# კორპორაციული აპლიკაციების აგება დაპროგრამების სერვის-ორიენტირებული ტექნოლოგიებით NoSQL ბაზაზე

გია სურგულაძე, ნინო კივილაძე, გიორგი კივილაძე

საქართველოს ტექნიკური უნივერსიტეტი

## რეზიუმე

განხილულია კორპორაციული მართვის ობიექტების პროგრამული უზრუნველყოფის დამუშავების თანამედროვე ფრეიმვორკები კომპანია მაიკროსოფთის პროგრამული პლატფორმების გამოყენებით: ASP.NET Web Forms, ASP.NET MVC და Silverlight. ყურადღება გამახვილებულია დაპროგრამების ტექნოლოგიების მახასიათებლებზე, შესაბამისი პრაქტიკული ექსპერიმენტების საფუძველზე გამოვლენილია მათი დადებითი და უარყოფითი მხარეები. შემოთავაზებულია რეკომენდაციები, რეალურ ამოცანებზე მუშაობისას, რა შემთხვევებში ავჯობებს დაპროგრამების ამა თუ იმ ტექნოლოგიის არჩევა. წარმოდგენილია ვებ-აპლიკაციების შემუშავების პრაქტიკული ამოცანა ადამიანური რესურსების მართვის ელექტრონული სისტემის მაგალითზე და მისი რეალიზაცია Microsoft Visual Studio, MsSQL Server პროგრამული ინსტრუმენტებით. განიხილება აგრეთვე მონაცემთა არარელაციური ბაზების გამოყენების შეასაძლებლობის კონცეფცია ამ სისტემებში.

**საკვანძო სიტყვები:** კორპორაციული მენეჯმენტი. პროგრამული აპლიკაცია. მართვის საინფორმაციო სისტემა. სერვის-ორიენტირებული დაპროგრამების ტექნოლოგია. პროგრამული ინჟინერია. არარელაციური ბაზები.

## 1. შესავალი

სახელმწიფო და კერძო სტრუქტურების ორგანიზაციული მართვის (მენეჯმენტის) ბიზნეს-პროცესების ავტომატიზაცია თანამედროვე კომპიუტერული ტექნიკისა და ინფორმაციული ტექნოლოგიების ინტეგრაციის საფუძველზე მეტად აქტუალურია [1,2]. ბოლო ათწლეულში განსაკუთრებით მოიმატა პროგრამული აპლიკაციების შექმნის და გამოყენების ტენდენციამ ინტერნეტული სისტემების ბაზაზე [3]. კორპორაციული ვებ-აპლიკაციები ედება აგრეთვე საფუძვლად „ელექტრონული მთავრობის“ და „ელექტრონული ბიზნესის“ სისტემებს. დესკტოპ-აპლიკაციებისგან განსხვავებით ვებ-აპლიკაციების გამოიყენება მნიშვნელოვანია შემდეგი ფაქტორების გამო:

- მომხმარებელი არ საჭიროებს სპეციალური პროგრამების ინსტალირებას. გაცილებით იოლია მისი დაინტერესება პროგრამული უზრუნველყოფით თუ მას ოპერატიულად ექნება შესაძლებლობა ჩაერთოს სამუშაო პროცესში. ვებ-აპლიკაცია არ საჭიროებს კლიენტის მანქანაზე ჩასატარებელ წინასწარ მოსამზადებელ სამუშაოებს, ინსტალაციას და კონფიგურირებას. მომხმარებელი პროგრამულ უზრუნველყოფასთან სამუშაოდ იყენებს მხოლოდ ვებ-ბრაუზერსა და ინტერნეტ-კავშირს;
- არ საჭიროებს განახლებებს. პროგრამული უზრუნველყოფის განახლება ხდება მხოლოდ ვებ-სერვერზე, ხოლო მომხმარებლები უკავშირდებიან რა სერვერს, ავტომატურად მუშაობენ განახლებულ ვერსიში. ეს გარანტიას იძლევა, რომ არც ერთი მომხმარებელი არ დარჩება ძველი ვერსიით და გამორიცხავს ვერსიებს შორის სხვაობით გამოწვეულ თავსებადობის პრობლემებს;
- სიტემის გამოყენებადობის ანალიტიკა. პროცესების დანახვა ბიზნეს-ხედვის კუთხით, თუ პროგრამული უზრუნველყოფის რა მოდულებს იყენებენ ხშირად მომხმარებლები, დიდ უპირატესობას იძლევა. ამ სტატისტიკაზე დაკვირვებით შესაძლებელი გახდა მომხმარებლისთვის ნაკლებად საინტერესო პროგრამული მოდულების გამოვლენა, რომელთა გაუმჯობესებით კვლავ დავუბრუნებთ აქტუალობას ამა თუ იმ პროგრამულ კომპონენტს;

