SIMS-ის აპლიკაციის სერვერსა და ინფორმაციის მომწოდებლებს შორის დგას ე.წ. Service Bus-ი, რომელიც ყველა მომწოდებელთან მუშაობს ერთგვაროვანი პროტოკოლით. ეს პროტოკოლი შემუშავებული და შეთანხმებული იქნა ბაკურიანის შეხვედრაზე, 2011 წლის დეკემბერში სხვა სამთავრობო სტრუქტურებთან ერთად.

პროტოკოლის ძირითადი კონცეფცია ემყარება ზოგად მოდელს, რომ SIMS-ისთვის საინტერესოა:

- გარკვეული ობიექტების (უმეტეს შემთხვევაში პიროვნებები) სიმრავლე
- შესაბამის უწყებებში ობიექტების ჩანაწერებზე განხორციელებული ცვლილებები
- ცვლილებების კლასიფიკაცია და თარიღობრივი შუალედებით ობიექტების იდენტიფიკატორების სიის მიღება.
- ობიექტის ფილტრაცია სამომავლოდ გაფართოებადი ფილტრების გამოყენებით.

შეხვედრაზე შევთანხმდით, რომ ინფორმაციის თითოეული მომწოდებელი შექმნიდა სერვისს, რომელსაც ექნებოდა ორი ძირითადი მეთოდის მხარდაჭერა:

```
EventsResult GetEvents(fromDate, toDate, eventClass[]); - მოვლენების სერვისი
ObjectsResult GetObject(objectID, fromDate, toDate, eventClass[]); - ობიექტის სერვისი

სადაც:
ფილტრის სტრუქტურა:
eventClass {
        eventID, - ცვლილების კლასიფიკატორი
        NameValueCollection; - ფილტრების ნუსხა
}

მოვლენების სერვისის დაბრუნებული რეზულტატის სტრუქტურა:
EventResult
{
        EventObject[] { - გადაცემული ფილტრით მოძიებული მოვლენების სიმრავლე
                EventID, - მოვლენის იდენტიფიკატორი
                EventDate, - მოვლენის თარიღი
                ObjectID[] - ობიექტების სიმრავლე, რომელზეც განხორციელდა მითითებული მოვლენა მითითებულ
                თარიღში
        }
}

ObjectsResult { - ობიექტების სიმრავლე (განსხვავებულია თითოეული პროვაიდერისთვის).
        Person[] { - ობიექტების, როგორც პიროვნებების სიმრავლე.
                Pin, - პირადი ნომერი
                FirstName,- სახელი
                LastName, - გვარი
                BirthDate, - დაბადების თარიღი
                ObjectData - პროვაიდერიზე დამოკიდებული ინფორმაციის დამატებითი სტრუქტურა
                }
}
ObjectData { - პროვაიდერის მიხედვით ნებისმიერი სტრუქტურა.
        ...
}
```

ამ შენთახმებისა და პრინციპების დაცვითაა აგებული SIMS-ის გარესამყაროსთან ინტეგრაციის სერვისები.

განვიხილოთ თითოეული ცალცალკე:

სამოქალაქო რეესტრი:

სერვისის ზოგადი აღწერა (დანიშნულება)
მოქალაქეთა რეესტრიდან მოქალაქეების შესახებ დეტალური ინფორმაციის დადგენა საჭიროებების მიხედვით. შესაძლებელია პირადი ნომრით, ან სახელი/გვარი/დაბადების წელი ფილტრებით პირების ძებნა. პირადი ნომრის მიხედვით, ასევე შესაძლებელია სურათების მოძიებაც. კონკრეტული თარიღისთვის ცვლილებების დადგენა და პირადი ნომრების სიის მიღება
უწყებები ან/და ელექტრონული სისტემები (სერვისის გამცემი და მიმღები მხარეები, მათ შორის გამტარი მხარეებიც)
აგებულია გამოყოფილი დაცული არხი (VPN) ჯანდაცვის სამინისტროს სერვერებსა და სახელმწიფო სერვისების განვითარების რეესტრის სერვერებს შორის.
სერვისზე პასუხისმგებელი პირი

- SIMS-ის ინფორმაციული სისტემის მხარდაჭერასა და გამართულ მუშაობაზე პასუხისმგებელი გუნდი:
  ○ მონაცემთა გაცვლისა და SIMS-ის სისტემაში ინტეგრირების გამართული მუშაობა;
  ○ დეველოპმენტის, სატესტო და რეალურ აპარატურულ/პროგრამულ გარემოებში სერვისის ინსტალაცია ან/და გამართულად ჰოსტინგის უზრუნველყოფა.
- ჯანდაცვის სამინისტრო:
  ○ აპარატურული, საკომუნიკაციო და საოპერაციო ინფრასტრუქტურის გამართული მუშაობის უზრუნველყოფა.

URL, სერთიფიკატის საჭიროება
SIMS-ის სერვისთან აუთენტიფიკაციისთვის გამოიყენება SIMS-ის Secured Token Service (STS) ინფრასტრუქტურა. მომხმარებელი, ვერ გამოიყენებს სერვისს, თუ მას არ აქვს აუთენტიფიკაცია გავლილი SIMS-ის შესაბამის STS სერვისზე და მისგან მიღებული დაცული ბილეთი (Token) ვადაგასულია. URL დეველოპმენტის გარემო: http://172.17.8.125:10080/CRAProxyService/CRAService.svc სატესტო გარემო: http://172.17.16.15/CRAProxyServiceTest/CRAService.svc რეალური გარემო: http://172.17.16.15/CRAProxyService/CRAService.svc Worknet: http://172.17.16.15/CRAProxyService/CRAService.svc
სერვისის კონტრაქტისა და მეთოდების დეტალური აღწერა

