TARABICA#
IT CONFERENCE
BELGRADE

PARTNERI KONFERENCIJE

Microsoft

Singidunum University

SPONZORI

ComTrade

acer

MDS

ewe comp

OBLAK TEHNOLOGIJE

MERA Software Services, Inc.

Merit SOLUTIONS

Hosted by EUnet eunethosting.com Hyper-V Cloud

JET BRAINS

Coca-Cola Hellenic Passion for Excellence

PRIJATELJI KONFERENCIJE

MSCommunity.ba

VEEAM IT JUST WORKS!™

CityExpert

Arhitektura i implementacija CQRS šablona sa Microsoft.NET-om

Skorić Milan<milan@smartwave.rs>

Cilj prezentacije

- CQRS - "***Command Query Responsibility Segregation***" nije samo za velika i kompleksna *„enterprise"* rešenja već je upotrebljiv i prikladan u svakom rešenju, pogotovo za „cloud".

O meni

- Prvi profesionalni kod napisan uz pomoć „beta“ verzije .Net Framework-a početkom 2001. godine
- Pionir mobilnih i elektronskih servisa za bankarstvo u regionu
- Jedan od osnivača

Smartwave

Agenda

- O tehnologijama
- Moderna arhitektura poslovnih aplikacija
- Šta je to CQRS šablon?
- Primer implementacije CQRS-a u Microsoft .Net