• ნაკლები ხარჯი. პროგრამული უზრუნველყოფის შექმნის ხარჯი დესკტოპ-აპლიკაციების შემთხვევაში მატულობს, განსაკუთრებით, თუ მოთხოვნებში გათვალისწინებულია დანართის მუშაობა რამდენიმე პროგრამულ პლატფორმაზე (მაგალითად, OS X, PC და Linux).

მნიშვნელოვანია განვასხვავოთ ვებ-საიტები და ვებ-აპლიკაციები.

ვებ-საიტი ინფორმაციული ხასიათისაა და მომხმარებელს აწვდის გარკვეული სახის ინფორმაციას. იგი შეიძლება შევადაროთ საინფორმაციო ბროშურას, რომელიც აძლევს მომხმარებელს ურთიერთქმედების საშუალებას, მაგრამ შეზღუდული რაოდენობით.

ვებ-აპლიკაცია ინტერაქტიული საიტია და შეიცავს ინტერაქტიულ ელემენტებს. ამის მაგალითებია Wikipedia, Facebook, Gmail და სხვა. ვებ-აპლიკაციების დანიშნულება არის მომხმარებლებთან ურთიერთქმედება ინტერაქტიულ რეჟიმში. ვებ-აპლიკაცია შედგება მდიდარი პროგრამული ლოგიკისგან და მომხმარებელს სთავაზობს გარკვეული დავალების ან ამოცანების შესრულების საშუალებას.

## 2. ძირითადი ნაწილი

ნაშრომის მიზანია კორპორაციული ბიზნესპროცესების სამართავად სერვისებზე ორიენტირებული ვებ-აპლიკაციების აგების ახალი ტექნოლოგიების გამოკვლევა და შესაბამისი პროგრამული უზრუნველყოფის დაპროექტების, დეველოპმენტის და ტესტირების ერთიანი მეთოდოლოგიის ჩამოყალიბება [3].

თეორიული შედეგების ექსპერიმენტული რეალიზაცია განხორციელებულია ადამიანური რესურსების მართვის ელექტრონული სისტემის მაგალითზე. დასმული მიზნის მისაღწევად აუცილებელია შემდეგი ძირითადი ამოცანების გადაწყვეტა:

- კორპორაციული მართვის ბიზნეს-პროცესების კლასიფიკაცია და მათი სისტემური, დიაგნოსტიკური ანალიზი პრობლემების გამოვლენით;

- ბიზნეს-პროცესების ავტომატიზებული მართვის მხარდამჭერი სისტემის ასაგებად IT-სამსახურის მიზნების განსაზღვრა;

- მსოფლიოში არსებული ახალი უსაფრთხო ინფორმაციულ ტექნოლოგიათა პრაქტიკების მიმოხილვა, მათი ანალიზი და შესაბამისი კონცეფციის შემუშავება კორპორაციული მენეჯმენტის შემდგომი სრულყოფის მიზნით უახლესი ინფორმაციული მეთოდოლოგიების, სტანდარტების და ტექნოლოგიების ბაზაზე [4];

