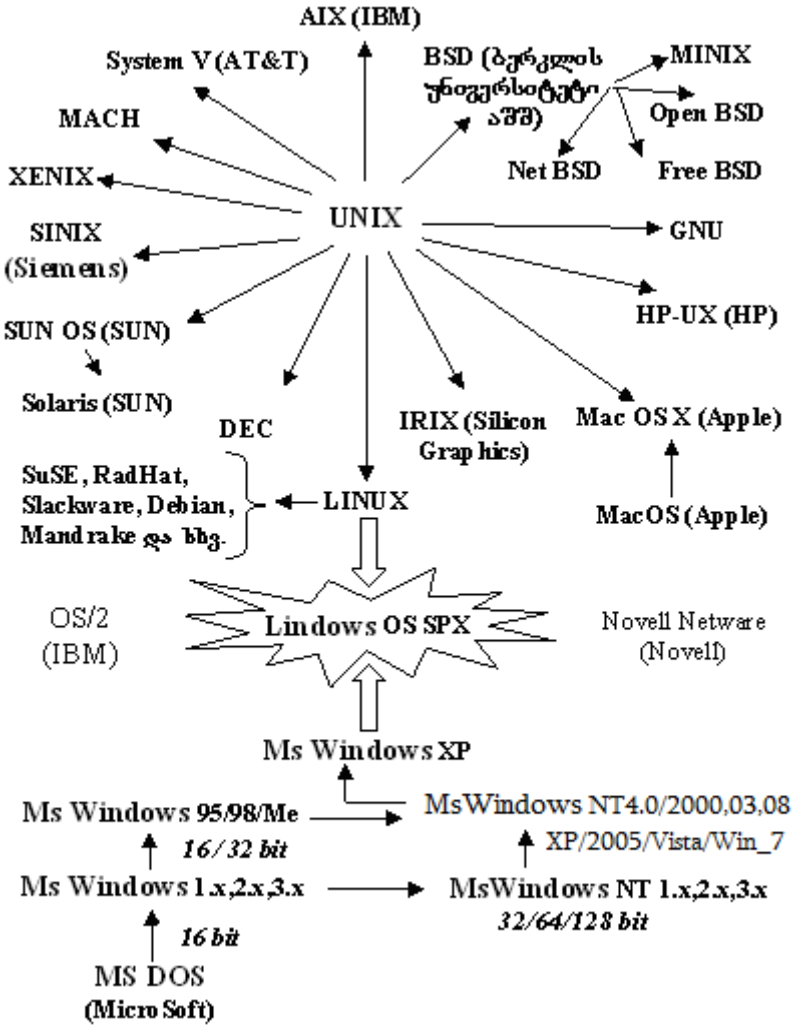
ტექნიკური ინფორმატიკა შეისწავლის ინფორმატიკის ტექნიკური უზრუნველყოფის (Hardware) საფუძვლებს, როგორიცაა მიკროპროცესორული ტექნიკა, კომპიუტერული არქიტექტურები, ქსელური და კომუნიკაციური სისტემები, კონტროლერები და პერიფერიული მოწყობილობანი, რობოტოტექნიკური და სენსორული სისტემები და ა.შ. იგი უშუალო კავშირშია ელექტროტექნიკასთან, განსაკუთრებით ციფრულ ტექნოლოგიებთან, აგრეთვე ლოგიკასა და დისკრეტულ მათემატიკასთან, გადამრთველ სქემათა თეორიასთან. ბოლო წლებში განსაკუთრებული ყურადღება ექცევა მულტიმედიალური ტექნოლოგიების შექმნას და განვითარებას, რაც კომპიუტერული აუდიო-ვიზუალური სისტემების აგების საფუძველია [2].

გამოყენებითი ინფორმატიკა ფართო სპექტრის მეცნიერებაა. იგი ეყრდნობა თეორიულ, პრაქტიკულ და ტექნიკურ ინფორმატიკათა მიღწევებს და შეისწავლის მათ პრაგმატულ გამოყენებას სხვადასხვა დარგების (ეკონომიკა და ბიზნესი, იურისპრუდენცია, ენერგეტიკა, მრეწველობა, ტრანსპორტი, მედიცინა, სოფლის მეურნეობა, განათლება, ენათმეცნიერება და სხვ.) რთული ტექნოლოგიური პროცესების კომპიუტერიზაციისა და ინფორმაციული საცავების შექმნისა და ადმინისტრირებისათვის. გადაწყვეტილებათა მიღების ხელშემწყობი კომპიუტერული სისტემები ყოველი დარგის აუცილებელი ინსტრუმენტი ხდება. ექსპერტული სისტემები მნიშვნელოვან როლს ასრულებს თანამედროვე დიაგნოსტიკისა და პროგნოზის ამოცანების გადასაწყვეტად.

1.2. პროგრამული პლატფორმები და დაპროგრამების ენები

ინფორმაციული ტექნოლოგიების საფუძველია პროგრამული პლატფორმები და დაპროგრამების ენები. განსაკუთრებული მნიშვნელობა აქვს პროგრამულ პლატფორმებს – ოპერაციულ სისტემებს, რომლებზეც დაშენებულია გამოყენებითი კომპიუტერული აპლიკაციები. ტერმინი ,,პროგრამული პლატფორმა'' შედარებით ახალია და იგი ,,აპარატულ პლატფორმასთან'' ასოცირდება. იგი განსაკუთრებით მაშინ გამოიკვეთა, როცა ფირმების Sun Microsystems და MicroSoft კონცეფციები შეეჯახა ერთმანეთს. ძირითადად ორი პროგრამული პლატფორმა (ოპერაციული სისტემა) Unix და Windows NT გახდა კონკურენტები სისტემური პროგრამების ბაზარზე [6]. Unix-ის მიმდევრებია: SunOS/Solaris, IBM AIX, HP-UX (Hewlett-Packard), SINIX (Siemens), IRIX (Silicon Graphics) და სხვ. (ნახ.1.1) ,,ასაკოვანი'' Unix გამოირჩევა მაღალმწარმოებლურობითა და საიმედოობით, რაც შედარებით ,,ახალგაზრდა'' Windows-ს ჯერ არ ახასიათებს. სამაგიეროდ Unix სისტემა რთულია ადმინისტრირების თვალსაზრისით და მოითხოვს მაღალი რანგის კვალიფიკაციის (ძვირადღირებულ) სპეციალისტებს.

დღეისათვის მოთხოვნილებანი სპეციალისტებზე სწორედ პლატფორმებისა და დაპროგრამების ენების ცოდნის კრიტერიუმებით განისაზღვრება. პროგრამული სისტემების მსოფლიო ბაზარზე ყველაზე პოპულარულია ქსელური პლატფორმები Unix (სტაბილური სისტემა დიდი და სუპერ-მანქანებისათვის), Windows (NT, XP, 200x - პერსონალური კომპიუტერებისათვის) და Linux (ახალი პლატფორმა, როგორც Unix-ვარიანტი PC-მანქანებისთვის) [6]. მნიშვნელოვანი პროგრესია მაიკროსოფთის მიერ ბოლო წლებში Windows Vista და Windows-7 სისტემების გამოშვება.

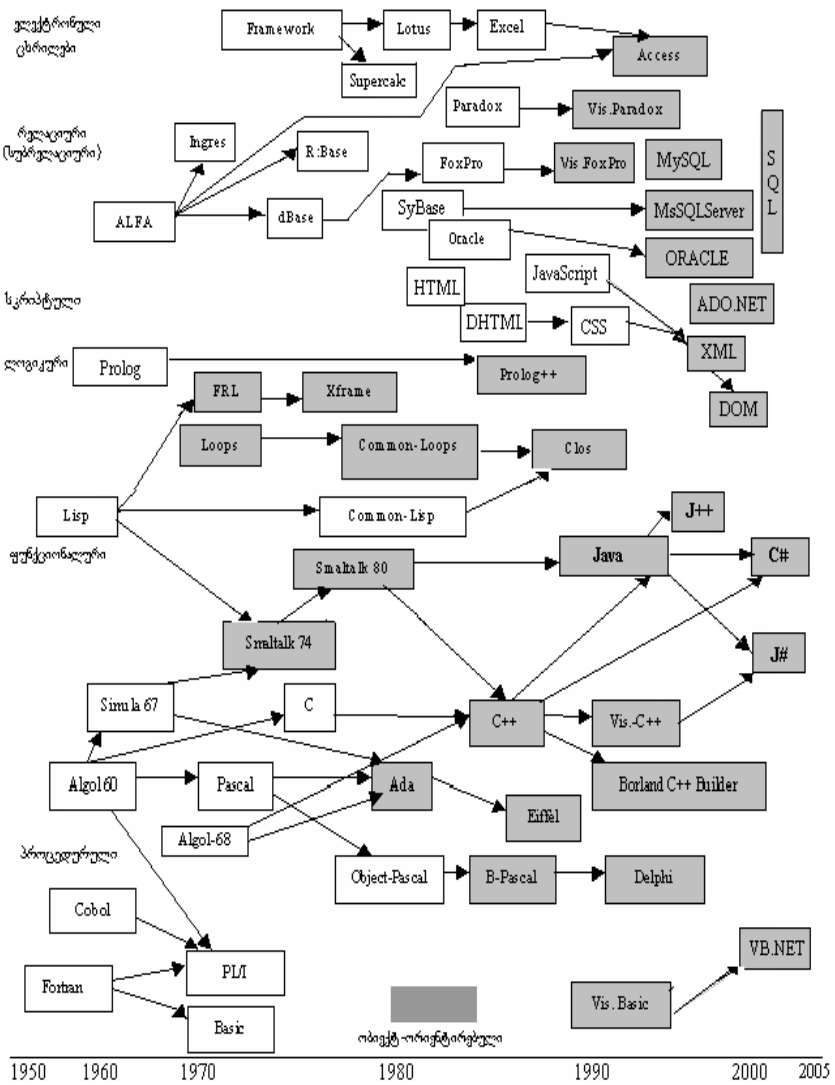


ნახ.1.1.პროგრამული პლატფორმების განვითარების ტენდენციები

ობიექტ-ორიენტირებული დაპროგრამება ერთ-ერთი აქტუალური და მძლავრი მეთოდოლოგიური საშუალებაა თანამედროვე ინფორმაციულ ტექნოლოგიებში. მისი მიზანია დიდი და რთული პროგრამული სისტემების კონსტრუირება. იგი თვისებრივად ახალი კონცეფციების მატარებელი დაპროგრამების ტექნოლოგიაა სისტემების ობიექტ-ორიენტირებული ანალიზისა და ობიექტ-ორიენტირებული დაპროექტების მეთოდებითა და რეალიზაციის მოქნილი ინსტრუმენტული საშუალებებით [5,7]. ამ მიმართულების ერთ-ერთი დამაარსებელია ტეხასის უნივერსიტეტის პროფესორი, დანიელი ბიარნ სტროუსტრუპი, რომელმაც ჩამოაყალიბა ობიექტ-ორიენტირებული დაპროგრამების ძირითადი თეორიული საფუძვლები: კლასები და ობიექტები, მონაცემთა აბსტრაქტული ტიპები, მემკვიდრეობითობა და პოლიმორფიზმი და ა.შ. [8]. მისი სახელმძღვანელო C++ ენის შესახებ პირველი იყო, რომლითაც ისწავლებოდა ამერიკის, ევროპისა და საქართველოს ტექნიკური უნივერსიტეტის შესაბამის სპეციალობებზე [22].

დღეისათვის მსოფლიოში 2000-მდე დაპროგრამების ენაა შექმნილი და მათი განვითარება ჯერაც არ დამთავრებულა (ნახ.1.2) ამ ენებმა გარკვეული როლი შეასრულა თანამედროვე კომპიუტერული ტექნოლოგიების შექმნისა და განვითარების საქმეში. დაპროგრამების ენების კლასიფიკაცია მათში რეალიზებული მეთოდებისა და სტილის თვალსაზრისით ხორციელდება: უნივერსალური პროცედურული, ფუნქციონალური, ლოგიკური, სკრიპტული, ობიექტ-ორიენტირებული, ვიზუალური და ა.შ. [12,14].

ვინაიდან ამერიკის და ევროპის, ასევე მსოფლიოს სხვა განვითარებულ ქვეყნების გამოთვლით სისტემათა ქსელებში UNIX ოპერაციული სისტემა უცვლელად დომინირებს (1975 წლიდან დღემდე), ხოლო ეს უკანასკნელი, როგორც ცნობილია C–ენაზეა დაწერილი (1972 წ.), ამიტომაც განსაკუთრებული პრაქტიკული ღირებულება აქვს **C->C++->Java->C#** ენათა შესწავლას მომავალი პროფესიონალი ინჟინერ-სისტემოტექნიკოსებისა და სისტემური პროგრამისტებისათვის.



ნახ.1.2. დაპროგრამების ენების განვითარების ტენდენციები

მონაცემთა ბაზების მართვის სისტემები კომპიუტერული დაპროგრამების ენათა მნიშვნელოვანი, რელაციებზე ორიენტირებული ორიგინალური კლასია, რომელიც ინგლისელი მეცნიერის, ედგარ-ფრანკ კოდის (გარდაცვალებამდე-2003 მოღვაწეობდა ამერიკაში, ინფორმატიკის სფეროში უდიდესი ღვაწლისთვის 1981 წ. მიენიჭა ტიურინგის პრემია) მონაცემთა მანიპულირების ენის ALFA-ს პროექტიდან იღებს სათავეს [17,19]. სისტემები MsAccess და MsSQL Server, ფირმა Oracle (სამივე ობიექტ-ორიენტირებული განაწილებული რელაციური ბაზების მართვის სისტემებია, რომლებიც დღეს ფართოდ გამოიყენება) და ა.შ.

მონაცემთა მანიპულირების ენები, რომლებიც რეალიზებულია რელაციურ მონაცემთა ბაზებში, როგორიცაა ალგებრული ენა ISBL (Information System Base Language), შეკითხვების ენა ეკრანული რედაქტორით QBE (Query By Example), მოთხოვნების სტრუქტურირებადი SQL ან SEQUEL ენები, რომლებიც ალგებრულ და აღრიცხვის ენებს შორის მდგომი ენებია და ა.შ. მეტად აქტუალური მიმართულებაა [12,15]. საინფორმაციო სისტემების მონაცემთა ბაზების ასაგებად ყოველთვის გამოიყენება არსთა-დამოკიდებულების მოდელი (Entity-Relationship-Model, ERM/SERM), რომელიც ასევე საპრობლემო სფეროს კონცეპტუალური (სემანტიკური) მოდელის სახელითაცაა ცნობილი [23]. ამ საკითხების გაფართოებითა და განვითარებით შეიქმნა **ცოდნის ბაზების** მიმართულება, რომელიც ფართოდ გამოიყენება ხელოვნური ინტელექტის და ექსპერტულ სისტემებში [12,23,30].

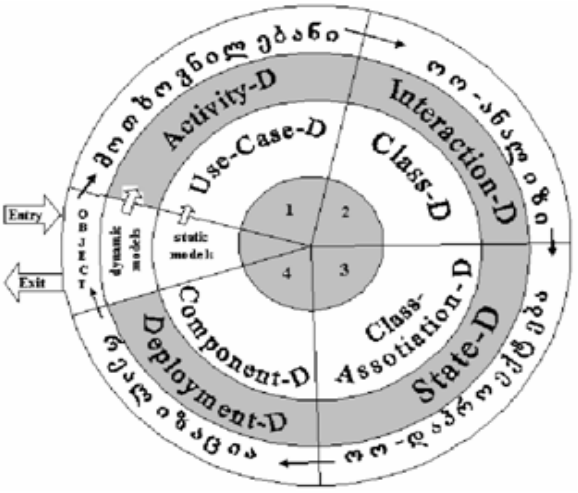
დაპროგრამების ტექნოლოგიების (Software Engineering) განვითარებასა და პოპულარიზაციას ხელი შეუწყო NATO-ს ეგიდით 1968/69 წ. ჩატარებულმა კონფერენციამ „Software Engineering“. დღეს კი Computer Aided Software Engineering (CASE) ტექნოლოგიები უდავოდ შეიძლება ჩაითვალოს კომპიუტერული დაპროგრამების ენებისა და ინსტრუმენტების უმაღლეს მწვერვალად.

კომპიუტერული ტექნოლოგიების შემდგომმა განვითარებამ, ახალი თაობის მანქანების (Pentium-4 პროცესორების ბაზაზე) და სუპერ მონიტორების შექმნამ დასაბამი მისცა მძლავრი გრაფიკული

საშუალებების განვითარებას. დაიძრა ახალი ტალღა დაპროგრამების ისტორიაში, .NET-ტექნოლოგიების სახით: Visual-C#, Visual-C++, Visual-J++, Visual-Basic.NET, ADO.NET, ASP.NET და ა.შ. [15].

ვიზუალური დაპროგრამების თეორიულ საფუძველს ობიექტ-ორიენტირებული ანალიზისა და ობიექტ-ორიენტირებული დაპროექტების მეთოდები წარმოადგენს (ნახ.1.3). ობიექტ-ორიენტირებული მიდგომის საფუძველზე შეიქმნა უნიფიცირებული მოდელირების ენა (UML-Unified Modeling Language), როგორც უახლესი სტანდარტი რთული პროგრამული აპლიკაციების ასაგებად [5,10].

ასეთი გრაფო-ანალიზური ენის განვითარებაში განსაკუთრებული წვლილი შეიტანეს IBM (Rational Rose) ფირმის მეცნიერებმა, დაპროექტებელ-დიზაინერებმა გრადი ბუჩმა, ივარ ჯაკობსონმა და ჯეიმს რამბომ [5,12].



ნახ.1.3. UML-ტექნოლოგიის 4-ეტაპიანი მოდელი

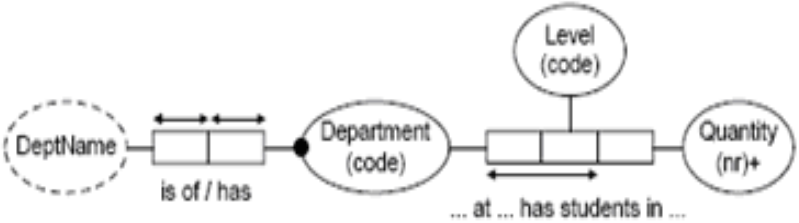
UML-ტექნოლოგია თეორიულ-პრაქტიკული ინფორმატიკის ბაზაზე ჩამოყალიბდა. იგი განაწილებული ავტომატიზებული სისტემების დაპროექტების მეთოდოლოგიური საფუძველია,

რომლის კონცეფციითაც შეიქმნა ისეთი ინსტრუმენტები, როგორიცაა Rational Rose, ParadigmPlus, MsVisio და სხვ. [11,15]. პროგრამული პაკეტების აგების პროცესის სტანდარტიზაცია სამი ძირითადი მიმართულების „გენეტიკური“ მემკვიდრეა: დაპროექტების ავტომატიზაცია, დაპროგრამების ავტომატიზაცია და მონაცემთა ბაზების აგების ავტომატიზაცა [29].

მართვის კომპიუტერული სისტემების პროგრამული უზრუნველყოფის აგების პროცესების ასეთი სრულფასოვანი ავტომატიზაცია ვიზუალური მოდელირების სახელწოდებით დამკვიდრდა. იგი მოდელების გრაფიკულ წარმოდგენას ეყრდნობა და ფლობს მოქნილ რევერსულ ტექნოლოგიას [11].

ასეთ პროგრამულ ინსტრუმენტებში საყურადღებო ადგილი უჭირავს **ობიექტ-როლურ მოდელირებას** (ORM), რომელთა საშუალებით, კატეგორიალური მიდგომის საფუძველზე ხორციელდება მონაცემთა და ცოდნის ბაზების [7,23], კოგნიტური სისტემების [24], აგრეთვე მონაცემთა საცავების დაპროექტების პროცესების ავტომატიზაცია [18]. 1.4 ნახაზზე ნაჩვენებია ORM-დიაგრამის საილუსტრაციო ფრაგმენტი შემდეგი ფაქტებით:

- f1-დეპარტამენტს აქვს სახელი;
- f2 დეპარტამენტის 1-ელ კურსზე სწავლობს 500 სტუდენტი.



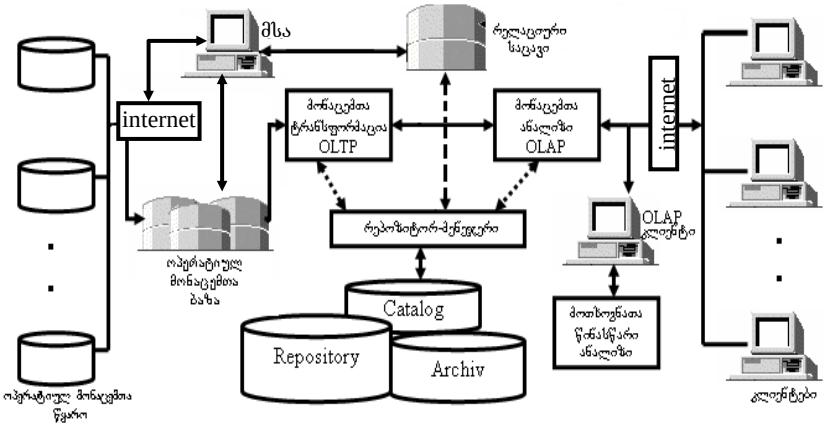
ნახ.1.4. ORM-დიაგრამის ფრაგმენტი

კორპორაციული მონაცემთა საცავების შექმნა თანამედროვე საინფორმაციო ტექნოლოგიების ერთ-ერთი უახლესი და აქტუალური მიმართულებაა [18].

მონაცემთა საცავი (Data warehouse) განიხილება როგორც რომელიმე კონკრეტული ორგანიზაციის ან დიდი საწარმოსთვის განკუთვნილი სპეციალური სუპერ-ბაზა, სადაც მიმდინარე

ოპერატიული სამუშაოს შესრულებისას თავს იყრის ქრონოლოგიურ ინფორმაციათა მთელი სპექტრი, რომელთა დანიშნულებაცაა მომხმარებლისთვის ინტერნეტ გვერდებზე მიზნობრივად განლაგებული ტექსტური, გრაფიკული და აუდიო-ვიზუალური საინფორმაციო ბლოკების მიწოდება.

1.5 ნახაზზე მოცემული გვაქვს განაწილებული ავტომატიზებული მართვის სისტემის მონაცემთა საცავის ზოგადი სქემა.



ნახ.1.5. მონაცემთა საცავის ზოგადი სქემა

მონაცემთა საცავის მუშაობის პრინციპი ასეთია: პირველ ეტაპზე Dwh-ის გამოყენებით რელაციურ ბაზებში ერთად თავმოყრილი მონაცემები ლაგდება გარკვეული სტრუქტურული თანამიმდევრობით, ხდება მათი „დაწმენდა“. მეორე ეტაპზე წარმოებს ინფორმაციის ტექნოლოგიური დამუშავება OLAP (Online Analytical Processing) მონაცემთა ოპერატიული ანალიზის გამოყენებით. მესამე ეტაპზე ეს მონაცემები მომხმარებლებს მიეწოდებათ ინტერნეტის საშუალებით. ინფორმაციული ბლოკები, რომლებიც მონაცემთა საცავებშია განაწილებული,

მიზანმიმართულად თავსდება ინტერნეტ გვერდებზე და ხელმისაწვდომია ფართო მომხმარებლისთვის.

OLAP უნიკალური ინსტრუმენტია, რომელიც საშუალებას იძლევა ინფორმაციის მრავალგანზომილებიანი ანალიზის ჩასატარებლად. იგი რელაციური ტიპის მონაცემთა საცავებისა და მონაცემთა ბაზებისთვისაა ეფექტურად გამოყენებადი. საყურადღებოა, რომ რელაციური ბაზების ფუძემდებელმა (1970 წ.), ე.ფ. კოდმა ჩამოაყალიბა (1993 წ.) ოპერატიული ანალიზის OLAP-ინსტრუმენტის საფუძველზე მონაცემთა საცავების დაპროექტებისა და ფუნქციონირების პრინციპები [31]. ეს 12 წესია, რომელსაც უნდა აკმაყოფილებდეს ნებისმიერი განაწილებული ავტომატიზებული სისტემა მონაცემთა საცავით, რათა ჩატარდეს საინფორმაციო ბლოკების სრულფასოვანი ოპერატიული ანალიზი: 1. მონაცემთა მრავალგანზომილებიანი კონცეპტუალური წარმოდგენა; 2. გამჭვირვალეობა; 3. მიღწევადობა; 4. ანგარიშთა დამუშავებისას მუდმივი წარმადობა; 5. კლიენტ-სერვერის არქიტექტურა; 6. გენერირებადი მრავალგანზომილებიანობა; 7. დინამიკური მართვის რეჟიმი; 8. მრავალმომხმარებლიანობა; 9. შეუზღუდავი განზომილების დამუშავების ოპერაციები; 10. მონაცემთა ინტუიციური მანიპულაცია; 11. ანგარიშების მიღების მოქნილი საშუალება; 12. შეუზღუდავი ზომები და აგრეგაციათა რაოდენობა [18,25].

კორპორაციული ქსელები (**MAN – Metropolitan Area Network**) განაწილებული მონაცემთა საცავების აუცილებელი ელემენტია. მათი განვითარება ბოლო ათწლეულში შეუქცევად პროცესს წარმოადგენს და პრინციპულად ახალ ამოცანებს უყენებს ინფორმატიკოსებს. თუ ლოკალურ კომპიუტერულ ქსელებში (LAN – Local Area Network) ქსელის ადმინისტრირების პრობლემა არცთუ მწვავე იყო, კორპორაციულ ქსელებში ამ საქმეს საკმაოდ დიდი ოდენობის კვალიფიცირებული პერსონალი ემსახურება და ორგანიზაციის გამართული მუშაობა მათზე დიდადაა დამოკიდებული.

კორპორაციული ქსელები ხასიათდება შემდეგი ძირითადი თვისებებით: გავრცელების გეოგრაფია, მომხმარებელთა და

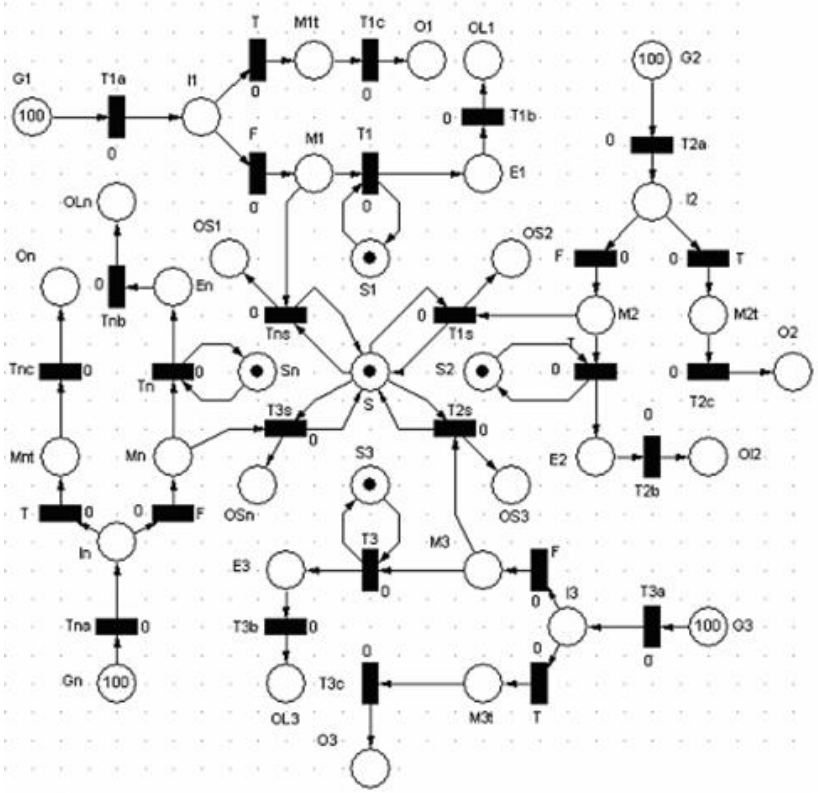
სერვისების დიდი რაოდენობა, აპარატურის მრავალფეროვნება, ინფორმაციის დიდი მოცულობა, უსაფრთხოება და შენახვის საიმედოობა. ასეთი სისტემების ფუნქციონირების ეფექტური მოდელირებისა და ანალიზისათვის გამოიყენება პეტრის ქსელები [19].

კორპორაციულ ქსელებში, როგორც წესი, მონაცემები პროგრამებისგან თითქმის სრულებით იზოლირებულად ინახება. კერძოდ, ხისტი დისკების მასივებში ცენტრალიზებული სახით. საგანგებო ქსელური ტექნოლოგიები უზრუნველყოფს მონაცემთა საცავების მართვას ფაილ-სერვერების მიერ.

კლასტერული არქიტექტურა არის ფაილ-სერვერის რეალიზაციის ყველაზე ოპტიმალური ტექნოლოგია. კლასტერი ორი ან მეტი კვანძისგან (კომპიუტერისგან) შემდგარ შიგა ქსელს წარმოადგენს, რომელიც აპარატულ და პროგრამულ დონეზე ერთი სერვერის სახითაა გაფორმებული. კლასტერი შეიცავს წინასწარ განსაზღვრული რესურსების სიმრავლეს (**IP**-მისამართები, ქსელური სახელები, სისტემური სერვისები, განაწილებული საქაღალდეები, გამოყენებითი პროგრამები და სხვა). ყოველი მათგანი დროის მოცემულ მომენტში კონკრეტული კვანძის კუთვნილებაა, ხოლო კვანძის მწყობრიდან გამოსვლის შემთხვევაში სპეციალური კლასტერის სისტემური სერვისი მას ავტომატურად გადაიტანს სხვა მოქმედ კვანძზე.

1.6 ნახაზზე მოცემულია კორპირაციული ქსელის მართვის მსგავსი სისტემების ზოგადი პეტრის ქსელის მოდელი [26]. იგი აგებულია გრაფო-ანალიზური რედაქტორით და რეალიზებულია მიზეზ-შედეგობრივი პრედიკატული პეტრის ქსელის სახით. უზრუნველყოფს მიმდევრობითი და პარალელური პროცესების მართვას და მათი შესრულების დროითი პარამეტრების ფიქსირებას. სისტემის მთავარი ოფისის სერვერი მოდელირდება პეტრის ქსელის პოზიციით (S). მოთხოვნების ფორმირებისათვის (დავუშვათ, იგი აკმაყოფილებს პუასონის განაწილებას) შემოტანილია რანდომ-გენერატორი (G). მოთხოვნის ანალიზისა (I) და მისი დამუშავების შედეგები გამოიცემა პოზიციებში (O_i-შედეგები მიიღება საერთო ლოკალური რესურსების გარეშე, OL_i-

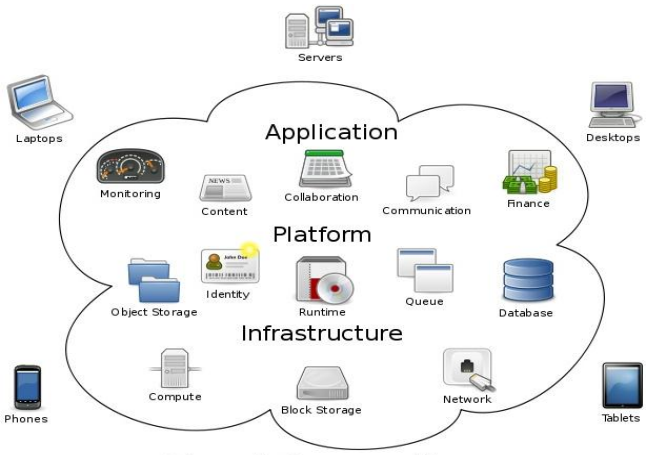
შედეგები მიიღება საერთო ლოკალური რესურსებით, OS_i-შედეგები მიიღება საერთო გლობალური რესურსებით).



ნახ.1.6. კორპორაციული ქსელის ფრაგმენტის მოდელი პეტრის ქსელით

„ღრუბლოვან გაანგარიშებათა” (Cloud Computing) ტექნოლოგია შედარებით ახალი, მაგრამ სწრაფად განვითარებადი მიმართულებაა IT-სერვისების სფეროში [33-35]. 2008 წელს გამოქვეყნდა IEEE-ს დოკუმენტი, რომელშიც მონაცემთა ღრუბლოვანი დამუშავება განსაზღვრული იყო, როგორც ახალი

„პარადიგმა, რომლის დროსაც ინფორმაცია მუდმივად ინახება სერვერებზე და დროებით კეშირდება კლიენტის მხარეს” (ნახ.1.7).



ნახ.1.7. ღრუბლოვანი გამოთვლები

შეიძლება ითქვას, რომ ღრუბლოვანი გაანგარიშებანი, როგორც რესურსების განაწილებისა და ვირტუალიზაციის ერთ-ერთი ტექნოლოგია, რომელიც რესურსებს და სიმძლავრეებს აწვდის მომხმარებელს ინტერნეტ-სერვისების სახით, უსაფრთხოების თვალსაზრისით ხასიათდება რიგი უპირატესობებით ტრადიციულ კომპიუტერთა ქსელური სტრუქტურების ტექნოლოგიებთან შედარებით. ამავდროულად, ღრუბლოვანი დამუშავების მეშვეობით კერძო თუ კორპორატიული ინფორმაციის შენახვა და გამოყენება ბევრად ნაკლები დანახარჯებით ხერხდება.

ღრუბლოვანი ტექნოლოგია მომხმარებელს სხვადასხვა დონის სერვისებს სთავაზობს:

ინფრასტრუქტურული სერვისი (IaaS - Infrastructure as a Service) - აპარატული საშუალებების (ყველაზე მარტივი მაგალითი: საკუთარი „ხისტი დისკი ღრუბელში"), ოპერაციული სისტემების და სისტემური პროგრამული უზრუნველყოფის „ღრუბლოვანი ალტერნატივა";

პლატფორმული სერვისი (PaaS - Platform as a Service) - ვებ-ბაზირებული და სხვა, მათ შორის, მონაცემთა ბაზების მართვის სისტემების აგებისა და მართვის საშუალებათა ,,ღრუბლოვანი ალტერნატივა";

პროგრამული სერვისი (SaaS - Software as a Service) - პროგრამული უზრუნველყოფის გამოყენების ,,ღრუბლოვანი ალტერნატივა".

ამასთან უნდა აღინიშნოს, რომ ინფორმაციის ,,ღრუბელში" განთავსებაზე საუბრისას, უწინარეს ყოვლისა, იგულისხმება ე.წ. ,,გარე ღრუბელი", რომელიც ინფორმაციის შენახვის ყველაზე იაფ საშუალებას წარმოადგენს და საუკეთესო არჩევანია ახლადდაფუძნებული ორგანიზაციებისთვის, რომელთაც კომპიუტერულ აპარატურის და პროგრამული უზრუნველყოფის შესაძენად და დასანერგად სერიოზული კაპიტალდაბანდება ჯერ კიდევ არ განუხორციელებიათ.

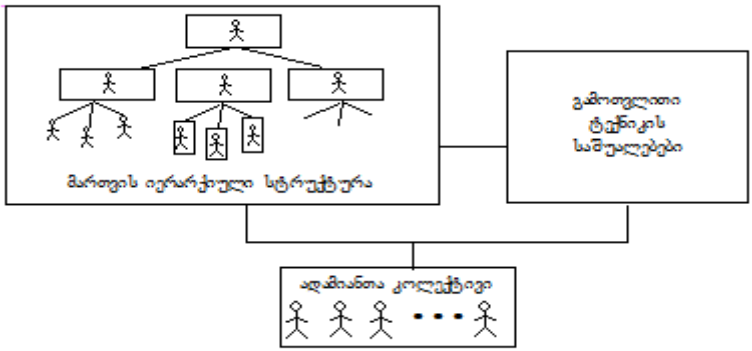
დასასრულ გვინდა აღვნიშნოთ, რომ ინფორმატიკა არის კომპლექსური მეცნიერებაა - თეორიული, პრაქტიკული და ტექნიკური ასპექტების თვალსაზრისით, რომელიც აერთიანებს ინფორმაციის მოპოვების, შენახვის, გადამუშავების, გადაცემისა და დაცვის ტექნოლოგიებს, სემანტიკური ბიზნეს-პროცესების მათემატიკური მოდელირების მეთოდებს, ფორმალური გრამატიკისა და უნიფიცირებული, ობიექტ-ორიენტირებული, ვიზუალური დაპროგრამების ინსტრუმენტულ საშუალებებს.

პრაგმატული ასპექტებით იგი უდავოდ წარმოადგენს სისტემების მართვის ზოგადი თეორიის ფუნდამენტს.

წინამდებარე სახელმძღვანელოში მოცემულია ინფორმატიკის დარგის ზემოაღნიშნული მიმართულებების დეტალური მიმოხილვა, თუ რას წარმოადგენს მართვის ავტომატიზებული სისტემები და რა ადგილი უკავია მას ინფორმატიკის, ანუ კომპიუტერულ მეცნიერებათა ანსამბლში.

1.3. მართვის ავტომატიზებული სისტემები

მართვის ავტომატიზებული სისტემების ცნების ერთ-ერთი კლასიკური განმარტება, რომელიც 70-იან წლებში დამკვიდრდა, ასე ჟღერს: „ადამიანთა კოლექტივების, ადმინისტრაციული და ეკონომიკური-მათემატიკური მეთოდების, ინფორმაციულ ბაზის, გამოთვლითი ტექნიკისა და კომუნიკაციურ საშუალებათა ერთობლიობა, რომლითაც ხორციელდება ოპტიმალური მართვა ადამიანთა საქმიანობის სხვადასხვა სფეროში“ (ნახ.1.8) [12].



ნახ.1.8. მართვის ავტომატიზებული სისტემა

„მართვის ავტომატიზებული სისტემების“ პირდაპირი ინგლისურენოვანი თარგმანი შეესაბამება Automated Control Systems. სიტყვა „მართვა“ (Control) შედარებით ფართე ცნებაა და იგი მოიცავს მართვას ტექნიკურ სისტემებში (კონტროლი) და მართვას ორგანიზაციულ სისტემებში (ანუ მენეჯმენტი). ამგვარად, მართვის ავტომატიზებული სისტემები შინაარსობრივად ახლოა მენეჯმენტის საინფორმაციო სისტემების (Management Information Systems) ცნებასთან.

ტექნიკურ სისტემებში მართვის პროცესი ხორციელდება ადამიანის ჩარევის გარეშე (რობოტები, რეგულატორები და სხვ.), ამიტომაც მათ ავტომატური მართვის სისტემებს უწოდებენ.

ორგანიზაციული ობიექტების ავტომატიზებული სისტემების მართვის პროცესში ინფორმაციის დამუშავება ხორციელდება კომპიუტერის საშუალებით, ხოლო შედეგების გამოყენება და

გადაწყვეტილების მიღება ადამიანის მიერ (ნახ.1.8). ამიტომაც მართვის ავტომატიზებულ სისტემებს მიაკუთვნებენ ,,ადამიან-მანქანურ'' სისტემათა კლასს [32].

ამგვარად, მართვის ავტომატიზებული სისტემების, ანუ მენეჯმენტის საინფორმაციო სისტემების ძირითადი ამოცანა არის ორგანიზაციული ობიექტების საწარმოო პროცესების (ანუ საქმიანი პროცესების, ბიზნეს-პროცესების) მენეჯმენტი. ასეთ სისტემებში კომპიუტერული ტექნიკის და ინფორმაციული ტექნოლოგიების დანერგვით ხორციელდება ავტომატიზაციის პროცესი, რაც მნიშვნელოვნად ამაღლებს მათ შრომის ნაყოფიერების დონეს მართვის პროცესების სრულყოფის ბაზაზე.

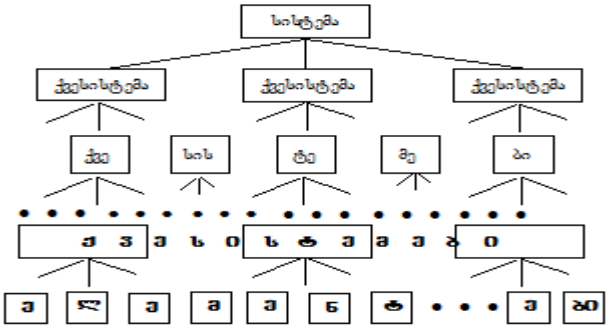
ახლა განვიხილოთ მოკლედ მართვის ავტომატიზებული სისტემის და მისი პროცესების მოდელირების და დაპროექტების ძირითად ტერმინთა არსები.

სისტემა: ელემენტებისა და მათი ურთიერკავშირების ერთობლიობაა, რომელსაც აქვს საერთო მიზნი და ფუნქციები მის მისაღწევად. ყოველ სისტემას აქვს თავისი მოქმედების საზღვრები, ბიზნეს-პროცესები და ბიზნეს-წესები (ბიზნეს-ლოგიკა).

სისტემის მაგალითებია: ადამიანი, სახელმწიფო, დარგობრივი სამინისტრო, კორპორაცია, აეროპორტი, საბაჟო, ქარხანა, ფერმა, საავადმყოფო, აფთიაქი, ავტომობილი, ჭადრაკი, ფეხბურთი და მრავალი სხვა.

თუ სისტემის მდგომარეობა არ იცვლება დროში, მაშინ მას სტატიკურს უწოდებენ, ხოლო თუ იცვლება, მაშინ - დინამიკურს.

როგორც აღვნიშნეთ, სისტემა შედგება ნაწილებისაგან. თუ სისტემის ეს ნაწილები, თავის მხრივ, შედგება ნაწილებისაგან, მაშინ მათ ქვესისტემებს უწოდებენ. თუ ნაწილები განუყოფელია, მაშინ მათ სისტემის ელემენტებს უწოდებენ. სისტემას შეიძლება გააჩნდეს რამდენიმე სხვადასხვა ქვესისტემის დონეები. 1.9. ნახაზზე წარმოდგენილია სისტემის იერარქიული (თანამიმდევრობა, უდაბლესიდან უმაღლესისაკენ დაქვემდებარების წესით) სქემა.



ნახ.1.9. სისტემის იერარქიული სტრუქტურა

ნებისმიერი სისტემა მოქმედებს გარემოში და პირიქით, გარემოს შეუძლია იმოქმედოს სისტემაზე, შეცვალოს მისი მდგომარეობა.

თუ გარემოს ეს ზემოქმედება სისტემაზე ისე ძლიერია, რომ იგი არ შეგვიძლია უგულებელვყოთ, მაშინ სისტემას ეწოდება ღია, ხოლო თუ ეს ზემოქმედება ისეთი უმნიშვნელოა, რომ იგი შეიძლება უგულებელვყოთ, მაშინ ეწოდება ჩაკეტილი. მაგალითად, ადამიანის ორგანიზმზე მოქმედებს გარემოს ფაქტორები: წნევა, ტემპერატურა, ჰაერის შედგენილობა, ტენიანობა. თანაც ეს ზემოქმედება საკმაოდ ძლიერია. ამ მხრივ, ადამიანი შეიძლება განვიხილოთ როგორც ღია სისტემა. ღია სისტემის მაგალითია საწარმო, რადგან მის ფუნქციონირებაზე გავლენას ახდენს ბაზარი, საწარმო-მიმწოდებლები, ზემდგომი ორგანოები და სხვ.

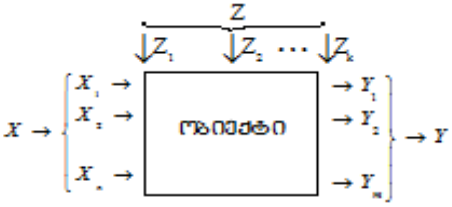
მოდელი. მოდელი არის მიახლოებითი ასახვა ანუ წარმოდგენა იმ მოვლენების და პროცესების, რომლებიც მიმდინარეობს სხვადახვა ობიექტებში, მათ შორის მართვის ობიექტებშიც.

მოდელები არსებობს სხვადასხვა სახის: წარმოსახვითი, მოდელ-ანალოგები, მათემატიკური და სხვ. მაგალითად, გლობუსი წარმოადგენს დედამიწის სივრცით წარმოსახვით მოდელს – სფეროს, რუკა – სიბრტყით წარმოსახვით მოდელს. არსებობს სახლების, საწარმოების, თვითმფრინავების მოდელ-ანალოგები და სხვ.

მოდელები საშუალებას გვაძლევს გამოვიკვლიოთ ობიექტების თვისებები ისე, რომ არ ჩავატაროთ ექსპერიმენტები თვით ობიექტებზე, რადგან ექსპერიმენტების ჩატარება ობიექტებზე ზოგჯერ საკმაოდ ძვირია, ზოგჯერ კი საერთოდ შეუძლებელიცაა.

ობიექტების მოდელების აგების პროცესს მოდელირება ეწოდება. ეკონომიკური მართვის სისტემებში ფართო გავრცელება ჰპოვა სიმბოლურმა, ანუ მათემატიკურმა მოდელირებამ, როდესაც სიმბოლოებისა და განსაზღვრული დამოკიდებულებების საშუალებით ობიექტის შემავალ და გამომავალ პარამეტრებს შორის წესდება კავშირები (f) მათემატიკური ფორმულების სახით. ეს პარამეტრები ობიექტში მიმდინარე პროცესთა მახასიათებლებია.

განვიხილოთ ზოგადად ობიექტი (ნახ.1.10). შემავალი პარამეტრები აღვნიშნოთ X (x_1,x_2, . . .,x_n)-ით, ხოლო გამომავალი Y (y_1,y_2, . . .,y_m)-ით.



ნახ.1.10. მართვის ობიექტი

გარდა ამისა, ობიექტებზე მოქმედებს გარემოს პარამეტრები Z (z_1,z_2,...,z_n), რომლებიც ასევე წარმოადგენს შემავალ პარამეტრებს და გავლენას ახდენს გამომავალ პარამეტრებზე. მაშინ მოცემული ობიექტის მათემატიკური მოდელი, ზოგადი სახით შეგვიძლია ჩავწეროთ შემდეგი ფორმულით:

$$Y = f(X,Z)$$

ან გაშლილი სახით:

$$Y_1 = f_1(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots z_k)$$
$$Y_2 = f_2(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots z_k)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$Y_m = f_m(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots z_k)$$

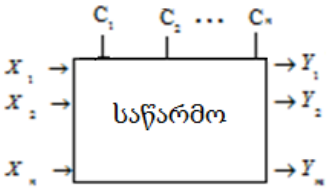
მართვის ობიექტის შემავალი პარამეტრების (X), გარემოს პარამეტრების (Z), გამომავალი პარამეტრების (Y) და მათ შორის კავშირების (f_i) დადგენა წარმოადგენს მათემატიკური მოდელირების კვლევის საგანს. ამჟამად სხვადასხვა მიმართულების მათემატიკური მოდელირების თეორია ფართოდ გამოიყენება როგორც ტექნოლოგიური პროცესების, ისე ეკონომიკური და პოლიტიკური სახის მართვის ობიექტების მათემატიკური მოდელების ასაგებად [12].

განვიხილოთ მაგალითი. დავუშვათ, რომ საწარმო უშვებს n სხვადასხვა სახის პროდუქციას. თითოეული სახის პროდუქცია საწარმოს აძლევს განსაზღვრულ მოგებას. საჭიროა ისე დაიგეგმოს საწარმოს მუშაობა, რომ ჯამში მან მიიღოს მაქსიმალური მოგება.

ავაგოთ ამ პროცესის მათემატიკური მოდელი.

პირველ რიგში შემოვიტანოთ ცვლადები, რომლებითაც აღვნიშნავთ დაგეგმვის პროცესის პარამეტრებს. თითოეული i-ური სახის გამოსაშვები პროდუქციის რაოდენობა აღვნიშნოთ X_i-ით (i=1,n). მოგება, რომელსაც საწარმოს აძლევს i-ური სახის ერთეული პროდუქცია, აღვნიშნოთ C_i-ით; საწარმოს მიერ მიღებული მოგება, პროდუქციის რეალიზაციის შედეგად, აღვნიშნოთ Y-ით (ნახ.1.11).

ამ შემთხვევაში X (x_1,x_2, . . .,x_n) და C (c_1,c_2, . . .,c_n) შემავალი პარამეტრებია, გამომავალი კი - Y (y_1,y_2, . . .,y_n). ამასთან, C ცნობილი სიდიდეებია (კონსტანტები), ხოლო X უცნობი (ცვლადი).



ნახ.1.11

თუ i-ური სახის ერთეული პროდუქციის რეალიზაციის შედეგად საწარმო ღებულობს მოგებას. მაშინ i-ური სახის X_i-ური რაოდენობის რეალიზაციის შედეგად მიიღებს C_i*X_i მოგებას. ხოლო საწარმოს მიერ მიღებული მოგება ყველა სახის პროდუქციის რეალიზაციის შედეგად იქნება:

$$C_1 \cdot X_1 + C_2 \cdot X_2 + C_3 \cdot X_3 + \ldots + C_n \cdot X_n = \sum_{i=1}^{n} C_i \cdot X_i .$$

ამრიგად, აღწერილი პროცესის მათემატიკურ მოდელს ექნება შემდეგი სახე:

$$Y = \sum_{i=1}^{n} C_i \cdot X_i .$$

ცხადია, ეს წარმოადგენს აღნიშნული პროცესის გამარტივებულ სახეს, რადგან არ გაგვითვალისწინებია მთელი რიგი ფაქტორები, რომლებიც გავლენას ახდენს ამ (კერძოდ, დაგეგმვის) პროცესზე.

კონკრეტული საწარმოსათვის n, C_i , X_i , და Y სიდიდეები მიიღებს ზუსტად განსაზღვრულ მნიშვნელობებს.

მოდელირების თეორია საშუალებას გვაძლევს ავაგოთ არა მარტო ობიექტებისა და პროცესების მათემატიკური მოდელები, არამედ გადავწყვიტოთ კიდეც ასეთი ტიპის ამოცანები. ანუ ამ შემთხვევაში განვსაზღვროთ X_i-ის მნიშვნელობები, რომელთა დროსაც Y იქნება მაქსიმალური.

1.4. მათემატიკური მოდელირება მართვის ავტომატიზებულ სისტემებში

მართვის ავტომატიზებული სისტემის (მას) შემუშავების დროს ფართოდ იყენებენ საინჟინრო პრაქტიკისათვის დამახასიათებელ მათემატიკურ მეთოდებს. დიდ საწარმოო სისტემებში მმართველი გადაწყვეტილებების პრაქტიკული განხორციელება დაკავშირებულია მატერიალური და შრომითი რესურსების საგრძნობ დანახარჯებთან. აქედან გამომდინარე, გადაწყვეტილების არასწორი ვარიანტის ამორჩევამ შეიძლება განაპირობოს დიდი მოცულობის დანაკარგები სამეურნეო მოღვაწეობაში.

რასაკვირველია, გადაწყვეტილების ამორჩევა შესაძლებელია იმ შემთხვევაში, როდესაც არსებობს ვარიანტების სიმრავლე. ამავე დროს ვარიანტის ხარისხის შეფასებისათვის საჭიროა მმართველი გადაწყვეტილების შედეგის პროგნოზირება. სწორედ ამის საშუალებას იძლევა მათემატიკური მოდელირება.

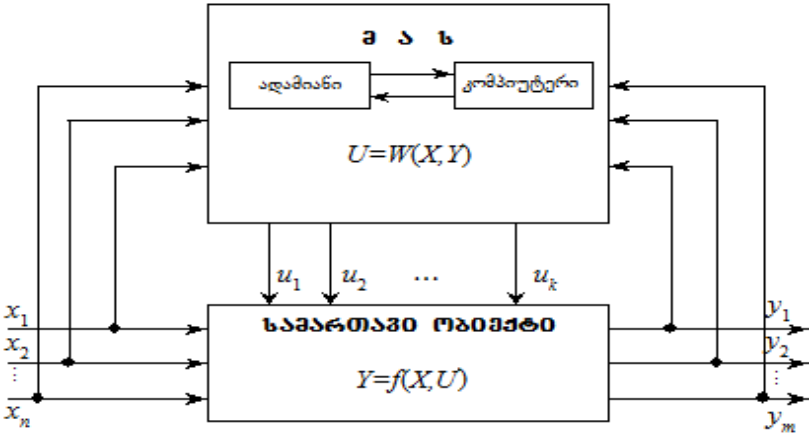
როგორც ცნობილია, მოდელი არის განსაზღვრულ ობიექტში მიმდინარე პროცესების ან მოვლენების მიახლოებითი ასახვა (წარმოდგენა). მოდელის ძირითადი დანიშნულებაა მისი გამოყენება მართვასა და პროგნოზირებაში. გარდა ამისა, მოდელი საშუალებას იძლევა გამოვიკვლიოთ სამართავი ობიექტის ცალკეული თვისებების ურთიერთგავლენა ისე, რომ არ ჩავატაროთ ობიექტზე რაიმე ექსპერიმენტი. მოდელის ეს თვისება განსაკუთრებით მნიშვნელოვანია იმ შემთხვევაში, როცა ობიექტზე ექსპერიმენტის ჩატარება მეტად ძვირია ან სულაც შეუძლებელია. **მას**-ებში დიდი გამოყენება პოვა სიმბოლურმა ან მათემატიკურმა მოდელებმა, სადაც სიმბოლოებით და დამოკიდებულებით მათემატიკური თანაფარდობების სახით მყარდება კავშირი ობიექტის შესასვლელებსა და გამოსასვლელებს შორის.

როგორც წესი, მმართველი გადაწყვეტილება ხასიათდება პარამეტრების სიმრავლით, რომლებიც წარმოადგენს ამონახსნის ელემენტებს. მაგალითად, გამოსაშვები პროდუქციის რაოდენობა და ასორტიმენტი, გადაზიდვების მოცულობები, საწარმოში დასაყენებელი აგრეგატების სიმძლავრე, სათავსოთა მოცულობები, სამართავი ობიექტის კოორდინატები და ა.შ. ობიექტის მართვა მდგომარეობს მისი პარამეტრების ისეთი მნიშვნელობების უზრუნველყოფაში, რომლებიც ობიექტში პროცესების აუცილებელ მიმდინარეობას განსაზღვრავენ. მმართველი გადაწყვეტილების შედეგის შეფასების მაჩვენებლები F_1, F_2,...,F_n, შეიძლება იყოს მატერიალური შრომითი და ენერგორესურსების დანახარჯები, ფულადი სახსრები, საწარმოოს მოგება, სისტემის საიმედო მუშაობა, სამუშაოების შესრულების დრო და ა.შ.

მას-ის ძირითად დანიშნულებას წარმოადგენს მმართველი გადაწყვეტილებების (ზემოქმედებების) გამომუშავება, რომლებიც სამართავ ობიექტებში უზურნველყოფს პროცესების ნორმალურ მსვლელობას. ამავე დროს **მას**-ის სამართავ ობიექტს წარმოადგენს ორგანიზაციული სისტემები ანუ სისტემები, რომლებშიც მონაწილეობს ადამიანი ან ადამიანთა კოლექტივი. ორგანიზაციული სისტემებისათვის კი დამახასიათებელია ორგანიზაციული მართვა, რომლის არსია: წარმოების ორგანიზაცია, მატერიალური და შრომითი რესურსების

განაწილება, წარმოების მომარაგება, გამოსაშვები პროდუქციის რაოდენობისა და ასორტიმენტის განსაზღვრა, გადასაზიდი პროდუქციის რაოდენობის და გადაზიდვის მიმართულების განსაზღვრა, საწარმოო პროცესების პარამეტრების ოპერატიული აღრიცხვა და ა.შ. [12].

მართვის ავტომატიზებულ სისტემებს ისევე, როგორც სხვა მართვის სისტემებს, გააჩნია კლასიკური ბლოკ-სქემა, რომელიც ნაჩვენებია 1.12 ნახაზზე. სამართავ ობიექტს აქვს შესასვლელების $X\ (x_1,x_2, \ldots,x_n)$ და გამოსასვლელების სიმრავლე $Y\ (y_1,y_2, \ldots,y_m)$. ობიექტის შესასვლელები და გამოსასვლელები პარამეტრებია, რომლებიც ახასიათებს ობიექტში მიმდინარე პროცესებს.



ნახ.1.12

სამართავი ობიექტის შესასვლელებსა და გამოსასვლელებს შორის მყარდება გარკვეული კავშირი:

$$Y=f(X), \qquad (1.1)$$

სადაც f არის ასახვა (ფუნქცია), რომელიც ახორციელებს სამართავი ობიექტის ფუნქციონირების აღწერას, ხოლო X და Y შესასვლელი და გამოსასვლელი ვექტორებია.

რაც შეეხება გამოსასვლელებს, გადაწყვეტილების მიმღები პირი (ადამიანი) დაინტერესებულია მათი გარკვეული მნიშვნელობებით, რადგანაც ისინი შეადგენს ობიექტის მართვის

მიზანს. Y ვექტორის აუცილებელი მნიშვნელობების მიღებას უზრუნველყოფს მართვის ავტომატიზებული სისტემა.

მას-ის შესასვლელზე მიეწოდება ინფორმაცია ობიექტის მიმდინარე მდგომარეობის შესახებ. თუ ობიექტის ფაქტობრივ და მიზნობრივ მდგომარეობას შორის არსებობს განთანხმება, მაშინ **მას**-ში გამომუშავდება მმართველი ზემოქმედება U_n, რომელიც გაიგზავნება სამართავ ობიექტზე ამ განთანხმების აღმოსაფხვრელად, ხოლო იმ შემთხვევაში, როდესაც განთანხმება არ არსებობს, მმართველი ზემოქმედება $U_n=0$.

აქედან გამომდინარეობს, რომ Y ვექტორის მნიშვნელობაზე გავლენას ახდენს არა მარტო *X* ვექტორის მნიშვნელობები, არამედ მმართველი ვექტორის მაჩვენებლები, რომლებიც განისაზღვრება შემდეგნაირად:

$$U=W(X,Y). \qquad (1.2)$$

ამ გამოსახულებაში *W* არის ასახვა (ფუნქცია), რომელიც წარმოადგენს მართვის შეფასების კრიტერიუმს. (1.2) გამოსახულების გათვალისწინებით, (1.1) გამოსახულება მიიღებს შემდეგ სახეს:

$$Y=F(X,U). \qquad (1.3)$$

ეს გამოსახულება წარმოადგენს მათემატიკურ მოდელს, რომელიც აღწერს სამართავი ობიექტის სტრუქტურას და მისი ფუნქციონირების მართვის კანონებს.

რადგანაც სამართავი ობიექტის ფუნქციონირება მიმდინარეობს არა იზოლირებულად, არამედ არსებულ გარემოცვაში (გარემოში), ეს უკანასკნელი გარკვეულად ზღუდავს ობიექტის დამახასიათებელი პარამეტრების დასაშვებ მნიშვნელობებს. ეს გარემოება აისახება მოდელში განსაზღვრული შეზღუდვების შემოტანის სახით:

$$g_i(x, y) \gtreqless b_i$$

სადაც g_i არის *i*-ური სახის რესურსის ხარჯვის ფუნქცია, ხოლო – b_i პარამეტრის ზღვრულად დასაშვები მნიშვნელობა.

ეს შეზღუდვები განსაზღვრავს მმართველი გადაწყვეტილების დასაშვებ არეს. ამავე დროს შესაძლებელია, რომ ამონახსნთა დასაშვები ვარიანტები არ იყოს ტოლფასოვანი.

ეს გარემოება წარმოშობს საუკეთესო ვარიანტის ამორჩევის ამოცანას. ამისათვის კი საჭიროა ამონახსნთა ვარიანტების შეფასება გარკვეული კრიტერიუმის საშუალებით, რომელიც ობიექტის ფუნქციონირების ყველაზე მნიშვნელოვანი მაჩვენებელია.

ამ მაჩვენებელს უწოდებენ მიზნობრივ ფუნქციას, ხოლო დასაშვები ამონახსენის ამორჩევა, რომელიც უზრუნველყოფს მიზნობრივი ფუნქციის ოპტიმალურ მნიშვნელობას, წარმოადგენს ოპტიმიზაციის ამოცანას.

იმ შემთხვევაში, როდესაც მოცემულია (ანუ წინასწარ ცნობილია) სამართავი ობიექტის შესასავლელი პარამეტრები და ჩვენზე დამოკიდებულია ამოხსნის ვარიანტის შერჩევა, მოდელი იქნება **დეტერმინირებული**.

იმ შემთხვევაში კი, როდესაც ობიექტის განსაზღვრული პარამეტრები წარმოადგენს შემთხვევით სიდიდეებს, მოდელი იქნება ალბათური (**სტოქასტური**).

ზოგიერთ შემთხვევაში ობიექტის ფუნქციონირება შესაძლებელია განხორციელდეს უცნობ პირობებში ანუ ობიექტზე უცნობი ფაქტორების მოქმედების დროს. ასეთი სახის პროცესებისათვის აგებული მოდელი ,,თამაშის'' ტიპის იქნება.

აქვე უნდა აღვნიშნოთ, რომ გადაწყვეტილებათა მიღების პროცესების ადეკვატური მოდელების ასაგებად არ არის საკმარისი მხოლოდ რაოდენობრივი მეთოდების გამოყენება. ბევრად ეფექტურია ისეთი მართვის სისტემების აგება, რომლებსაც საფუძვლად უდევს გადაწყვეტილებათა მიღების თვისებრივი მოდელები ანუ მოდელები, რომლებიც იყენებს სემანტიკურ, აზრობრივ ინფორმაციას.

ამ სახის მოდელებს განეკუთვნება ლოგიკური, გრაფული, ხელოვნური ინტელექტის მოდელები. მათ რიცხვშია ისეთი ქსელური გრაფული მოდელები, როგორიცაა ,,პეტრის ქსელები''. ისინი, როგორც მათემატიკური მოდელები, განსაკუთრებით ეფექტურად გამოიყენება განაწილებული მართვის სისტემების იმიტაციური მოდელირების ამოცანებისთვის.

1.5. პროგრამული ინჟინერია UML ტექნოლოგიის ბაზაზე

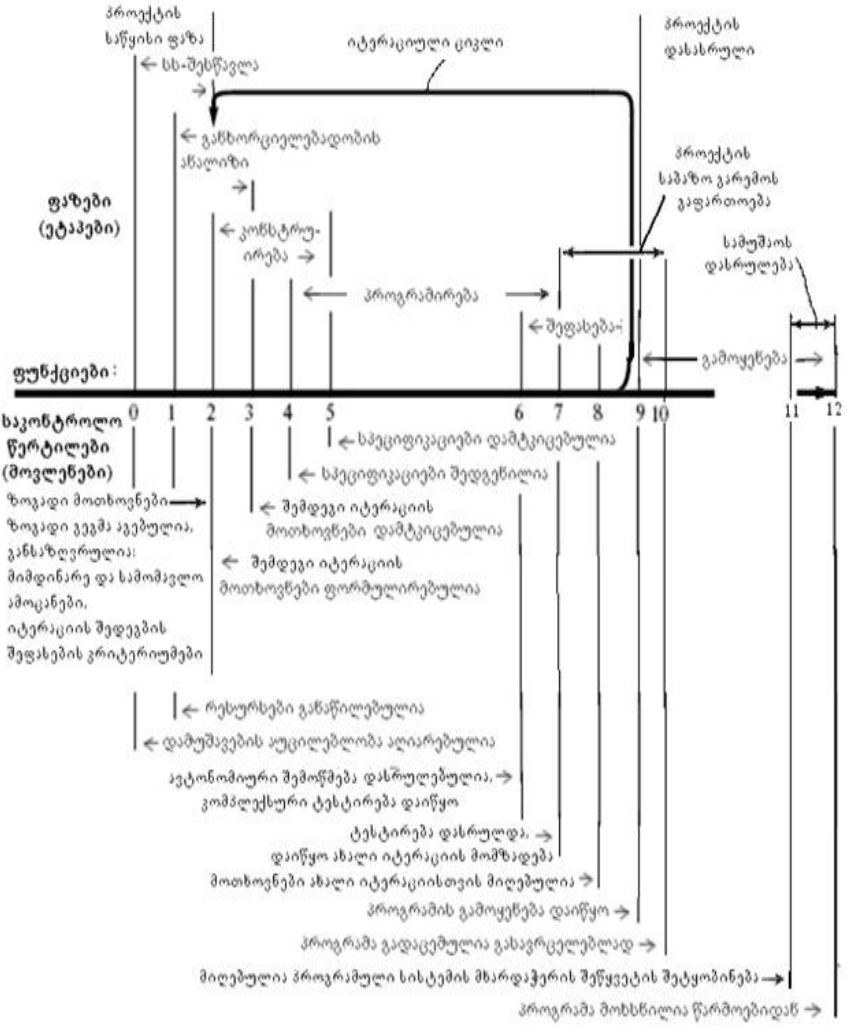
მართვის ავტომატიზებული სისტემების სრულყოფილი, საიმედო და მოქნილი პროგრამული უზრუნველყოფის (Software Engineering) სწრაფად დაპროექტება, რეალიზაცია, დანერგვა და შემდგომი თანხლება სისტემის დამკვეთ ორგანიზაციაში მეტად მნიშვნელოვანი ამოცანაა. მისი ეფექტურად გადაწყვეტა ბევრადაა დამოკიდებული როგორც საპროექტო-დეველოპმენტის გუნდის შემადგენლობა-გამოცდილებაზე, ასევე IT-ინფრასტრუქტურასა და CASE-ინსტრუმენტებზე.

ხშირად შეუძლებელია სრულყოფილი და საიმედო სისტემების აგება „სწრაფად“ (მოქნილად – Agile Programing) ისეთი მეთოდებით, როგორიცაა მაგალითად, ექსტრემალური დაპროგრამება [38]. ობიექტ-ორიენტირებული დაპროგრამების მეთოდი, რომელიც უნიფიცირებული მოდელირების ენის (UML) საშუალებით დამკვიდრდა, უნივერსალურია და მისი გამოყენებით პროგრამის სასიცოცხლო ციკლი მოითხოვს მისი აუცილებელი ეტაპების იტერაციულ განვითარებას [5,39].

1.13 ნახაზზე ნაჩვენებია პროგრამული უზრუნველყოფის დამუშავების სასიცოცხლო ციკლის განტერისეული მოდელის ეტაპები იტერაციული ბიჯებით [40].

პროგრამული სისტემის მენეჯმენტის საკონტროლო (0-12) წერტილებში, ეტაპების მიხედვით ხორციელდება იტერაციული სამუშაოები (დაბრუნება უკანა წერტილებში განმეორებითი პროცედურების ჩასატარებლად), სისტემის ფუნქციონალობის სისრულის დაზუსტების ან გაფართოების მიზნით.

ექსტრემალური პროგრამირების მეთოდის სასიცოცხლო ციკლის მოდელში ძირითადი ყურადღება მახვილდება საპრობლემო ამოცანის სწორად ჩამოყალიბებაში დამკვეთის მიერ ბიზნეს-ანალიტიკოსთან ერთად, ნაკლებად იხარჯება დრო უნივრსალური დიაგრამების აგებასა და საანგარიშო დოკუმენტაციის გაფორმებაზე, და რა თქმა უნდა, ხდება ძირითადი ეტაპების (კონსტრუირება-დაპროგრამება) ფაზათა შერწყმა [38].



1.13. განტერის სასიცოცხლო ციკლის მოდელი იტერაციით ობიექტ-ორიენტირებული დაპროგრამების მეთოდისთვის

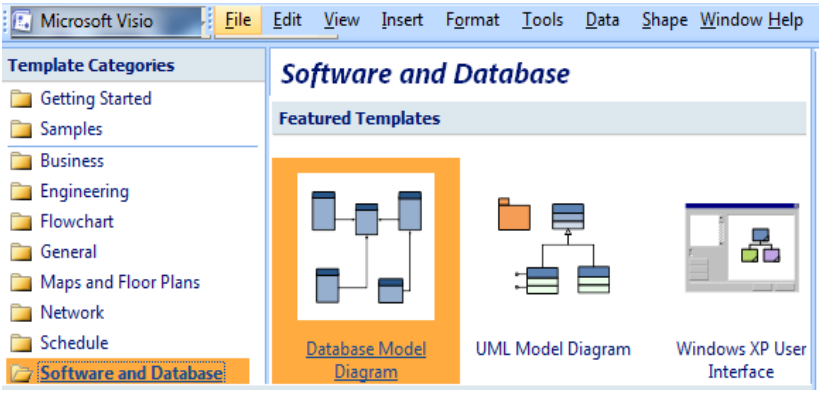
აქედან გამომდინარე, პროგრამული სისტემის მენეჯერი, კონკრეტული პროექტის ამოცანების და მოთხოვნების შესაბამისად, უნდა განსაზღვრავდეს როგორც პროგრამირების მეთოდის, ეტაპთა

ფაზების და იტერაციათა მოთხოვნების შერჩევა-ფორმირებას, ასევე მუშა გუნდის შემადგენლობას. ამ პროცესში მონაწილე როლებია: დამკვეთი, პროექტის მენეჯერი, ბიზნეს-პროცესების სპეციალისტი (ბიზნეს-ანალიტიკოსი), სისტემის არქიტექტორი, დეველოპერი-პროგრამისტი, ტესტირების სპეციალისტი და სხვ. პროგრამული სისტემის პროექტის მენეჯერი ახორციელებს ყველა საკონტროლო წერტილის მონიტორინგს.

დიდი პროექტებისათვის, რომელშიც რესურსები და დროითი ფაქტორები, შედარებით კრიტიკული არაა, ხდება ობიექტ-ორიენტირებული მიდგომის ყველა ეტაპის და ფაზის გამოყენება შესაბამისი საკონტროლო წერტილების აუცილებელი მონიტორინგით და რეპორტებით. ამ დროს სრული მოცულობით ხორციელდება უნიფიცირებული მოდელირების ენის (UML/2) და შესაბამისი ინსტრუმენტული საშუალების, მაგალითად, MsVisio ან Enterprise Architect პაკეტის გამოყენება [41,42].

1.6. Ms_Visio სამუშაო გარემო

გამოყენებითი პროგრამული უზრუნველყოფის (Applied Software) ობიექტ-ორიენტირებული ანალიზის, დაპროექტების და რეალიზაციის ცალკეული ეტაპების მოდელირების მიზნით განიხილება UML მეთოდოლოგიის ინსტრუმენტული საშუალება MsVisio Professional. ეს პაკეტი ძალზე პოპულარული და მრავალფუნქციურია (ნახ.1.14) [44].