სერვისის კონტრაქტი წარმოადგენს 5 მეთოდისგან შემდგარ კლასს:
`/// <summary>`
`/// პირების სიის მიღება სახელის, გვარისა და დაბადების თარიღის გაერთიანებული კრიტერიუმებით.`
`/// იყენებს სამოქალაქო რეესტრის სერვისის GetDataUsingCriteria მეთოდს`
`/// </summary>`
`/// <param name="request">იხილეთ <paramref name="GetObjectsRequest"/> კლასის აღწერა</param>`
`/// <returns>იხილეთ <paramref name="GetCRAObjectsResponse"/> კლასის აღწერა</returns>`
`[OperationContract]`
`GetCRAObjectsResponse GetObjects(GetObjectsRequest request);`
 	
`/// <summary>`
`/// პირადი ნომრის მიხედვით სურათების მიღება`
`/// </summary>`
`/// <param name="request">იხილეთ <paramref name="GetObjectRequest"/> კლასის აღწერა</param>`
`/// <returns>იხილეთ <paramref name="GetObjectImagesResponse"/> კლასის აღწერა</returns>`
`[OperationContract]`
`GetObjectImagesResponse GetObjectImages(GetObjectRequest request);`
 	
`/// <summary>`
`/// პირადი ნომრის მიხედვით პიროვნების მონაცემების მიღება`
`/// </summary>`
`/// <param name="request">იხილეთ <paramref name="GetObjectRequest"/> კლასის აღწერა</param>`
`/// <returns>იხილეთ <paramref name="GetCRAObjectResponse"/> კლასის აღწერა</returns>`
`[OperationContract]`
`GetCRAObjectResponse GetObject(GetObjectRequest request);`

```
        /// <summary>
        /// ცვლილებების სიის მიღება მოვლენის მიხედვით. ამ ეტაპზე რეალიზებულია, მხოლოდ გარდაცვალების მოვლენის დამუშავება.
        /// </summary>
        /// <param name="request">იხილეთ <paramref name="GetObjectRequest"/> კლასის აღწერა</param>
        /// <returns>იხილეთ <paramref name="GetCRAObjectsResponse"/> კლასის აღწერა</returns>
        [OperationContract]
        GetEventsResponse GetEventsList(GetEventsRequest request);

        /// <summary>
        /// მოქალაქეთა რეესტრის საცნობარო მასალების სიის მიღება
        /// </summary>
        /// <returns>იხილეთ <paramref name="ReferenceDataResponse"/> კლასის აღწერა</returns>
        [OperationContract]
        ReferenceDataResponse GetReferenceData();
    }
```