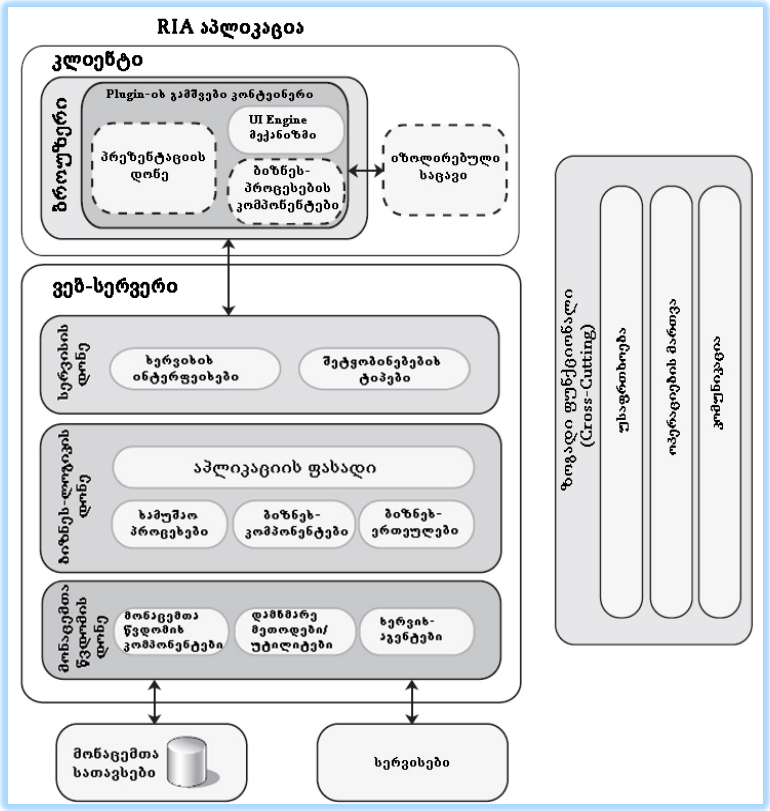
ტრადიციული ASP.NET Web Forms ტექნოლოგია დადებით მხარეებთან ერთად, უფრო გართულდა. ამის მიზეზები იყო: ViewState ობიექტის წონა, დიდი ზომის მონაცემთა ბლოკების ტრანსფერი კლიენტსა და სერვერს შორის; გვერდის სასიცოცხლო ციკლის სირთულე; მცდარი ილუზია განცალკევებული კონცეფციების შესახებ; HTML მარკირებაზე შეზღუდული კონტროლი; ტესტირების სისტემების დაბალი მხარდაჭერა.

Web Forms ტექნოლოგიის კრიტიკის საპასუხოდ Microsoft-ის კომპანიამ შეიმუშავა ახალი პროგრამული პლატფორმები. 2007 წლის ოქტომბერში ოფიციალურად გამოვიდა ASP.NET MVC ტექნოლოგია. მასში კომბინირებულია Model-View-Controller (MVC) არქიტექტურის ეფექტურობა, Agile Development მიდგომის უახლესი იდეოლოგია და ASP.NET პლატფორმის საუკეთესო ნაწილები [3]. MVC ტექნოლოგიის უპირატესობებია: MVC არქიტექტურა, განვრცობადობა, HTML კოდზე და HTTP პროტოკოლზე მაღალი ხარისხის კონტროლის მექანიზმები, ტესტირებადობა, მარშრუტიზაციის მძლავრი სისტემა და ASP.NET პლატფორმის საუკეთესო ნაწილების გამოყენება.

MVC ტექნოლოგიის შეზღუდვებად მოიაზრება მდიდარი სამომხმარებლო ინტერფეისის კონტროლების ნაკლებობა და ინტერაქტიულობის მისაღწევად ლოგიკის იმპლემენტირება თითქმის

თავიდან. გარდა ამისა, სხვადასხვა კონტროლის მდგომარეობის შენახვა-შენარჩუნება Web-Forms აპლიკაციებთან შედარებით რთული მისაღწევია ViewState ობიექტის არ არსებობის გამო.