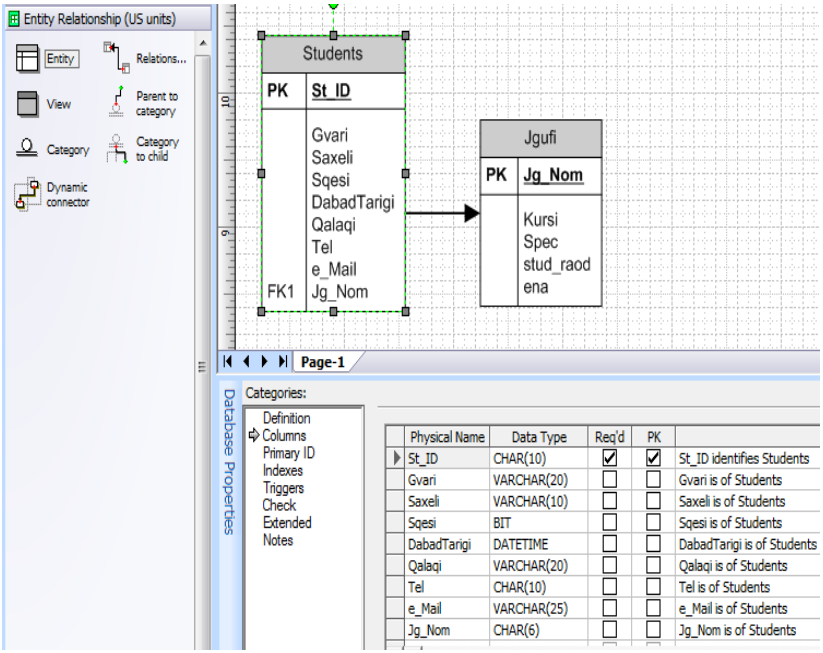


ნახ.1.14. Ms Visio–ს საწყისი გვერდი

ნახაზიდან Template Categories–ში ჩანს იმ სფეროთა კატეგორიების სიმრავლე, (მაგალითად, Business, Engineering, Network და ა.შ..), რომელთათვისაც შესაძლებელია ამ ინსტრუმენტის გამოყენება. ჩვენ განსაკუთრებულ ყურადღებას გავამახვილებთ კატეგორიაზე Software and Database, რომელიც ერთ-ერთი ეფექტური მექანიზმია პროგრამული და მონაცემთა ბაზების მართვის სისტემების დაპროექტების პროცესში შესაბამისი დიაგრამების ასაგებად.

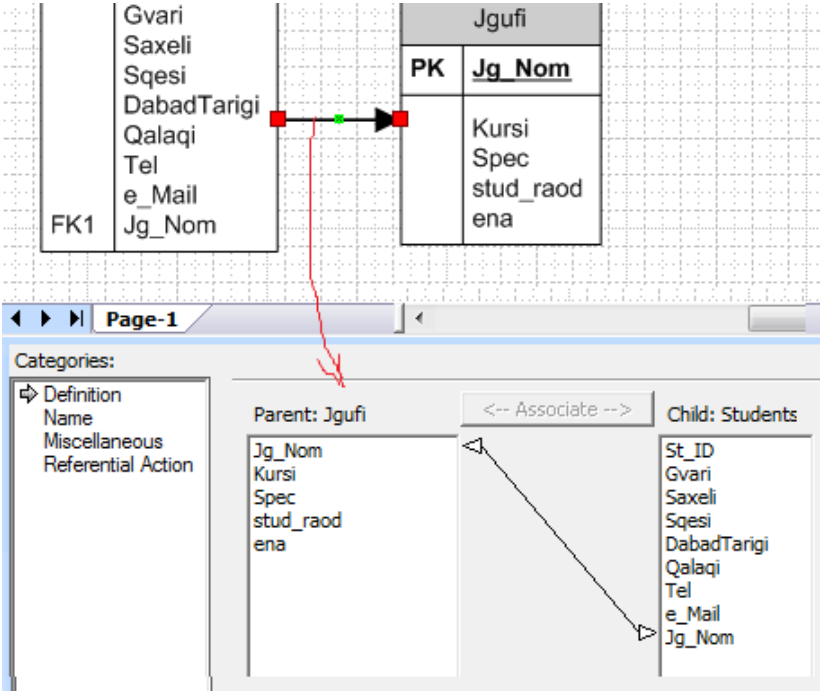
1.7. ER დიაგრამის ვიზუალური აგება

Database Model Diagram არჩევით ეკრანზე გამოიტანება მონაცემთა ER-მოდელის ასაგები რედაქტორი (Entity-Relationship Model). 1.15 ნახაზზე ნაჩვენებია ინტერფეისი კონკრეტული „სტუდენტთა-ჯგუფის“ მონაცემთა მოდელისათვის.



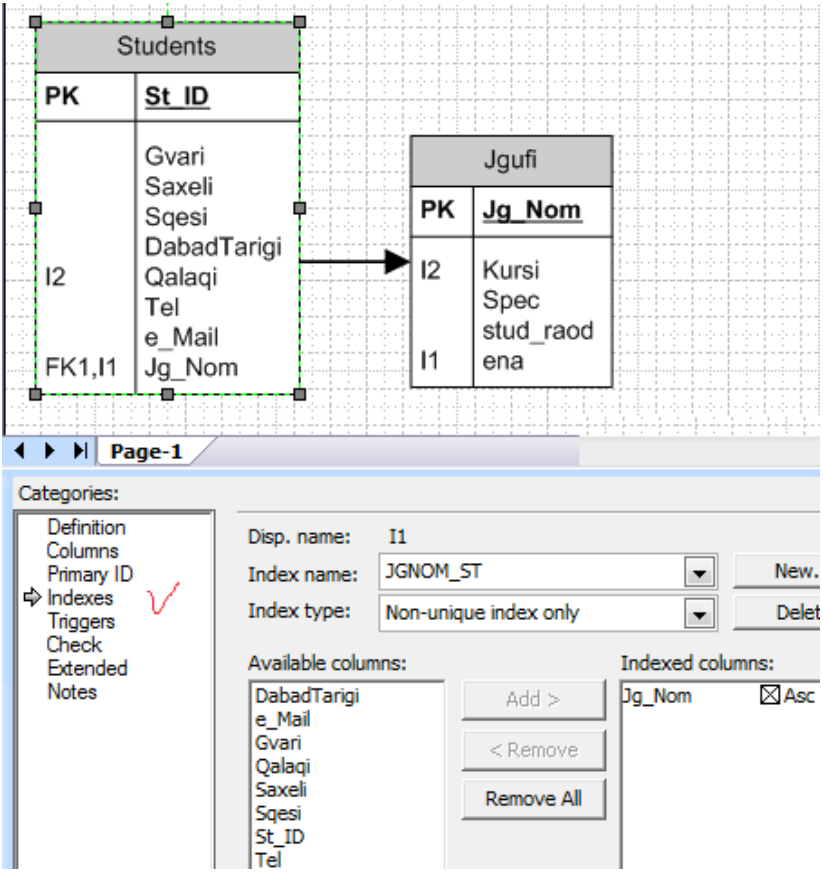
ნახ.1.15. Database Model Diagram –ის აგება

სტუდენტების და ჯგუფების არსებს (Entities) შორის რელაციური კავშირის ასაგებად ინსტრუმენტების პანელიდან ავიღებთ Relationship სიმბოლოს და ბოლოებით შევაერთებთ ცხრილებს. ისარი მიმართული უნდა იყოს შვილიდან მშობლისკენ. თუ შეერთება განხორციელდა სწორად, მაშინ შვილის ატრიბუტთა ჩამონათვალში ავტომატურად გაჩნდება მშობლის გასაღებური ატრიბუტის შესაბამისი ველი (ნახ.1.16). ჩვენ შემთხვევაში ესაა Jg_Nom ატრიბუტი. სტუდენტის ცხრილში მას წინ მიეწერება FK1, ანუ Foreign Key 1.



ნახ.1.16. რელაციური კავშირი „მშობელი-შვილი“ ობიექტებს შორის

1.17 ნახაზზე ნაჩვენებია ცხრილებში ინდექსური I1, I2 ველების განსაზღვრა. მათი დანიშნულებაა ინდექსური ფაილების შექმნა, რომლებშიც სტრიქონები (ბაზის ჩანაწერები) მოწესრიგდება კლებადობით (Desc) ან ზრდადობით (Asc).



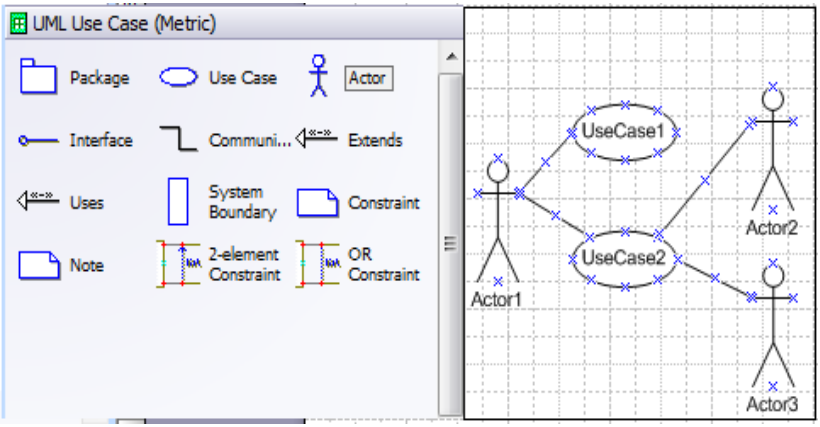
ნახ.1.17. ინდექსური ატრიბუტების განსაზღვრა

მონაცემთა ძებნის (Find) დაჩქარება ხორციელდება ასეთ ატრიბუტებზე პირდაპირი მიმართვის განხორციელებით მოწესრიგებულ ინდექსურ ფაილში.

1.8. UML-დიაგრამების ვიზუალური აგება

1.8.1. Use Case დიაგრამა

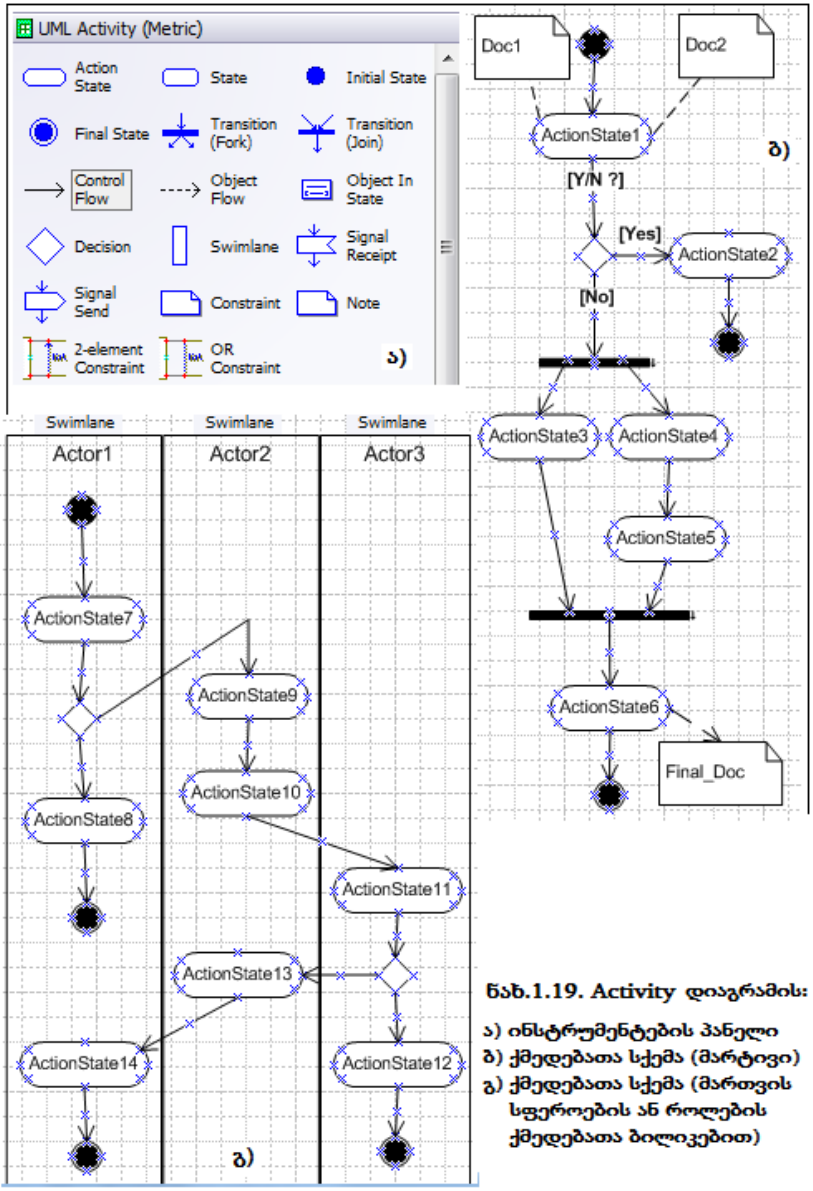
პირველი დიაგრამა, რომელიც UML ტექნოლოგიით უნდა აიგოს, არის გამოყენებით შემთხვევათა (პრეცედენტების) UseCase დიაგრამა. იგი როლების (Actors) და ფუნქციების (Actions) ურთიერთდაკავშირებული სქემაა (ნახ.1.18). აქ UseCase1 ეკუთვნის მხოლოდ პირველ როლს, ხოლო UseCase2 – ის შესასრულებლად ორივე როლი მონაწილეობს.



ნახ.1.18. UseCase დიაგრამის აგების ინტერფეისი

1.8.2. Activity დიაგრამა

ბიზნეს-პროცესების და ბიზნეს-წესების გაცნობის, ანალიზის და სტრუქტურული ფორმალიზაციის საფუძველზე აიგება აქტიურობის, ან ქმედებათა დიაგრამა. იგი კონკრეტული როლის (როლების) კონკრეტული ფუნქციაა, რომელიც შედეგება იერარქიულად სივრცესა და დროში დალაგებული მიმდევრობით ან პარალელურად შესასრულებელი სუბ-ქმედებებისგან. აქვს ერთი დასაწყისი და რამდენიმე შესაძლო დასასრული, ბიზნეს-წესებით განსაზღვრული განშტოების ან შეერთების პროცედურები, საწყისი, შუალედური ან საშედეგო დოკუმენტაცია და ა.შ. (ნახ.1.19 ა-გ).



ნახ.1.19. Activity დიაგრამის:

ა) ინსტრუმენტების პანელი
ბ) ქმედებათა სქემა (მარტივი)
გ) ქმედებათა სქემა (მართვის სფეროების ან როლების ქმედებათა ბილიკებით)

ქმედებათა დიაგრამა მიეკუთვნება პროცესების აღწერის დინამიკურ მოდელს, იგი ასახავს საკვლევი ობიექტის ქცევას.

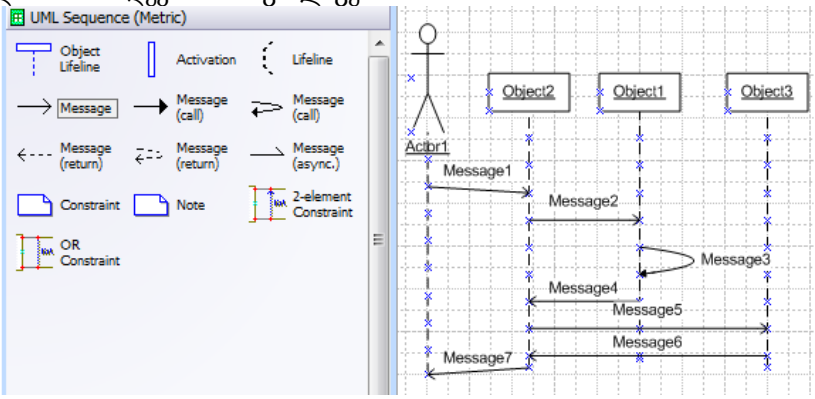
ასეთი დინამიკური მოდელების პროცესების გამოსაკვლევად გამოიყენება პეტრის ქსელები [19,44].

საბოლოოდ, ამ ორი სახის (UseCase, Activity) დიაგრამათა ერთობლიობის ანალიზის საფუძველზე კეთდება დასკვნები საავტომატიზაციო ობიექტის მართვის სისტემის ფუნქციონალური და არაფუნქციონალური მოთხოვნილებების განსაზღვრის შესახებ.

ამავდროულად დგება მომავალი პროგრამული სისტემის შექმნის ტექნიკური დავალება შესაბამის ეკონომიკურ გაანგარიშებებთან ერთად, რომელიც დამკვეთ ორგანიზაციის ხელმძღვანელობასთან კონსულტაციების შემდეგ ორმხრივად მტკიცდება. ამის შემდეგ იწყება ობიექტ-ორიენტირებული ანალიზის ეტაპი, რომელზეც აიგება სისტემის მომხმარებელთა ინტერაქტიული სქემები: მიმდევრობითი და თანამოქმედების დიაგრამათა სახით.

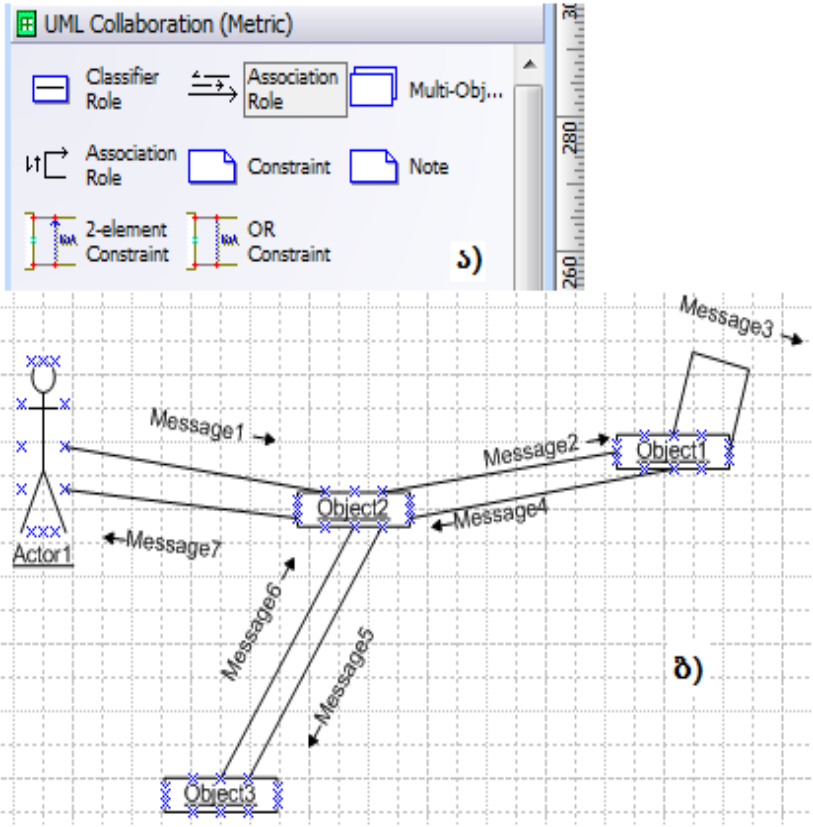
1.8.3. Sequence და Collaboration დიაგრამები

მიმდევრობითობის დიაგრამა აღწერს საპრობლემო სფეროს კონკრეტული ამოცანის შესრულების სცენარს. აქ ხდება როლის სისტემასთან ურთიერთქმედების ბიზნეს-პროცესის ქმედებათა და მათი მაინიცირებელ, სინქრონულ ან ასინქრონულ შეტყობინებათა დროში მიმდევრობით განლაგება.



ნახ.1.20. Sequence დიაგრამის აგების ინტერფეისი

1.21 ნახაზზე ნაჩვენებია თანამოქმედების დიაგრამის აგეწბის ინსტრუმენტული პანელი (ა) და თვით დიაგრამა (ბ), რომელიც შესაბამისი მიმდევრობითობის 1.20 დიაგრამის ტრანსფორმაციით იქნა მიღებული. აქ შეტყობინებათა და მონაცემთა გაცვლის მიმდევრობა არაა დროში დალაგებული, არამედ ჩანს კლასის ობიექტებს შორის კავშირების და ინფორმაციული ნაკადების გაცვლის სემანტიკა.

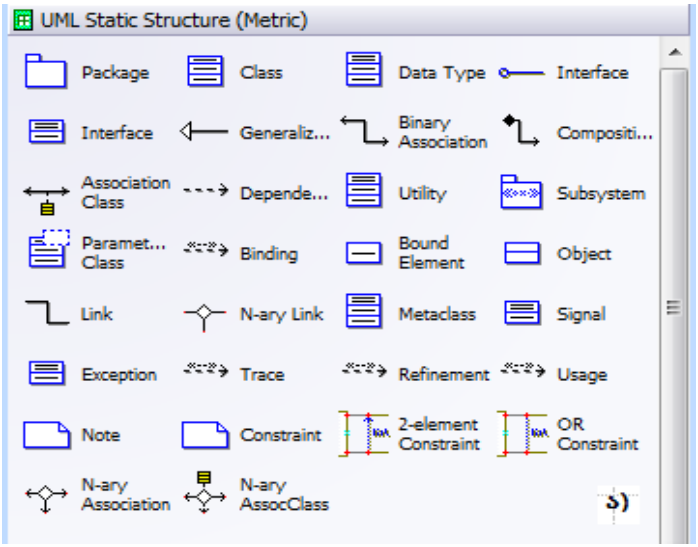


ნახ.1.21. Collaboration დიაგრამის აგების ინტერფეისი (ა) და სქემის მაგალითი (ბ)

1.8.4. Class- და Class-Assotiation დიაგრამები

კლასი ერთგვაროვან ობიექტთა ერთობლიობის სტრუქტურაა. მაგალითად, ყველა მოქალაქე (ზოგადად ერსონ), სტუდენტები, ლექტორები, ავტომანქანები, ცხოველები და ა.შ.

ტერმინი ,,კლასიფიკაცია“ სწორედ განსახილველი სფეროს ობიექტების სისტემატურ მოწესრიგებას ემსახურება (გენერალიზაცია (იერარქიაში განზოგადება ზევით) და სპეციფიკაცია (იერარქიაში დეტალიზაცია ქვევით)). MsVisio-ში კლასების და კლასთაშორის კავშირების მოდელირებისთვის გამოიყენება Static Structure ინსტრუმენტების პანელი (ნახ.1.22-ა).

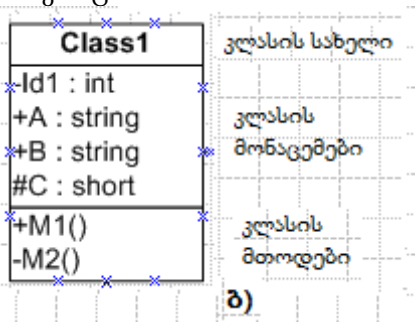


ნახ.1.22-ა

ობიექტ-ორიენტირებული მოდელირების და პროგრამირების გაგებით, კლასი არის ,,დასახელების“, ,,კლასის მონაცემების“ და ,,კლასის მეთოდების“ ინკაფსულაცია.

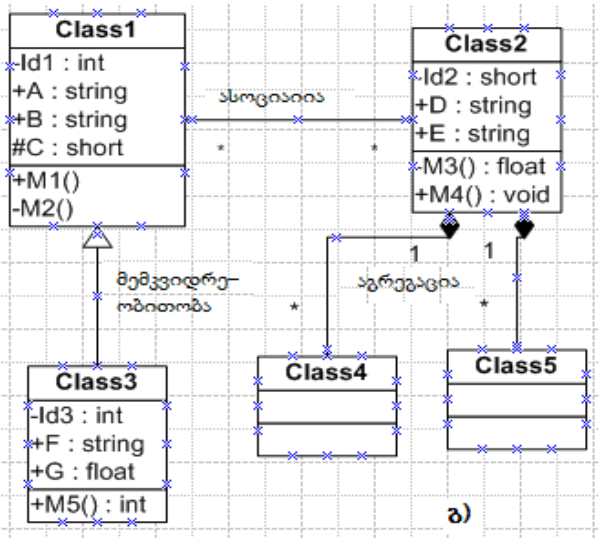
მართვის სფეროს შესაბამისი კლასი, ზოგადად ასე უნდა გამოიყურებოდეს (ნახ.1.22-ბ). კლასის ატრიბუტებს შეესაბამება მონაცემთა გარკვეული ტიპი (int, float, string ან სხვ.) და

„ხილვადობის“ („-“ private, „+“ public, „#“ protection). ასევეა კლასის მეთოდებისთვისაც.



ნახ.1.22. ინკაფსულაცია

კლასის მეთოდები (ან ფუნქციები) ის პროგრამული მოდულებია, რომლებიც ამუშავებს ამ კლასის მონაცემებს. მათი ინიციალიზაცია ხდება გარედან შემოსული შეტყობინების საფუძველზე.



ნახ.1.22. კლასთაშორისი კავშირების დიაგრამა (გ)

კლასებსშორისი კავშირები შეიძლება იყოს: მემკვიდრეობითი, აგრეგატული, რელაციური და ასოციაციური (ნახ.1.22-გ):

- **მემკვიდრეობითი** (Generalization) ასახავს „გენეტიკურ“, განზოგადოებულ კავშირებს კლასებს შორის. ასეთ დროს ერთი კლასი („შვილი“) მთლიანად იღებს მეორე კლასის („მშობელი“) ყველა ატრიბუტს, მეთოდს და კავშირს;
- **აგრეგირებული** (Aggregation) ნიშნავს კავშირს „მთელი“-„ნაწილი“. მაგალითად, „ავტომობილი“ - „ძარა, ძრავი, საბურავები და ა.შ.“;
- **ასოციაციური** (Assotiation) ნიშნავს სემენტიკურ კავშირს კლასებს შორის. ის შეიძლება გამოისახოს ერთ- ან ორმიმართულებიანი (იგივეა, რაც უისრო) ხაზით. ისარი გვიჩვენებს შეტყობინების გადაცემის მიმართულებას. ასოციაციური კავშირის რეალიზება ხდება ერთ კლასში დამატებით მეორე კლასის ატრიბურტის ჩასმით. ეს ჰგავს პირველადი (Primary) და მეორადი გასაღებური ატრიბუტების შეერთებას;
- **რელაციური** (Dependency) ნიშნავს ერთი კლასის დამოკიდებულებას მეორეზე. იგი ერთმიმართულებიანი წყვეტილი ისრით გამოიხატება. მასში დამატებითი დამაკავშირებელი ატრიბუტები არ გამოიყენება.

1.23 ნახაზზე ილუსტრირებულია კლასთა ასოციაციის დიაგრამა მემკვიდრეობითი კავშირების საფუძველზე. ისარი მიმართულია „შვილიდან“ „დედისკენ“, რაც მათ ცალსახა დამოკიდებულებაზე მეტყველებს. „შვილს“ ჰყავს ერთი „დედა“, ხოლო „დედას“ შეიძლება ჰყავდეს რამდენიმე „შვილი“, ამიტომაც ეს არაა ცალსახა.

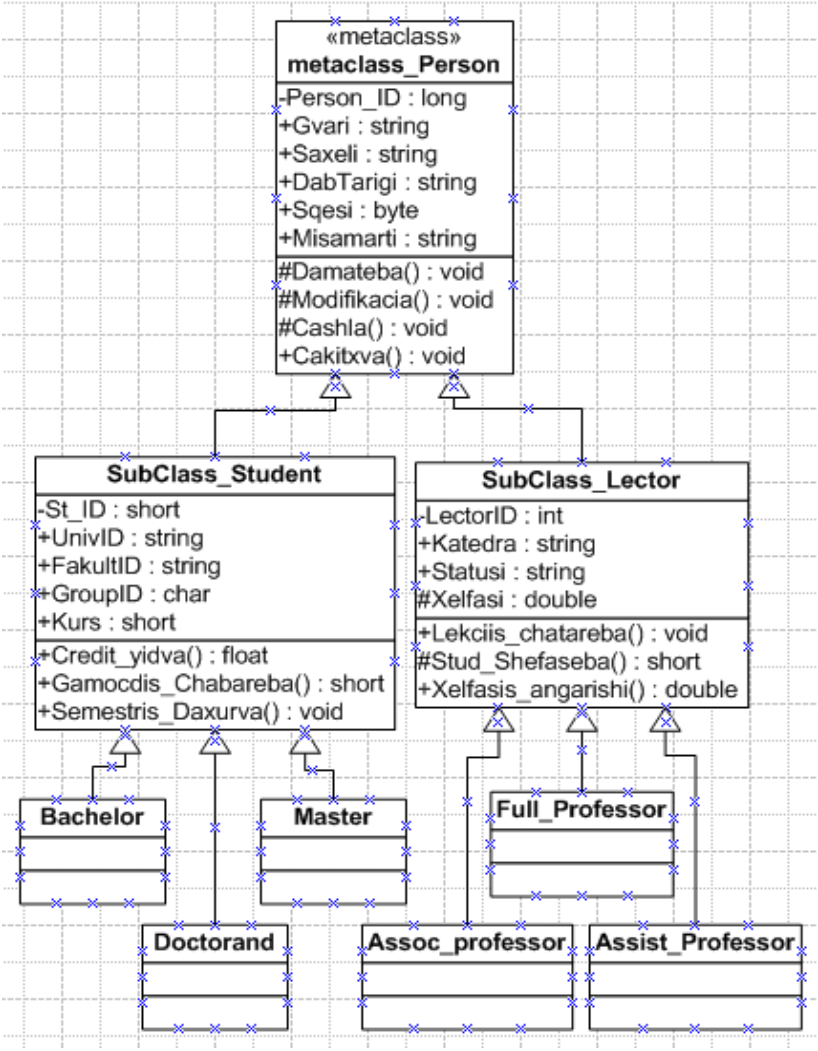
მშობელი კლასი ლიტარატურაში ზოგჯერ „მეტაკლასად“ (MetaClass) მოიხსენიება, რომელიც შედგება ქვეკლასებისგან (SubClasses). შეიძლება იერარქიაში ქვეკლასი იყოს მის ქვევით მდგარი კლასისთვის მეტაკლასი. მაგალითად, SubClass_Student არის ქვეკლასი MetaClass_Person კლასისთვის და, ამავდროულად იგი არის მეტაკლასი სამი ქვეკლასისთვის: Bachelor, Master და Doctorand.

იგივე შეიძლება ითქვას კლასებისთვის:

Metaclass_Person <- SubClass_Lector <- {Full_Professor, Assoc_Professor, Assist_Professor},

სადაც როლები ასეა განაწილებული:

„მშობელი“-Person, „შვილი“-Lector და „შვილიშვილები“ Full_Professor, Assoc_Professor, Assist_Professor.

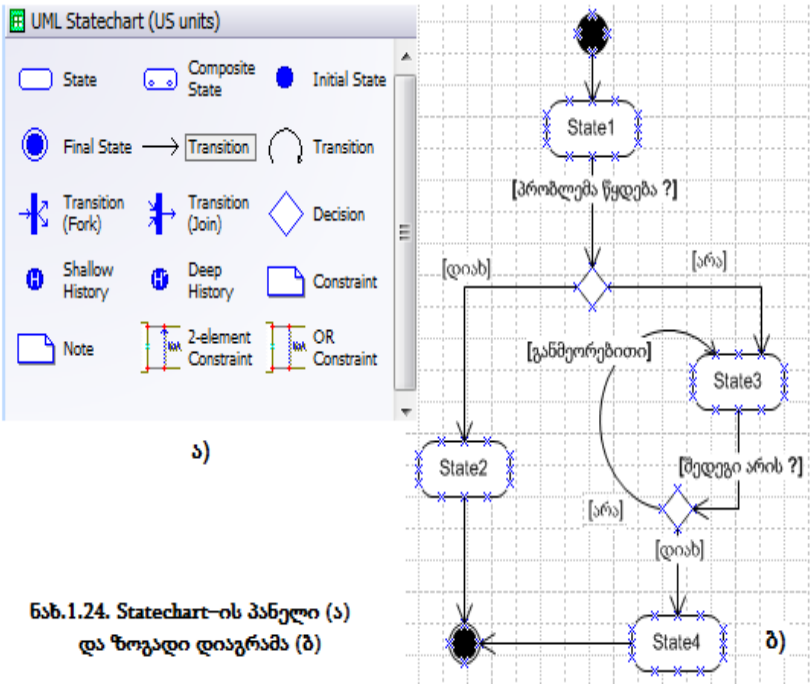


ნახ.1.23. Class-Assotiation დიაგრამა მემკვიდრეობითი კავშირებით

1.8.5. Statechart დიაგრამა

ყოფაქცევის დიაგრამებიდან ადრე განვიხილეთ ქმედებათა (Activity) დიაგრამა და ინტერაქტიული (Sequence, Collaboration) დიაგრამები. არსებობს კიდევ ერთი ასეთი სახის დიაგრამა, **კლასების მდგომარეობათა** დიაგრამა - Statechart-D. იგი აღწერს ქმედებებს, ობიექტთა მდგომარეობებს, მდგომარეობათა გადასვლებს და მოვლენებს.

1.24-ა,ბ ნახაზებზე ნაჩვენებია ინსტრუმენტული პანელის და მდგომარეობათა დიაგრამის ფრაგმენტი ზოგადი მაგალითისთვის.



ნახ.1.24. Statechart–ის პანელი (ა) და ზოგადი დიაგრამა (ბ)

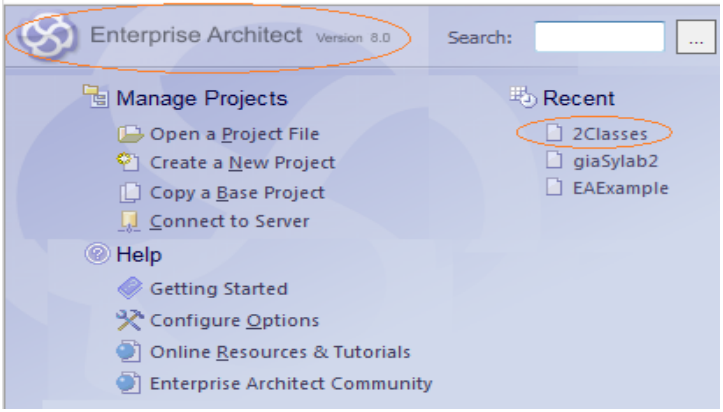
მისი გამოყენება ყველა კლასისთვის არაა საჭირო. აუცილებელია მხოლოდ მაშინ, როდესაც კლასი შეიძლება იმყოფებოდეს რამდენიმე მდგომარეობაში და თითოეულ მათგანში მისი ქცევა იყოს სხვადასხვანაირი.

1.8.6. კლასების დიაგრამიდან პროგრამული კოდის გენერაცია

თანამედროვე CASE-ტექნოლოგიები, რომლებიც სისტემების დაპროგრამების ავტომატიზაციაზეა ორიენტირებული, მაგალითად, Rational Rose, Visual Paradigm, Enterprise Architect და მრავალი სხვა [42], ახორციელებენ რევერსული დაპროგრამების კონცეფციას. ანუ კლასების დიაგრამიდან შესაძლებელია პროგრამული კოდის გენერაცია და პირიქითაც, კოდიდან აიგება ავტომატურად გრაფიკული დიაგრამა.

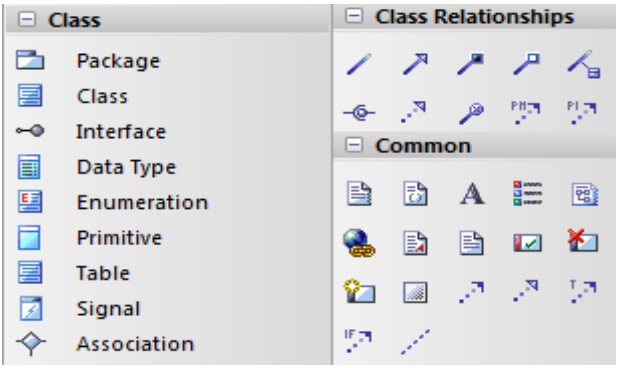
MsVisio არ მიეკუთვნება ასეთი სიმძლავრის ინსტრუმენტს. მისი საშუალებით დიალოგურ რეჟიმში იხაზება დიაგრამები (UML-ის სტანდარტულ აღნიშვნათა საერთაშორისო ნორმებით), მაგრამ კოდის გენერაცია არაა შესაძლებელი.

Ms Visual Studio .NET Framework-ისთვის შექმნილია ინსტრუმენტები და მათი ინტეგრაციით .NET გარემოში, შესაძლებელია დიაგრამებიდან კოდის გენერაცია. ასეთი პაკეტები ყოველთვის ფასიანია და ძვირადღირებული. აქ განვიხილავთ SparX ფირმის Enterprise Architect პროდუქტის ამ კონკრეტულ ფუნქციას, კლასების დიაგრამიდან კოდის გენერაციის ამოცანას. 1.25 ნახაზზე ნაჩვენებია პაკეტის ამუშავების საწყისი გვერდი.

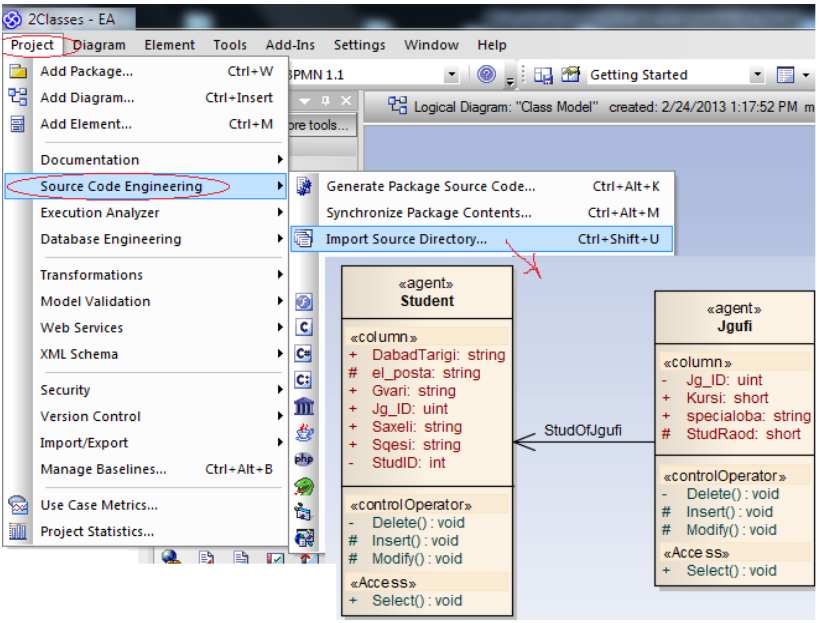


ნახ.1.25. Enterprose Architect საწყისი გვერდი

მაგალითისთვის ვიხილავთ ამ გარემოში ორი კლასის (2Clases მოდელი) აგების და მათი პროგრამულ კოდში გადაყვანის ამოცანას. 1.26 ნახაზზე მოცემულია Enterprise Architect პაკეტის კლასთა დიაგრამის აგების ინსტრუმენტების პანელი.

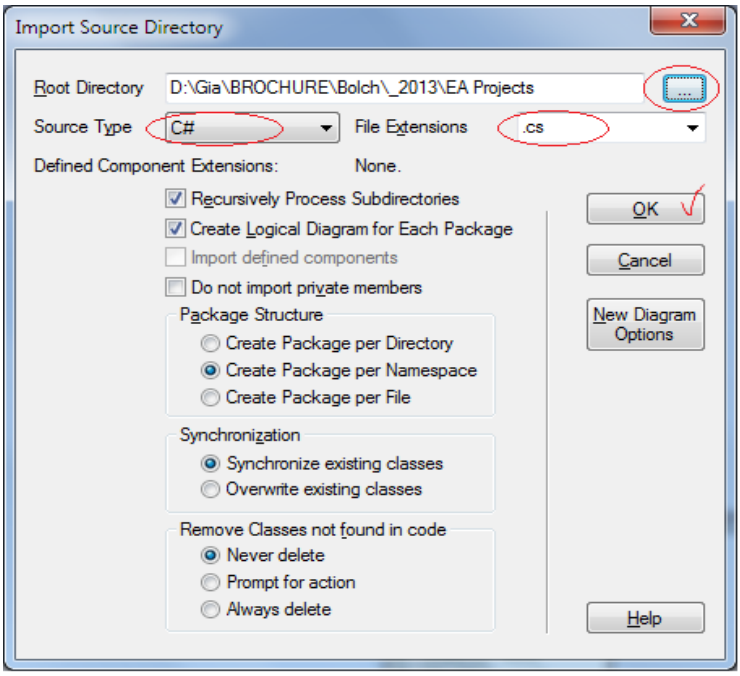


ნახ.1.26.

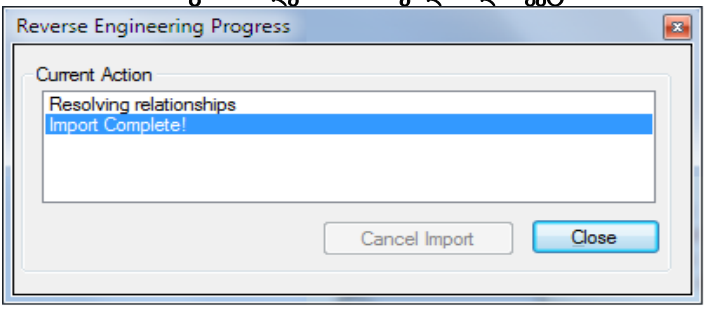


ნახ.1.27. Student და Jgufi კლასების მომზადება „Code Engineering“ პროცესისთვის Enterprise Architect გარემოში

Import Source Directory-ის არჩევის შემდეგ გამოვა 1.28 ნახაზზე ნაჩვენები ფანჯარა, რომელშიც უნდა განისაზღვროს ზოგიერთი მნიშვნელოვანი პარამეტრი, მაგალითად, ენა (C#), მომავალი კოდის შესანახი ადგილი (დირექტორია) და ა.შ., ბოლოს „Ok "და მივიღებთ 1.29 ნახაზზე მოცემულ შედეგს.

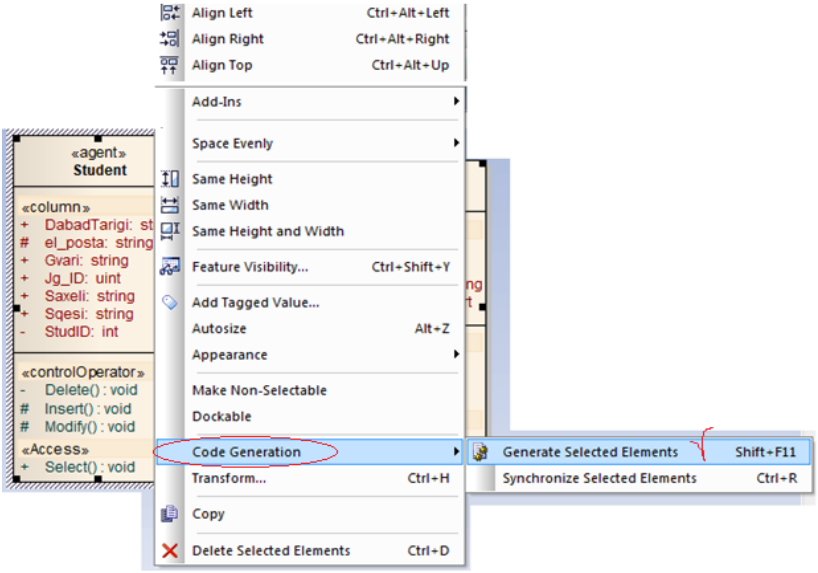


ნახ.1.28. განისაზღვროს C#-კოდის დირექტორია

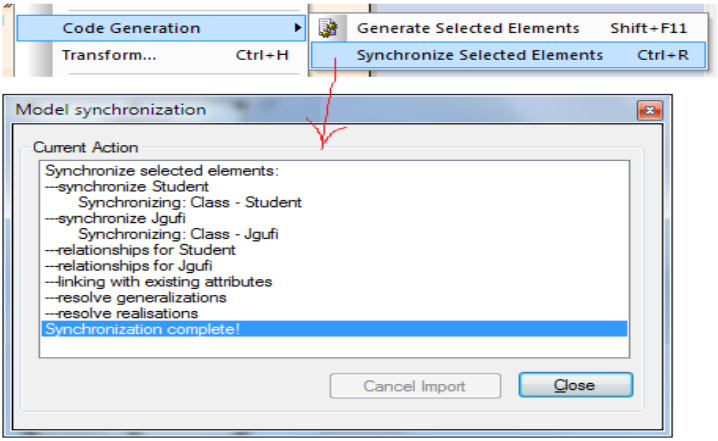


ნახ.1.29. იმპორტი დასრულებულია

ახლა უნდა ჩატარდეს უშუალოდ კოდის გენერაცია წინასწარ მომზადებული (Student-Jgufi) კლასების დიაგრამიდან. ვააქტიურებთ კლასებს და მაუსის მარჯვენა ღილაკით გამოტანილ კონტექსტური მენიუდან ვირჩევთ „Code Generation“-ს.

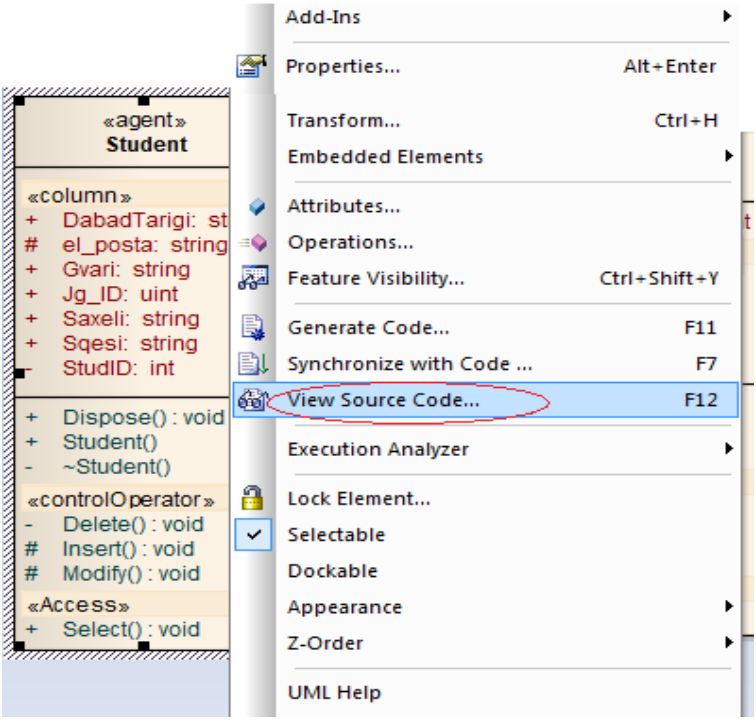


ნახ.1.30. კოდის გენერაციის დაწყება



ნახ.1.31. არჩეული ელემენტების სინქრონიზაციის შედეგი

ბოლო ფაზაზე საჭიროა გამოვიტანოთ ეკრანზე კლასების ბაზაზე გენერირებული კოდის ლისტინგები.

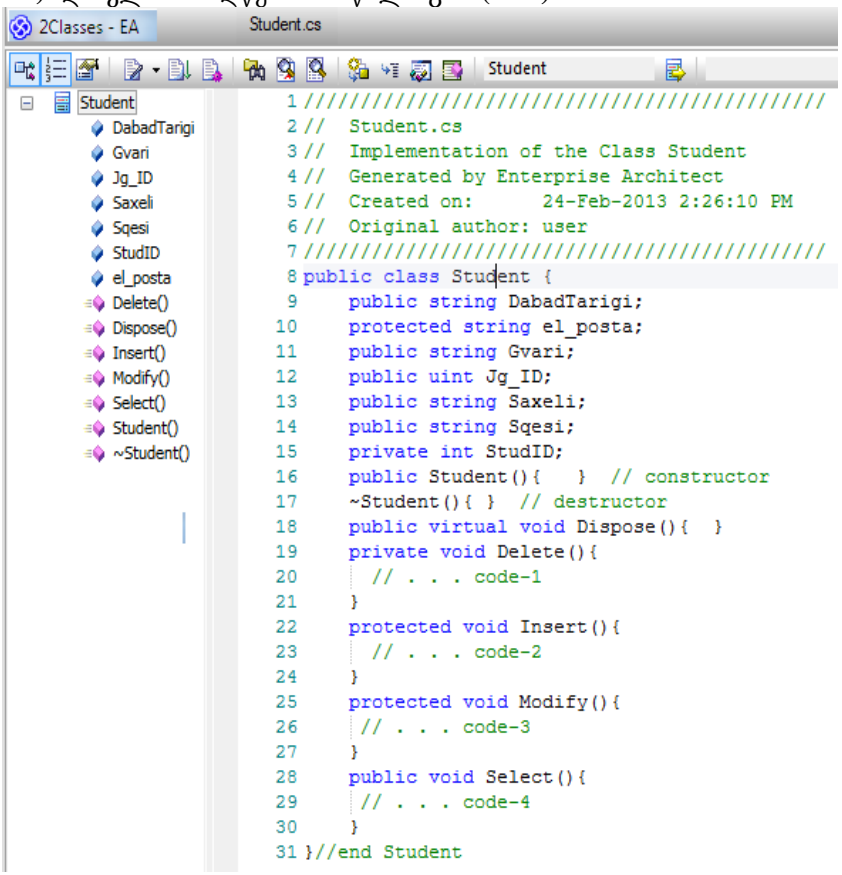


ნახ.1.32. კოდის გამოტანის პუნქტი მენიუში

1.33 ნახაზზე ნაჩვენებია Enterprise Architect გარემოში Student კლასის დიაგრამიდან ავტომატურად გენერირებული C#-კოდის საწყისი ტექსტი.

ფანჯრის მარცხენა ნაწილში მოთავსებულია Student კლასის კაფსულა, თავისი მონაცემებით და მეთოდებით, მათ შორის კონსტრუქტორით და დესტრუქტორით (მეთოდები, რომელთაც აქვს კლასის იდენტური სახელი).

ფანჯრის მარჯვენა ნაწილში სისტემას გამოაქვს პროგრამის ტექსტი, რომელიც შედგება კომენტარული ნაწილის (სტრიქონები 1-7) და კლასის აღწერის ნაწილისგან (8-31).



ნახ.1.33-ა. C#-კოდის ლისტინგი Student კლასისთვის

მომდევნო ლისტინგში მოცემულია Jgufi კლასის საწყისი ტექსტი. აქაც, C#-კოდის ტექსტი შედგემა კომენტარული ნაწილისგან, რომელშიც ასახულია პროგრამის ზოგადი მახასიათებლები, სახელი, ინსტრუმენტი, შექმნის თარიღი, ავტორი. პროგრამის ტექსტი კლასიკური ფორმატით აღიწერება

კლასის მონაცემები ხილვადობის private, public და protection ატრიბუტებით. შემდეგ მოსდევს კონსტრუქტორის public Jgufi() { } და დესტრუქტორის ~Jgufi() { } სტრიქონები. Dispose() მეთოდი გამოიყენება პროგრამის შესრულების დამთავრების შემდეგ ოპერაციული სისტემის მიერ გამოყოფილი რესურსების გასათავისუფლებლად.

```
///////////////////////////////////////////////////////////
// Jgufi.cs
// Implementation of the Class Jgufi
// Generated by Enterprise Architect
// Created on:      24-Feb-2013 2:26:15 PM
// Original author: user
///////////////////////////////////////////////////////////
public class Jgufi {
        private uint Jg_ID;
        public short Kursi;
        public string specialoba;
        protected short StudRaod;
        public Student m_Student;
        public Jgufi() {  }     // კონსტრუქტორი
        ~Jgufi() {  }    // დესტრუქტორი
        public virtual void Dispose() {  }
        private void Delete(){
        // . . . code-1
        }
        protected void Insert(){
        // . . . code-2
        }
        protected void Modify(){
        // . . . code-3
        }
        public void Select(){
        // . . . code-4
        }
}//end Jgufi
```

ამით დავამთავრეთ ჩვენ უნიფიცირებული მოდელირების ენის და მისი ინსტრუმენტების მოკლე აღწერა.

II თავი

რიგების თეორია (მასობრივი მომსახურების სისტემები)

2.1. მასობრივი მომსახურების მოდელები: ზოგადი პრინციპები და თეორიული ასპექტები

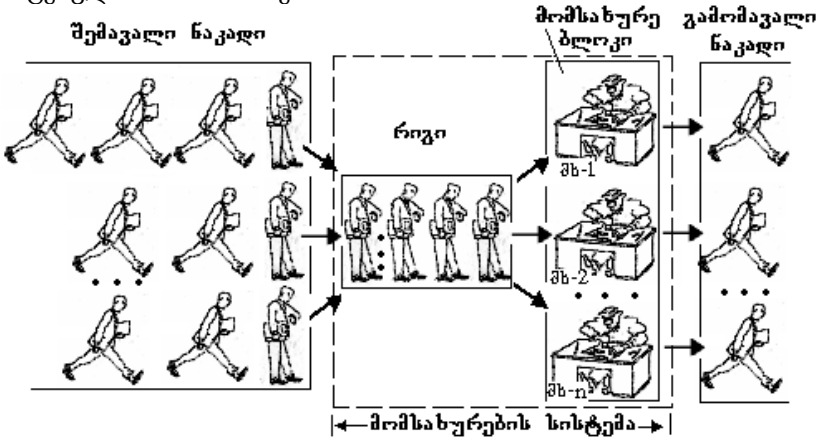
ადამიანთა პრაქტიკულ საქმიანობაში გვხვდება ბევრი ისეთი პროცესი, როდესაც მომსახურე საშუალებების შეზღუდული რაოდენობა ემსახურება მასიურ მოთხოვნილებებს. ამასთან, ჩვეულებრივი მომსახურე საშუალებების რაოდენობა ბევრად უფრო ნაკლებია, ვიდრე მოთხოვნილებათა რაოდენობა. მომსახურე საშუალებების შეზღუდული რაოდენობიდან გამომდინარე, ჩნდება მოთხოვნილებათა რიგები მომსახურების სფეროში [12,36,37].

მაგალითად: - მყიდველთა რიგი დიდი სასურსათო მაღაზიის სალაროსთან; - ტელევიზორების რიგი, რომლებიც საჭიროებს გარკვეულ შეკეთებას; - სამოქალაქო თვითმფრინავების ჯგუფი დიდ აეროპორტში, რომლებიც ელოდება აფრენის ნებართვას; - პროგრამების ერთობლიობა, რომლებიც გამზადებულია კომპიუტერზე სარეალიზაციოდ და ა.შ.

ეს მაგალითები შეიცავს მომსახურების განსხვავებულ ფიზიკურ შინაარს. მიუხედავად ამისა აღმოჩნდა, რომ მომსახურების ამ პროცესების დახასიათება და მათ შორის კავშირები შეიძლება გავაერთიანოთ ერთი ფორმალური თეორიის ჩარჩოში, კერძოდ, მასობრივი მომსახურების ან რიგების თეორიაში.

მასობრივი მომსახურების თეორიის ძირითად ამოცანას წარმოადგენს ოპტიმალური რაოდენობრივი დამოკიდებულების დადგენა მოთხოვნების შემავალ ნაკადს, მომსახურე საშუალებებსა და მოთხოვნების გამომავალ ნაკადს შორის. ამ ამოცანის ერთ–ერთი ძირითადი თავისებურება ისაა, რომ მათში მიმდინარე მოვლენები შემთხვევით ხასიათს ატარებს. კერძოდ, მოთხოვნათა შემავალი ნაკადი არის შემთხვევითი სიდიდე. მომსახურების დროც აგრეთვე შემთხვევით სიდიდეს წარმოადგენს და აქედან გამომდინარე, გამომავალი ნაკადიც შემთხვევით ხასიათს ატარებს.

მასობრივი მომსახურების სისტემის (მმს) ზოგადი სქემა მოცემულია 2.1 ნახაზზე.



ნახ.2.1. მმს-ის ზოგადი სქემა n-მომსახურე ხელსაწყოთი (მხ)

მასობრივი მომსახურების სისტემის ძირითად კომპონენტებს წარმოადგენს შემავალი ნაკადი, რიგი, მომსახურების საშუალებები და გამომავალი ნაკადი.

შემავალი ნაკადი შედგება მოთხოვნებისაგან, რომლებიც შედის მმს–ში მომსახურებისათვის.

მოთხოვნა არის მიმართვა მომსახურების მოთხოვნილებათა დასაკმაყოფილებლად.

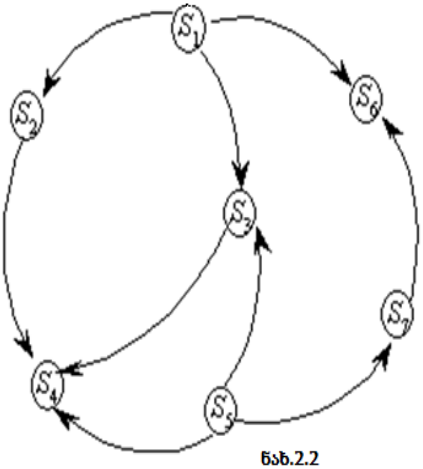
რიგს შეადგენს მოთხოვნები, რომლებიც იმყოფება მმ–ის სისტემაში და ელოდება მომსახურების დაწყებას.

გამომავალი ნაკადი – მოთხოვნები, რომლებიც ტოვებს სისტემას, როგორც მომსახურების შედეგად, ასევე მომსახურების გარეშე.

ამრიგად, მმს-ში მიმდინარე პროცესები განისაზღვრება შემთხვევითი ფაქტორებით. ასეთ პროცესებს ეწოდებათ შემთხვევითი. როგორც ცნობილია, სისტემას ახასიათებს დინამიკურობის თვისება, რომლის თანახმადაც იგი იცვლება დროის მიხედვით. ე.ი. გადადის ერთი მდგომარეობიდან მეორეში. თუ ეს გადასვლები შემთხვევით ხასიათს ატარებს, მაშინ ამბობენ,

რომ სისტემაში ხორციელდება შემთხვევითი პროცესი. მაგალითის სახით შეგვიძლია განვიხილოთ მომხმარებელთა მოთხოვნები მას–ში, რომლებიც შეიძლება იყოს რეგლამენტირებული (როცა შეკვეთები შედის დროის ფიქსირებულ მომენტში), ასევე შემთხვევითი (როცა შეკვეთები შედის დროის ნებისმიერ მომენტში). გასაგებია, რომ ამ შემთხვევაში მოთხოვნათა მომსახურების პროცესი შემთხვევითია.

თუ სისტემის მომავალი მდგომარეობის ალბათობა დამოკიდებულია მხოლოდ აწმყო მდგომარეობაზე (წინა მდგომარეობისგან დამოუკიდებლად), მაშინ მათ მარკოვის პროცესებს უწოდებენ. მარკოვის შემთხვევით პროცესს ეწოდება პროცესი დისკრეტული მდგომარეობებით, თუ სისტემის გადასვლა ერთი მდგომარეობიდან მეორეში ხორციელდება დროის დისკრეტულ მომენტებში, ე.ი. ნახტომებით. მარკოვის შემთხვევითი პროცესები განიხილება აგრეთვე უწყვეტი მდგომარეობებისათვის. ამ სისტემებში ერთი მდგომარეობიდან მეორეში გადასვლა ხდება მწყობრად. შემთხვევითი პროცესი შეიძლება წარმოვადგინოთ გრაფის სახით, რომლის მწვერვალები ასახავს მდგომარეობებს, ხოლო რკალები ერთი მდგომარეობიდან მეორეში გადასვლებს (ნახ.2.2). S-ით აღნიშნულია სისტემის i-ური მდგომარეობა.



ნახ.2.2

თუ სისტემის ერთი მდგომარეობიდან მეორეში გადასვლა ხდება დროის მკაცრად განსაზღვრულ მომენტებში, მაშინ ასეთი შემთხვევითი პროცესები წარმოადგენს პროცესებს დისკრეტული დროით.

თუ სისტემის ერთი მდგომარეობიდან მეორეში გადასვლა ხდება დროის შემთხვევით მომენტებში, მაშინ ასეთი პროცესები წარმოადგენს პროცესებს უწყვეტი დროით.

მარკოვის შემთხვევითი პროცესები დისკრეტული მდგომარეობითა და დროით, შეიძლება წარმოდგენილ იქნას S_1, S_2,..., S_i,..., S_n მდგომარეობათა მიმდევრობის სახით:

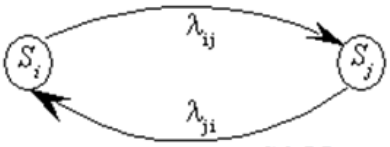
$$S_1 \rightarrow S_2 \rightarrow \ldots \rightarrow S_i \rightarrow \ldots \rightarrow S_n \qquad (2.1)$$

ზოგად შემთხვევაში მიმდევრობა შეიძლება იყოს ნებისმიერი, ე.ი. სისტემის გადასვლები შეიძლება განხორციელდეს ნებისმიერ მდგომარეობაში, მათ შორის ისეთშიც, რომელშიც იმყოფება სისტემა. მაგალითად:

$$S_1 \rightarrow S_2 \rightarrow S_2 \rightarrow S_4 \rightarrow S_6 \rightarrow S_3 \rightarrow S_5 \qquad (2.2)$$

როგორც აღვნიშნეთ, სისტემებში დისკრეტული მდგომარეობით ერთი მდგომარეობიდან მეორეში გადასვლები ხორციელდება ნახტომებით, ბიჯებით. თუ ყოველი ბიჯისთვის ნებისმიერი S_i მდგომარეობიდან S_j მდგომარეობაში გადასვლის ალბათობა არ არის დამოკიდებული იმაზე, თუ როგორ მივიდა სისტემა S_i მდგომარეობაში, მაშინ ასეთ მიმდევრობას ეწოდება მარკოვის ჯაჭვი.

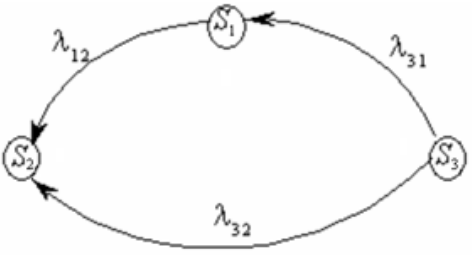
მარკოვის ჯაჭვის ანალიზის დროს, თითოეული გადასვლისას ნაჩვენები უნდა იყოს λ_{ij} ნაკადის ინტენსიურობა, რომელსაც სისტემა გადაჰყავს S_i მდგომარეობიდან S_j -ში. λ_{ij} ინტენსიურობას ჩვეულებრივ, მიუთითებენ მდგომარეობების გრაფის რკალებზე (ნახ.2.3).



ნახ.2.3

ვთქვათ, $p_i(t)$ არის ალბათობა იმისა, რომ t დროის მომენტში სისტემა იმყოფება S_i მდგომარეობაში. მარკოვის პროცესის მდგომარეობათა გრაფის მეშვეობით, შესაძლებელია განისაზღვროს $p_1(t)$, $p_2(t)$,..., $p_i(t)$,..., $p_n(t)$ მდგომარეობის ალბათობები კოლმოგოროვის განტოლებების თანახმად. განვიხილოთ ამ განტოლებების აგების პრინციპები მდგომარეოათა გრაფის

მაგალითზე, რომელიც გამოსახულია 2.4 ნახაზზე. მოცემული გრაფის მდგომარეობათა ალბათობები აღვნიშნოთ $p_1(t), p_2(t), p_3(t)$.



ნახ.2.4

პირველ რიგში განვსაზღვროთ $p_1(t)$. ამისათვის t–ს მივცეთ Δt ნაზრდი. შემდეგ განვსაზღვროთ $p_1(t + \Delta t)$ ალბათობა იმისა, რომ $t + \Delta t$ მომენტში სისტემა იქნება S_1 მდგომარეობაში. ეს მოვლენა შეიძლება განხორციელდეს 2 ხერხით:

1) t დროის მომენტში სისტემა იმყოფება S_1 მდგომარეობაში და Δt დროის განმავლობაში არ გამოვიდა ამ მდგომარეობიდან;

2) t დროის მომენტში სისტემა იმყოფება S_3 მდგომარეობაში და Δt დროის განმავლობაში გადავა S_1 მდგომარეობაში.

ალბათობა იმისა, რომ t დროის მომენტში სისტემა იყო S_1 მდგომარეობაში და Δt დროის განმავლობაში არ გამოვიდა S_1–დან, ტოლია ორი ალბათობის ნამრავლისა. პირველი თანამამრავლი $p_1(t)$ არის ალბათობა იმისა, რომ t დროის განმავლობაში სისტემა იმყოფება S_1 მდგომარეობაში, ხოლო მეორე თანამამრავლი $p_{12}(\Delta t)$ არის პირობითი ალბათობა იმისა, რომ Δt დროის განმავლობაში სისტემა არ გადავა S_2 მდგომარეობაში იმ პირობით, რომ Δt მომენტამდე სისტემა იმყოფებოდა S_1 მდგომარეობაში. ეს პირობითი ალბათობა განისაზღვრება შემდეგნაირად. შემთხვევითი პროცესების თეორიიდან ცნობილია, რომ

$$\lim_{\Delta t \to 0} \frac{p_{ij}(\Delta t)}{\Delta t} = \lambda_{ij}, \qquad (2.3)$$

სადაც $p_{ij}(\Delta t)$ პირობითი ალბათობაა იმისა, რომ Δt დროის განმავლობაში სისტემა S_i მდგომარეობიდან გადავა S_j მდგომარეობაში, იმ პირობით, რომ Δt დროის მომენტამდე სისტემა იმყოფებოდა S_i მდგომარეობაში.

(2.3)–დან შეგვიძლია დავწეროთ, რომ

$$p_{ij}(\Delta t) = \lambda_{ij}\Delta t + o(\Delta t),$$

სადაც $o(\Delta t)$ უსასრულოდ მცირე სიდიდეა უფრო მაღალი ხარისხის, ვიდრე (Δt). ჩვენი მაგალითისათვის:

$$p_{12}(\Delta t) = \lambda_{12}\Delta t + o(\Delta t),$$

ხოლო საძიებელი პირობითი ალბათობა ტოლია:

$$1-p_{12}(\Delta t)=1 - \lambda_{12}\Delta t,$$

სიზუსტით უსასრულო მცირე $o(\Delta t)$–მდე.

მეორე შემთხვევაში ალბათობა იმისა, რომ სისტემა t დროის მომენტში იმყოფებოდა S_3 მდგომარეობაში და Δt დროის მონაკვეთში გადავიდა S_1 მდგომარეობაში, ასევე განისაზღვრება $p_3(t) \cdot p_{31}(\Delta t)$ ნამრავლით, სადაც $p_{31}(\Delta t)$ – გადასვლის პირობითი ალბათობაა S_3 მდგომარეობიდან S_1 მდგომარეობაში. $p_{31}(\Delta t)$ ალბათობა განისაზღვრება შემდეგნაირად:

$$p_{31}(\Delta t) = \lambda_{31} \cdot \Delta t.$$

ამის შემდეგ, $p_1(t + \Delta t)$ ალბათობას განვსაზღვრავთ, როგორც 1 და 2 მოვლენების ალბათობათა ჯამს (რადგანაც ეს მოვლენები არათავსებადია):

$$p_1(t + \Delta t) = p_1(t)\,(1 - \lambda_{12}\,\Delta t) + p_3(t)\,\lambda_{31}\,\Delta t.$$

მოცემულ გამოსახულებაში $p_1(t)$ გადავიტანოთ ტოლობის მარცხენა მხარეს და ორივე მხრე გავყოთ Δt–ზე, მივიღებთ:

$$\frac{p_1(t + \Delta t) - p_1(t)}{\Delta t} = \lambda_{12} p_1(t) + \lambda_{31} p_3(t),$$

მოცემულ განტოლებაში გადავიდეთ ზღვარზე, როცა $\Delta t \to 0$

$$\lim_{\Delta t \to 0} \frac{p_1(t+\Delta t) - p_1(t)}{\Delta t} = \lambda_{12} p_1(t) + \lambda_{31} p_3(t).$$

მაშინ მივიღებთ:

$$\frac{dp_1(t)}{dt} = \lambda_{12} p_1(t) + \lambda_{31} p_3(t).$$

დიფერენციალური განტოლებები დანარჩენი მდგომარეობებისთვის განისაზღვრება ანალოგიურად. საბოლოოდ, მივიღებთ დიფერენციალურ განტოლებათა სისტემას:

$$\begin{cases} \frac{dp_1(t)}{dt} = \lambda_{12} p_1(t) + \lambda_{31} p_3(t) \\ \frac{dp_2(t)}{dt} = \lambda_{12} p_1(t) + \lambda_{32} p_3(t) \\ \frac{dp_3(t)}{dt} = -(\lambda_{12} + \lambda_{32}) p_3(t) \end{cases}.$$

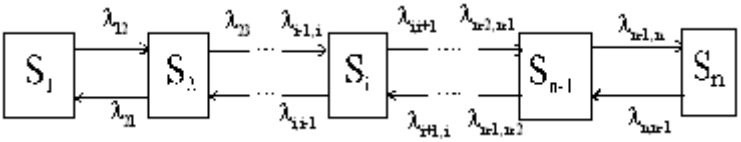
მიღებული სისტემა წარმოადგენს კოლმოგოროვის განტოლებათა სისტემას.

$p_i(t)$ ალბათობის განსაზღვრისათვის აუცილებელია მოვახდინოთ მოცემული სისტემის ინტეგრირება, რისთვისაც აუცილებელია მივცეთ მას საწყისი პირობები, რომელიც ჩვეულებრივ განისაზღვრება S სისტემის საწყისი მდგომარეობით. მაგალითად, თუ S სისტემისათვის მივიღებთ, რომ საწყის მომენტში ის იმყოფებოდა S_2 მდგომარეობაში, მაშინ საწყისი პირობები იქნება: $t = 0, p_1 = 0, p_2 = 1, p_3 = 0.$

ზოგად შემთხვევაში კოლმოგოროვის განტოლებათა სისტემა შეიძლება აიგოს განსაზღვრული წესის გამოყენებით საანალიზო პროცესის მდგომარეობების გრაფის არსებობის შემთხვევაში. მოვიყვანოთ ეს წესი. თითოეული S_i მდგომარეობისათვის განისაზღვრება ამ მდგომარეობის ალბათობის წარმოებული და ჩაიწერება განტოლების მარცხენა მხარეში. მარჯვენა ნაწილის წევრთა რაოდენობა განისაზღვრება რკალების რაოდენობით,

რომლებიც დაკავშირებულია მოცემული S მდგომარეობის შესახებ გრაფის მწვერვალთან. ამასთან, განტოლების მარჯვენა მხარის ნებისმიერი წევრი განისაზღვრება რკალზე ნაჩვენები გადასვლის ინტენსიურობის ნამრავლით იმ მდგომარეობის ალბათობაზე, საიდანაც გამოდის რკალი. თუ რკალი გამოდის მწვერვალიდან, მაშინ მისი შესაბამისი წევრი განტოლებაში იქნება მინუს ნიშნით. თუ რკალი შედის მწვერვალში, მაშინ – პლუს ნიშნით.