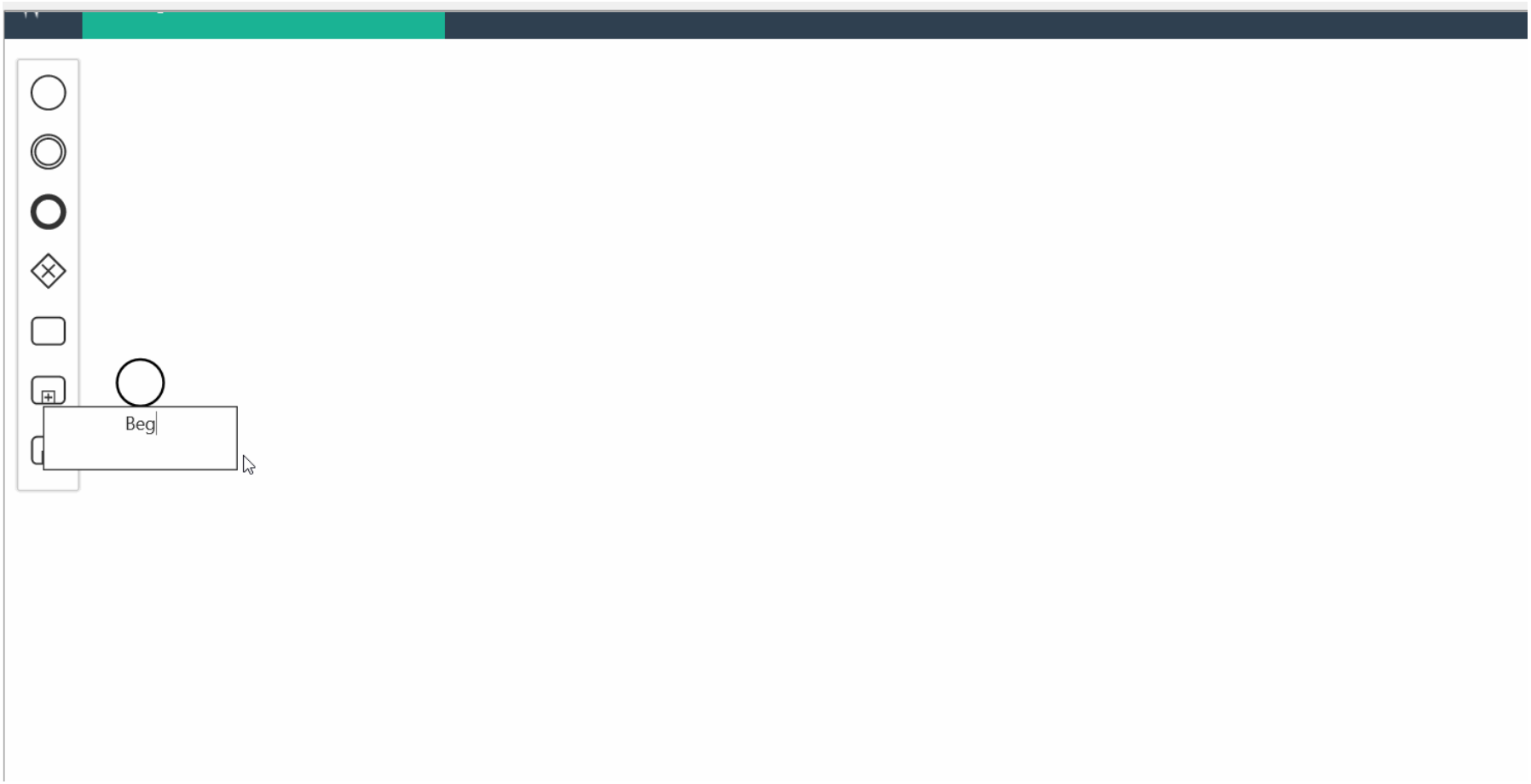
- Slobodno pitajte

Agenda

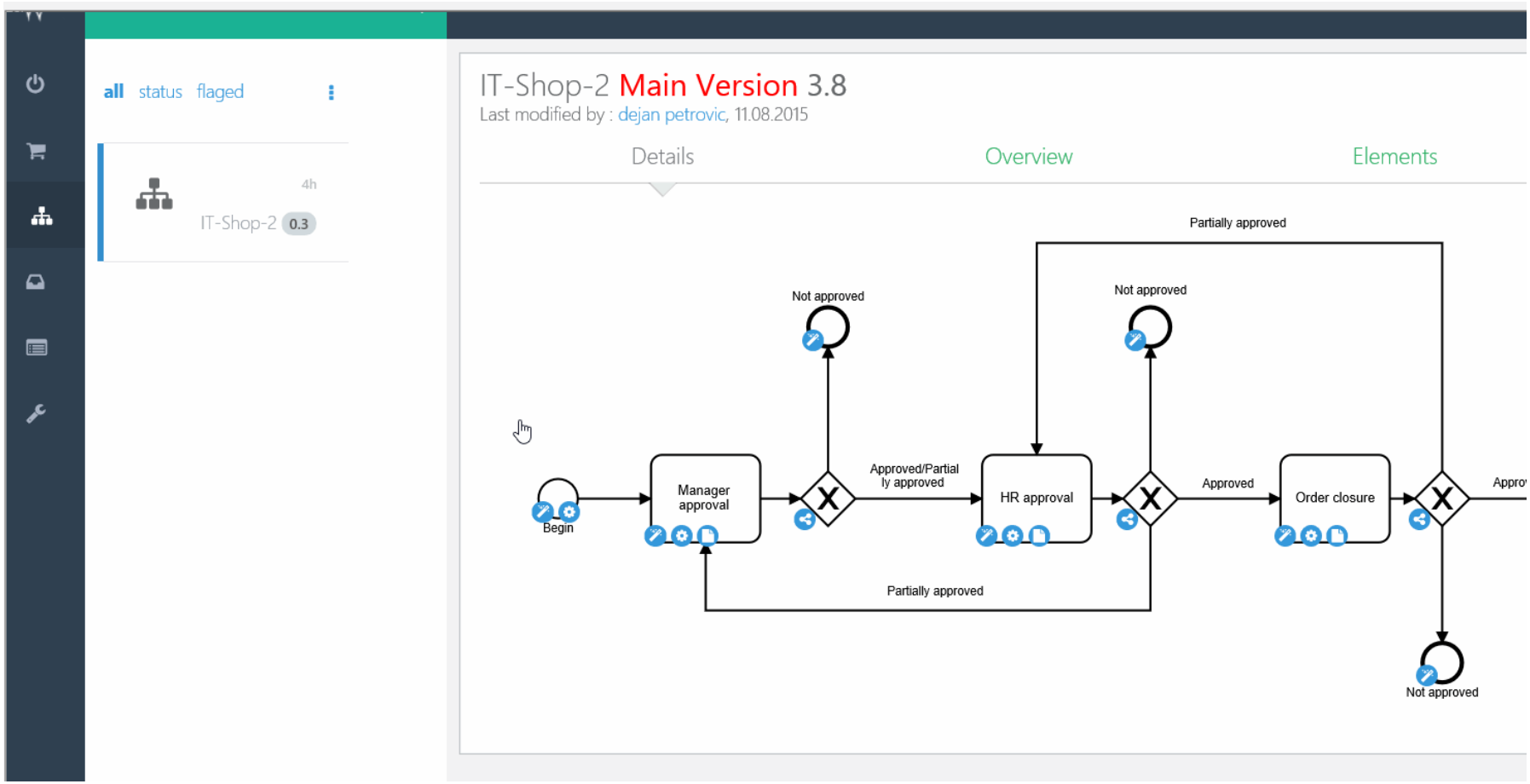
- **Molim vas da popunite anketu pre izlaska.**
- Red drugi, zaokružite broj sale (005) i ocenu, hvala!

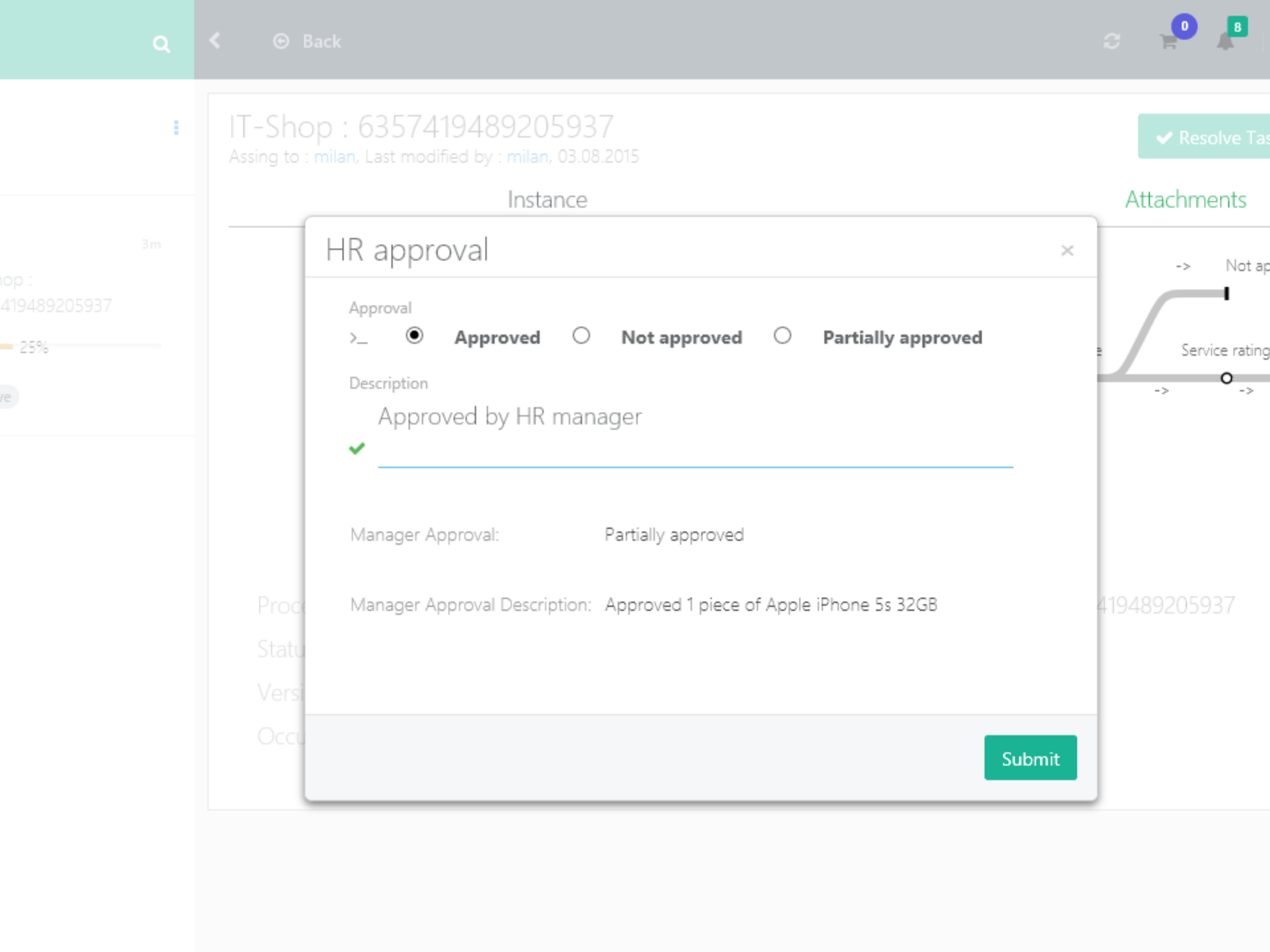
Pojedinačne ocene								Predavač(i)					Predavanje				
slot	sala							odlično				loše	odlično				loše
I	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1
II	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1
III	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1
IV	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1
V	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1
VI	001	004	005	015	016	018	046	5	4	3	2	1	5	4	3	2	1