ამ პრობლემების გადაწყვეტა შეუძლია ვებ-დაპროგრამების Silverlight ტექნოლოგიას, რომელიც უზრუნველყოფს მდიდარ სამომხმარებლო ინტერფეისს (Rich Interactive Applications - RIA) [5-7]. იგი ინტერაქტიულო აპლიკაციების შემუშავების საშუალებას იძლევა, კარგად ესადაგება თანამედროვე პროგრამული უზრუნველყოფების მიმართ წაყენებულ მოთხოვნებს, როგორიცაა: სამომხმარებლო ინტერფეისის სიმდიდრე და ხელმისაწვდომობის მაღალი ხარისხი; შეიცავს Runtime პაკეტს, რომელიც ეშვება კლიენტის კომპიუტერზე და არქიტექტურულად ზის მომხმარებელსა და სერვერს შორის. RIA-აპლიკაციები ხასიათდება უფრო კომპლექსური და მდიდარი კლიენტი მხარის (Client-Side) კოდის მხარდაჭერით, ვიდრე შესაძლებელი იყო ტრადიციული ვებ-აპლიკაციებისათვის. ამიტომაც ვებ-სერვერზე დატვირთვა „მსუბუქდება" და ნაწილდება კლიენტის მანქანაზე. 1-ელ ნახაზზე ნაჩვენებია ტიპური RIA-იმპლემენტაციის სტრუქტურა [7,8]. ტიპური RIA-აპლიკაცია დანაწილებულია პრეზენტაციის, ბიზნეს-ლოგიკის და მონეცემთა წვდომის სამ დონეზე (ნახ.2).