სერვისში გამოყენებული კლასების აღწერა იხილეთ ქვემოთ:

```
    /// <summary>
    /// მონაცემთა კონტრაქტები გაერთიანებულია http://SIMS.ExternalServices/CRAService/Entities Namespace-ში.
    /// პირების სიის მიღების სერვისის პარამეტრების კლასი
    /// </summary>
    [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class GetObjectsRequest
    {
        /// <summary>
        /// პირადი ნომერი
        /// </summary>
        [DataMember]
        public string PIN { get; set; }
        /// <summary>
        /// სახელი
        /// </summary>
        [DataMember]
        public string FirstName { get; set; }
        /// <summary>
        /// გვარი
        /// </summary>
        [DataMember]
        public string LastName { get; set; }
        /// <summary>
        /// დაბადების წელი
        /// </summary>
        [DataMember]
        public short? BirthYear { get; set; }
        /// <summary>
        /// დაბადების თვე
        /// </summary>
        [DataMember]
        public short? BirthMonth { get; set; }
        /// <summary>
        /// დაბადების რიცხვი
        /// </summary>
        [DataMember]
        public short? BirthDay { get; set; }
    }


    /// <summary>
    /// მონაცემთა კონტრაქტები გაერთიანებულია http://SIMS.ExternalServices/CRAService/Entities Namespace-ში.
    /// პირების სიის მიღების სერვისის დაბრუნებული მნიშვნლობების კლასი
    /// ასევე იხილეთ <seealso cref="BaseResponse"/> საბაზისო კლასის აღწერა
    /// </summary>
    [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class GetCRAObjectsResponse : BaseResponse
    {
        /// <summary>
        /// პირების სიმრავლე
        /// </summary>
        [DataMember]
        public CRAPerson[] Persons { get; set; }
    }


    /// <summary>
    /// სერვისებიდან მიღებული პასუხის საბაზისო კლასი
    /// </summary>
    [DataContract]
    public class BaseResponse
```

```
    {
        /// <summary>
        /// შეცდომის კოდი - მნიშვნელობები დამოკიდებულია სერვისის პროვაიდერზე
        /// </summary>
        [DataMember]
        public string ErrorCode { get; set; }

        /// <summary>
        /// შეცდომის განმარტება - მნიშვნელობები დამოკიდებულია სერვისის პროვაიდერზე
        /// </summary>
        [DataMember]
        public string ErrorMessage { get; set; }
    }


    /// <summary>
    /// მოქალაქეთა რეესტრიდან მიღებული პიროვნების მონაცემების კლასი. ანალოგიურია სამოქალაქო რეესტრის
კლასისა.
    /// </summary>
    [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class CRAPerson : PersonBase
    {
        /// <summary>
        /// მამის სახელი
        /// </summary>
        [DataMember]
        public string FatherName { get; set; }
        /// <summary>
        /// დაბადების ადგილი
        /// </summary>
        [DataMember]
        public string BirthPlace { get; set; }
        /// <summary>
        /// მოქალაქეობის ქვეყნის დასახელება
        /// </summary>
        [DataMember]
        public string Citizenship { get; set; }
        /// <summary>
        /// მოქალაქეობის ქვეყნის კოდი
        /// </summary>
        [DataMember]
        public string CitizenshipCode { get; set; }
        /// <summary>
        /// დოკუმენტის იდენტიფიკატორი
        /// </summary>
        [DataMember]
        public long DocumentID { get; set; }
        /// <summary>
        /// დოუმენტის სტატუსი
        /// იხილეთ <see cref="CRADocumentStatus"/> ჩამონათვალის აღწერა
        /// </summary>
        [DataMember]
        public CRADocumentStatus? DocumentStatus { get; set; }
        /// <summary>
        /// დოკუმენტის ტიპი
        /// </summary>
        [DataMember]
        public short? DocumentTypeID { get; set; }
        /// <summary>
        /// დოკუმენტის სტატუსის დასახელება
        /// </summary>
        [DataMember]
        public string DocumentStatusName { get; set; }
        /// <summary>
        /// ორმაგი მოქალაქეობის ქვეყნის დასახელება
        /// </summary>
        [DataMember]
        public string DoubleCitizenship { get; set; }
        /// <summary>
        /// ორმაგი მოქალაქეობის ქვეყნის კოდი
        /// </summary>
        [DataMember]
        public string DoubleCitizenshipCode { get; set; }
        /// <summary>
        /// ფაქტიური საცხოვრებელი მისამართი
        /// </summary>
        [DataMember]
        public string FaLivingPlace { get; set; }
        /// <summary>
        /// სქესი
        /// </summary>
```

```
        [DataMember]
        public System.Nullable<short> Gender { get; set; }
        /// <summary>
        /// IDCard-ის გაცემის თარიღი
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> IdCardIssueDate { get; set; }
        /// <summary>
        /// IDCard-ის გამცემი ორგანო
        /// </summary>
        [DataMember]
        public string IdCardIssuer { get; set; }
        /// <summary>
        /// IDCard-ის ნომერი
        /// </summary>
        [DataMember]
        public string IdCardNumber { get; set; }
        /// <summary>
        /// IDCard-ის სერია
        /// </summary>
        [DataMember]
        public string IdCardSerial { get; set; }
        /// <summary>
        /// IDCard-ის ვადა
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> IdCardValidDate { get; set; }
        /// <summary>
        /// მონაცემები IDCard-ის საფუძველზეა თუ არა წამოღებული.
        /// </summary>
        [DataMember]
        public bool IsIdCard { get; set; }
        /// <summary>
        /// რეგისტრაციის მისამართი ტექსტურად
        /// </summary>
        [DataMember]
        public string LivingPlace { get; set; }
        /// <summary>
        /// რეგისტრაციის მისამართის იდენტიფიკატორი, მისამართების სტრუქტურიდან
        /// </summary>
        [DataMember]
        public System.Nullable<long> LivingPlaceID { get; set; }
        /// <summary>
        /// რეგისტრაციის მისამართის რეგისტრაციის თარიღი
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> LivingPlaceRegDate { get; set; }
        [DataMember]
        /// <summary>
        /// რეგისტრაციის მისამართის რეგისტრაციის დასრულების თარიღი
        /// </summary>
        public System.Nullable<System.DateTime> LivingPlaceRegEndDate { get; set; }
        /// <summary>
        /// პასპორტი გაცემის თარიღი
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> PaspIssueDate { get; set; }
        /// <summary>
        /// პასპორტის გამცემი ორგანო
        /// </summary>
        [DataMember]
        public string PaspIssuer { get; set; }
        /// <summary>
        /// პასპორტის ნომერი
        /// </summary>
        [DataMember]
        public string PaspNumber { get; set; }
        /// <summary>
        /// პასპორტის ვადა
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> PaspValidDate { get; set; }
        /// <summary>
        /// რეგიონის იდენტიფიკატორი, მისამართების სტრუქტურდან, რეგისტრაციის მისამართის მიხედვით
        /// </summary>