Digitalizacija



Metapodaci


all status flaged
4h
IT-Shop-2 0.3
IT-Shop-2 Main Version 3.8
Last modified by : dejan petrovic, 11.08.2015
Details
Overview
Elements
Partially approved
Not approved
Not approved
Begin
Manager approval
Approved/Partial ly approved
HR approval
Approved
Order closure
Partially approved
Not approved


Back
0
8
IT-Shop : 6357419489205937
Assing to : milan, Last modified by : milan, 03.08.2015
Resolve Tas
Instance
Attachments
HR approval
Approval
Approved
Not approved
Partially approved
Description
Approved by HR manager
Manager Approval: Partially approved
Manager Approval Description: Approved 1 piece of Apple iPhone 5s 32GB
Submit
Not ap
Service rating

Ciljevi za Smartwave

- Dostupan
- Brz odziv
- Laka integracija
- Agilnost
- Jedostavno za postavljanje
- Telemetrija

- Ako je moguće i jeftino

Fokus

ASP.NET MVC

XAMARIN

ASP.NET Web API

SignalR, EF Code Frist

Visual Studio & .NET

U oblacima sa BizSpark-om

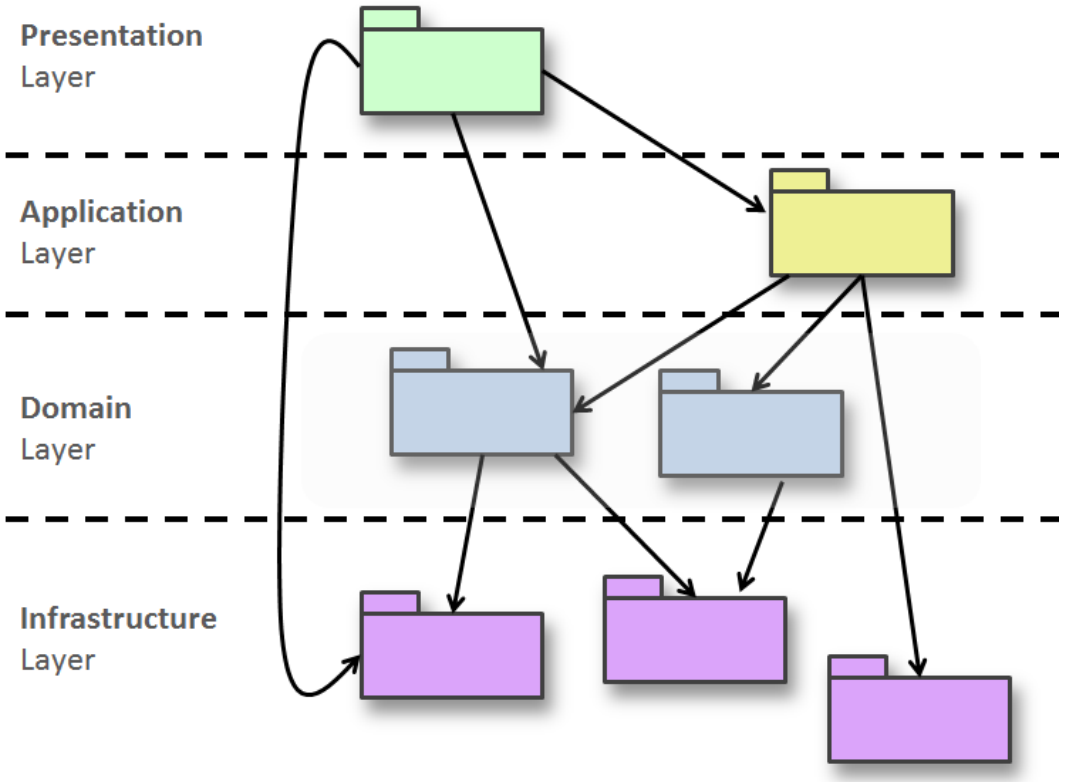
- Jednostavno se postavlja uz Visual Studio
- Nema „gvožđa“ za održavanje
- Skalabilnost dostupna na klik
- Multi-tenant

https://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-tools-multi-tenant-row-level-security/