**ნახ.1. RIA-აპლიკაციის სტრუქტურა**

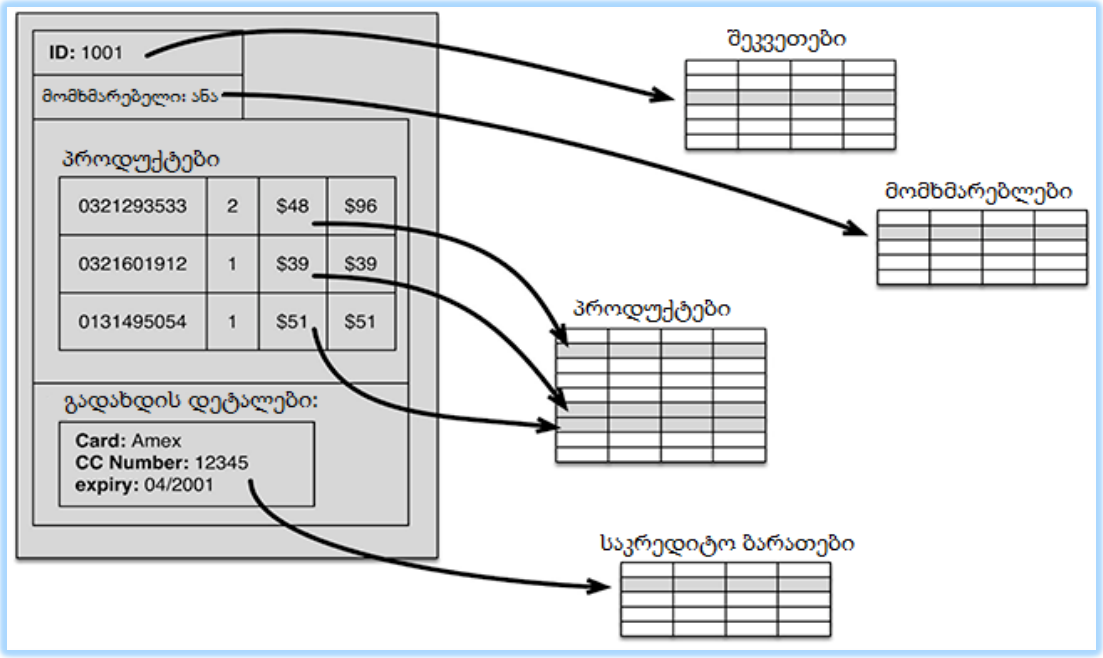

**ნახ.2. RIA აპლიკაციის განთავსება განაწილებულ სერვერებზე**

- პრეზენტაციის დონე შეიცავს სამომხმარებლო ინტერფეისს (UI) და პრეზენტაციის ლოგიკის კომპონენტებს;
- ბიზნეს-ლოგიკის დონე შეიცავს ბიზნეს-ლოგიკას, სამუშაო პროცესებს და ბიზნეს-ერთეულებს;
- მონაცემთა წვდომის დონე შეიცავს მონაცემთა წვდომის და სერვის-აგენტ კომპონენტებს.

### 3. მონაცემთა ბაზის გამოყენების კონცეფცია

ბოლო წლების განმავლობაში ლიტერატურულ წყაროებში გამოჩნდა და ვითარდება ახალი მიმართულება მონაცემთა „არარელაციური“ ბაზების შესახებ (NoSQL) [9-11]. იგი ძირითადად დიდ მონაცემთა ბაზებზეა (BigDatabase) ორიენტირებული. ასეთ სისტემებში რელაციური ალგებრის და SQL-ენის გამოყენების ეფექტიანობა ძალზედ კლებულობს.

ამიტომაც, დღეისათვის დიდი კორპორაციული ობიექტების განაწილებული საინფორმაციო სისტემების აგების მიზნით აქტუალური და პერსპექტიულია არარელაციური მონაცემთა ბაზების (NoRDB) გამოყენება [12]. მათ მონაცემთა საცავებში (Data Repository, Data Warehouse) მოთავსებულია ასეულობით მილიონი ჩანაწერი, რომელთა რელაციური ბაზებით დამუშავება (რელაციური ოპერაციების განხორციელების დროითი მახასიათებლების გაუარესების გამო) არაეფექტურია (ნახ.3).




**ნახ.3. არარელაციური და რელაციური მონაცემთა ბაზების სტრუქტურების შედარება**

როგორც წესი, მონაცემთა საცავებში NoSQL ბაზები ოპტიმიზირებულია OLAP და არა OLTP მოთხოვნებზე. შესაბამისად, თუ ცნობილია, რომ ამოცანის მიზნებიდან გამომდინარე, უფრო ხშირად გრძელი და მძიმე ტრანზაქციების გამოყენება იქნება საჭირო, გადაწყვეტილების მისაღებად უნდა განისაზღვროს, რამდენად ეფექტური იქნება ტრადიციული რელაციური ბაზების ნაცვლად არარელაციური ბაზების გამოყენება.

ზოგიერთ ამოცანაში ახალი ჩანაწერების დამატების სისწრაფე უფრო მნიშვნელოვანია, ვიდრე მათი გარანტირებული ჩაწერა (მაგალითად, ლოგ–სერვერი ან სოციალური ქსელი). ამ შემთხვევაში სისტემას აკონფიგურირებენ fire-and-forget პრინციპით, რაც გულისხმობს, რომ ახალ ჩანაწერებს უბრალოდ „ვისვრით“ და არ ველოდებით ჩაწერის დადასტურებას [11].

არარელაციურ ბაზებში მნიშვნელოვან საკითხად ითვლება სერვერებზე მონაცემთა სინქრონიზაციის პროცესი (Replication) და დიდი ბაზების დანაწევრების (Database partitioning) Sharding ტექნოლოგია.