მარკოვის ჯაჭვს ხშირად გამოსახავენ ,,გაქრობისა და გამრავლების“ პროცესის სახით. გრაფს – ‘‘გაქრობა და გამრავლება“ აქვს შემდეგი სახე (ნახ.2.5):



ნახ.2.5

მოცემულ გრაფში ნებისმიერი S_i მდგომარეობა, გარდა კიდურა S_1 და S_n მდგომარეობებისა, დაკავშირებულია ორ მეზობელ მდგომარეობასთან, ე.ი. S_i მდგომარეობიდან შეიძლება გადასვლა მხოლოდ S_{i-1} და S_{i+1} მდგომარეობებში. კიდურა S_1 მდგომარეობიდან გადასვლა შეიძლება მხოლოდ S_2 მდგომარეობაში, ხოლო კიდურა S_n მდგომარეობიდან – მხოლოდ S_{n-1} მდგომარეობაში.

,,გაქრობის და გამრავლების“ პროცესის მდგომარეობათა ალბათობებისათვის კოლმოგოროვის განტოლებათა სისტემა შეიძლება ჩაიწეროს ზემოთ მოყვანილი წესის მიხედვით.

ჩვეულებრივ მმს–ის გამოკვლევა იწყება მოთხოვნათა შემავალი ნაკადის შესწავლით. შემოვიღოთ $X(t)$ ფუნქცია, რომელიც ახასიათებს მოთხოვნათა შემავალ ნაკადს და განსაზღვრავს მოთხოვნათა რაოდენობას 0,t დროის განმავლობაში. დროის ამ შუალედში მმს–ში შეიძლება შევიდეს 1,2 ან ზოგადად K მოთხოვნა, მაგრამ შეიძლება ისეც მოხდეს, რომ საერთოდ არც ერთი მოთხოვნა არ შევიდეს. აქედან გამომდინარეობს, რომ

შემავალი ნაკადი წარმოადგენს შემთხვევით სიდიდეს. ამიტომ $X(t)$ ფუნქციაც, რომელიც ასახავს მოთხოვნათა შესვლის პროცესს მმს–ში, აგრეთვე შემთხვევითი სიდიდეა. ცხადია, იგი ღებულობს მხოლოდ მთელ მნიშვნელობებს.

მოთხოვნათა ნაკადი, რომელიც შედის მმს–ში, ზოგადად შეიძლება განაწილდეს ნებისმიერი კანონის მიხედვით. ამასთან, მასობრივი მომსახურების კლასიკურ თეორიაში ძირითადად გამოიყენება უმარტივესი ნაკადი, რადგანაც ითვლება, რომ ასეთი ნაკადის არსებობის შემთხვევაში მმს მუშაობს უაღრესად მძიმე პირობებში.

ნაკადს ეწოდება უმარტივესი, თუ მას გააჩნია სტაციონარულობის, ორდინალურობისა და მომდევნო შედეგის უქონლობის თვისებები. განვიხილოთ ეს თვისებები.

თუ სისტემაში მოთხოვნათა შესვლის ალბათობა t_i, t_j დროის მომენტში დამოკიდებულია ამ მონაკვეთის სიდიდეზე და არ არის დამოკიდებული დროის ათვლის დასაწყისზე, მაშინ ნაკადის ამ თვისებას ეწოდება სტაციონარულობა. ეს თვისება საშუალებას იძლევა განვსაზღვროთ ნაკადის მახასიათებლები დროის ნებისმიერ მომენტში, ე.ი. თუ ნაკადი გამოკვლეულია რომელიღაც (t_i, t_j) დროის შუალედში, მაშინ საჭირო აღარ არის მისი გამოკვლევა სხვა (t_{i+l}, t_{j+n}) შუალედებში.

თუ მოთხოვნათა რაოდენობა, რომელიც შესულია მმს–ში t დროის მომენტის შემდეგ, არ არის დამოკიდებული t დროის მომენტამდე შესული მოთხოვნების რაოდენობაზე, მაშინ ნაკადის ასეთ თვისებას ეწოდება მომდევნო შედეგის უქონლობა.

თუ t დროის მომენტში მმს–ში შედის არაუმეტეს ერთი მოთხოვნისა ან ორი, ან მეტი მოთხოვნის შესვლის ალბათობა 0–ის ტოლია, მაშინ ამ თვისებას ეწოდება ორდინარულობა.

უმარტივესი ნაკადი განაწილებულია პუასონის კანონით და აქვს შემდეგი სახე:

$$p_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t},$$

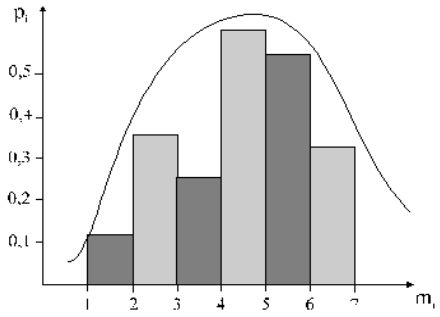
სადაც $p_k(t)$ არის t დროის მომენტში k რაოდენობის მოთხოვნათა შესვლის ალბათობა; k – მოთხოვნათა რაოდენობა; t – მიმდინარე

დრო; λ – ნაკადის პარამეტრია (ნაკადის ინტენსიურობა ან მმს–ში შემავალი მოთხოვნათა რაოდენობაა დროის ერთეულში).

პუასონის ნაკადისათვის, მმს–ში დროის ერთეულში შემოსული მოთხოვნათა რიცხვის მათემატიკური მოლოდინი ან საშუალო მნიშვნელობა $M_k(t)$ ტოლია ნაკადის λ ინტენსიურობისა:

$$M_k(t) = \lambda.$$

მმს–ის პრაქტიკული ამოცანების ამოხსნის დროს, პირველ რიგში აუცილებელია დავადგინოთ, თუ რა კანონით არის განაწილებული მოთხოვნათა შემავალი ნაკადი. ამისათვის საჭიროა ვაწარმოოთ დაკვირვება რეალურ შემავალ ნაკადზე. ამასთან, დაკვირვების მთელი დრო უნდა დავყოთ განსაზღვრულ T_i ინტერვალებად საბოლოო წერტილებით t_i, t_{i+1}, სადაც i = (1, *l*), ხოლო *l* ინტერვალთა საერთო რიცხვია.



ნახ.2.6

ნაკადზე დაკვირვების შედეგად მიღებული სტატისტიკური მონაცემების საფუძველზე განისაზღვრება სტატისტიკური ალბათობები p_i^* (სიხშირეები) და აიგება სიხშირის ჰისტოგრამა (ნახ.2.6).

ამასთან,

$$p_i^* = \frac{m_i}{n},$$

სადაც m_i ინტერვალების რაოდენობაა მოთხოვნათა ერთნაირი რიცხვით; n – დაკვირვებათა საერთო რიცხვი.

დაკვირვების სტატისტიკური მონაცემები საშუალებას იძლევა აგრეთვე განისაზღვროს $M_k(t)$ მათემატიკური მოლოდინი მოთხოვნათა რიცხვისა:

$$M_k(t) = \frac{\sum_{i=0}^{\ell} x_i m_i}{\sum_{i=0}^{\ell} m_i}$$

სადაც x_i მოთხოვნათა რიცხვია i–ურ ინტერვალში.

როგორც აღვნიშნეთ, პუასონის ნაკადისათვის მოთხოვნათა რიცხვის მათემატიკური მოლოდინი დროის ერთეულში ტოლია ნაკადის λ ინტენსიურობისა, ე.ი. $M_k(t) = \lambda$. ამიტომ λ–ს ნაპოვნი მნიშვნელობის საფუძველზე განისაზღვრება P_{m_i} ალბათობების თეორიული მნიშვნელობები, რომლებიც განაწილებულია პუასონის კანონით:

$$p_{m_i} = \frac{\lambda^{m_i}}{m_i!} e^{-\lambda}.$$

ამის შემდეგ, ჰისტოგრამის გამომსახველ გრაფიკზე აიგება P_{m_i} განაწილების თეორიული მრუდი (ნახ.2.6). იმის შესამოწმებლად, რომ შემავალი ნაკადი განაწილებულია პუასონის კანონით, საჭიროა ვისარგებლოთ χ^2 თანხმობის კრიტერიუმით, რომელიც საშუალებას იძლევა შევაფასოთ სტატისტიკური და თეორიული განაწილების შესაბამისობის ხარისხი. შევადგინოთ m_i და $n \cdot P_{m_i}$ რიცხვების შესადარებელი ცხრილი 2.1:

ცხრ.2.1

T_i	t_1, t_2	t_2, t_3	...	t_{i-1}, t_i
m_i	m_1	m_2	...	m_i
Pm_i	Pm_1	Pm_2	...	Pm_i
$n \bullet Pm_i$	$n \bullet Pm_1$	$n \bullet Pm_2$	...	$n \bullet Pm_i$

სტატისტიკურ და თეორიულ განაწილებას შორის χ^2 სხვაობის ზომის განსაზღვრა წარმოებს შემდეგი ფორმულით:

$$\chi^2 = \sum_{i=0}^{l} \frac{(m_i - n \cdot p_{m_i})^2}{n \cdot p_{m_i}},$$

სადაც l თანრიგების რაოდენობაა.

ამის შემდეგ, აუცილებელია განვსაზღვროთ თავისუფლების ხარისხი r = l – s. სადაც s დამყარებული კავშირების რიცხვია (პუასონის კანონისათვის s = 2). χ^2–ისა და r–ის ნაპოვნი მნიშვნელობების მიხედვით, სპეციალური ცხრილით (χ^2–ის მნიშვნელობათა ცხრილი) განისაზღვრება p ალბათობა იმისა, რომ სტატისტიკური განაწილება პუასონისეულია. თუ ეს ალბათობა p ≥ 0,1, მაშინ პუასონისეული განაწილების ჰიპოთეზა სწორია; იმ შემთხვევაში, როცა p < 0,1 – ჰიპოთეზა არასწორია.

მმს–ის ერთ–ერთ ძირითად მახასიათებელს, შემავალ ნაკადთან ერთად, წარმოადგენს **მომსახურების დრო**, რომელიც იხარჯება ერთი მომსახურების ხელსაწყოთი ერთი მოთხოვნის მომსახურებაზე. ამასთან ითვლება, რომ მომსახურების დრო თუ ამოიწურა, მაშინ მოთხოვნა ტოვებს მმს–ს მომსახურებული. მომსახურების დრო წარმოადგენს შემთხვევით სიდიდეს. ჯერ ერთი, მომსახურე ხელსაწყოები არაერთგვაროვანია, მათ გააჩნიათ სხვადასხვა საექსპლუატაციო მახასიათებლები. მეორეც, თვით მოთხოვნებიც არ არის ერთგვაროვანი. მომსახურების დროის აღსაწერად აუცილებელია განაწილების კანონის გამოყენება. აღვნიშნოთ $p(t_{მომს} < t)$ ალბათობა იმისა, რომ მომსახურების დრო $t_{მომს}$ ნაკლებია წინასწარ მოცემულ t დროზე, მაშინ ფუნქცია $F(t) = p(t_{მომს} < t)$ წარმოადგენს მომსახურების დროის განაწილების კანონს. მასობრივი მომსახურების კლასიკურ თეორიაში მიღებულია, რომ მომსახურების დრო განაწილებულია მაჩვენებლიანი კანონით:

$$F(t) = 1 - e^{-\nu t},$$

სადაც ν მომსახურების პარამეტრია. ის განისაზღვრება როგორც

$$\nu = \frac{1}{\bar{t}_{მომს}}, \text{ სადაც}$$

$\bar{t}_{\text{მომს}}$ – მოთხოვნის მომსახურების საშუალო დრო, ანუ ν–ს უკუმნიშვნელობა წარმოადგენს მომსახურების დროის მათემატიკურ მოლოდინს $M(t)$:

$$M(t) = \frac{1}{\nu}.$$

ν – ს ხშირად უწოდებენ მომსახურების ინტენსიურობას.

მმს–ის კლასიფიკაცია ძირითადად ხორციელდება მოთხოვნის ქცევის მიხედვით, რომელიც შედის მმს–ში იმ მომენტში, როდესაც ყველა მომსახურე ხელსაწყო დაკავებულია. თუ ამასთან, მოთხოვნილება არ ელოდება რომელიმე მომსახურების ხელსაწყოს განთავისუფლებას და ტოვებს სისტემას მომსახურების გარეშე, მაშინ ასეთ მმს–ებს ეწოდება **მმს დანაკარგებით** ან **უარის თქმით**. ხოლო ასეთ მოთხოვნებს ხშირად უწოდებენ ,,მოუთმენელ კლიენტებს''.

თუ მოთხოვნების სისტემაში შესვლის დროს ყველა მომსახურების ხელსაწყო დაკავებულია, მაგრამ ამასთან მოთხოვნა არ ტოვებს სისტემას და ელოდება რომელიღაც მომსახურების ხელსაწყოს განთავისუფლებას, მაშინ მმს–ს ეწოდება **მმს რიგებით**, ან მმს **მოლოდინით**, ან მმს **უდანაკარგოდ**.

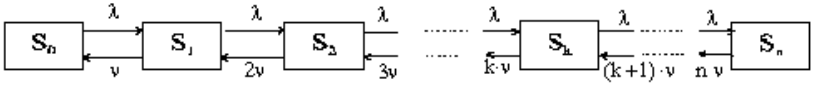
მმს–ის ფუნქციონირების შესაფასებლად შემოღებულია სპეციალური ეფექტურობის კრიტერიუმი. მმს–ის უარის თქმით დახასიათების ერთ–ერთ ძირითად კრიტერიუმად ითვლება მოთხოვნის მომსახურებაზე უარის თქმის ალბათობა. ეს კრიტერიუმი ახასიათებს მმს–ის გამტარუნარიანობას. მმს–ის უარის თქმით ფუნქციონირების შესაფასებლად გამოიყენება აგრეთვე ეფექტურობის ისეთი კრიტერიუმი, როგორიცაა მომსახურების ხელსაწყოთა საშუალო რიცხვი, რომლებიც დაკავებულია მოთხოვნათა მომსახურებით.

მმს–ის ლოდინით ფუნქციონირების შესაფასებლად გამოყენებულია შემდეგი კრიტერიუმები: რიგის სიგრძე, თავისუფალი მომსახურე ხელსაწყოების რიცხვი. რიგის სიგრძე შემთხვევითი სიდიდეა, იგი განისაზღვრება იმ მოთხოვნების საშუალო რიცხვით, რომლებიც ელოდება მომსახურების დაწყებას. მისი მეშვეობით შესაძლებელია შევაფასოთ მოთხოვნათა მოცდენის

ხარისხი. თავისუფალი მომსახურების ხელსაწყოთა რიცხვი ახასიათებს მმს–ის დატვირთვის ხარისხს.

განვიხილოთ მმს დანაკარგებით ან უარის თქმით.

ვთქვათ, მმს–ს გააჩნია n მომსახურე ხელსაწყო. ხშირად ასეთ მმს–ს უწოდებენ n–არხიანს. მოცემული მმს–ის ფუნქციონირება შეიძლება წარმოვადგინოთ შემდეგი მდგომარეობების გრაფის სახით (ნახ.2.7):



ნახ.2.7

S_k მმს–ის მდგომარეობაა, k = (1,n); λ – მოთხოვნათა ნაკადის ინტენსიურობა; ν – მომსახურების ინტენსიურობა; S_0 – მდგომარეობა, რომლის დროსაც ყველა მომსახურების ხელსაწყო თავისუფალია; S_1 – დაკავებულია ერთი მომსახურე ხელსაწყო, დანარჩენი თავისუფალია; S_2 – დაკავებულია 2 მომსახურე ხელსაწყო, დანარჩენი თავისუფალია; S_k – დაკავებულია k მომსახურე ხელსაწყო, დანარჩენი თავისუფალია; S_n – დაკავებულია n მომსახურე ხელსაწყო.

შემოვიღოთ შემდეგი დაშვებები. ერთი მომსახურე ხელსაწყო ერთდროულად ემსახურება მხოლოდ ერთ მოთხოვნას. თუ მმს–ში რიგითი მოთხოვნის შემოსვლის დროს არის ერთი მაინც თავისუფალი მომსახურე ხელსაწყო, მაშინ ის დაუყოვნებლივ იწყებს მის მომსახურებას. მომსახურების დროის დამთავრების შემდეგ მოთხოვნა ტოვებს მმს–ს მომსახურებული. მმს–ში შედის უმარტივესი ნაკადი და მომსახურების დრო განაწილებულია მაჩვენებლიანი კანონით.

თუ სისტემა იმყოფება S_0 მდგომარეობაში, მაშინ მოთხოვნათა ნაკადის შესვლის დროს λ ინტენსიურობით ის გადავა S_1 მდგომარეობაში. S_1 მდგომარეობაში მმს–ზე მოქმედებს ორი ნაკადი: λ მოთხოვნათა ნაკადი, რომელსაც მმს გადაჰყავს S_2 მდგომარეობაში და ν მომსახურების ნაკადი, რომელსაც მმს

გადაჰყავს S_0 მდგომარეობაში. სხვაგვარად რომ ვთქვათ, ერთი მომსახურე ხელსაწყოს განთავისუფლების შემთხვევაში, ე.ი. როდესაც მთავრდება ერთი მოთხოვნის მომსახურება, მმს გადადის S_0 მდგომარეობაში λ ინტენსიურობით. ზოგად შემთხვევაში, მმს გადადის S_k მდგომარეობიდან S_{k+1} მდგომარეობაში λ ინტენსიურობის ნაკადის მოქმედებით, ხოლო S_k მდგომარეობიდან S_{k-1} მდგომარეობაში $\nu \cdot k$ მომსახურების ნაკადის მოქმედებით, რადგანაც ამ შემთხვევაში ყველა k მომსახურე ხელსაწყო თავისუფლდება.

ალბათური მდგომარეობისთვის განტოლების შედგენის წესის თანახმად, შევადგენთ კოლმოგოროვის დიფერენციალურ განტოლებათა სისტემას მოცემული მმს–თვის (ნახ.2.7):

$$\begin{cases} \dfrac{dp_0(t)}{dt} = -\lambda p_0(t) + \nu p_1(t) \\ \dfrac{dp_1(t)}{dt} = -(\lambda + \nu) p_1(t) + \lambda p_0(t) + 2\nu p_2(t) \\ \dfrac{dp_k(t)}{dt} = -(\lambda + k\nu) p_k(t) + \lambda p_{k-1}(t) + (k+1)\nu p_{k+1}(t) \\ \dots \\ \dfrac{dp_n(t)}{dt} = -n\nu p_n(t) + \lambda p_{n-1}(t) \end{cases} \qquad (2.4)$$

სადაც $p_k(t)$ ალბათობაა იმისა, რომ t დროის მომენტში მომსახურებით დაკავებულია k მომსახურების ხელსაწყო.

მიღებული დიფერენციალურ განტოლებათა სისტემა წარმოადგენს ერლანგის სისტემას. მისი ამოხსნა დაკავშირებულია სიძნელეებთან, ამიტომ მასობრივი მომსახურების თეორიაში ეძებენ ზღვრულ ამონახსნს სტაციონარულ რეჟიმში, როცა $t \to \infty$. ამისათვის (2.4) სისტემაში გადავიდეთ ზღვარზე, როცა $t \to \infty$. ამასთან, გამოვიყენოთ მარკოვის ცნობილი თეორემის შედეგი:

$$\lim p_k(t) = p_k.$$

მაშინ (2.4) სისტემის განტოლებათა მარცხენა ნაწილი ტოლი იქნება 0–ის, ხოლო მარჯვენა ნაწილში $p_k(t)$ ალბათობების

ადგილზე დარჩება p_k ალბათობები და მივიღებთ შემდეგ ალგებრულ განტოლებათა სისტემას:

$$\begin{cases} 0 = -\lambda p_0 + \nu p_1 \\ 0 = -(\lambda + \nu) p_1 + \lambda p_0 + 2\nu p_2 \\ 0 = -(\lambda + k\nu) p_k + \lambda p_{k-1}(t) + (k+1)\nu p_{k+1} \\ \cdots \\ 0 = -n\nu p_n + \lambda p_{n-1} \end{cases} \qquad (2.5)$$

მოცემული სისტემა ამოვხსნათ შემდეგნაირად. აღვნიშნოთ $Z_k = \lambda \cdot p_{k-1} - \nu \cdot k \cdot p_k$,, მაშინ (2.5)-ის პირველი განტოლებიდან გამომდინარეობს, რომ

$$Z_1 = \lambda \cdot p_0 - \nu \cdot p_1 = 0.$$

ადვილი შესამოწმებელია, რომ

$$Z_1 = Z_2 = \ldots = Z_k = Z_{k+1} = \ldots = Z_n = 0.$$

თუ გამოვიყენებთ Z_k–ს გამოსახულებას, მივიღებთ:

$$\lambda \cdot p_{k-1} = \nu \cdot k \cdot p_k.$$

ვიპოვოთ p_k:

$$p_k = \frac{\lambda}{\nu \cdot k} p_{k-1}; \quad k = \overline{(1,n)}.$$

დავუშვათ $k = 1$, მაშინ $p_1 = \frac{\lambda}{\nu} p_0$.

როცა $k = 2$, $p_2 = \frac{\lambda}{\nu \cdot 2} \cdot p_1 = \left(\frac{\lambda}{\nu}\right)^2 \cdot \frac{1}{1 \cdot 2} \cdot p_0$.

როცა $k = 3$, $p_3 = \frac{\lambda}{\nu \cdot 3} \cdot p_2 = \left(\frac{\lambda}{\nu}\right)^3 \cdot \frac{1}{1 \cdot 2 \cdot 3} \cdot p_0$.

ზოგად შემთხვევაში:

$$p_k = \left(\frac{\lambda}{\nu}\right)^k \cdot \frac{1}{k!} \cdot p_0 \qquad (2.6)$$

რადგან $p_k(t)$ მდგომარეობათა ალბათობები შეადგენს მოვლენათა სრულ ჯგუფს, ამიტომ

$$\sum_{k=0}^{n} p_k(t) = 1 .$$

მაგრამ, სტაციონარული რეჟიმის შემთხვევაში, როცა $t \to \infty$, შეიძლება ჩაიწეროს:

$$\sum_{m=0}^{n} p_m = 1 \qquad (2.7)$$

(2.7)-ში p_m–ის მაგივრად ჩავსვათ (2.6) გამოსახულება, მივიღებთ:

$$\sum_{m=0}^{n} \left(\frac{\lambda}{\nu}\right)^m \cdot \frac{1}{m!} \cdot p_0 = 1 \text{ , აქედან}$$

$$p_0 = \frac{1}{\sum_{m=0}^{n} \left(\frac{\lambda}{\nu}\right)^m \cdot \frac{1}{m!}} .$$

როცა $k = n$, ყველა მომსახურე ხელსაწყო იქნება დაკავებული და თუ ამ მომენტში მმს–ში შემოვიდა მოთხოვნა, მაშინ ის უარს მიიღებს მომსახურებაზე. თუ გამოვიყენებთ (2.3) და (2.4) გამოსახულებებს, ვიპოვით p_n ალბათობას:

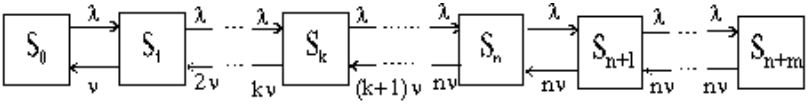
$$p_n = \frac{\left(\frac{\lambda}{\nu}\right)^n \cdot \frac{1}{n!}}{\sum_{m=0}^{n} \left(\frac{\lambda}{\nu}\right)^m \cdot \frac{1}{m!}}$$

მოცემული ალბათობა წარმოადგენს მომსახურებაზე უარის თქმის ალბათობას. ამის შემდეგ განვსაზღვროთ დაკავებულ მომსახურე ხელსაწყოთა რიცხვის M მათემატიკური მოლოდინი:

$$M = \sum_{k=1}^{n} k \cdot p_k = \sum_{k=1}^{n} k \cdot \left(\frac{\lambda}{\nu}\right)^k \frac{1}{k!} p_0 = \sum_{k=1}^{n} \frac{1}{(k-1)!} \cdot \left(\frac{\lambda}{\nu}\right)^k \cdot p_0 .$$

ახლა განვიხილოთ **მმს ლოდინით**.

მმს ლოდინით ანალიზის დროს ვიყენებთ ყველა იმ დაშვებას, რომელიც იყო მიღებული მმს უარის თქმით განხილვის დროს. ვთქვათ, მმს–ს გააჩნია n მომსახურების ხელსაწყო. დავუშვათ, მმს–ში შედის მოთხოვნათა უმარტივესი ნაკადი λ ინტენსიურობით და ერთი მომსახურების ხელსაწყოთი მოთხოვნის მომსახურების ინტენსიურობა ν -ს ტოლია. რიგში მყოფი მოთხოვნების რაოდენობა აღვნიშნოთ m–ით. მოვიყვანოთ მოცემული მმს–ის მდგომარეობათა გრაფი (ნახ.2.8).



ნახ.2.8

მმს–ის მდგომარეობის განსაზღვრის დროს გავითვალისწინოთ რიგის არსებობა ან არარსებობა. რადგანაც სისტემას გააჩნია n მომსახურების ხელსაწყო, რიგი შეიქმნება მხოლოდ მას შემდეგ, რაც დაკავებული იქნება ყველა n ხელსაწყო. განვიხილოთ მმს–ის მდგომარეობები 2.8 ნახაზზე გამოსახული გრაფის შესაბამისად.

S_0 – ყველა ხელსაწყო თავისუფალია; S_1 – დაკავებულია ერთი მომსახურების ხელსაწყო, დანარჩენი თავისუფალია; S_k – დაკავებულია k მომსახურების ხელსაწყო, დანარჩენი თავისუფალია; S_n – დაკავებულია n მომსახურების ხელსაწყო, დანარჩენი თავისუფალია. როცა $n < k \le m$, სისტემაში წარმოიშობა რიგი; მაშინ S_{n+1} – მდგომარეობა, რომლის დროსაც დაკავებულია n მომსახურების ხელსაწყო და ერთი მოთხოვნა დგას რიგში; S_{n+m} – დაკავებულია n მომსახურების ხელსაწყო და m მოთხოვნა დგას რიგში.

მოცემულ სისტემაში ერთი მდგომარეობიდან მეორეში გადასვლა ხორციელდება ისევე, როგორც მმს უარის თქმით, მაგრამ შემდეგი ფაქტორის გათვალისწინებით.

გადასვლა S_{n+i} მდგომარეობიდან S_{n+i-1} -ში, სადაც $i = (1,m)$ გამოსახავს n მომსახურების ხელსაწყოებიდან ერთ–ერთის განთავისუფლებას. ე.ი. მმს-ში რიგის არსებობის შემთხვევაში მომსახურების ნაკადს ექნება $n \cdot \nu$ ინტენსიურობა.

მმს ლოდინის შემთხვევაში კოლმოგოროვის განტოლებათა სისტემას ექნება ისეთივე სახე, როგორც მმს უარის თქმის დროს. ამიტომ, თუ არ გავიმეორებთ წინა გამოთვლებს, შეგვიძლია განვსაზღვროთ ზღვრული ამონახსნები, ე.ი. S_k მდგომარეობის ზღვრული ალბათობები:

$$p_1 = \frac{\lambda}{\nu} \cdot \frac{1}{1!} \cdot p_0, \quad p_2 = \left(\frac{\lambda}{\nu}\right)^2 \cdot \frac{1}{2!} \cdot p_0, \ldots, p_k = \left(\frac{\lambda}{\nu}\right)^k \cdot \frac{1}{k!} \cdot p_0, , \ldots, p_n =$$

$$= \left(\frac{\lambda}{\nu}\right)^n \cdot \frac{1}{n!} \cdot p_0 \quad , \text{სადაც } (0 \le k \le n).$$

ეს გამოსახულებები მართებულია იმ შემთხვევისათვის, როცა მმს–ში არ არის რიგი. რიგის წარმოშობის შემთხვევაში შესაბამისი მდგომარეობის ალბათობები შეიძლება განისაზღვროს შემდეგნაირად.

ვისარგებლოთ

$$p_k = \frac{\lambda}{\nu \cdot k} \cdot p_{k-1}$$

დამოკიდებულებით.

აქედან

$$p_{n+1} = \frac{\lambda}{\nu \cdot n} \cdot p_n$$

რადგანაც განთავისუფლების ნაკადის ინტენსიურობა ტოლია $\nu \cdot n$ (მომსახურების ხელსაწყოების რაოდენობა არ აჭარბებს n–ს), მაშინ:

$$p_{n+1} = \frac{\lambda}{\nu \cdot n} \left(\frac{\lambda}{\nu}\right)^n \cdot \frac{1}{n!} p_0 = \left(\frac{\lambda}{\nu}\right)^{n+1} \cdot \frac{1}{n \cdot n!} \cdot p_0 .$$

ანალოგიურად

$$p_{n+2} = \left(\frac{\lambda}{\nu}\right)^{n+2} \cdot \frac{1}{n^2 \cdot n!} \cdot p_0, \ldots, p_{n+m} = \left(\frac{\lambda}{\nu}\right)^{n+m} \cdot \frac{1}{n^m \cdot n!} \cdot p_0 .$$

მდგომარეობის p_n ალბათობები შეადგენს მოვლენათა სრულ ჯგუფს, ამიტომ

$$\sum_{k=0}^{m} p_k = 1 .$$

წარმოვადგინოთ მოცემული ჯამი შემდეგნაირად:

$$\sum_{k=0}^{n} p_k + \sum_{k=n+1}^{m} p_k = 1 . \qquad (2.8)$$

(2.8) გამოსახულებაში პირველი ჯამი შედგენილია მდგომარეობათა ალბათობებისაგან იმ შემთხვევისთვის, როდესაც არ არის რიგი, ხოლო მეორე – როდესაც რიგი არის. წარმოვადგინოთ (2.8) გამოსახულება შემდეგნაირად:

$$\sum_{k=0}^{n} \left(\frac{\lambda}{\nu}\right)^k \cdot \frac{1}{k!} \cdot p_0 + \sum_{k=n+1}^{m} \frac{1}{n^k \cdot n!} \left(\frac{\lambda}{\nu}\right)^k \cdot p_0 = 1$$

და ვიპოვოთ p_0:

$$p_0 = \frac{1}{\sum_{k=0}^{n} \left(\frac{\lambda}{\nu}\right)^k \cdot \frac{1}{k!} + \sum_{k=n+1}^{m} \frac{1}{n^k \cdot n!} \left(\frac{\lambda}{\nu}\right)^k}$$

თუ გვეცოდინება მდგომარეობის ალბათობები, შეგვიძლია განვსაზღვროთ მმს–ის ლოდინით ფუნქციონირების შეფასების კრიტერიუმები. რადგანაც რიგის სიგრძე შემთხვევითი სიდიდეა, ამიტომ განვსაზღვროთ მისი M_1 მათემატიკური მოლოდინი:

$$M_1 = \sum_{k=n+1}^{m} (k-n) \cdot p_k = \sum_{k=n+1}^{m} \frac{(k-n)}{n^k \cdot n!} \left(\frac{\lambda}{\nu}\right)^k \cdot p_0 .$$

თუ გვაქვს M_1, შეგვიძლია K_1 კოეფიციენტის მეშვეობით შევაფასოთ მოთხოვნის მოცდენის სიდიდე რიგში:

$$K_1 = \frac{M_1}{m} = \frac{1}{m}\sum_{k=n+1}^{m}(k-n)\cdot p_k = \sum_{k=n+1}^{m}\frac{(k-n)}{n^k\cdot n!}\left(\frac{\lambda}{\nu}\right)^k\cdot p_0,$$

სადაც K_1 მოთხოვნის მოცდენის კოეფიციენტია.

ასევე შემთხვევითი სიდიდეა თავისუფალი მომსახურების ხელსაწყოების რიცხვი, ამიტომ განვსაზღვროთ მისი M_2 მათემატიკური მოლოდინი

$$M_2 = \sum_{k=0}^{n-1}(n-k)\cdot p_k = \sum_{k=0}^{n}\frac{(n-k)}{k!}\left(\frac{\lambda}{\nu}\right)^k\cdot p_0 .$$

M –ის საფუძველზე, K_2 კოეფიციენტის მეშვეობით შეგვიძლია შევაფასოთ მომსახურების ხელსაწყოს მოცდენის სიდიდე:

$$K_2 = \frac{M_2}{n} = \sum_{k=0}^{n-1}\frac{(n-k)}{n}\cdot p_k = \sum_{k=0}^{n-1}p_k - \sum_{k=0}^{n-1}k\cdot p_k =$$

$$= \sum_{k=0}^{n-1}\left(\frac{\lambda}{\nu}\right)^k\cdot\frac{1}{k!}\cdot p_0 - \frac{1}{n}\sum_{k=0}^{n-1}\left(\frac{\lambda}{\nu}\right)^k\cdot\frac{1}{(k-1)!}\cdot p_0 .$$

ახლა განვიხილოთ ერთი მაგალითი კონკრეტული ამოცანის გადასაწყვეტად.

„საზღვაო პორტს აქვს სამი ნავმისადგომი. დატვირთული ტანკერების საშუალო რაოდენობა, რომელიც შემოდის პორტში 1 თვის განმავლობაში არის 20-ის ტოლი. ერთი ტანკერის მომსახურებას სჭირდება საშუალოდ 6 დღე. საჭიროა შეფასდეს პორტის ფუნქციონირების მაჩვენებლები“.

მოცემული ამოცანა მიეკუთვნება სისტემას **ლოდინით**. განვსაზღვროთ მისი ძირითადი მახასიათებლები:

ალბათობა იმისა, რომ ყველა ნავმისადგომი იქნება თავისუფალი, ანუ ნავსადგურში არ შემოვა არც ერთი გემი:

$$p_0 = \frac{1}{\sum_{k=1}^{n}\left(\frac{\lambda}{\nu}\right)^k \cdot \frac{1}{k!} + \sum_{k=n+1}^{m} \frac{1}{n^k \cdot n!} \cdot \left(\frac{\lambda}{\nu}\right)^k}.$$

სადაც $\nu = \frac{1}{t}$ და რადგანაც $\bar{t}=0,2$ თვეს, $\nu=5$.

$$p_0 = \frac{1}{\left(\frac{15}{5}\right)^0 \cdot \frac{1}{0!} + \left(\frac{15}{5}\right)^1 \cdot \frac{1}{1!} + \left(\frac{15}{5}\right)^2 \cdot \frac{1}{2!} + \left(\frac{15}{5}\right)^3 \cdot \frac{1}{3!} + \frac{1}{3^4 \cdot 3!}\left(\frac{15}{5}\right)^4 + \frac{1}{3^5 \cdot 3!}\left(\frac{15}{5}\right)^5 + \frac{1}{3^6 \cdot 3!}\left(\frac{15}{5}\right)^6} = 0,1.$$

განვსაზღვროთ რიგში მყოფი გემების რაოდენობის მათემატიკური მოლოდინი:

$$M_1 = \sum_{k=n+1}^{m} \frac{(k-n)}{n^k \cdot n!}\left(\frac{\lambda}{\nu}\right)^k \cdot p^0 = \left[\frac{1}{3^4 \cdot 3!}\left(\frac{15}{5}\right)^4 + \frac{1}{3^5 \cdot 3!}\left(\frac{15}{5}\right)^5 + \frac{1}{3^6 \cdot 3!}\left(\frac{15}{5}\right)^6\right] \cdot 0,01 =$$

$$= 0,1 \cdot 25,44 = 2,6.$$

გემების რიგში მოცდენის სიდიდე იქნება:

$$K_1 = \frac{M_1}{m} = \frac{2,6}{17} = 0,15.$$

ეს ნიშნავს, რომ თითოეული გემი თავის დროის 15% მოცდენილი იქნება. ახლა განვიხილოთ თავისუფალი (მოცდენილი) ნავმისადგომების საშუალო რაოდენობა:

$$M_2 = \sum_{k=0}^{n-1} \frac{(n-k)}{k!} \cdot \left(\frac{\lambda}{\nu}\right)^k \cdot p_0 = \left[\frac{2}{0!}\left(\frac{15}{5}\right)^0 + \frac{1}{1!}\left(\frac{15}{5}\right)^1\right] \cdot 0,1 = 0,5.$$

ხოლო ნავმისადგომის მოცდენის სიდიდე

$$K_2 = \frac{M_2}{n} = \frac{0,5}{3} = 0,16.$$

ამრიგად, თითოეული ნავსადგომი მოცდენილი იქნება თავის დროის 16%.

2.2. რიგების სახეები მასობრივ მომსახურების სისტემებში

მასობრივ მომსახურების სისტემებში შემავალ და გამომავალ მოთხოვნათა ნაკადის ინტენსივობის გამოსათვლელი მეთოდი არის ერთ-ერთი ძირითადი საკითხი. როგორც ცნობილია, აქ მისაღებია ინგლისელი მათემატიკოსის, დავიდ კენდალის (1918-2007) ნოტაცია რიგების თეორიის სტანდარტიზაციის და კლასიფიკაციის საკითხებზე (56). რიგების აღწერის გაფართოებული მოდელი მოიცემა ექვსეულით:

< A/S/c/K/N/D >, სადაც

A რიგში მოთხოვნების შემოსვლის სტატისტიკური განაწილებაა (M, თუ პროცესი მარკოვულია); S - რიგში მოთხოვნის მომსახურების განაწილება (M-მარკოვული ან ექსპონენციალური, E-ერლანგის განაწილება, G-საერთო განაწილება და ა.შ.); c – იდენტური მომსახურე ობიექტების რაოდენობა ($c \geq 1$); K-კლიენტების max-რაოდენობაა, რომელთა მომსახურება ხდება (თუ K-ზე მეტია, მაშინ კლიენტი არ იცდის რიგში. თუ K არაა მოცემული, მაშინ კლიენტების რაოდენობაზე არაა შეზღუდვა, უსასრულოა); N-კლიენტთა max-რაოდენობა, რომელიც შეიძლება მოვიდეს სისტემაში (თუ არაა მოცემული, მაშინ ∞). D–მომსახურების დისციპლინა (FIFO, LIFO, SIRO (Service In Random Order), PNPN (Priority service) ან PS (Processor Sharing)) [56].

გამოკვლევების შედეგად დადგინდა, რომ მარკოვის პროცესები თამაშობს ფუნდამენტურ როლს მასობრივი მომსახურების სისტემების კვლევისას. მარკოვის პროცესების ძირითადი შედეგების ანალიზზე დაყრდნობით გაირკვა, რომ მისი ერთ-ერთი კერძო შემთხვევაა გამრავლებისა და გაქრობის პროცესი. მომდევნო პარაგრაფებში აღიწერება რიგების სახეები, რომლებიც გვხვდება მასობრივი მომსახურების სისტემებში.

2.2.1. M/M/1 სისტემა

სისტემის სტაციონალურ რეჟიმში მუშაობის ერთ-ერთი მნიშვნელოვანი პირობაა შემავალი და გამავალი ნაკადი იყოს თანაბარი, ამ პრინციპის გათვალისწინებით არსებობს კლასიკური M/M/1 სახის სისტემა, რომელსაც გამრავლების და გაქრობის სიტემას უწოდებენ.

გამრავლებისა და გაქრობის პრცესს აქვს ერთი მეტად მნიშვნელოვანი თვისება: დროის ის შუალედი, რომელ მომენტშიც ხდება გამრავლება და დროის შუალედში, რომელ მომენტშიც ხდება გაქრობა (როდესაც სისტემა არ არის თავისუფალი და ვერ ღებულობა მოთხოვნებს) აღიწერება განაწილების მაჩვენებლიანი კანონით (ეს მიუთითებს იმას, რომ პროცესი არის მარკოვული). ეს პროცესი ზოგადად შეიძლება ჩამოვაყალიბოთ შემდეგნაირად: M/M/1 სახის სისტემა არის ისეთი სისტემა, სადაც დროის შუალედი მეზობელ მოთხოვნათა შორის განაწილებულია მაჩვენებლიანი კანონით, აგრეთვე მომსახურების დროც განაწილებულია მაჩვენებლიანი კანონით და სისტემა შეიცავს მხოლოდ ერთ მომსახურე მოწყობილობას.

თუ ავიღებთ კოეფიციენტებისთვის შემდეგ მნიშვნელობებს:

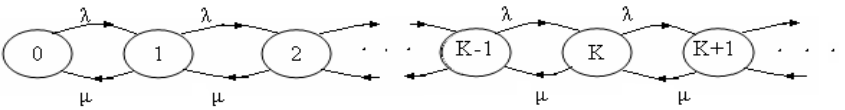
$\lambda_k = \lambda$, k=0,1,2, . . .

$\mu_k = \mu$, k=1,2, 3, . . .

ეს მიუთითებს, რომ გამრავლების ყველა ინტენსივობა λ არის მუდმივი და თანაბარი და აგრეთვე გაქრობის ყველა ინყენსივობაც μ არის მუდმივი და თანაბარი. (ამ შემთხვევაში ორ მეზობელ მოთხოვნას შორის შუალედის საშუალო სიგრძე ტოლია: $\bar{t} = 1/\lambda$ და მომსახურების საშუალო დრო კი ტოლია $\bar{x} = 1/\mu$; ეს განპირობებულია იმით, რომ ორივე შემთხვევითი სიდიდე $\bar{t}$ და $\bar{x}$ განაწილებულია მაჩვენებლიანი კანონით. აღსანიშნავია ის

გარემოება, რომ სისტემის მდგომარეობის სივრცე არის უსასრულო და რომ მოთხოვნათა მომსახურება ხორციელდება მათი დადგომის თანამიმდევრობით.

გადასასვლელთა ინტენსივობის დიაგრამა მოცემულია M/M/1 ტიპისთვის (ნახ.2.9).



ნახ.2.9 გადასასვლელთა ინტენსივობის დიაგრამა M/M/1 ტიპისთვის

ალბათობა იმისა, რომ სისტემაში დაყენებული ყველა მოთხოვნა დადგება მომსახურებაზე ტოლია:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{\mu} \qquad \text{ან} \qquad p_k = p_0 \left(\frac{\lambda}{\mu}\right)^k, \qquad k \geq 0. \quad . \quad . \quad . \quad (2.9)$$

განხილული სისტემისათვის ერგოდიულობის პირობა მდგომარეობს იმაში, რომ $S_1 < \infty$ და $S_2 = \infty$; პირველი პირობა მოცემული შემთხვევისათვის ჩაიწერება შემდეგი სახით:

$$S_1 = \sum_{k=0}^{\infty} \frac{p_k}{p_0} = \sum_{k=0}^{\infty} \left(\frac{\lambda}{\mu}\right)^k < \infty.$$

ტოლობის მარცხენა მხარე სრულდება მხოლოდ მაშინ, როცა $\lambda / \mu < 1$. ერგოდიულობის მეორე პირობა ღებულობს შემდეგ სახეს:

$$S_2 = \sum_{k=0}^{\infty} \frac{1}{\lambda (p_k / p_0)} = \sum_{k=0}^{\infty} \frac{1}{\lambda} \left(\frac{\mu}{\lambda}\right)^k = \infty.$$

ეს უკანასკნელი პირობა სრულდება მაშინ როდესაც $\lambda / \mu < 1$. ე. ი. აუცილებელი და საკმარისი პირობა იმისა, რომ M/M/1 სისტემა არის ერგოდიული არის ის, რომ უნდა სრულდებოდეს უტოლობა: $\lambda < \mu$.

იმისათვის, რომ ვიპოვოთ p_0 უნდა გამოვიყენოთ ფორმულა:

$$p_0 = 1/\left[1+\sum_{k=1}^{\infty}\left(\frac{\lambda}{\mu}\right)^k\right].,$$ რადგან $\lambda < \mu$ მოვიღებთ:

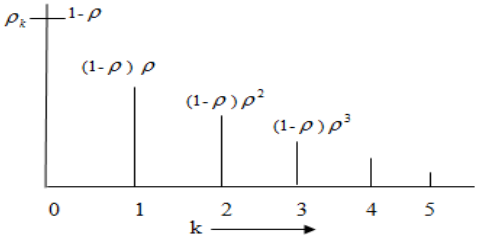
$$p_0 = \frac{1}{1+\frac{\lambda/\mu}{1-\lambda/\mu}} = 1-\frac{\lambda}{\mu} \quad . \quad . \quad . \qquad (2.10)$$

როგორც ვიცით ρ=λ/μ. სტაციონალურობის პირობის თანახმად უნდა სრულდებოდეს უტოლობა: 0≤ρ<1; ეს პირობა გვაძლევს იმის გარანტიას, რომ შესრულდეს უტოლობა $p_0 > 0$; განტოლება (2.9) - ის თანახმად საბოლოოდ მივიღებთ:

$$p_k = (1-\rho)\rho^k, k = 0,1,2,... \quad . \quad . \quad . \qquad (2.11)$$

(2.11) ტოლობის თანახმად ალბათობა იმისა, რომ სისტემა შეიცავს k მოთხოვნას. მნიშვნელოვანია აღვნიშნოთ, რომ p_k ალბათობა განისაზღვრება λ და μ-ს ρ -სთან დამოკიდებულებაზე.

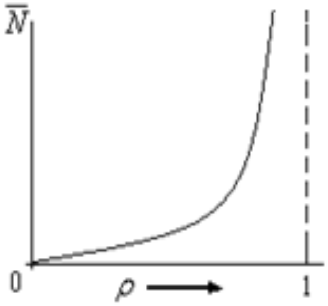
2.10 ნახაზზე ნაჩვენებია განხილული სისტემისათვის p_k ალბათობის მნიშვნელობა, იმ შემთხვევაში როცა $\rho = 1/2$. როგორც ნახაზიდან ჩანს ეს განაწილება არის გეომეტრიული.



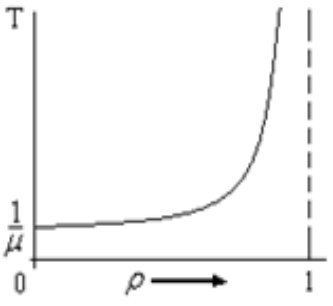
ნახ.2.10 სტაციონალური p_k ალბათობა მმს-ის M/M/1 ტიპისთვის

M/M/1 სისტემა კვლევისას დავინახავთ, რომ სისტემაში არსებული თითქმის ყველა მნიშვნელოვანი განაწილების ალბათობა განეკუთვნება უშემდეგმოქმედო განაწლების ტიპს. მასობრივი მომსახურების სისტემებში მნიშვნელოვანია მოთხოვნათა საშუალო $\overline{N}$ რიცხვი. ეს სიდიდე მოცემულია ტოლობით:

$$\overline{N} = \sum_{k=0}^{\infty} k p_k = (1-\rho)\sum_{k=0}^{\infty} k\rho^k = \frac{\rho}{1-\rho} \qquad (2.12)$$



ნახ.2.11 M/M/1- ტიპის სისტემაში მოთხოვნათა საშუალო რიცხვი



ნახ.2.12. M/M/1- ტიპის სისტემაში მოთხოვნათა დაყენების საშუალო დრო, როგორც ρ-ს ფუნქცია

სისტემაში მოთხოვნათა საშუალო რიცხვის გრაფიკი მოცემულია 2.11 ნახაზზე, ანალოგიური მეთოდით ვპოულობთ, რომ სისტემაში მოთხოვნათა რიცხვის დისპერსია ტოლია:

$$\sigma_N^2 = \sum_{k=0}^{\infty} (k - \overline{N})^2 p_k;$$

$$\sigma_N^2 = \frac{\rho}{(1-\rho)^2}. \qquad . \quad . \quad . \qquad (2.13)$$

(2.13) არის ლიტლის ფორმულა, საიდანაც შესაძლებელია მივიღოთ სისტემაში მოთხოვნათა შემოსვლის საშუალო დრო:

$$T = \frac{\overline{N}}{\lambda}$$

$$T = \left(\frac{\rho}{1-\rho}\right)\left(\frac{1}{\lambda}\right); \qquad . \quad . \quad . \qquad (2.14)$$

$$T = \frac{1/\mu}{1-\rho}.$$

მოთხოვნათა სისტემაში შემოსვლის საშუალო დროის ρ-ზე დამოკიდებულების გრაფიკი ნაჩვენებია 2.12 ნახაზზე. T-სიდიდე, რომელიც შეესაბამება წერტილს $\rho=0$, ტოლია მოთხოვნის მომსახურების საშუალო მნიშვნელობის; სხვა სიტყვებით, რომ ვთქვათ, ამ შემთხვევაში მოთხოვნა არ ელოდება რიგში და კმაყოფილდება საშუალოდ 1/μ-წამში.

როდესაც სისტემაში ρ მიისწრაფის ერთისკენ, როგორც მოთხოვნათა საშუალო რიცხვი, ასევე მოთხოვნათა შემოსვლის საშუალო დრო უსასრულოდ იზრდება.

ინტერესს მოკლებული არ არის გამოვთვალოთ ალბათობა იმისა, როდესაც სისტემა შეიცავს k-ზე ნაკლებ მოთხოვნებს:

$$P[\ \geq k \text{ მოთხოვნა სისტემაში}] = \sum_{i=k}^{\infty} p_i = \sum_{i=k}^{\infty} (1-\rho)\rho_1^i = \rho^k . \quad (2.15)$$

ამ თვალსაზრისით ალბათობა იმისა, რომ სისტემაში დაყენებული მოთხოვნათა რიცხვი გადააჭარბებს ზღვრულ მნიშვნელობას, აღიწერება გაქრობის გეომეტრიული პროგრესიით, რომელიც დამოკიდებულია ამ ზღვრულ რიცხვზე და მიისწრაფვის ნულისკენ.

2.2.2. M/M/m სისტემა: m მომსახურე მოწყობილობით

1900-იანი წლების დასაწყისში დანიელი მეცნიერი აგნერ ერლანგი, როგორც მასობრივი მომსახურების სისტემების ფუძემდებელი, განიხილავდა მას, როგორც სატელეფონო ქსელის მუშაობის ერთ-ერთ მოდელს.

იგი აღნიშნავდა, რომ სისტემაში შემოსული მოთხოვნები უნდა დადგეს რიგში და დაელოდოს მომსახურებას. ერლანგი განიხილავდა აგრეთვე სატელეფონო სისტემის ისეთ მოდელსაც, როგორიცაა მაგალითად, მმს-ის M/M/m ტიპი. ეს ისეთი შემთხვევაა, როდესაც არ ხდება ლოდინი. თუ სისტემაში შემოვა მოთხოვნა და ამ დროს სისტემის მომსახურე მოწყობილობა დაკავებულია, მაშინ მოთხოვნა იკარგება (სისტემა დანაკარგებით).

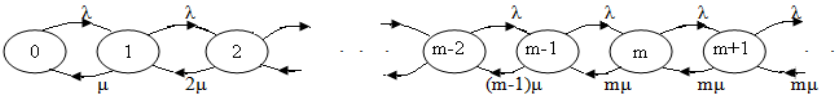
განვიხილოთ M/M/m სისტემა, რომელსაც აქვს მოთხოვნათა განუსაზღვრელი რაოდენობის მიღების საშუალება და დაყენებულ მოთხოვნათა მუდმივი ინტენსივობა. ჩავთვალოთ, რომ სისტემაში შეიძლება გამოყენებულ იქნას მაქსიმუმ m მომსახურე მოწყობილობა. ამ პირობის ფორმულირება შესაძლებელია გამრავლებისა და გაქრობის პროცესის დახმარებით:

$$\lambda_k = \lambda, \qquad k=0,1,2,\ldots;$$

$$\mu_k = \min[k\mu, m\mu] = \left\{ \begin{matrix} k\mu, 0 \le k \le m; \\ m\mu, m \le k. \end{matrix} \right\} \quad \ldots \qquad (2.16)$$

$$\frac{\lambda_k}{\mu_{k+1}} < C < 1 \quad \ldots \qquad (2.17)$$

2.17 უტოლობიდან გამომდინარე იოლად დავრწმუნდებით, რომ მოცემულ შემთხვევაში ერგოდიულობის პირობას აქვს შემდეგი სახე: $\lambda / \mu < 1$; გადასასვლელების ინტენსივობის დიაგრამა ამ პროცესისთვის წარმოდგენილია 2.13 ნახაზზე.



ნახ.2.13. მმს-ისთვის გადასასვლელების ინტენსივობის დიაგრამა M/M/m ტიპი

მოცემული დიაგრამა ასახავს მომსახურე მოწყობილობასთან წარმოქმნილი რიგის შემთხვევაში მოთხოვნა როგორ გადადის უახლოეს მომსახურე ხელსაწყოში.

ალბათობა იმისა, რომ სისტემაში შემოსული k მოთხოვნა დადგება თუ არა მომსახურებაზე, ტოლია:

$$P_k = P_0 \prod_{i=0}^{k=o} \frac{\lambda_i}{\mu_{i+1}}, \qquad k=0,1,2, \quad \ldots \qquad (2.18)$$

ზემოთ მოყვანილი ტოლობის საშუალებით შეგვიძლია განვსაზღვროთ P_k ალბათობის მნიშვნელობა, ხოლო თუ მას გავყოფთ ორ ნაწილად, როგორც დამოკიდებულებას μ_k-სი k-ზე, სადაც შესაბამისად k≤m-ზე, მივიღებთ შემდეგ ტოლობას:

$$P_k = P_0 \prod_{i=0}^{k=o} \frac{\lambda}{(i+1)\mu} = P_0 \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}., \quad \ldots \quad (2.19)$$

ანალოგოურად მოვიღებთ, როცა k≥m,

$$P_k = P_0 \prod_{i=0}^{k=o} \frac{\lambda}{(i+1)\mu} \prod_{j=m}^{k-1} \frac{\lambda}{m\mu} = P_0 \left(\frac{\lambda}{\mu}\right)^k \frac{1}{m! m^{k-m}}, \ldots \quad (2.20)$$

თუ გავაერთიანებთ (4.11) და (4.12) ტოლობებს, მივიღებთ:

$$P_k = \begin{cases} P_0 \dfrac{(m\rho)^k}{k!}, k \le m; \\ P_0 \dfrac{(\rho)^k m^m}{m!}, k \ge m; \end{cases} \quad \text{სადაც} \quad \rho = \frac{\lambda}{m\mu} < 1. \quad \ldots \quad (2.21)$$

ρ გამოსახავს დაკავებული მოწყობილობების ლოდინის კოეფიციენტს. λ არის შემოსული ნაკადის ინტენსივობა, ხოლო $1/\mu$ მომსახურების საშუალო დრო. ე.ი. 2.21 განტოლება გამოსახავს სტაციონალურ ალბათობას იმისას, რომ სისტემაში შემოსული k მოთხოვნა დადგება თუ არა მომსახურებაზე.

P_0–ის განსაზღვრისათვის დავწეროთ შემდეგი ტოლობა:

$$P_0 = \left[1 + \sum_{k=1}^{m-1} \frac{(m\rho)^k}{k!} + \sum_{k=m}^{\infty} \frac{(m\rho)^k}{m!} * \frac{1}{m^{(k-m)}}\right]^{-1},$$

მაშასადამე

$$P_0 = \left[1 + \sum_{k=1}^{m-1} \frac{(m\rho)^k}{k!} + \left(\frac{(m\rho)^k}{m!}\right)\left(\frac{1}{1-\rho}\right)\right]^{-1}. \quad . \quad (2.22)$$

ალბათობა იმისა, რომ დაყენებული მოთხოვნა ჩადგება რიგში მოცემულია ტოლობით:

$$P(\text{ლოდინის}) = \sum_{k=m}^{\infty} p_k = \sum_{k=m}^{\infty} p_0 \frac{(m\rho)^k}{m!} * \frac{1}{m^{k-m}}.$$

ამ თვალსაზრისით,

$$P(\text{ლოდინისა}) = \frac{\left(\frac{(m\rho)^m}{m!}\right)*\left(\frac{1}{1-\rho}\right)}{\left[\sum_{k=0}^{m-1}\frac{(m\rho)^k}{K!}+\left(\frac{(m\rho)}{m!}\right)*\left(\frac{1}{1-\rho}\right)\right]} \qquad (2.23)$$

ეს ფორმულა ფართოდ გამოიყენება სატელეფონო სისტემებში; იგი განსაზღვრავს ალბათობას იმისას, რომ m ხაზის მიმართ დაყენებული მოთხოვნათა ნაკადი ვერ დაიკავებს ვერცერთ თავისუფალ ადგილს, ანუ მომსახურე მოწყობილობას, რის შედეგადაც მოხდება მოთხოვნის დაკარგვა.

ე.ი. ალბათობა იმისა რომ მოთხოვნა სისტემაში ვერ დაიკავებს ვერცერთ თავისუფალ ადგილს, გამოითვლება (4.15) ფორმულით. განხილულ ფორმულას უწოდებენ C–ერლანგის ფორმულას, ხოლო შესაბამისი ალბათობა აღინიშნება $C(m, \lambda/\mu)$. ევროპაში მას შემდეგნაირად აღნიშნავენ $E_{2,m}(\lambda/\mu)$.

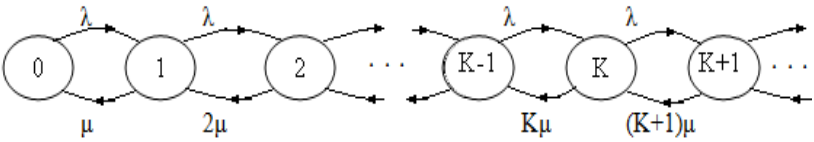
2.2.3. M/M/∞ სისტემა – დაუყოვნებლად მომსახურება (მოწყობილობათა უსასრულო რაოდენობა)

განვიხილოთ შემთხვევა, რომლის ინტერპრეტაცია შეიძლება მოვახდინოთ დაუყოვნებლად მომსახურე მოწყობილობასთან, სადაც მომსახურების ინტენსივობა იზრდება წრფივად იმ რიცხვის ზომამდე, რომელიც შეესაბამება მომლოდინე მოთხოვნას. ე.ი. ესაა სისტემა, რომელშიც ყოველთვის მოიძებნება ახალი მომსახურე მოწყობილობა, რომელიც დააკმაყოფილებს ყოველ ახალ შემოსულ მოთხოვნას.

$$\lambda_k = \lambda, \qquad k=0,1,2, \ . \ . \ .$$

$$\mu_k = k\mu, \qquad k=1,2,3, \ . \ . \ .$$

ამ შემთხვევისათვის გადასასვლელების ინტენსივობის დიაგრამას აქვს 2.14 ნახაზზე გამოსახული სახე:



ნახ.2.14 გადასვლელების ინტენსივობის დიაგრამა M/M/∞-ტიპის მმს-ისთვის

ხოლო ალბათობა იმისა, რომ სისტემაში შემოსული ყველა მოთხოვნა დაკმაყოფილდება, გამოისახება ტოლობით:

$$P_k = P_0 \prod_{i=0}^{k-1} \frac{\lambda}{(i+1)\mu}. \quad \ldots \qquad (2.24)$$

ამ შემთხვევაში ერგოდიულობის პირობა მოცემულია მარტივი უტოლობის სახით:

$$\lambda / \mu < \infty.$$

ამით შეიძლება ვაჩვენოთ, რომ სისტემა, რომელიც გადავსებულია შემოსულ მოთხოვნათა ნაკადით, არის ისეთივე სისტემა, როგორიც დაუყოვნებლად მომსახურე სისტემა.

მასაჩუსეტის უნივერსიტეტის პროფესორის, ჯონ ლიტლის ფორმულის (*Little's law*) საშუალებით შესაძლებელია გამოვსახოთ სისტემაში შემოსული მოთხოვნის დაყოვნების T-საშუალო დრო:

$$T = \frac{\alpha}{\mu^2 (1 - e^{-\alpha/\mu})}; \qquad T = 1/\mu. \quad \ldots \qquad (2.25)$$

ეს პირობა ძალაშია იმ შემთხვევაში, როდესაც სისტემაში შემოსული ყოველი მოთხოვნისათვის გარანტირებულია მისთვის განკუთვნილი ერთი მომსახურე მოწყობილობა მაინც.

ამ შემთხვევაში სისტემაში შემოსული მოთხოვნის დაყოვნების დრო ტოლია ამ მოთხოვნის მომსახურების დროის, რომელიც საშუალოდ $1/\mu$ წამის ტოლია.

2.2.3. M/G/1 სისტემა

M/G/1-სისტემა აღწერს მმს-ის ისეთ პროცესს, სადაც შუალედი მოთხოვნის მოწოდების მომენტსა და მომსახურებას შორის განაწილებულია არამაჩვენებლიანი კანონით. ასეთი სახის სისტემაში ჩვენ არ შეგვიძლია განვსაზღვროთ შემოსულ მოთხოვნათა რაოდენობა ან სისტემაში მოთხოვნის დაყენების რეალური დრო.

M/G/1-სისტემა ხასიათდება ისეთი პროცესის ინტენსივობით, როგორიცაა პუასონის ინტენსივობა, λ მოთხოვნით წამში და საერთო მომსახურების განაწილების ხანგრძლივობით B(x), აგრეთვე $\overline{X}$ წამში მომსახურების საშუალო დროთი და K-M მომენტით, რომელიც ტოლია $\overline{X}^k$.

პუსონის განაწილების გათვალისწინებითა და აგრეთვე იმის გათვალისწინებით, რომ სისტემაში მოთხოვნათა რაოდენობა არ იცვლება ერთზე მეტად, სამართლიანია ტოლობა: $p_k = r_k = d_k$.

ძირითადი ტოლობა, რომელიც აღწერს ურთიერთკავშირს შემთხვევით სიდიდეებს შორის მმს-ის M/G/1-ტიპის სისტემებში, როდესაც სისტემაში დაყენებული მოთხოვნის მომსახურება ხდება მოთხოვნისთანავე, გამოითვლება შემდეგნაირად:

$$q_{n+1} = \begin{cases} q_n - 1 + \nu_{n+1} \\ \nu_{n+1} > 0 \end{cases}, \text{ სადაც}$$

$$q_n > 0, \quad \ldots \qquad (2.26)$$

ამ შემთხვევაში q_n მოთხოვნათა რაოდენობაა, რომელიც რჩება სისტემაში C_n მოთხოვნის დაკმაყოფილების შემთხვევაში, ხოლო ν_n მოთხოვნათა რაოდენობაა, რომელიც მოხვდება სისტემაში X_n მომსახურების დროის მანძილზე. თანამიმდევრობა $\{q_n\}$ ქმნის მარკოვულ ჯაჭვს, დისკრეტული მდგომარეობითა და უწყვეტი დროით. ზემოთ მოცემული ტოლობიდან, სადაც წარმოდგენილია სრული ინფორმაცია სისტემის ერთი

მდგომარეობიდან მეორეში გადასვლაზე და მის სტაციონალურ ქცევაზე შეგვიძლია M/G/1-სისტემისთვის დამახასიათებელი რამდენიმე შედეგი მივიღოთ.

ყველაზე ცნობილი შედეგი, რომელიც დამახასიათებელია M/G/1-სისტემისთვის არის პლიაჩკა-ხინჩინის ფორმულა, რომელსაც შემდეგი სახე აქვს:

$$W = \frac{\overline{\lambda X}^2 / 2}{1-\rho} \quad . \ . \ . \qquad (2.27)$$

იგი გამოსახავს ლოდინის საშუალო დროისათვის რიგში წონასწორობის მდგომარეობას. გამოსახულებაში მრიცხველი ტოლია დროის მათემატიკური ლოდინის, რომლის განმავლობაში ახალი მოწოდებული მოთხოვნა უნდა დაელოდოს რიგს, სანამ დამთავრდება წინა მოთხოვნის მომსახურება (თუკი ეს არსებობს), რომელიც ახალი მოთხოვნის მიწოდების მომენტში იმყოფებოდა მომსახურე ხელსაწყოში. პლიაჩკა-ხინჩინის ფორმულის გამოყენებით შეგვიძლია განვსაზღვროთ სისტემაში დაყენებულ მოთხოვნათა საშუალო რაოდენობა:

$$\overline{N} = \rho + \frac{\overline{\lambda^2 X^2} / 2}{1-\rho}; \qquad (2.28)$$

სისტემისთვის დამახასიათებელი მნიშვნელობის მაქსიმუმი, რისი გაკეთებაც შეიძლება სხვადასხვა სიდიდეების განაწილებისას, ძირითადად ესაა მათი წარმონაქმნის პოვნა.

წარმონაქმნი აღვნიშნოთ P_k სიდიდით და გამოვსახოთ შემდეგი ფორმულის საშუალებით:

$$Q(z) = \sum_{k=1}^{\infty} P_k Z^k = B(\lambda - \lambda_z) \frac{(1-\rho)(1-z)}{B(\lambda - \lambda_z) - z}, \ . \ . \ . \quad (2.29)$$

სადაც $B(\lambda - \lambda_z)$ არის მომსახურების დროის წერტილში ლაპლასის წარმონაქმნის სიმკვრივის განაწილება, ხოლო ტოლობის მარჯვენა მხარეს ჩაწერილ გამოსახულებას კი ეწოდება

პლიაჩკა–ხინჩინის წარმონაქმნთა ტოლობა სისტემაში დაყენებულ მოთხოვნათა რაოდენობისათვის. ქედან შეგვიძლია გამოვთვალოთ ლოდინის დრო ლაპლასის წარმონაქმნისათვის:

$$W(s) = \frac{s(1-\rho)}{s - \lambda + \lambda B(s)}, \quad . \quad . \quad . \qquad (2.30)$$

ხოლო სისტემაში ლაპლასის წარმონაქმნის შესვლის დრო გამოისახება შემდეგნაირად:

$$S(s) = B(s)\frac{S(1-\rho)}{S - \lambda + \lambda B(s)}; \qquad (2.31)$$

მნიშვნელოვანი ალბათობის პროცესი, რომელიც ჯერ არ განხილულა, არის სისტემაში t დროის განმავლობაში დაუმთავრებელი სამუშაო, რომელსაც პირობითად $v(t)$ – თი აღვნიშნავთ. ეს პროცესი შეგვიძლია წარმოვადგინოთ, როგორც მარკოვის პროცესი, რომლის სიდიდე წარმოადგენს დახარჯულ დროს, სისტემის ყველა მოთხოვნიდან განთავისუფლებაზე, არსებული დროის t მომენტში, იმ პირობით, რომ ამ მომენტის შემდეგ სისტემაში აღარ მოხვდება ახალი მოთხოვნები.

რიგით მოხმარებული დაუმთავრებელი სამუშაო სისტემისათვის წარმოადგენს მოთხოვნის ლოდინის დროს, რომელიც მიწოდებულია t მომენტში, ამიტომ $v(t)$ – ს ხანდახან უწოდებენ „ვირტუალურ“ ლოდინის დროს.

მოცემული სისტემის შემთხვევაში, მიწოდების წესით მომსახურებასა და პუასონის შემავალი ნაკადით მოცემულ დაუმთავრებელ სამუშაოს აქვს ისეთივე სტატისტიკა, როგორიც ლოდინის ფაქტობრივ დროს მოთხოვნილების მიწოდებისას.

ზოგადი აღნიშვნის მეთოდის გამოყენებით ზოგიერთი ზემოთ მოყვანილი წარმონაქმნები შეიძლება მივიღოთ როგორც ალბათობა. ეს დაფუძნებულია იმაზე, რომ ყველა მიწოდებულ მოთხოვნას ესმება ნიშანი (1 – 2), მაშინ შემავალი ნაკადის მწარმოებელი ფუნქცია $P(2,t)=E[2^{N(t)}]$ შეიძლება წარმოვადგინოთ,

რომ არცერთ მოთხოვნას, მოწოდებულს (0,1) მონაკვეთში, არ დაესმება ნიშანი (ანუ არ იქნება აღნიშნული). ანალოგიურად განიხილება სხვა მონაკვეთებიც. თუ განვიხილავთ ამ მონაკვეთებიდან თავისუფლად შემავალ პუასონის ნაკადსა და P ალბათობას იმისა, რომ X დროის მანძილზე სისტემაში შემოსული არცერთი მოთხოვნა არ იქნება წარმოდგენილი:

$$P = X(\lambda - \lambda_2)$$

განვიხილოთ ისევ რაღაც დროის მონაკვეთი და მასზე დამოუკიდებელი პუასონის პროცესი. დავუშვათ რომ წერტილები გენერირდება პუასონის პროცესით, მაშინ Q=X(λ).

ამგვარად ძირითად მიმართულებებს, რომლებიც ხშირად გვხვდება მასობრივი მომსახურების თეორიაში შეიძლება მივცეთ საინტერესო ინტერპრეტაცია.

2.2.4. G/M/1 სისტემა

G/M/1 – სისტემა ხასიათდება მოთხოვნათა შემოსვლის მომენტებს შორის განაწილების თანაბარი კანონით. მომსახურების დრო განაწილებულია მაჩვენებლიანი კანონის საშუალო 1/λ მნიშვნელობით.

ძირითადი რეკურენტული დამოკიდებულება, რომელიც აღწერს G/M/1 – სისტემის მუშაობას, მსგავსია M/G/1-სისტემის და გამოისახება შემდეგი სახით:

$$q'_{n+1} = q'_n + 1 - V'_{n+1}C_n \quad . \; . \; . \qquad (2.32)$$

სადაც q'_n - მოთხიენათა რაოდენობაა, რომელიც გამოსახავს სისტემაში შემოსულ C_n მოთხოვნებს, ხოლო V'_{n+1} მოთხოვნათა რაოდენობაა, რომელიც მოთავსებულია C_n და C_{n+1} მოთხოვნათა შემოსვლის მომენტებს შორის. გარკვეული თანამიმდევრობით $\{q_n\}$ ქმნის მარკოვის რგოლს. მოთხოვნათა რაოდენობის განაწილება, რომელიც აღნიშნავს სისტემაში კვლავ შემოსულ მოთხოვნებს გამოისახება $r_k = (1-\sigma)\sigma^k$ ფორმულით, სადაც k=0, 1, 2, . . .

ლოდინის დრო გამოისახება ტოლობით:

$$W(y) = 1 - \sigma e^{-\mu(1-\sigma)y},$$

ხოლო ლოდინის საშუალო დრო: $W = \frac{\sigma}{\mu(1-\sigma)}$;

ლოდინის დრო ნაწილდება მაჩვენებლიანი კანონით და არ არის დამოკიდებული განაწილების სახეობაზე განსაზღვრული დროის მონაკვეთში მოთხოვნის მისაღებად.

2.2.5. G/M/m სისტემა

G/M/m - სისტემა ხასიათდება დროის თანაბარი განაწილებით მიწოდების მომენტებსა და გამოყენებული ხელსაწყოების ერთ რიგს შორის, თითოეული მათგანი განაწილებულია მაჩვენებლიანი კანონით დროის მოხმარების საშუალო მნიშვნელობით. $1/\mu$ მოთხოვნა სრულდება მიწოდებისთანავე, სადაც ძირითადი პარამეტრია σ. მოცემულ შემთხვევაში σ აღინიშნება, როგორც ერთადერთი მნიშვნელობა $0 \le \sigma < 1$ არეში. $\sigma = A(m\mu - m\mu\sigma)$, ამ ფორმულით განვსაზღვრავთ რიგების სიგრძის განაწილებას, რომლებსაც თან ახლავს ახალი მოთხოვნები იმ პირობით, რომ მოთხოვნა უნდა ჩადგეს რიგში, ეს პროცესი განისაზღვრება შემდეგნაირად: $(1-\sigma)\sigma^n$, $n \ge 0$.