A arhitektura

Kreirana za laku zamenu umesto česte ponovne upotrebe

Tradicija



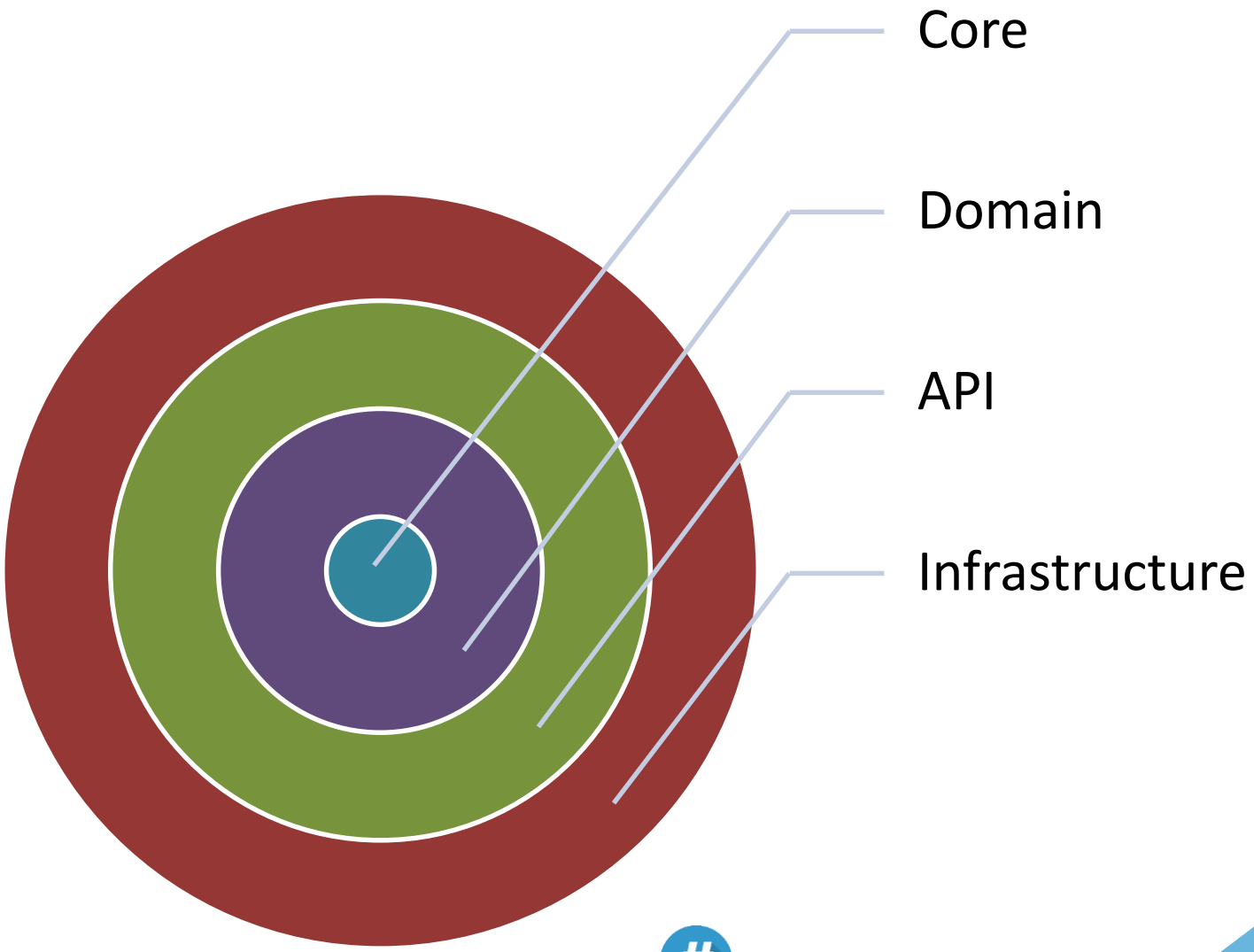
PREDNOSTI

- Jednostavno se pravi
- Puna podrška menadžmenta
- Već utabane staze

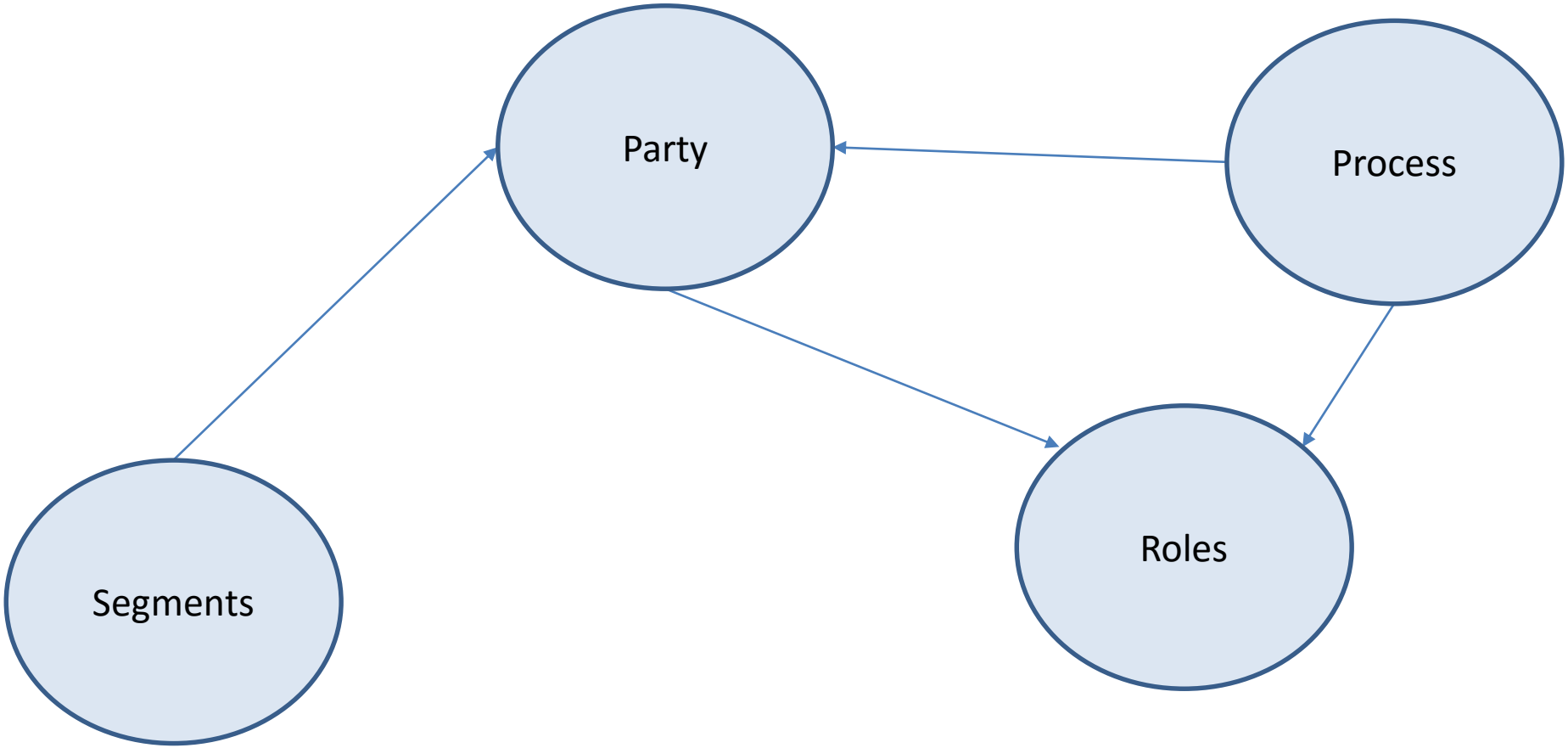
MANE

- Limitirana skalabilnost
- Teško se održava a često i sporo
- Prekomerna konverzija poruka koji se šetaju među slojevima

#Onokad baza nije centar



Domenski model - DDD



DDD iz prakse sa EF

Izbegavajte javne konstruktore bez parametara

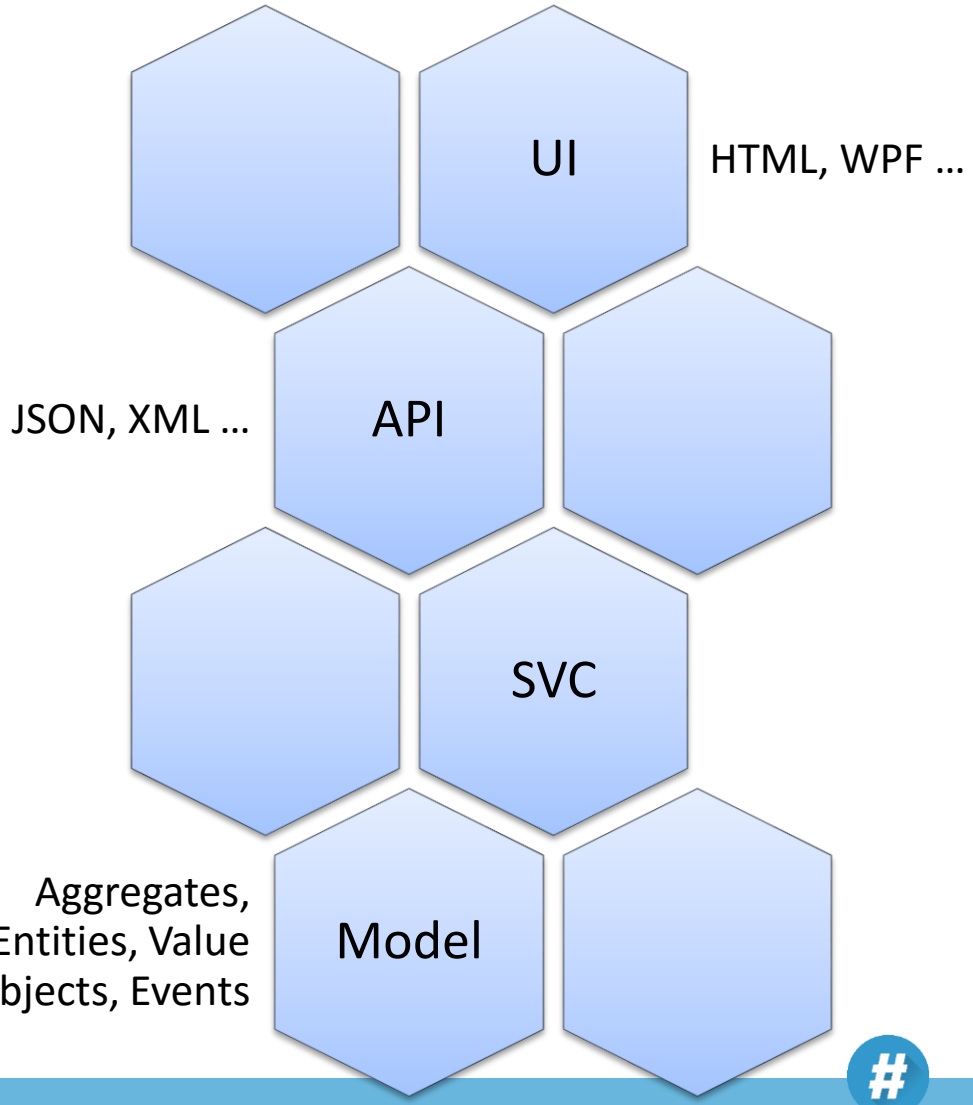
```
//EF requires a parameter-less constructor, but it can be private
private Comment()
{

}

//By requiring the rest of the application to only call this constructor,
//we can ensure we have a valid initialisation of an instance
public Comment(string correlationId, string content)
{
    this.SetCommentContent(correlationId, content);

    CommentCreatedEvent change = new CommentCreatedEvent(this);

    this.ApplyChange(change);
}
```