        [DataMember]
        public System.Nullable<long> RegionID { get; set; }
        /// <summary>
        /// რეგიონის დასახელება
        /// </summary>
        [DataMember]
```

```
        public string RegionName { get; set; }
        /// <summary>
        /// გაუქმების თარიღი
        /// </summary>
        [DataMember]
        public System.Nullable<System.DateTime> RejectedDate { get; set; }
        /// <summary>
        /// სამოქალაქო რეესტრიდან მიღებული პასუხის კოდი
        /// იხილეთ <see cref="CRAResponseStatus"/> ჩამონათვალის აღწერა
        /// </summary>
        [DataMember]
        public CRAResponseStatus ResponseCode { get; set; }
        /// <summary>
        /// სამოქალაქო რეესტრის პასუხის იდენტიფიკატორი
        /// </summary>
        [DataMember]
        public string ResponseID { get; set; }
        /// <summary>
        /// ჩანაწერის სტატუსი - იხილეთ სამოქალაქო რეესტრის დოკუმენტაცია
        /// </summary>
        [DataMember]
        public short? Status { get; set; }
        /// <summary>
        /// პიროვნების სტატუსი - იხილეთ სამოქალაქო რეესტრის დოკუმენტაცია
        /// </summary>
        [DataMember]
        public short? PersonStatus { get; set; }
        /// <summary>
        /// პიროვნების მდგომარეობა - იხილეთ სამოქალაქო რეესტრის დოკუმენტაცია
        /// </summary>
        [DataMember]
        public long? PersonCondition { get; set; }
        /// <summary>
        /// ჩანაწერის მდგომარეობა - იხილეთ სამოქალაქო რეესტრის დოკუმენტაცია
        /// </summary>
        [DataMember]
        public long? Condition { get; set; }
        /// <summary>
        /// პიროვნება გარდაცვლილია თუ ცოცხალი.
        /// </summary>
        [DataMember]
        public bool IsPersonDead { get; set; }
    }


    /// <summary>
    /// საქმოქალაქო რეესტრის დოკუმენტის სტატუსი
    /// </summary>
    public enum CRADocumentStatus
    {
        /// <summary>
        /// გაუქმებული
        /// </summary>
        [EnumMember]
        Rejected,
        /// <summary>
        /// აქტიური
        /// </summary>
        [EnumMember]
        Active
    }


    /// <summary>
    /// სამოქალაქო რეესტრიდან მიღებული რეზულტატის კოდები
    /// </summary>
    public enum CRAResponseStatus
    {
        /// <summary>
        /// გაურკვეველი შეცდომა
        /// </summary>
        [EnumMember]
        UNEXPETC_ERROR,
        /// <summary>
        /// IDCard-ის ფორმატის შეცდომა
        /// </summary>
        [EnumMember]
        ID_CARD_FORMAT_ERROR,
        /// <summary>
        /// IDCard-ის პარამეტრების შეცდომა
        /// </summary>
```

```
        [EnumMember]
        ID_CARD_PARAMETERS_ERROR,
        /// <summary>
        /// პირადი ნომრის ფორმატის შეცდომა
        /// </summary>
        [EnumMember]
        PRIVATE_NUMBER_FORMAT_ERROR,
        /// <summary>
        /// სერვისი შესრულდა წარმატებით
        /// </summary>
        [EnumMember]
        OK,
        /// <summary>
        /// პირადი ნომერი ვერ იქნა მოძიებული
        /// </summary>
        [EnumMember]
        PRIVATE_NUMBER_NOT_FOUND,
        /// <summary>
        /// IDCard-ის სერია არ არის სწორი
        /// </summary>
        [EnumMember]
        ID_CARD_SERIAL_NOT_MATCHED,
        /// <summary>
        /// IDCard-ის ნომერი არ არის სწორი
        /// </summary>
        [EnumMember]
        ID_CARD_NUMBER_NOT_MATCHED,
        /// <summary>
        /// IDCard-ი ვერ იქნა მოძიებული
        /// </summary>
        [EnumMember]
        ID_CARD_NOT_FOUND,
        /// <summary>
        /// ინფორმაცია პიროვნების შესახებ დახურულია
        /// </summary>
        [EnumMember]
        PERSON_INFO_IS_CLOSED,
    }


    /// <summary>
    /// პიროვნების ინფორმაციის მიღების სერვისის რეზულტატი
    /// ასევე იხილეთ <seealso cref="BaseResponse"/> საბაზისო კლასის აღწერა
    /// </summary>
    [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class GetCRAObjectResponse : BaseResponse
    {
        /// <summary>
        /// მოქალაქის ინფორმაცია
        /// იხილეთ <see cref="CRAPerson"/> კლასის აღწერა
        /// </summary>
        [DataMember]
        public CRAPerson Object { get; set; }

        /// <summary>
        /// მოქალაქის ინფორმაცია გარდაცვალების შესახებ
        /// იხილეთ <see cref="CRAPersonDeathInfo"/> კლასის აღწერა
        /// ამ ინფორმაციის სიზუსტისთვის სამოქალაქო რეესტრს უნდა დაემატებინა ახალი ველები, რომელიც ჯერ არ დაუმატებია.
        /// </summary>
        [DataMember]
        public CRAPersonDeathInfo DeathInfo { get; set; }
    }


    /// <summary>
    /// პიროვნების გარდაცვალების ინფორმაციის აღწერის კლასი
    /// </summary>
    [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class CRAPersonDeathInfo
    {
        /// <summary>
        /// გარდაცვალების რეალური თარიღი
        /// </summary>
        [DataMember]
        public DateTime? DeathDate { get; set; }
        /// <summary>
        /// გარდაცვალების რეგისტრაციის თარიღი
        /// </summary>
        [DataMember]
        public DateTime? RegistrationDate { get; set; }
```

```
    }


    /// <summary>
    /// პიროვნების სურათების მიღების სერვისის რეზულტატის კლასი
    /// ასევე იხილეთ <seealso cref="BaseResponse"/> საბაზისო კლასის აღწერა
    /// </summary>
   [DataContract(Namespace = CRA.NS.DataContractsNS)]
    public class GetObjectImagesResponse : BaseResponse
    {
       /// <summary>
        /// სერვისის რეზულტატი წარმოადგენს სურათების მასივს.
        /// სურათი აღწერილია ბაიტების მასივის სახით
       /// </summary>
        [DataMember]
        public byte[][] Images { get; set; }
    }


    /// <summary>
    /// მოვლენების სიის მიღების სერვისის პარამეტრები
    /// </summary>
    [DataContract]
    public class GetEventsRequest
    {
        /// <summary>
        /// მოვლენის თარიღის საწყისი ფილტრი
        /// </summary>
        [DataMember]
        public DateTime? FromDate { get; set; }

        /// <summary>
        /// მოვლენის თარიღის საბოლოო ფილტრი
        /// </summary>
        [DataMember]
        public DateTime? ToDate { get; set; }

        /// <summary>
        /// მოვლენის კლასიფიკატორის მასივი
        /// იხილეთ <see cref="EventClass"/> კლასის აღწერა
        /// </summary>
        [DataMember]
        public EventClass[] Events { get; set; }
    }


    /// <summary>
    /// მოვლენების კლასიფიკატორისა და პარამეტრების კლასი
    /// </summary>
    [DataContract]
    public class EventClass
    {
        /// <summary>
        /// მოვლენის იდენტიფიკატორი - მნიშვნელობა დამოკიდებულია სერვის პროვაიდერზე
        /// </summary>
        [DataMember]
        public int EventID { get; set; }
        /// <summary>
        /// მოვლენების ფილტრების Dictionary, სადაც Key წარმოადგენს პარამეტრის დასახელებას, ხოლო Value
მნიშვნელობას.
        /// </summary>
        [DataMember]
        public Dictionary<string, object> EventParameters { get; set; }
    }
```