უნდა აღინიშნოს რომ, როგორც G/M/1 – სისტემაში, რიგთა სიგრძე აღიწერება გეომეტრიული გადანაწილებით. აღვნიშნოთ r_k - თი ალბათობა იმისა, რომ ახლად შემოსული მოთხოვნა, k მოთხოვნას უსწრებს წინ აღწერილ სისტემაში.

ახლად შემოსული მოთხოვნის ალბათობა გამოითვლება შემდეგი ფორმულის საშუალებით:

$$R_k = \begin{cases} rk/j, & 0 \le k \le m-2, \\ \sigma^{k-m+1} & m-2 < k. \end{cases} \quad \dots \quad (2.33)$$

R_k-სთვის უნდა გამოვთვალოთ j და m-1 მნიშვნელობები, მოცემული $0 \le k \le m-2$ უტოლობის გათვალისწინებით j გამოითვლება შემდეგი ფორმულის სახით:

$$j = \frac{1}{[1/(1-\sigma)] + \sum_{k=0}^{m-2} R_k}; \quad \ldots \qquad (2.34)$$

R_{k-1}-ის ანუ ახლად შემოსული წინა მოთხოვნის დაყენების ალბათობა გამოითვლება შემდეგი ფორმულის საშუალებით:

$$R_{k-1} = \frac{R_k - \sum_{i=k}^{m-2} R_{ipik} - \sum_{i=m-1}^{\infty} \sigma^{i+1-m} pik}{pk-1,k}; \quad \ldots \quad (2.35)$$

სადაც p_i გადატანითი ალბათობა არატრივიალურია და გამოითვლება შემდეგი ოთხი ტოლობით i და j ინდექსებზე დამოკიდებულებით:

$$p_{ij} = 0,\ j > i+1,$$

$$\rho_{ij} = \int_0^{\infty} \binom{i+1}{j} [1-e^{-\mu t}]^{i+1-j} e^{-\mu t j} dA(t),$$

$j \le i+1 \le m$;

$$\beta_n = p_{i,i+1-n} = \int_0^{\infty} \frac{(m\mu t)^n}{n!} e^{-m\mu t} dA(t),$$

$0 \le n \le i+1-m,\ m \le i$;

$$\rho_{ij} = \int_0^{\infty} \binom{m}{j} e^{j\mu t} \left| {}_0^{-t} \int \frac{(m\mu y)^{i-m}}{(i-m)!} (e^{-\mu y} - e^{-\mu t})^{m-j} m\mu dy \right|$$

აღნიშნული ტოლობების გამოყენებით, შეგვიძლია ვიპოვოთ ლოდინის საშუალო დრო:

$$W = \frac{j\sigma}{m\mu(1-\sigma^2)}; \quad \ldots \qquad (2.36)$$

აღსანიშნავია, რომ G/M/m სისტემისათვის ლოდინის დრო განაწილებულია მაჩვენებლიანი კანონის მიხედვით.

2.2.6. G/G/1 სისტემა

მმს-ში ყველა შესრულებული სამუშაო აღიწერება ისეთი შემთხვევითი სიდიდეებით, როგორიცაა დროის მონაკვეთები: მეზობელ მოთხოვნათა მიწოდების t_n ხანგრძლივობასა და მომსახურების x_n ხანგრძლივობას შორის. ამ სისტემის ზოგადი აღწერისას უნდა აღინიშნოს, რომ ეს სიდიდეები არ ჩნდება ცალ-ცალკე, არამედ ყოველთვის ფიგურირებს ერთად სხვადასხვა სახით; ამიტომ განვიხილავთ ახალ სიდიდეს, რომელიც დაკავშირებულია (n-m) მოთხოვნების C_n-თან, კერძოდ $u_n = x_n - t_{n+1}$.

ეს შემთხვევითი სიდიდე წარმოადგენს სხვაობას C_n-მოთხოვნის მომსახურების x_n სიდიდესა და t_{n+1} „შესვენების“ სიდიდეს შორის. ეს სხვაობა საშუალოდ იქნება უარყოფითი ე.ი. შესვენება იქნება სისტემის დატვირთვაზე მეტი.

თუ ავიღებთ ასეთი სისტემის საშუალო მნიშვნელობას, მივიღებთ შემდეგ ტოლობას:

$$E[u_n] = \bar{t}(p-1).$$

ეს გამოსახულება ერთის მხრივ არ არის დამოკიდებული n –ზე, მეორეს მხრივ, მას აქვს უარყოფითი მნიშვნელობა $p<1$;

G/G/1 სისტემის გამოკვლევის დროს მნიშვნელოვანია ლოდინის w_n დრო, რომელიც სჭირდება C_n მოთხოვნის დაკმაყოფილებას. ეს შემთხვევითი სიდიდე პირდაპირ კავშირშია შემდეგ ტოლობასთან:

$$w_{n+1} = \max[0, w_n + u_n] \quad . \ . \ . \qquad (2.37)$$

მოყვანილ ტოლობაში მნიშვნელოვანი ადგილი უკავია u_n სიდიდეს. (2.37) არის G/G/1 სისტემის ძირითადი ტოლობა. თანამიმდევრობა $\{w_n\}$ ქმნის მარკოვის პროცესს უწყვეტი დროითა და მდგომარეობით. ზემოთ აღწერილი ოპერატორის არჩევის საშუალებით მაქსიმუმი ჩაიწერება სემდეგი სახით:

$(x)^+ = \max(0, x)$ სტაბილური სისტემისათვის ($p<1$) არსებობს განსაზღვრული შემთხვევითი სიდიდე $\tilde{w}$, რომელიც აღწერს სტაციონალური ლოდინის დროს. ზემოთ მოყვანილი ტოლობიდან შეიძლება დავინახოთ, რომ $\tilde{w}$-ს უნდა ჰქონდეს ისეთივე განაწილების კანონი, როგორც $(w_n + u_n)$.

G/G/1 სისტემის ძირითადი არსი მდგომარეობს იმაში, რომ სისტემაში დაყენებული მოთხოვნები არ უნდა იდგნენ რიგში.

2.2.7. M/M/1/K სისტემა სასრული დამგროვებლით

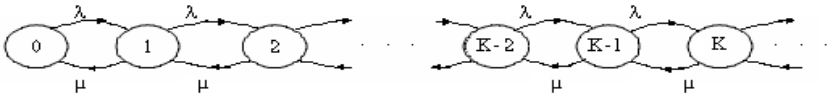
განვიხილოთ პირველი მმს, რომლისთვისაც ფიქსირებულია რიგში მდგომ მოთხოვნათა მაქსიმალური რაოდენობა; კერძოდ, სისტემაში არის მოთხოვნათა K მაქსიმალური რაოდენობა (აღნიშნულ რაოდენობაში გათვალისწინებულია აგრეთვე მომსახურებაზე მდგარი მოთხოვნებიც), თუ ამ რაოდენობის ზემოთ ერთი მოთხოვნაც კი შემოვა სისტემაში, იგი ღებულობს უარს და მოთხოვნა ტოვებს სისიტემას მომსახურების გარეშე.

სისტემაში ახალი მოთხოვნის შემოსვლა ხდება პუასონის კანონის თანახმად, მაგრამ სისტემა ღებულობს მხოლოდ ისეთ მოთხოვნებს, რომელთა რაოდენობა მკაცრად განსაზღვრულია და აუცილებლად უნდა იყოს მოთხოვნათა K რაოდენობაზე ნაკლები. სატელეფონო სისტემაში, როდესაც მომხმარებელი ღებულობს მომსახურებაზე უარს, ასეთი სისტემა არის დანაკარგებით; სისტემას რომლის K=1, უწოდებენ დაბლოკილ გამოძახებათა ამოვარდნის სისტემას, რომელსაც აქვს ერთი მომსახურე მოწყობილობა.

საინტერესოა ის გარემოება, რომ თითქოსდა ასეთი რთული სისტემა შეიძლება შევუთავსოთ ისეთ მარტივ მოდელს, როგორიცაა გაქრობის და გამრავლების პროცესის მოდელი. ამ თვალსაზრისით, როდესაც სისტემა გადაივსება მოთხოვნებით, ხდება შემავალი პუასონის ნაკადის დროითი გადაფარვა:

$$\lambda_k = \begin{cases} \lambda, k < K; \\ 0, k \geq K; \end{cases} \qquad \mu_k = \mu, \quad \text{k=1,2, . . . , K.}$$

$\frac{\lambda_k}{\mu_{k+1}} < C < 1$ უტოლობიდან ჩანს, რომ ეს სისტემა ყოველთვის ერგოდიულია. გადასასვლელების ინტენსივობის დიაგრამა მოცემულია 2.17 ნახაზზე.



ნახ.2.17. გადასასვლელთა ინტენსივობის დიაგრამა M/M/1/K სისტემისთვის

$$p_k = p_0 \prod_{i=0}^{k=1} \frac{\lambda_i}{\mu_{i+1}}, \qquad \text{k=1,2, . . .} \qquad (2.38)$$

პირდაპირ თუ გადავალთ (4.28) ტოლობაზე მივიღებთ:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{\mu}, \qquad \text{k≤K,}$$

ან

$$p_k = p_0 \left(\frac{\lambda}{\mu} \right)^k, \qquad \text{k≤K. . . .} \qquad (2.39)$$

ამას გარდა ადგილი აქვს შემდეგ დამოკიდებულებას:

$$p_k = 0, \quad \text{k> K . . .} \qquad (2.40)$$

იმისათვის, რომ ვიპოვოთ p_0, (2.38) და (2.39) შევიტანოთ (2.41)-ში

$$p_0 = \frac{1}{1+\sum_{k=1}^{\infty}\prod_{i=0}^{k-1}\frac{\lambda_i}{\mu_{i+1}}} \quad \ldots \quad (2.41)$$

მივიღებთ:

$$p_0 = \left[1+\sum_{k=1}^{K}\left(\frac{\lambda}{\mu}\right)^k\right]^{-1} = \left[1+\frac{(\lambda/\mu)(1-(\lambda/\mu)^K)}{1-\lambda/\mu}\right]^{-1},$$

შესაბამისად

$$p_0 = \frac{1-\lambda/\mu}{1-(\lambda/\mu)^{K+1}}.$$

საბოლოოდ კი მივიღებთ:

$$p_k = \begin{cases} \dfrac{1-\lambda/\mu}{1-(\lambda/\mu)^{K+1}}\left(\dfrac{\lambda}{\mu}\right)^k, & 0\le k\le K; \\ 0, \text{ სხვა შემთხვევებისთვის} \end{cases} \quad \ldots \quad (2.42)$$

სისტემისათვის სადაც ხდება დაბლოკილ გამოძახებათა ამოვარდნა ჩაიწერება შემდეგი განტოლება:

$$p_k = \begin{cases} \dfrac{1}{1+\lambda/\mu}, k=0 \\ \dfrac{\lambda/\mu}{1+\lambda/\mu}, k=1=K \\ 0, \text{ სხვა შემთხვევებისთვის} \end{cases} \quad (2.43)$$

2.2.8. M/M/m სისტემა (m მომსახურე მოწყობილობითა და დანაკარგებით)

კვლავ განვიხილოთ სიტემა, რომელიც უზრუნველყოფს დაბლოკილ გამოძახებათა ამოვარდნას და m მომსახურე მოწყობილობის არსებობას. ყოველი ახალი მოთხოვნა იკავებს მისთვის განკუთვნილ მომსახურე მოწყობილობას, ხოლო თუ

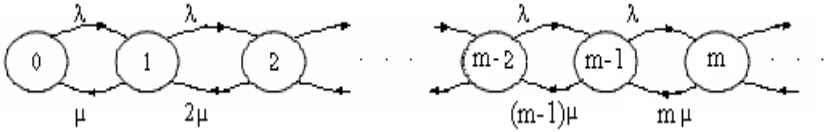
სისტემაში მოთხოვნის შემოსვლის მომენტში ყველა მომსახურე მოწყობილობა დაკავებულია, მაშინ ხდება მოთხოვნათა დაკარგვა.

როგორც ზემოთ აღვნიშნეთ, ეს შემთხვევაც შეესაბამება გაქრობის და გამრავლების პროცესს:

$$\lambda_k = \begin{cases} \lambda, & k<m; \\ 0, & k\geq m; \end{cases}$$

$$\mu_k = k, \qquad k=1,2,....m.$$

აქაც ყოველთვის გარანტირებულია ერგოდიულობის პირობა. 2.18 ნახაზზე გამოსახულია გადასასვლელთა ინტენსივობის დიაგრამა სასრული მარკოვის ჯაჭვით:



ნახ.2.18 გადასასვლელთა ინტენსივობის დიაგრამა M/M/m სისტემისთვის

(2.38) ფორმულის გამოყენებით მოვიღებთ:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{(1+i)\mu}, \qquad k\leq m,$$

ან

$$p_k = \begin{cases} p_0 \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}, & k\leq m; \\ 0, & k>m. \end{cases} \qquad \ldots \qquad (2.44)$$

p_0 -გამოისახება შემდეგი განტოლების საშუალებით:

$$p_0 = \left[\sum_{k=0}^{m} \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!}\right]^{-1}.$$

ეს კერძო შემთხვევა დიდ ინტერესს იწვევს სატელეფონო სისტემებში. განხილულ სისტემაში p_m ალბათობა გამოსახავს

დროის იმ შუალედს, როდესაც ყველა მომსახურე m მოწყობილობა დაკავებულია:

$$p_m = \frac{(\lambda/\mu)^m / m!}{\sum_{k=0}^{m} (\lambda/\mu)^k / k!}. \quad \ldots \qquad (2.45)$$

(2.45) ფორმულას უწოდებენ ერლანგის ფორმულას დანაკარგებით. მას ხშირად ასეთი სახითაც წერენ:

$$B(m, \lambda/\mu).$$

2.2.9. M/M/1//M*: სისტემის დატვირთვის სასრული წყარო და ერთი მომსახურე მოწყობილობა

განვიხილოთ შემთხვევა, სადაც სისტემაში შემოსული მოთხოვნათა ნაკადი არ არის უსასრულო და არ არის პუასონის კანონით განაწილებული. ე.ი. ამ შემთხვევაში გვაქვს მოთხოვნათა სასრული წყარო. სისტემის სტრუქტურა ასეთია: გვაქვს დატვირთის M წყარო; მოთხოვნა დგას ან სისტემაში (იგი წარმოადგენს რიგს რომალსაც ემსახურება ერთი ხელსაწყო) ან სისტემის გარეთ და ემზადება მოთხოვნის დასაყენებლად. თუ მოთხოვნა დგას მოსამზადებელ ჯგუფში, მაშინ მოთხოვნის სისტემაში დაყენების დროის მომენტი არის შემთხვევითი სიდიდე, რომელიც განაწილებულია მაჩვენებლიანი კანონით და მისი საშუალო მნიშვნელობა ტოლია $1/\lambda$ წამის. ჩავთვალოთ, რომ ყველა კლიენტი მოქმედებს ერთმანეთისგან დამოუკიდებლად, ეს ნიშნავს რომ თუ სისტემაში არის k მოთხოვნა, მაშინ M-k მოთხოვნა შედის მოსამზადებელ მოთხოვნათა რაოდენობაში და შესაბამისად სისტემაში დაყენებულ მოთხოვნათა საერთო ინტენსივობა ტოლია: $\lambda(M-k)$.

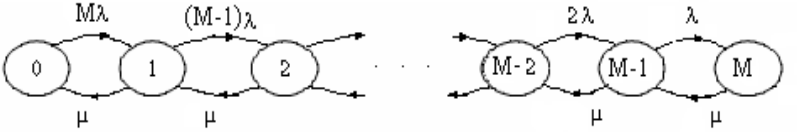
ნათელია, რომ ეს სისტემა თვითრეგულირებადია. როდესაც იგი გადაიტვირთება, წარმოიქმნება მოთხოვნათა დიდი რიგი, რის შემდეგაც მოთხოვნების დაყენების ინტენსივობა ზედმეტ მოთხოვნებს ანადგურებს და განტვირთავს სისტემას.

წარმოდგენილო მოდელი შეესაბამება გაქრობის და გამრავლების პროცესს და ხასიათდება შემდეგი პარამეტრებით:

$$\lambda_k = \begin{cases} \lambda(M-k), & 0 \le k \le M; \\ 0, & \text{სხვა შემთხვევაში} \end{cases}$$

$$\mu_k = \mu, \qquad k=1,2,3,\ldots$$

სისტემა არის ერგოდიული. მას შეუძლია M მოთხოვნათა დაგროვება და საჭირო შემთხვევაში გამოყენება. გადასავლელთა ინტენსივობის დიაგრამა გამოსახულია ნახ.2.19-ზე



ნახ.2.19 გადასავლელთა ინტენსივობის დიაგრამა M/M/1//M სისტემისათვის

2.18 განტოლების გამოყენებით p_k - მიიღებს შემდეგ სახეს:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda(M-i)}{\mu}, \qquad 0 \le k \le M.$$

ამ თავლსაზრისით,

$$p_k = \begin{cases} p_0 \left(\frac{\lambda}{\mu}\right)^k \frac{M!}{(M-k)!}, 0 \le k \le M \\ 0, k > M \end{cases} \qquad \ldots (2.46)$$

p_0 -ისთვის კი მივიღებთ:

$$p_0 = \left[\sum_{k=0}^{M} \left(\frac{\lambda}{\mu}\right)^k \frac{M!}{(M-k)!} \right]^{-1} \qquad (2.47)$$

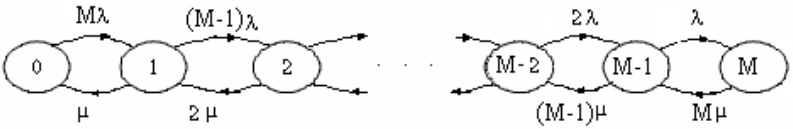
2.2.10. M/M/∞/M: დატვირთვის წყაროს სასრული რაოდენობა და უსასრულო მომსახურე მოწყობილობები

ისევ განვიხილოთ შემთხვევა, როდესაც მოცემული გვაქვს სისტემის დატვირთვის სასრული წყარო, მაგრამ წინა სისტემიდან განსხვავებით მის ყოველ მოთხოვნას ემსახურება ცალკე მოწყობილობა. ასეთი მოდელი წარმოდგენილია შემდეგი თვალსაზრისით:

$$\lambda_k = \begin{cases} \lambda(M-k), & 0 \le k \le M; \\ 0, & \text{სხვა შემთხვევაში} \end{cases}$$

$$\mu_k = k\mu, k = 1,2,3,\ldots$$

ეს სისტემაც ერგოდიულია. მისი გადასავლელთა ინტენსივობის დიაგრამა მოცემულია 2.20 ნახაზზე.



ნახ.2.20 გადასავლელთა ინტენსივობის დიაგრამა M/M/∞//M სისტემისათვის

ამ სისტემის p_k -ალბათობა 2.11 განტოლების გათვალისწინებით მიიღებს შემდეგ სახეს:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda(M-i)}{(i+1)\mu} = p_0 \left(\frac{\lambda}{\mu}\right)^k \binom{M}{k}, 0 \le k \le M \quad \ldots \quad (2.48)$$

სადაც ბინომინალური კოეფიციენტი განისაზღვრება შემდეგნაირად:

$$\binom{M}{k} = \Delta \frac{M!}{k!(M-k)!}$$

p_0 -ის განსაზღვრით კი მივიღებთ:

$$p_0 = \left[\sum_{k=0}^{M} \left(\frac{\lambda}{\mu}\right)^k \binom{M}{k}\right]^{-1},$$

მაშასადამე, $p_0 = \frac{1}{(1+\lambda/\mu)^M}$.

საბოლოოდ:

$$p_k = \begin{cases} \frac{\left(\frac{\lambda}{\mu}\right)^k \binom{M}{k}}{(1+\lambda/\mu)^M}, & 0 \le k \le M; \\ 0, \text{ სხვა შემთხვევაში} \end{cases} \quad \ldots \quad (2.49)$$

სირთულეს არ წარმოადგენს გამოვთვალოთ სისტემაში დაყენებული მოთხოვნათა საშუალო რიცხვი:

$$\overline{N} = \sum_{k=0}^{M} k p_k = \frac{\sum_{k=0}^{M} k \left(\frac{\lambda}{\mu}\right)^k \binom{M}{k}}{(1+\lambda/\mu)^M}.$$

ვიცით, რომ $\overline{N} = \frac{\rho}{1-\rho}$, ამიტომ საბოლოოდ მოთხოვნათა საშუალო რიცხვი გამოითვლება შემდეგი ფორმულით:

$$\overline{N} = \frac{M\lambda/\mu}{1+\lambda/\mu}.$$

2.2.11. M/M/m/K/M: დატვირთვის წყაროს სასრული რაოდენობა, m მომსახურე მოწყობილობა და სასრული დამგროვებელი

ეს სისტემა ზემოთ განხილულ სისტემებთან შედარებით ითვლება რთულ სისტემად, შესაბამისად მისი პარამეტრებიც შეიძლება გამოყენებულ იქნას განხილულ ყველა შემთხვევაში.

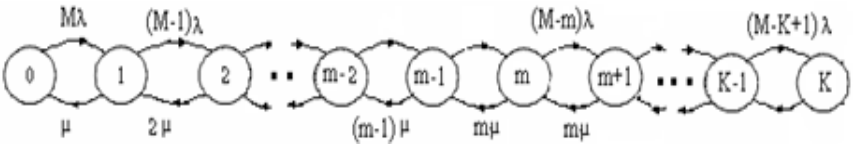
M/M/m/K/M სისტემაში არსებობს M მოთხოვნათა სასრული რაოდენობა, სადაც ყველა შემოსულ მოთხოვნათა ინტენსივობა ტოლია λ, ამას გარდა სისტემა შეიცავს m მოსახურე მოწყობილობას, მათ შორის ყოველი მოწყობილობა აღიწერება μ პარამეტრით.

საბოლოოდ სისტემას აქვს მოთხოვნათა ლოდინის სასრული რაოდენობა, ისე რომ მოთხოვნათა ლოდინის საერთო რაოდენობა (რიგს დამატებული მომსახურებაზე მდგარი მოთხოვნა) არ უნდა აღემატებოდეს K-ს. სავარაუდოა, რომ $M \geq K \geq m$;

სისტემაში შემოსული მოთხოვნები, მაშინ როდესაც მასში უკვე არის K მოთხოვნა, ვარდება სისტემიდან და დაუყოვნებლივ დგება მოსამზადებელ ჯგუფში, თითქოს ისინი სრულად დაკმაყოფილებულია. ეს პროცესი გამოისახება გაქრობის და გამრავლების პროცესის პარამეტრებით:

$$\lambda_k = \begin{cases} \lambda(M-k), 0 \leq k \leq K-1 \\ 0, \text{ დანარჩენ შემთხვევებში;} \end{cases} \quad \mu_k = \begin{cases} k\mu, 0 \leq k \leq m; \\ m\mu, k \geq m. \end{cases}$$

2.21 ნახაზზე გამოსახულია სასრული დიაგრამა, სადაც წარმოდგენილია გადასასვლელების ინტენსივობა.



ნახ.2.21. გადასავლელთა ინტენსივობის დიაგრამა M/M/m/K/M სისტემისთვის

იმისათვის, რომ გამოვიყენოთ 2.11 განტოლება, უნდა განვიხილოთ ორი პირობა,

1) $0 \leq k \leq m-1$, მივიღებთ:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda(M-i)}{(i+1)\mu} = p_0 \left(\frac{\lambda}{\mu}\right)^k \binom{M}{k}, 0 \leq k \leq m-1. \quad \ldots \quad (2.50)$$

2) $m \leq k \leq K$, მივიღებთ:

$$p_k = p_0 \prod_{i=0}^{m-1} \frac{\lambda(M-i)}{(i+1)\mu} \prod_{i=m}^{k-1} \frac{\lambda(M-i)}{m\mu} =$$

$$= p_0 \left(\frac{\lambda}{\mu}\right)^k \binom{M}{k} \frac{k!}{m!} m^{m-k}, \quad m \le k \le K \cdot \cdot \cdot \quad (2.50)$$

სტაციონალური ალბათობა, როდესაც სისტემაში გვაქვს სუფთა დანაკარგების შემთხვევა (როცა M≥K=m), გამოითვლება შემდეგი ფორმულით:

$$p_k = \frac{\binom{M}{k}\left(\frac{\lambda}{\mu}\right)^k}{\sum_{i=0}^{m}\binom{M}{i}\left(\frac{\lambda}{\mu}\right)^i}, \qquad k=0,1,2,..., \ m. \qquad \cdot \cdot \cdot \ (2.51)$$

ამ განაწილებას უწოდებენ ერლანგის განაწილებას.

როგორც ვნახეთ მასობრივი მომსახურების სისტემების საკმაოდ საინტერესო სტრუქტურა, შესაძლებელია აღწერილი იყოს გამრავლებისა და გაქრობის პროცესით. ვნახეთ კერძო შემთხვევები, სადაც დემონსტრირებული იყო სხვადასხვა მაგალითები: შემთხვევა როდესაც გვქონდა რამდენიმე მომსახურე მოწყობილობა, დატვირთვის წყაროს სასრული რაოდენობა, შემთხვევა სასრული რიგით და ამ შესაძლებლობით მოცემული სხვადასხვა კომბინაციები. საბოლოოდ კი ვღებულობდით სისტემის სტაციონალურ ალბათობის $\{p_k\}$ გამოსათვლელ ფორმულას.

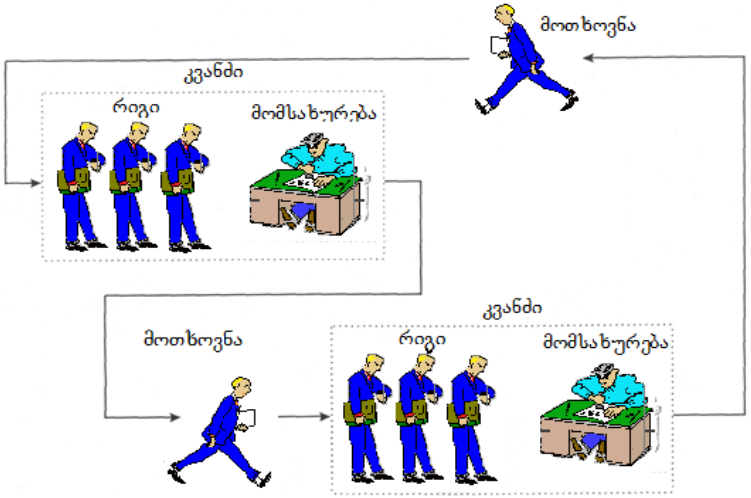
ჩვენ განვიხილეთ მხოლოდ ისეთი სისტემები რომელთათვისაც სტაციონალური ალბათობა შესაძლებელი იყო გვეპოვნა მოცემული განტოლების დახმარებით.

განხილული სისტემების დახმარებით შესაძლებელია გადაჭრილი იქნას ბევრი ისეთი ეკონომიკური ამოცანა, რომელთა ამოხსნაც ჩვეულებრივი მათემატიკური აპარატით შეუძლებელია.

III თავი
მასობრივი მომსახურების მოდელები და WinPepsy-ინსტრუმენტი

3.1 რიგების ქსელი

რიგების ქსელი შედგება ცალკეული რიგებისა და მომსახურე ობიექტებისგან. რიგი, რომელიც ელოდება მომსახურებას, შეიძლება წარმოვადგინოთ როგორც მომლოდინე სისტემა ან კვანძი. 3.1 ნახაზზე მოცემულია რიგების ქსელი და რიგების სისტემა (კვანძი):



ნახ.3.1. რიგების ქსელი და რიგების სისტემა

გრაფში დგება შეკვეთები, რომლებიც გადაეცემა კვანძიდან კვანძში. ყოველ კვანძს გააჩნია გარკვეული სტრატეგია, რომელიც აწესრიგებს თუ როგორი სახით უნდა დადგეს მოთხოვნა რიგში და როგორ უნდა მოხდეს მისი რედაქტირება.

მოცემული მოთხოვნის დროებითი შეყოვნებისას, მომსახურე ობიექტი ყოველ კვანძში აღწერს მომსახურების დროის განაწილებას სპეციალური პარამეტრებით.

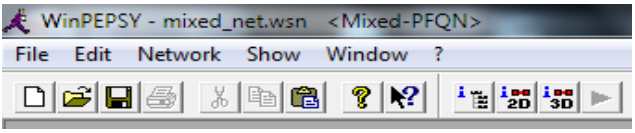
ქსელი სადაც განთავსებულია სხვადასხვა სახის შეკვეთები, ერთიანდება დავალებათა კლასში, რომელიც იყოფა ორ კლასად:

1. ჩაკეტილი კლასი: ქსელური გრაფი ჩაკეტილია. ამ შემთხვევაში ვერცერთი ახალი მოთხოვნა ვერ შემოვა ქსელში და აგრეთვე ვერცერთი მოთხოვნა ვერ დატოვებს რიგს. ქსელი მუდმივად ინარჩუნებს თანაბარ მოთხოვნათა რაოდენობას;

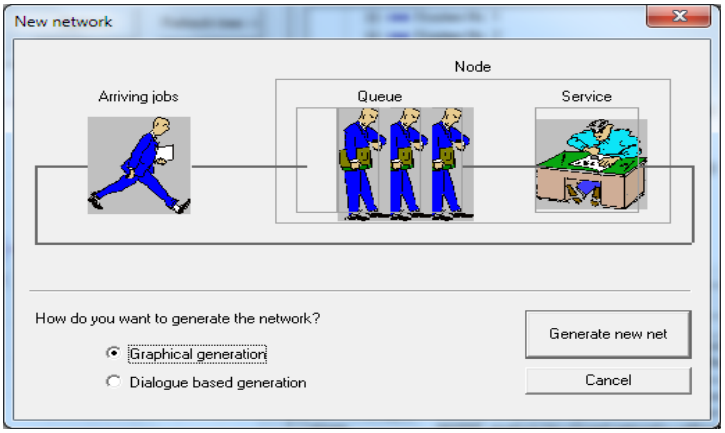
2. ღია კლასი: ქსელური გრაფი შეიცავს საწყის და სასრულ მოთხოვნათა წყაროს. ამ შემთხვევაში მოთხოვნათა წყარო შეიძლება ვარეგულიროთ.

3.2. ახალი ქსელის აგება

ახალი ქსელის შესაქმნელად WinPepsy რედაქტორში მთავარი მენიუდან (ნახ.3.2-ა) ავირჩევთ File->New და მივიღებთ 3.2-ბ ნახაზზე მოცემულ სქემას.

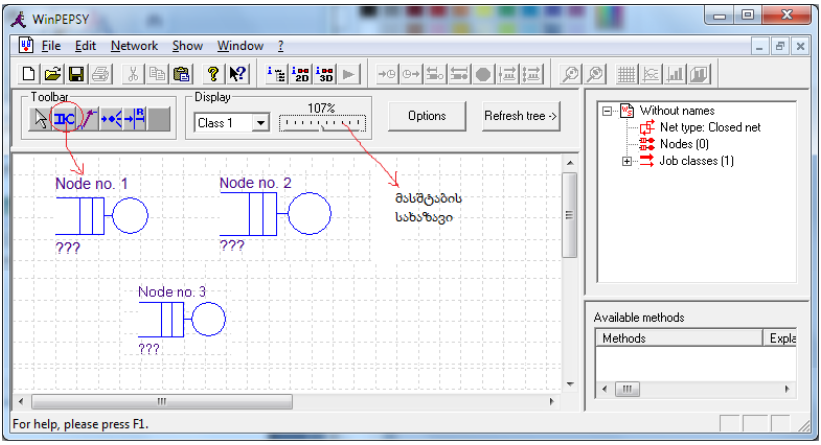


ნახ.3.2-ა



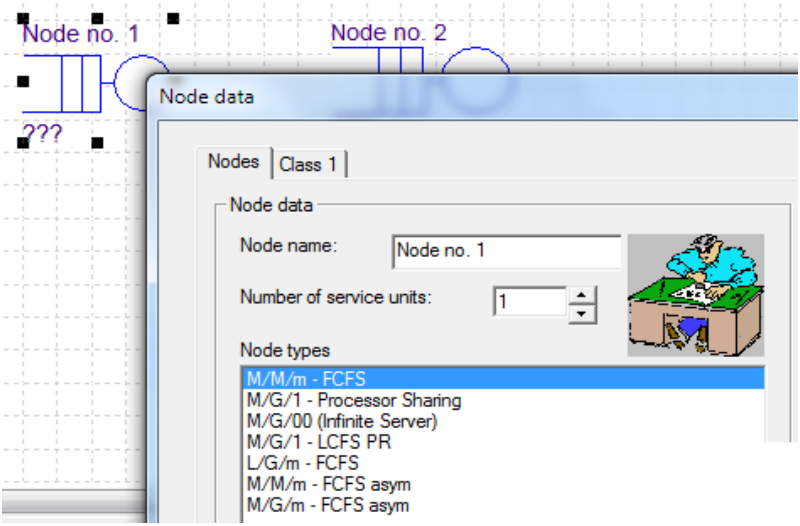
ნახ.3.2-ბ. ახალი ქსელის შექმნა

„გრაფიკული გენერაციის“ არჩევით ეკრანზე გამოვა 3.3 ნახაზზე ნაჩვენები გრაფიკული რედაქტორის ფანჯარა. ვირჩევთ გრაფიკული მენიუდან მითითებულ სიმბოლოს (**„კვანძი“**) და გადმოგვაქვს მუშა არეში (სამი კვანძი: no 1-3). მასშტაბის სახაზავით შეგვიძლია ვცვალოთ სქემის ზომები.



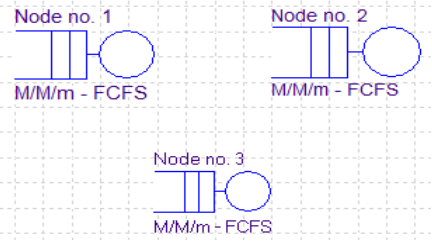
ნახ.3.3

კვანძის ტიპი აირჩევა მაუსის მარჯვენა ღილაკით (ნახ.3.4-ა).



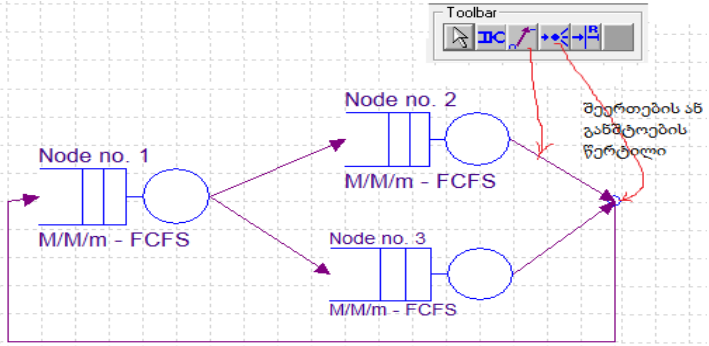
ნახ.3.4-ა

თუ ავირჩიეთ M/M/m FCFS (First-Come, First-Served) სამივე კვანძისთვის, მივიღებთ 3.4-ბ ნახაზს.



ნახ.3.4-ბ

კვანძების დაკავშირება ხდება რედაქტორის მენიუში „ისრის“ სიმბოლოს გააქტიურებით. თუ შევაერთებთ კვანძებს და შევკრავთ უკუკავშირით, მივიღებთ 3.5 ნახაზზე მოცემულ ქსელს.



ნახ.3.5

საბაზო დიალოგური ფანჯრის შექმნისას მომხმარებელმა ზუსტად უნდა იცოდეს თუ როგორ უნდა გამოიყურებოდეს მისი საანალიზო ფანჯარა. ამავე დროს იგი შედგენილი უნდა იყოს მარტივად და გასაგებად. ამ ფანჯარაშივე შეგვიძლია დავარეგისტრიროთ დამატებითი ცვლილებებიც.

ქსელის საბაზისო დიალოგური ფანჯრის სახით წარმოდგენის მნიშვნელოვანი მხარე ის არის, რომ თვალნათლივ დავინახავთ გადასავლელის ხასიათს და განვსაზღვრავთ, თუ რა ალბათობით იქნება შესაძლებელი შემოსული მოთხოვნების შესრულება.

მოცემული დიალოგური ფანჯარა გამოჩნდება მაშინ, როდესაც მომხმარებელი საბაზო დიალოგურ ფანჯარასა და ქსელს

შორის, გრაფიკული რედაქტორის დახმარებით შექმნის ერთობლივ გრაფს. ეს გრაფი მეტად მნიშვნელოვანია შემდგომი შედეგების მისაღებად (ნახ.3.2-ბ).

ქსელის გრაფიკულ რედაქტორს აქვს კომფორტული ინტერფეისი, რაც ხელსაყრელს ხდის მონაცემთა ინტერაქტიულ გრაფიკულ შეტანას. ეს პროცესი არსებითად მნიშვნელოვანია იმ მომხმარებელთათვის, რომელთაც აქვთ მხოლოდ მიახლოებითი წარმოდგენა გასაანალიზებელ ქსელზე და სურვილი აქვთ მასზე გარკვეული მანიპულირება მოახდინონ თვალსაჩინოდ.

ყველაზე დიდი დადებითი მხარე, რომელიც გააჩნია ასეთ ქსელურ გრაფს არის ის, რომ იგი შედგება დუბლირებული საბაზო ქსელისგან, სადაც შესაძლებელია ქვექსელებიდან მოხდეს მონაცემთა გადაცემა, წაშლა ან შეცვლა. ამის შედეგად ციკლური ქსელური სტრუქტურა დიდი დანახარჯების გარეშე საშუალებას მოგვცემს კვანძების გადასასვლელები, ერთი კლასიდან მეორეში უკავშირდებოდეს ერთმანეთს და მონაწილეობა მიიღოს ქსელის საერთო მუშაობაში.

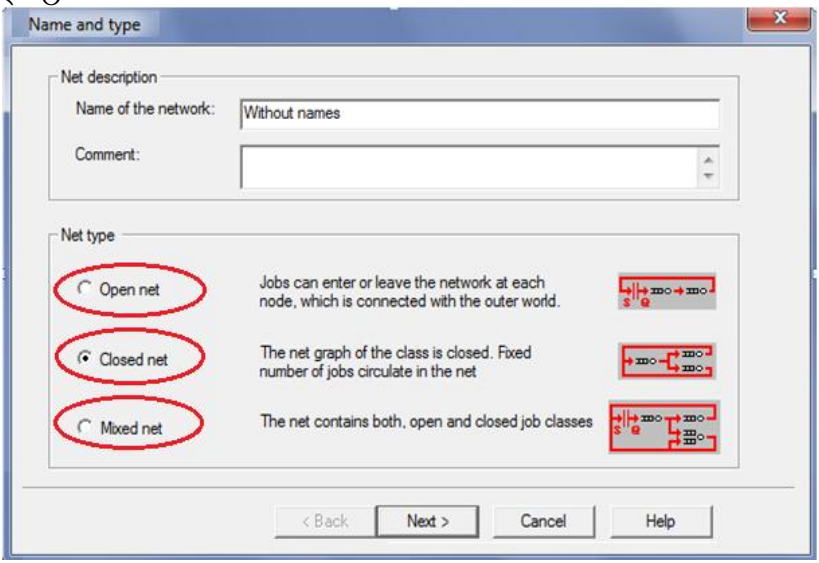
დიალოგურ ფანჯარაში სასურველ მოთხოვნათა შერჩევისას არსებული მეთოდის გამოყენებით შეიქმნება რამდენიმე პატარა სურათი, რომელთა დახმარებითაც რიგების ქსელში მოხდება ცალკეული ელემენტების სიმბოლიზირება. მოგვიანებით ვნახავთ, რომ ეს დეტალი ძალზედ მნიშვნელოვანია, რათა მომხმარებელმა გაიგოს თუ ქსელის რა ნაწილზე იყო მოსული მოთხოვნა.

3.3. საბაზო დიალოგური ფანჯრის წარმოდგენა ქსელში

საბაზო დიალოგური ფანჯარა ქსელში წარმოადგენს რამდენიმე გვერდიან დიალოგურ ფანჯარას. მომხმარებელს შეუძლია გადაფურცლოს ნებისმიერ დროს წინ და უკან, აგრეთვე ცალკეულ გვერდებს შორის. დიალოგურ ფანჯრიდან ამ პროცესის განხორციელება შესაძლებელია „Wizards“ ღილაკით.

სახელი და ტიპი. ფანჯრის პირველ გვერდზე შეიძლება წარმოვადგინოთ ქსელის სახელი და კომენტარები. კომენტარი შეიძლება შეიცავდეს რამდენიმე სტრიქონს.

Ctrl+Enter“ ღილაკებით შესაძლებელია დამატებითი მონაცემების შეტანა, რის შედეგადაც მომხმარებელი მიიღებს ახალ კომენტარებს. თუ მივიღებთ გრძელ კომენტარს, რომელიც ეკრანზე არ ეტევა, მაშინ იგი მოძრავი ხაზის დახმარებით მარჯვენა მხარეს განთავსდება. შემდეგ ამ მხარესვე უნდა შევარჩიოთ ქსელის ტიპი. 3.6 ნახაზზე წარმოდგენილია საბაზო დიალოგურ ქსელის სახელი და ტიპი.



ნახ.3.6. საბაზო დიალოგური ქსელის სახელი და ტიპი

ამისათვის არსებობს სამი ალტერნატიული ღილაკი:

1. ღია ქსელი: ასეთი ტიპის ქსელში მოთხოვნა დგება მოთხოვნათა წყაროდან და ნებისმიერ განსაზღვრულ წერტილში შეუძლია დატოვოს ქსელი;

2. ჩაკეტილი ქსელი: ამ შემთხვევაში ქსელური გრაფი ჩაკეტილია. შეუძლებელია რომელიმე მოთხოვნა დადგეს ქსელში ან მოთხოვნამ დატოვოს ქსელი. აქ ხდება მყარი ცირკულაცია, ქსელში რეგისტრირდება სტაბილურ მოთხოვნათა რიცხვი;

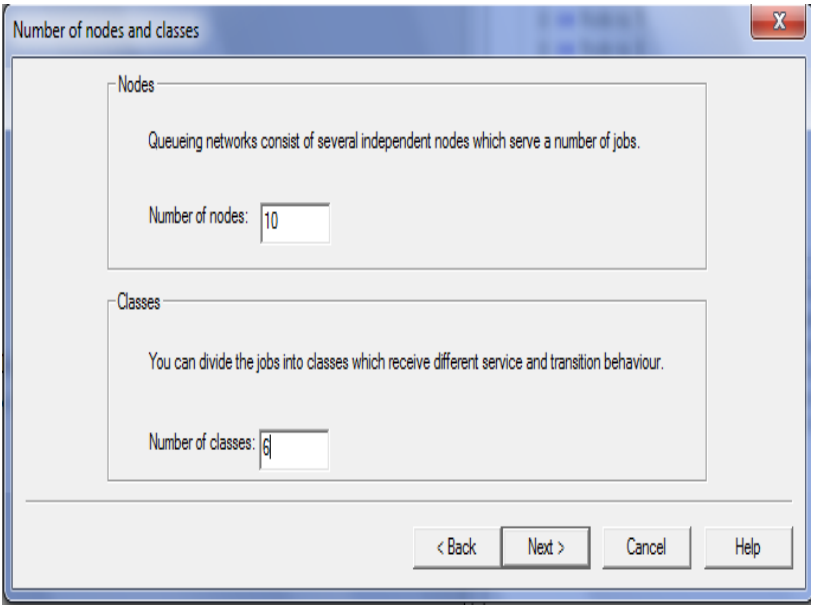
3. შერეული ქსელი: იგი შეიცავს როგორც ღია, ასევე ჩაკეტილ ქსელს.

მას შემდეგ, რაც მომხმარებელი დააფიქსირებს მისთვის სასურველ მდგომარეობას, დიალოგური ფანჯრიდან ირჩევს ღილაკს: Next. შემდეგ გადამრთველი დაიწყებს მუშაობას, რომელიც აფიქსირებს კვანძების რაოდენობას და განსაზღვრავს მოთხოვნათა კლასს.

3.4 კვანძების რაოდენობა და მოთხოვნათა კლასები

დიალოგური ფანჯრის ამ გვერდზე შეიძლება ქსელში განისაზღვროს კვანძების რაოდენობა და შესაბამისი მონაცემთა კლასები (ნახ.3.7). თითოეული ქსელისათვის მითითებული უნდა იყოს კვანძების რაოდენობა და მონაცემთა კლასი.

მომხმარებელს ასევე შეუძლია უკან გადაფურცლოს ფანჯარა და ნებისმიერ დროს შეიტანოს რომელიმე კვანძი ან მოთხოვნათა კლასი ქსელის შესავსებად, ასევე შესაძლებელია კვანძის და მოთხოვნათა კლასის ამოგდება.



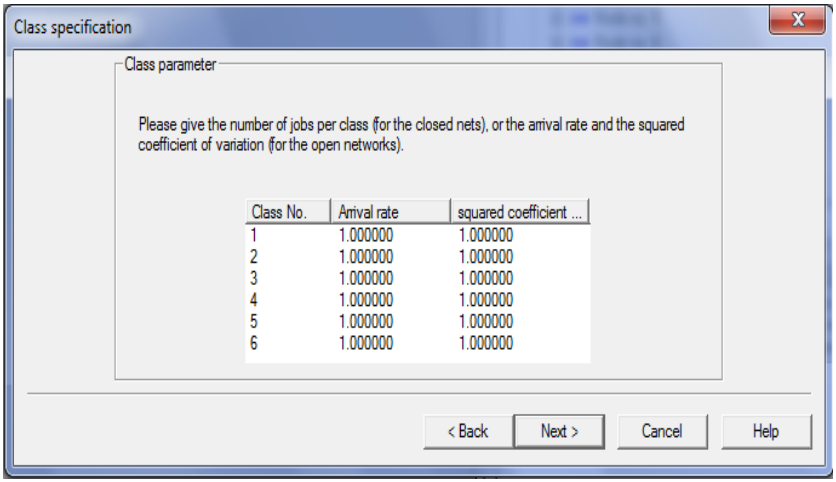
ნახ.3.7. კვანძების და კლასების რაოდენობა

3.6. კლასთა პარამეტრები

მონაცემთა კლასი, დინამიკური მოდელირებისას მუშავდება გამომთვლელ მანქანაში, რათა შესაძლებელი გახდეს მომხმარებლისთვის მნიშვნელოვანი, მოდელირების პროცესში წარმოქმნილი განსხვავებების ზუსტი აღწერა. მომხმარებლის ყოველი მოქმედება, რომელიც მნიშვნელოვნად დაკავშირებულია გამოთვლითი სისტემის სიმძლავრესთან, უნდა შეესაბამებოდეს მონაცემთა კლასის მოდელს.

ყოველ დახურულ კლასში მოთხოვნათა რაოდენობა უნდა იყოს მკაცრად განსაზღვრული, სადაც მოხდება ამ მოთხოვნათა ცირკულაცია. აგრეთვე მნიშვნელოვანია ის ფაქტი, რომ ყოველ დახურულ კლასს, როგორც მინიმუმ უნდა ახლდეს ერთი მოთხოვნა მაინც, რათა საჭიროების შემთხვევაში სწრაფად მოხდეს ჩანაცვლება და არ დაირღვეს სისტემის მუშაობის პრინციპი. ასეთი სახის მოთხოვნას უწოდებენ შემავსებელს (აღნიშვნა k).

ღია კლასში კი პირიქით, საჭიროა მივუთითოთ მოთხოვნის შემოსვლის ინტენსივობა, რომელიც λ ასოთი აღინიშნება. იგი შეიძლება წარმოვადგინოთ როგორც მოთხოვნათა საშუალო რაოდენობა დროის ერთეულში. 3.8 ნახაზზე მოცემულია კლასთა პარამეტრების ასახვის ფანჯარა.



ნახ.3.8. კლასთა პარამეტრები

ყოველივე ამის შემდეგ მომხმარებელი, როგორც კი სპეციფიკაციის მიხედვით პირველ გვერდზე შეარჩევს თუ როგორი სახისაა მისთვის სასურველი კლასი (ღია ან ჩაკეტილი), შეუძლია მიუთითოს მოთხოვნათა რაოდენობა და მათი შემოსვლის ინტენსივობა. ხოლო თუ მომხმარებელი გამოიყენებს შერეულ კლასს, მაშინ მას საშუალება ექნება გამოიყენოს შედარებით იოლად მოდიფიცირებული გვერდი.

მომხმარებელს თაგვის დახმარებით მარტივად შეუძლია შეარჩიოს მისთვის სასურველი კლასი. შერჩეულ მოთხოვნათა კლასის მიხედვით კი განისაზღვრება შესაბამისი მომსახურე ობიექტის მუშაობის პრინციპი.

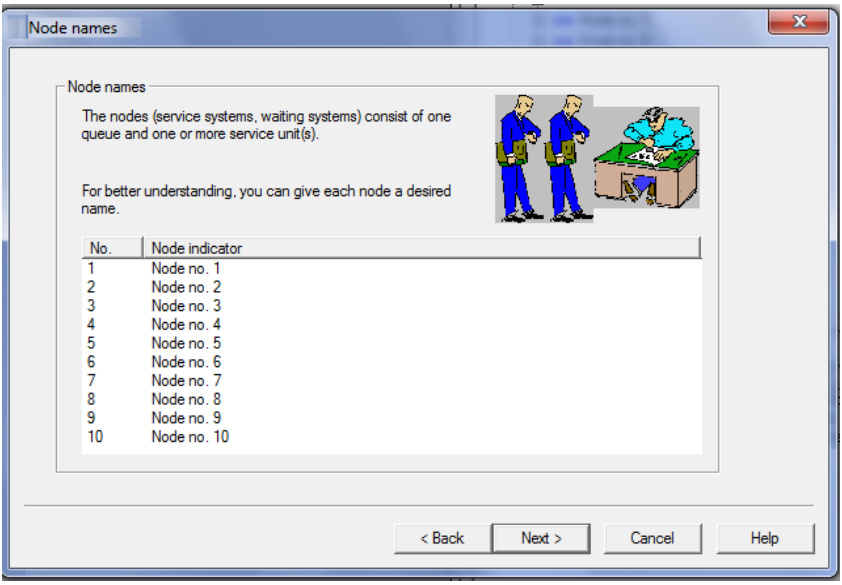
3.7. კვანძის აღწერა და ტიპები

მოდელი უფრო მეტად თვალსაჩინო, რომ გახდეს, ამისათვის მომხმარებელს შეუძლია ყოველი კვანძი ანბანურ-ციფრული სიმბოლოებით აღწეროს. ასეთი აღწერის შედეგად მიღებული კვანძის დასახელება არ უნდა იყოს ორმოც სიმბოლოზე მეტი.

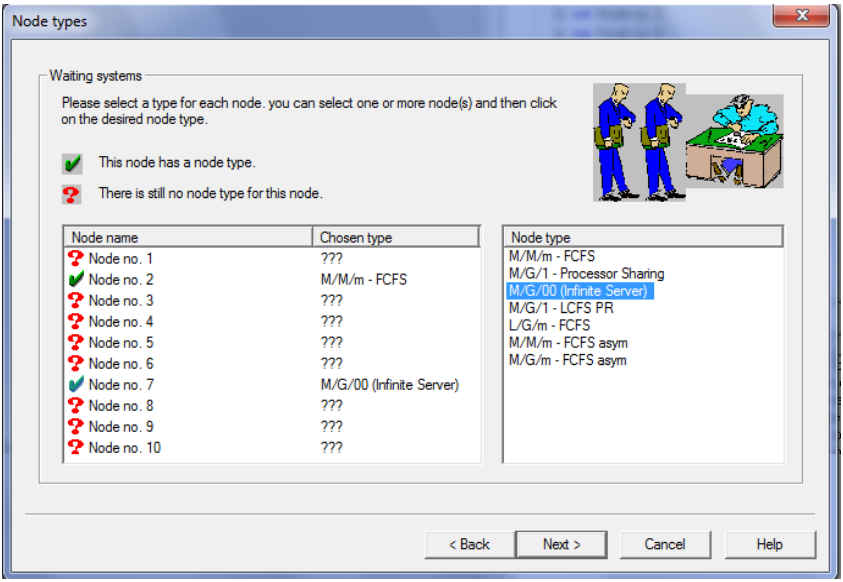
დასაწყისში WinPepsy–ს გვერდებზე ყოველი კვანძისთვის იქნება წარმოდგენილი სახელი „Node names. <nr>“.

მომხმარებელს შეუძლია შეხედულებისამებრ შეცვალოს ნებისმიერი სახელი. ასევე განსხვავებულ კვანძებს შეიძლება დაერქვას იდენტური სახელები. 3.9 ნახაზზე წარმოდგენილია დიალოგური ფანჯარა კვანძების დასახელებით.

კვანძი შედგება მოთხოვნათა რიგისგან და ერთი ან რამდენიმე მომსახურე ობიექტისგან. თუ შემოსულ მოთხოვნას ყველა მომსახურე ობიექტი დაკავებული დახვდება, მაშინ ის დგება რიგში და ელოდება მომსახურებას. ყოველი კვანძისათვის ქსელმა უნდა დააფიქსიროს კვანძის ტიპი, რაც მდგომარეობს კვანძის ტიპის უნიფიცირებულ აღწერაში, რომლის შესრულებაც შესაძლებელია „Noden type“ სისტემის დახმარებით (ნახ.3.10).



ნახ.3.9. დიალოგურ ფანჯარაში კვანძების დასახელება



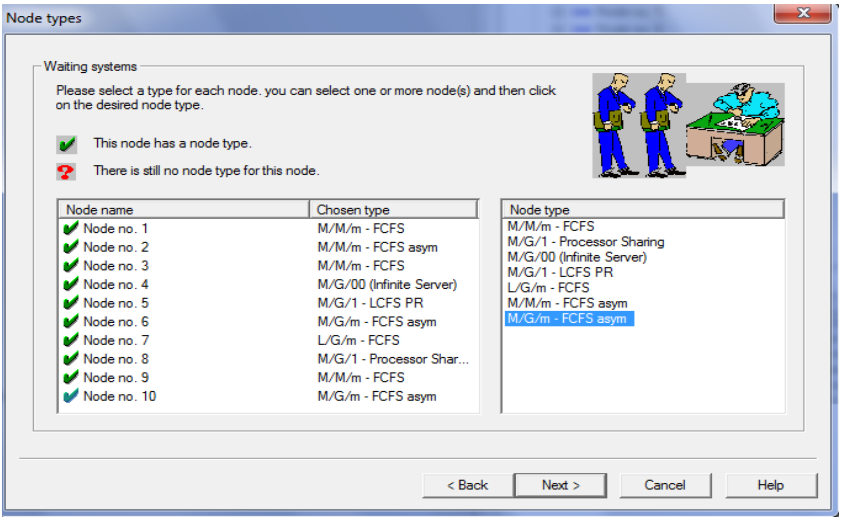
ნახ.3.10. დიალოგურ ფანჯარაში გამოსახული კვნძის ტიპი

გამოიყენება აგრეთვე დ. კენდალის ნოტაცია (რიგების დისციპლინა) – A/B/m, სადაც A აღნიშნავს ინფორმაციის განაწილების საშუალო დროს, B არის არსებულ რიგში მომსახურების განაწილება, ხოლო m – იდენტური მომსახურე ობიექტების რაოდენობა [56]. განაწილების სისტემები შემდეგი სახისაა: ექსპონენციალური (M-სიმბოლო, მარკოვული პროცესი) და საერთო განაწილების სისტემა (G-სიმბოლო), რომელებიც ამ შემთხვევაში აქტიურად გამოიყენება.

არსებობს რამდენიმე სახის რიგების დისციპლინა:

- FCFC (First-Come-First-Served) პირველი მოვიდა პირველი მომსახურდა (იგივეა FIFO - რიგი);
- LCFS (Last-Come-First-Served) ბოლო მოვიდა პირველი მომსახურდა (იგივეა LIFO - სტეკი);;
- PS (Processor-Sharing) პროცესორული დროის განაწილება;
- IS (Infinite-Server) უსასრულო სერვერი.

მუშაობის პროცესში ერთმანეთს უკავშირდება შერჩეული და გამოტოვებული დისციპლინები და პერსონალთან მომსახურე ასიმეტრიული სისტემები.



ნახ.3.11. დიალოგურ ფანჯარაში გამოსახული კვნძის ტიპები

შემოთავაზებული კვანძების სახეობები მომხმარებელს ეკრანზე უჩვენებს ორი სახის ჩამონათვალს. მარცხენა ჩამონათვალში არის ქსელში არსებული ტიპები და იმ კვანძების სახეობები, რომლებიც ამ ტიპებს შეესაბამება, ხოლო მარჯვენა მხარეს მოთავსებულია ჩამონათვალი, თუ რომელი სახეობაა გამოყენებული.

პირველ რიგში მომხმარებელმა მაუსის კურსორის დახმარებით უნდა მონიშნოს ერთი ან რამდენიმე კვანძი, რომელსაც ის განსაზღვრულ სახეობასთან მიაერთებს. ერთდროულად შესაძლებელია შევარჩიოთ რამდენიმე კვანძი, შემდეგ ერთმანეთის მიყოლებით კვანძები მოახდენს მარკირებას, ან კიდევ მაუსის კურსორით მოვნიშნავთ ჯერ პირველ სასურველ კვანძს, მერე დავაჭერთ ერთროულად ბოლო კვანძის ტიპს და Shift ღილაკს, რის შედეგადაც მოხდება ყველა კვანძის ერთდროული მონიშვნა.

Ctrl კლავიშის დახმარებით შეიძლება მოვახდინოთ კვანძების მოწესრიგება. რამდენიმე კვანძის ტიპის შერჩევის შემდეგ, მარჯვენა ველში მაუსის კურსორით მოვნიშნავთ ტიპის შესაბამის კვანძის დასახელებას, რის შედეგადაც ჩამონათვალში გამონათდება შერჩეული კვანძის სახეობა და პატარა მწვანე მარყუჟი.

კვანძი, რომელიც სახეობის შერჩევის გარეშე დარჩება, გვერდით გაუჩნდება პატარა წითელი კითხვის ნიშანი, რაც დაგვეხმარება, რომ შემდეგ ფურცელზე გადასვლისას აღნიშნულ კვანძს შევურჩიოთ შესაბამისი ტიპი.

3.8. ვარიაციის კოეფიციენტი

შემთხვევითი სიდიდის მდებარეობის მახასიათებლების – საშუალო ტიპიური მნიშვნელობათა გარდა, აგრეთვე გამოიყენება კიდევ სხვა მახასიათებლები, რომელთაგან ყოველი მათგანი აღწერს განაწილების ამა თუ იმ თვისებას. ასეთ მახასიათებლად ყველაზე ხშირად გამოიყენება ვარიაციის კოეფიციენტი.

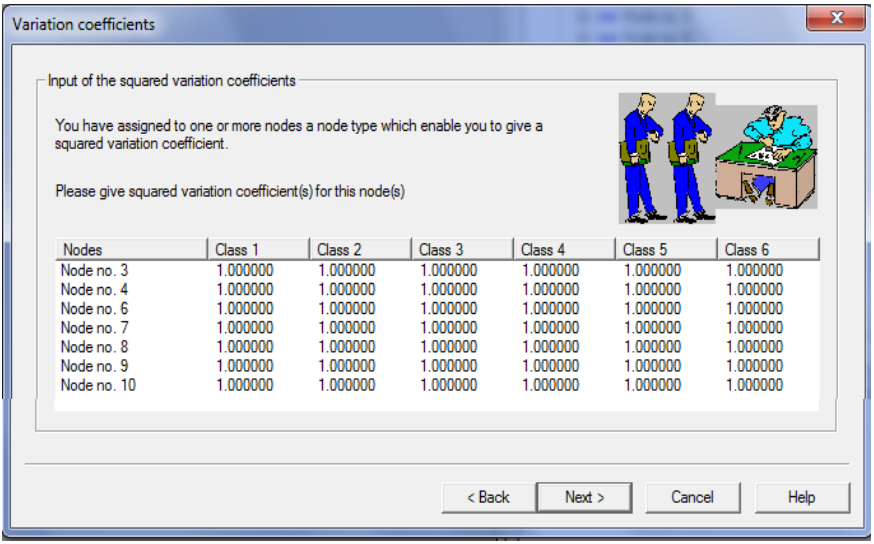
რამდენიმე ტიპიურად განსხვავებული კვანძის შემთხვევაში, მომხმარებელს შეუძლია თითოეული შემთხვევითად შერჩეული კვანძისთვის მიუთითოს ვარიაციის კოეფიციენტი.

ვარიაციის კოეფიციენტი გამოისახება საშუალო კვადრატული გადახრით, რადგან შემთხვევითი სიდიდის დისპერსიას შემთხვევითი სიდიდის კვადრატის განზომილება აქვს.

გაბნევის თვალსაჩინო დახასიათებისათვის უფრო მოხერხებულია ვისარგებლოთ სიდიდით, რომლის განზომილება ემთხვევა შემთხვევითი სიდიდის განზომილებას. ამისათვის დისპერსიიდან იღებენ კვადრატულ ფესვს. მიღებულ სიდიდეს უწოდებენ შემთხვევითი X სიდიდის საშუალო კვადრატულ გადახრას:

$$C_x = \frac{\sigma_x}{X_{\text{ვარ}}}$$

სადაც σ_X აღნიშნავს საშუალო კვადრატული გადახრას, ხოლო $X_{\text{ვარ.}}$ – არის შემთხვევითი სიდიდე. მომხმარებელს შეუძლია ყოველი კლასისათვის მიუთითოს განსხვავებული მნიშვნელობის ვარიაციის კოეფიციენტი.



ნახ.3.12. ვარიაციის კოეფიციენტის შერჩევა

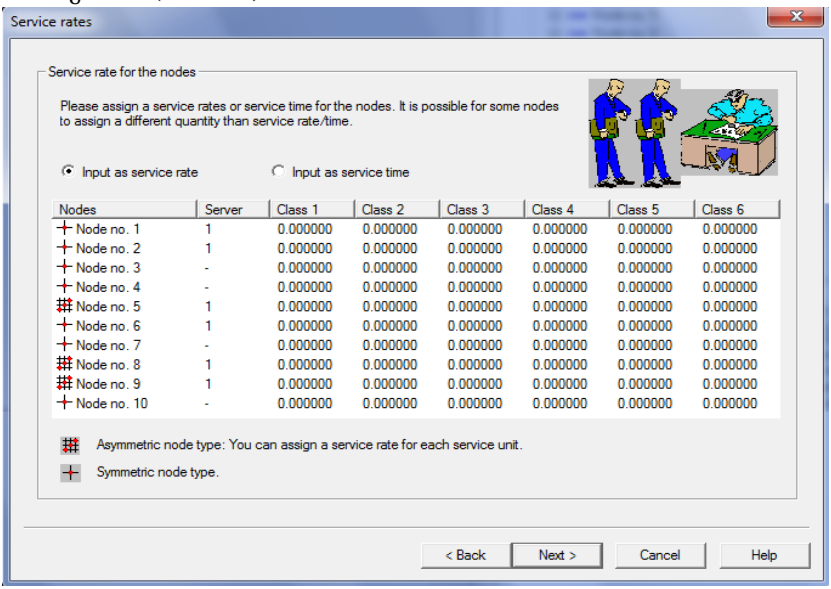
3.9. მომსახურების დრო და მოსახურების ნორმები

როგორც უკვე აღვნიშნეთ, კვანძში შემოსული თითოეული მოთხოვნა ხასიათდება მომსახურეობის დროით – $t_მ$. იგი ყველა ცალკეული მოთხოვნის შემთხვევაში განისაზღვრება მოთხოვნის ხასიათის მიხედვით, რომელიც d ასოთო აღინიშნება. იგი მიუთითებს მომსახურე პერსონალის საჭირო რიცხვს და სამუშაოს შესრულების V – სიჩქარეს, საიდანაც: $t_მ = \frac{d}{v}$.

აქედან შეიძლება საშუალო მომსახურეობის დროის გათვლა:

$$\mu = \frac{1}{t_მ}$$

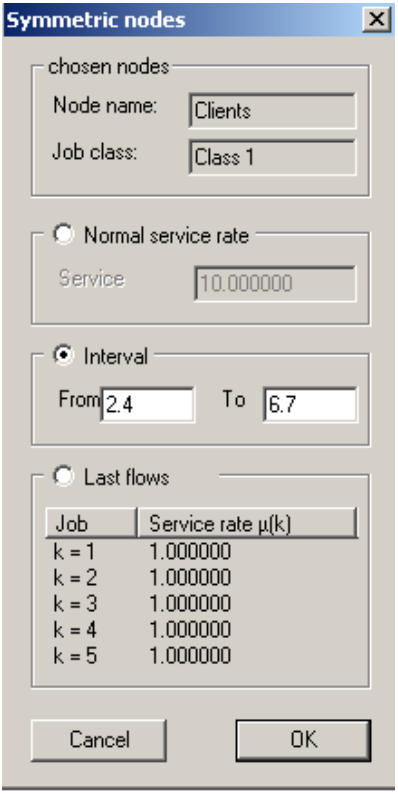
მომხმარებელმა პირველ რიგში უნდა განსაზღვროს თუ მომსახურების პროცესის როგორი ფორმის შერჩევა უნდა. მომხმარებელს შესაბამისად შეუძლია შეარჩიოს ფანჯრის ზედა ნაწილში განაცხადი „მომსახურების დროით“ ან „მომსახურების ნორმებით“ (ნახ.3.13).



ნახ.3.13. განაცხადის შერჩევა

მოცემულ ცხრილში შემოთავაზებულია თითოეული კვანძისათვის შესაბამისად იდენტური მომსახურე ობიექტი და მომსახურების დრო. მომსახურების ნორმები კი მოცემულია ყველა კლასისათვის. სერვერების რაოდენობა შესაძლებელია შევიტანოთ ასიმეტრიული და „multiple server“ (მრავალჯერადი მომსახურება: კვანძს შეუძლია რამდენიმეჯერ გამოიყენოს ერთი და იგივე ობიექტი) შესაბამისი კვანძებისათვის ფანჯრის ქვედა ნაწილის დახმარებით.

მომსახურების დროს და მომსახურების ნორმების ან სერვერების რაოდენობის, შესაბამისი ველი მომხმარებელს შეუძლია ფანჯარაში არსებული ცხრილიდან მაუსის დახმარებით გამოიტანოს ეკრანზე. შესაბამისად მივიღებთ ფანჯარას, სადაც კვანძების მოსახურების პროცესისთვის მოთავსებული იქნება სიმეტრიული და ასიმეტრიული დიალოგური ფანჯრები (ნახ.3.14-ა,ბ). ორივე შემთხვევაში ფანჯრის ველის ზედა ნაწილში მოთავსებულია კვანძის სახელი და კლასი, რაც შესაძლებელს ხდის შემდგომ პროცესზე მომსახურების დროის და მომსახურების ნორმების პროცესის სწორად შერჩევას.



ნახ.3.14-ა. სიმეტრიული კვანძები

სიმეტრიული კვანძის შემთხვევაში მომსახურების პროცესი შეიძლება ჩამოვაყალიბოთ სამი სახის სპეციფიკაციის მიხედვით:

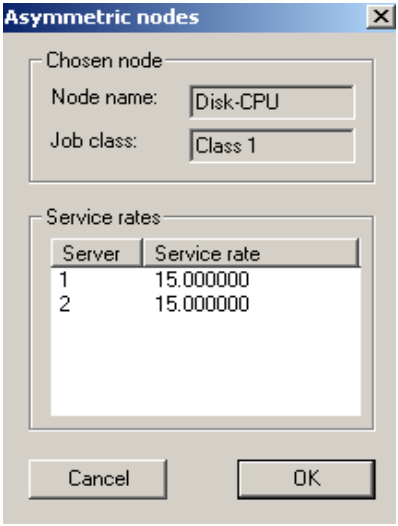
- ნორმალური მომსახურების დროით და მომსახურების ნორმებით;
- მომსახურება ინტერვალებით;
- მომსახურება დატვირთვით.

მომხმარებელმა პირველ რიგში უნდა განსაზღვროს თუ მომსახურების რომელ სახეს შეარჩევს. ისინი განთავსებულია დიალოგური ფანჯრის შესაბამის ველებში: „Normal", „Interval" ან „Last flows".

ამ პროცესის რეალიზება შესაძლებელია მხოლოდ იმ შემთხვევაში, თუ გვეცოდინება შემოსული მოთხოვნების მომსახურების სპეციფიკაცია.

თუ „Last flows" (დატვირთვით) ველს შევარჩევთ, მაშინ მითითებულ უნდა იყოს თითოეული შემოსული მოთხოვნისთვის მომსახურების დრო. შესაძლებელია მომსახურების პროცესის შეცვლა მომდევნო კვანძებისა და კლასებისათვის.

როგორც აღვნიშნეთ, გვაქვს ასიმეტრიული კვანძის ამორჩევის საშუალებაც (ნახ.3.14-ბ). გამონათდება დიალოგური ფანჯარა, სადაც მომხმარებელს შეუძლია სერვერიებისთვის შეარჩიოს მომსახურების დრო და მომსახურების ნორმები.

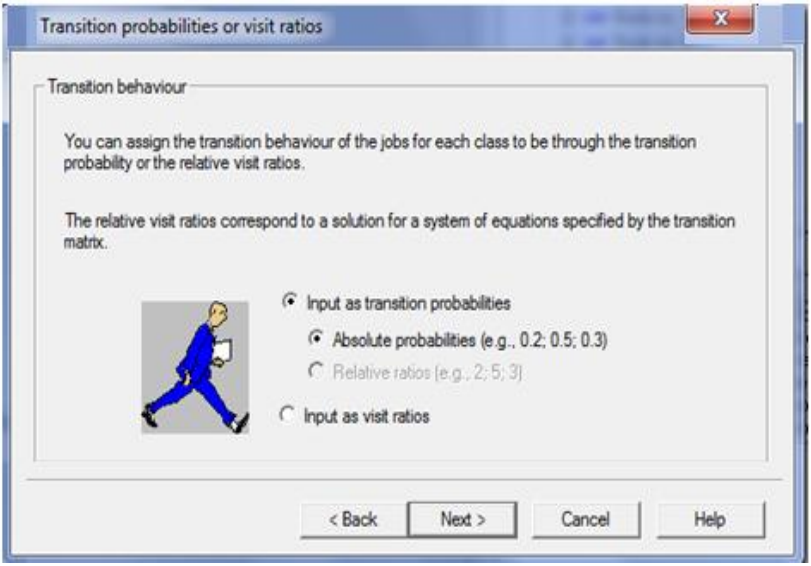


ნახ.3.14-ბ. ასიმეტრიული კვანძის მომსახურების დრო

3.10. გადასასვლელის ხასიათი და ალბათობის განსაზღვრა

ამ გვერდზე მომხმარებელს შეუძლია შეარჩიოს ქსელში მოთხოვნათა მოხვედრის ხასიათი, მოცემული ასორტიმენტის ორი პროცესის დახმარებით:

- გადასასვლელის ალბათობის დადგენა, რომლის მიხედვითაც განსაზღვრული კვანძიდან შემოსული მოთხოვნა რომელიმე განსაზღვრულ კლასში შეიძლება შევცვალოთ სხვა კვანძით ან კლასით;
- მოთხოვნათა შემოსვლის სიხშირე. იგი საშუალებას იძლევა განისაზღვროს მოთხოვნათა შემოსვლის საშუალო სიხშირე რომელიმე კონკრეტულ კვანძში.