Koristite privatne članove a ponašanje držite uz objekte

```
//Use Private setters - properties can't be set directly
[DataMember]
public string Content { get; private set; }

//Method must be called, wich can be validated and easily tested
public void SetCommentContent(string correlationId, string content)
{
    if (string.IsNullOrWhiteSpace(content)) {
        throw new ArgumentNullException("Content parameter must be provided!");
    }
    this.Content = content;
    this.CorrelationId = correlationId;
```

http://thedatafarm.com/data-access/entity-framework-private-constructors-and-private-setters/

Microservice



- Svaki „*mikro*“ deo sadrži sve što je neophodno da bude potpuno samostalan

CQ(R)S iz perspektive koda

- CQS

```
public class CustomerService
{
   // Commands
   void MakeCustomerPreferred (CustomerId)
   void ChangeCustomerLocale(CustomerId, NewLocale)
   void CreateCustomer(Customer)
   void EditCustomerDetails(CustomerDetails)

   // Queries
   Customer GetCustomer(CustomerId)
   CustomerSet GetCustomersWithName(Name)
   CustomerSet GetPreferredCustomers()
}
```

From: https://gist.github.com/1964094

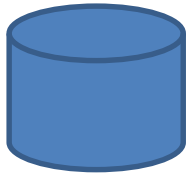
- CQRS (najprostiji)

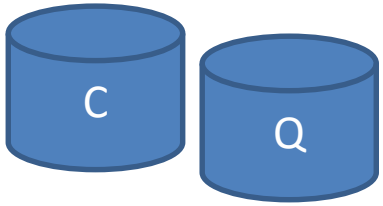
```
public class CustomerWriteService
{
   // Commands
   void MakeCustomerPreferred (CustomerId)
   void ChangeCustomerLocale(CustomerId, NewLocale)
   void CreateCustomer(CreateCustomer)
   void EditCustomerDetails(CustomerDetails)
}

public class CustomerReadService
{
   // Queries
   Customer GetCustomer(CustomerId)
   CustomerSet GetCustomersWithName(Name)
   CustomerSet GetPreferredCustomers()
}
```

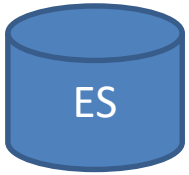
From: https://gist.github.com/1964094

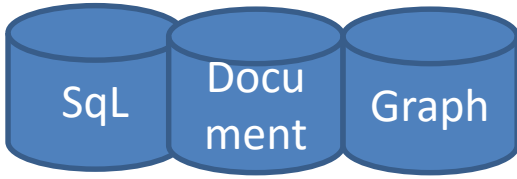
Tipovi implementacija

- Regularna implementacija – vraća novac
  - Transakcija po akciji
  - Upiti vraćaju DTO
- Standardna implementacija
  - Domenski model je fokusiran na ponašanje koje menja stanje
  - Tabela po upitu
- "Event Sourcing"
  - Čuvaju se samo događaji na osnovi kojih se gradi stanje domena
  - Automatski rešen istorijat i audit podataka