მოვლენების სახეობები აღწერილია ჩამონათვალში:

```
    /// <summary>
    /// მოვლენების კლასიფიკატორი
    /// </summary>
    [DataContract]
    public enum EventTypes : int
    {
        /// <summary>
        /// ნულოვანი მნიშვნელობა
        /// </summary>
        [EnumMember]
        Nothing = 0,
        /// <summary>
        /// დაბადება
        /// </summary>
```

```
        [EnumMember]
        Birth = 1,
        /// <summary>
        /// მოქალაქეობიდან გასვლა
        /// </summary>
        [EnumMember]
        CitizenshipLeave = 2,
        /// <summary>
        /// მოქალაქეობის მიღება
        /// </summary>
        [EnumMember]
        CitizenshipEntrance = 3,
        /// <summary>
        /// გარდაცვალება
        /// </summary>
        [EnumMember]
        Death = 4

    }
```

ამ ჩამონათვალიდან რეალიზებულია მხოლოდ `Death` მოვლენა, რადგან სხვა ცვლილებების იდენტიფიცირება სერვის პროვაიდერთან შეუძლებელია.

აღსანიშნავია, რომ SIMS-ი საკუთარ სერვისს უკავშირდება წინასწარგამზადებული კლასით `CRAClient`, რომელიც ერთერთ მეთოდში (`GetPersonsSIMSWrapped`) ახდენს გარკვეულ გამოთვლებს. მეთოდი აბრუნებს `CRAPersonSIMSWrapper` კლასის ობიექტს, რომლის აღწერაც იხილეთ ქვემოთ:

```
    /// <summary>
    /// 1.    გარდაცვალების აღნიშვნა
    /// PersonStatus=2 და PersonCondition=2
    /// 2.    საქართველოს მოქალაქე
    /// PersonStatus=1 და  PersonCondition თავსდება (2,3,6,7,9,10,14,44,50,51,52,53)
    /// ან
    /// PersonStatus=1 და  PersonCondition თავსდება (1,4,5,8,11,2,1) და დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (22,7,8,11,26,12,27,21,39,1,14,49,29,16,20,19) და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს: „მიიღო უცხო ქვეყნის მოქალაქეობა“ ან „საქართველოს მოქალაქეობიდან გასვლა“ ან „საქართველოს მოქალაქეობის შეწყვეტა“
    /// 3.    საქართველოს ორმაგი მოქალაქე
    /// არის საქართველოს მოქალაქე და
    /// PersonStatus=1 და  PersonCondition თავსდება (7,9,51,52) ან
    /// ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა (DoubleCitizenship) არ არის „საქართველო“, „ცარიელი“, „არმქონე“  და ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა ((DoubleCitizenship)) შეესაბამება რომელიმე ქვეყნის დასახელების მნიშვნელობას (ამ ნაწილში შესაძლებელია გამოიყენოთ ორმაგი მოქალაქეობის ქვეყნის კოდი DoubleCitizenshipCode (GEO, ENG  და ა.შ.)).
    /// 4.    მოქალაქეობის არმქონე პირი
    /// დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42,3,40,2) და მოქალაქეობის ქვეყნის დასახელების მნიშვნელობაში (Citizenship)  ფიქსირდება „არმქონე“ და PersonStatus=1 და PersonCondition არ თავსდება (2,3,6,7,9,10,14,44,50,51,52,53) და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს „მიიღო უცხო ქვეყნის მოქალაქეობა“, „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“
    /// 5.    პატიმრობა
    /// PersonStatus=1 და  PersonCondition თავსდება (4,11).
    /// 4 - პატიმრობა; 11 - წინასწარი დაკავება
    /// 6.    პირადი ნომრის გაუქმება გაყალბება
    /// თუ ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) მოიცავს „ყალბ“ ან  „პირადი ნომრის გაორება“ ან  „პირადი ნომრის გაუქმება“ ან  „პირადი ნომრის შეცვლა“ ან  „შეცდომა პირად ნომერში“ ან „ბათილად ცნობა“ („რეგისტრაციის ბათილად ცნობა“-ის გარდა) ან  „შეცდომა პირად ნომერში“
    /// </summary>
    public class CRAPersonSIMSWrapper
    {
        public CRAPerson Person { get; set; }

        public int CRAStatusValueSetID { get; private set; }
        /// <summary>
        /// თავისუფლების აღკვეთა - პატიმრობა
        /// </summary>
```

```
        public int ImprisonmentInPrisonValueSetID { get; private set; }
        /// <summary>
        /// თავისუფლების აღკვეთა - წინასწარი დაკავება
        /// </summary>
        public int ImprisonmentPreTrialDetentionValueSetID { get; private set; }
        public int CitizenshipStatusValueSetID { get; private set; }

        /// <summary>
        /// 1.       გარდაცვალების აღნიშვნა
        /// PersonStatus=2 და PersonCondition=2
        /// </summary>
        public bool? IsPersonDead { get; private set; }
        /// <summary>
        /// 2.       საქართველოს მოქალაქე
        /// PersonStatus=1 და  PersonCondition თავსდება (2,3,6,7,9,10,14,44,50,51,52,53)
        /// ან
        /// PersonStatus=1 და  PersonCondition თავსდება (1,4,5,8,11,2,1)
        /// და დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება
(22,7,8,11,26,12,27,21,39,1,14,49,29,16,20,19)
        /// და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს: „მიიღო უცხო ქვეყნის
მოქალაქეობა“ ან „საქართველოს მოქალაქეობიდან გასვლა“ ან „საქართველოს მოქალაქეობის შეწყვეტა“
        /// </summary>
        public bool? IsGeorgianCitizen { get; private set; }
        /// <summary>
        /// 3.       საქართველოს ორმაგი მოქალაქე
        /// არის საქართველოს მოქალაქე და
        /// PersonStatus=1 და  PersonCondition თავსდება (7,9,51,52) ან
        /// ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა (DoubleCitizenship) არ არის
„საქართველო“, „ცარიელი“, „არმქონე“
        /// და ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა ((DoubleCitizenship)) შეესაბამება
რომელიმე ქვეყნის დასახელების მნიშვნელობას
        /// (ამ ნაწილში შესაძლებელია გამოიყენოთ ორმაგი მოქალაქეობის ქვეყნის კოდი
DoubleCitizenshipCode (GEO, ENG  და ა.შ.)).
        /// </summary>
        /// <param name="person"></param>
        /// <returns></returns>
        public bool? IsDoubleCitizen { get; private set; }
        /// <summary>
        /// 4.       მოქალაქეობის არმქონე პირი
        /// დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42,3,40,2)
        /// და მოქალაქეობის ქვეყნის დასახელების მნიშვნელობში (Citizenship)  ფიქსირდება „არმქონე“
        /// და PersonStatus=1 და  PersonCondition არ თავსდება (2,3,6,7,9,10,14,44,50,51,52,53)
        /// და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს „მიიღო უცხო ქვეყნის