ნახ.3.15. გადასასვლელის ხასიათის შერჩევა

მომხმარებელმა რომ მოახდინოს კვანძების გადასავლელთა სპეციფიკაცია, უნდა გაითვალისწინოს შემდეგი ფაქტორები:

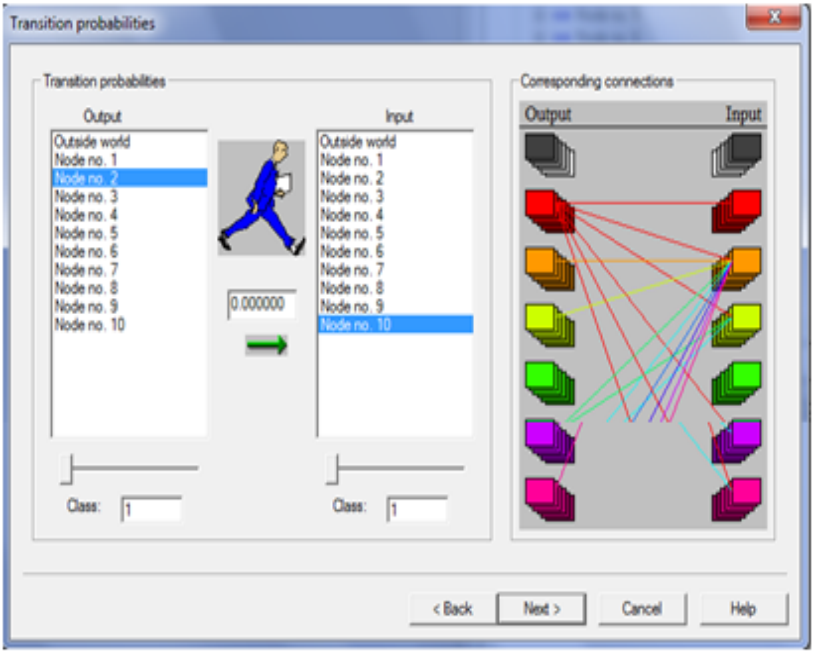
1. გამავალი კვანძის განსაზღვრა. გამავალია ის კვანძი, სადაც მოთხოვნა ტოვებს რიგს, რათა გადავიდეს სხვა კვანძში. იგი არსებობს ფანჯარაში ჩამონათვალ კვანძების ნუსხაში;

2. შემავალი კვანძი. შემავალია ის მიზნობრივი კვანძი, სადაც მოთხოვნა იცვლის სახეს. მიზნობრივი კვანძის კლასის ანალოგი შეიძლება მოვძებნოთ გამავალ კვანძშიც;

3. გადასასვლელის ალბათობის მაჩვენებელი. თუ კურსორი დაყენებულია მონაცემთა შეტანის ველში ორივე ჩამონათვალს შორის, შესაძლებელია გადსასვლელის ალბათობა გამოვთვალოთ ორივე სახის ჩამონათვალ კვანძებისთვის.

საბოლოოდ შესაძლებელია კვანძის გადასასვლელის შეცვლა მანამ, სანამ გადასავლელზე ალბათობა არ იქნება 0-ის ტოლო ან კიდევ არ მივიღებთ ჩვენთვის მისაღებ დასაშვებ ალბათობას.

ფანჯრის მარჯვენა მხარეს წარმოდგენილია არსებული კვანძების გადასასვლელების ვიზუალური მხარე (ნახ.3.16).

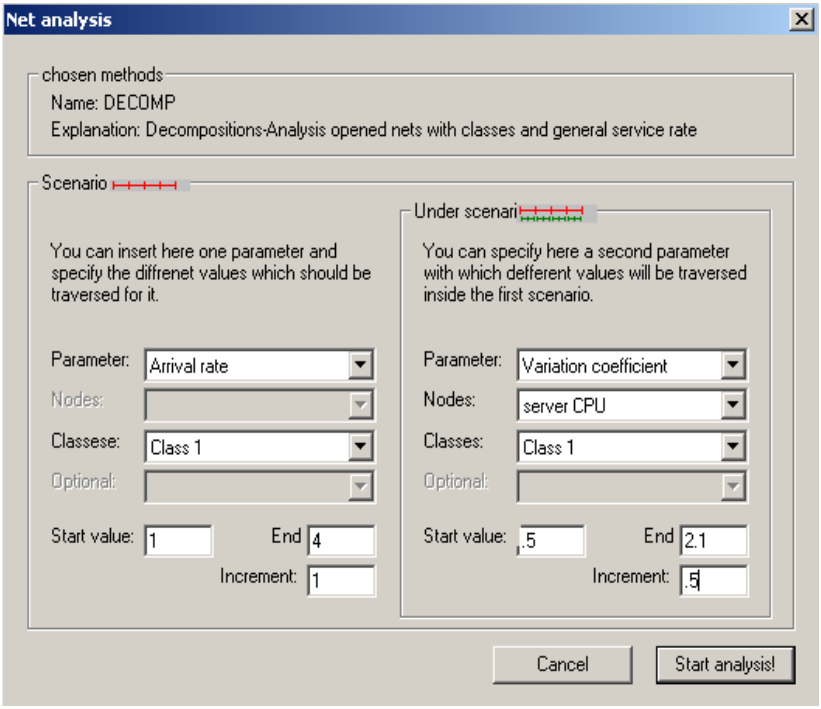


ნახ.3.16 გადასავლელის ალბათობა

თითოეული კვანძი წარმოდგენილია პატარა მართკუთხედის სახით. კვანძების გადასასვლელები სიმბოლიზირდება ფერადი ხაზებით.

3.11. მოთხოვნათა ნაკადის შემოსვლის სიხშირე

თუ მომხმარებელს სურს გადასასვლელის ხასიათის განსაზღვრა მოთხოვნათა ნაკადის შესასვლელის სიხშირის სპეციფიკაციის მიხედვით, მაშინ უნდა გადავიდეს შემდეგ გვერდზე, სადაც ასახულია კვანძებსა და კლასებს შორის კავშირი



(ნახ.3.17).

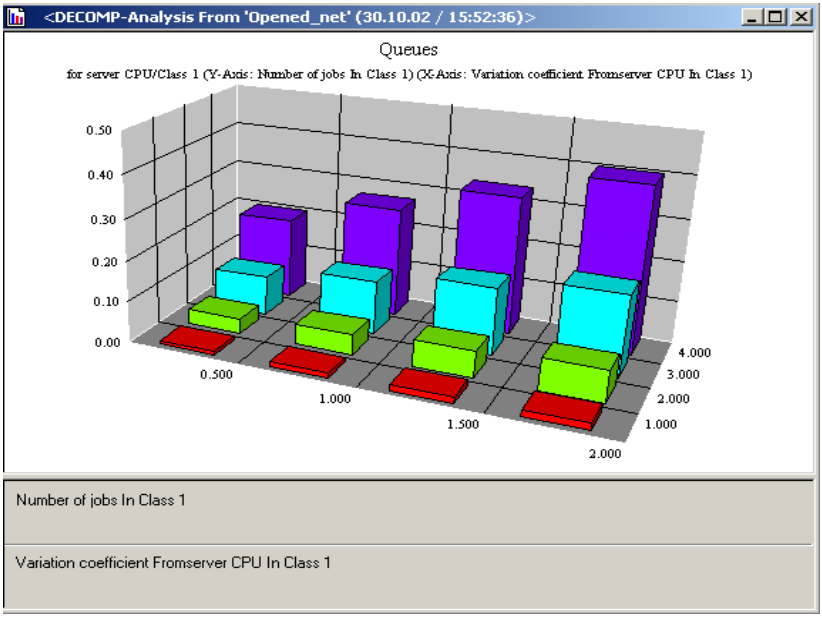
ნახ.3.17. ნაკადის შემოსვლის ანალიზი

მოთხოვნათა ნაკადის შესასვლელზე არსებული მოთხოვნათა სიხშირესა λ_iდა ქსელის სიხშირეს λ შორის არის შემდეგნაირი დამოკიდებულება:

$$e_i = \frac{\lambda_i}{\lambda}$$

3.12. მონაცემთა ანალიზი და გრაფიკული ასახვა

მას შემდეგ, რაც გადასასვლელის ხასიათის სპეციფიკაციის განსაზღვრა მოხდა, შეიძლება ვთქვათ, რომ ქსელის აღრიცხვა დამთავრებულია. შემოწმების შედეგად მიღებულია, რომ ყველა დაყენებული მოთხოვნა სტრუქტურულად სრულყოფილად არის ჩამოყალიბებული. მომხმარებელს შეუძლია მოითხოვოს დეტალური ინფორმაცია და ვიზუალური შედეგი (ნახ.3.18).



ნახ.3.18. მოთხოვნის საფუძველზე აგებული ქსელის ასახვის საბოლოო ვიზუალური შედეგი

თუ ყველა მონაცემი სწორად გვაქს შეტანილი, მაშინ დიალოგურ ფანჯარაში შეიძლება დავასრულოთ მუშაობა ღილაკით „Start analysis“, შემდეგ ეტაპზე იწყება მონაცემთა ავტომატური გადამოწმება, ყველა იმ მეთოდის დახმარებით, რაც გამოყენებული იყო ქსელში.

IV თავი:

პეტრის ქსელები – მოდელირების და ანალიზის ინსტრუმენტული საშუალება

4.1 პეტრის ქსელების თეორიული საფუძვლები

პეტრის ქსელების ისტორია 1962 წლიდან იწყება, როცა გერმანელმა ინჟინერმა, კარლ ადამ პეტრიმ დარმშტადტის ტექნიკურ უნივერსიტეტში დაიცვა სადოქტორო დისერტაცია თემაზე ,,კომუნიკაცია ავტომატებით''.

ამ ნაშრომში მან პირველად ჩამოაყალიბა და დაასაბუთა იდეა ორი განსხვავებული ტიპის კვანძებითა და მათი დამაკავშირებელი მიმართული რკალებით აგებული მუშა ქსელების შესახებ, რომლებიც ერთი მოდელის ფარგლებში გააერთიანებდა კონკრეტულ და აბსტრაქტულ პროცესებს და მონაცემებს.

სახელი ,,პეტრის ქსელები'' თეორიამ მოგვიანებით მიიღო ავტორის პატივსაცემად. თავისი თეორიის საფუძვლად პეტრიმ სასრული ავტომატების, სიმრავლეთა და გრაფების თეორიის ელემენტები გამოიყენა [19,45,46].

ერთის მხრივ პეტრის ქსელები ფართოვდება თეორიულად, სულ უფრო მძლავრი ხდება მისი მათემატიკური აპარატი, იქმნება ახალი თეორიული კლასები, მეორეს მხრივ მატულობს პეტრის ქსელების პრაქტიკული გამოყენების არეალი და სიხშირე ინფორმატიკის აქტუალურ მიმართულებებთან კავშირში, რაც სქემატურად ნაჩვენებია 4.0 ნახაზზე.

ნახ.4.0. პეტრის ქსელები და მათი გარემოცვა

4.1.1. სიმრავლეები

სიმრავლეთა თეორია პეტრის ქსელების ერთერთი ბაზისია. განვიხილოთ მოკლედ მისი ძირითადი ელემენტები.

საწყისი აღნიშვნები:

N = {0,1,...} – ნატურალურ რიცხვთა სიმრავლე;

Z = {…,-1,0,1,…} – მთელ რიცხვთა სიმრავლე

Boolean = {true,false} – ბულის სიმრავლე

სიმრავლე არაერთგვაროვან ობიექტთა ერთობ-ლიობაა. მათ სიმრავლის **ელემენტები** ეწოდება.

a არის **A**-სიმრავლის ელემენტი, თუკი ფლობს თვისებას **a∈A** („მიეკუთვნება“). სიმრავლე მოიცემა ელემენტთა ჩამონათვალით $\mathbf{A = \{a_1, a_2, ..., a_n\}}$, ან გარკვეულ **p(a)** ფუნქციაზე დაყრდნობით, რომლის შედეგი სიმრავლის ელემენტისთვის აუცილებელ პირობას აკმაყოფილებს:

$$\mathbf{A = \{a \mid p(a)\}.}$$

ცარიელი სიმრავლე **Ø**-სიმბოლოთი აღინიშნება და გამოისახება პირობით **Ø = {a | a≠a},** რადგან პირობა **a≠a** ყოველთვის მცდარია.

სიმრავლეთა თეორიაში განისაზღვრება შემდეგი ძირითადი დამოკიდებულებები და ოპერაციები: **ქვესიმრავლე (A⊆B), ჭეშმარიტი ქვესიმრავლე (A⊂B), გაერთიანება (A∪B), თანაკვეთა (A∩B), სხვაობა (A\B),** სადაც:

A⊆B, როცა ნებისმიერი **a∈A**–თვის მართებულია **a∈B**

A⊂B, როცა **A⊆B** და **A≠B**

A∪B = { a | a∈A ან **a∈B }**

A∩B = { a | a∈A და **a∈B }**

A\B = { a | a∈A და **a∉B }**

ცარიელი სიმრავლე ნებისმიერი არაცარიელი სიმრავლის ქვესიმრავლეა: **Ø⊂A. A** და **B** სიმრავლეებს **განცალკევებული სიმრავლეები** ეწოდება, თუ **A∩B=Ø.**

სიმრავლე შეიძლება შეიცავდეს ელემეტებს, რომლებიც თავადაა სიმრავლეები**. A**-სიმრავლის ყველა შესაძლო ქვესიმრავლეთა სიმრავლე **Π(A)**-თი აღინიშნება, ნატურალურ

რიცხვთა სიმრავლე 0-ის ჩათვლით – **N**-ით, ლოგიკურ მნიშვნელობათა (ჭეშმარიტი ან მცდარი) სიმრავლე – **B**-თი.

$A_1,A_2,...,A_n$ (**$n \in N$**) სასრულ სიმრავლეთა **პროდუქტი (დეკარტული ნამრავლი)** განისაზღვრება შემდეგნაირად:

$$A_1xA_2x...xA_n = \{ (a_1,a_2,...,a_n) \mid a_i \in A_i \ i=1,...,N \}$$

პროდუქტის ცალკეულ ელემენტს **n**-კორტეჟი ეწოდება. ყოველი **i**-სთვის, სადაც **$1<i<n$**, **a_i**-ს კორტეჟის **i**-ური ელემენტი ეწოდება (**$a_1,...,a_n$**). **წყვილი** განისაზღვრება, როგორც **n**-კორტეჟის კერძო შემთხვევა, 2-კორტეჟი (ბინარული).

თუ ყველა სიმრავლე **$A_1=A_2=...A_n=A$** მსგავსია, პროდუქტი ჩაიწერება **A^n** სახით. გარდა ამისა, **$A^1 = A$** და **$A^0 = Ø$.**

4.1.2. მულტისიმრავლეები (კომპლექტები)

მულტისიმრავლე განისაზღვრება სიმრავლეში ერთი და იმავე ელემენტის რამდენჯერმე ასახვისთვის. მაგალითად, პეტრის ქსელის პოზიციაში რამდენიმე მსგავსი მარკერის აღსაწერად.

მულტისიმრავლე B არაცარიელ საბაზო A სიმრავლეზე, ეწოდება ფუნქციას:

$$B:A \rightarrow N$$

სადაც საბაზო **A** სიმრავლის ყოველი **$a \in A$** ელემენტის **სიხშირე B** მულტისიმრავლეში აისახება ფორმატით **B(a).** სიხშირის სიდიდე შეიძლება 0-ის ტოლიც იყოს. სიმრავლე მულტისიმრავლის სპეციალური შემთხვევაა, სადაც სიხშირის მნიშვნელობებია 0 ან 1.

მულტისიმრავლის ასახვის გაფართოებული ფორმა შემდეგია: **[a,a,...,a,b,...,b,...],** სადაც ყოველი ელემენტი თავისი სიხშირის მიხედვით მეორდება.

მულტისიმრავლეთა სიმრავლე საბაზო **A** სიმრავლეზე აღინიშნება **μA**-თი. მულტისიმრავლე ასევე შეიძლება გამოისახოს სიმბოლური ჯამის სახით, რომელიც **$a \in A$** ელემენტის სიხშირეს და სახელს შეიცავს:

$$B = \sum_{a \in A} B(a)a$$

თუ **B(a) = 1,** მაშინ ჯამურ ასახვაში იგი საერთოდ გამოიტოვება და იწერება მხოლოდ **a**.

B∈μA მულტისიმრავლეში**, a∈A** ელემენტს ეწოდება **B**-ს წევრი და ჩაიწერება **a∈B,** თუ **B(a)>0** და პირიქით, თუ **B(a)<0,** მაშინ **a∉B.** ცარიელი მულტისიმრავლე ∅ წევრებს არ შეიცავს: **∀a∈A, ∅(a)=0.**

მულტისიმრავლის **სიმძლავრე (კარდინალურობა)** მისი ყველა ელემენტის **სიხშირეთა ჯამს** ეწოდება და განისაზღვება შემდეგნაირად:

$$|B| = \sum_{a \in A} B(a)$$

თუ **|B|** სასრულია, მაშინ მულტისიმრავლე **B**-ს **სასრული მულტისიმრავლე** ეწოდება.

ორი მულტისიმრავლე, B_1 და B_2 **ტოლია ($B_1=B_2$),** თუ **$\forall a \in A, B_1(a)= B_2(a)$.**

B_1 ნაკლებია ან ტოლია B_2-ის (ანუ B_2 მოიცავს B_1-ს) თუ **$\forall a \in A, B_1(a) \leq B_2(a)$.**

მულტისიმრავლეებზე ძირითად ოპერაციებს წარმოადგენს:

შეკრება: **$B=B_1+B_2$**, თუ **$\forall a \in A, B(a) = B_1(a) + B_2(a)$**

გამოკლება: $B=B_1-B_2$, თუ **$\forall a \in A, ((B_1(a) \geq B_2(a)) \wedge ((B(a) = B_1(a) - B_2(a))$**

სკალარული ნამრავლი: მულტისიმრავლის **$B_1 \in \mu A$** და ნატურალური რიცხვის **$n \in N$** სკალარული ნამრავლი განისაზღვრება როგორც **$B=n*B_1$,** თუ **$\forall a \in A, B(a) = n*B_1(a)$,** სადაც **"*"** არითმეტიკული გამრავლების ოპერაციაა**.**

4.1.3. პეტრის ქსელების ძირითადი ცნებები

პეტრის ქსელების სტანდარტიზაციის პროცედურა ეგრეთ წოდებული პროექტ 15909-ის ფარგლებში 1995 წლიდან მიმდინარეობს. მასში მონაწილეობენ ისეთი ავტორიტეტული ორგანიზაციები, როგორიცაა სტანდარტიზაციის საერთაშორისო ორგანიზაცია (ISO), ინფორმაციულ ტექნოლოგიათა ერთიანი ტექნიკური კომიტეტი (JTC1) და საერთაშორისო

ელექტროტექნიკური კომისია (IEC). პროექტი 3 ნაწილისგან შედგება:

1. მაღალი დონის პეტრის ქსელები – კონცეფცია, განსაზღვრებები და გრაფიკული ნოტაცია;

2. პეტრის ქსელების გაცვლითი ფორმატი, სავარაუდოდ XML-ის ბაზაზე;

3. მოდულური პეტრის ქსელების კონსტრუქციები – იერარქიულობა, დროითი და სტოქასტური გაფართოებები.

ქვემოთ მოცემულია პეტრის ქსელის **ძირითადი ცნებები:**

საბაზო სიმრავლე (Basis Set). ობიექტების საწყისი სიმრავლე მულტი-სიმრავლეების (კომპლექტების) შესაქმნელად.

მულტისიმრავლე ანუ კომპლექტი (Multiset). ობიექტების ნაკრები, სადაც ერთგვაროვანი ობიექტების განმეორება შესაძლებელია.

მულტისიმრავლის კარდინალურობა (Cardinality). მულტისიმრავლის ელემენტების საერთო რაოდენობა.

პოზიცია (Place). ქსელის ტიპიზებული კვანძი. ქსელის გრაფში წრით ან ელიფსით გამოისახება.

გადასასვლელი (Transition). ქსელის არატიპიზებული კვანძი, რომელიც მართკუთხედით გამოისახება.

რკალი (Arc). ქსელის მიმართული კავშირის ხაზი, რომელიც აერთებს პოზიციებს გადასასვლელებთან (შემავალი რკალი) ან პირიქით (გამომავალი რკალი).

შემავალი პოზიცია (Input Place). გადასასვლელთან შემავალი რკალით შეერთებული პოზიცია.

გამომავალი პოზიცია (Output Place). გადასასვლელთან გამომავალი რკალით შეერთებული პოზიცია.

პოზიციის ტიპი (Place Type). პოზიციასთან დაკავშირებულ მონაცემთა ელემენტების არაცარიელი სიმრავლე.

მარკერი (Marker). პოზიციასთან დაკავშირებული და შესაბამისი პოზიციის ტიპის მონაცემთა ელემენტი.

მარკირება (Marking). ყველა პოზიციაში შემავალ მარკერთა ერთობლიობა.

საწყისი მარკირება (Initial Marking). ყველა პოზიციაში შემავალ მარკერთა ერთობლიობა ქსელის მუშაობის დასაწყისში.

პოზიციის მარკირება (Place Marking). პოზიციაში მოთავსებულ მარკერთა მულტისიმრავლე.

გადასასვლელის გახსნის პირობა (Transition Condition). გადასასვლელთან დაკავშირებული ლოგიკური (ბულის) ტიპის გამოსახულება.

გადასასვლელის გახსნის რეჟიმი (Transition Mode). ცვლადების დაკავშირება გადასასვლელის გახსნის პირობასთან ისე, რომ გადასასვლელის გახსნა ნებადართული გახდეს.

გადასასვლელის გახსნის ნებართვა (Enabling a Transition). გადასასვლელი რომ გაიხსნას, ყოველი პოზიციის მარკირება უნდა აკმაყოფილებდეს მისი და გადასასვლელის დამაკავშირებელი რკალის მოთხოვნას (რკალის გამოსახულება), რაც ნიშნავს, რომ მარკირება შეიცავს მარკერების მინიმუმ იმავე მულტისიმრავლეს, რაც რკალის გამოსახულებაზეა ასახული.

გადასასვლელის გახსნა (Transition Occurence). თუ გადასასვლელის გახსნა ნებადართულია, იგი შეიძლება გაიხსნას. ამ დროს გადასასვლელის ყოველი შემავალი პოზიციიდან მოიხსნება მარკერები გახსნის რეჟიმის შესაბამისად, ხოლო ყოველ გამომავალ პოზიციაში გამომავალი რკალების გამოსახულებათა შესაბამისი მარკერები ჩაემატება. პოზიცია შეიძლება ერთდროულად შემავალი და გამომავალი იყოს (**მარყუჟი**)

გადასასვლელის ცვლადები (Transition Variables). რკალებისა და გადასასვლელის გახსნის პირობაში შემავალი ცვლადების ერთობლიობა.

რკალის ანოტაცია (Arc Annotation). გამოსახულება, რომელიც შეიძლება შეიცავდეს კონსტანტებს, ცვლადებს და ოპერატორებს რკალთან დაკავშირებული პოზიციის ტიპის მულტისიმრავლიდან.

მიღწევადი მარკირება (Reachable Marking). მარკირება, რომელიც მიიღება ქსელის საწყისი მარკირებიდან გადასასვლელთა გარკვეული მიმდევრობის გახსნის შემდეგ.

მიღწევად მარკირებათა სიმრავლე (Reachability Set). საწყისი მარკირებიდან მიღწევად მარკირებათა სიმრავლე თვით საწყისი მარკირების ჩათვლით.

ალგებრა (Algebra). მათემატიკური სტრუქტურა, რომელიც შეიცავს სიმრავლეთა სიმრავლეს და ფუნქციათა სიმრავლეს, რომლებიც ამ სიმრავლეთა დომენებსა და ქვედომენებზე მოქმედებს.

ტიპი (Sort). მონაცემთა სტრუქტურის სახელი.

არგუმენტის ტიპი (Argument Sort). ოპერატორის არგუმენტის ტიპი.

გამომავალი ტიპი (Output Sort). ოპერატორის შედეგის ტიპი.

არულობა (Arity) – ფუნქციაში შემავალი (არგუმენტები) და გამომავალი (შედეგი) ტიპები (მაგ., ბინარული, **n**-არული).

ტიპიზაცია (Typisation). ტიპის დაკავშირება პოზიციასთან.

აღწერები (**Declarations**). გამოსახულებათა სიმრავლე სიმრავლეთა, კონსტანტების, პარამეტრების მნიშვნელობათა, ტიპიზებული ცვლადებისა და ფუნქციების განსაზღვრისათვის, რომლებიც მაღალი დონის პეტრის ქსელებზე აისახება.

ოპერატორი (Operator). სიმბოლოთა ერთობლიობა (აბრევიატურა) ფუნქციის სახელის წარმოსადგენად.

პარამეტრი (Parameter). მუდმივა (კონსტანტა), რომელიც სიმრავლეში განსაზღვრულ სიდიდეთა არეს შეიცავს.

მინიჭება (Assignment). მნიშვნელობის მინიჭება ცვლადების სიმრავლის კონკრეტული ცვლადისათვის.

სიგნატურა (Signature). ალგებრული სტრუქტურა, რომელიც ტიპების და ოპერატორების სიმრავლეებისგან შედგება.

ბულის სიგნატურა (Bool Signature). სიგნატურა, რომელიც ბულის (ლოგიკურ) ტიპს შეიცავს.

მრავალტიპური სიგნატურა (Many-sorted Signature). სიგნატურა, სადაც ტიპების სიმრავლის კარდინალურობა ერთზე მეტია.

ცვლადიანი სიგნატურა (Signature with Variables). სიგნატურა, რომელიც შეიცავს ცვლადების სახელებს, ტიპებს და ოპერატორებს.

თერმი (Term). სიგნატურის საფუძველზე შედგენილი გამოსახულება, რომელიც შეიცავს მუდმივებს, ცვლადებს და ოპერატორებს.

დახურული თერმი (Closed Term). თერმი, რომელიც შეიცავს კონსტანტებს და ოპერატორებს, მაგრამ არა ცვლადებს.

თერმის მნიშვნელობა (Term Evaluation). შედეგი, რომელიც მიიღება თერმის ცვლადებითვის მნიშვნელობების მინიჭებისა და ფუნქციათა შედეგების გამოთვლის შემდეგ.

მაღალი დონის პეტრის ქსელი (High Level Petri Net). ალგებრული სტრუქტურა, რომელიც შეიცავს: პოზიციების სიმრავლეს; გადასასვლელთა სიმრავლეს; ტიპების სიმრავლეს; ტიპების პოზიციებზე და ტიპების გადასასვლელებზე დამაკავშირებელ ფუნქციებს; პრეფუნქციებს შემავალი და პოსტფუნქციებს გამომავალი მარკირებების განსაზღვრისათვის; საწყის მარკირებას.

პეტრის ქსელის გრაფი (Petri Net Graph). მიმართული გრაფი ორი ტიპის კვანძებითა (პოზიციები და გადასასვლელები) და მათი დამაკავშირებელი რკალებით. დაშვებულია კავშირები „პოზიცია-გადასასვლელი“ ან „გადასასვლელი-პოზიცია“, მაგრამ არა „პოზიცია-პოზიცია“ ან „გადასასვლელი-გადასასვლელი“.

მაღალი დონის პეტრის ქსელის გრაფი (High Level Petri Net Graph). ქსელის გრაფისა და ანოტაციების (წარწერების) ერთობლიობა, რომელიც შეიცავს პოზიციათა ტიპებს, რკალების ანოტაციებს, გადასასვლელთა გახსნის პირობებს, შესაბამის განსაზღვრებებს განსაზღვრებათა სიაში და ქსელის საწყის მარკირებას.

მიღწევადობის გრაფი (Reachibility Graph). მიმართული გრაფი, სადაც კვანძები მიღწევად მარკირებებს შეესაბამება, რკალები – გადასასვლელთა გახსნის ოპერაციას.

პარამეტრიზებული მაღალი დონის პეტრის ქსელის გრაფი (Parameterized High Level Petri Net Graph). მაღალი დონის პეტრის ქსელის გრაფი, რომელშიც პარამეტრები განისაზღვრება.

4.2. მაღალი დონის პეტრის ქსელები (სემანტიკური მოდელი)

4.2.1. პეტრის ქსელი HLPN

პეტრის ქსელის სემანტიკური მოდელის აღწერის პროცესში გამოიყენება შემდეგი აბრევიატურები: **HLPN** – მაღალი დონის პეტრის ქსელი და **HLPNG** – მაღალი დონის პეტრის ქსელის გრაფი.

HLPN წარმოადგენს სტრუქტურას

$$\mathbf{HLPN = (P, T, D; Type, Pre, Post, M_0),}$$

სადაც:

- **P - პოზიციად** წოდებული ელემენტების სასრული სიმრავლეა;
- **T - გადასასვლელებად** წოდებული ელემენტების სასრული სიმრავლე, ისე, რომ $\mathbf{P \cup T = \varnothing;}$
- **D** – არაცარიელი **დომენების** სასრული სიმრავლე, რომლის ყოველ ელემენტს **ტიპი** ეწოდება;
- **Type :** $\mathbf{P \cap T \in D}$ წარმოადგენს ტიპების პოზიციებზე დაკავშირებისა და გადასასვლელის გახსნის რეჟიმის განსაზღვრის ფუნქციას;
- **Pre, Post : TRANS->μPLACE** წარმოადგენენ **წინასწარ** (გადასასვლელის გახსნამდე) და **შედეგის** (გადასასვლელის გახსნის შემდგომ) ასახვებს, სადაც
  - $\mathbf{TRANS = \{ (t,m) \mid t \subseteq T, m \subseteq Type(t) \}}$
  - $\mathbf{PLACE = \{ (p,g) \mid p \subseteq P, g \subseteq Type(p) \}}$
- $\mathbf{M_0 \subseteq \mu PLACE}$ მულტისიმრავლეა, რომელსაც **ქსელის საწყისი მარკირება** ეწოდება;
- $\mathbf{M \subseteq \mu PLACE}$ მულტისიმრავლეა, რომელსაც **ქსელის მარკირება** ეწოდება;

გადასასვლელის გაშვების რეჟიმების სასრული სიმრავლე, $\mathbf{T_\mu \in \mu TRANS}$ ნებადართულია **M**-მარკირებაში**,** თუ $\mathbf{Pre(T_\mu) \le M,}$ სადაც **Pre**-ს წრფივი გაფართოება შემდეგი სახისაა:

$$\mathbf{Pre(T_\mu) = \Sigma_{tr \in TRANS}\, mult(tr,T_\mu) Pre(tr)}$$

თუ გადასასვლელის გახსნის რეჟიმთა მულტისიმრავლე $\mathbf{T_\mu}$ ნებადართულია **M** მარკირებაში, მაშინ გადასასვლელის გახსნის პროცედურას **ბიჯი** ეწოდება და მისი შესრულების შედეგად მიღებული ახალი მარკირება გამოისახება ფორმულით:

$$\mathbf{M' = M - Pre(T_\mu) + Post(T_\mu)}$$

ბიჯის ფორმალური ასახვა შეიცავს საწყის და შედეგის მარკირებებს, აგრეთვე გადასასვლელთა გახსნის დაშვებული რეჟიმების მულტისიმრავლეს:

$$M \xrightarrow{T_\mu} M'$$

4.2.2. მაღალი დონის პეტრის ქსელის გრაფი – HLPNG

მაღალი დონის პეტრის ქსელის გრაფი წარმოადგენს სტრუქტურას:

$$\mathbf{HLPNG = (NG, Sig, H; Type, AN, M_0),}$$

სადაც

- **NG = (P,T;F) ქსელის გრაფად** იწოდება, რომელშიც
  - **P** კვანძების სასრული სიმრავლეა (**პოზიციები**);
  - **T** - კვანძების სასრული სიმრავლე (**გადასასვლე-ლები**) და $\mathbf{P \cup T = \varnothing}$**;**
  - $\mathbf{F \subseteq (P \times T) \cap (T \times P)}$ **- რკალებად** წოდებული მიმართული მონაკვეთების სიმრავლე**;**
- **Sig = (S, O, V)** წარმოადგენს გრაფის ნატურალურ-ლოგიკურ **სიგნატურას.**
- $\mathbf{H = (S_H, O_H)}$ სიგნატურისთვის განსაზღვრული მრავალსორ-ტიანი **ალგებრაა;**
- $\mathbf{Type : P \to S_H}$ ტიპების პოზიციებზე **დანიშვნის ფუნქციაა;**
- **AN = (A, TC)** ქსელის **ანოტაციათა** წყვილია, სადაც:
- $\mathbf{TC: T \to TERM(O \cap V)_{Bool}}$ წარმოადგენს **ფუნქციას**, რომელიც გადასასვლელებს ლოგიკური გამოსახუ-ლების ტიპის ანოტაციით აფართოებს;

- **M_0:P->$U_{p\in P}$μType(p),** ისე, რომ **∀p∈P, M_0(p)∈μType(p) საწყისი მარკირების ფუნქციაა,** რომელიც მარკერთა მულტისიმრავლეს ყოველი პოზიციის **ტიპთან** კორექტულად აკავშირებს.

გრაფიკულად **პოზიცია** წრეებით ან ელიფსებით გამოისახება. პოზიციის ანოტაცია (წარწერები) შედგება მინიმუმ პოზიციის სახელის, პოზიციასთან დაკავშირებული ტიპის სახელისა და საწყისი მარკირებისგან. თუ საწყისი მარკირება ცარიელია, იგი შეიძლება არ გამოისახოს.

გადასასვლელს მართკუთხედი ან შავი ხაზი გამოსახავს; გადასასვლელის ანოტაცია შედგება მინიმუმ მისი სახელისგან; თუ **გადასასვლელის გაშვების პირობა** მოცემულია, იგი მართკუთხედის შიგნით აისახება და გამოიტოვება მხოლოდ მაშინ, როცა ყოველთვის ჭეშმარიტია.

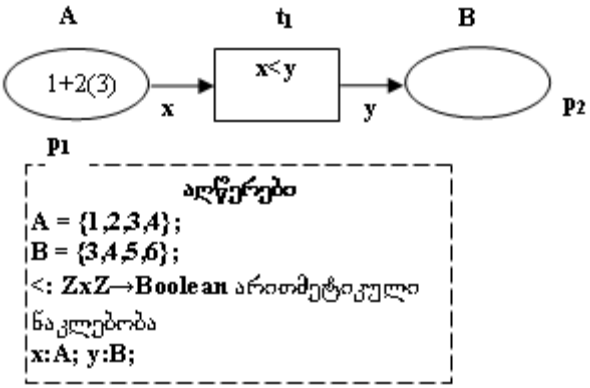
რკალს შეიძლება ჰქონდეს როგორც ერთ- ან ორმხრივმიმართული მონაკვეთის, ასევე მიმართული მრუდის სახე.

თუ ერთი და იგივე პოზიციისა და გადასასვლელის დამაკავშირებელ, შემავალ და გამომავალ რკალებს მსგავსი ანოტაციები აქვთ, ორმხრივმიმართული რკალების გამოსახვა ამ შემთხვევაში ერთი ორისრიანი რკალის სახითაც ნებადართულია. რკალის ანოტაციად ზემოთ აღწერილი **თერმების მულტისიმრავლეები** გამოიყენება.

მარკერი პეტრის ქსელში პოზიციათა მარკირების ელემენტს წარმოადგენს და შესაბამისად, პოზიციის გვერდზე (ან შიგნით) გამოისახება მისი მარკირების ფარგლებში **სიმბოლური ჯამის** სახით.

ფრჩხილები გამოიყენება მიმდინარე მარკირებაში მარკერის **სიხშირისა** (ფრჩხილებს გარეთ) და მარკერის **მნიშვნელობის** (ფრჩხილებს შიგნით) გამოსაყოფად.

პეტრის ქსელების ზემოთმოყვანილი ნოტაციის მაგალითი მოცემულია 4.1 ნახაზზე.



ნახ.4.1. HLPN - გრაფი

მოცემულია 2 **პოზიცია (p_1 და p_2)**, 1 **გადასასვლელი (t_1)** და **დამაკავშირებელი რკალები**. აღიწერება 2 **ტიპი**, **A** და **B**, რომლებიც **საბაზო სიმრავლეებს** წარმოადგენს და ნატურალურ რიცხვთა სხვადასხვა ქვესიმრავლეებს შეიცავს. ცვლადი **x A**-ტიპისაა, **y – B**-ტიპის. გადასასვლელი შეიცავს **გახსნის პირობას x<y**, რისთვისაც აღწერების სიაში „ნაკლებობის" **ოპერატორი** განისაზღვრება. რკალზე (p_1,t_1) დართულია **ანოტაცია** – ცვლადი **x**, ხოლო რკალზე (t_1,p_2) **ანოტაცია** – ცვლადი **y**.

პოზიცია p_1 **ტიპიზებულია A**-ტიპით და გააჩნია **საწყისი მარკირება 1+2(3),** რომელიც წარმოადგენს მულტისიმრავლეს $M_0(p_1)$**={(1,1),(2,0),(3,2),(4,0)},** სადაც ყოველი წყვილის პირველი ელემენტი ნატურალური რიცხვია **A={1,2,3,4}** საბაზო სიმრავლიდან, ხოლო მეორე – მისი **სიხშირე** მულტისიმრავლეში. პოზიცია p_2 **ტიპიზებულია B**-ტიპით და მისი **საწყისი მარკირება** ცარიელ მულტისიმრავლეს წარმოადგენს $M_0(p_2)=\{\varnothing\}$**.**

საწყისი მარკირებიდან გამომდინარე, გადასასვლელი t_1 **ნებადართულია** გადასასვლელის გახსნის შემდეგ **რეჟიმებში: {(1,3), (1,4), (1,5), (1,6), (3,4), (3,5), (3,5)}**, სადაც ყოველი წყვილის პირველი ელემენტი **x**-ცვლადის მნიშვნელობაა, ხოლო მეორე – **y**-ცვლადისა, ისე, რომ ყოველი წყვილისთვის **x<y**.

მოცემულ ქსელში გაშვების რეჟიმების **პარალელიზმიც** ფიქსირდება. მაგალითად, რეჟიმების მულტისიმრავლე **(1,3)+2(3,5)** t_1 გადასასვლელს პარალელურად ნებადართულს ხდის. შეიძლება ასეთი მაგალითის განხილვაც: **(1,5)+(3,4); (1,6)+(3,5)+(3,6)**.

თუ გაიხსნება **ნებადართული მარკირება**, მაგალითად, **(1,3) რეჟიმთა მულტისიმრავლეში** (რომელიც ამ შემთხვევაში მხოლოდ 1 ტიპის 1 ელემენტისგან შედგება), მაშინ **შედეგის მარკირებებს** შემდეგი სახე ექნება:

$M(p_1)$={(1,0),(2,0),(3,2),(4,0)};

$M(p_2)$={(3,1),(4,0),(5,0),(6,0)};

(1,3)+2(3,5) რეჟიმთა მულტისიმრავლეში გადასასვლელის გახსნა შემდეგ **შედეგის მარკირებებს** ჩამოაყალიბებს:

$M(p_1)$={∅};

$M(p_2)$={(3,1),(4,0),(5,2),(6,0)};

4.3. პეტრის ქსელების კლასიფიკაცია

პეტრის ქსელების ევოლუციის პირველ ეტაპზე მათი 3 თეორიული კლასი განისაზღვრა (I, II და III დონის პეტრის ქსელები), მაგრამ დღეისათვის მათი რაოდენობა ორამდე ჩამოვიდა - განისაზღვრება **დაბალი** და **მაღალი** დონის პეტრის ქსელები, ამასთან მეორე კლასი პირველს მოიცავს.

ძველი კლასიფიკაცია პოზიციებზე, გადასასვლელებსა და რკალებზე იყო ორიენტირებული და განასხვავებდა მათ ისეთ მახასიათებლებს, როგორიცაა მარკერთა მაქსიმალური რაოდენობა პოზიციაში, რკალების ჯერადობა და სხვა.

ახალ კლასიფიკაციაში ყურადღება უშუალოდ მარკერთა სემანტიკაზეა გამახვილებული.

კერძოდ, დაბალი დონის პეტრის ქსელებში დაიშვება მხოლოდ **„შავი"** მარკერები ყოველგვარი შინაგანი სტრუქტურის გარეშე, ხოლო მაღალი დონის პეტრის ქსელები დამატებით წინასწარ განსაზღვრული სტრუქტურის **„ფერად"** მარკერებსაც შეიცავს, თუმცა უნდა აღინიშნოს, რომ ტერმინები „შავი" და „ფერადი" სიმბოლურია და ლიტერატურაში მათ ხშირად განსხვავებული სახელებით მოიხსენიებენ.

მარკერთა არსი განსაზღვრავს შემდგომ პოზიციებისა და გადასასვლელების, აგრეთვე რკალების ანოტაციის შიგთავსს. ,,შავმარკერიან'' ქსელებში მხოლოდ არატიპიზებული, ერთგვაროვანი პოზიციები და გადასასვლელებია დაშვებული, ,,ფერად'' ქსელებში ყველა პოზიციისთვის საკუთარი ტიპი განისაზღვრება შესაბამისი ტიპის მარკერთა დომენით. ერთი ტიპის პოზიციაში მეორე ტიპის მარკერის არსებობა დაუშვებელია.

გადასასვლელები ფართოვდება გადასასვლელის გაშვების პირობებით, რომელიც ლოგიკურ გამოსახულებას წარმოადგენს და შეიძლება ჭეშმარიტი ან მცდარი იყოს.

რკალის ანოტაცია დაბალი დონის პეტრის ქსელში ან საერთოდ გამოიტოვება (რკალში ერთ გაშვებაზე მხოლოდ ერთი ,,შავი'' მარკერი გადაადგილდება) ან ნატურალურ რიცხვს წარმოადგენს, რომელიც გადასაადგილებელ მარკერთა რაოდენობას ასახავს (რკალის ჯერადობა), მაშინ, როცა მაღალი დონის პეტრის ქსელში რკალის ანოტაცია შეიძლება შეიცავდეს უფრო რთულ მონაცემებსაც, რომლებიც ქვემოთ განიხილება.

დაბალი დონის პეტრის ქსელების ქვეკლასებიდან შეიძლება დავასახელოთ ავტომატური პეტრის ქსელები, მარკირებული გრაფები, პეტრის ქსელები თავისუფალი არჩევანით, ელემენტარული სისტემური ქსელები, C/E-ქსელები, უსაფრთხო S/T ქსელები, S/T (კლასიკური) ქსელები და სხვა, ხოლო მაღალი დონის პეტრის ქსელების ყველაზე კარგად გამოკვლეულ და განსაზღვრულ ქვეკლასს **სისტემური პეტრის ქსელები** წარმოადგენს.

ავტომატურ პეტრის ქსელებში ანუ მდგომარეობათა მანქანებში **(State Machines)** ყოველ გადასასვლელს შეიძლება ჰქონდეს მაქსიმუმ 1 შესასვლელი და 1 გამოსასვლელი. იგი **მკაცრად შენახვადი** პეტრის ქსელია (მარკერების საერთო რაოდენობა მასში არასდროს იცვლება). ავტომატური პეტრის ქსელებით შეიძლება **კონფლიქტების**, მაგრამ არა **პარალელიზმის** მოდელირება.

მარკირებულ გრაფებში (Marked Graphs) ყოველი პოზიცია ზუსტად 1 გადასასვლელის შესასვლელს და ზუსტად 1 გადასასვლელის გამოსასვლელს წარმოადგენს. იგი თეორიულად

ავტომატური პეტრის ქსელების ორეულია, ამოდელირებს პარალელიზმს, მაგრამ კონფლიქტებს - ვერა.

მარკირებულ გრაფებში არსებობს **ციკლები** – შეკრული (ჩაკეტილი) გზა რომელიმე გადასასვლელიდან იმავე გადასასვლელამდე, რომელიც გადასასვლელთა გარკვეული მიმდევრობის გახსნით მიიღება. ციკლის გაშვების შედეგად მარკირებულ გრაფში მარკერების საერთო რაოდენობა არ იცვლება, თუმცა, ზოგადად, მარკირებული გრაფი შენახვადი არ არის (მასში მერკერების მთლიანი რაოდენობა შეიძლება იცვლებოდეს).

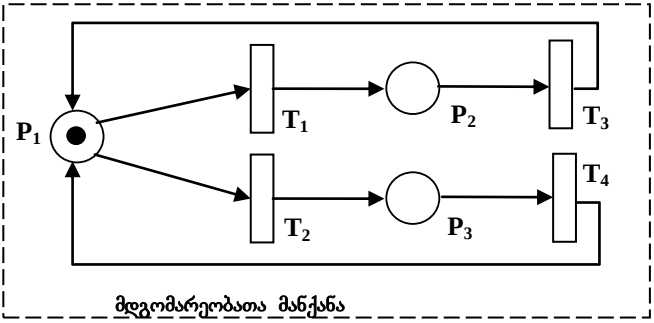
პეტრის ქსელებში თავისუფალი არჩევანით (Free Choise Petri Nets) მართვადი კონფლიქტის ცნება შემოდის: თუ რამდენიმე გადასასვლელს შემავალი პოზიციისთვის **კონფლიქტი** აქვს, პეტრის ქსელში თავისუფალი არჩევანით ისინი ყველა ნებადართული უნდა იყოს, ანუ საკონფლიქტე პოზიცია ერთადერთი შემავალი პოზიცია უნდა იყოს ყველა მოკონფლიქტე გადასასვლელისთვის.

ზემოთ აღწერილი 3 ქვეკლასის პეტრის ქსელების ნიმუშები მოცემულია 4.2 ნახაზზე.

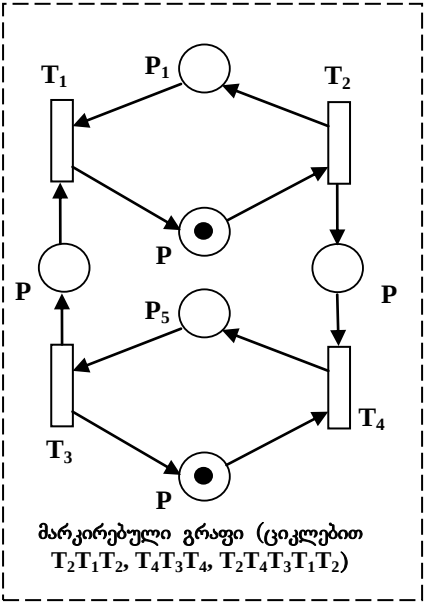
S/T-ქსელები (**State/Transition Nets**) **კლასიკური** პეტრის ქსელების წარმომადგენელია. იგი შედგება მსგავსი მარკერებისგან, რომელთა გრაფიკული ფორმა პატარა შავი წრეა პოზიციის ფარგლებში.

S/T-ქსელებში პოზიცია შეიძლება ერთზე მეტ მარკერს შეიცავდეს, ხოლო მარკერების დიდი ოდენობის შემთხვევაში პოზიციაში მათი რაოდენობა რიცხობრივად ჩაიწერება.

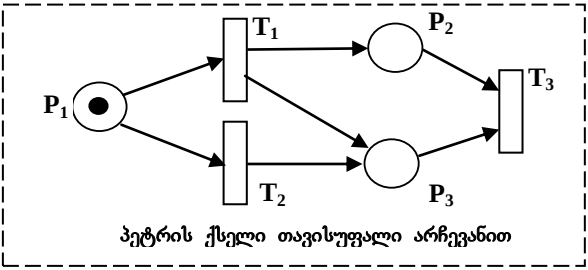
უსაფრთხო S/T-ქსელების პოზიციებში მარკერთა რაოდენობა არ უნდა აღემატებოდეს 1-ს.



მდგომარეობათა მანქანა



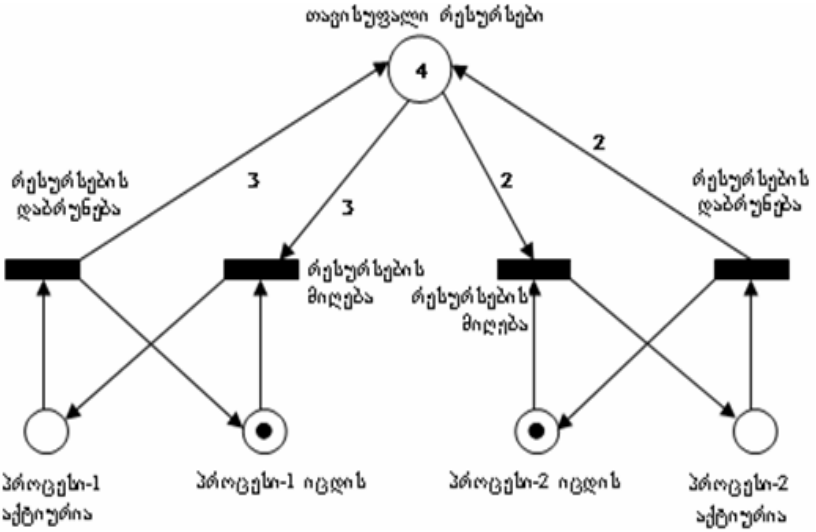
მარკირებული გრაფი (ციკლებით $T_2T_1T_2$, $T_4T_3T_4$, $T_2T_4T_3T_1T_2$)



პეტრის ქსელი თავისუფალი არჩევანით

ნახ.4.2. დაბალი დონის პეტრის ქსელების 3 ქვეკლასი

გადასასვლელის გაშვების აუცილებელი პირობაა ყველა შემავალ პოზიციაში დამაკავშირებელი რკალის ჯერადობაზე არანაკლები ოდენობის მარკერების მოგროვება. რკალების ჯერადობა ნატურალური რიცხვით გამოისახება (ნახ.4.3).



ნახ.4.3. კლასიკური პეტრის ქსელი „რესურსების განაწილების“ ამოცანისთვის

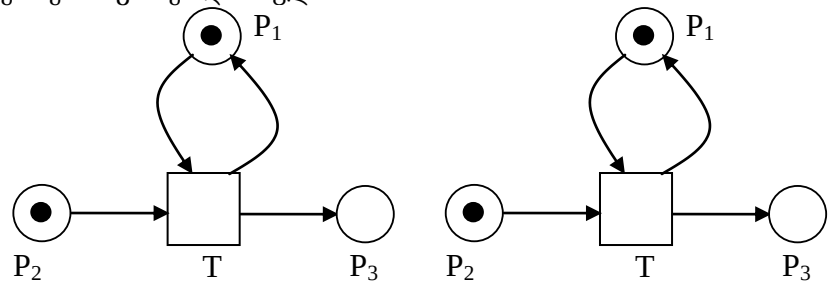
ელემენტარული სისტემური ქსელები (**Elementary System Nets**) **S/T**-ქსელების მსგავსად დაბალი დონის პეტრის ქსელების კლასში შედის, როგორც ყველა სხვა ქსელი, რომლებშიც მხოლოდ ერთგვაროვანი მარკერებია დასაშვები.

განსხვავება ისაა, რომ ელემენტარულ სისტემურ ქსელებში ერთი პოზიცია შეიძლება შეიცავდეს არაუმეტეს ერთ მარკერს. შესაბამისად, მასში ჯერადი რკალები არ არსებობს და რკალი ჭდეს (წარწერას) არ საჭიროებს. ამასთან, ელემენტარულ ქსელურ სისტემებში აკრძალულია გადასასვლელის გაშვება, თუ ერთი მაინც გამომავალი პოზიცია მარკერს უკვე შეიცავს [47].

მისგან განსხვავებით უსაფრთხო **S/T**-ქსელებისთვის მნიშვნელობა არა აქვს გამომავალი პოზიციების მდგომარებას,

მათში გადასასვლელი გაიშვება, თუ ყველა შემავალ პოზიციაში მარკერი იქნება. ამასთან, უსაფრთხო **S/T**-ქსელები კრძალავენ პოზიციაში ერთზე მეტი მარკერის არსებობას [48]. აქედან გამოდინარე, უსაფრთხო პეტრის ქსელში მარყუჟის არსებობა გადასასვლელის გახსნას ხელს ვერ უშლის, ელემენტარულ სისტემურ ქსელში კი პირიქით.

4.4 ნახაზზე მოცემული, გრაფიკულად სრულებით მსგავსი ელემეტარული სისტემური ქსელი და **S/T**-ქსელი განსხვავდება შესრულების მანერით: პირველში გადასასვლელი **T** ვერ გაიშვება, მეორეში იგი ნებადართულია.



ა)ელემენტარული სისტემური ქსელი

ბ) უსაფრთხო **S/T**-ქსელი

ნახ.4.4. გრაფიკულად მსგავსი და შესრულების წესებით განსხვავებული პეტრის ქსელები

4.4. სისტემური პეტრის ქსელები

System Petri Nets - ყველა ზემოაღწერილი ქვეკლასისგან განსხვავებით, მაღალი დონის პეტრის ქსელის ქვეკლასს წარმოადგენს.

მაღალი დონის პეტრის ქსელების სტანდარტული ნოტაციის თანახმად, სისტემური პეტრის ქსელებისთვის განისაზღვრება **კონსტანტები, ცვლადები** და **ფუნქციები**, რომელთა ერთობლიობას სისტემური პეტრის ქსელის **სტრუქტურა** ეწოდება, ხოლო გადასასვლელებისთვის განისაზღვრება გახსნის პირობა, რომელსაც

„**გადასასვლელის დამცავი ფუნქცია**” ეწოდება [47]. სისტემური პეტრის ქსელების ძირითადი განსაზღვრებები შემდეგია:

განსაზღვრება–1. ვთქვათ Σ ქსელია. Σ-ს A უნივერსუმი თითოეული $p \in P_\Sigma$ პოზიციისთვის აფიქსირებს მდგომარეობათა A_p სიმრავლეს, რომელსაც A-ში p-ს **დომენი** ეწოდება.

განსაზღვრება–2. ვთქვათ Σ ქსელია A უნივერსუმით, მაშინ

1. Σ-ს **მდგომარეობა** a თითოეული პოზიციისთვის განსაზღვრავს სიმრავლეს $a(p) \subseteq A_p$;
2. ვთქვათ $t \in T_\Sigma$. **ქმედება** m თითოეული მოსაზღვრე $f=(p,t)$ ან $f=(t,p)$ რკალისთვის განსაზღვრავს სიმრავლეს $m(f) \subseteq A_p$.

ეს ნიშნავს, რომ სისტემური ქსელების პოზიციებში ცვლადის მნიშვნელობათა სიმრავლის მოთავსება შეიძლება, რკალსაც ელემენტარული პეტრის ქსელებისგან განსხვავებით ერთზე მეტი მნიშვნელობის გატარება შეუძლია, რაც მაღალი დონის პეტრის ქსელებს ახასიათებს.

სისტემური პეტრის ქსელების შინაარსი სტრუქტურების ცნებაზეა დაფუძნებული. განვსაზღვროთ თავიდან კონსტანტისა და ფუნქციის ცნებები.

განსაზღვრება–3. დავუშვათ $A_1,\ldots,A_k$ არის ქვესიმრავლეთა სიმრავლე.

1. დავუშვათ $a \in A_i$, რომელიღაც $1<i<k$ - თვის. მაშინ a-ს ეწოდება **კონსტანტა** სიმრავლეში $A_1,\ldots,A_k$ და A_i-ს ეწოდება a-ს **კლასი**.

2. $i=1,\ldots,n+1$ - თვის დავუშვათ $B_i \in \{A_1,\ldots,A_k\}$, და ვთქვათ, $f: B_1 \times \ldots \times B_n \to B_{n+1}$ არის ფუნქცია. მაშინ f-ს ეწოდება ფუნქცია $A_1,\ldots,A_k$ სიმრავლეებზე. სიმრავლეები $B_1,\ldots,B_n$ წარმოადგენს f-ის არგუმენტების ტიპებს, ხოლო B_{n+1} შედეგის ტიპს. $n+1$ კორტეჟი $(B_1,\ldots,B_{n+1})$ წარმოადგენს f-ის **არეს** და შემდეგნაირად ჩაიწერება $B_1 \times \ldots \times B_n \to B_{n+1}$.

სისტემური პეტრის ქსელის **სტრუქტურა** კონსტანტებისა და ფუნქციების სიმრავლეს ეწოდება.

განსაზღვრება–4. დავუშვათ $\mathbf{A_1,\ldots,A_k}$ არის ქვესიმრავლეთა სიმრავლე, $\mathbf{a_1,\ldots,a_l}$ კონსტანტები $\mathbf{A_1,\ldots,A_k}$-ში და $\mathbf{f_1,\ldots,f_m}$ ფუნქციები $\mathbf{A_1,\ldots,A_k}$-ზე. მაშინ $\mathbf{A=(A_1,\ldots,A_k;a_1,\ldots,a_l;f_1,\ldots,f_m)}$ წარმოადგენს **სტრუქტურას**. $\mathbf{A_1,\ldots,A_k}$ **მატარებელი სიმრავლეებია**, $\mathbf{a_1,\ldots,a_l}$ - **კონსტანტები**, ხოლო $\mathbf{f_1,\ldots,f_m}$ - **ფუნქციები**.

სტრუქტურების ფუნქციათა შემადგენლობა **თერმების** ცნებით აღიწერება. თერმები შეიცავს **ცვლადებს** და **კონსტანტებს** და ისევე როგორც ცალკეული ცვლადები, კონკრეტულ მომენტში კონკრეტულ მნიშვნელობას ღებულობს.

თერმებისა და ცვლადების უკეთ წარმოსადგენად განვიხილოთ "მოთხოვნათა მომსახურების" სისტემა. იგი შედგება მონაცემთა 3 მომხმარებლისაგან **(u,v,w),** რომელთა ციკლურ მომსახურებას აწარმოებს მონაცემთა 2 მენეჯერი (**m** და **n**).

საწყის მდგომარეობაში თითოეული მომხმარებელი ლოკალურ მდგომარეობაშია, ხოლო გარკვეულ მომენტში მომხმარებელი მოითხოვს მონაცემებს ორივე მენეჯერისგან და მხოლოდ მას შემდეგ, რაც ორივესგან დაკმაყოფილდება, უბრუნდება ლოკალურ მდგომარეობას.

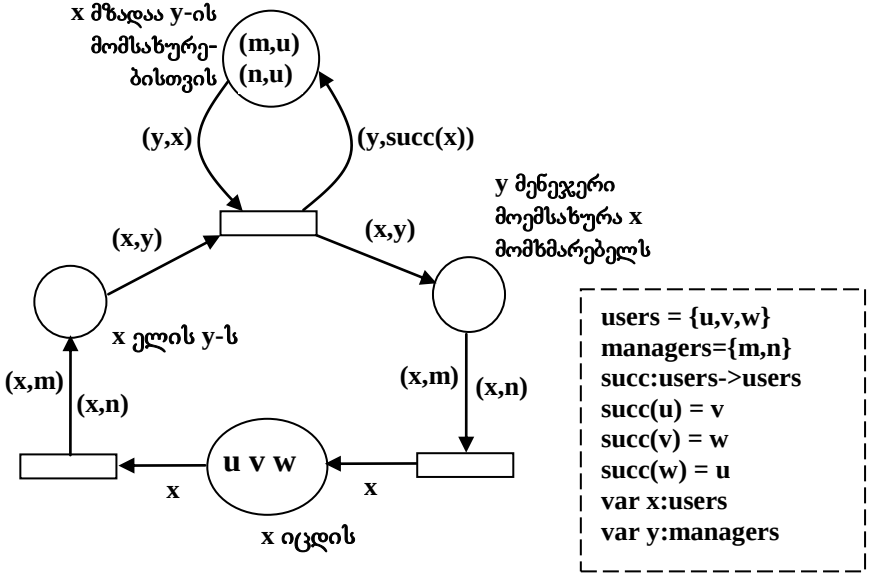
მომსახურებას პირველად **u** ღებულობს, მერე **v** და ბოლოს **w**. 4.5 ნახაზზე წარმოდგენილია სისტემური პეტრის ქსელი მოცემული სისტემისათვის.

ამ შემთხვევაში პეტრის ქსელის სტრუქტურას შემდეგი სახე ექნება:

"მოთხოვნათა მომსახურება"=(users,managers,
users×managers, managers×users,u,v,w,m,n,succ)

სადაც **succ** (ინგლისური სიტყვიდან **success** - "წარმატება") მოთხოვნის წარმატებით შესრულების მაუწყებელი ფუნქციაა, მისი არგუმენტი **x** ცვლადია, რომელიც **users** ტიპისაა და შესაბამისად, მნიშვნელობებს მხოლოდ **{u,v,w}** სიმრავლიდან იღებს, ხოლო თავად **succ**-ის მნიშვნელობა რიგში მომდევნო ადგილას მდგარი

მომხმარებელია, რომელიც ფუნქციის მეშვეობით მენეჯერთან მიმართვის უფლებას ღებულობს.



ნახ.4.5. სისტემური პეტრის ქსელი „მოთხოვნათა მომსახურების ამოცანისთვის“

სისტემურ ქსელზე რკალთა ზოგიერთი წარწერა თერმია, მაგალითად, **(x,m)** და იგი შეიცავს ცვლადსაც **(x)** და კონსტანტასაც **(m)**, როგორც ზემოთ აღვნიშნეთ.

სისტემური ქსელის მუშაობის პირველ ბიჯზე, ამოცანის პირობის თანახმად, მომსახურებას ღებულობს **u** მომხმარებელი, დავუშვათ **m** მენეჯერისგან. მაშინ თერმი **(y,x)=(m,u),** თერმი **(x,m)=(u,m)** და ცვლადი **x=u**.

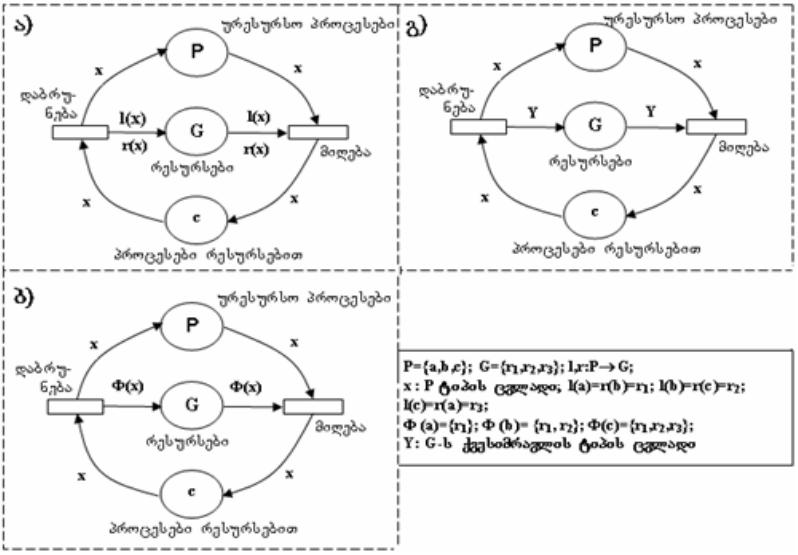
თუ **x**-ს სხვა მნიშვნელობას მივანიჭებთ, მაგალითად **v**-ს, გადასასვლელი „მომსახურების მოთხოვნა" კი გაიხსნება, მაგრამ თავად „მომსახურება" - ვერა, რადგან ამ დროს **(y,x)** თერმის მნიშვნელობა **(m,v)** უნდა გახდეს, რის საშუალებასაც პოზიციის („**y** მზადაა **x**-ის მომსახურებისთვის") საწყისი შემცველობა არ იძლევა და შესაბამისად, ქსელი არ იმუშავებს.

უფრო რთული შემთხვევებისთვის თერმების სიმრავლე შემოიტანება (**სიმრავლე-თერმები**), რომელიც სამი ტიპისაა: **კონსტანტა-თერმი, ფუნქცია-თერმი** და **ცვლადი-თერმი**.

კონსტანტა-თერმი ქსელში არა ცალკეული კონსტანტების, არამედ კონსტანტების (მაგალითად, ნატურალური რიცხვების) სიმრავლეთა ასახვისთვის გამოიყენება და ამ შემთხვევისთვის პოზიციის აღმნიშვნელი **ჭდე** არა კონსტანტების სიმრავლეს, არამედ სიმრავლის აღმნიშვნელ სიმბოლოს (დიდ ლათინურ ასოს) წარმოადგენს, ხოლო თავად სიმრავლე ქსელის სტრუქტურის განსაზღვრათა ბლოკში აღიწერება.

ფუნქცია-თერმი რკალის ანოტაციაა, აღინიშნება **Φ-**სიმბოლოთი და პეტრის ქსელში კონსტანტების ცვლადი რაოდენობის ტრანსპორტირებას ემსახურება არგუმენტ-ცვლადის მნიშვნელობის შესაბამისად.

ცვლადი-თერმი ასევე რკალის ანოტაციას წარმოადგენს და ქსელში მთლიანად სიმრავლის ან მის ქვესიმრავლეთა გადასატანად გამოიყენება. თერმების სიმრავლეთა ტიპების მაგალითები მოცემულია 4.6 ნახაზზე.



ნახ.4.6. სიმრავლე-თერმები

4.5. პეტრის ქსელის გაფართოებული ტიპები

პეტრის ქსელების ქვეკლასიდან განისაზღვრება **პეტრის ქსელის ტიპი**. ახალი ტიპის განსაზღვრის არე შეიძლება მრავალნაირი იყოს. ყოველ პეტრის ქსელს გააჩნია შემდეგი საერთო ელემენტები: **პოზიციები, გადასასვლელები და რკალები,** რომლებითაც პეტრის ქსელის გრაფი იქმნება.

პეტრის ქსელის ახალი **ტიპის** განსაზღვრისას საფუძვლად სწორედ პეტრის ქსელის გრაფია აღებული და იგი შემდგომი ასახვებითა და ფუნქციებით პეტრის ქსელის კონკრეტულ ტიპამდე ფართოვდება.

პეტრის ქსელის სხვადასხვა ტიპები ერთმანეთისგან შეიძლება განსხვავდებოდეს მარკერთა ტიპებით და მათგან გამომდინარე ერთიანი მარკირების სისტემით, ქსელის ელემენტების აღწერით (ჭდეები) ან/და გადასასვლელთა გაშვების წესებით.

ჭდეები პეტრის ქსელის ელემენტებზე, ძირითადად მხოლოდ წარწერებია და შეიცავს **ელემენტის სინტაქსს, მაგრამ არა სემანტიკას**. შესაბამისად, ისინი ქსელის შესრულების პროცესში ვერაფერს ცვლის. ჭდეების დანიშნულება პეტრის ქსელის სინტაქსური კონტროლია.

ამის მიუხედავად, ახალი ტიპის ჭდის განსაზღვრა უკვე საკმარისია იმისთვის, რომ ახალი პეტრის ქსელის ტიპი იქნეს განსაზღვრული.

პოზიციებზე, გადასასვლელებზე ან/და რკალებზე დროითი დაყოვნების განსაზღვრას დროითი პეტრის ქსელის ტიპი შემოაქვს, დაყოვნების დროთა ალბათურ განაწილებას – სტოქასტური პეტრის ქსელის ტიპი და ასე შემდეგ.

პეტრის ქსელების ტიპების განსაზღვრის უფრო ფართო გარემო მაღალი დონის პეტრის ქსელების ქვეკლასებია (მაგალითად, სისტემური პეტრის ქსელები). ქვემოთ მოკლედ აღვწერთ ყველაზე კარგად დამუშავებულ და გავრცელებულ პეტრის ქსელის ტიპებს.

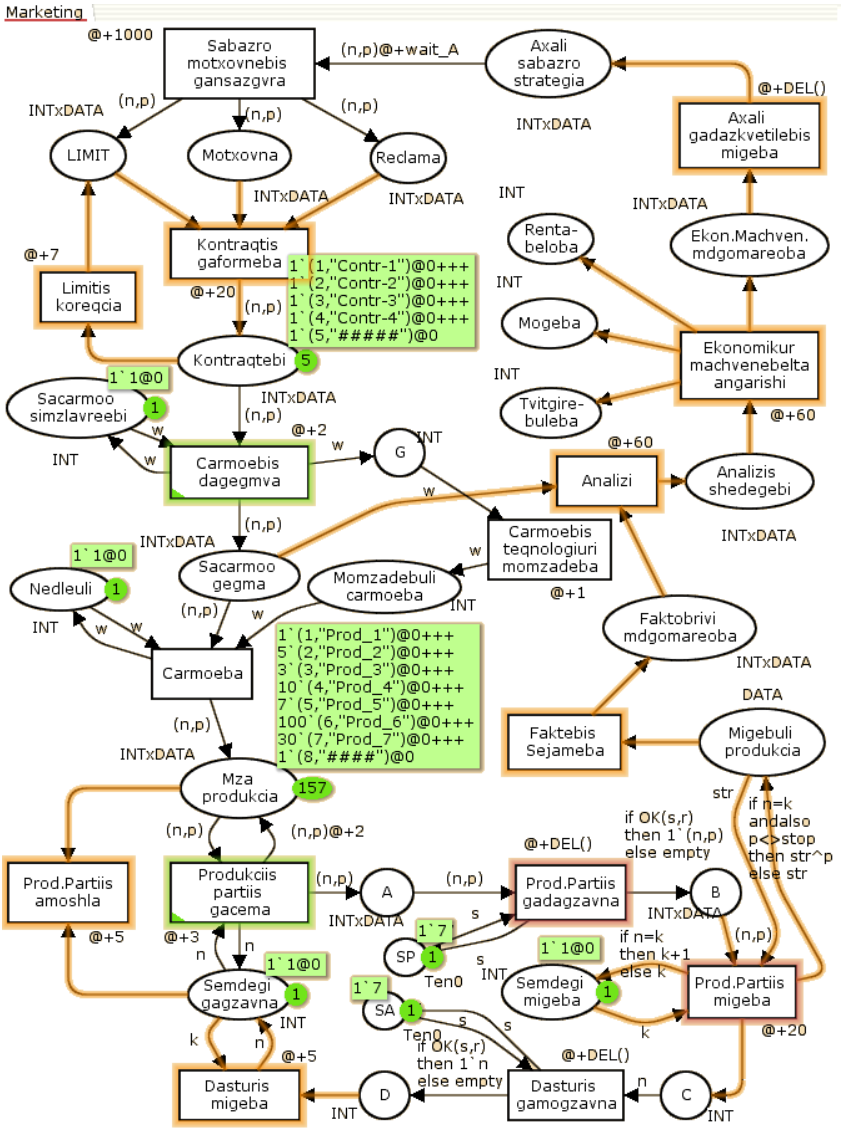
4.5.1. ფერადი პეტრის ქსელები: CPN ინსტრუმენტი

ფერადი პეტრის ქსელები (Coloured Petri Nets) მაღალი დონის პეტრის ქსელებია და სხვადასხვა ფერის მარკერებს შეიცავს [49,50]. ტერმინი ,,ფერადი'' ქსელში განსხვავებული მარკერების არსებობაზე მიანიშნებს - ამგვარი ქსელების დაბალი დონის პეტრის ქსელებისგან გამოსარჩევად, რომლებიც ერთგვაროვან, ,,შავ'' მარკერებს შეიცავს. ფერადი პეტრის ქსელის სტრუქტურა რთულია და იგი მრავალი სახეობის ჭდეებს შეიცავს.

