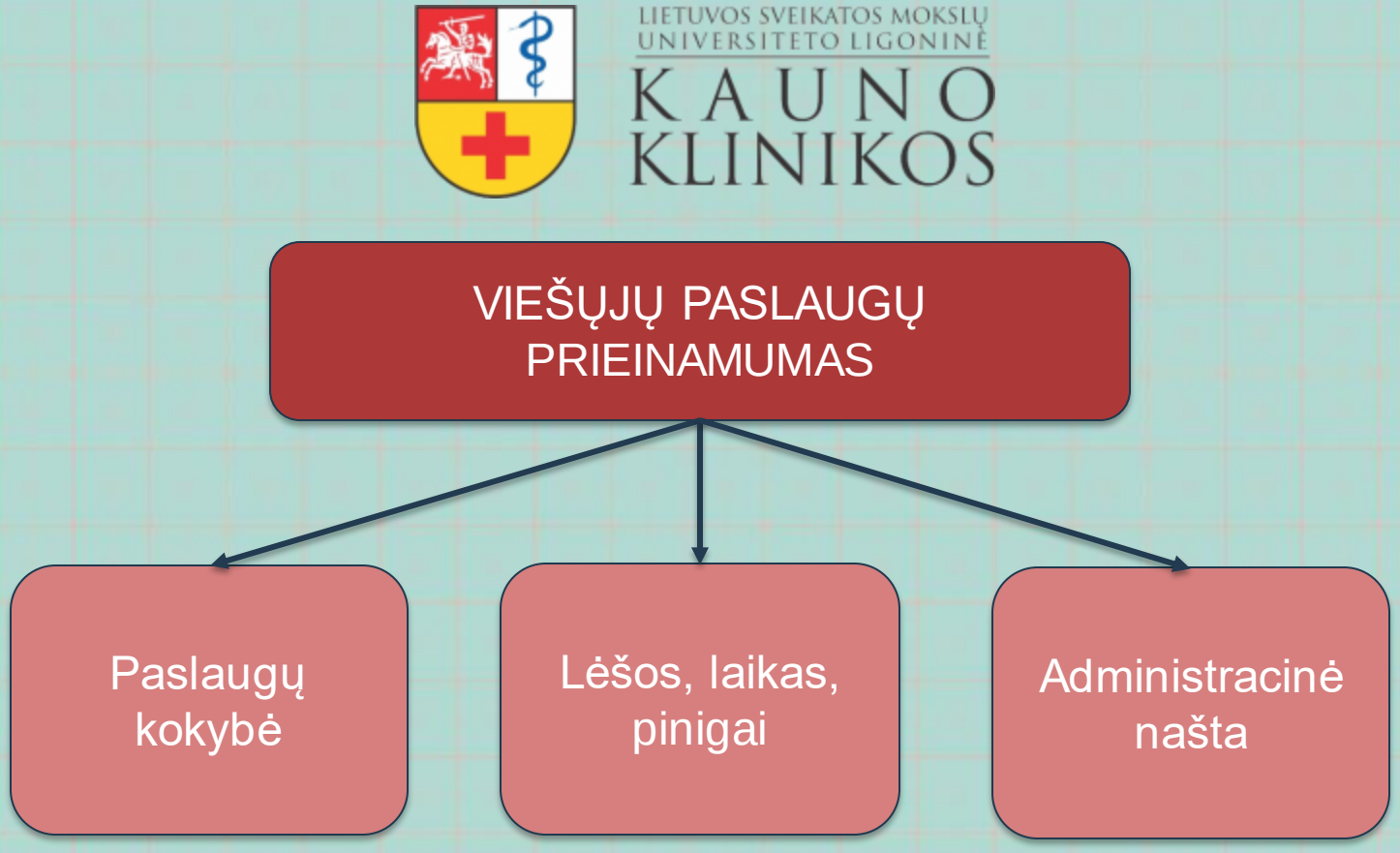
ktu
kauno
technologijos
universitetas
1922

# SVEIKATOS PRIEŽIŪROS PASLAUGŲ PERKĖLIMAS Į ELEKTRONINĘ ERDVĘ

AIVARAS ŠIMULIS, IFK-1
DOC. DALIUS MAKACKAS

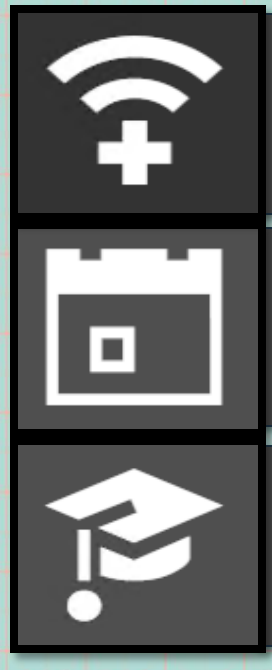
# NAGRINĖJIMA PROBLEMA

LIETUVOS SVEIKATOS MOKSLŲ UNIVERSITETO LIGONINĖ
KAUNO KLINIKOS



# DARBO TIKSLAS IR UŽDAVINIAI

**Tikslas** – sukurti sveikatos priežiūros paslaugų modulius:



- Nuotolinių stebėjimų
- Apsilankymų
- Švietimo

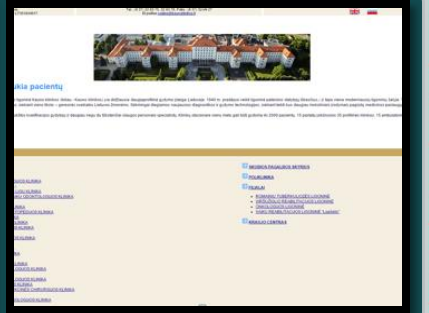
**Uždaviniai**:

- Išanalizuoti panašias IS
- Išanalizuoti technologijas ir įrankius
- Apibrėžti specifikaciją
- Suprojektuoti
- Realizuoti ir ištestuoti
- Paruošti dokumentaciją

# SISTEMŲ ANALIZĖ

| pasveik.lt | sergu.lt | santa.lt | kaunoklinikos.lt |
|---|---|---|---|



**VERTINIMO KRITERIJAI:**

- Apsilankymų planavimas;
- Savo sveikatos būklės stebėjimas;
- Galimybė sužinoti apie profilaktines programas;
- Galimybė padėti sau;
- Paslaugų kokybė;

# SISTEMŲ ANALIZĖS REZULTATAI

- Orientacija į individualius vartotojus; −
- Pasižymi suteikiamų paslaugų gausa; −
- Pagrindinė suteikiama paslauga nėra vien tik statinė informacija; −
- Suteikiama paslaugų kokybė yra priimtina; + −

- Bendradarbiavimas tarp paciento ir gydytojo; −
- Galimybė stebėti savo sveikatos būklę. −
- Gauti rekomendacijas ir patarimus; −

Įgyvendinamas dėl unikalumo;

- Tvarkyti apsilankymus; + −
- Siųsti priminimus ir kitus informacinius pranešimus; + −

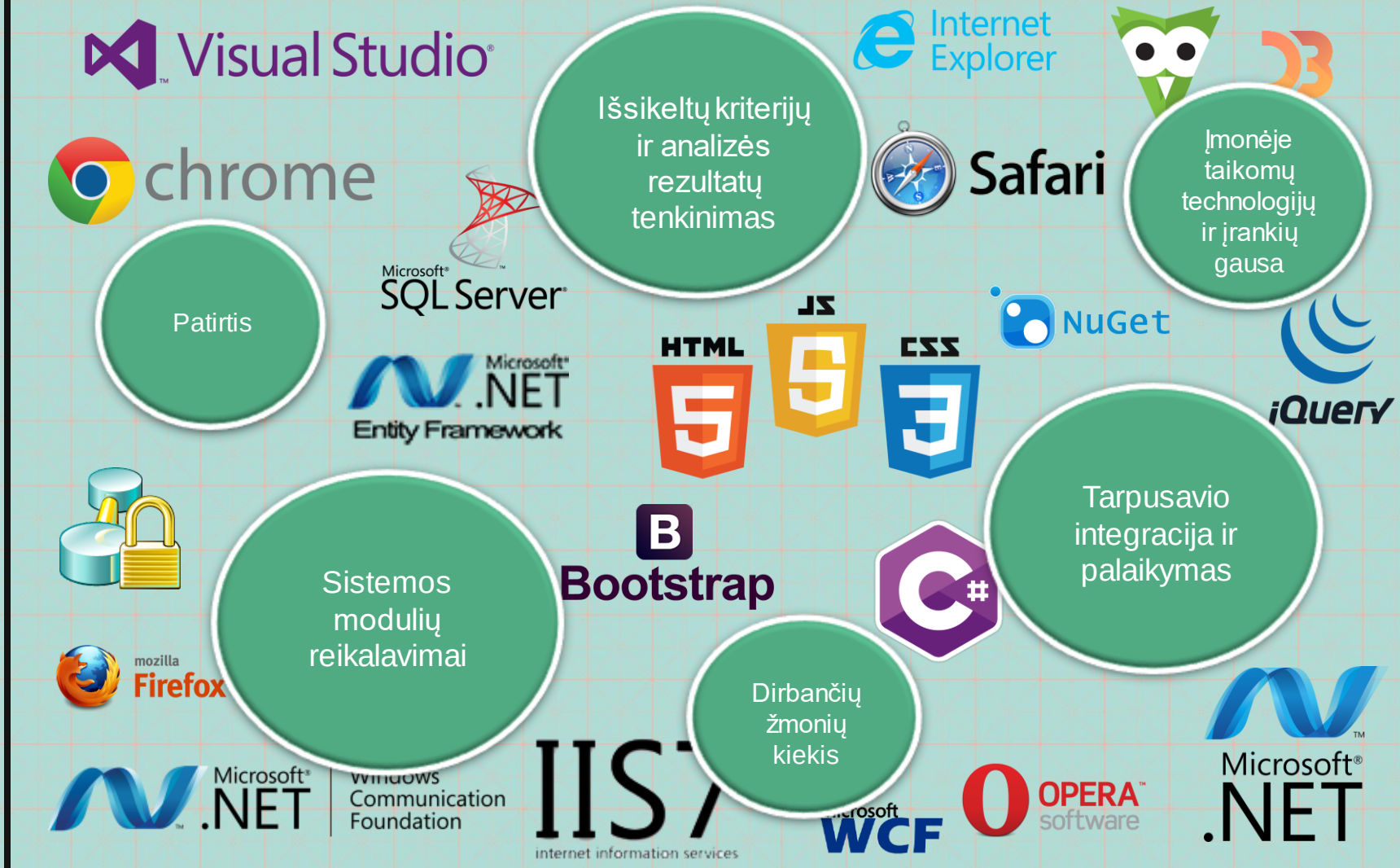
Įgyvendinamas dėl susietumo ir bendros integracijos sistemoje;