რეპლიკაცია არის პროცესი, რომლის დროსაც მონაცემები სინქრონიზირდება ერთი, მთავარი სერვერიდან რამდენიმე სათადარიგო სერვერზე. რეპლიკაციის დროს მთავარ (primary) სერვერზე ჩაწერისა და კითხვის ოპერაციების განხორციელება შესაძლებელია, სათადარიგო სერვერებიდან კი მხოლოდ წაკითხვის [10]. რეპლიკაციას რამდენიმე ძირითადი დადებითი მხარე გააჩნია:

- მონაცემების კითხვის გაუმჯობესება – რამდენი ასლი სერვერიც დაკონფიგურირდება, იმდენი დამოუკიდებელი წყაროდან შეიძლება ინფორმაციის პარალელურ რეჟიმში წაკითხვა;
- Disaster Recovery – პრობლემის ან გეგმიური სამუშაოების დროს შესაძლებელია რომელიმე სათადარიგო (standby) სერვერის ძირითად (primary) სერვერად ქცევა;
- აღარ არის საჭირო backup-ებისა და ინდექსების რეორგანიზაციის გამო მონაცემთა ბაზის გაჩერება ან შენელება;

მთავარ სერვერთან კავშირის დაკარგვის ან გათიშვის შემთხვევაში სათადარიგო სერვერები აწყობს არჩევნებს და ირჩევს ახალ primary სერვერს.

Sharding ტექნოლოგია მონაცემთა ბაზის დანაწევრების (database partitioning) ერთ-ერთი სახეა. მისი მეშვეობით დიდი ზომის უმართავი ბაზა იყოფა პატარა, სწრაფ და ადვილად მართვად ნაწილებად [10]. Sharding–ისა და რეპლიკაციის დროს მოთხოვნის ზრდასთან ერთად შესაძლებელია ასობით და ათასობით ახალი მანქანის დამატება და თითოეული ახალი მანქანა გააუმჯობესებს მონაცემთა ბაზის სამივე ძირითად მახასიათებელს: პროცესორს, ოპერატიულ მეხსიერებას და ხისტ დისკოს.

ხშირად ადმინისტრატორები მონაცემთა ბაზებს საჭიროზე მეტ ოპერატიულსა და პროცესორებს უყოფენ. ამ დროს bottleneck (ბოთლის შევიწროებული ყელი) პრობლემას ვაწყდებით დისკების სიმცირეში. სერვერი დიდ დროს კარგავს დისკიდან ინფორმაციის ამოკითხვაზე. ეს პრობლემა განაწილებულ გარემოში მუშაობისას მინიმუმამდეა დაყვანილი:

- გვჭირდება სწრაფი კითხვა? – ვამატებთ ახალ რეპლიკასეტს, ანუ ბაზის ახალ ასლს;
- გვჭირდება სწრაფი ჩაწერა/წაკითხვა? – შარდინგ ტექნოლოგიით უფრო მცირე ნაწილებად ვყოფთ მონაცემთა ბაზას, შესაბამისად, უფრო მეტი დისკო წერს და კითხულობს ერთდროულად.

## 4. დასკვნა

კორპორაციული ობიექტების პროგრამული აპლიკაციების დამუშავება ეფექტურია მაიკროსოფთის ASP.NET MVC და Silverlight პლატფორმების გამოყენებით. შემოთავაზებულია რეკომენდაციები, რეალურ ამოცანებზე მუშაობისას, რა შემთხვევებში აჯობებს დაპროგრამების ამა თუ იმ ტექნოლოგიის არჩევა. მონაცემთა წვდომის და სწრაფი დამუშავების თვალსაზრისით მნიშვნელოვანი შედეგების მიღება შეიძლება კორპორაციული სისტემების დიდ მონაცემთა საცავებში არარელაციური (NoSQL) ბაზების გამოყენებით.

## ლიტერატურა –References – Литература:

1. სურგულაძე გ. (2015). კორპორაციული მენეჯმენტის სისტემების Windows დეველოპმენტი (WPF, Workflow, WCF ტექნოლოგიები). ნაწ.1,2,3. სტუ. „IT-კონსალტინგის ცენტრი“. თბილისი.