4.7 ნახაზზე მოცემულია ფერადი პეტრის ქსელის ფრაგმენტი საწარმოო ფირმის მარკეტინგული პროცესების მოდელირებით [51]. აქ გადასასვლეების ბლოკებში ნაჩვენებია, მაგალითად, საბაზრო მოთხოვნილებათა განსაზღვრის, კონტრაქტების ფორმირების, წარმოების დაგეგმვის, წარმოების ტექნოლოგიური მომზადების, პროდუქციის წარმოების, დამკვეთებზე გადაცემა-გაგზავნის, შედეგების ანალიზის და ახალი გადაწყვეტილებების მიღების პროცესები. თითოეული მათგანი უნდა გაიშალოს დამოუკიდებელი პეტრის ქსელით და მოხდეს მათი ანალიზი, ამასთანავე შეიქმნება ერთიანი იერარქიული სისტემა ჩადგმული პეტრის ქვექსელებით.

ფერადი პეტრის ქსელების გრაფო-ანალიზური CPN-ინსტრუმენტი იყენებს ობიექტ-ორიენტირებული, ვიზუალური დაპროგრამების პრინციპებს, მისი ენა CPN საშუალებას იძლევა აღიწეროს ქსელის ფერადი კომპონენტები (მარკერები), ცვლადები, კონსტანტები და თვით პოზიციების, გადასასვლელებისა და რკალების ტექსტები, რაც ერთგვარ კომფორტს ქმნის ქსელის წასაკითხად და გასაგებად.

4.8 ნახაზზე ნაჩვენებია ამ ინსტრუმენტის ფუნქციების, დახმარების, ოფციებისა და აღწერის (Declarations) ჩვენი ქსელის მაგალითზე. მოვიყვანთ ზოგიერთ განმარტებებს CPN-ის წასაკითხად. ქსელის ყოველ პოზიციას გააჩნია მინიმუმ ორი ჭდე: სახელი, რომელიც აღმნიშვნელი წრის ან ელიფსის შიგნით იწერება და მარტივი ან შედგენილი ტიპი (პოზიციის გვერდით, კურსივით, საკვანძო სიტყვა type, color ან string).



ნახ.4.7. მარკეტინგული პროცესის ფრაგმენტი CPN-ის გარემოში

მაგალითად, პოზიცია ,,კონტრაქტები“ INTxDATA ტიპისაა, რომელიც წინასწარ განსაზღვრული INT და DATA ტიპების დეკარტული ნამრავლით წარმოიქმნება. ფერადი პეტრის ქსელი შეიცავს ,,ფერად“ მარკერებს, რომლებიც კონკრეტული ტიპის შესაძლო მნიშვნელობათა სიმრავლე ან მულტისიმრავლეა.

ნახაზზე ნაჩვენებია სისტემაში კონსტანტების (საკვანძო სიტყვა val), ცვლადების (var) და ფუნქციების (fun) აღწერა.

სხვადასხვა ტიპის მონაცემთა შორის კავშირების ასახვისთვის გამოიყენება სიმრავლეთა და კომპლექტე-ბის თეორიის ელემენტები. გარდა მონაცემთა ტიპისა, ყოველი პოზიციის გვერდით შეიძლება აისახოს მოცემულ მომენტში შემავალი ფერადი მარკერები.

ნახ.4.8. CPN-სამუშაო გარემო

საინიციალიზაციო მარკირება ხაზგასმული ტექსტის სახით გამოიტანება. მაგალითად, საწყის მდგომარეობაში პოზიცია ,,კონტრაქტები“ შეიცავს INTxDATA ტიპის ფერად მარკერთა 5-ელემენტიან სიმრავლეს: **{1,,(1,”კონტრაქტი–1“), 1,,(2, ,, კონტრაქტი–2“), 1,,(3, ,, კონტრაქტი–3“), 1,,(4, ,, კონტრაქტი–4“), 1,,(5, ,,##### “) }**. აქ ბოლო, მე-5 ელემენტი შეესაბამება დასასრულის იდენტიფიკაციას - stop.

საყურადღებოა „1“-იანი ყოველი ელემენტის დასაწყისში (მას კოეფიციენტი ეწოდება), რომელიც მიუთითებს, რომ პოზიციაშია არაუმეტეს 1 ცალი მოცემული ფერის მონაცემი (ანუ არსებობს მხოლოდ ერთი კონტრაქტი ნომრით „კონტრაქტი–1“, რომლის ფერია - რიგითი ნომერია 1). ამ შემთხვევაში გვაქვს მონაცემთა ელემენტების სიმრავლე.

მეორე მაგალითი, პოზიცია „მზა პროდუქცია“ შედგება 157 ელემენტისგან (1+5+3+10+7+100+30+1), რომლებიც 7 სხვადასხვა (მარკერების ფერის) დამზადებული პროდუქტის რაოდენობას, ანუ მულტისიმრავლეს ასახავს.

პროცესების შესრულების დრო (დაყოვნება) აისახება გადასავლელთან სიმბოლოს და დროის ერთეულის (მაგალითად, @+7, @+wait) მითითებით, სადაც wait წინასწარ განსაზღვრული კონსტანტაა.

ამავე ნახაზზე ასახულია არადეტერმინირებული ლოგიკური გამოსახულება (პირობის ბლოკი) ფერადი პეტრის ქსელის რკალებზე, რომელიც გადასასვლელთა გაშვების სხვადასხვა პირობებს და შედეგებს ასახავს, ანუ ლოგიკური პირობის ჭეშმარიტებისას გადასასვლელს განსხვავებული მნიშვნელობა მიეწოდება (ან გადასასვლელიდან განსხვავებული მნიშვნელობა გამოვა), მცდარობისას – განსხვავებული.

მაგალითად, გადასასველს „პროდუქციის პარტიის გადაგზავა“ გამოსასვლელ რკალზე აქვს ლოგიკური პირობა - თუ გამოგზავნილი პროდუქციის ნომერი (n) ემთხვევა კლიენტის კონტრაქტით მისაღებ პროდუქციის ნომერს (k), მაშინ გვაქვს „true“, წინააღმდეგ შემთხვევაში „false“, რაც იმას ნიშნავს, რომ საჭირო პროდუქცია არაა მოსული. თუ ყველაფერი წესრიგშია, მაშინ მიმღები უგზავნის მწარმოებელს შეტყობინებას გადასასვლელით „დასტურის გამოგზავნა“. პროდუქციის და შეტყობინების გადაცემათა ქსელში შემთხვევითი პროცესის არსებობა განპირობებულია დაყოვნების ცვლადი დროის გამო, რაც აისახება colset NetDelay=int with 25..75, fun DEL( )=NetDelay.ran( ) random-ფუნქციით. ლოგიკური პირობის მნიშვნელობა სხვადასხვა შემთხვევებში სხვადასხვანაირად განისაზღვრება. ინტერაქტიულ სიმულატორებში ჭეშმარიტება-მცდარობას თავად მომხმარებელი

განსაზღვრავს, ავტომატური სიმულაციისას – შემთხვევით სიდიდეთა გენერატორი.

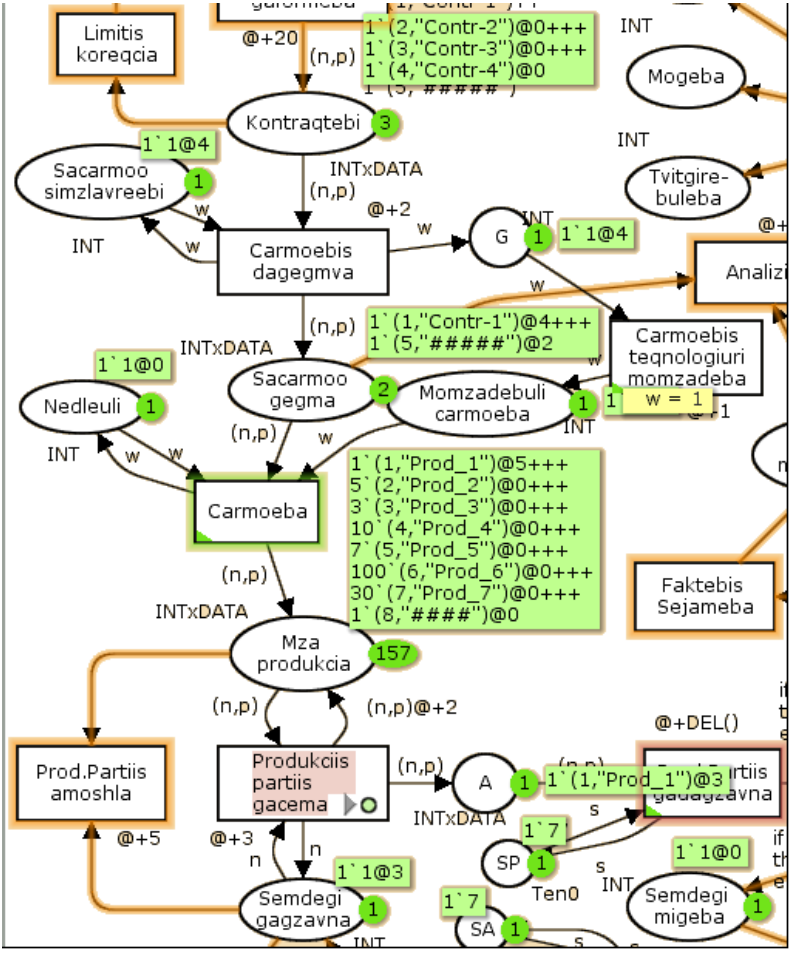
4.9 ნახაზზე ნაჩვენებია CPN-გარემოში პეტრის ქსელის აგებისა და იმიტაციური მოდელირების ვიზუალური კომპონენტები. სიმულაციის მე-3 ღილაკი (მწვანე რგოლით) საშუალებას იძლევა იტერაციულად, ხელით ავამუშავოთ ჩვენთვის საჭირო გადასასვლელი. მე-6 ღილაკი იძლევა საბოლოო მარკირების სურათს. 1-ელი ღილაკი – კი აღადგენს საწყის მარკირებას, ექსპერიმენტის თავიდან ჩასატარებლად.

ნახ.4.9 პეტრის ქსელის შექმნისა და იმიტაციური მოდელირების ინსტრუმენტები

4.10 ნახაზზე ნაჩვენები გვაქვს ჩვენი ქსელის ფრაგმენტი რამდენიმე ბიჯის შემდეგ, სადაც ჩანს მარკერების შეცვლილი მდგომარეობა. თავიდან გაიშვება გადასასვლელი „წარმოების დაგეგმვა“ (იხ. ნახ.4.7), ვინაიდან მის შესასვლელ პოზიციებში „კონტრაქტები“ და „საწარმოო სიმძლავრეები“ მზადაა მარკერები, რომლებიც გადაადგილდება პოზიციებში „საწარმოო გეგმა“ (n=1, p="Contr_1") და "G" (w=1). ეს უკანასკნელი არის სიგნალი იმის შესახებ, რომ 1-ელი კონტრაქტით გათვალისწინებული პროდუქციის საწარმოებლად საჭიროა „წარმოების ტექნოლოგიური მომზადება“, რასაც ასრულებს შესაბამისი გადასასვლელი. ამგვარად, თუ ნედლეულიც შემოსულია (პოზიციაში „ნედლეული“ არის 1 მარკერი), მაშინ გადასასვლელი „წარმოება“ ამუშავდება. პარალელურად ქსელში გაიშვება „პროდუქციის პარტიის გაცემის“ გადასასვლელი და მარკერი გადავა „გაგზავნის“ -პოზიციაში (n=1, p="Prod_1"). ტრანსპორტირების გარკვეული დროის შემდეგ

(სტოქასტიკური დრო: @+DEL( )) პროდუქცია მიაღწევს დამკვეთამდე და ა.შ.

CPN-ის სიმულაციის ინსტრუმენტით შესაძლებელია მარკირებათა მდგომარეობებისა და სტატისტიკური ანალიზის ჩატარება, შესაბამისი დიაგრამების აგებით.



ნახ.4.10. იმიტაციური მოდელირების შუალედური ეტაპი

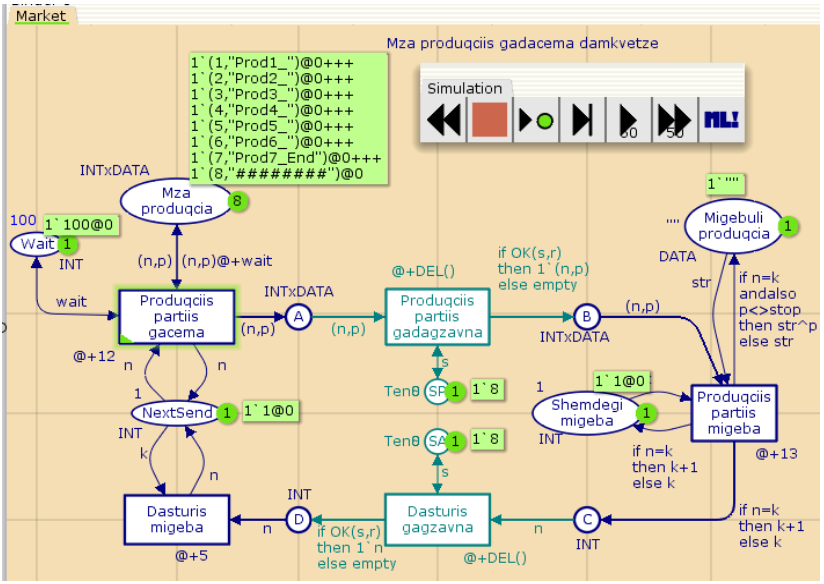
ფერადი პეტრის ქსელებში კარგადაა შერწყმული პეტრის ქსელებისა და დაპროგრამების თეორია (იერარქიულობა, მოდულურობა – დიდი სისტემების მოდელირებისთვის), რაც მის დიდ პრაქტიკულ ღირებულებასაც განაპირობებს თანამედროვე ინფორმაციულ ტექნოლოგიათა გამოყენების მრავალ სფეროში, განსაკუთრებით ბიზნესისა და მარკეტინგის მენეჯმენტის ამოცანების გადასაწყვეტად.

ქსელში ინფორმაციული ნაკადების მოძრაობის დიდი სირთულის გამო საჭიროა კვლევის ობიექტის დეკომპოზიციის განხორციელება, რაც CPN მოდელის წარმოდგენით მოხდება მისი იერარქიულად დაკავშირებული მოდულების საფუძველზე (ისე, როგორც პროგრამული პაკეტი შედგება მოდულებისგან, სტრუქტურული დაპროგრამების პრინციპებით) [49,52]

ამგვარად, პროდუქციის საწარმოო ფირმის მარკეტინგული პროცესების მოდელირებისათვის გვექნება შემდეგი ძირითადი იერარქიული მოდულები:

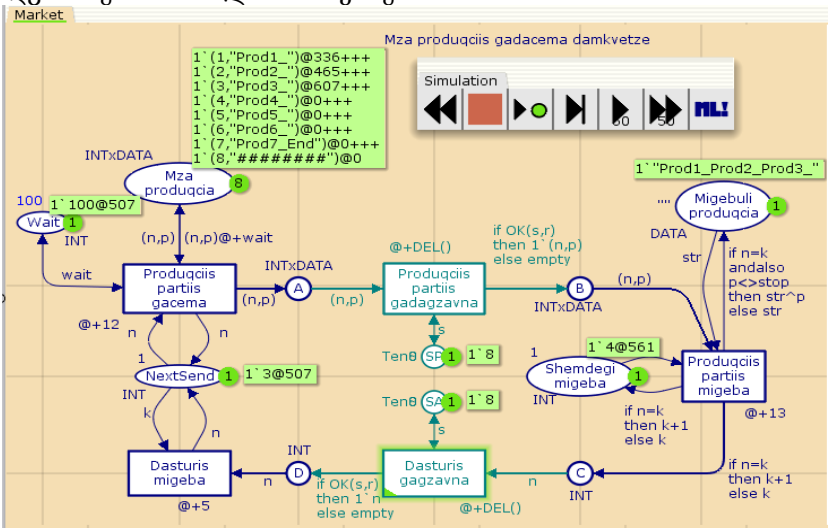
1. საბაზრო მოთხოვნების განსაზღვრის;
2. პროდუქციის წარმოების დაგეგმვის;
3. წარმოების ტექნიკური მომზადების და პროდუქციის წარმოების;
4. პროდუქციის გაცემის (სასაწყობო მეურნეობა);
5. პროდუქციის გადაგზავნის (ტრანსპორტირება);
6. პროდუქციის მიღების და დამკვეთის შეტყობინების;
7. ფაქტობრივი მდგომარეობის აღრიცხვის;
8. საწარმოო და სარეალიზაციო გეგმების შესრულების ანალიზის;
9. ეკონომიკური მაჩვენებლების ანგარიშისა და ანალიზის;
10. ახალი საბაზრო სტრატეგიის ფორმირების და ა.შ.

4.11 ნახაზზე ჩანს, რომ მარკერს Next_Send–ში აქვს დროითი ჭდე. ინტუიციურად ეს ნიშნავს, რომ მიმწოდებელს არ შეუძლია ახალი Prod.partiis_gacema –ის ან Receive Acknow–ის გაშვება, თუ ერთ-ერთი მაინც ამათგან უკვე გაშვებულია და არ დამთავრებულა.

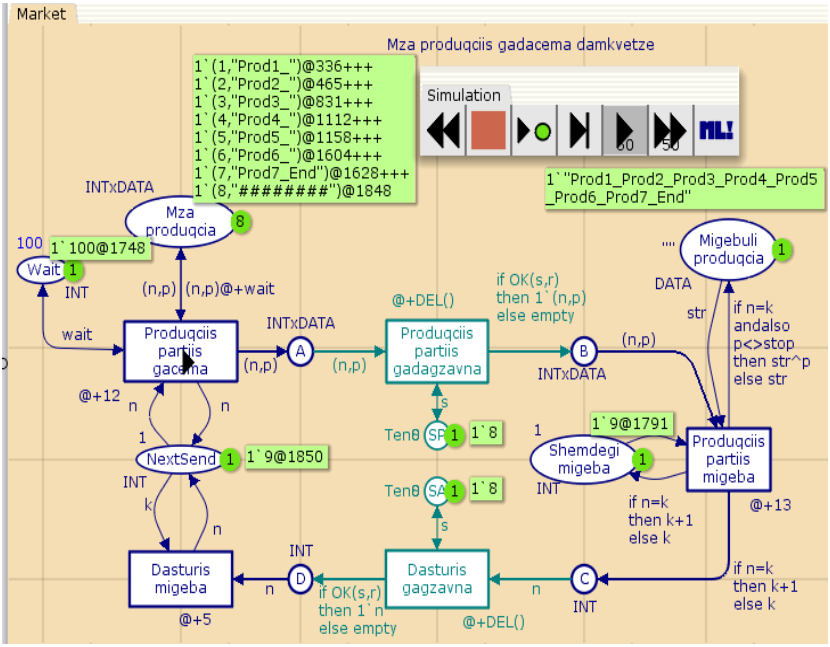


ნახ.4.11-ა. საწყისი მდგომარეობა

სიმულაციის რამდენიმე ბიჯის შემდეგ მიიღება დამყარებული მდგომარეობა საბოლოო მარკირებით.



ნახ.4.11-ბ. შუალედური მდგომარეობა
(პროდუქციის 3 პარტიის გადაცემის შემდეგ)



ნახ.4.11-გ. საბოლოო მდგომარეობა
(პროდუქციის 7 პარტიის გადაცემის შემდეგ)

Shemdegi_migeba–ის დროითი ჭდით ჩანს, რომ პროდუქციის ბოლო პარტია მიღებულ იქნა 1791 დრითი ერთელისას, ხოლო NextSend–ის დროითი ჭდე გვიჩვენებს, რომ ბოლო შეტყობინება პროდუქციის მიღების შესახებ მოვიდა 1850 დროით ერთეულში.

დროითი ჭდეები პოზიციაზე MzaProduqcia მიუთითებს პროდუქციის პარტიების (განმეორებითი) გადაცემის დროებზე. მაგალითად, პირველი პარტია გადაიცა 336 დროითი ერთეულისთვის, მეორე 465, მესამე 831 და ა.შ.

ჩვენი დროითი CPN-მოდელით შეიძლება გამოვიკვლიოთ მარკეტინგული პროცესის „პროდუქციის–გადაგზავნის“ შესრულების მახასიათებლები. მაგალითად, პაკეტების განმეორებითი გადაცემის დაყოვნების დროის (wait) სხვადასხვა მნიშვნელობისათვის. ხანმოკლე დაყოვნება ზრდის შანსს

განმეორებითი გადაგზავნების თავიდან ასაცილებლად. იგი ასევე ზრდის შანსს, რომ ოპერაცია Dasturis_migeba გადაიდოს, რადგან პროცესი Prod.partiis_gacema დაკავებულია განმეორებითი გადაგზავნით.

გრძელი დაყოვნება ნიშნავს, რომ საჭირო იქნება დიდხანს ცდა, სანამ მიმწოდებელი დარწმუნდება, რომ პაკეტი ან დასტური იქნა დაკარგული. სიმულაციის პროცესში, სხვადასხვა wait-მნიშვნელობით შეიძლება დადგინდეს ოპტიმალური მნიშვნელობა განმეორებითი გადაცემის დაყოვნებისათვის.

ახლა განვიხილოთ მარკეტინგის პროცესისთვის მზა პროდუქციის დამკვეთებზე მიწოდების (რეალიზაციის) გეგმის შესრულების (აღრიცხვის) შესაბამისი CPN პეტრის ქსელის ანალიზის ამოცანა ე.წ. მდგომარეობათა სივრცის საფუძველზე.

4.5.2. პეტრის ქსელის მდგომარეობათა სივრცე

მდგომარეობათა სივრცე (State Space) არის კვლევის ობიექტის შესაბამისი მოდელის ყველა შესაძლო მდგომარეობის ერთობლიობა. თვით მდგომარეობა, როგორც ეს კლასიკურ პეტრის ქსელებშია მიღებული, ასახავს მარკერთა განაწილებას ქსელის პოზიციების მიხედვით, ანუ მარკირებებს. ქსელის რომელიმე გადასასვლელის ამუშავების (გაშვების) შემდეგ ხდება მის შესასვლელ და გამოსასვლელ პოზიციებში მარკერთა რაოდენობის ცვლილებები. ამ დროს ქსელი გადადის ახალ მდგომარეობაში.

ასეთი პროცესი შეიძლება რომელიმე ბიჯზე დაიბლოკოს, ანუ ჩიხში შევიდეს, რაც იმის მაუწყებელია, რომ ასეთი მოდელი და მისი შესაბამისი რეალური ობიექტი ვერ მიაღწევს მიზანს, საბოლოო შედეგს. ამგვარად ქსელი ყოფილა არასაკმარისად მდგრადი და იგი მოითხოვს კორექტირებას.

ჩვენ შემთხვევაში საქმე გვაქვს მზა პროდუქციის მიწოდებასთან დამკვეთებზე, რომლის გეგმაც კონტრაქტების საფუძველზე იქნა შედგენილი და მისი შესრულება აუცილებელია (რათა არ მოხდეს ხელშეკრულების დარღვევასთან დაკავშირებული საჯარიმო სანქციების დაწესება).

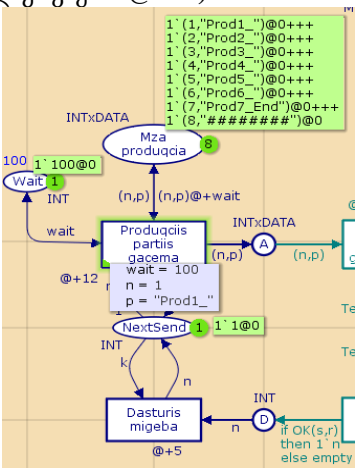
ჩვენი მოდელის ფრაგმენტის საფუძველზე, რომელიც წინა პარაგრაფში განვიხილეთ, ხდება მზა პროდუქციის გაცემა საწყობიდან, შემდეგ ტრანსპორტირება და დამკვეთამდე მიტანა. დამკვეთი, პროდუქციის მიღებისთანავე აგზავნის დასტურის შეტყობინებას და მიმწოდებელი ამის შემდეგ ზრუნავს მომდევნო პარტიის დამზადებასა და მიწოდებაზე.

არაა გამორიცხული შემთხვევები, რომ პროდუქციის პარტია ვერ მივიდეს დროულად დამკვეთთან (გარკვეული ობიექტურ-სუბიექტური მიზეზების გამო), ან დაიკარგოს დასტურის შეტყობინება. ასეთ შემთხვევებში საჭიროა ინფორმაციის დროულად გამოკვლევა და არშესრულებული პროცედურის გამეორება. ფერადი პეტრის ქსელის გადასასვლელები, როგორებიცაა Produqciis partiis gacema, Produqciis partiis gadagzavna, Produqciis partiis migeba, dasturis gagzavna და ა.შ. ხასიათდება დროითი დაყოვნებებით, რომლებიც ან კონსტანტური მნიშვნელობისაა, ან შემთხვევითი რიცხვების დიაპაზონიდან აიღება სისტემის მიერ. ამგვარად, CPN-ინსტრუმენტით შესაძლებელია მდგომარეო-ბათა სივრცის ანგარიშის მთლიანი პროცესის სრული ავტომატიზაცია, რაც მნიშვნელოვნად აჩქარებს ქსელის დიაგნოსტიკის პროცესს მისი რეალურ ობიექტთან ადეკვატურობის შესახებ, ანუ რამდენად სწორად ასახავს მოდელი რეალური ობიექტის ყოფაქცევას.

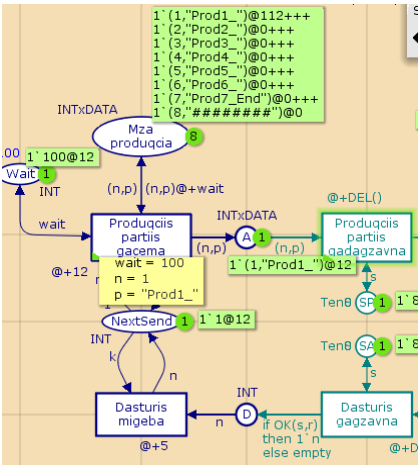
მდგომარეობათა სრული სივრცე – ორიენტირებული გრაფით აისახება, რომელშიც მწვერვალები შეესაბამება ქსელის დასაშვებ მარკირებებს, ხოლო რკალები – მოვლენებს დამაკავშირებელი ელემენტებით. ე.ი. M1 მდგომარეობიდან (მარკირებიდან) სისტემა გადადის M2 მდგომარეობაში, როდესაც არსებობს რკალი დამაკავშირებელი (n, p)- ელემენტით, სადაც n-ფერადი მარკერია, ხოლო p- ინფორმაციული ნაწილი. 4.12-ა ნახაზზე ნაჩვენებია პეტრის ქსელის საწყისი მდგომარეობის ფრაგმენტი {n=1, p=”Prod1”} ელემენტით.

4.12-ბ ნახაზი კი შეესაბამება პეტრის ქსელის ახალ მარკირებას პირველი ბიჯის შემდეგ. აქ შესამჩნევია, რომ A-პოზიციაში გაჩნდა ახალი, 1 მარკერი, რომლის ფერი=1, მონაცემი=”Prod1”. ამასთანავე ეს მარკერი მოვიდა ქსელის

ამუშავებიდან t=12 დროითი ერთეულის (მაგ., წუთი) შემდეგ (ვინაიდან Produqciis partiis gacemis გადასასვლელის დროითი დაყოვნებაა @+12).

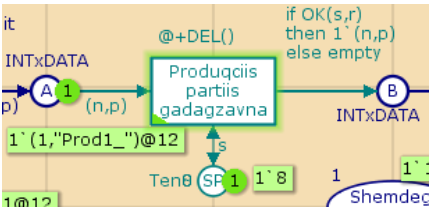


ნახ.4.12-ა. საწყისი მარკირება ბიჯის



ნახ.4.12-ბ. მარკირება პირველი შემდეგ

ახლა გააქტიურდა Produqciis partiis gadagzavnis გადასასვლელი და შესაძლებელია ასევე Produqciis partiis gacemis გადასასვლელის ხელახალი გაშვებაც. ეს ორივე პროცესი შეიძლება შესრულდეს პარალელურად, ისინი ერთმანეთს ხელს არ უშლის. 4.12-გ ნახაზზე ნაჩვენებია პროდუქციის პარტიის გადაგზავნის გადასასვლელის აქტიური მდგომარეობა. აქ მარკერები არის A და SP პოზიციებშიც.



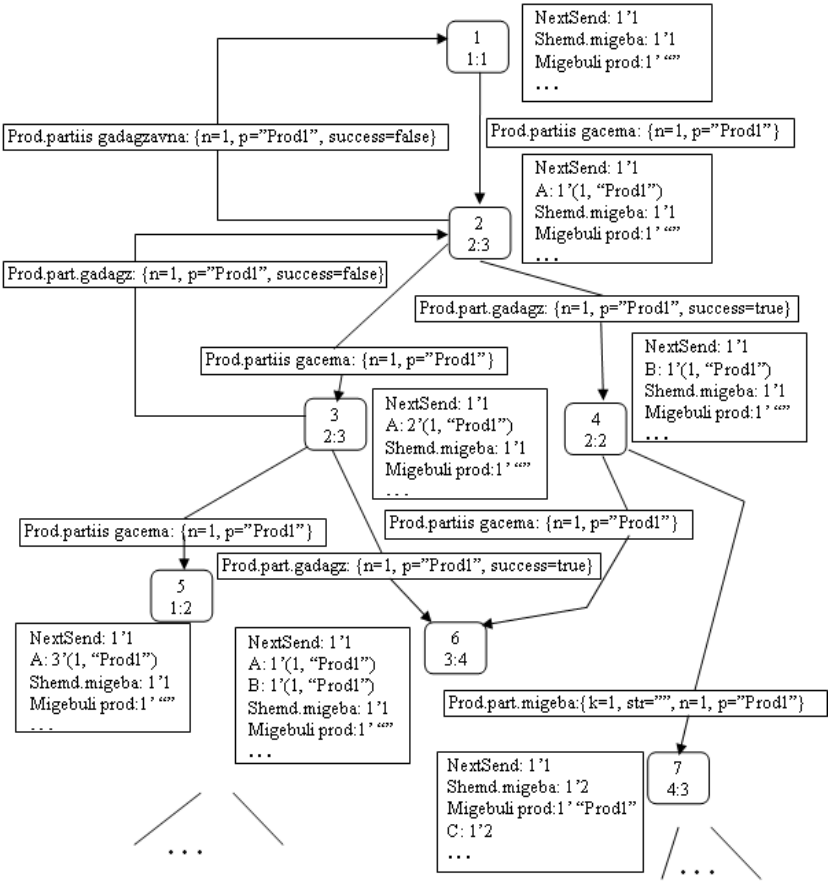
ნახ.4.12-გ

ამ გადასასვლელიდან B–პოზიციაში შემავალი რკალი ლოგიკურ პირობას აკონტროლებს, ანუ დასაშვებია ორი შემთხვევა:

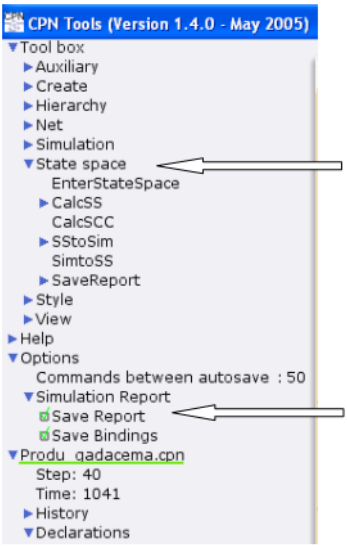
TP+=(Produqciis_partiis_gadagzavna, <n=1,p="Prod1", success=true>),
TP-=(Produqciis_partiis_gadagzavna, <n=1,p="Prod1", success=false>).

აქ ეს ორი დამაკავშირებელი ელემენტი TP+ და TP- იმყოფება კონფლიქტში ერთმანეთთან, ანუ ერთის შესრულება მეორეს გამორიცხავს. პირველით მოდელირდება ქსელში პროდუქციის პარტიის წარმატებით გადაცემა, ხოლო მეორეთი კი – ამ პარტიის დანაკარგია სახეზე.

4.13. ნახაზზე ნაჩვენებია აღწერილი პროცესის შესაბამისად ჩვენი ქსელის მდგომარეობათა სივრცის ფრაგმენტი, რომელიც, როგორც აღვნიშნეთ, ორიენტირებული გრაფითაა წარმოდგენილი.



ნახ.4.13. მდგომარეობათა სივრცის ფრაგმენტი CPN-მოდელისათვის

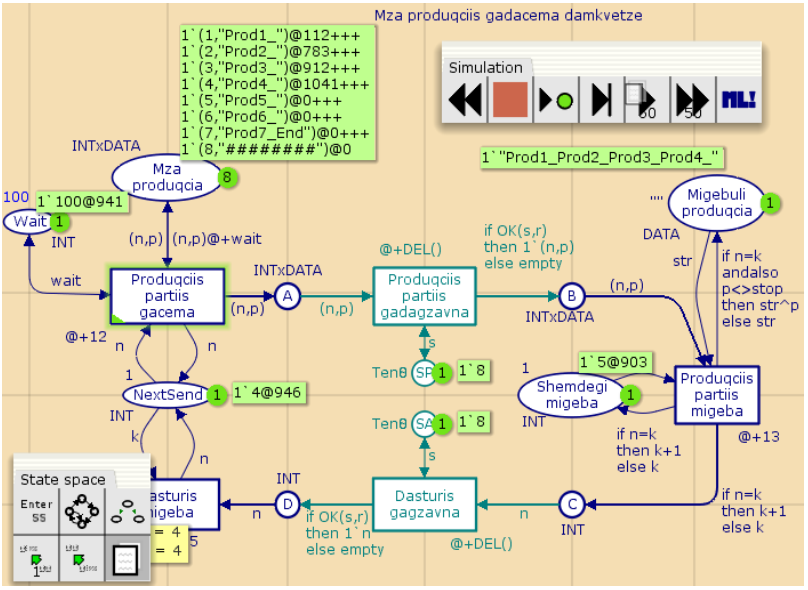


4.14 ნახაზზე მოცემულია იმიტაციური მოდელირების (სიმულაციის) და მდგომარეობათა სივრცის ანალიზის რეპორტების მომზადების ინსტრუმენტი.

აღნიშნული CheckBox-ის ჩართვის შემთხვევაში, რეპორტები ავტომატურად მოთავსდება C:\temp საქაღალდეში, რომელიც წინასწარ უნდა შეიქმნას.

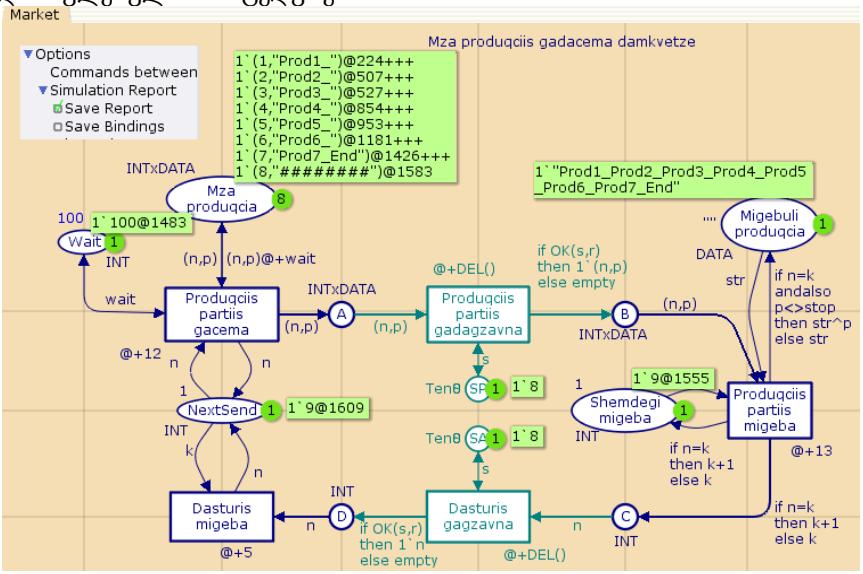
ნახ.4.14

4.15. ნახაზზე მოცემულია CPN-ქსელის ფუნქციონირების ფრაგმენტი 4-პროდუქციის პარტიის ნორმალური გადაგზავნისთვის.



ნახ.4.15. State Space - ინსტრუმენტით

4.16 ნახაზზე ნაჩვენებია მარკეტინგის დეპარტამენტის პროდუქციის მიწოდების CPN-ქსელის შესაბამისი სქემა დასრულებული პროცედურებით.



ნახ.4.16. საბოლოო მდგომარეობა და იმიტაციის შედეგად Save Report–ის მიღება

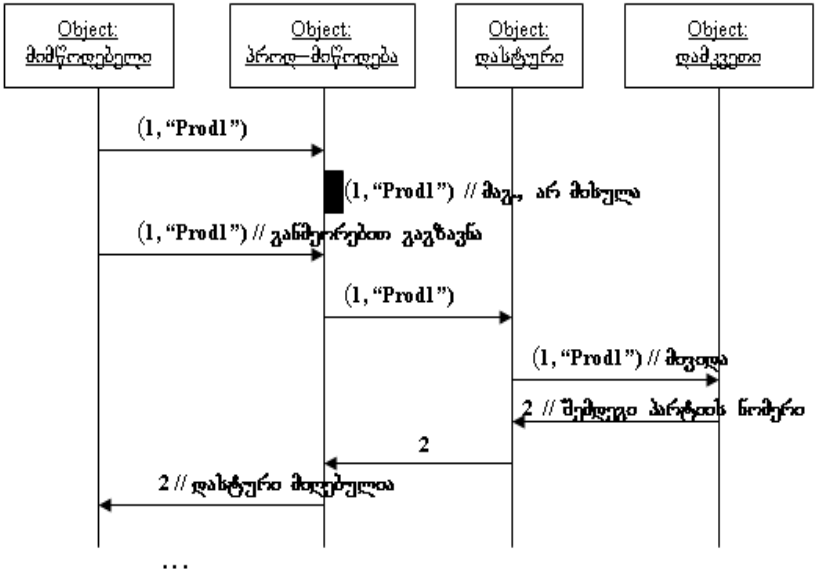
4.17 ნახაზზე ასახულია CPN-ქსელის ფუნქციონირების დროს კონსტანტების, ცვლადების (როგორც კლასის მონაცემების) და ფუნქციების (კლასის მეთოდების) ჩამონათვალი.

ზემოაღნიშნულიდან კარგად ჩანს, რომ ფერადი პეტრის ქსელების CPN-ინსტრუმენტი ობიექტ-ორიენტირებული მოდელირების და დაპროგრამების მეთოდების მატარებელია.

```
Declarations
  colset INT = int timed;
  colset DATA = string;
  colset INTxDATA = product INT * DATA timed;
  var n, k, wait: INT;
  var p, str: DATA;
  val stop = "########";
  colset Ten0 = int with 0..10;
  colset Ten1 = int with 1..10;
  var s: Ten0;
  var r: Ten1;
  fun OK(s:Ten0,r:Ten1) = (r<=s);
  colset NetDelay
  fun DEL() = NetDelay.ran();
Market
```

ნახ.4.17

აღნიშნული პროცესების შესრულება უნიფიცირებული მოდელირების ენის UML-ტექნოლოგიაში მოგვაგონებს შეტყობინებათა (Messages) მართვას ინტერაქტიურობის დინამიკურ მოდელში, რომელსაც მიმდევრობითობის დიაგრამით (Sequence-D) ვიცნობთ. 4.18 ნახაზზე მოცემული გვაქვს ასეთი დიაგრამის ფრაგმენტი:



ნახ.4.18. მიმდევრობითობის დიაგრამა

დასასრულ, შეიძლება დავამატოთ, რომ დღეისათვის ფერადი პეტრის ქსელები და მისი კომპიუტერული რეალიზაცია „**CPN-Tools**“ მსოფლიოს 40 ქვეყნის 400-ზე მეტ ორგანიზაციაში გამოიყენება სისტემების მოდელირების ინსტრუმენტად (მათგან 100-მდე კომერციულ კომპანიაში) [19]. განსაკუთრებით მაღალია მოთხოვნილება ამ ინსტრუმენტზე ამერიკის, ევროპისა და ჩინეთის ქვეყნების უნივერსიტეტებში.

4.5.3. დროითი პეტრის ქსელები

დროითი პეტრის ქსელები ფაქტობრივად ყოველი ტიპის პეტრის ქსელისთვის დროითი გაფართოების დამატებით მიიღება. დროითი გაფართოება აუცილებელია რეალური საპრობლემო სფეროს მოდელირებისთვის, მის გარეშე პეტრის ქსელი მხოლოდ სისტემის რაოდენობრივი ანალიზისთვის გამოდგება.

დროითი პეტრის ქსელი 4 ტიპის არსებობს: პოზიციურ-დროითი (Timed Places Petri Nets - TPPNs), ტრანზაქციულ-დროითი (Timed Transition Petri Nets - TTPNs), რკალურ-დროითი და მარკერულ-დროითი [11].

პოზიციურ-დროითი ტიპისთვის განისაზღვრება დაყოვნების ერთი და იგივე დრო პოზიციაში მოთავსებული ყველა მარკერისთვის და დროის ათვლა იწყება შესაბამისი გადასასვლელის გააქტიურებისთანავე (როცა მისი გახსნა ნებადართული ხდება). ყველა შემავალი პოზიციის დაყოვნების დროის გასვლის შემდეგ გადასასვლელი გაიხსნება.

ტრანზაქციულ-დროით პეტრის ქსელებში დაყოვნების დრო გადასასვლელისთვის (ტრანზაქციისთვის) განისაზღვრება. პეტრის ქსელების ეს ტიპი 2 ქვეტიპს შეიცავს: **წინასწარი არჩევანისა** და **შეჯიბრების** მოდელებს.

წინასწარი არჩევანის შემთხვევაში გადასასვლელი გააქტიურებისთანავე იღებს მონოპოლურ უფლებას ყველა შემავალ პოზიციაში მოთავსებულ მარკერების იმ ოდენობაზე, რაც მისი გახსნისთვის აუცილებელია (სხვა პოზიციებთან კონფლიქტში იმარჯვებს). ამის შემდგომ იწყება დაყოვნების დროის ათვლა. მისი გასვლისთანავე გადასასვლელი გაიშვება პეტრის ქსელის წესების მიხედვით, ანუ გადასასვლელის გააქტიურებას აუცილებლად მისი გახსნა მოჰყვება.

შეჯიბრის მოდელში მთავარი დროითი ფაქტორია, მარკერები ყველა აქტიურ გადასასვლელს ეკუთვნის და გაივლის მას, რომლის დაყოვნების დროც უფრო მალე გავა.

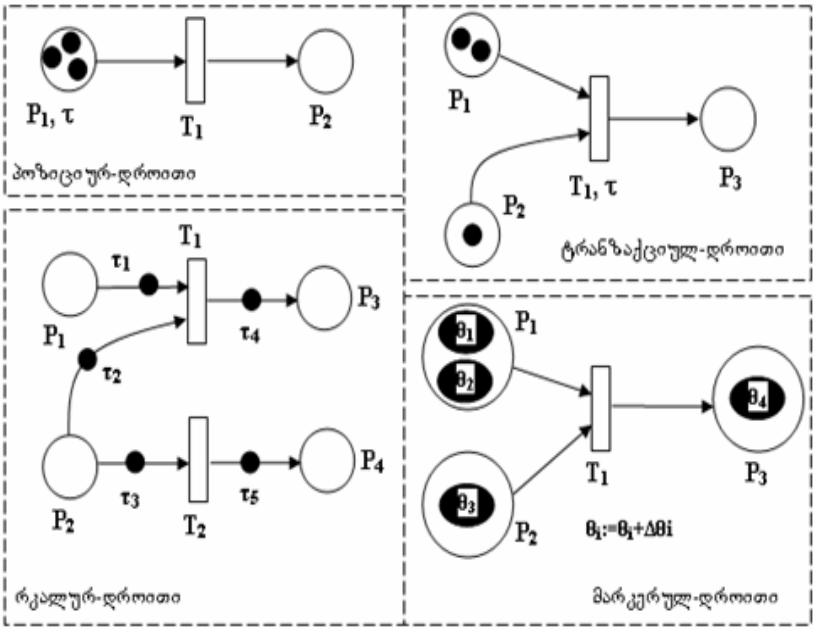
რკალურ-დროით პეტრის ქსელში დროითი დაყოვნების სიდიდეები რკალებს ენიჭება, განისაზღვრება რკალში მარკერის „მოგზაურობის“ დრო და გადასასვლელის გახსნა შესაძლებელია

მხოლოდ მაშინ, როცა ყველა შემავალ რკალში მოძრავი მარკერი მიაღწევს გადასასვლელს.

მარკერთა „მოგზაურობა“ გადასასვლელისკენ იწყება მხოლოდ მაშინ, როცა გადასასვლელის გახსნა ნებადართული ხდება. გახსნის შემდგომაც ყოველ რკალს ენიშნება მასში მარკერის „მოგზაურობის“ დრო, სანამ იგი გამომავალ პოზიციას მიაღწევს.

მარკერულ-დროითი პეტრის ქსელი ყოველი მარკერისათვის ცალკე დაყოვნების დროის განსაზღვრას მოითხოვს. ამგვარი ტიპი მოხერხებულია დროითი პრიორიტეტების მოდელირებისთვის.

დროითი პეტრის ქსელის სხვადასხვა ტიპები 4.19 ნახაზზეა მოცემული.



ნახ.4.19. დროითი პეტრის ქსელის ტიპები

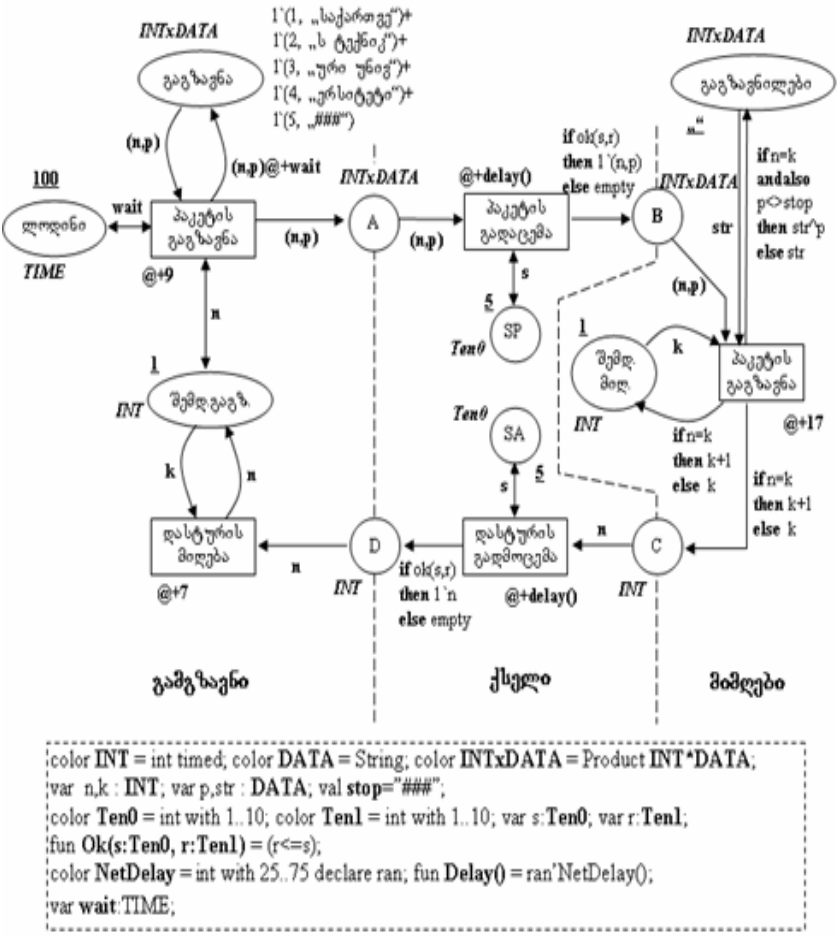
ცხადია, პეტრის ქსელების დროითი გაფართოების შემოტანა მოდელირებისას ახალ პრობლემებს წარმოშობს. მაგალითად,

ტრანზაქციულ-დროით პეტრის ქსელებში გასარკვევია, თუ როგორ უნდა ვმართოთ იმ გადასასვლელთა დაყოვნების დროები, რომლებმაც „შეჯიბრის“ შედეგად მარკერი დაკარგა და ხელახალ გააქტიურებას ელოდება გასახსნელად.

არსებობს ახალი, დროითი დაყოვნების განსაზღვრის 2 ვარიანტი: დაფიქსირდება მარკერის დაკარგვისას დარჩენილი დრო (**Continue**) და გადასასვლელის შემდგომი გააქტიურებისას დროის „ჩამოყრა“ დარჩენილი დროიდან გაგრძელდება ან დროითი დაყოვნების საწყისი მნიშვნელობა ხელახლა განისაზღვრება (**Restart**). ამ ვარიანტებზე დაყრდნობით მთლიანად ტრანზაქციულ-დროითი პეტრის ქსელებითვის დროით გაფართოებათა მოდიფიცირების 3 სტრატეგია განისაზღვრება:

- **სრული რესტარტი (Resampling)** – ნებისმიერი გადასასვლელის გახსნისთანავე ქსელის ყველა გადასასვლელის დაყოვნების დრო თავიდან განისაზღვრება, არანაირი ინფორმაცია არ ინახება
- **ნაწილობრივი რესტარტი (Enabling Memory)** – მარკერწართმეული გადასასვლელების დაყოვნების დრო თავიდან განისაზღვრება (რესტარტი), ხოლო დანარჩენები (რომლებიც გააქტიურებულია) ჩვეულებრივად აგრძელებს დროის „ჩამოყრას“.
- **დროის შენახვა (Age Memory)** – გადასასვლელის გაშვებისას ყველა გადასასვლელის მიმდინარე დრო ინახება და გადასასვლელის შემდგომი გააქტიურებისას დროითი დაყოვნება შენახვის დროითი პუნქტიდან აგრძელებს შემცირებას.

უნდა აღინიშნოს, რომ დასაშვებია **ჰიბრიდული** პეტრის ქსელების აგება დროითი და არადროითი ელემენტებით, რაც ხშირად სისტემების მოდელირების და ანალიზის ყველაზე ეფექტურ ინსტრუმენტს წარმოადგენს. დროითგაფართოებიანი, ჰიბრიდული, ფერადი პეტრის ქსელის კომპლექსური მაგალითი 4.20 ნახაზზეა გამოსახული.



ნახ.4.4.20. დროითი ფერადი პეტრის ქსელი მარტივი ქსელური პროტოკოლისთვის

4.5.4. სტოქასტური პეტრის ქსელები

ტრანზაქციულ-დროით პეტრის ქსელებს, სადაც გადასასვლელის დაყოვნების დრო შემთხვევით განაწილებულ ექსპონენციალურ ფუნქციას წარმოადგენს, **ალბათური** ანუ **სტოქასტური პეტრის ქსელები** (**Stochastic Petri Nets**) ეწოდება.

სტოქასტური პეტრის ქსელი, რომელიც **დროითთან** ერთად **არადროით (მყისიერ)** გადასავლელებსაც შეიცავს, **განზოგადებულ სტოქასტურ პეტრის ქსელს** (**Generalized Stochastic Petri Nets**) წარმოადგენს [19]. ამგვარი ქსელის ქცევა **ალბათური** (მაგალითად, **მარკოვის**) **პროცესებით** აღიწერება.

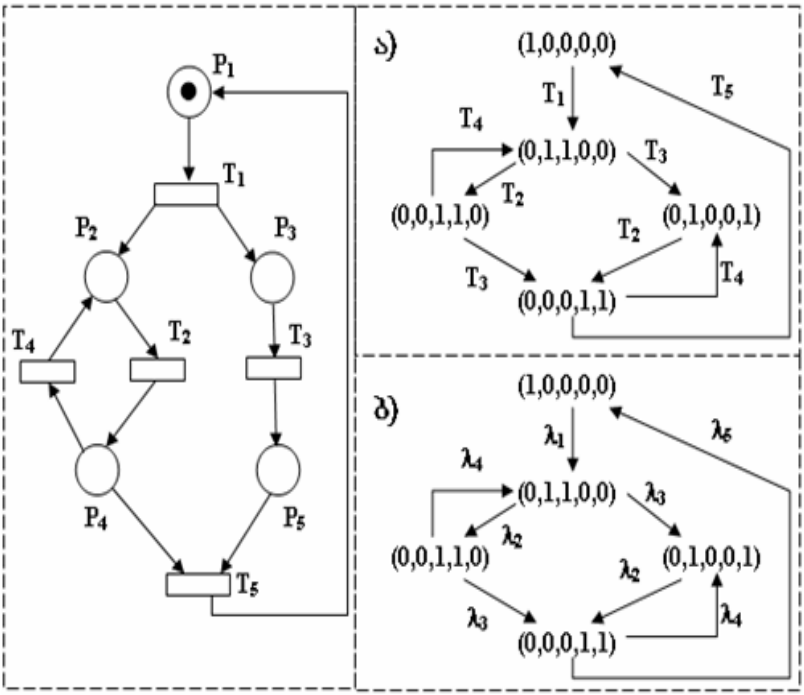
მათემატიკურად სტოქასტური პეტრის ქსელი მიიღება პეტრის ქსელის განსაზღვრებაზე $\Lambda = \{\lambda_1, \lambda_2, \ldots \lambda_m\}$ სიმრავლის დამატებით, სადაც გადასასვლელთა გაშვების დრო არის ექსპონენციალურად განაწილებული და შემთხვევითი λ_i სიდიდის განაწილებაა:

$$F_{\lambda_i}(x) = 1 - e^{-\lambda_i x}$$

სტოქასტური პეტრის ქსელის მაგალითი მოცემულია 4.21 ნახაზზე, სადაც გადასასვლელი T_1 ნებადართულია $M_0=(1,0,0,0,0)$ საწყის მარკირებაში.

გადასასვლელის დაყოვნების დრო ექსპონენციალურადაა განაწილებული და დამოკიდებულია λ_1 სიდიდეზე (გადასასვლელის კოეფიციენტი), ისე რომ გადასასვლელის გახსნის საშუალო დრო არის $\frac{1}{\lambda_1}$.

T_1-ის გახსნის შემდეგ მიიღება მარკირება $M_1=(0,1,1,0,0)$. ახლა პარალელურად ნებადართულია გადასასვლელები T_2 და T_3. თუ პირველად გაიხსნა T_2, მაშინ მიიღება მარკირება $M_2 = (0,0,1,1,0)$, ხოლო თუ T_3, მაშინ - $M_3 = (0,1,0,0,1)$.



ნახ.4.21. სტოქასტური პეტრის ქსელი: ა - მიღწევად მარკირებათა „სიით“; ბ - ეკვივალენტურ მარკოვის ჯაჭვით

მომდევნო მარკირებები უკვე იმაზეა დამოკიდებული, „შეჯიბრს“ რომელი გადასასვლელი მოიგებს.

ალბათობა იმისა, რომ პირველად $\mathbf{T_2}$ გადასასვლელი გაიხსნება, არის:

$$P[T_2] = P[\lambda_2 < \lambda_3] = \int_0^{\infty} \left(\int_0^{x} \lambda_2 e^{-\lambda_2 y} dy \right) \lambda_3 e^{-\lambda_3 x} dx =$$

$$= \int_0^{\infty} \left(1 - e^{-\lambda_2 x}\right) \lambda_3 e^{-\lambda_3 x} dx = \frac{\lambda_2}{\lambda_2 + \lambda_3}$$

ანალოგიურად, $\mathbf{T_3}$ გადასასვლელისთვის გვექნება:

$$P[T_3] = \frac{\lambda_3}{\lambda_2 + \lambda_3}.$$

ამ ფორმულებით ცხადი ხდება ისიც, რომ მარკირებათა ცვლილების ალბათობები გარკვეულ წინა მარკირებებში ყოფნის დროზე („წინაისტორიაზე") არ არის დამოკიდებული.

სტოქასტური პეტრის ქსელების წარმოდგენა და რაოდენობრივი ანალიზი შეიძლება შესაბამისი **მარკოვის პროცესების** ანალიზით განხორციელდეს, რაც ასევე 4.21 ნახაზზეა ასახული. ამ მძლავრი მათემატიკური აპარატის ინტეგრაცია სტოქასტურ პეტრის ქსელებს მიმზიდველ მოდელირების საშუალებად აქცევს, განსაკუთრებით **კონფლიქტების** მოდელირებისთვის.

4.5.4. ობიექტური პეტრის ქსელები

Object Petri Nets (და მისი გაფართოებები: **ობიექტ-ორიენტირებული** და **მაჩვენებლიანი** პეტრის ქსელები) პეტრის ქსელების თეორიისა და ობიექტ-ორიენტირებული დაპროგრამების თეორიის შეჯერებით მიღებული პეტრის ქსელის ტიპია [19]. ობიექტური პეტრის ქსელი ერთი **სისტემური** და რამდენიმე **ობიექტური** ქსელისგან შედგება, სადაც ობიექტური ქსელები **მარკერთა** როლში გამოდის. ფაქტობრივად, მიიღება პეტრის ქსელების სიმრავლე ერთი პეტრის ქსელის პოზიციებში.

ობიექტური ქსელები **ელემენტარულ სისტემურ ქსელებს** წარმოადგენს, ხოლო სისტემური ქსელი **მაღალი დონის პეტრის ქსელია**, რომლის პოზიციებშიც დაშვებულია როგორც ობიექტური ქსელების, ასევე ჩვეულებრივი შავი მარკერების არსებობა, ოღონდ არა ერთსა და იმავე პოზიციაში. შესაბამისად, რკალის ანოტაცია შეიძლება იყოს ნატურალური რიცხვი შავი მარკერებისთვის ან

ობიექტური ქსელების განსაზღვრულ იდენტიფიკატორთა სიმრავლე.

სისტემური ქსელის ყოველ გადასასვლელს შეუძლია ობიექტური ქსელის გადატანა **არაუმეტეს** ერთი შემავალი პოზიციიდან **არაუმეტეს** ერთ გამომავალ პოზიციაში. ამასთანავე ერთ გაშვებაზე არაუმეტეს ერთი ობიექტური ქსელის გადატანაა ნებადართული.

ობიექტურ პეტრის ქსელებს სხვა ტიპის პეტრის ქსელებისგან გადასასვლელის როლის ზრდაც გამოარჩევს: სისტემური და ობიექტური ქსელების ზოგიერთ გადასასვლელს ემატება სპეციალური ფუნქცია, რომელსაც **ინტერაქცია** ეწოდება. ინტერაქცია 2 ტიპისაა: **სისტემ-ობიექტური** და **ობიექტ-ობიექტური**. პირველი სისტემური და ობიექტური ქსელების გადასასვლელთა სინქრონულ ურთიერთობას უზრუნველყოფს, მეორე – ობიექტური ქსელების ურთიერთსინქრონიზაციას.

სისტემ-ინტერაქციული გადასასვლელის გაშვების წესი შემდეგია: თუ სისტემური ქსელის გადასასვლელი ინტერაქციულია და მისი ინტერაქცია ქსელში მარკერის სახით მოძრავი ობიექტური ქსელის ნებადართულ ინტერაქციას თანხვდება, მაშინ სისტემური ქსელის ინტერაქციული გადასასვლელის გაშვებისას ქსელში მოძრაობის პარალელურად გაიხსნება ობიექტური ქსელის ინტერაქციული გადასასვლელიც.

სხვა შემთხვევაში (თუ ინტერაქციები არ თანხვდება, ან სისტემური ან ობიექტური ქსელის გადასასვლელები ინტერაქციებს არ შეიცავს), ობიექტური ქსელი სისტემურში უცვლელი სახით გადაადგილდება. სწორედ ამგვარი მიდგომა განაპირობებს ობიექტური პეტრის ქსელების **ობიექტ-ორიენტირებულ** ხასიათს.

ზემოთ განხილული ფერადი პეტრის ქსელები სტრუქტურული დაპროგრამების თეორიასთან მჭიდრო კავშირში იმყოფება, შესამაბისად, შესაძლებელია ფერადი პეტრის ქსელებიდან მოქნილი გადასვლა ობიექტურ პეტრის ქსელებზე (როგორც სტრუქტურულიდან ობიექტორიენტირებული დაპროგრამების იდეოლოგიაზე) გარკვეული ახალი თვისებების შემოტანით.

რამდენიმე მარტივი შესაბამისობა ობიექტ-ორიენტირებულ დაპროგრამებასა და ობიექტურ პეტრის ქსელებს შორის 4.1 ცხრილშია მოცემული.

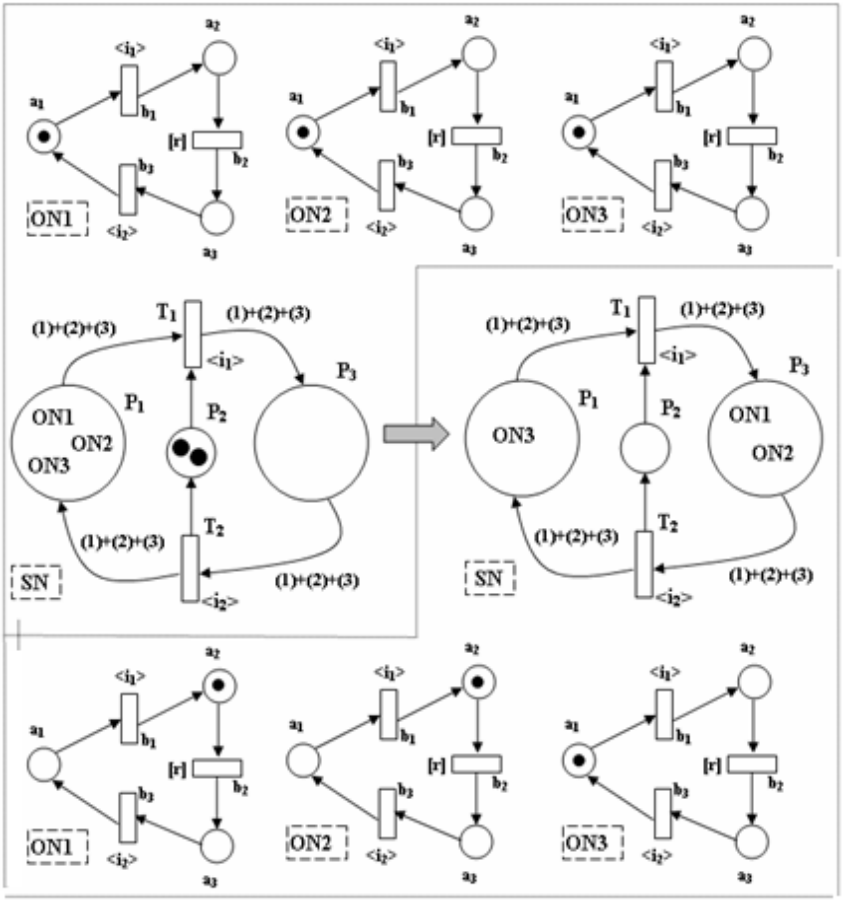
ობიექტ-ორიენტირებული დაპროგრამებისა და ობიექტური პეტრის ქსელების ეკვივალენტური ელემენტები **ცხრ.4.1**

ობიექტ-ორიენტირებული დაპროგრამება	ობიექტური პეტრის ქსელები
პროგრამული მოდული	სისტემური ქსელი
კლასი	ობიექტური ქსელის განსაზღვრება
ობიექტი	ობიექტური ქსელი კონკრეტული მარკირებით
ცვლადი	სისტემური ქსელის მარკერი
კლასის წევრი-ცვლადი	ობიექტური ქსელის მარკერი
გარე ფუნქცია	სისტემური ქსელის ინტერაქციული გადასასვლელი
კლასის წევრი-ფუნქცია	ობიექტური ქსელის გადასასვლელი

გავავლოთ პარალელი: გარე ფუნქციას (სისტემური ქსელის გადასასვლელი) კლასის წევრი-ცვლადის (ობიექტური ქსელის მარკერი) მოდიფიცირება (სხვა პოზიციაში გადანაცვლება) შეუძლია მხოლოდ მოცემული კლასის ობიექტის (მარკირებული ობიექტური ქსელი) გავლით, შესაბამისი წევრი-ფუნქციის გამოძახებით (ინტერაქციული გადასასვლელის გაშვება), როგორც ეს 4.22 ნახაზზეა მოცემული.

სისტემური ქსელის **(SN)** გადასასვლელი $\mathbf{T_1}$ და ობიექტური ქსელების **(ON1, ON2, ON3)** $\mathbf{b_1}$-გადასასვლელები შეიცავს ერთნაირ **სისტემ-ობიექტურ ინტერაქციებს** ($\mathbf{<i_1>}$), რაც ნიშნავს, რომ თუ $\mathbf{T_1}$-ის გახსნისას ობიექტური ქსელი **ON1** გადაადგილდება, მაშინ იგი გამოიყენებს ერთ შავ მარკერს $\mathbf{P_2}$ პოზიციიდან და გადასასვლელის გახსნისას $\mathbf{P_1}$-დან მოხვდება $\mathbf{P_3}$ პოზიციაში და ამავდროულად ობიექტურ ქსელ **ON1**-ში მარკერი $\mathbf{a_1}$ პოზიციიდან $\mathbf{a_2}$-ში მოხვდება $\mathbf{b_1}$-ის გახსნის შედეგად.

ობიექტურ ქსელებში გადასასვლელ $\mathbf{b_2}$-ისთვის განსაზღვრულია **ობიექტ-ობიექტური ინტერაქცია [r],** რომელიც ობიექტურ ქსელებს სისტემურ ქსელში პოზიციის შეუცვლელად საკუთარი შიდა მარკერების გადაადგილების საშუალებას აძლევს.



ნახ.4.22. სისტემური და ობიექტური ქსელების მარკირებათა ცვლილებები

მაგალითად, 4.22 ნახაზის ქვედა ნაწილში ობიექტური ქსელები **ON1** და **ON2** სისტემური ქსელის $\mathbf{P_3}$ პოზიციაში იმყოფება და რადგან მათი $\mathbf{b_2}$ გადასასვლელები ნებადართულია, ისინი გაიშვება კიდეც სინქრონულად $\mathbf{P_3}$ პოზიციის დატოვების გარეშე, რის შემდეგაც უკვე სისტემური ქსელის $\mathbf{T_2}$ გადასასვლელიც ნებადართული გახდება.

V თავი:

პრაქტიკული ამოცანების გადაწყვეტის მაგალითები რიგების თეორიის და პეტრის ქსელების გამოყენებით

5.1. სერვისული რესურსების მართვის მახასიათებლების კვლევა

მასობრივი მომსახურების თეორიის მიხედვით, პირობითად „მომსახურე ორგანოს" სახით შეიძლება განვიხილოთ კორპორაციული ორგანიზაციის (ბანკი, შემოსავლების სამსახური ან სხვ.) თანამშრომელი (მომხმარებელთან უშუალო კონტაქტი) ან კომპიუტერული ქსელის სერვერზე განთავსებული სერვისები (პროგრამული პროდუქტები მონაცემთა ბაზებით). ორივე შემთხვევაში პროცესი მსგავსი მოდელით აიგება (მოთხოვნების ნაკადი, მომსახურების დრო, რიგების სიგრძე და ა.შ.), ოღონდაც თვით ამ მაჩვენებელთა მნიშვნელობები იქნება განსხვავებული.

ასეთი მულტიპროცესორული ქსელური კონფიგურაციის სისტემების დაპროექტებისას საჭიროა მრავალი მახასიათებლის გათვალისწინება, რომელთა ოპტიმალური მნიშვნელობების შერჩევა ძალზე მნიშვნელოვანია და ამავე დროს რთულიც. ამ სიდიდეთა ოპტიმიზაცია არა მარტო გაზრდის კომპიუტერული ქსელის წარმადობას, არამედ შეამცირებს მის შესაქმნელად საჭირო ხარჯებსაც. მნიშვნელოვანია გავითვალისწინოთ ისეთი მომენტები, როგორიცაა სიმძლავრეების, საერთო რესურსების და ა.შ. ოპტიმალური განაწილება.

კომპიუტერულ ქსელებში მიმდინარე მოვლენების (დინამიკური პროცესების) მოდელირებისათვის მოსახერხებელია პეტრის ქსელების გამოყენება, რაოდენობრივი მახასიათებლების ანალიზისათვის კი - მასობრივი მომსახურების სისტემების თეორია [53]. ჩვენი მიზანია შევიმუშავოთ სერვის-ორიენტირებული არქიტექტურის კომპიუტერული სისტემისთვის სერვისების რეალიზაციის ალგორითმული სქემები და შესაბამისი პროგრამული

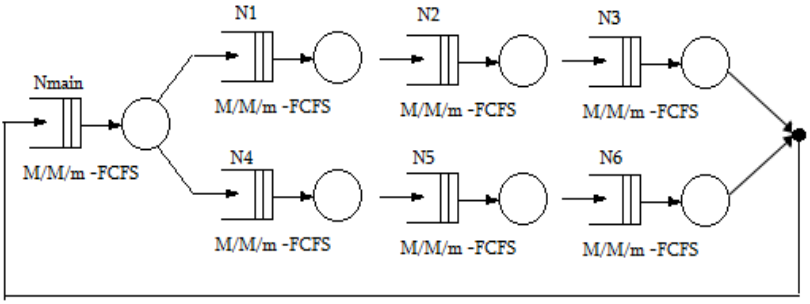
პაკეტები, რომელთა დანიშნულებაცაა ქსელის მომხმარებელთა მოთხოვნების დაკმაყოფილება ამ სერვისებით და კომპიუტერული ქსელის სიმძლავრეების ანალიზი და მათი ოპტიმალური განაწილება.