- Išmokti pažinti savo ligą ir ją atitinkamai įveikti; + −
- Atlikti mokomosios medžiagos testus ir klausimynus; −

Įgyvendinamas dėl unikalios orientacijos į atskirą vartotoją;

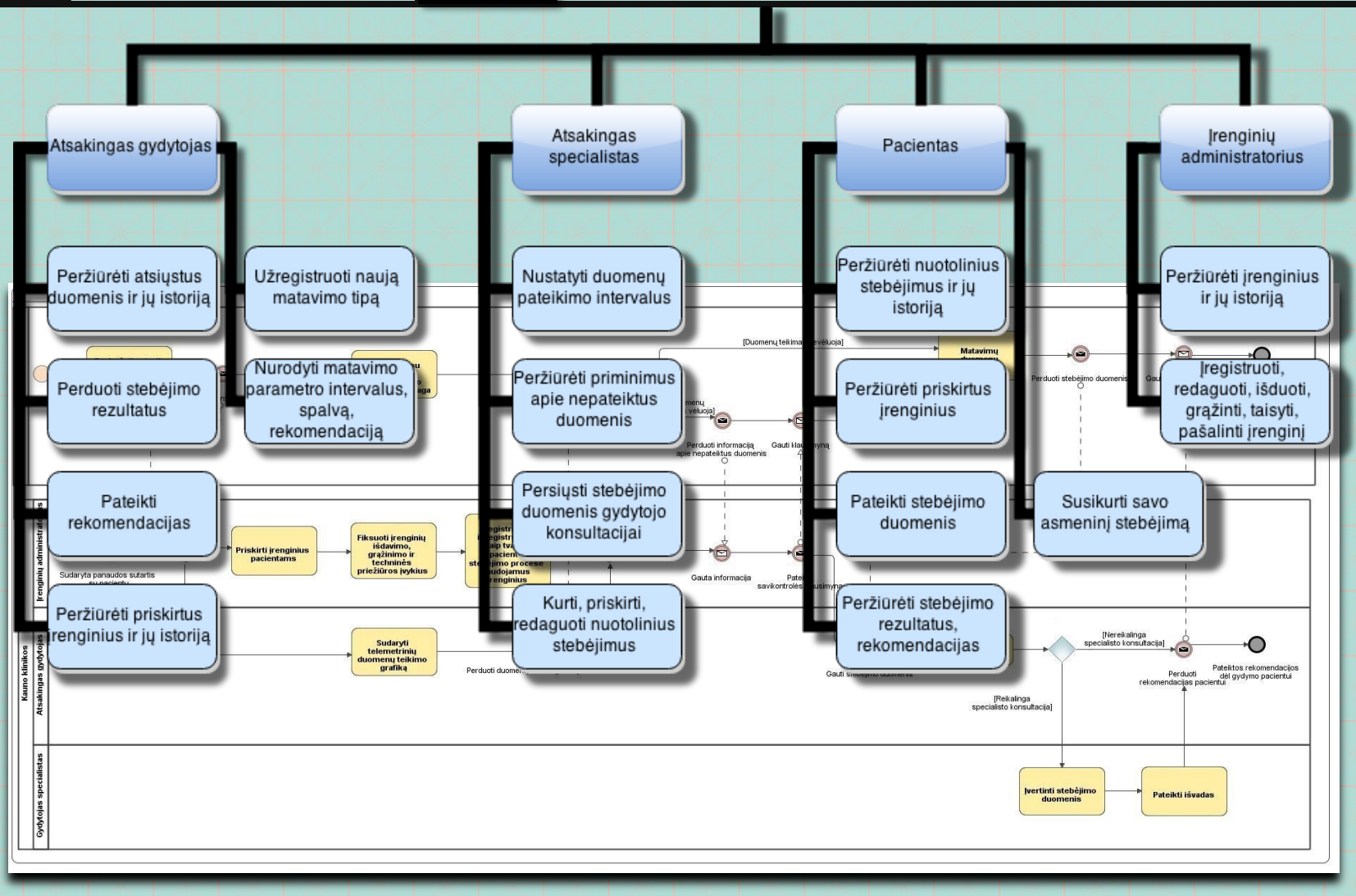
\+ Turi  − Neturi