მოქალაქეობა“, „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“
        /// </summary>
        public bool? IsNoCitizen { get; private set; }
        /// <summary>
        /// 5.       პატიმრობა
        /// PersonStatus=1 და  PersonCondition თავსდება (4,11).
        /// 4 - პატიმრობა;
        /// </summary>
        public bool? IsZek { get; private set; }
        /// <summary>
        /// 5.       პატიმრობა
        /// PersonStatus=1 და  PersonCondition თავსდება (4,11).
        /// 11 - წინასწარი დაკავება
        /// </summary>
        public bool? IsZekInAdvance { get; private set; }

        /// <summary>
        /// 6.       პირადი ნომრის გაუქმება გაყალბება
        /// თუ ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) მოიცავს „ყალბ“ ან  „პირადი ნომრის
გაორება“ ან  „პირადი ნომრის გაუქმება“ ან  „პირადი ნომრის შეცვლა“ ან  „შეცდომა პირად ნომერში“ ან
„ბათილად ცნობა“ („რეგისტრაციის ბათილად ცნობა“-ის გარდა) ან  „შეცდომა პირად ნომერში“
        /// </summary>
        public bool? IsPINCanceled { get; set; }
```

```
        /// <summary>
        /// 7.       საქართველოში მცხოვრები უცხო ქვეყნის მოქალაქე (მობინადრე)
        ///I ვარიანტი
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (1, 4, 5, 8, 11, 12, 13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3, 40, 2) და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს: „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“ და
        ///როდესაც დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3) მაშინ დოკუმენტის მოქმედებს ვადა (IdCardValidDate) მეტი უნდა იყოს მიმდინარე თარიღზე
        ///
        ///II ვარიანტი
        ///პირი გავიდა საქართველოს მოქალაქეობიდან და მასზე გაცემულია ბინადრობის ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია:
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (12,13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (21,39,1,16)
        ///III ვარიანტი
        ///არის მცირე რაოდენობა შემთხვევებისა, როდესაც პირს შეუწყდა ბინადრობის ნებართვა  და მასზე გაცემულია ახალი ბინადრობის ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია.
        ///ამ სიტუაციის დასარეგულირებლად ჩვენ ვიყენებთ ახალი ბინადრობის ნებართვის გაცემის ფაქტის აღნიშვნის თარიღს რაც, სამწუხაროდ ვებ სერვისით არ მოგეწოდებათ აქედან გამომდინარე ამ პრობლემას ბოლომდე მოგვარება ამ ეტაპზე შეუძლებელია. თუმცა მომსახურებისთვის თქვენ თუ გამოიყენებთ აქტიური დოკუმენტის არსებობობის პრინციპს ეს პრობლემა მოიხსნება.
        ///ასეთ შემთხვევაში სიტუაცია შემდეგი იქნება
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (12,13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3, 40, 2)
        ///და
        ///ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) მოიცავს: „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“ ან დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3) და დოკუმენტის მოქმედებს ვადა (IdCardValidDate) ნაკლებია მიმდინარე თარიღზე.
        ///7. საქართველოში მცხოვრები უცხო ქვეყნის მოქალაქე (მობინადრე)
        ///I ვარიანტი
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (1, 4, 5, 8, 11, 12, 13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3, 40, 2) და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს: „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“ და
        ///როდესაც დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3) მაშინ დოკუმენტის მოქმედებს ვადა (IdCardValidDate) მეტი უნდა იყოს მიმდინარე თარიღზე
        ///
        ///II ვარიანტი
        ///პირი გავიდა საქართველოს მოქალაქეობიდან და მასზე გაცემულია ბინადრობის ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია:
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (12,13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (21,39,1,16)
        ///III ვარიანტი
        ///არის მცირე რაოდენობა შემთხვევებისა, როდესაც პირს შეუწყდა ბინადრობის ნებართვა  და მასზე გაცემულია ახალი ბინადრობის ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია.
        ///ამ სიტუაციის დასარეგულირებლად ჩვენ ვიყენებთ ახალი ბინადრობის ნებართვის გაცემის ფაქტის აღნიშვნის თარიღს რაც, სამწუხაროდ ვებ სერვისით არ მოგეწოდებათ აქედან გამომდინარე ამ პრობლემას ბოლომდე მოგვარება ამ ეტაპზე შეუძლებელია. თუმცა მომსახურებისთვის თქვენ თუ გამოიყენებთ აქტიური დოკუმენტის არსებობობის პრინციპს ეს პრობლემა მოიხსნება.
        ///ასეთ შემთხვევაში სიტუაცია შემდეგი იქნება
        ///არ არის საქართველოს მოქალაქე და არ არის მოქალაქეობის არმქონე პირი და
        ///PersonStatus=1 და  PersonCondition თავსდება (12,13) და
        ///დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3, 40, 2)
        ///და
```

```
        ///ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) მოიცავს: „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“ ან დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42, 3) და დოკუმენტის მოქმედებს ვადა (IdCardValidDate) ნაკლებია მიმდინარე თარიღზე.
        /// </summary>
        public bool? IsPermanentResident { get; set; }

        public CRAPersonSIMSWrapper(CRAPerson person)
        {
            Guard.ArgumentNotNull(person, "person");
            this.Person = person;
            this.IsPersonDead = GetIsPersonDead(person);
            this.IsDoubleCitizen = GetIsDoubleCitizen(person);
            this.IsNoCitizen = GetIsNoCitizen(person);
            this.IsGeorgianCitizen = GetIsGeorgianCitizen(person);
            this.IsZek = GetIsZek(person);
            this.IsZekInAdvance = GetIsZekInAdvance(person);
            this.IsPINCanceled = GetIsPINCanceled(person);
            this.IsPermanentResident = GetIsPermanentResident(this.IsNoCitizen ?? false,
this.IsGeorgianCitizen ?? false,
                                                              person);

            this.CRAStatusValueSetID = GetCRAPersonStatus(this);
            this.ImprisonmentInPrisonValueSetID = GetCRAPersonInPrison(this);