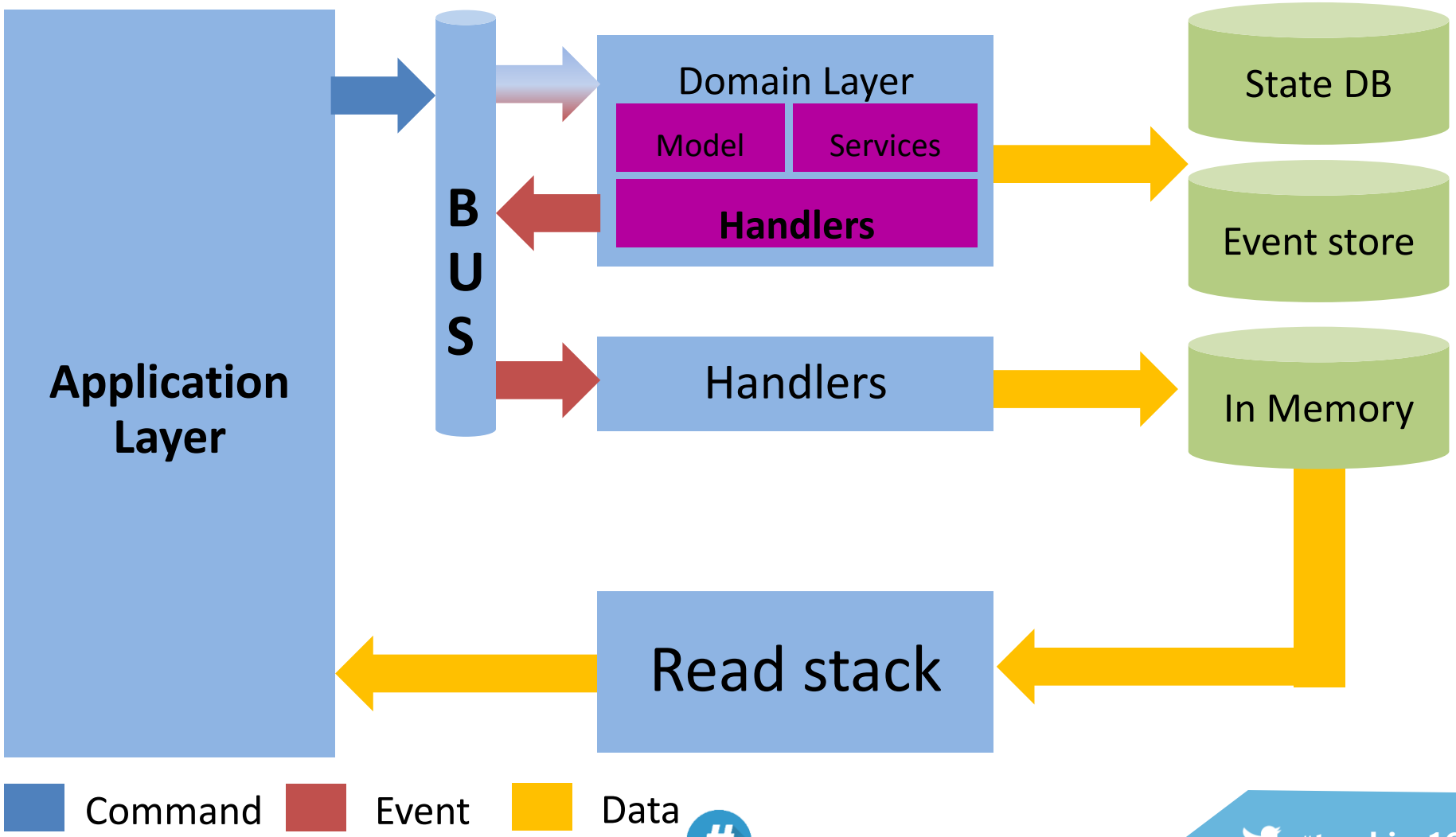




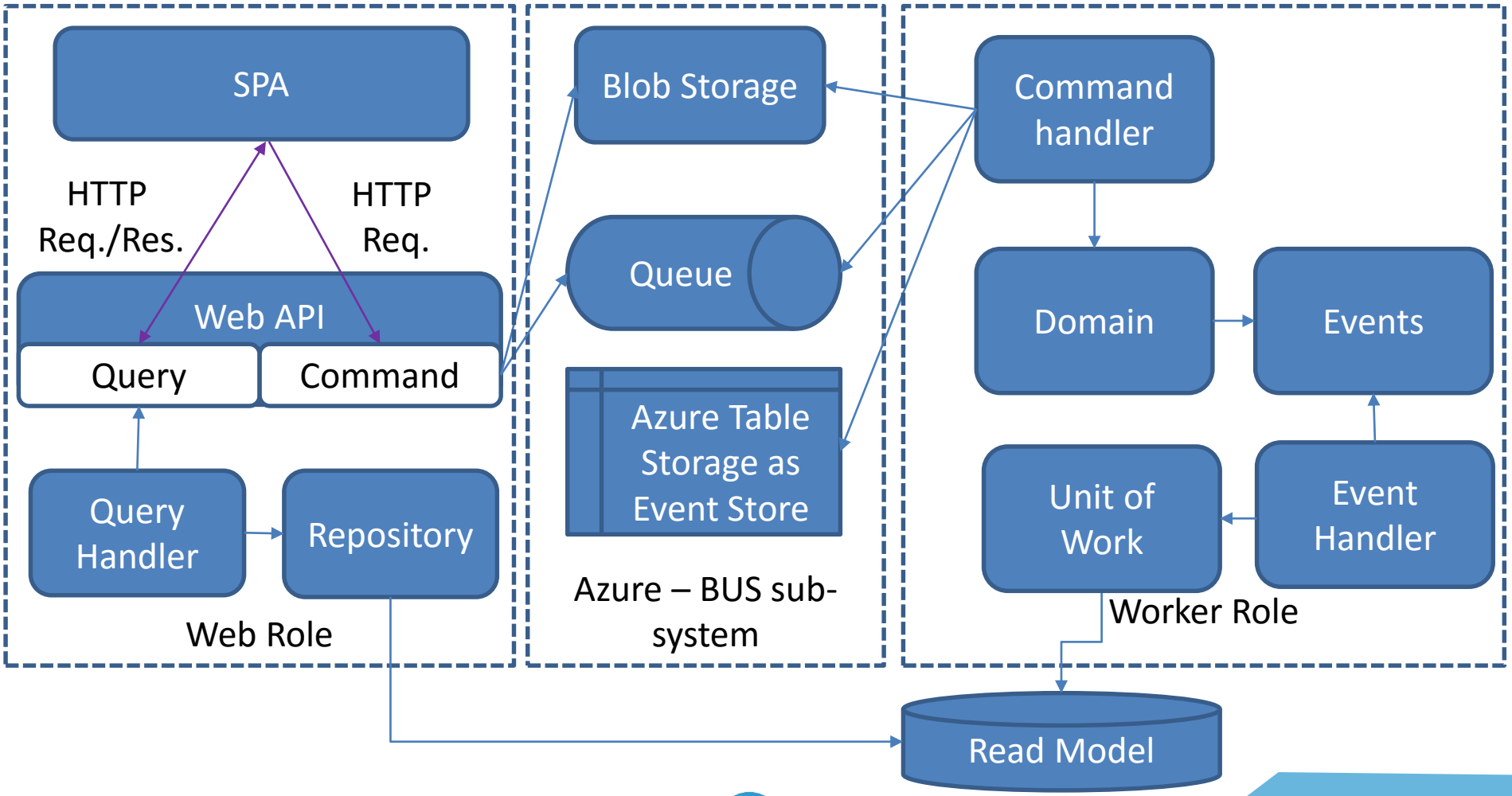




Standardna implementacija



Azure implementacija



Demo

WebAPI

```
[Route("query/{name:alpha}")]
[HttpPost]
public HttpResponseMessage Query(string name, [FromBody] dynamic dataTransferObject,
    [FromUri] QueryOptions options)
{
    IQueryMessage query = Mapper.Map<IQueryMessage>(name, ToDynamic(dataTransferObject));

    Result r = _queryDispatcher.Dispatch<IQueryMessage>(query, GetCurrentContext(options));

    return ToQueryResponse(r, options);
}

[Route("command/{name:alpha}")]
[HttpPost]
public HttpResponseMessage Command(string name, [FromBody] dynamic dataTransferObject,
    [FromUri] CommandOptions options)
{
    ICommandMessage cmd = Mapper.Map<ICommandMessage>(name, ToDynamic(dataTransferObject));

    Result r = _bus.Send(cmd, GetCurrentContext(options));

    return ToResponse(r, options);
}
```

Komanda

```
//  A message sent to tell the system to do something
[Serializable]
public class CreateComment : CommandMessage
{

    public string Content { get; set; }

}
```

```
[Serializable]
public abstract class CommandMessage : ICommandMessage
{
    public CommandMessage()
    {
        this.TimeStamp = TimeProvider.Now();
        this.CorrelationId = IdentityGenerator.GenerateUniqueTickIdentifer();
    }

    public DateTime TimeStamp { get; protected set; }

    public string CorrelationId { get; protected set;}
}
```

Obrada komande

```
public class CreateCommentCommandHandler : CommandHandler<CreateComment>
{
    public CreateCommentCommandHandler()
        : this(ServiceLocator.Current.TryGet<IUnitOfWork>(Extensions.GetUnitOfWorkName<Comment>()))...

    public CreateCommentCommandHandler(IUnitOfWork unitOfWork)
        : base(unitOfWork)...

    protected override void Handle(CreateComment command, InvocationContext context)
    {
        Comment item = new Comment(command.CorrelationId, command.Content);

        var output = this.UnitOfWork.Save<Comment>(item, context);

        this.Result = output.ToResult();
    }
}
```

Provera ispravnosti

```
public class CanAddCommentValidationHandler : IValidationHandler<CreateComment>
{
    public IEnumerable<ValidationResult> Validate(CreateComment command)
    {
        List<ValidationResult> output = new List<ValidationResult>();

        if (command == null || string.IsNullOrWhiteSpace(command.CorrelationId))
            output.Add(new ValidationResult(
                2210, "Content", "Content of message correlation Id is null", string.Empty));

        if (command == null || string.IsNullOrWhiteSpace(command.Content))
            output.Add(new ValidationResult(
                2220, "Content", "Content of comment is null or empty", command.CorrelationId));