# TECHNOLOGIJOS IR ĮRANKIAI

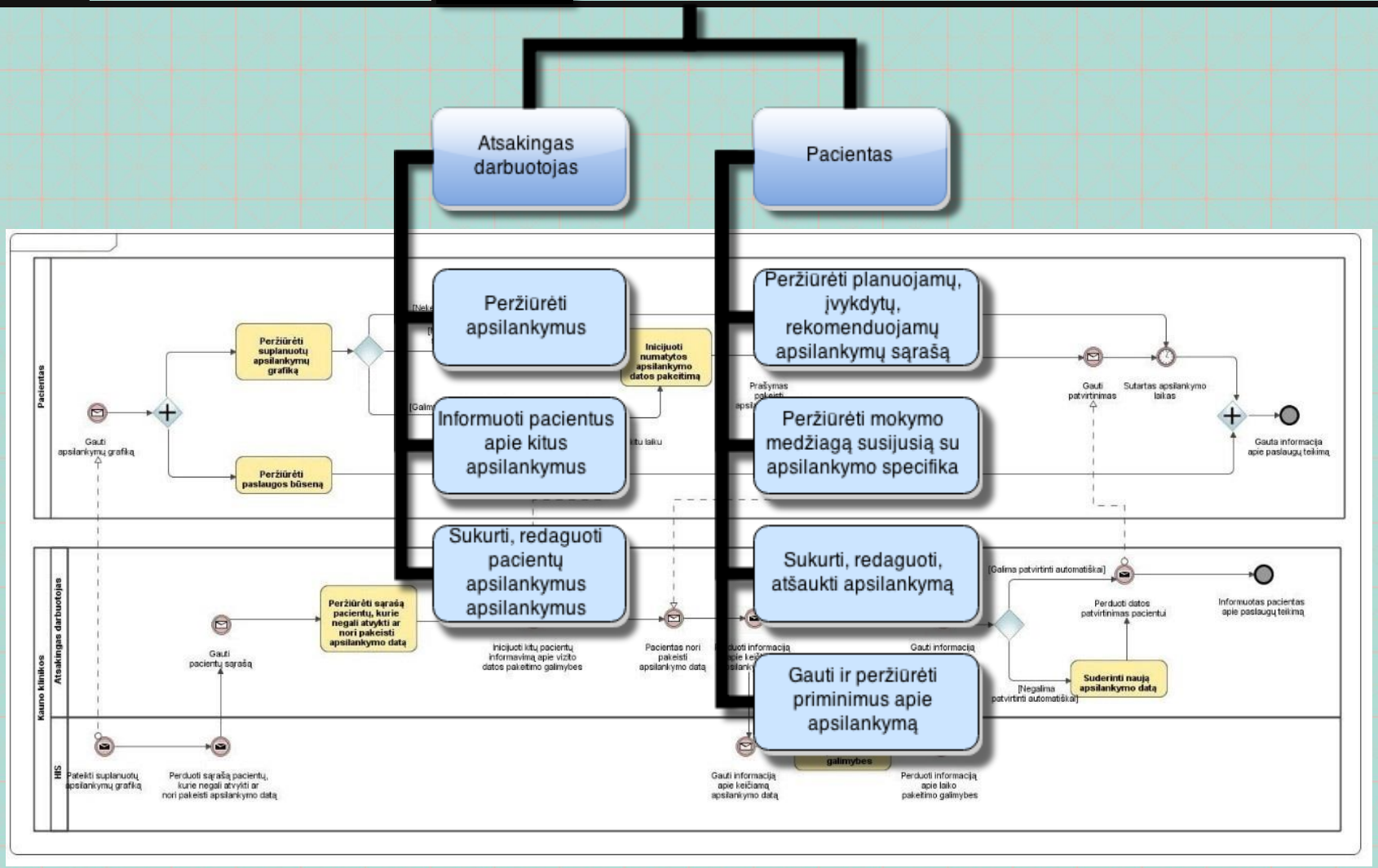


ktu kauno technologijos universitetas 1922

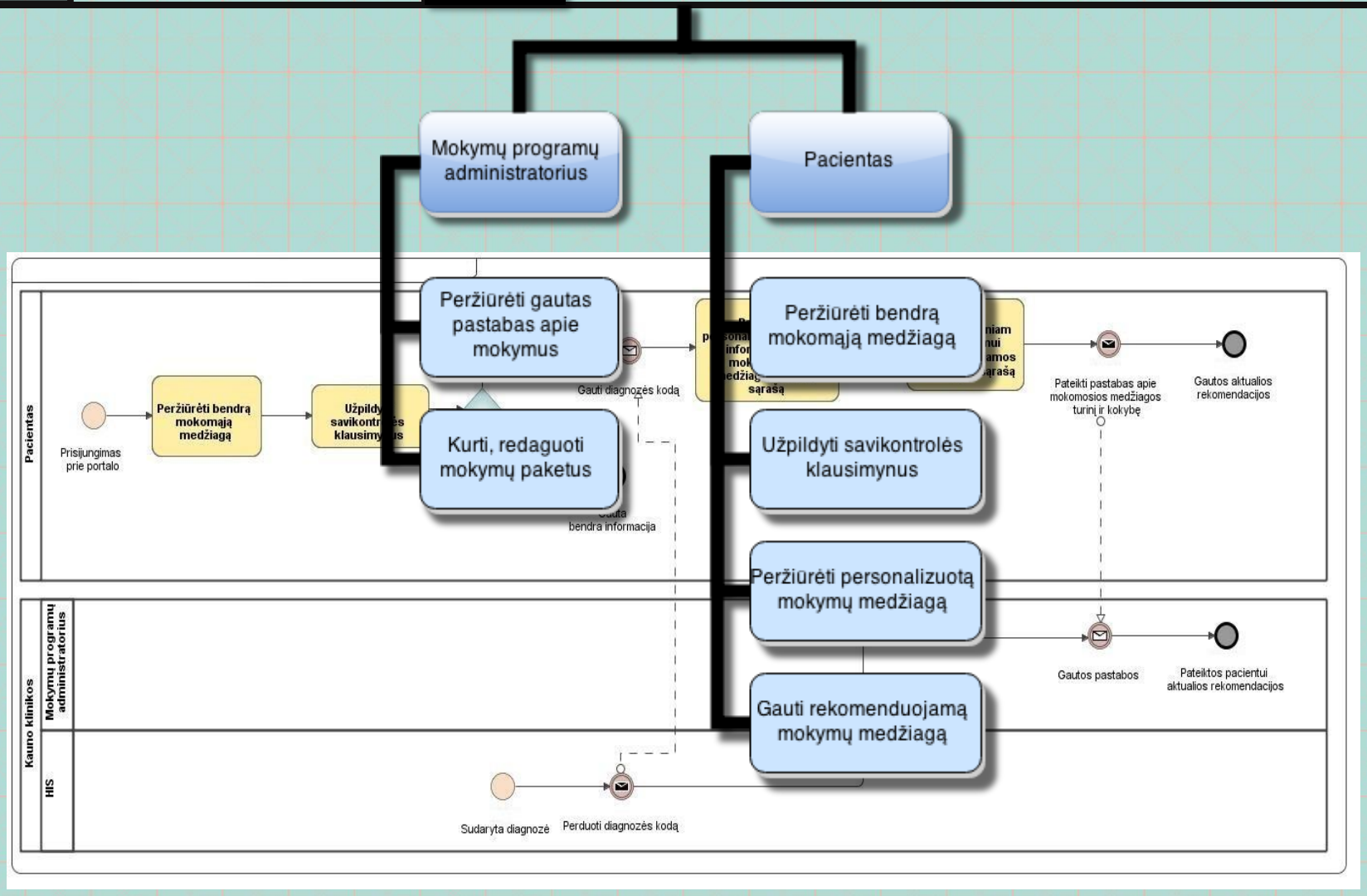
# NUOTOLINIŲ STEBĖJIMŲ MODULIS



# APSILANKYMŲ MODULIS



# ŠVIETIMO