            this.ImprisonmentPreTrialDetentionValueSetID = GetCRAPersonPreTrialDetention(this);
            this.CitizenshipStatusValueSetID = GetCRAPersonCitizenship(this);
        }

        /// <summary>
        ///  6.      პირადი ნომრის გაუქმება გაყალბება
        ///  თუ ჩანაწერის გაუქმების მიზეზი (DocumentStatusName) მოიცავს „ყალბ“ ან  „პირადი ნომრის გაორება“ ან  „პირადი ნომრის გაუქმება“ ან  „პირადი ნომრის შეცვლა“ ან  „შეცდომა პირად ნომერში“ ან „ბათილად ცნობა“ („რეგისტრაციის ბათილად ცნობა“-ის გარდა) ან  „შეცდომა პირად ნომერში“
        /// </summary>
        /// <param name="person"></param>
        /// <returns></returns>
        private bool? GetIsPINCanceled(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;

            if (pers.DocumentStatusName.Contains("ყალბ")
                ||
                pers.DocumentStatusName.Contains("პირადი ნომრის გაორება")
                ||
                pers.DocumentStatusName.Contains("პირადი ნომრის გაუქმება")
                ||
                pers.DocumentStatusName.Contains("პირადი ნომრის შეცვლა")
                ||
                pers.DocumentStatusName.Contains("შეცდომა პირად ნომერში")
                ||
                pers.DocumentStatusName.Contains("ბათილად ცნობა")
                &&
                !pers.DocumentStatusName.Contains("რეგისტრაციის ბათილად ცნობა"))
                return true;

            return false;
        }

        /// <summary>
        ///  5.      პატიმრობა
        ///  PersonStatus=1 და  PersonCondition თავსდება (8,21).
        ///  8 - პატიმრობა; 21 - წინასწარი დაკავება
        /// </summary>
        /// <param name="person"></param>
        /// <returns></returns>
        private bool? GetIsZekInAdvance(CRAPerson person = null)
        {
```

```
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            if (pers.PersonStatus == 1 && pers.PersonCondition == 11)
                return true;
            return false;
        }

        /// <summary>
        ///  5.      პატიმრობა
        ///  PersonStatus=1 და  PersonCondition თავსდება (8,21).
        ///  8 - პატიმრობა; 21 - წინასწარი დაკავება
        /// </summary>
        /// <param name="person"></param>
        /// <returns></returns>
        private bool? GetIsZek(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;

            if (pers.PersonStatus == 1 && pers.PersonCondition == 4)
                return true;
            return false;
        }

        /// <summary>
        /// 4.       მოქალაქეობის არმქონე პირი
        /// დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება (42,3,40,2)
        /// და მოქალაქეობის ქვეყნის დასახელების მნიშვნელობში (Citizenship)  ფიქსირდება „არმქონე“
        /// და PersonStatus=1 და  PersonCondition არ თავსდება (2,3,6,7,9,10,14,44,50,51,52,53)
        /// და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს „მიიღო უცხო ქვეყნის მოქალაქეობა“, „ბინადრობის ნებართვის შეწყვეტა“, „მიიღო საქართველოს მოქალაქეობა“
        /// </summary>
        private bool? GetIsNoCitizen(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            var docTypes = new short?[] { 2, 3, 40, 42 };
            var notPersonConditions = new long?[] { 2, 3, 6, 7, 9, 10, 14, 44, 50, 51, 52, 53 };
            var notDocumentStatusName = new string[] { "მიიღო უცხო ქვეყნის მოქალაქეობა", "ბინადრობის ნებართვის შეწყვეტა", "მიიღო საქართველოს მოქალაქეობა" };

            if (pers.PersonStatus == 1
                &&
                pers.Citizenship.Contains("არმქონე")
                &&
                !notPersonConditions.Contains(pers.PersonCondition)
                &&
                docTypes.Contains(pers.DocumentTypeID)
                &&
                !notDocumentStatusName.Any(a => pers.DocumentStatusName != null && pers.DocumentStatusName.Contains(a)))
                return true;
            return false;
        }

        /// <summary>
        ///  3.      საქართველოს ორმაგი მოქალაქე
        ///  არის საქართველოს მოქალაქე და
        ///  PersonStatus=1 და  PersonCondition თავსდება (15,17,31,32) ან
        ///  ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა (DoubleCitizenship) არ არის „საქართველო“, „ცარიელი“, „არმქონე“
        ///  და ორმაგი მოქალაქეობის ქვეყნის დასახელების მნიშვნელობა ((DoubleCitizenship)) შეესაბამება რომელიმე ქვეყნის დასახელების მნიშვნელობას
```

```
        ///  (ამ ნაწილში შესაძლებელია გამოიყენოთ ორმაგი მოქალაქეობის ქვეყნის კოდი
DoubleCitizenshipCode (GEO, ENG  და ა.შ.)).
        /// </summary>
        /// <param name="person"></param>
        /// <returns></returns>
        private bool? GetIsDoubleCitizen(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            var notDocumentStatusName = new string[] { "საქართველო", "ცარიელი", "არმქონე" };
            var personConditions = new long?[] { 7, 9, 51, 52 };
            if (pers.PersonStatus == 1 && personConditions.Contains(pers.PersonCondition)
                ||
                !string.IsNullOrEmpty(pers.DoubleCitizenship)
                &&
                !notDocumentStatusName.Any(a => pers.DoubleCitizenship.Contains(a)))
                return true;
            return false;
        }

        /// <summary>
        ///  2.      საქართველოს მოქალაქე
        ///  PersonStatus=1 და  PersonCondition თავსდება (2,3,14,15,17,18,29,26,30,31,32,35)
        ///  ან PersonStatus=1 და  PersonCondition თავსდება (1,8,13,16,21,12,7)
        ///  და დოკუმენტის ტიპის იდენტიფიკატორი (DocumentTypeID) თავსდება
(22,7,8,11,26,12,27,21,39,1,14,49,29,16,20,19)
        ///  და ჩანაწერის გაუქმების მიზეზი (DocumentStatusStr) არ მოიცავს: „მიიღო უცხო ქვეყნის
მოქალაქეობა“ ან „საქართველოს მოქალაქეობიდან გასვლა“ ან „საქართველოს მოქალაქეობის შეწყვეტა“
        /// </summary>