2. სურგულაძე გ., ბულია ი. (2012). კორპორაციულ Web-აპლიკაციათა ინტეგრაცია და დაპროექტება. მონოგრ., სტუ. თბილისი.

3. სურგულაძე გ., გულიტაშვილი მ., კივილაძე ნ. (2015). Web-აპლიკაციების ტესტირება, ვალიდაცია და ვერიფიკაცია. მონოგრ., სტუ. „IT-კონსალტინგის ცენტრი“. თბილისი.

4. სურგულაძე გ., ურუშაძე ბ. (2014). საინფორმაციო სისტემების მენეჯმენტის საერთაშორისო გამოცდილება (BSI, ITIL, COBIT). სახელმძღ., სტუ. „ტექნიკური უნივერსიტეტი“, თბილისი.

5. Software Architecture and Design. Microsoft. Msdn, Ch.1. https://msdn.microsoft.com/en-us/library/ee658093.aspx

6. Key Principles of Software Architecture. Common application architecture. Microsoft. Msdn, Ch.2. https://msdn.microsoft.com/en-us/library/ee658124.aspx

7. Designing Rich Internet Applications. Microsoft. Msdn,Ch.23. https://msdn.microsoft.com/en-us/library/ee658083.aspx

8. კივილაძე ნ. (2016). პროგრამული დანართის არქიტექტურა და RIA-აპლიკაციების დაპროექტება. სტუ-ს შრ.კრ. „მართვის ავტომატიზებული სისტემები", №1(21). გვ.243-248

9. Working with NoSQL Databases (.NET APIs for NoSQL). http://www.codeproject.com-/Articles/667371/Working-with-NoSQL-Databases

10. Jeremy Likness. Sterling NoSQL OODB for .NET 4.0, Silverlight 4 and 5, and Windows Phone 7. https://sterling.codeplex.com/

11. MongoDB manual – http://docs.mongodb.org/manual/

12. Implementation and API - http://api.mongodb.com/c/0.6/write_concern.html

13. Evolution of Database. https://mhaadi.wordpress.com/2010/10/18/theevolution of-database/

## CONSTRUCTION OF CORPORATE APPLICATION WITH SERVICE-ORIENTED SOFTWARE TECHNOLOGIES

Surguladze Gia, Kiviladze Nino, Kiviladze Giorgi
Georgian Technical University

### Summary

The work carries out the discussion about Microsoft's modern web development frameworks and applying them in the development of the large scale line of business applications. ASP.NET Web Forms, Silverlight and MVC programming frameworks are analyzed and provided comparisons between them. The advantages and disadvantages of each framework are carried out and some of the recommendations are provided for using these frameworks in real life scenarios. A practical implementation project is provided. A human resource management system (HRMS) web application is a software solution for small to mid-sized businesses to help automate and manage their HR, payroll, management and accounting activities. The application is developed using Microsoft Visual Studio, MsSQL Server software tools. We also consider the possibility of using NoSQL databases in these systems.

## ПОСТРОЕНИЕ КОРПОРАТИВНЫХ АППЛИКАЦИЙ С ПРИМЕНЕНИЕМ СЕРВИС-ОРИЕНТИРОВАННЫХ ПРОГРАММНЫХ ТЕХНОЛОГИЙ

Сургуладзе Г., Кивиладзе Н., Кивиладзе Г.
Грузинский Технический Университет

### Резюме

Обсуждаются современные фреймворки веб-девелопмента корпорации Microsoft и их применение в разработке крупномасштабных бизнес-приложений. ASP.NET Web Forms, Silverlight и MVC технологии программирования анализируются и представляются результаты их сравнения. Описываются преимущества и недостатки каждого из этих фреймворков и предлагаются некоторые рекомендации для их эффективного использования в реальных жизненных сценариев. Предлагается практическая задача разработки веб-аппликации для управления человеческими ресурсами на базе пакетов программ MsVisual Studio, MsSQL Server. Рассматривается также концепция возможности использования не-реляционных баз данных в этих системах.