MODULIS



# LOGINĖ ARCHITEKTŪRA

**„MVC+VM+S“**

M – modelis;

V – vaizdas;

C – kontroleris;

VM – vaizdo modelis;

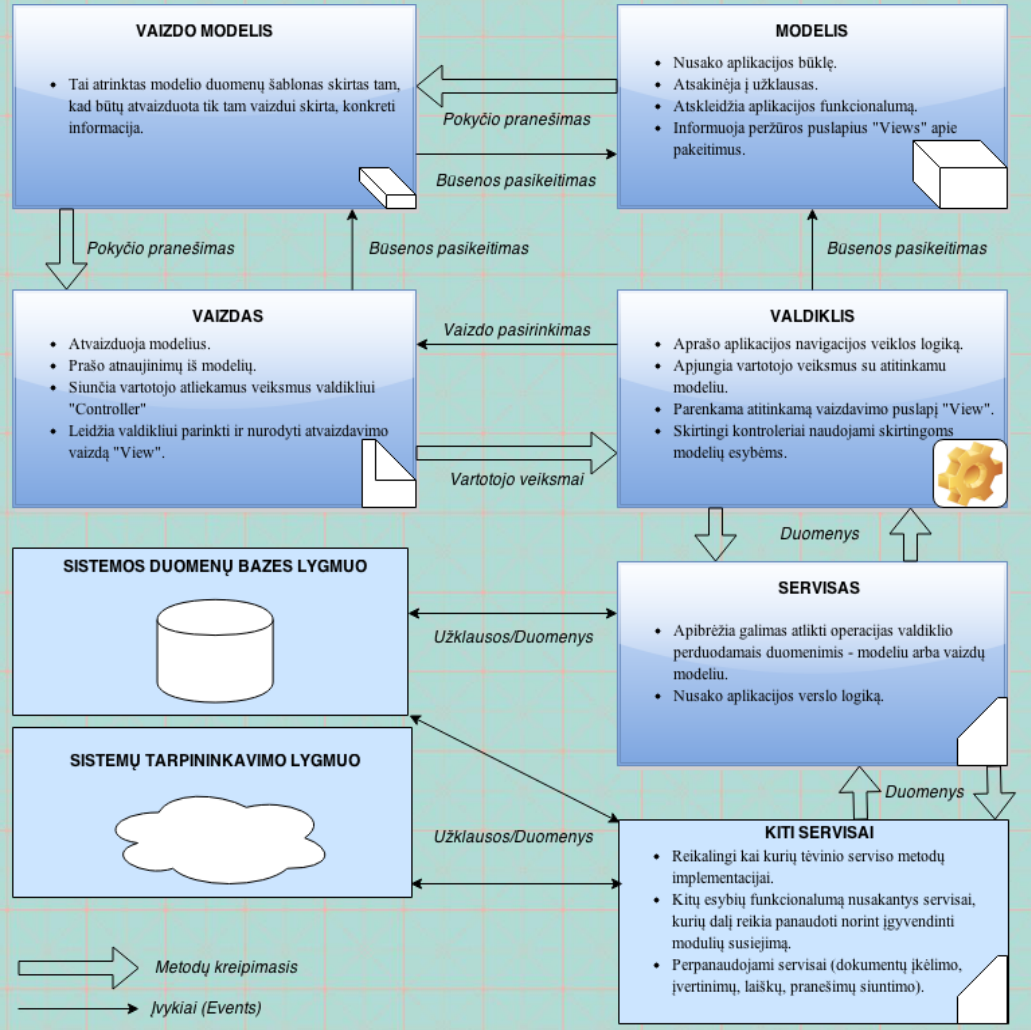
S – servisas;

**Būdinga**:

- „SoC“ principas.
- Perpanaudojamumas.