        private bool? GetIsGeorgianCitizen(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            var personConditions = new long?[] { 2, 3, 6, 7, 9, 10, 14, 44, 50, 51, 52, 53 };

            var personConditionsOR = new long?[] { 1, 4, 5, 8, 11, 2, 1 };
            var docTypes = new short?[] { 22, 7, 8, 11, 26, 12, 27, 21, 39, 1, 14, 49, 29, 16, 20,
19 };
            var notDocumentStatusName = new string[] { "მიიღო უცხო ქვეყნის მოქალაქეობა",
"საქართველოს მოქალაქეობიდან გასვლა", "საქართველოს მოქალაქეობის შეწყვეტა" };

            if (personConditions.Contains(pers.PersonCondition) && pers.PersonStatus == 1
                ||
                (
                pers.PersonStatus == 1 && personConditionsOR.Contains(pers.PersonCondition)
                &&
                docTypes.Contains(pers.DocumentTypeID)
                &&
                !notDocumentStatusName.Any(a => pers.DocumentStatusName != null &&
pers.DocumentStatusName.Contains(a)))
                )
                return true;
            return false;
        }

        /// <summary>
        ///  1.      გარდაცვალების აღნიშვნა
        ///  PersonStatus=2 და PersonCondition=2
        /// </summary>
        private bool? GetIsPersonDead(CRAPerson person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            if (pers.PersonCondition == 2 && pers.PersonStatus == 2) return true;
            return false;
```

```
        }

        private bool? GetIsPermanentResident(bool isNotCitizen, bool isGeorgianCitizen, CRAPerson
person = null)
        {
            if (person == null && this.Person == null)
                return null;
            var pers = person ?? this.Person;
            if (isNotCitizen || isGeorgianCitizen || pers.PersonStatus != 1)
                return false;

            var result = false;
            //I. ვარიანტი
            var personConditions = new long?[] { 1, 4, 5, 8, 11, 12, 13 };
            var docTypes = new short?[] { 42, 3, 40, 2 };
            var documentStatusName = new string[] { "ბინადრობის ნებართვის შეწყვეტა", "„მიიღო
საქართველოს მოქალაქეობა" };
            if (personConditions.Contains(pers.PersonCondition)
                && docTypes.Contains(pers.DocumentTypeID)
                && !documentStatusName.Any(a => pers.DocumentStatusName.Contains(a))
                && (pers.DocumentTypeID == 42 || pers.DocumentTypeID == 3) ? pers.IdCardValidDate >
DateTime.Today : true)
                return true;

            //II. ვარიანტი - პირი გავიდა საქართველოს მოქალაქეობიდან და მასზე გაცემულია ბინადრობის
ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია:
            personConditions = new long?[] { 12, 13 };
            docTypes = new short?[] { 21, 39, 1, 16 };
            if (personConditions.Contains(pers.PersonCondition) &&
docTypes.Contains(pers.DocumentTypeID))
                return true;

            //III. ვარიანტი - არის მცირე რაოდენობა შემთხვევებისა, როდესაც პირს შეუწყდა ბინადრობის
ნებართვა  და მასზე გაცემულია ახალი ბინადრობის ნებართვა თუმცა ბინადრობის მოწმობა არ მიუღია.
            personConditions = new long?[] { 12, 13 };
            docTypes = new short?[] { 42, 3, 40, 2 };
            if (personConditions.Contains(pers.PersonCondition)
                 && docTypes.Contains(pers.DocumentTypeID)
                 && !documentStatusName.Any(a => pers.DocumentStatusName.Contains(a))
                 || (pers.DocumentTypeID == 42 || pers.DocumentTypeID == 3) && pers.IdCardValidDate
<= DateTime.Today)
                return true;

            return false;
        }


        public static int GetCRAPersonStatus(CRAPersonSIMSWrapper person)
        {
            if (person.IsPersonDead == true)
                return 427; //გარდაცვლილი
            if (person.IsPINCanceled == true)
                return 430; //გაუქმებული
            if (person.Person.PersonStatus == 1)
                return 426; //აქტიური
            return 431; //უცნობია
        }

        public static int GetCRAPersonInPrison(CRAPersonSIMSWrapper person)
        {
            if (person.IsZek == null || person.IsZek == false) return 4910; //არ არის
            return 4909; //არის
        }

        public static int GetCRAPersonPreTrialDetention(CRAPersonSIMSWrapper person)
        {
            if (person.IsZekInAdvance == null || person.IsZekInAdvance == false) return 4913; //არ
არის
```

```
            return 4912; //არის
        }

        public static int GetCRAPersonCitizenship(CRAPersonSIMSWrapper person)//short? statusID,
long? conditionID, string citizenshipID, string doubleCitizensshipID)
        {
            /*
             * 454 საქართველოს მოქალაქე
             * 455 მუდმივად მცხოვრები არამოქალაქე
             * 456 ორმაგი მოქალაქე
             * 4846 საქართველოში სტატუსის მქონე მოქალაქეობის არმქონე
             * 457 არ არის
             */
            //if (statusID == null)
            //    return 2024; //უცნობია
            //if ((statusID & 1) == 1 && (conditionID & 7) == 7) return 456; //ორმაგი მოქალაქე
            //if ((statusID & 1) == 1 && ((conditionID & 12) == 12 || (conditionID & 13) == 13))
return 455; //მუდმივად მცხოვრები არამოქალაქე
            //if ((statusID & 2) == 2 && (conditionID & 44) == 44) return 457; //არ არის
            //return 454; //საქართველოს მოქალაქე
            if (person.IsDoubleCitizen == true)
                return 456; //ორმაგი მოქალაქე

            if (person.IsNoCitizen == true)
                return 4846; //საქართველოში სტატუსის მქონე მოქალაქეობის არმქონე

            if (person.IsPermanentResident == true)
                return 455; //მუდმივად მცხოვრები არამოქალაქე

            if (person.IsGeorgianCitizen == true)
                return 454; //საქართველოს მოქალაქე

            return 457; //არ არის
        }

    }
```