5.1.1. პროცესების კვლევა სტატიკურ რეჟიმში მასობრივი მომსახურების მეთოდებით

პირველ ეტაპზე ავაგოთ და გამოვიკვლიოთ განაწილებული სისტემის სერვისული რესურსების მართვის პროცესის მასობრივი მომსახურების მოდელი სტაციონარული რეჟიმისათვის [53].

განვიხილოთ კორპორაციული ქსელის მოდელი, სადაც არის რამდენიმე მომხმარებელი და რამდენიმე სერვერი (მომსახურე) სერვისებით. დავუშვათ, რომ სერვერთაგან ერთ-ერთი ასრულებს გამანაწილებლის ფუნქციას, ე.ი. იღებს მომხმარებლისაგან მოთხოვნას და უგზავნის მას მომსახურებისათვის იმ სერვერს, რომელიც თავისუფალია. თუ ყველა სერვერი დაკავებულია, მოთხოვნა დგება რიგში და ელოდება ერთ-ერთი მათგანის განთავისუფლებას.

სერვერი (Ni, i=1,6), მიიღებს რა მოთხოვნას გამანაწილებელი სერვერიდან (Nmain), ემსახურება მას გარკვეული სერვისებით და შედეგებს უბრუნებს ისევ გამანაწილებელ სერვერს, რომელიც, თავის მხრივ პასუხს აგზავნის მომხმარებელთან (ნახ.5.1).



ნახ.5.1

უნდა ვიგულისხმოთ, რომ მოთხოვნები მომხმარებლებისგან მოდის უწყვეტად, გარკვეული სიხშირით. თითოეული სერვერი ერთეული მოთხოვნის მომსახურებას ანდომებს გარკვეულ დროს. იმ შემთხვევაში როდესაც, მოთხოვნათა ფორმირების სიხშირე დიდია, გამანაწილებელ სერვერთან წარმოიქმნება რიგი. თუკი მოთხოვნათა ფორმირების სიხშირე ძალზე დიდია, ქსელი შეიძლება გადაიტვირთოს და ვეღარ შეძლოს ფუნქციონირება.

ჩვენი მიზანია ქსელის არსებული პარამეტრების მეშვეობით დავადგინოთ მისი მუშაობის კრიტიკული წერტილი, შევარჩიოთ ისეთი მახასიათებლები, რომლებიც უზრუნველყოფს მის ნორმალურ ფუნქციონირებას და შევქმნათ პროგრამული პროდუქტი, რომელიც ყოველივე ამას განახორციელებს. მასობრივი მომსახურების თეორიის თვალსაზრისით ზემოთ აღწერილი სისტემა არის M/M/m ტიპის.

განვიხილოთ მახასიათებლები და მათ შორის კავშირები, რომლებიც გააჩნია ქსელს. აქვე უნდა აღვნიშნოთ, რომ ქსელის ფუნქციონირებას განვიხილავთ სტაციონარულ რეჟიმში. ამ შემთხვევაში, როგორც ცნობილია, გარკვეულ იდეალიზაციასთან გვაქვს საქმე.

რეალურად დროის ყოველ t მომენტში სისტემაში არსებობს მოთხოვნათა რაღაც k რაოდენობა. ალბათობა იმისა, რომ დროის მოცემულ t მომენტში სისტემაში იმყოფება k მოთხოვნა, აღვნიშნოთ Pk(t)-თი. ჩვენ უნდა ვიგულისხმოთ, რომ t-ს ზრდასთან ერთად ალბათობა Pk(t) თანდათან მუდმივი ხდება. ამ შემთხვავაში Pk(t)-ს ნაცვლად შეიძლება გამოვიყენოთ Pk, რომელიც უკვე აღარ არის დროის ფუნქცია. ეს დაშვება არ გულისხმობს იმას, რომ სისტემა არ გადადის ერთი მდგომარეობიდან მეორეში, რა თქმა უნდა, დროის მიხედვით იცვლება ქსელში არსებული მოთხოვნების რაოდენობა, მაგრამ

ალბათობა იმისა, რომ სისტემაში საკმარისად დიდი დროის გასვლის შემდეგ იმყოფება k მოთხოვნა, გამოიხატება Pk-თი.

სერვისულ პროგრამულ პაკეტებში ფუნქციების დასაპროგრამებლად გამოვიყენოთ აღნიშნული კლასიკური მოდელები. ამგვარად, სერვერების რაოდენობით, შემოსულ მოთხვნათა ინტენსივობით და დროით, რომელსაც ანდომებს სერვერი თითოეული მოთხოვნის მომსახურებას, შეგვეძლება დავადგინოთ ქსელის სხვადასხვა მახასიათებელი.

აღვნიშნოთ მოთხოვნათა მოსვლის ინტენსივობა λ-თი, ხოლო თითოეული მოთხოვნის მომსახურების დროს Ts-ით. ამ შემთხვევაში ერგოდიულობის პირობა არის: $\lambda * Ts < 1$.

ქსელს გააჩნია შემდეგი მახასიათებლები:

1. მოძრაობის ინტენსივობა: $u = \lambda * Ts$.
2. სერვერის დატვირთვა: $\rho = u / m$.

იმისათვის, რომ სისტემა იყოს სტაბილური, სერვერს უნდა შეეძლოს თავი გაართვას მოთხოვნათა მოსვლის საშუალო ინტენსივობას, ეს კი ნიშნავს, რომ მოძრაობის ინტენსივობა უნდა იყოს სერვერთა რაოდენობაზე ნაკლები, ან რაც იგივეა, სერვერის დატვირთვა უნდა იყოს ერთზე ნაკლები, ე.ი. $u < m$ ან $\rho < 1$.

M/M/n სახის სისტემების კვლევისას მნიშვნელოვანი ადგილი უკავია ერლანგის ფუნქციას. ეს ფუნქცია განსაზღვრავს იმის ალბათობას, რომ ყველა სერვერი დაკავებულია, და იმავდროულად იმის ალბათობასაც, რომ მოსულ მოთხოვნას მოცდა მოუწევს. ერლანგის ფუნქციისთვის გამოვიყენებთ გამოსახულებას:

$$Ec(m,u) = (u^m / m!) / (u^m / m! + (1-\rho)\sum_{k=0}^{m-1}(u^m / m!))$$

მომხმარებლისთვის დიდი მნიშვნელობა აქვს მოთხოვნის რიგში დგომის (მოცდის) საშუალო დროს, იგი გამოითვლება ფორმულით:

$$Tw = \frac{Ec(m,u)Ts}{m(1-\rho)}$$

აუცილებელია განვსაზღვროთ მოთხოვნის სისტემაში ყოფნის საშუალო დრო: $Tq = Tw + Ts$.

ალბათობა იმისა, რომ მოთხოვნის სისტემაში ყოფნის დრო ნაკლებია t-ზე დამოკიდებულია u = m-1, თუ არა. თუ ეს პირობა სრულდება, მაშინ ადგილი აქვს შემდეგ ტოლობას:

$$P(\text{სისტემაში ყოფნის დრო}<t)=1-(1+\frac{t}{Ts}Ec(m,u))e^{\frac{-t}{Ts}}$$

წინააღმდეგ შემთხვევაში:

$$P(\text{სისტემაში ყოფნის დრო}<t)=1+\frac{B+Ec(m,u)}{B}e^{\frac{-t}{Ts}}+\frac{Ec(m,u)}{B}e^{-(m-u)\frac{t}{Ts}}$$

სადაც B=m-1-u.

დროის ყოველ მომენტში ქსელში იარსებებს მოთხოვნათა გარკვეული რაოდენობა. რაც ნაკლები მოთხოვნაა ქსელში, მით უკეთ ფუნქციონირებს იგი. ალბათობა იმისა, რომ ქსელში არის k მოთხოვნა არის Pk სადაც

$$P_k=\frac{u}{k!}P_0$$

როცა k <= m და

$$P_k=\frac{u^k}{m!m^{k-m}}P_0$$

როცა $k >= m.$

P_0 არის ალბათობა იმისა, რომ ქსელში საერთოდ არაა მოთხოვნა.

ეს რაც შეეხებოდა ალბათობებს. თვით სისტემაში არსებულ მოთხოვნათა რაოდენობა კი არის Lq, სადაც

$$Lq=u+\frac{pEc(m,u)}{1-\rho}$$

თუკი ქსელში არის m ან m-ზე ნაკლები მოთხოვნა, მაშინ იმ მოთხოვნების რაოდენობა, რომლებიც რიგში დგას 0-ის ტოლია, ხოლო თუ ვიცით, რომ x მოთხოვნა რიგში დგას, მაშინ მთლიანად სისტემაში იქნება x+m მოთხოვნა. ასე, რომ გვაქვს შემდეგი მახასიათებლები:

ალბათობა იმისა, რომ არცერთი მოთხოვნა არ იცდის:

$$\text{P(არცერთი მოთხოვნა არ იცდის)}=\sum_{k=0}^{m} P_{\text{k}}$$

ალბათობა იმისა, რომ x მოთხოვნა დგას რიგში:

$$\text{P(x მოთხოვნა იცდის)}=P_{x+m} \quad \text{სადაც } \text{x>m}$$

მომლოდინე მოთხოვნათა საშუალო რიცხვი:

$$Lw=\frac{pEc(m.u)}{1-\rho}$$

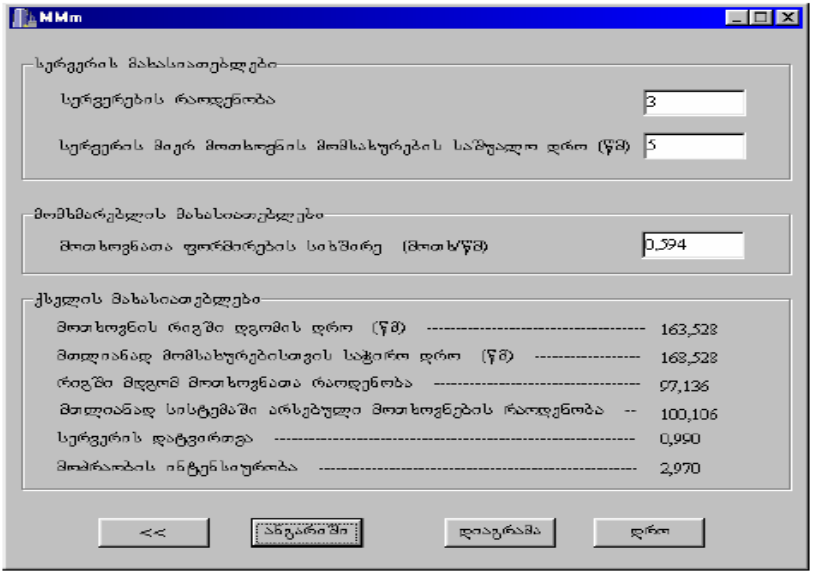
რეჟიმს. ჩვენ მიერ შექმნილი პროგრამული საშუალება სწორედ ამ სიდიდეებს და ფორმულებს იყენებს ქსელის პარამეტრების ანალიზისათვის და მათი ოპტიმალური მნიშვნელობის შერჩევისათვის.

იგი, იღებს რა ინფორმაციას ქსელში მოთხოვნების მოსვლის სიხშირეზე, სერვერთა რაოდენობასა და თითოეული მოთხოვნის მომსახურების დროზე, ანგარიშობს ისეთ პარამეტრებს როგორიცაა მოთხოვნის რიგში დგომის დრო, ბუფერში მოთავსებული მომლოდინე მოთხოვნათა რაოდენობა, სერვერის დატვირთვა და მოძრაობის ინტენსივობა, სხვადასხვა ალბათობები და ა.შ.

გარდა ამისა, გამოითვლის მოცემულ პირობებში ოპტიმალური მუშაობისათვის საჭირო პარამეტრებს და აგებს მათ შორის დამოკიდებულებათა გრაფიკებს.

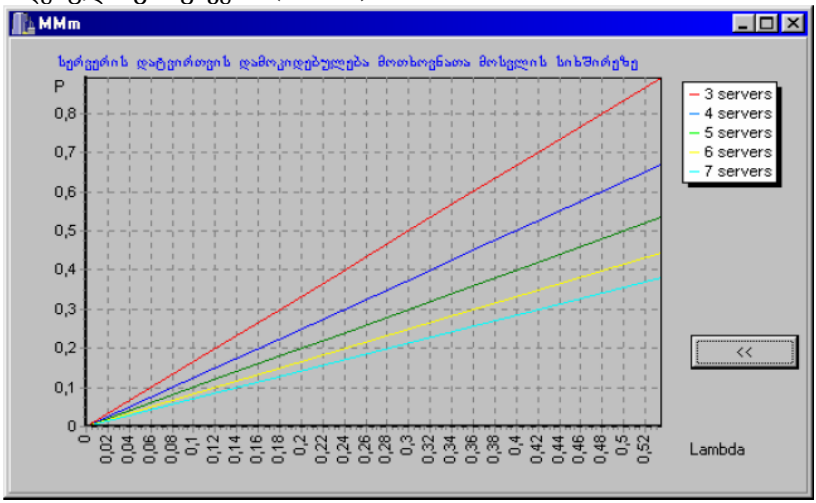
5.2 ნახაზზე მოცემულია C++ ენის ინსტრუმენტით აგებული მომხმარებლის ინტერფეისი, რომელიც მუშაობს ვიზუალური და ტრადიციული დაპროგრამების კომპონენტების რევერსული ტექნოლოგიით. როგორც ნახაზიდან ჩანს, მომხმარებელს შეუძლია შეიტანოს (და ცვალოს) სამი პარამეტრის მნიშვნელობა: სერვერების რაოდენობა, მომსახურების საშუალო დრო და მოთხოვნათა რაოდენობის ინტენსიურობა.

ღილაკით „ანგარიში“ სისტემა გაიანგარიშებს ქსელის ძირითად მახასიათებლებს, კერძოდ: სერვერის დატვირთვა, მოძრაობის ინტენსიურობა, მოთხოვნის რიგში დგომის დრო, მთლიანად მომსახურებისთვის საჭირო დრო, რიგში მდგომ მოთხოვნათა რაოდენობა, სისტემაში მყოფ მოთხოვნათა საერთო, რაოდენობა.



ნახ.5.2

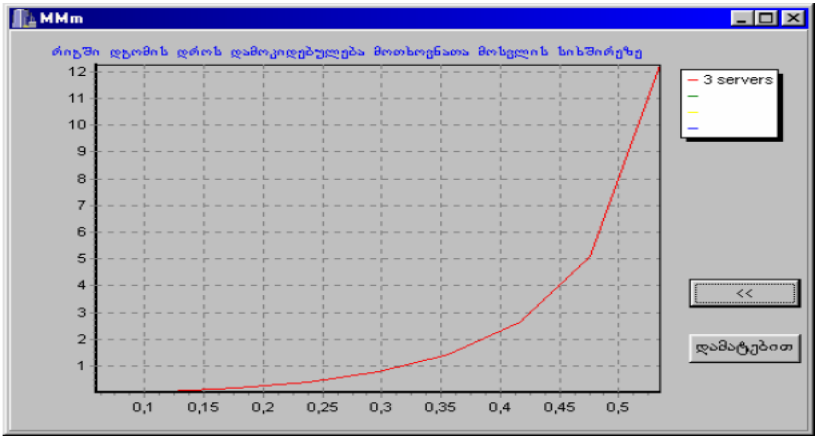
ღილაკით „დიაგრამა“ გამოიტანება გაანგარიშების შედეგად მიღებული გრაფიკები (ნახ.5.3).



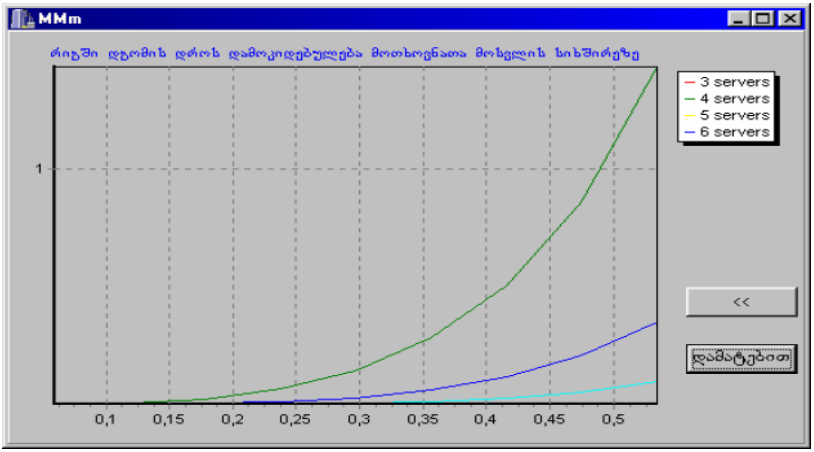
ნახ.5.3

მოცემულია პროგრამულად მიღებული დიაგრამა სერვერის დატვირთვის დამოკიდებულებისა მოთხოვნათა მოსვლის სიხშირეზე სერვერების სხვადასხვა რაოდენობისათვის (მაგალითად, 3-:-7).

5.4 და 5.5 ნახაზებზე მოცემულია დიაგრამები მოთხოვნათა რიგში დგომის დროის დამოკიდებულებისა მოთხოვნათა მოსვლის სიხშირეზე სერვერების სხვადასხვა რაოდენობის დროს (3-:-6). ბოლო დიაგრამაზე კარგად ჩანს, თუ როგორ იკლებს მოთხოვნათა რიგში დგომის დრო მომსახურე არხების მომატებით.



ნახ.5.4



ნახ.5.5

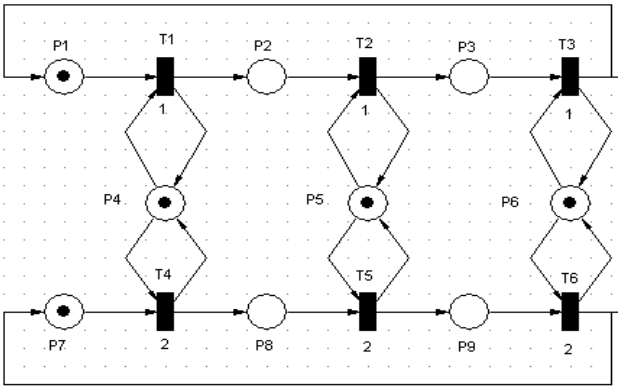
5.1.2. პროცესების კვლევა დინამიკურ რეჟიმში პეტრის ქსელებით

ახლა გავაანალიზოთ კომპიუტერული ქსელის მოქმედება დინამიკურ რეჟიმში. ამ შემთხვევაში სერვერების მიერ კლიენტთა მოთხოვნების დაკმაყოფილების პროცესი შეიძლება მოდელირებულ იქნას ტრანზიტული დროითი პეტრის ქსელის (Timed Transition Petri Net) საშუალებით [19].

პეტრის ქსელის დროითი გაფართოება ჩვენს შემთხვევაში იქნება ალბათური (სტოქასტური). ამგვარად, ასეთი ქსელის ანალიზი შესაძლებელია მარკოვის მეთოდების გამოყენებით, რომელშიც დრო ექსპონენციალურადაა განაწილებული [36,53].

სტოქასტური პეტრის ქსელის მისაღებად საჭიროა ,,პოზიცია-გადასასვლელების ქსელს" დაემატოს გადასასვლელთა გაშვების (დაყოვნების, მოლოდინის) დროთა მომენტები (მაგალითად, μ1, μ2, . . . , μn). განვიხილოთ კერძო მაგალითი კორპორაციული ქსელისათვის, ორი სერვერით და სამი კლიენტით.

5.6 ნახაზზე მოცემულია სტოქასტური პეტრის ქსელის საწყისი მდგომარეობა. მარკერის არსებობა S1(p1,p2,p3) და S2(p7,p8,p9) სერვერებში ნიშნავს მათ მზადყოფნაზე კლიენტების მომსახურებისათვის.



ნახ.5.6. სტოქასტური პეტრის ქსელის საწყისი მდგომარეობა

დავუშვათ, რომ C(p4,p5,p6) კლიენტის პოზიციებში მარკერები მუდმივადაა, ე.ი. მოთხოვნები არსებობს და ისინი ელოდება სერვერის მომსახურებას. როგორც აღვნიშნეთ, Tj გადასასვლელის გახსნის დროა (ანუ მომსახურების დაყოვნების დრო). Tj-ური გადასასვლელის გახსნის საშუალო დრო იქნება 1/μ, სადაც μ გადასასვლელის გახსნის ინტენსივობაა.

სისტემის მდგომარეობები, ანუ მარკერების სიმრავლე შეიძლება ასე ჩაიწეროს:

p პოზიციები და t გადასასვლელები:

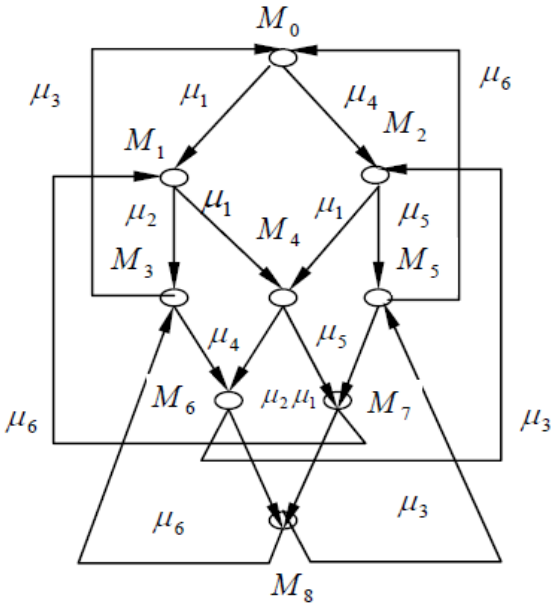
M1__100111001
M2__010111001
M3__100111100
M4__010111001
M5__100111010
M6__001111100
M7__010111010
M8__001111010

სადაც Mi, 0<=i<=K მდგომარეობებია (მარკირებები); Tj, 1<=j<=L გადასასვლელები; μj, 1<=j<=L – დაყოვნების დრო გადასასვლელის გასაღებად. ჩვენ შემთხვევაში m=3 და n=2, ამიტომ კომბინაცია იქნება $m^n = 9$.

გადასასვლელების გახსნის ორგანიზება, როცა სისტემა ყველა მდგომარეობას გადის ნაჩვენებია 5.7 ნახაზზე, რომელსაც პეტრის ქსელის მიღწევადობის გრაფს უწოდებენ.

ასეთი სტოქასტური პეტრის ქსელის რაოდენობრივი ანალიზი შეიძლება განხორციელდეს შესაბამისი მარკოვის პროცესების ანალიზით.

განვიხილოთ მარკოვის ჯაჭვის მაგალითი, რისთვისაც ამ ნახაზზე გრაფის რკალებზე მივამაგროთ გადასასვლელების გაშვების μ კოეფიციენტები.



ნახ.5.7. პეტრის ქსელის მიღწევადობის გრაფი

ჩვენთვის საინტერესოა დავადგინოთ სისტემის თითოეულ მდგომარეობაში გადასვლის ალბათობები, ამისათვის საჭიროა შევადგინოთ კოლმოგოროვის განტოლებათა სისტემა:

P5*μ6+P3*μ3-P0*(μ1+μ4)=0

P7*μ6+P0*μ1-P1*(μ2+μ4)=0

P0*μ4+P6*μ3-P2*(μ1+μ5)=0

P6*μ3+P0*μ4-P3*(μ1+μ5)=0

P1*μ4+P2*μ1-P4*(μ2+μ5)=0

P2*μ5+P8*μ3-P5*(μ1+μ6)=0

P3*μ4+P4*μ2-P6*(μ3+μ5)=0

P4*μ5+P5*μ1-P7*(μ2+μ6)=0

P6*μ5+P7*μ2-P8*(μ3+μ6)=0

P0+P1+P2+P3+P4+P5+P6+P7+P8=1

მაგალითად, თუ დავუშვებთ, რომ μ1=3, μ2=5, μ3=2, μ4=3, μ5=1, μ6=7, მაშინ ალბათობათა მნიშვნელობები, შესაბამისად იქნება:

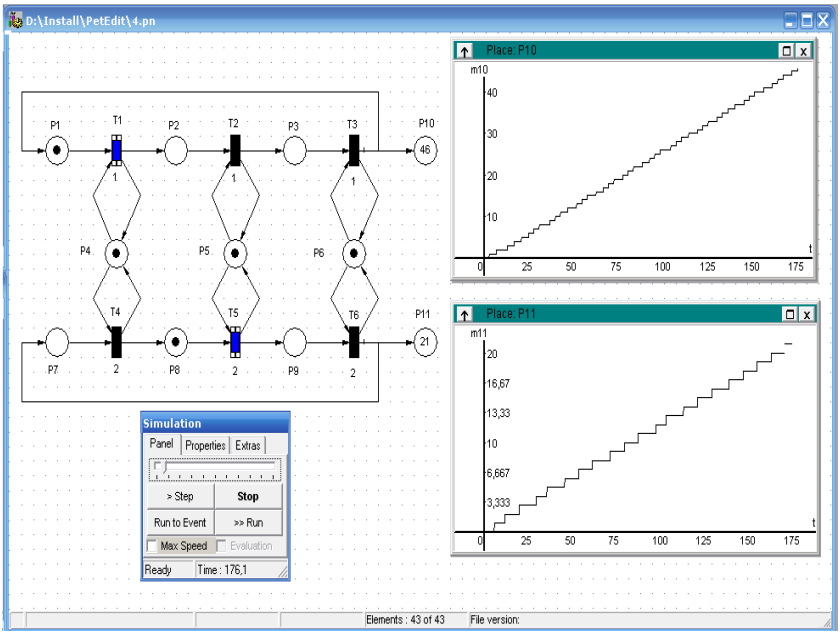
P0=0.11; P1=0.05; P2=0.07;
P3=0.26; P4=0.06; P5=0.02;
P6=0.37; P7=0.01; P8=0.05.

უნდა აღინიშნოს, რომ კომპიუტერული ქსელისათვის ერთი მდგომარეობიდან მეორეში გადასვლის ინტენსივობა μ არის სერვერის მიერ შესაბამისი კლიენტის მოთხოვნის მომსახურებისათვის საჭირო დროის შებრუნებული სიდიდე, ე.ი. 1/Ts. ამ განტოლებათა სისტემის ამოხსნით გაუსის მეთოდით მივიღებთ სისტემის ერთი მდგომარეობიდან მეორეში გადასვლის ალბათობებს $\mathbf{P_0, P_1, P_2, \ldots, P_8}$.

5.6 ნახაზზე წარმოდგენილი პეტრის ქსელის გრაფისთვის PetEdit რედაქტორში ავაგოთ შესაბამისი მოდელი. სერვერებისთვის შევირჩიოთ პირობითად განსხვავებული მწარმოებლურობა, კერძოდ ერთი ამუშავებს მოთხოვნებს 1 წმ-ში, მეორე კი - 2 წამში (ამ მნიშვნელობების ცვლილებით შესაძლებელია შემდგომი ექსპერიმენტების ჩატარება).

5.8 ნახაზზე ნაჩვენებია მიღებული პეტრის ქსელის მოდელი და სიმულაციის შედეგები. 10 და 11 პოზიციები ასახავს სერვერების მიერ შესრულებული პროცედურების ჯამურ რაოდენობას.

როგორც დიაგრამებიდან ჩანს, პირველი სერვერის სწრაფქმედების, ან სერვისების დამუშავების პროცედურების ხანგრძლივობა თითქმის ორჯერ ნაკლებია. ამიტომაც შედეგები მე-10 პოზიციაში ორჯერ მეტია. თუ სერვერული სისტემებისთვის მოხდება სერვისული ოპერაციების დამუშავების დროის წინასწარ განსაზღვრა, მაშინ შედეგებიც შესაბამისად აისახება.



ნახ.5.8. პეტრის ქსელის სიმულაციის რეჟიმი მახასიათებლებით

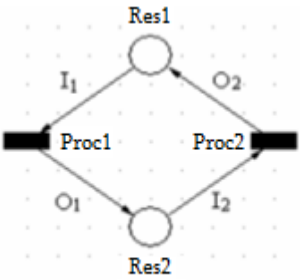
5.2. განაწილებული სისტემების რესურსების ადმინისტრირების ამოცანები

5.2.1. ჩიხური სიტუაციების მართვა

განიხილება ოპერაციულ სისტემების პლატფორმაზე მონაცემთა ბაზების ფაილების კოლექტიური გამოყენების დროს ჩიხური პროცესების არსებობისა და მათი გამორიცხვის შესაძლებლობანი. შემუშავებულია ასეთი პროცესების მართვის მოდელი პეტრის ქსელის ინსტრუმენტის საფუძველზე. ჩატარებულია ამ მოდელის იმიტაციური გამოკვლევა და აგებულია შესაბამისი დროითი მახასიათებლები [54,55].

თანამედროვე მულტიპროცესორულ სისტემებში, მათ შორის ლოკალურ კომპიუტერულ ქსელებში, რომლებიც გამოთვლითი რესურსების საერთო გამოყენების კონცეფციას ეყრდნობა, განსაკუთრებული მნიშვნელობა ენიჭება პროცესების ეფექტურად ორგანიზაციის საკითხს ჩიხური სიტუაციების აღმოსაფხვრელად.

პროცესი ჩიხურია (deadlock), თუ იგი ელოდება ისეთი ხდომილების შესრულებას, რომელიც არასოდეს მოხდება. ორი ან რამდენიმე პროცესი შეიძლება მოხვდეს ჩიხში, თუ თითოეული მათგანი აბლოკირებს რესურსებს (მაგალითად, მონაცემთა ბაზის ცხრილებს, ან მის ფრაგმენტებს), რომლებიც ესაჭიროება სხვა პროცესებს და თვითონ კი მოითხოვს ისეთ რესურსებს, რომლებიც ბლოკირებულია სხვა პროცესების მიერ. ოპერაციულ სისტემას ჩვენ განვიხილავთ როგორც გამოთვლითი რესურსების ადმინისტრატორს, ხოლო რესურსებად გვევლინება ცენტრალური პროცესორი, ოპერატიული მეხსიერება, დისკოები, ფაილური სისტემები, პროგრამები და მონაცემთა ბაზები, პრინტერები, ქსელური არხები და ა.შ.



ნახ.5.9. ჩიხური სიტუაცია

5.9 ნახაზზე ნაჩვენებია ელემენტარული ჩიხური ოპერაციის მაგალითი, ჩაწერილი პეტრის ქსელის გრაფით. აქ Proc1, Proc2 პროცესებია, ხოლო Res1, Res2 - რესურსები. პოზიცია-გადასასვლელთა შემაერთებელი რკალები შემდეგი დანიშნულ;ებითაა: I_1 : Res1-რესურსი გამოეყო Proc1-პროცესს; O_1: Proc1-პროცესი მოითხოვს Res2-რესურსს; I_2 : Res2-რესურსი გამოეყო Proc2-პროცესს; O_2: Proc2-პროცესი მოითხოვს Res1-რესურსს.

როგორც ნახაზიდან ჩანს, Proc1 პროცესს ბლოკირებული აქვს Res1 რესურსი და მუშაობის გასაგრძელებლად სჭირდება

Res2 რესურსი. Proz2 პროცესს კი პირიქით, ბლოკირებული აქვს Res2 რესურსი და მუშაობის გასაგრძელებლად სჭირდება Res1 რესურსი. ამგვარად, ორივე პროცესი იმყოფება მუდმივად მოლოდინის რეჟიმში.

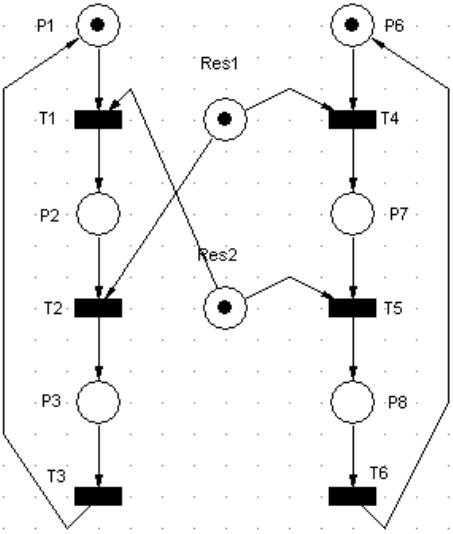
ჩიხური პროცესების არსებობისათვის ოთხი აუცილებელი პირობა იქნა განსაზღვრული [54]. ურთიერთგამორიცხვის (პროცესებს აქვს რესურსების მონოპოლური გამოყენების უფლება), დამატებითი რესურსების მოლოდინის (პროცესებს აქვს უკვე გამოყოფილი რესურსები, მაგრამ ელოდება დამატებითს), არაგადანაწილებადობის (პროცესებს არ შეიძლება ჩამოერთვას რესურსები მათ საბოლოო შესრულებამდე) და წრიული მოლოდინის (არსებობს პროცესების წრიული ჯაჭვი, რომელშიც ყოველი პროცესი აბლოკირებს ერთ ან რამდენიმე რესურსს, რომელიც ესაჭიროება ჯაჭვში მომდევნო პროცესს).

ჩიხური პროცესების მართვის პრობლემა ოპერაციულ სისტემებში განიხილება შემდეგი ამოცანების გადაწყვეტით:

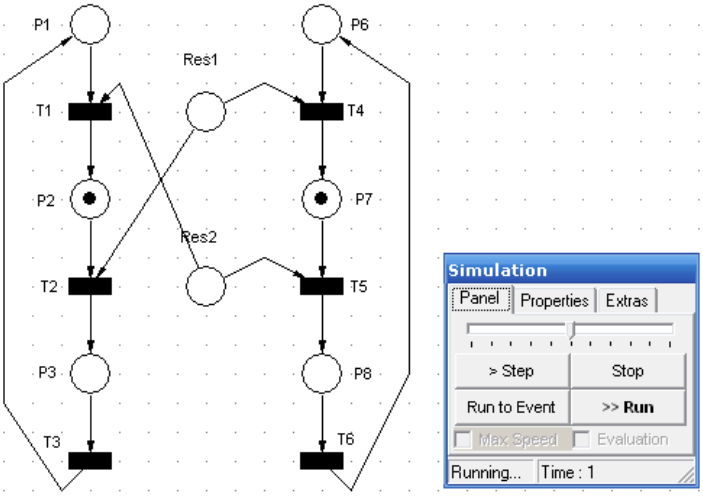
- ჩიხების თავიდან აცილება. თუ ჩიხების არსებობის აღწერილი პირობებიდან მოხერხდება ერთი ან რამდენიმე პირობის მოხსნა, მაშინ შესაძლებელია ჩიხების აღმოცენების თავიდან აცილება;
- ჩიხების გერდის ავლა. აქ პრინციპულად დასაშვებია ჩიხური სიტუაციის არსებობა, მაგრამ მისი მოახლოვებისას მიიღება შესაბამისი გამაფრთხილებელი ზომები. ამ დროს შესაძლებელია რესურსების უფრო რაციონალური გამოყენება, ვიდრე წინა შემთხვევაში;
- ჩიხების აღმოჩენა. ამ დროს ჩიხური სიტუაციები ლოკალიზდება და ოპერატორს მიეწოდება სათანადო ინფორმაცია მათ შესახებ;
- ჩიხური სიტუაციის აღდგენა. ესაა ჩიხური სიტუაციიდან გამოსვლა მიმდინარე მუშაობის შედეგების გარკვეული დანაკარგებით.

ახლა განვიხილოთ კონკრეტული შემთხვევა ორი პროცესისთვის (Proc1, Proc2), რომლებიც ორი საერთო რესურსის

(Res1, Res2) გამოყენებით ასრულებენ გარკვეულ პროცედურათა მიმდევრობას. 5.10 ნახაზზე წარმოდგენილია შესაბამისი პეტრის ქსელის გრაფი საწყის და შუალედურ (ჩიხურ) მდგომარეობაში.



ა)

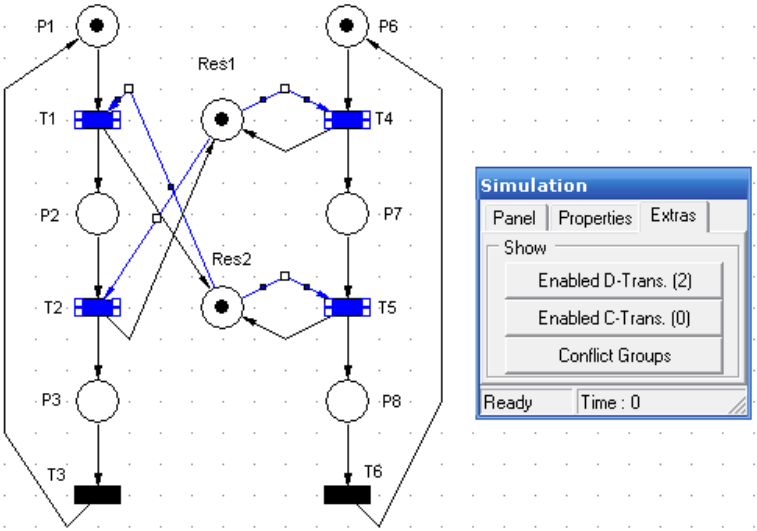


ბ)

ნახ.5.10. საწყისი მდგომარეობა (ა), ჩიხური სიტუაცია ბლოკირებული რესურსებით (ბ)

ორივე რესურსი ბლოკირებულია შუალედურ პროცედურაში და ელოდება მეორე რესურსს. ამ შემთხვევაში პეტრის ქსელი უძლურია პროცესის გასაგრძელებლად. საჭიროა დამატებითი რკალების შემოტანა, რომლებიც უზრუნველყოფს ბლოკირებული რესურსების გათავისუფლებას.

5.11 ნახაზზე დამატებულია აღნიშნული რკალები. აქვე ნაჩვენებია კონფლიქტურ გადასასვლეთა ჯგუფი.

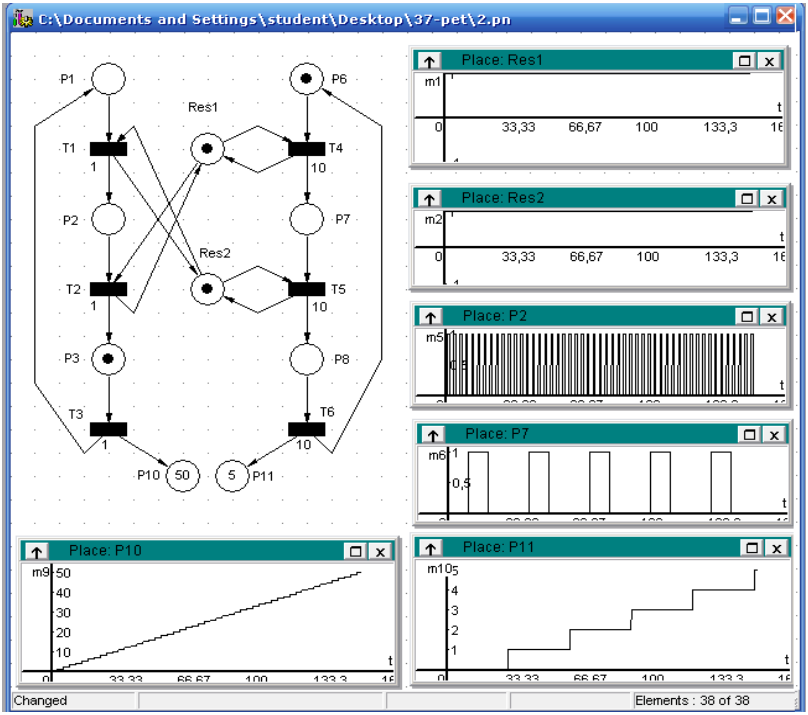


ნახ.5.11. კონფლიქტური ჯგუფის გადასასვლელები

5.12 ნახაზზე ნაჩვენებია პეტრის ქსელის გრაფის იმიტაციის პროცესის შედეგები, მათი ცალკეული პოზიციების დროითი დიაგრამებით. სქემაზე Proc1-ის გადასასვლელების (პროცედურათა შესრულების) დაყოვნების დრო არის, პირობითად, 1 წმ, ხოლო Proc2-ის 10 წმ.

მონაცემთა განაწილებული ბაზების ადმინისტრირებისათვის ნებისმიერი ოპერაციული სისტემის პლატფორმაზე, ჩიხური პროცესების მართვა, ანუ აღმოჩენა და მისი დროული გამორიცხვა შესაძლებელია შესაბამისი პროცესების მოდელირებით პეტრის ქსელის გრაფო-ანალიზური ინსტრუმენტის საფუძველზე, რაც

ზემოთ იყო ილუსტრირებული. აგებული მოდელის იმიტაციური პროცესის გამოკვლევა იძლევა შესაბამის დროითი მახასიათებლებს გარკვეული დასკვნების გასაკეთებლად.

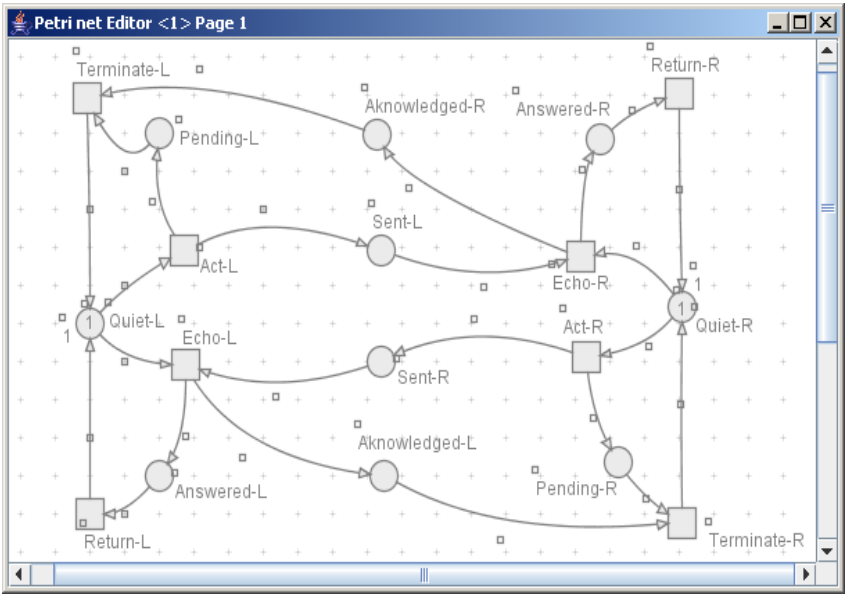


ნახ.5.12. პროცესების შესრულების დროითი მახასიათებლები

5.2.2. ჩიხების აღმოფხვრის ალგორითმები

როგორც აღვნიშნეთ, ქსელური სისტემების მოდელირების პროცესში ერთერთ მთავარ პრობლემას ჩიხური სიტუაციების წარმოშობა წარმოადგენს. განაწილებულ სისტემაში **ჩიხი** წარმოიშობა მაშინ, როცა ორი ქვესისტემა ერთდროულად **ურთიერთლოდინის** მდგომარეობაში იმყოფება ან კონფლიქტია რომელიმე რესურსისთვის [19].

პირველი შემთხვევა სიმპტომატურია ოპერაციული სისტემების პროცესებისა და ქსელური პროტოკოლებისთვის. 5.13-ა ნახაზზე მოცემულია ელემენტარული პეტრის ქსელი ტიპური შემთხვევისათვის („ცნობების გაგზავნა-მიღების ამოცანა").



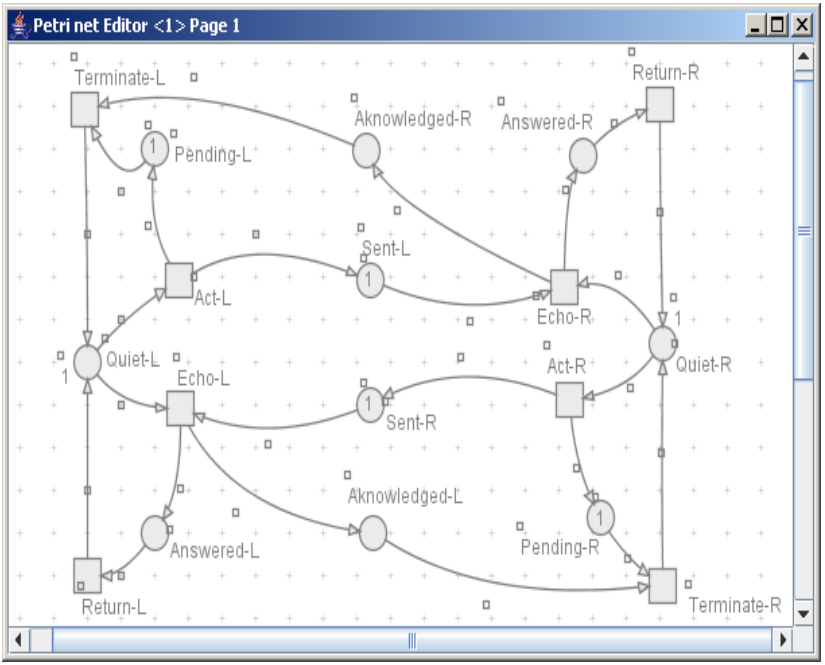
ნახ.5.13-ა. „ცნობათა გაგზავნა-მიღების" ამოცანის საწყისი მარკირება

აქ ორი პროცესი (ან ქსელის კვანძი) ერთმანეთს ცნობებს (მესიჯებს) უგზავნის ინფორმაციის გასაცვლელად, ამასთან ინფორმაციის მორიგი პორციის გასაგზავნად აუცილებელი პირობაა დასტურის მიღება წინა შეტყობინების წარმატებით მისვლის შესახებ.

ინფორმაციის გაგზავნის და დასტურის მიღების ოპერაციათა ერთობლიობა არის სეანსი. L-პროცესი გზავნის ცნობას (გადასასვლელი Act-L) და გადადის ლოდინის მდგომარეობაში (პოზიცია Pending-L). პროცესი R შეტყობინების მიღებისთანავე გამოდის პასიური მდგომარეობიდან (Quiet-R) და გზავნის მიღების დასტურს (გადასასვლელი Echo-R), რის შემდეგაც უბრუნდება პასიურ მდგომარეობას (გადასასვლელი Return-R).

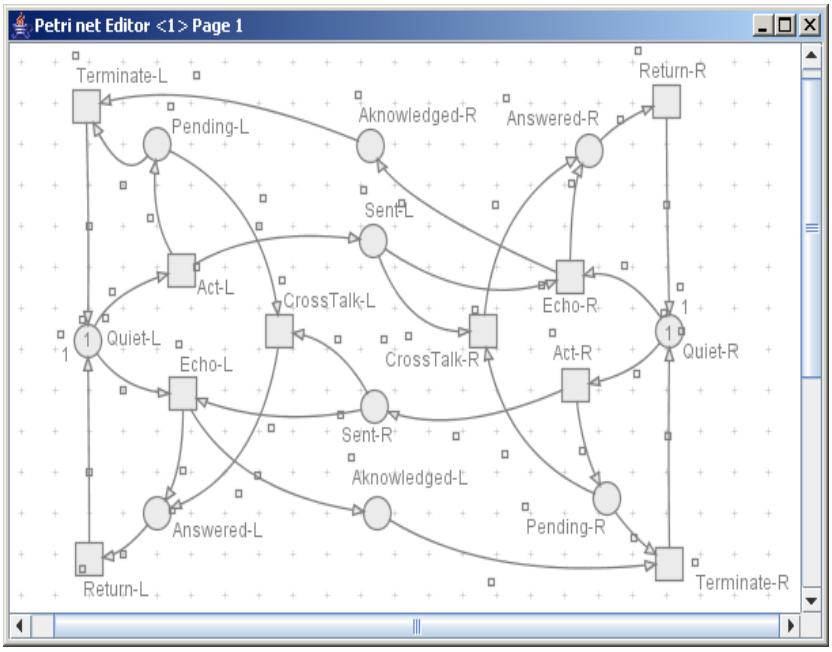
დასტურის მოსვლა გაწყვიტავს L-პროცესის ლოდინის მდგომარეობას (გადასასვლელი Terminate-L) და დააბრუნებს მას პასიურ მდგომარეობაში (პოზიცია Quiet-L). ორივე პროცესის პასიურ მდგომარეობაში დაბრუნებით პროცესი სრულდება.

ამგვარ სისტემაში ჩიხი წარმოიშობა მაშინ, როცა ორივე აგენტი ერთსა და იმავე პროცესში გადაწყვეტს ცნობის გაგზავნას (ნახ.5.13-ბ). ამ დროს ორივე პროცესი უსასრულოდ ელის ადრესატისგან დასტურის მოსვლას (პეტრის ქსელის ყველა გადასასვლელი ბლოკირებულია).



ნახ.5.13-ბ. ჩიხი

ჩიხის თავიდან ასაცილებლად შემუშავებულია ალგორითმი **CrossTalk**, რაც პეტრის ქსელში 2 სპეციალური გადასასვლელის ჩამატებას გულისხმობს (თითო-თითო პროცესისთვის), რომელთაც სისტემის ჩიხიდან გამოყვანა შეუძლია (ნახ.5.14).



ნახ.5.14. ჩიხის აღმოფხვრის CrossTalk-ალგორითმი

გადასასვლელი **CrossTalk** ერთ პროცესს აძლევს უფლებას მეორის მდგომარეობა შეამოწმოს და თავად ლოდინის მდგომარეობაში მყოფმა, თუ დაადგინა, რომ მეორე პროცესიც იცდის, გაწყვიტოს ლოდინი და პირდაპირ დასტურის მიღების მდგომარეობაში გადავიდეს.

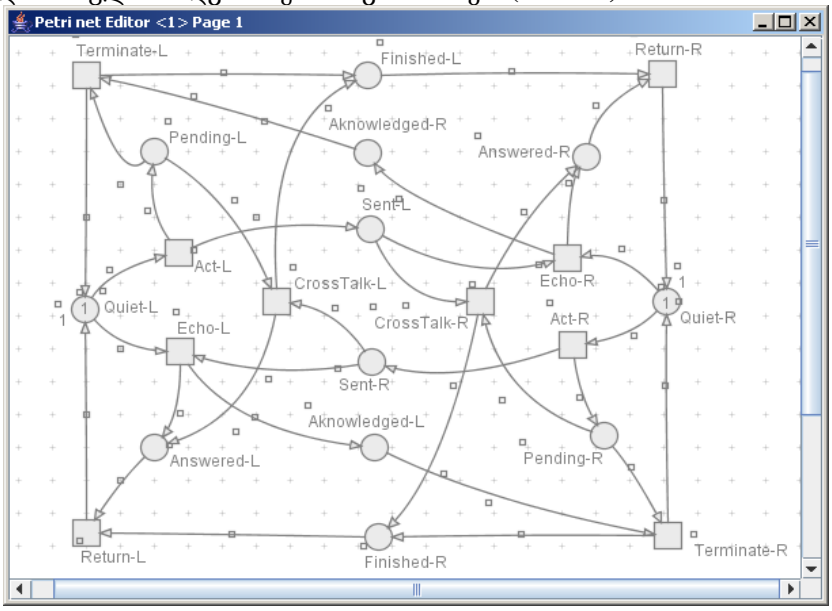
დაპროგრამების ტერმინებით ეს ნიშნავს, რომ ლოგიკური ცვლადი, რომელიც პროცესის მდგომარეობის მნიშვნელობას ინახავს, ლოკალურიდან (მხოლოდ თავისი პროცესისთვის გამოყენებადი) გლობალურ ცვლადად უნდა გარდაიქმნას, რომელთან მიმართვა (და საჭიროებისამებრ მნიშვნელობის შეცვლაც) რამდენიმე პროცესს შეეძლება.

5.14 ნახაზზე მოცემული პეტრის ქსელიც ,,ცნობების გაგზავნა-მიღების'' ამოცანის სრულყოფილი მოდელი არაა. ქსელში ჩიხი უკვე გამოირიცხა, მაგრამ სემანტიკურად იგი ჯერ კიდევ დასამუშავებელია. კერძოდ, რიგ სისტემებში (პირველ რიგში,

კომპიუტერულ ქსელებში) მონაცემთა მორიგი პაკეტის გაგზავნა აკრძალულია მანამ, სანამ წინა პაკეტის წარმატებით მიღების შესახებ დასტური არ მოვა, ანუ მორიგი სეანსის დაწყებამდე ყოველი წინა სეანსი ბოლომდე უნდა იქნას მიყვანილი.

5.14 ნახაზზე ეს წესი ირღვევა (სეანსის შეცდომა), მაგალითად, გადასასვლელთა შემდეგი მიმდევრობით გაშვებისას "Act-L –> Echo-R –> Return-R –> Act-R –> CrossTalk-L". ამ დროს პოზიციაში "Sent-R" მოხვედრილი მარკერი ეკუთვნის სისტემის ახალ (მეორე) პროცესს (R-დან L-ში ცნობის გაგზავნა) და იგი აღწევს L-პროცესს მანამ, სანამ ეს უკანასკნელი პირველ პროცესს დაასრულებდეს (ქვესისტემა L პასიურ მდგომარეობაში ჯერ კიდევ არ გადასულა ანუ R-იდან დასტური არ მიუღია).

პირველი პროცესი დაუსრულებელი დარჩება, რაც პროცეს R-ის მიერ დაბრუნებული დასტურის "დაკარგვას" ნიშნავს. პრობლემის მოსაგვარებლად ქსელს ორი ახალი პოზიცია (Finished-L და Finished-R) ემატება, რომლებიც "პროცესის დასასრულის" მდგომარეობას გამოსახავს (ნახ.5.15).



ნახ.5.15. პროცეს-ორიენტირებული CrossTalk

აქ მოცემულ პეტრის ქსელში ახალი პოზიციები უკვე გამორიცხავს პროცესის მიერ ახალი პროცესის დაწყებას ძველის დასრულებამდე, ამასთან მოცემული პეტრის ქსელი პროცესების პარალელიზმსაც ამოდელირებს: L- და R-პროცესებს ცნობების პარალელურად გაგზავნა-მიღება შეუძლია.

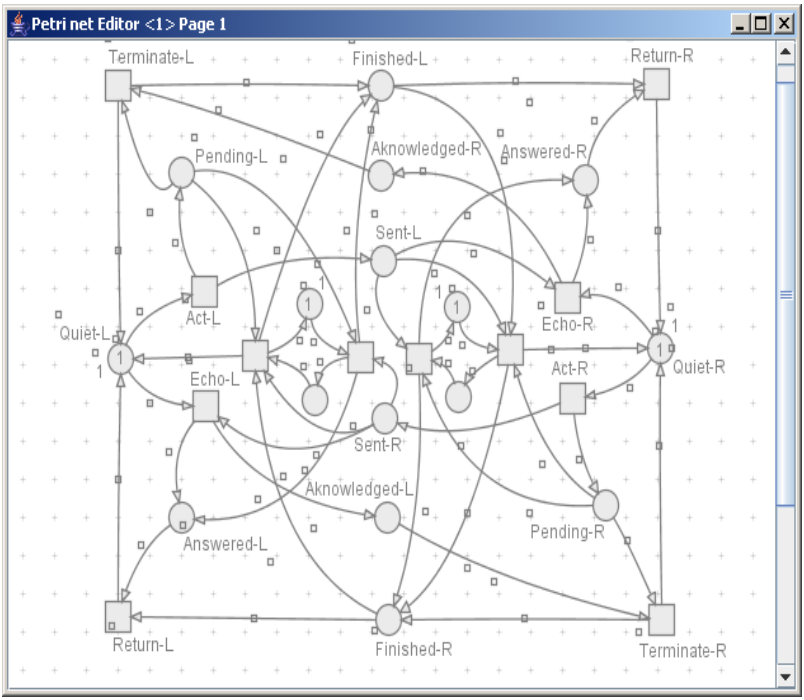
ამასთან, მოდელი ვერ წყვეტს **სინქრონიზაციის** ამოცანას, რომელიც დაისმება იმ შემთხვევაში, თუ პროცესები ცნობების გაგზავნა-მიღებისთვის არა 2 ურთიერთდამოუკიდებელ, არამედ ერთ, საერთო გადაცემის ფიზიკურ არხს იყენებს.

თუ ამ დროს ორივე პროცესი ინფორმაციას პარალელურად გაგზავნის, ისინი გზაში შეხვდება და ერთმანეთს დაამახინჯებს, ხოლო დამახინჯების ფაქტს 5.15 ნახაზზე მოცემული პეტრის ქსელი ვერ ასახავს, ამიტომ საჭიროა **პრიორიტეტების** შემოღება იმ მიზნით, რომ მოცემულ დროის მომენტში ინფორმაციის გადაცემა მხოლოდ ერთი პროცესისთვის იყოს შესაძლებელი.

მარტივ შემთხვევაში ერთი პროცესი მეორის მიმართ ცალსახად პრიორიტეტულია, ანუ ერთი პროცესის **CrossTalk**-გადასასვლელი გაიშვება მეორისაზე ადრე და პროცესის მიმდინარე პროცესიც მეორე სისტემის შესაბამის პროცესზე ადრე დამთავრდება.

უფრო რთული ვარიანტია პეტრის ქსელი ალტერნატიული პრიორიტეტებით, რომელიც 5.16 ნახაზზეა გამოსახული. ალტერნატიულ-პრიორიტეტებიანი პეტრის ქსელის მოდელი ერთდროულად **პარალელიზმისა** და **სინქრონიზაციის** ამოცანებს წყვეტს, მხოლოდ იმ პირობით, რომ ქსელის მუშაობას **L**-პროცესი იწყებს.

ორივე პროცესის მხრიდან ცნობების ერთდროული გაგზავნის შემთხვევაში L-პროცესი კავშირის არხის მონოპოლური მფლობელი ხდება და პირველი იყენებს ალტერნატიულ გადასასვლელს (CrossTalk-L) R-პროცესის საკონტროლო მარკერის მითვისების ხარჯზე (Sent-R პოზიციიდან), თავისი ბიჯის შესრულებისთანავე გადავა Answered-L პოზიციაში (“R-პროცესისგან ცნობა მიღებულია”) და “აიძულებს” R-პროცესს ცნობის ხელახლა გაგზავნას (Quiet-R პოზიციიდან, რომელიც ამ ოპერაციების შედეგად მარკერს ხელახლა ღებულობს).



ნახ.5.16. „ინფორმაციის გაგზავნა-მიღების” სისტემა ალტერნატიული პრიორიტეტებით

ამჯერად **L**-პროცესი ცნობას ჩიხური სიტუაციის გარეშე მიიღებს და პროცესს დაასრულებს, რის შემდეგაც მმართველ ფუნქციას უკვე **R**-პროცესი აიღებს და მორიგი პროცესი დაიწყება.

5.3. ურთიერთგამორიცხვის ალგორითმები

ურთიერთგამორიცხვის პრობლემა ყველაზე გამოკვეთილად ოპერაციული სისტემების, კომპიუტერული ქსელების და მონაცემთა ბაზების მართვის სისტემების დაპროექტებისას წარმოიშობა, როცა რამდენიმე პროცესი საერთო რესურსებს ინაწილებს. რესურსებთან ერთდროული მიმართვა ხშირად ჩიხურ სიტუაციებს წარმოშობს ან რესურსების არასასურველ განაწილებას განაპირობს [19].

ურთიერთგამორიცხვის ალგორითმებს **MUTEX**-ალგორითმები მათი ინგლისური სახელწოდების შემოკლებული ვარიანტის მიხედვით ეწოდება (**MUTual EXclusion** - ,,ურთიერთგამორიცხვა").

ალგორითმების არსის გასაგებად წარმოვიდგინოთ სისტემა, რომელიც შედგება ორი ქვესისტემისაგან (ან რომელშიც 2 პროცესი მუშაობს) - **L** და **R**. თითოეული პროცესის მოქმედება შეზღუდულია სამ მდგომარეობაში ციკლური გადასვლებით. ესენია პასიური, ლოდინის და კრიტიკულ უბანზე მუშაობის მდგომარეობები.

პასიური მდგომარეობიდან ლოდინის მდგომარეობაში ორივე პროცესი ურთიერთდამოუკიდებლად გადადის. **კრიტიკული უბნის** მდგომარეობაში ორივე პროცესის ერთდროულად ყოფნა არ შეიძლება (ურთიერთგამორიცხვის თვისება).

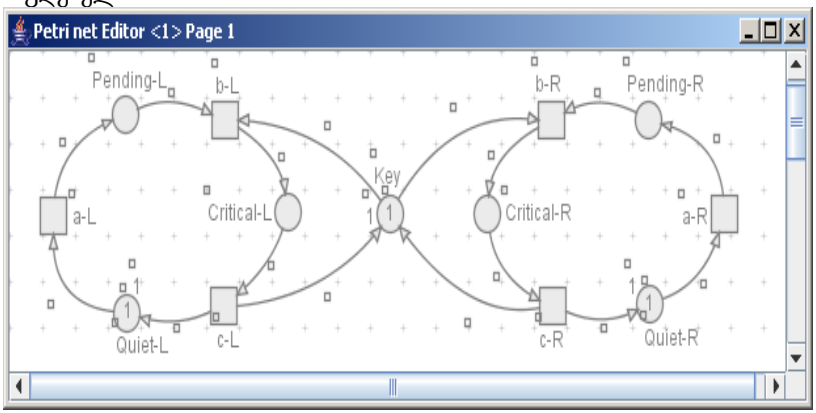
ვერცერთი პროცესი ლოდინის მდგომარეობიდან პასიურში ისე ვერ დაბრუნდება, თუ კრიტიკული უბანი არ გაიარა (ევოლუციურობის თვისება).

არსებობს MUTEX-ის 2 საწყისი ალგორითმი რესურსების ,,უკიდურესად უსამართლო'' და ,,უკიდურესად სამართლიანი'' განაწილებისთვის. მათ მხოლოდ თეორიული ღირებულება გააჩნია.

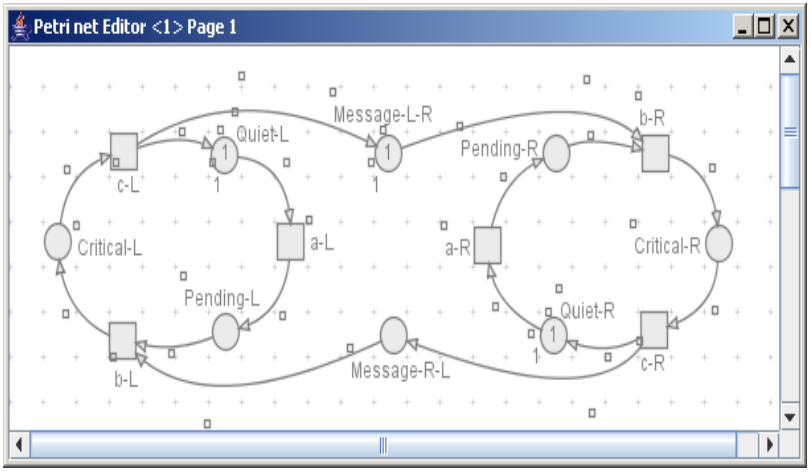
კონფლიქტური MUTEX-ალგორითმი (ნახ.5.17-ა) ერთი პროცესისგან საერთო რესურსის მუდმივი მითვისების შესაძლებლობას ასახავს. სურათზე key რესურსისათვის პროცესებს კონფლიქტი მოსდით და მისი გადაჭრის საშუალებაც არ ჩანს, ორივე პროცესს შეუძლია მიიტაცოს რესურსი და კრიტიკულ უბანზე რამდენჯერაც უნდა, იმდენჯერ იმუშაოს.

ალტერნატიულ MUTEX-ალგორითმში (ნახ.5.17-ბ) რესურსები ყოველთვის სამართლიანად ნაწილდება, ზედმეტად სამართლიანადაც. ერთი პროცესი კრიტიკული უბნიდან გამოსვლისას მეორეს აცნობებს, რომ კრიტიკულ უბანზე მუშაობა დაასრულა, რის შემდეგაც მეორე პროცესი ვალდებულია კრიტიკულ უბანზე იმუშაოს, სხვაგვარად პირველი პროცესი ხელახლა ვერ მოხვდება კრიტიკულ უბანზე და პირიქით.

რეალური სისტემების უმრავლესობისთვის ამგვარი მიდგომა (ისევე, როგორც კონფლიქტი საერთო რესურსისთვის) მიუღებელია.



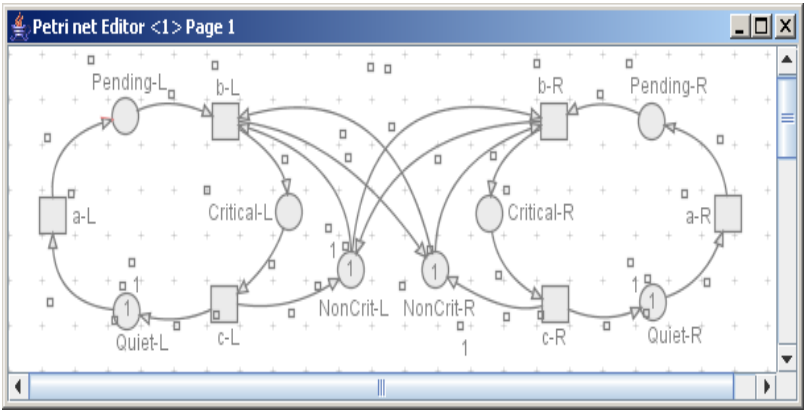
ნახ.5.17-ა. კონფლიქტური MUTEX-ალგორითმი



ნახ.5.17-ბ. ალტერნატიული MUTEX-ალგორითმი

პრობლემა გადაწყდება, თუ ავაგებთ **მდგომარეობის შემოწმების MUTEX**-ალგორითმს (ნახ.5.18). მასში პროცესებს სპეციალური ალმების **noncrit-L** და **noncrit-R** საშუალებით

ერთმანეთის მდგომარეობის შემოწმება შეუძლია და როცა კრიტიკული უბანი თავისუფალია, თითოეულ პროცესს მასში იმდენჯერ შეუძლია მოხვდეს, რამდენჯერაც უნდა.

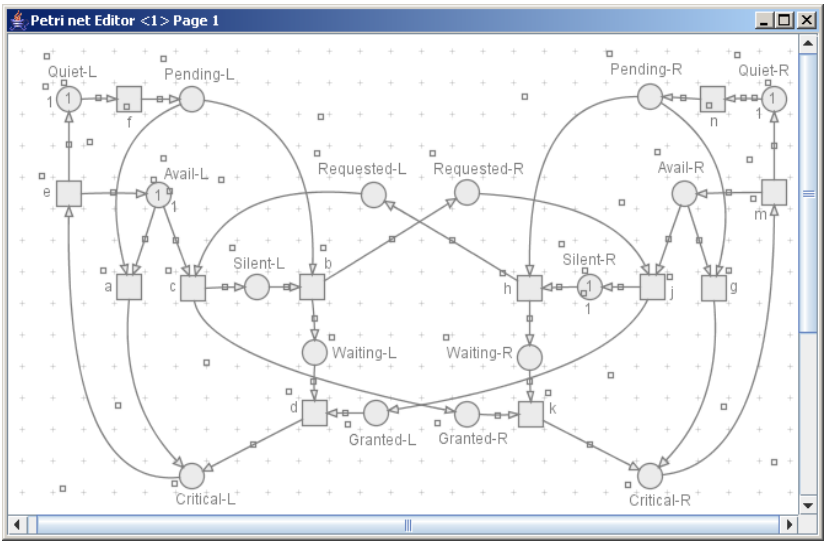


ნახ.5.18. მდგომარეობის შემოწმების MUTEX-ალგორითმი

5.3.1. მარკერული MUTEX-ალგორითმი

სამივე ზემოაღნიშნული ალგორითმი არასრულყოფილია. გაცილებით მისაღები იქნება, პროცესებს ერთმანეთისაგან კრიტიკულ უბანზე მუშაობის უფლების მოთხოვნა და ნებართვის გადაცემა რომ შეეძლოს. ამგვარ სისტემას **მარკერული MUTEX-**ალგორითმი აღწერს (ნახ.5.19).

მარკერი კრიტიკულ უბანზე მუშაობის უფლებას წარმოადგენს, რომელსაც დროის მოცემულ მომენტში მხოლოდ ერთი პროცესი ფლობს (მარკერი პოზიციაში Avail-L ან Avail-R). პროცესს კრიტიკულ ზონაში მოხვედრა მხოლოდ მარკერის ფლობის შემთხვევაში შეუძლია, ხოლო თუ მარკერი არ გააჩნია, შეუძლია მეორე პროცესისგან მისი მოთხოვნა (პოზიცია Requested-L ან Requested-R), რომელიც მარკერის მფლობელმა პროცესმა შეიძლება დააკმაკოფილოს და თვითონ უმარკეროდ დარჩეს (პოზიცია Granted-L ან Granted-R) ან არ დააკმაყოფილოს და ისევ თვითონ გავიდეს კრიტიკულ უბანზე.



ნახ.5.19. მარკერული MUTEX-ალგორითმი

ნახაზზე მარკერს L-პროცესი ფლობს (პოზიცია Avail-L) და მოქმედების ორი ვარიანტი აქვს: a გადასასვლელით კრიტიკულ ზონაში მოხვდება (პოზიცია Critical-L) ან c-თი პასიურ მდგომარეობაში გადავა (Silent-L), რათა R-პროცესმა მოთხოვნის შემთხვევაში მარკერის მიღება შეძლოს. კრიტიკულ ზონაში მომუშავე L-პროცესი e გადასასვლელით საწყის მდგომარეობას დაუბრუნდება და ამასთან მარკერს Avail-L პოზიციაში აბრუნებს.

ამით ერთი ციკლი დასრულებულია, ხოლო მორიგ ციკლს ის პროცესი იწყებს, რომელსაც მარკერი არა აქვს (მარკერს მოითხოვს). წინა ციკლის შესრულების შედეგის მიხედვით მომთხოვნი შეიძლება იყოს ისევ R-პროცესი (Pending-R), თუ წინა ციკლში მარკერი არ გადაცემულა ან L-პროცესი (Pending-L) – თუ გადაიცა.