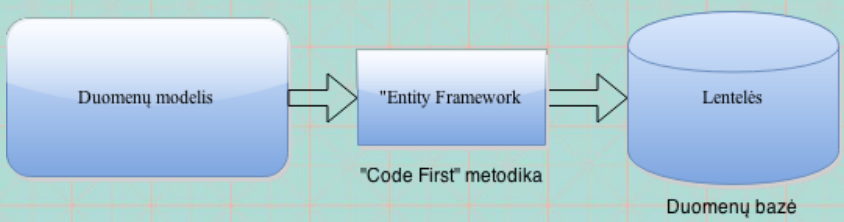
**Nauda**:

- Organizuoto, tvarkingesnio kodo rašymas.
- Lengviau palaikyti sistemą ir ją tobulinti.



# SĄVEIKA SU DBVS

## „ENTITY FRAMEWORK CODE FIRST“

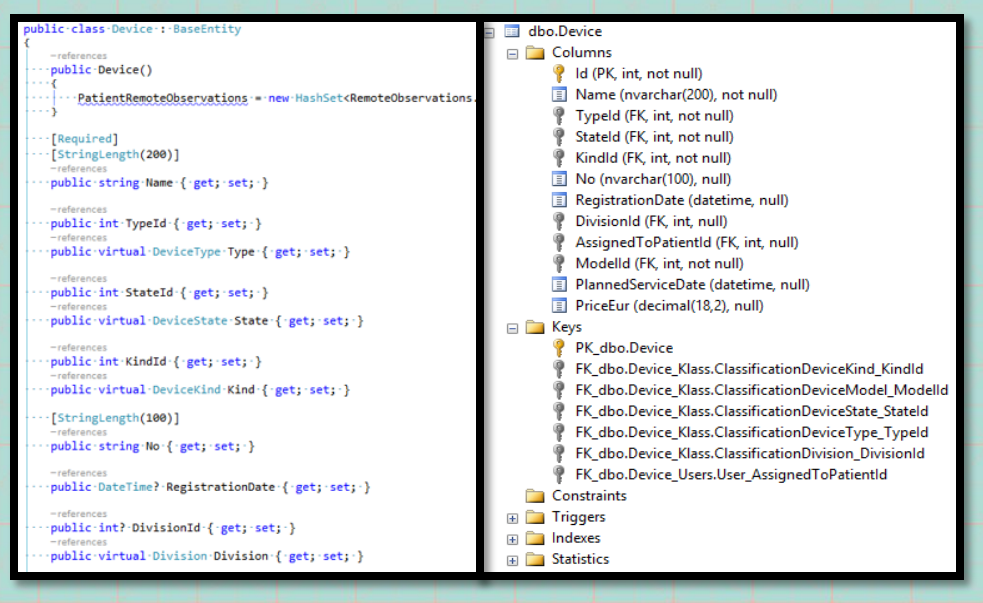



**Būdinga**:

- Naudojamas „DDD“ metodikoje.
- Kuriamos klasės, tuomet sugeneruojamos duomenų bazės lentelių atitikmenys.

**Nauda**:

- Koncentracija ties duomenų modelių kūrimu.
- Patogu konfigūruoti ir keisti.



```
public class Device : BaseEntity
{
    public Device()
    {
        PatientRemoteObservations = new HashSet<RemoteObservations>();
    }

    [Required]
    [StringLength(200)]
    public string Name { get; set; }

    public int TypeId { get; set; }
    public virtual DeviceType Type { get; set; }

    public int StateId { get; set; }
    public virtual DeviceState State { get; set; }

    public int KindId { get; set; }
    public virtual DeviceKind Kind { get; set; }

    [StringLength(100)]
    public string No { get; set; }

    public DateTime? RegistrationDate { get; set; }

    public int? DivisionId { get; set; }
    public virtual Division Division { get; set; }
```

```
dbo.Device
  Columns
    Id (PK, int, not null)
    Name (nvarchar(200), not null)
    TypeId (FK, int, not null)
    StateId (FK, int, not null)
    KindId (FK, int, not null)
    No (nvarchar(100), null)
    RegistrationDate (datetime, null)
    DivisionId (FK, int, null)
    AssignedToPatientId (FK, int, null)
    ModelId (FK, int, not null)
    PlannedServiceDate (datetime, null)
    PriceEur (decimal(18,2), null)
  Keys
    PK_dbo.Device
    FK_dbo.Device_Klass.ClassificationDeviceKind_KindId
    FK_dbo.Device_Klass.ClassificationDeviceModel_ModelId
    FK_dbo.Device_Klass.ClassificationDeviceState_StateId
    FK_dbo.Device_Klass.ClassificationDeviceType_TypeId
    FK_dbo.Device_Klass.ClassificationDivision_DivisionId
    FK_dbo.Device_Users.User_AssignedToPatientId
  Constraints
  Triggers
  Indexes
  Statistics
```

# SISTEMŲ DUOMENŲ KEITIMOSI LYGMENS REALIZACIJA

## „WCF“ SERVISAI

**Būdinga**:

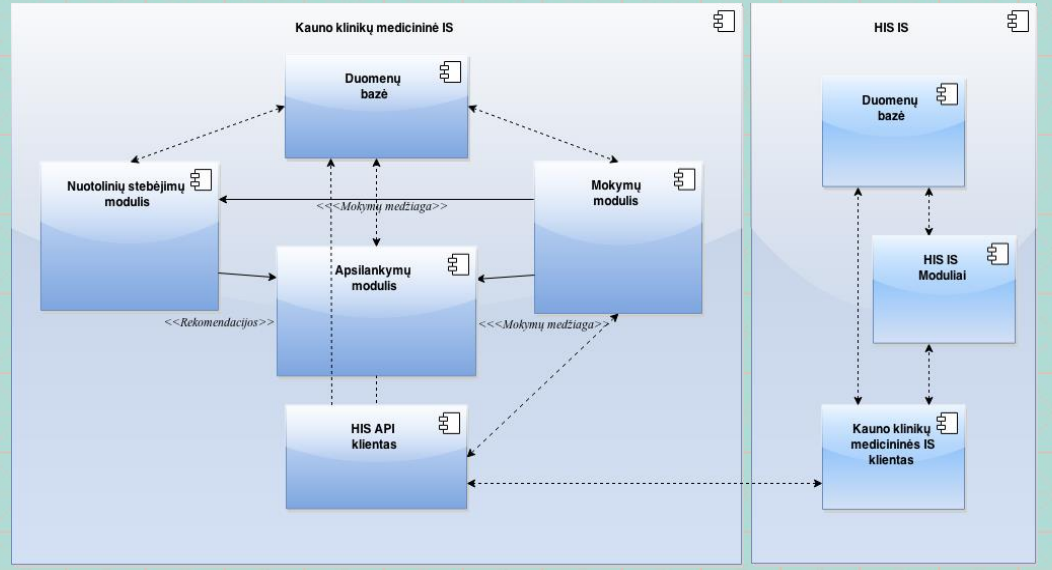
- Siųsti pranešimus iš vienos sistemos paslaugos taško į kitą.

**Nauda**:

- Gauti ar apsikeisti reikalingais duomenimis tarp atskirų sistemų.

**Panaudojimo pavyzdys:**

- Laisvų talonų gavimas registruojantis apsilankymams




```
[DataContract(Namespace = Namespaces.Contracts)]
public class TicketInfoRequest
{
    [DataMember(Name = "SESSION_ID", Order = 10)]
    public string SessionId { get; set; }

    [DataMember(Name = "TICKET_ID", Order = 20)]
    public int TicketId { get; set; }
}
```

```
[DataContract(Namespace = Namespaces.Contracts)]
public class TicketInfoResponse
{
    [DataMember(Name = "TICKET_ID", Order = 10)]
    public int Id { get; set; }

    [DataMember(Name = "VISIT_TYPE_ID", Order = 20)]
    public int VisitTypeId { get; set; }

    [DataMember(Name = "VISIT_PURPOSE_ID", Order = 30)]
    public int VisitPurposeId { get; set; }

    [DataMember(Name = "START_TIME", Order = 40)]
    public DateTime StartTime { get; set; }

    [DataMember(Name = "CABINET", Order = 50)]
    public string Cabinet { get; set; }
```

# PRANEŠIMŲ SIUNTIMAS

Sukurtas, pakeistas, atšauktas, galimas ankstesnis, artėjantis apsilankymas.

Pateikta rekomendacija, sukurta konsultacija, įkelti/pateikti duomenys.

„SmtpClient“

„MailMessage“

```
private void SendCancelDoctorVisitNotification(Visit visit, string cancelComment)
{
    var container = new
    {
        Date = VisitFormatter.FormatDateWithoutSeconds(visit.Date),
        Specialty = visit.Specialty.Name,
        Doctor = visit.Doctor.FirstName + " " + visit.Doctor.LastName,
        CancelReason = cancelComment
    };

    messageSender.Send(Templates.VisitCancelDoctorVisit, container, visit.UserId,
        deliveryType: MessageDeliveryType.Email | MessageDeliveryType.System);
}
```