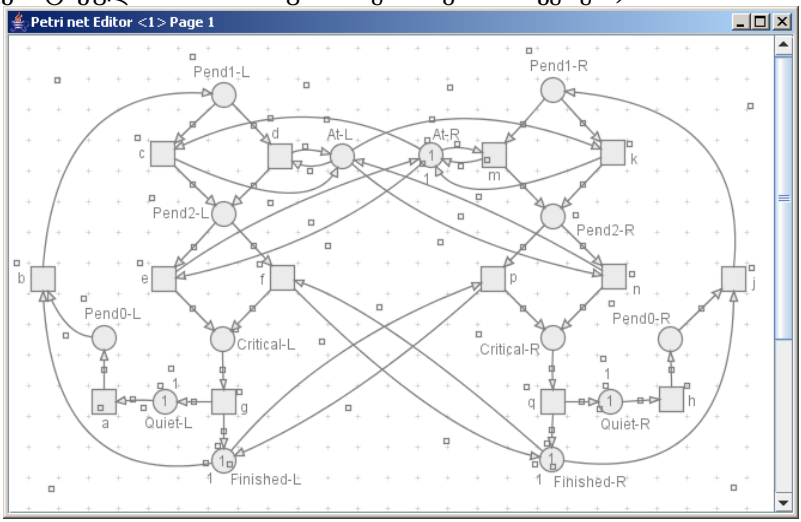
5.3.2. პიტერსონის MUTEX-ალგორითმი

ურთიერთგამორიცხვის ამოცანის კიდევ უფრო მოქნილი ვარიანტი პიტერსონის MUTEX-ალგორითმია, სადაც რესურსის მისაღებად მზადმყოფი პროცესის ლოდინი სამ მდგომარეობადაა დანაწევრებული. მაგალითად, L-პროცესისთვის Pend0-L, Pend1-L

და Pend2-L (ნახ.5.20). Pend0-L პოზიციას თეორიული ღირებულება არ გააჩნია, პასიური მდგომარეობიდან (Quiet-L) მასში გადასვლა ალგორითმის შესრულების საერთო სტრუქტურას არ არღვევს.

მაგრამ ამ მდგომარეობის არსებობა მოცემულ პეტრის ქსელში მაინც საჭიროა, რადგან მასში გადასვლით პროცესი გამოთქვამს სურვილს (და არა პრეტენზიას) კრიტიკულ ზონაში მუშაობის ნებართვაზე (ანუ მეორე პროცესს ჯერჯერობით ხელს არ უშლის), რაც რეალური სისტემების დაპროგრამებისას შეიძლება გახდეს საჭირო.

ალგორითმში 4 ალამი, იგივე საკვანძო პოზიციაა: Finished-L და Finished-R (პროცესის დასასრული), At-L და At-R (კრიტიკულ ზონაში სამუშაო ნებართვის მარკერები).



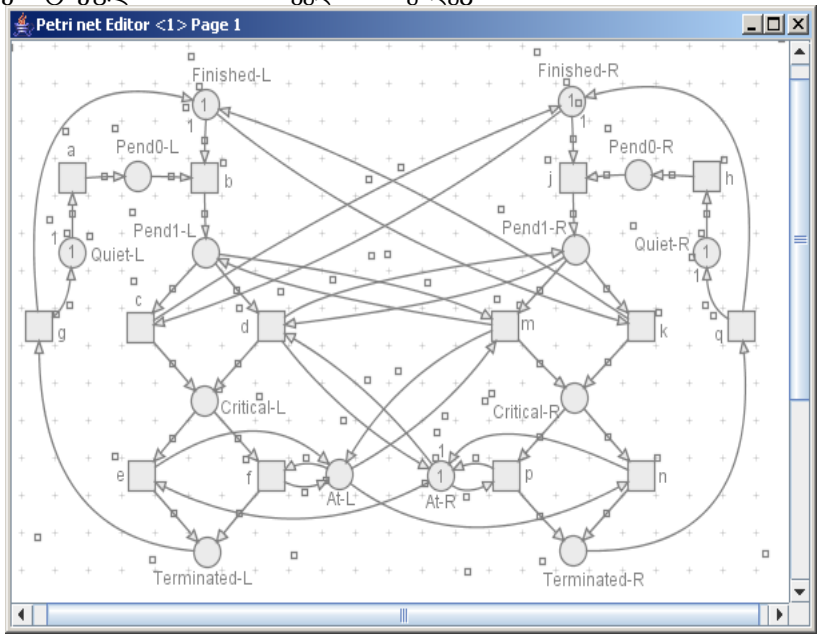
ნახ.5.20. პიტერსონის MUTEX-ალგორითმი

ამ ოთხ პოზიციაში მარკერების არსებობა განსაზღვრავს პროცესების **კრიტიკულ ზონაში** მოხვედრის მიმდევრობას. კერძოდ, თუ კრიტიკულ ზონაში მუშაობის სურვილს მხოლოდ ერთი, **L**-პროცესი გამოთქვამს (ანუ პოზიციაში **Finished-R** მარკერი შენარჩუნებულია), მაშინ **L**-პროცესი კრიტიკულ ზონას დაუბრკოლებლად აღწევს (გადასასვლელთა მიმდევრობა **a-b-c-f**),

მაგრამ საკმარისია **R**-პროცესმა თვითონაც მოისურვოს კრიტიკულ ზონაში მუშაობა (გადასასვლელები **h-j**), რომ პროცესებს მოქმედებათა **სინქრონიზაცია** მოუწევს, ოღონდ სამართლიანად: პროცესები ერთმანეთს ვერ „გადაუსწრებს", რომელი პროცესიც მარკერს პირველი მოითხოვს, კრიტიკულ ზონაშიც პირველი მოხვდება იმ გარანტიით, რომ მისი კრიტიკული ზონიდან გამოსვლისთანავე მეორე პროცესიც მიიღებს მასში მოხვედრის მარკერს (უფლებას).

5.3.3. დეკერის MUTEX-ალგორითმი

დეკერის MUTEX-ალგორითმი პიტერსონისას გარეგნულად საკმაოდ ჰგავს (ნახ.5.21). ძირითადი შინაარსობრივი განსხვავება შემდეგშია: პიტერსონის ალგორითმში **At-L** პოზიცია მარკერს ღებულობს მანამ, სანამ **L**-პროცესი კრიტიკულ ზონაში მოხვდებოდეს, ხოლო დეკერისაში - პირიქით, **L**-პროცესის კრიტიკულ ზონაში მოხვედრის შემდეგ.



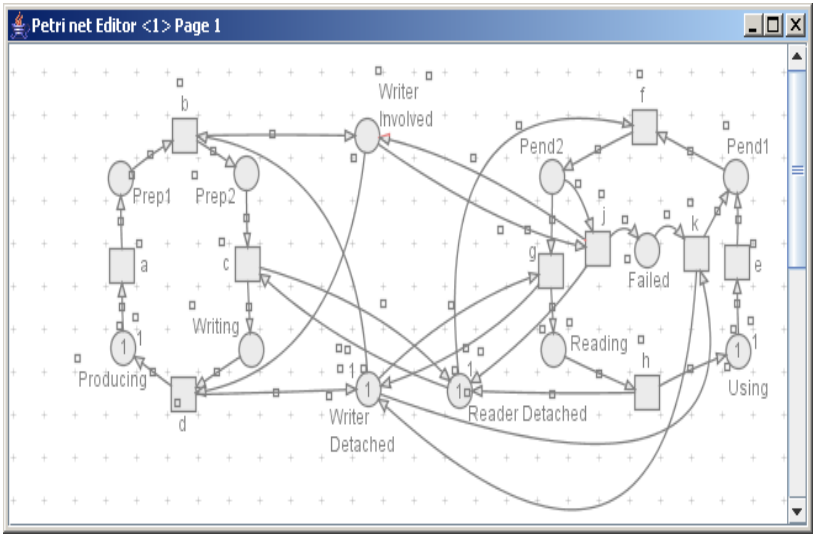
ნახ.5.21. დეკერის MUTEX-ალგორითმი

5.3.4. ოვიცკი-ლამპორტის MUTEX-ალგორითმი

აქამდე განხილული ყველა ალგორითმი ორი თანასწორი პროცესის ურთიერთქმედებას აღწერდა.

რეალური სისტემების უმრავლესობაში პროცესებს ენიჭება პრიორიტეტები და საერთო რესურსებთან მიმართვისას მაღალპრიორიტეტიანი პროცესი უფლებამოსილია დაბალ-პრიორიტეტიანი „გამოაძევოს“.

ოვიცკი-ლამპორტის MUTEX ალგორითმში ამგვარი სისტემის მოდელია წარმოდგენილი. მასში **განაწილებული ცვლადის წაკითხვა-განახლების** ოპერაციათა ურთიერთქმედება აისახება (ნახ.5.22).



ნახ.5.22. ოვიცკი-ლამპორტის MUTEX-ალგორითმი

სისტემა ორი პროცესისგან შედგება: ჩამწერისა (Writer) და წამკითხველისგან (Reader), რომლებიც საერთო მონაცემთა ცვლადის მნიშვნელობას ცვლის (ჩამწერი) ან ამოიკითხავს (წამკითხველი). ჩამწერი პროცესი პრიორიტეტულია.

ალგორითმი სამ ალამს იყენებს: Writer Detached (ჩამწერი მოიხსნა), Writer Involved (ჩამწერი ჩაერია) და Reader Detached (წამკითხველი მოიხსნა).

პირველი და მესამე ალამი შესაბამისად ჩამწერ და წამკითხველ პროცესებს მოხსნის კრიტიკული უბნიდან და მეორე პროცესს სამოქმედო გზას უხსნის, მეორე ალამი კი ჩამწერი პროცესის წამკითხველის მოქმედებაში ჩარევას ამოდელირებს.

იგი აღკვეთს წამკითხველისგან ცვლადის წაკითხვის ოპერაციას, თუ იმავდროულად ჩამწერი ცვლადის განახლებას დააპირებს.

პროცესების მიმდევრობა ასეთია: ჩამწერს მომზადებული აქვს ცვლადის ახალი მნიშვნელობა (პოზიცია Producing), რომელიც a-b-c გადასასვლელთა მიმდევრობის გახსნით უნდა გადაიტანოს კრიტიკულ ზონაში (პოზიცია Writing) და განაწილებული ცვლადის მნიშვნელობა განაახლოს.

დავუშვათ, წამკითხველმაც უკვე წაიკითხა და გამოიყენა განაწილებული ცვლადის წინა მნიშვნელობა (პოზიცია Using) და მოინდომა თავის კრიტიკულ ზონაში (პოზიცია Reading) e-f-g გადასასვლელების გხსნის გზით გადასვლა განაწილებული ცვლადის ახალი მნიშვნელობის წასაკითხად.

წარმოიშობა ჩიხური სიტუაცია: g გადასასვლელის გახსნა აკრძალულია, რადგან პოზიციაში Writer Detached (ჩამწერი მოიხსნა) მარკერი არ არის და ვერც c გადასასვლელი გაიხსნება, რადგან პოზიციიდან Reader Detached (წამკითხველი მოიხსნა) მარკერი წამკითხველმა აიღო. ჩიხიდან გამოსავალს პოზიცია Writer Involved (ჩამწერი ჩაერია) წარმოადგენს, რომელშიც არსებული მარკერის მონაწილეობით იხსნება j გადასასვლელი და წამკითხველი მდგომარეობაში Failed (წაკითხვა ჩაიშალა) გადადის, ხოლო ჩამწერი დაუბრკოლებლად აღწევს კრიტიკულ ზონას, განაახლებს ცვლადს და d გადასასვლელით თავის პირვანდელ მდგომარეობას უბრუნდება, რის შემდეგაც პოზიციას Writer Detached (ჩამწერი მოიხსნა) მარკერი უბრუნდება და წამკითხველს შეუძლია განაწილებული ცვლადის წასაკითხად ახალი პროცესი წამოიწყოს (გადასასვლელთა მიმდევრობა k-f-g).

5.4. მოდელირება და ანალიზი WinPepsy ინსტრუმენტით

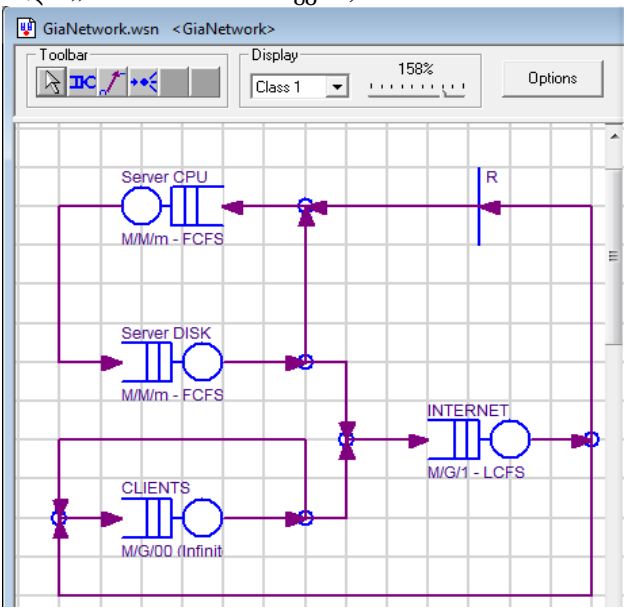
5.4.1. „კლიენტ-სერვერ“ ჩაკეტილი ქსელის მოდელირება და ანალიზი

ჩაკეტილი ქსელის შესახებ ინფორმაცია განხილული იყო მე-4 თავში. ახლა WinPepsy ინსტრუმენტის გარემოში ავაგებთ კონკრეტული ქსელის სქემას და გამოვიკვლევთ მის მახასიათებლებს [57].

ავაგოთ მარტივი ჩაკეტილი ქსელი კლიენტ-სერვერ არქიტექტურის მაგალითისთვის, რომელთა შორის კავშირი ინტერნეტით ხორციელდება.

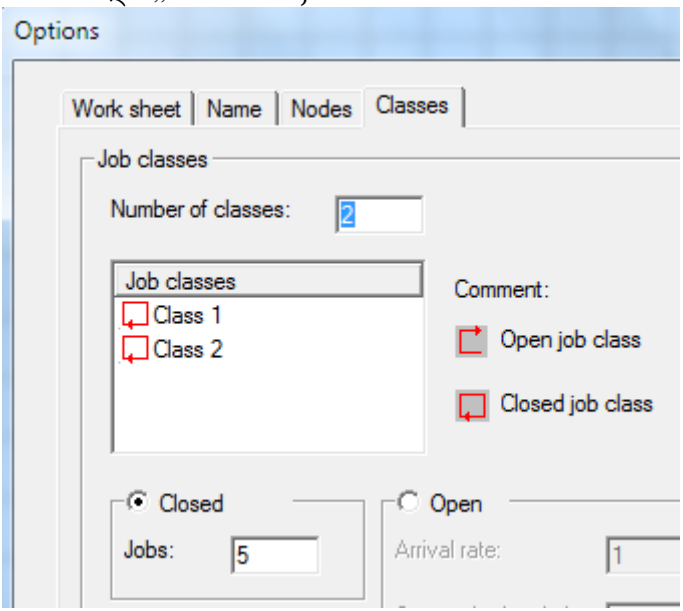
1. ავირჩიოთ მენიუში „File“ და პუნქტი „New“. დიალოგში სისტემის შეკითხვაზე ვირჩევთ გრაფიკულ გენერაციას და ღილაკს „ Generate new network“.

2. მივიღეთ WinPepsy რედაქტორის ცარიელი ფანჯარა, რომელშიც ინსტრუმენტების გამოყენებით ავაგოთ 5.23 ნახაზზე ნაჩვენები სქემა (კვანძის შეტრიალება შეიძლება მაუსის მარჯვენა ღილაკით და „ Orientation“ არჩევით).



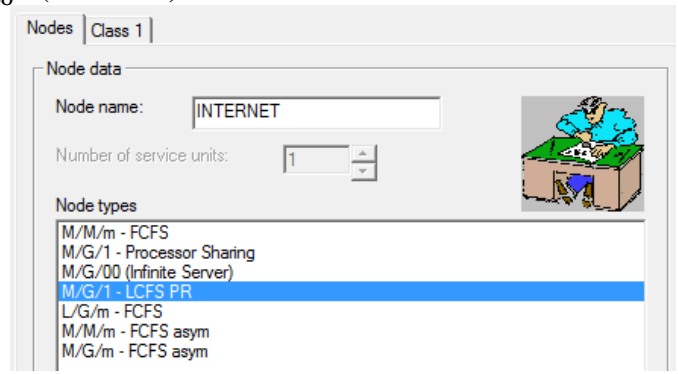
ნახ.5.23.

3. ღილაკით „Options“ გამოიტანება დიალოგური ფანჯარა (ნახ.5.24), სადაც „Classes“ გვერდზე შევცვალოთ „number of classes“ 2-ით და „number of jobs“ 5-ით.



ნახ.5.24-ა

4. კვანძზე დაკლიკვით შევალთ ფანჯარაში, სადაც „Nodes“ გვერდზე შეიძლება ქსელის კვანძებზე სახელების და ტიპების შერჩევა (ნახ.5.24-ბ).



ნახ.5.24-ბ

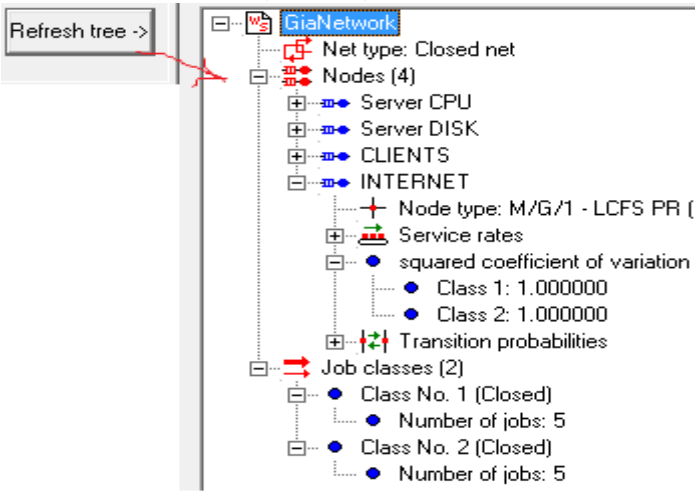
5. კვანძისთვის Server CPU შევცვალოთ „service units“ 3-ით.

6. სერვისის ტარიფები (service rates) კვანძებისთვის ასე გავანაწილოთ:

Node name	Class 1	Class 2
Server CPU	6	6
Server disk	14	14
Internet	16	18
Clients	10	20

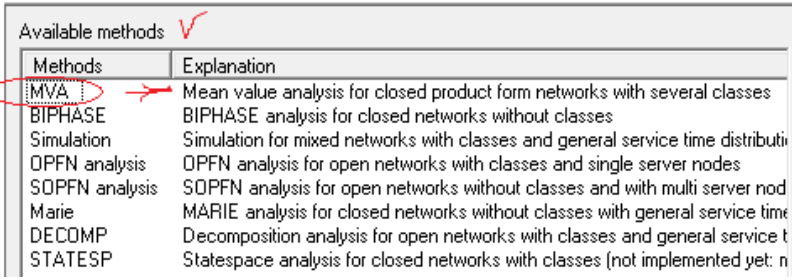
7. კვანძთაშორის გადასასვლელებზე შეიძლება შეიცვალოს ალბათობათა მნიშვნელობები (ისრებზე მაუსის დაკლიკვის შემდეგ). ჩვენ შემთხვევაში Server CPU-დან Server DISK-კენ ისარზე ავიღოთ 1, ხოლო ყველა დანარჩენზე 0.5;

8. პანელზე „Display“-ს კომბობოქსში ავირჩიოთ Class 2. ხელახლა ავაწყოთ კავშირები კვანძებს შორის და შევცვალოთ პარამეტრები Options–ში. job=5 და ჩავსვათ ალბათობებიც კავშირებისთვის პირველის მსგავსად. ბოლოს ავამოქმედოთ ღილაკი „ Refresh tree ->“. (ნახ.5.25).



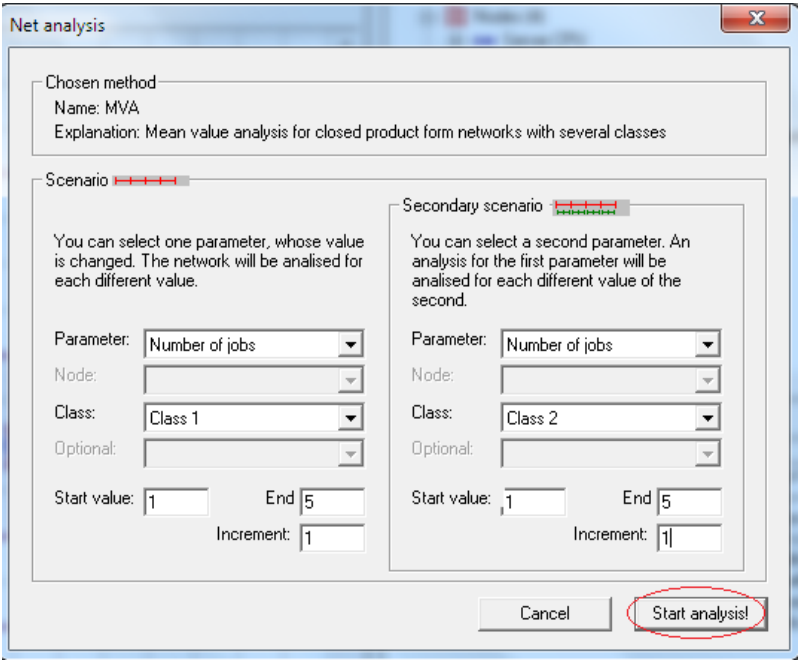
ნახ.5.25

9. ამის შემდეგ შეიძლება ქსელის ანალიზის ჩატარება, ფანჯრის მარჯვენა ქვედა ნაწილში MVA–მეთოდის არჩევით (ნახ.5.26-ა).

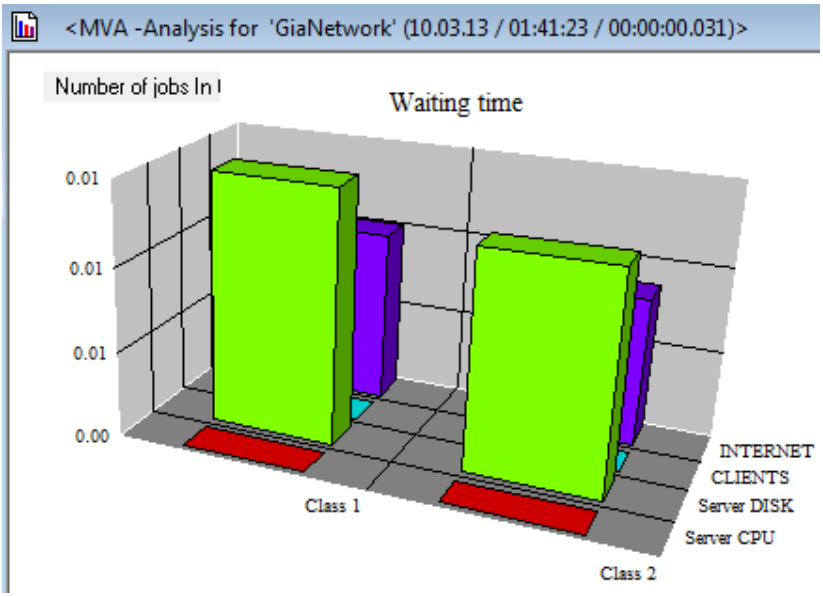


ნახ.5.26-ა

10. დიალოგურ ფანჯარაში ავირჩევთ პარამეტრებს (ნახ.5.26-ბ) და Start-ღილაკით მივიღებთ ანალიზის შედეგს (ნახ.5.26-გ).

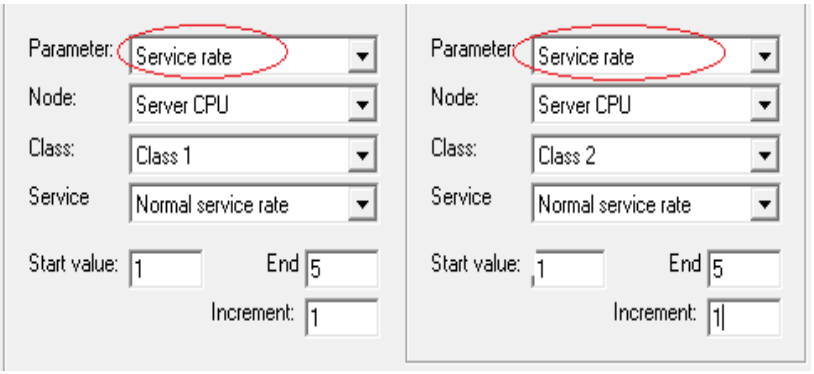


ნახ.5.26-ბ

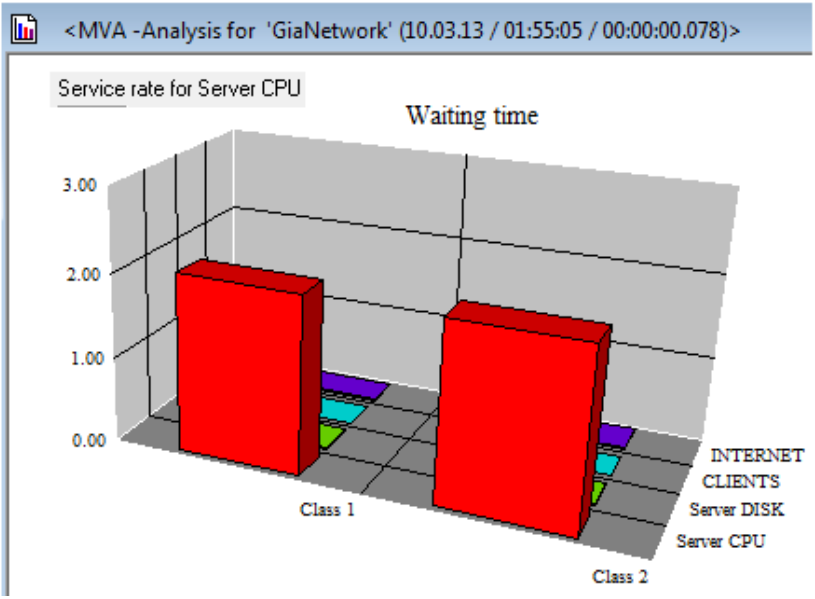


ნახ.5.26-გ

11. აქ შესაძლებელია ექსპერიმენტების გაგრძელება სხვადასხვა მახასიათებლების ანალიზის მისაღებად. მაგალითად, 5.27-ა ნახაზზე ნაჩვენებია მომსახურების ინტენსიურობის პარამეტრის არჩევა, შემდეგ კი მისი ანალიზი და შედეგები (ნახ.5.27-ბ).

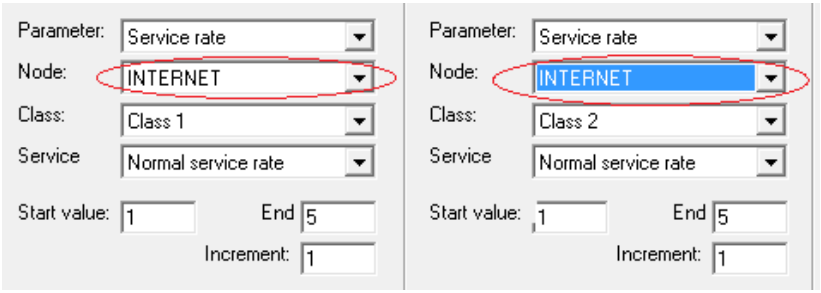


ნახ.5.27-ა

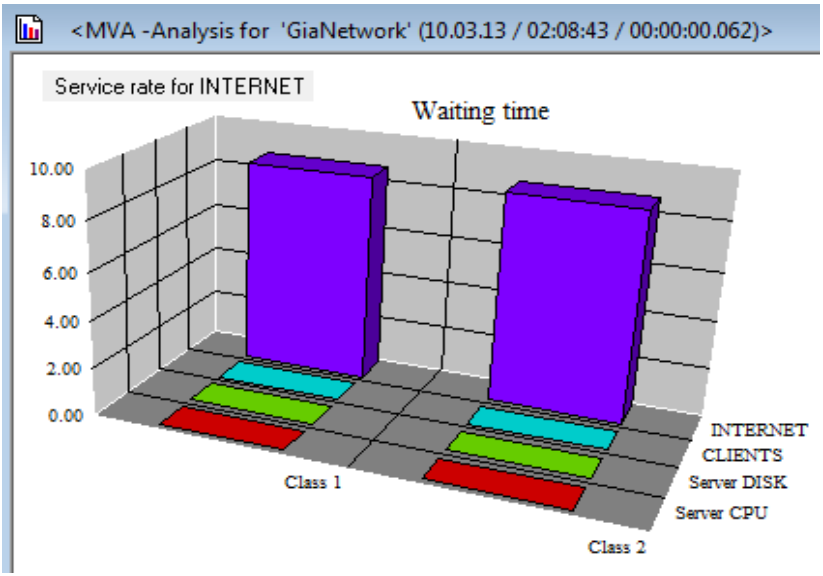


ნახ.5.27-ბ

12. 5.28-ა ნახაზზე ნაჩვენებია INTERNET–კვანძის მომსახურების ინტენსიურობის პარამეტრის არჩევა და ანალიზის შედეგები (ნახ.5.28-ბ).

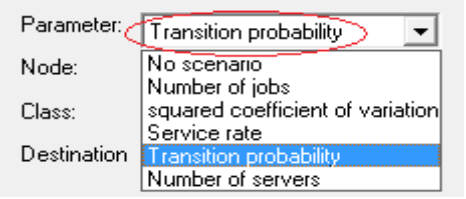


ნახ.5.28-ა

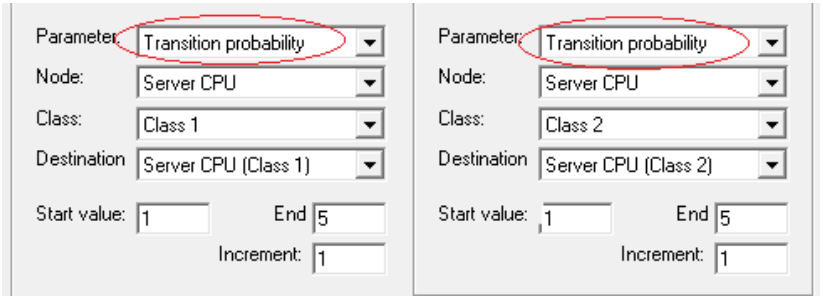


ნახ.5.28-ბ

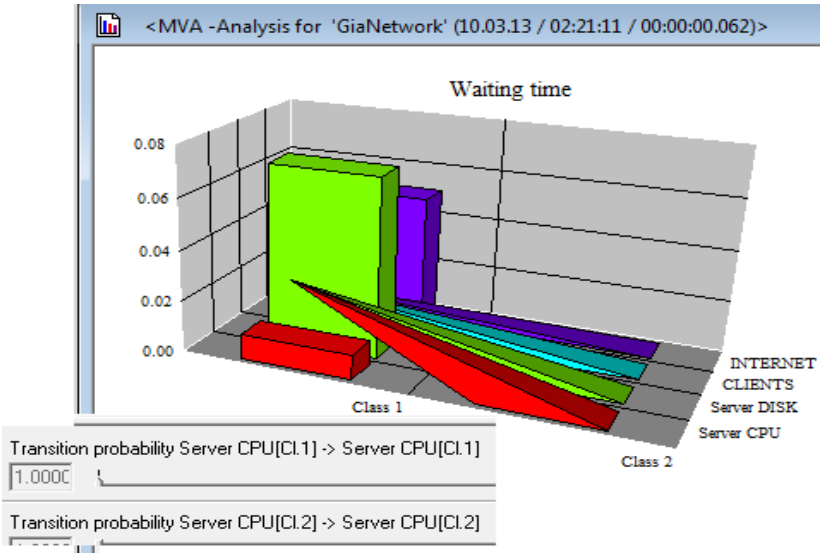
13. პარამეტრების კომბობოქსიდან (ნახ.5.29) ავირჩიოთ გადასასვლელთა ალბათობების ანალიზი (Transition probability) ServerCPU-თვის (ნახ.5.30-ა), შედეგებით 5.30-ბ ნახაზზე.



ნახ.5.29-ა



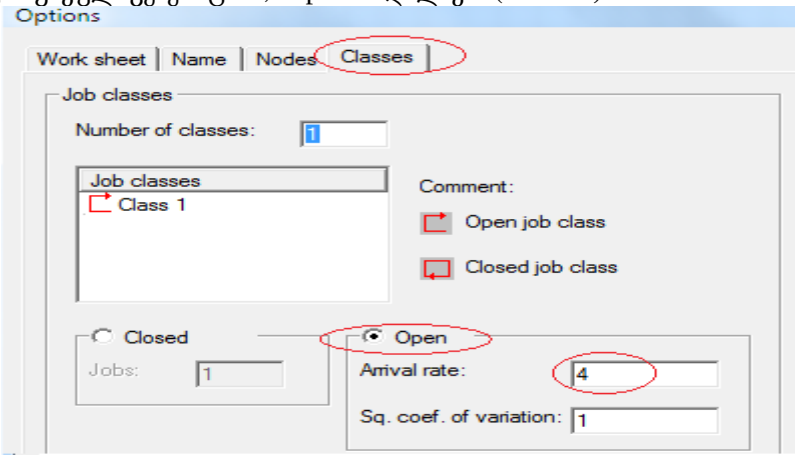
ნახ.5.30-ა



ნახ.5.30-ბ

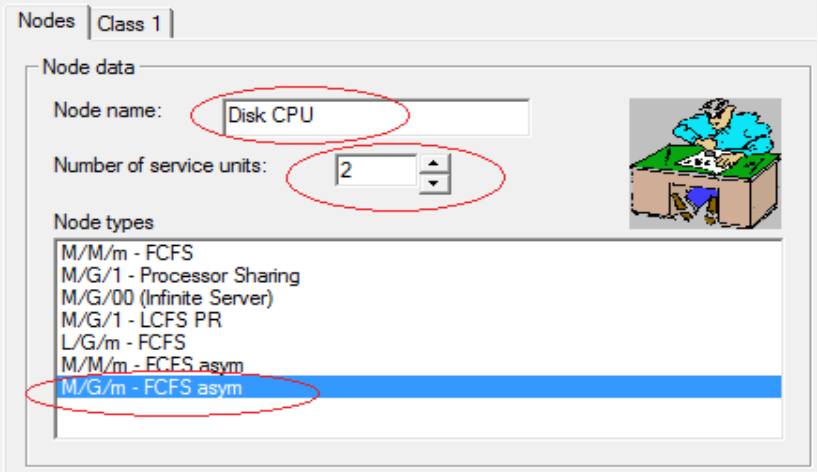
5.4.2. „კლიენტ-სერვერ“ ღია ქსელის მოდელირება და ანალიზი

1. WinPepsy –ის ამუშავების შემდეგ File->New და ვირჩევთ გრაფიკულ გენერაციას, Options ღილაკს (ნახ.5.31).



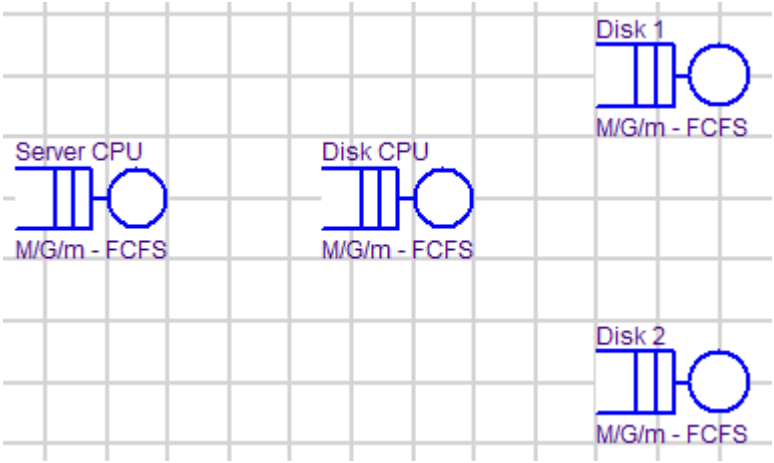
ნახ.5.31. Open ქსელის და Arival rate=4 არჩევა

2. WinPepsy-ის ცარიელ რედაქტორში ვიწყებთ კვანძების მოდელირებას პარამეტრებით (ნახ.5.32).



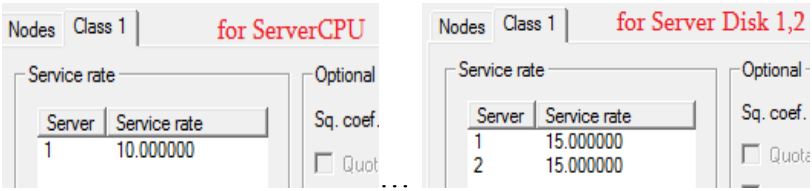
ნახ.5.32-ა. Disk CPU კვანძის პარამეტრების შერჩევა

მიიღება 5.32-ბ ნახაზზე ნაჩვენები სურათი, სადაც ჩანს კვანძების სახელები და მათი ტიპები.



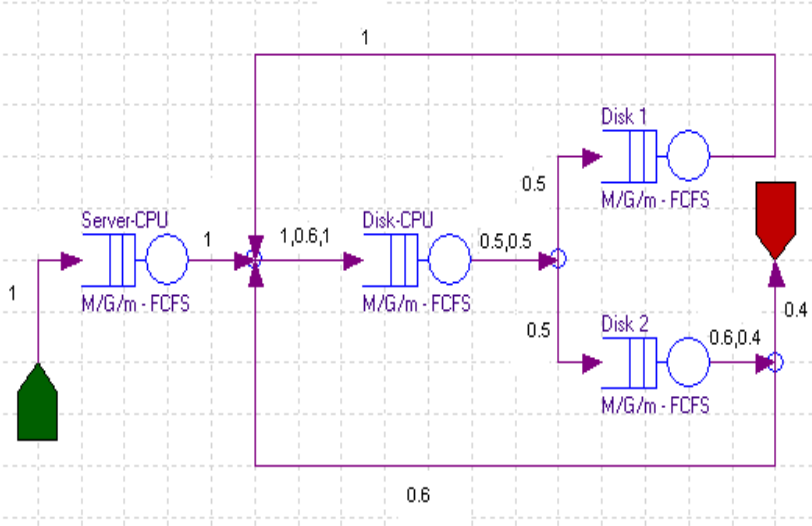
ნახ.5.32-ბ. ქსელის საწყისი სქემა ოთხი კვანძით და ტიპებით

3. ახლა თითოეულ კვანძს უნდა მივანიჭოთ მომსახურების ტარიფი (Service rates). ServerCPU=10; ServerDisk1 და2=15 და15 (ნახ.5.32-გ);.



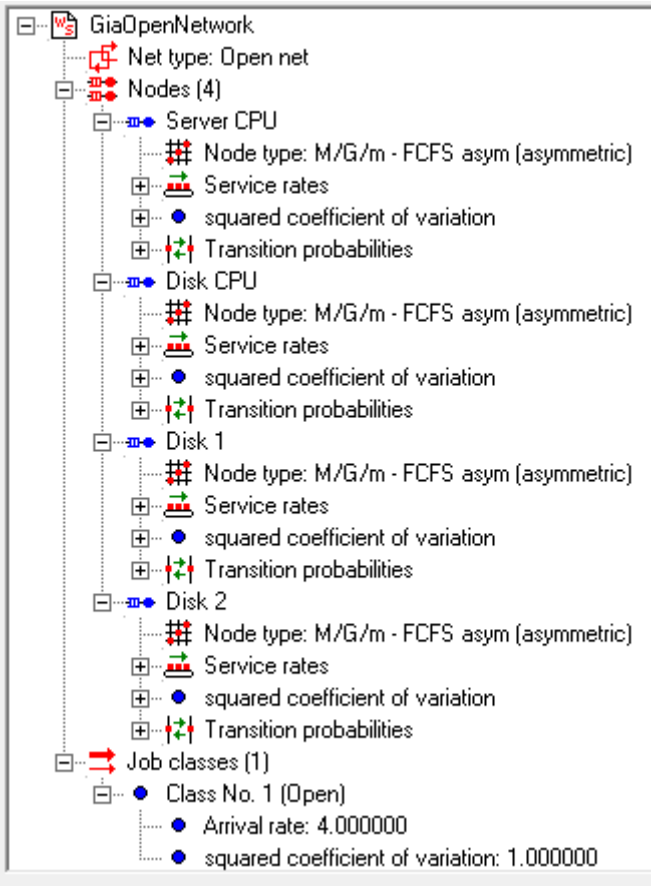
ნახ.5.32-გ. ქსელის კვანძებზე Service rates მნიშვნელობების მინიჭება

4. ქსელის სქემას უნდა დაემატოს კვანძთაშორისი გადასასვლელები შესაბამისი ალბათობებით. აგრეთვე ორი ელემენტი: ქსელში შესვლის წერტილი (ხუთკუთხედი მარცხენა ქვედა ნაწილში) და ქსელიდან გამოსვლის წერტილი (მარჯვნივ ხუთკუთხედი). ეს სიმბოლოები აუცილებელია ღია ქსელებისთვის. შედეგი ასახულია 5.32-დ ნახაზზე.



ნახ.5.32-დ. ქსელის სქემა გადასასვლელებით და ალბათობებით

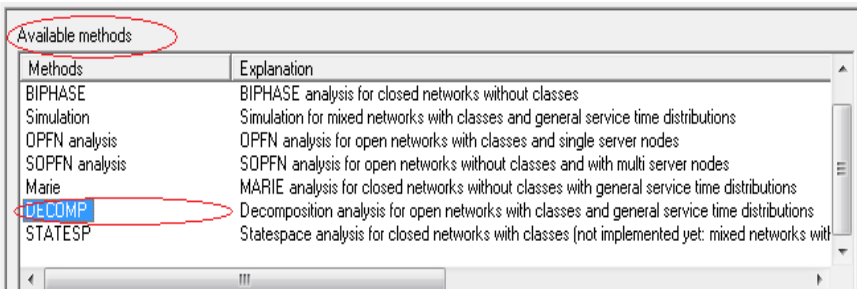
5. ავამოქმედოთ „Refresh tree ->“ ღილაკი. მარჯვენა მხარეს მოდიფიცირებულ ხეს ექნება 5.33 ნახაზზე ნაჩვენები სახე.



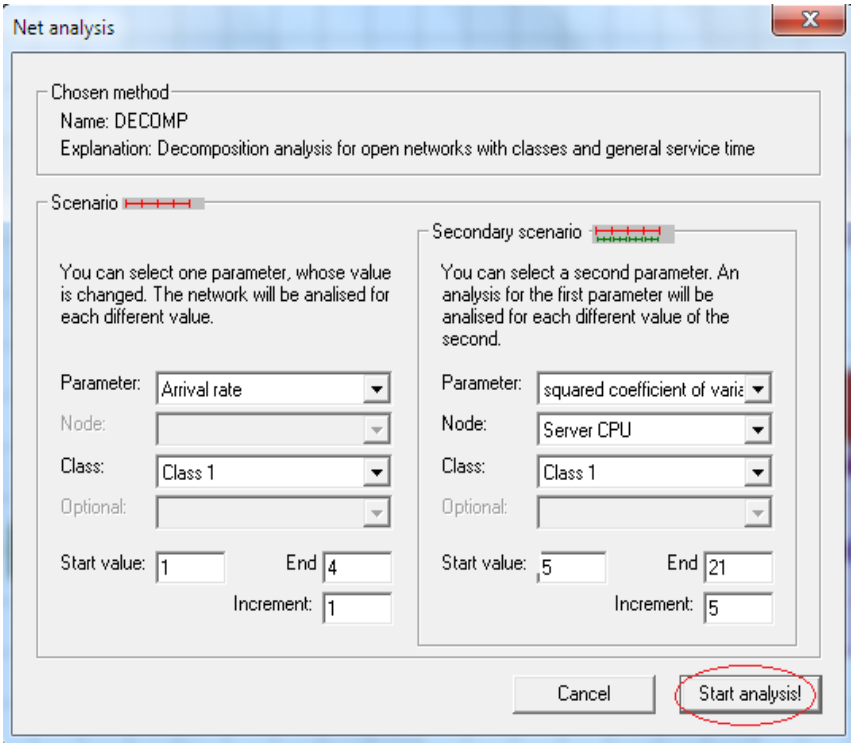
ნახ.5.33

ამგვარად, მოდელი მზადაა ანალიზის ჩასატარებლად. ეკრანის ქვედა მარჯვენა კუთხეში მოცემულია მეთოდები, რომლებიც გამოიყენება ამისათვის (ნახ.5.34).

ჩვენ ავირჩიოთ DECOMP მეთოდი. გამოჩნდება დიალოგური ფანჯარა (ნახ.5.35-ა), რომელშიც შევარჩევთ ჩვენთვის სასურველ გასაანალიზებელ პარამეტრებს.

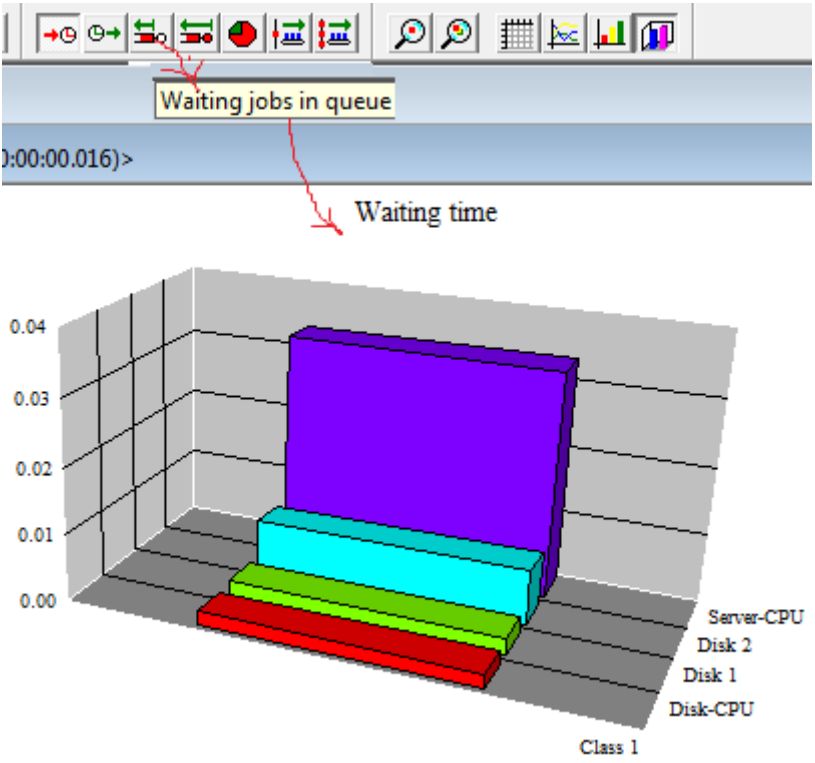


ნახ.5.34. DECOMP მეთოდის ამუშავება



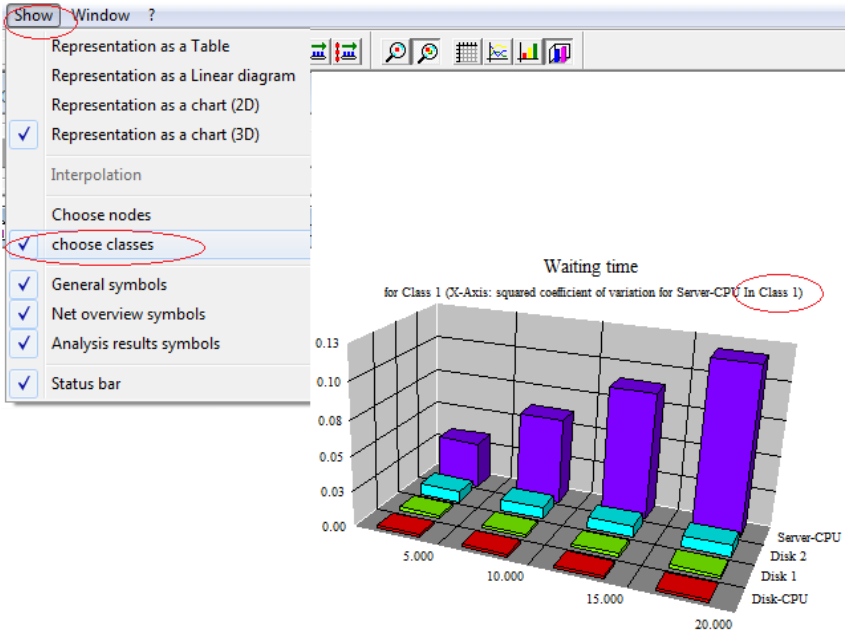
ნახ.5.35-ა. ქსელის ანალიზის ფანჯარა, შერჩეული პარამეტრებით

Start analysis ღილაკის ამოქმედების შედეგი მოცემულია 5.35-ბ ნახაზზე.

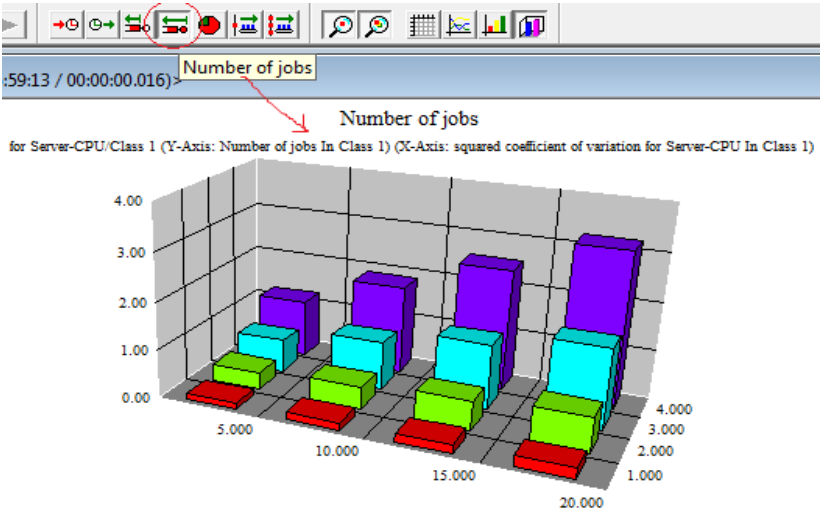


ნახ.5.35-ბ. მოთხოვნათა ლოდინის დრო რიგში

სხვა მახასიათებლების გასაანალიზებლად საჭიროა გამოვიყენოთ  პანელის ღილაკები და მენიუში Show-პუნქტი (ნახ.5.35-გ).



ნახ.5.35-გ. ლოდინის დრო

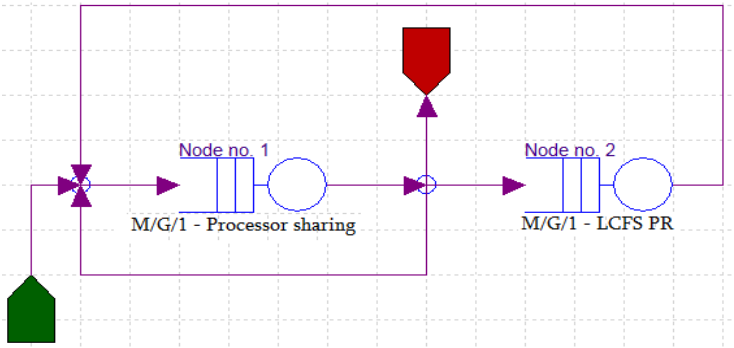


ნახ.5.35-დ. მოთხოვნათა რაოდენობა რიგში

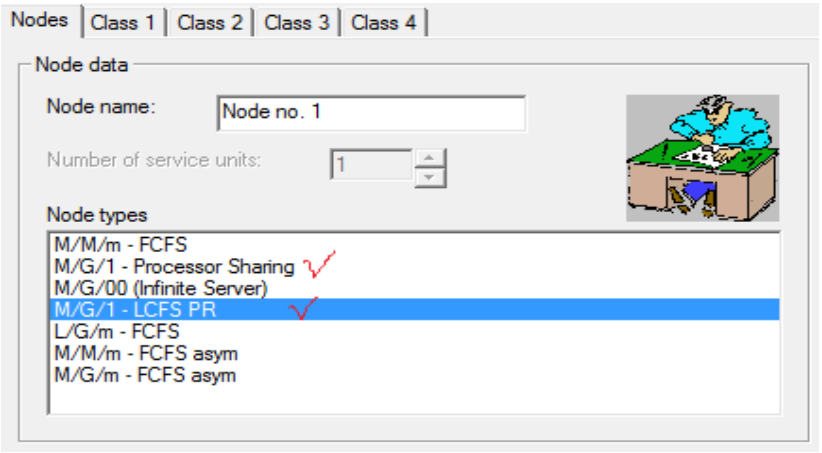
5.3.3. ჰიბრიდული ქსელის მოდელირება და ანალიზი

ჰიბრიდული ანუ შერეული ტიპის ქსელი ისეთი ქსელია, რომელსაც აუცილებლად აქვს მინიმუმ ერთი ღია კლასი და ერთიც ჩაკეტილი კლასი (ანუ ორივე სახეზეა).

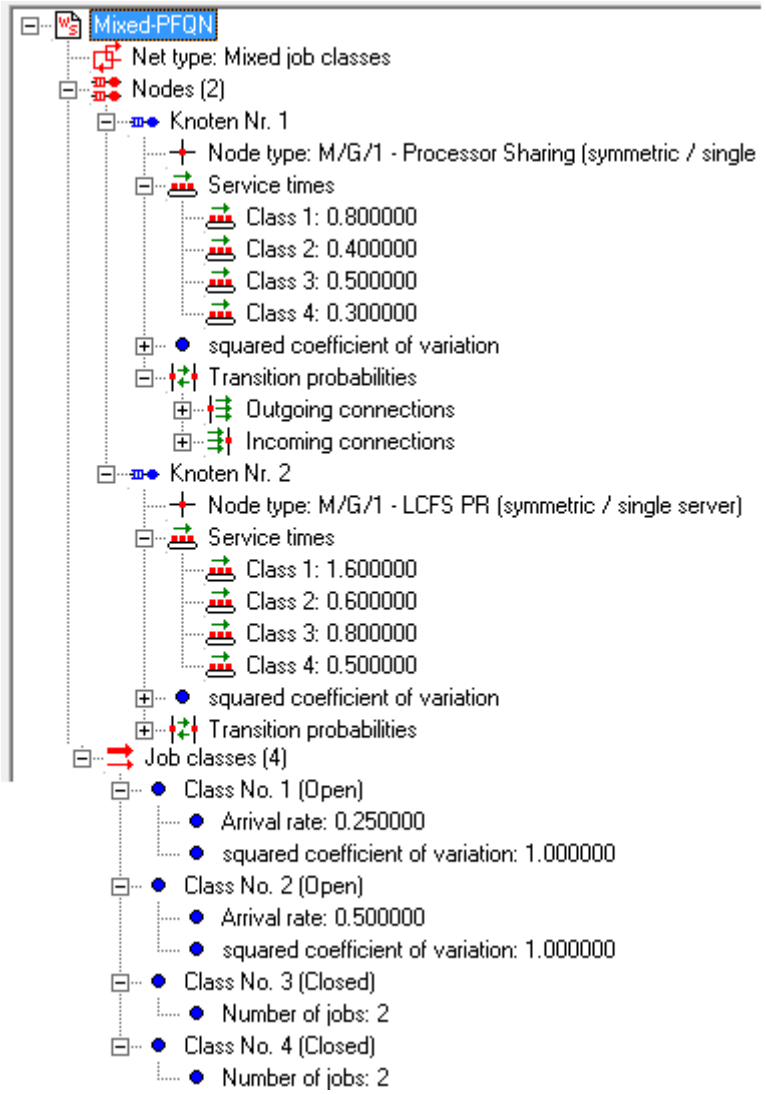
ქსელის აგების თვალსაზრისით WinPepsy რედაქტორში ღია კლასი აიგება ღია ქსელის კლასის წესებით, ხოლო ჩაკეტილი კლასი – ჩაკეტილი ქსელის კლასის წესებით. განვიხილოთ კონკრეტული მაგალითი, რომელიც 5.36 ნახაზზეა ასახული.



ნახ.5.36. ჰიბრიდული ქსელის მოდელი



ნახ.5.37. კვანძის ტიპების შერჩევა



ნახ.5.38. კვანძის პარამეტრების მნიშვნელობები

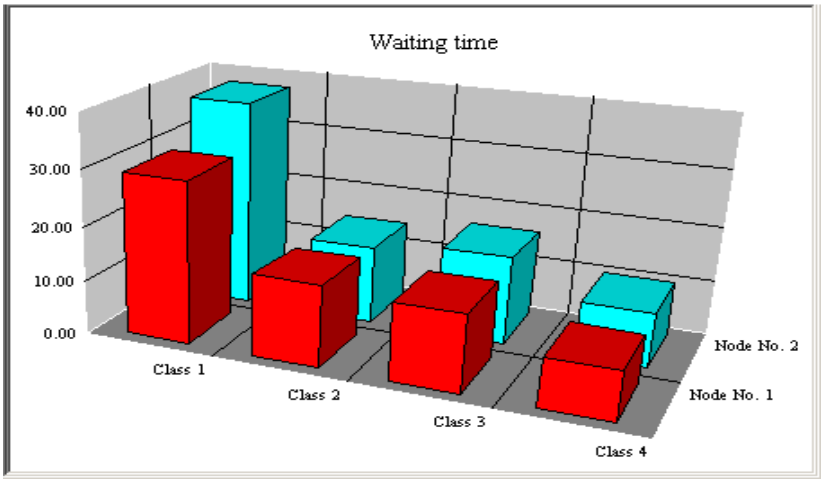
აგებული მოდელის ანალიზის ჩასატარებლად შევირჩიოთ მეთოდი Simulation (ნახ.5.39).

Available methods

Methods	Explanation
MVA	Mean value analysis for closed product form networks with several classes
BIPHASE	BIPHASE analysis for closed networks without classes
Simulation	Simulation for mixed networks with classes and general service time distributions
OPFN analysis	OPFN analysis for open networks with classes and single server nodes
SOPFN analysis	SOPFN analysis for open networks without classes and with multi server nodes
Marie	MARIE analysis for closed networks without classes with general service time dist
DECOMP	Decomposition analysis for open networks with classes and general service time
STATESP	Statespace analysis for closed networks with classes (not implemented yet: mixed

ნახ.5.39. ჰიბრიდული ქსელის ანალიზის მეთოდი

ანალიზის ერთი შედეგი 2-კვანძისა და 4-კლასისთვის დაყოვნების დროის მიხედვით მოცემულია 5.40 ნახაზზე.



ნახ.5.40. ქსელის ანალიზის შედეგების ფრაგმენტი

ამით დავასრულებთ პრაქტიკული მაგალითების განხილვას მოდელირებისა და ანალიზის ინსტრუმენტების საშუალებით, კერძოდ პეტრის ქსელებისა და რიგების თეორიის გამოყენებით.

აღნიშნული საინჟინრო ინფორმატიკის ტექნოლოგიები დღესაც აქტუალურია და გარკვეულ ინტერესს წარმოადგენს თანამედროვე მართვის ავტომატიზებული სისტემების დაპროექტების, მოდელირების და პროგრამული რეალიზაციის თვალსაზრისით.

ლიტერატურა:

1. სტუ-ს „მართვის ავტომატიზებული სისტემების“ კათედრა. www.gtu.ge/katedrebi/kat94 (http://test.gtu.ge/katedrebi/kat94).

2. http://de.wikipedia.org/wiki/Informatik

3. Humboldt University Berlin: www.informatik.hu-berlin

4. http://en.wikipedia.org/wiki/Software_engineering

5. Booch G., Jacobson I., Rumbaugh J. Unified Modeling Language for Object-Oriented Development. Rational Software Corporation, Santa Clara, 1996.

6. ბოტჰე კ., სურგულაძე გ., დოლიძე თ., შონია ო., გრ. სურგულაძე. თანამედროვე პროგრამული პლატფორმები და ენები (WindowsNT, Unix, Linux, C++, Java, XML). დამხმ.სახ., ISBN 99940-14-11-0. სტუ, თბილისი. 2002.

7. სურგულაძე გ., ვედეკინდი ჰ., თოფურია ნ. განაწილებული ოფის-სისტემების მონაცემთა ბაზების დაპროექტება და რეალიზაცია UML-ტექნოლოგიით. მონოგრ.,ISBN 99940-57-17-0. სტუ. თბილისი. 2006.

8. Страуструп Б. Язык программирования Си++. Пер. с англ., Москва, Радио и связь. 1991.

9. Meyer-Wegener K., Surguladze G. Programmierung mit C/C++/C#. 99940-40-69-3. Erlangen-Tbilissi. GTU. Tbilissi. 2004.

10. Bothe K., Surguladze G. Objektorientierte Modellierung und Programmierung mit der UML. BerliniTbilissi. GTU. Tbilissi. 2003.

11. რეისიგი ვ., სურგულაძე გ., გულუა დ. ვიზუალური ობიექტ-ორიენტირებული დაპროგრამების მეთოდები (BorlandC++Builder,PetriNet). დამხმ.სახ., ISBN99928-943-9-3. სტუ, თბ., 2002.

12. ჩოგოვაძე გ., გოგიჩაიშვილი გ., სურგულაძე გ., შეროზია თ., შონია ო. მართვის ავტომატიზებული სისტემების დაპროექტება და აგება (თეორიული და პრაქტიკული ინფორმატიკა). სახელმძღვანელო. ISBN 99928-882-7-X. სტუ, თბილისი. 2001.

13. სურგულაძე გ. დაპროგრამების საფუძვლები (C&C++ ენების ბაზაზე). დამხმ.სახ., ISBN 99940-56-16-6. სტუ, თბილისი. 2006.

14. სურგულაძე გ. ობიექტ-ორიენტირებული დაპროგრამების მეთოდი (C++&Java ენების ბაზაზე)“. დამხმ.სახ., ISBN 99940-56-18-2. სტუ, თბილისი. 2007.

15. სურგულაძე გ., დოლიძე თ., ყვავაძე ლ. კომპონენტურ-ვიზუალური დაპროგრამება: ინტერფეისების აგება C# da C++ ენებზე მონაცემთა განაწილებული ბაზებისათვის. დამხმ.სახ., ISBN 99940-48-99-6. სტუ, თბილისი. 2006.

16. სურგულაძე გ., თურქია ე. ბიზნეს–პროცესების მართვის ავტომატიზებული სისტემების დაპროექტება. მონოგრ., ISBN 99940-14-81-1, სტუ. თბილისი. 2003.

17. Codd E. F. A Relational Model for Large Shared Data Banks, Comm. ACM, Vol.13, No. 6, June 1970. Relational Model for Database Management - Version 2, Addison-Wesley 1990.

18. სურგულაძე გ. პეტრიაშვილი ლ. მონაცემთა საცავის აგების ტექნოლოგია ინტერნეტული ბიზნესის სისტემებისათვის. მონოგრ., ISBN 99940-40-36-7. სტუ. თბილისი. 2005.

19. სურგულაძე გ., გულუა დ. განაწილებული სისტემების ობიექტ-ორიენტირებული მოდელირება უნიფიცირებული პეტრის ქსელებით. მონოგრ.,ISBN99940-48-07-4. სტუ. თბ., 2006.

20. ახობაძე მ., ბოსიკაშვილი ზ., გოგიჩაიშვილი გ., სურგულაძე გ., სუხიაშვილი თ., ღვინეფაძე გ. სასამართლო საქმეთა წარმოების ქსელური მართვის ავტომატიზებული სისტემა. მონოგრ., ISBN 99940-48-63-5. სტუ. თბილისი. 2006.

21. სურგულაძე გ., შონია ო., ყვავაძე ლ. მონაცემთა განაწილებული ბაზების მართვის სისტემები (MsSQL Server, Access, InterBase, JDBC, Oracle). 99940-35-18-5. სტუ. თბილისი. 2004.

22. გოგიჩაიშვილი გ., სურგულაძე გ., შონია ო. დაპროგრამების მეთოდები: C & C++. სახელმძღვანელო. სტუ, თბილისი. 1997.

23. ჩოგოვაძე გ., სურგულაძე გ., შონია ო. მონაცემთა და ცოდნის ბაზების აგების საფუძვლები. სახელმძღვანელო. სტუ, თბილისი. 1996.

24. Прангишвили А., Прокопьев С. Информационные технологии согласования управленческих решений по выбору целей и стратегий в конфликтологии. Georgian Electronic Scientific Journal. 2005, №3(7). http://gesj.internet-academy.org.ge.

25. Surguladze G., Petriashvili L., Okhanashvili M., Kvavadze L. Construction of Multi-dimensional Analysis Packet of Commercial Objects with Decision Cube Components. Georgian Engineering News, No 4. 2005.

26. Surguladze G., Petriashvili L., Shonia o., Surguladze Gr. The Visual, Objectoriented Modelling, Design, Analysis and Implementation using .NET technology and Petri Nets. Bullet.of Georg.Acad.of Science. N172-2, 2005.

27. Гогичаишвили Г.Г., Сургуладзе Г.Г. Разработка прикладного программного обеспечения интегрированных информационных систем управления на основе UML. Georgian Electronic Scientific Journal. 2002, №1. http://gesj.internet-academy.org.ge.

28. რეისიგი ვ., სურგულაძე გ., გულუა დ. დაპროგრამების სწავლებისა და სერტიფიცირების პროცესის მოდელირება სისტემური პეტრის ქსელებით. სტუ შრ.კრ. №4(437), 2001.

29. ბოტჰე კ., სურგულაძე გ., კაშიბაძე მ. მემკვიდრეობითობა მართვის ინფორმაციული სისტემების დაპროგრამებაში: მონაცემთა ბაზებიდან UML-ტექნოლოგიამდე. სტუ შრ.კრ. №4(437), თბ., 2001.

30. Гогичаишвили Г.Г., Мануков С.Г. Формирование диагноза в экспертной системе диагностики стоматологических заболеваний. Сб.тр.ГТУ, АСУ, №1(2), 2007.

31. Codd E.F, Codd S.B., Salley C.T. Providing OLAP to User-Analysts: An IT Mandate, Codd & Associates, Ann Arbor/Michigan, 1993.

32. ჩოგოვაძე გ. მართვის ავტომატიზებული სისტემების აგების საფუძვლები. თსუ, თბილისი, 1981.

33. http://en.wikipedia.org/wiki/Cloud_computing

34. King R. Cloud Computing: Small Companies Take Flight. http://www.businessweek.com/stories/2008-08-04/cloud-computing-small-companies-take-flightbusinessweek-business-news-stock-market-and-financial-advice. 2008.

35. Hewitt C. ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing // IEEE Internet Computing, 2008, vol.12, no. 5, p. 96-99.

36. Bolch G., Greiner S., De Meer H., Trivedi K. Queueing Networks and Markov Chains, Modeling and Performance Evaluation with Computer Science Application. John Wiley & Sons, 1998. 726 S.

37. Bolch G. Leistungsbewertung von Rechensystemen mittels analytischer Warteschlangenmodelle. Teubner. 1989. 311 S.

38. Бек К. Шаблоны реализации корпоративных приложений. Экстремальное программирование: Пер. с англ. М.: Вильямс, 2008

39. სურგულაძე გ., გულიტაშვილი მ., კაკულია ი., ჩერქეზიშვილი გ., ჯავახიშვილი ი. პროგრამული სისტემების სასიცოცხლო ციკლის პროცესის მოდელირება უნივერსალური და ექსტრემალური პროგრამირების პრინციპების კომპრომისული გადაწყვეტით. სტუ-ს შრ.კრ. მას-№1(8), 2008. გვ.63-70

40. Скопин И.Н. Основы менеджмента программных проектов. www.intuit.ru/ department/se/msd. 2004.

41. დაპროგრამების მეთოდები: საკურსო პროექტების აგება (UML, MsVisio, C++). ISBN 978-9941-14-125-5. სტუ, თბ., 2007

42. თურქია ე. ბიზნეს–პროექტების მართვის ტექნოლოგიური პროცესების ავტომატიზაცია. სტუ. თბ., 2010

42. Bazan P., Bolch G., German R. WinPepsi Guide. Erlangen-Nurnberg Univ., 2003.

43. Microsoft Visio 2010. http://www.osalt.com/visio.

44. სურგულაძე გ., კაშიბაძე მ. ორგანიზაციულ სისტემებში ინფორმაციული რესურსების მართვა. ISBN978-9941-14-447-6. სტუ. თბ., 2009.

45. Питерсон Дж. Теория Сетей Петри и моделли-рование систем. Пер.с англ., Москва, «Мир», 1983.

46. სურგულაძე გ., გულუა დ., თურქია ე. ბიზნეს-პროცესების მოდელირება პეტრის ქსელებით. ISBN978-9941-14-125-6. სტუ. თბ., 2008.

47. Reisig, Wolfgang: Elements of Distributed Algorithms : Modeling and Analysis with Petri Nets. Berlin ; Heidelberg ; New York et al : Springer, 1998.

48. Starke, Peter: Analyse von Petri-Netz-Modellen. Stuttgart : B. G. Teubner, 1990 (Leitfäden und Monographien der Informatik).

49. Jensen K., Kristensen M.L., Wells L. Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems. University of Aarhus. Denmark. 2007.

50. სურგულაძე გ., ბულია ი., ოხანაშვილი მ., ქრისტესიაშვილი ხ. კორპორაციული მენეჯმენტის ბიზნეს-პროცესების მოდელირება და კვლევა ფერადი პეტრის ქსელებით. სტუ-ს შრ.კრ. „მას“-№1(12), თბილისი, 2012, გვ.73–82.

51. სურგულაძე გ., თურქია ე., ოხანაშვილი მ., სურგულაძე გ. მარკეტინგული პროცესების მართვის ერთი მოდელის შესახებ ფერადი პეტრის ქსელებით- №2(5), 2008. გვ. 9–16.

52. CPN Tools. www.daimi.au.dk/CPNTools/. გადწმ. 1.02.13

53. ბოლხი გ., სურგულაძე გ., პეტრიაშვილი ლ., ჩიხრაძე ბ. მულტიპროცესორული სისტემების რესურსების მართვის პროგრამული უზრუნველყოფის დამუშავება Borland_C++Builder ინსტრუმენტით. სტუ-ს შრ., 4(437), თბილისი, 2001.

54. Дейтель Г. Введение в операционные системы. Пер. с англ., Мир, М., 1987.

55. სურგულაძე გ., კაშიბაძე მ. ოპერაციული სისტემები: პროცესების მართვის კვლევა პეტრის ქსელების თეორიის გამოყენებით. სტუ. თბ., 1993.

56. Kendall's notation. http://en.wikipedia.org/wiki/Kendall %27 s_notation. გადწმ. 5.01.13.

57. Bazan P., Bolch G., German R. WinPEPSY Guide. Erlangen-Nuernberg Univ., 2004.

58. Kuki A., Sztrik J., Bolch G.. Software Tools for Network Modelling. 6 th Intern. Conf. on Applied Informatics Eger, Hungary, January 27–31, 2004. pp.289-296.

იბეჭდება ავტორთა მიერ წარმოდგენილი სახით

გადაეცა წარმოებას 30.03.2013 წ. ხელმოწერილია დასაბეჭდად 15.04.2013 წ. ოფსეტური ქაღალდის ზომა 60X84 1/16. პირობითი ნაბეჭდი თაბახი 6. ტირაჟი 100 ეგზ.

საგამომცემლო სახლი „ტექნიკური უნივერსიტეტი“
თბილისი, მ. კოსტავას 77