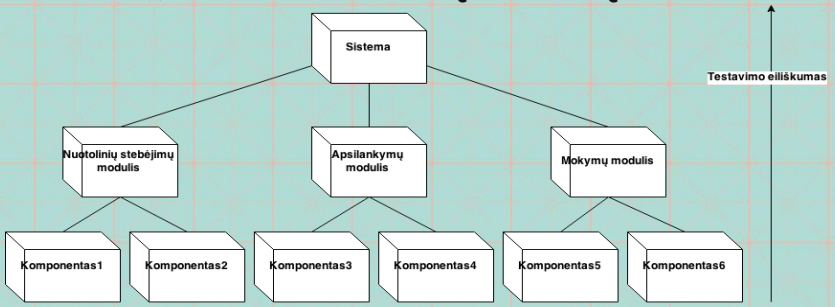
```
public ServiceResult Send(Templates type, object container, int userId, int? organizationId = null,
    MessageDeliveryType? deliveryType = null)
{
    if (!deliveryType.HasValue)
    {
        deliveryType = GetDefaultDeliveryTypes(userId);
    }

    var languageCode = templatingService.GetLanguageCode(null, userId);
    var msg = templatingService.Format(templatingService.GetTemplate(type.Value, languageCode), container);

    return Send(msg, userId, deliveryType.Value, organizationId);
}
```

```
private ServiceResult SendMail(SendEmailModel model)
{
    var result = new ServiceResult();
    var client = new SmtpClient();

    var email = new MailMessage { Subject = model.Subject, Body = model.Body, IsBodyHtml = true };
    email.To.Add(new MailAddress(model.Recipient.Address, model.Recipient.FullName));

    if (model.Async)
    {
        try
        {
            Task.Factory.StartNew(() =>
            {
                try
                {
                    client.Send(email);
                }
                catch (Exception ex)
                {
                    Logger.Error("SendMail async failed with exception", ex);
                    var dataAccess = ServiceLocator.Resolve<ISiteDataAccess>();
                    SaveFailedMessage(dataAccess, model.Subject, model.Body, model.Recipient.Address,
                        model.Recipient.FullName, MessageDeliveryType.Email, ExceptionHelper.GetInnerExceptionMessages(ex));
                }
            });
        }
        catch (Exception e)
        {
            Logger.Error("SendMailAsync exception", e);
            result.AddError("Mail sending failed", ExceptionHelper.GetInnerExceptionMessages(e));
        }
    }
    else
    {
        try
        {
            client.Send(email);
        }
        catch (Exception ex)
        {
            Logger.Error("SendMail exception", ex);
            result.AddError("Mail sending failed", ExceptionHelper.GetInnerExceptionMessages(ex));
        }
    }

    return result;
}
```

# TESTAVIMAS

## „JUODOSIOS DĖŽĖS“


## „IŠ APAČIOS Į VIRŠŲ“





```
private User CreatePatient(ISiteDataAccess dataAccess)
{
    var entity = new User
    {
        FullName = "Vardenis Pavardenis",
        FirstName = "Vardenis",
        LastName = "Pavardenis",
        PersonCode = null,
        CreateOn = DateTime.Today,
        FailedPasswordAnswerAttemptWindowStart = DateTime.Today,
        FailedPasswordAttemptWindowStart = DateTime.Today,
        LastActivityDate = DateTime.Today,
        LastLockoutDate = DateTime.Today,
        LastLoginDate = DateTime.Today,
        LastPasswordChangedDate = DateTime.Today,
        Email = DefaultPatientEmail,
        UserName = "varpa001",
        LanguageId = 1
    };

    dataAccess.Insert(entity);

    return entity;
}
```

```
public void Test_GetTimes_ReturnsValidResponse()
{
    using (ShimsContext.Create())
    {
        // Arrange
        var now = DateTime.Now;
        Epika.Core.Mvc.Security.Fakes.ShimSiteWorkContext.AllInstances.CurrentLanguageCodeGet = ctx => "lt-LT";

        Visit visit;
        using (var dataAccess = new SiteDataAccess())
        {
            DoVisitCleanup(dataAccess);

            var patient = CreatePatient(dataAccess);
            var doctor = CreateDoctor(dataAccess, ValidDoctorHisKey); // Pass valid doctor HisKey
            var specialty = CreateSpecialty(dataAccess, ValidSpecialtyHisKey); // Pass valid doctor specialty HisKey
            dataAccess.Save();

            visit = CreateVisit(dataAccess, visitStateRegisteredId, now.AddDays(7), patient, doctor, specialty, true);
            dataAccess.Save();

            CreateUserMailHistory(dataAccess, visit.Id, now.AddDays(-7));
            dataAccess.Save();
        }

        // Act
        var service = ServiceLocator.Resolve<VisitEarlierTimeNotificationSendService>();
        var response = service.GetTimeHandler(visit.DoctorId.GetValueOrDefault(), visit.SpecialtyId.GetValueOrDefault(),
            now, visit.Date);

        // Assert
        Assert.IsNotNull(response.LastTicketDate, "Nebuvo gautas vėliausias laisvas laikas.");
        Assert.IsNotNull(response.List, "Gautų laisvų laikų sąrašas tuščias.");
        Assert.IsTrue(response.List.Count() != 0, "Gautų laisvų laikų sąrašo skaičius yra lygus 0");
    }
}
```

# DARBO REZULTATAI IR IŠVADOS

- Į e-erdvę perkeltos nuotolinių stebėjimų, apsilankymų, švietimo paslaugos;
- Naujos, geresnės ir patogesnės galimybės pasinaudoti paslaugomis;
- Sudarytos galimybės duomenų mainams tarp Kauno klinikų bei kitų institucijų informacinių sistemų;


**Ateityje**:

- Naujos e-paslaugos.
- Sistemos susiejimas su elektroniniais stebėjimo prietaisais.

ktu
1922
kauno
technologijos
universitetas

# AČIŪ UŽ DĖMESĮ

SVEIKATOS PRIEŽIŪROS PASLAUGŲ PERKĖLIMAS Į ELEKTRONINĘ ERDVĘ
AIVARAS ŠIMULIS, IFK-1 GR.
DOC. DALIUS MAKACKAS
MOB: +370 624 71012 / AIVARAS.SIMULIS@KTU.EDU
HTTPS://PORTALAS.KAUNOKLINIKOS.LT