        return output;
    }
}
```

Command Bus

```
public Result Send(ICommandMessage command, InvocationContext context)
{
    Result result = command.StopwatchWraper((x) =>
    {
        Result output = new Result(command);

        IEnumerable<ValidationResult> vResults = new List<ValidationResult>();

        output.Success = this.IsValid(command, out vResults);//validate the command before submitting

        output.ValidationResults = vResults;

        if (output.Success)
        {
            CommandOptions option = context.Options as CommandOptions;

            //hook the right command handler with command message and execute the command operation
            if (option != null && option.IsAsync)
            {
                var task = Task.Factory.StartNew(() => this.Dispach(command, context));
            }
            else
            {
                output = this.Dispach(command, context);
            }
        }

        return output;
    });

    return result;
}
```

Upit

```
public class FindAllCommentsQueryHandler : QueryHandler<FindAllComments>
{
    public FindAllCommentsQueryHandler()
        : this(ServiceLocator.Current.TryGet<IRepository>(Extensions.GetRepositoryName<FindAllComments>()))...

    public FindAllCommentsQueryHandler(IRepository repository)
        : base(repository)...

    protected override Result Retrieve(FindAllComments query, InvocationContext ctx)
    {
        Result output = new Result();

        var q = this.Repository.BuildQuery<Comment>(ctx.Options as QueryOptions);

        var id = query.Key.ToLong(0);

        if (id != 0)
            q = q.Where(i => i.Id == id);

        var result = this.Repository.ExecuteList<Comment>(q);

        output.Success = (result != null && result.Count() > 0);

        if (output.Success)
        {
            output.RecordsAffected = result.Count();
            output.Data = result;
        }

        return output;
    }
}
```

Read-Only EF Context

```
public class CommentRepository : Repository, ICommentRepository
{
    public CommentRepository()
        :base(new CommentsContext())
    {
        this._Context = this.dataContext as CommentsContext;
    }

    private readonly CommentsContext _Context;

    public IQueryable<Comments.Domain.Comment> Comments
    {
        get { return this._Context.Comments; }
    }
}
```

Domenski događaji

```
[Serializable]
public class CommentCreatedEvent : DomainEvent<Comment>
{
    public CommentCreatedEvent(Comment data)
    {
        this.AggregateRootID = data.Id;
        this.EntityVersion = data.Version;
        this.Data = data;
        this.CorrelationId = data.CorrelationId;
    }

}
```

Događaji iz domenskih objekata

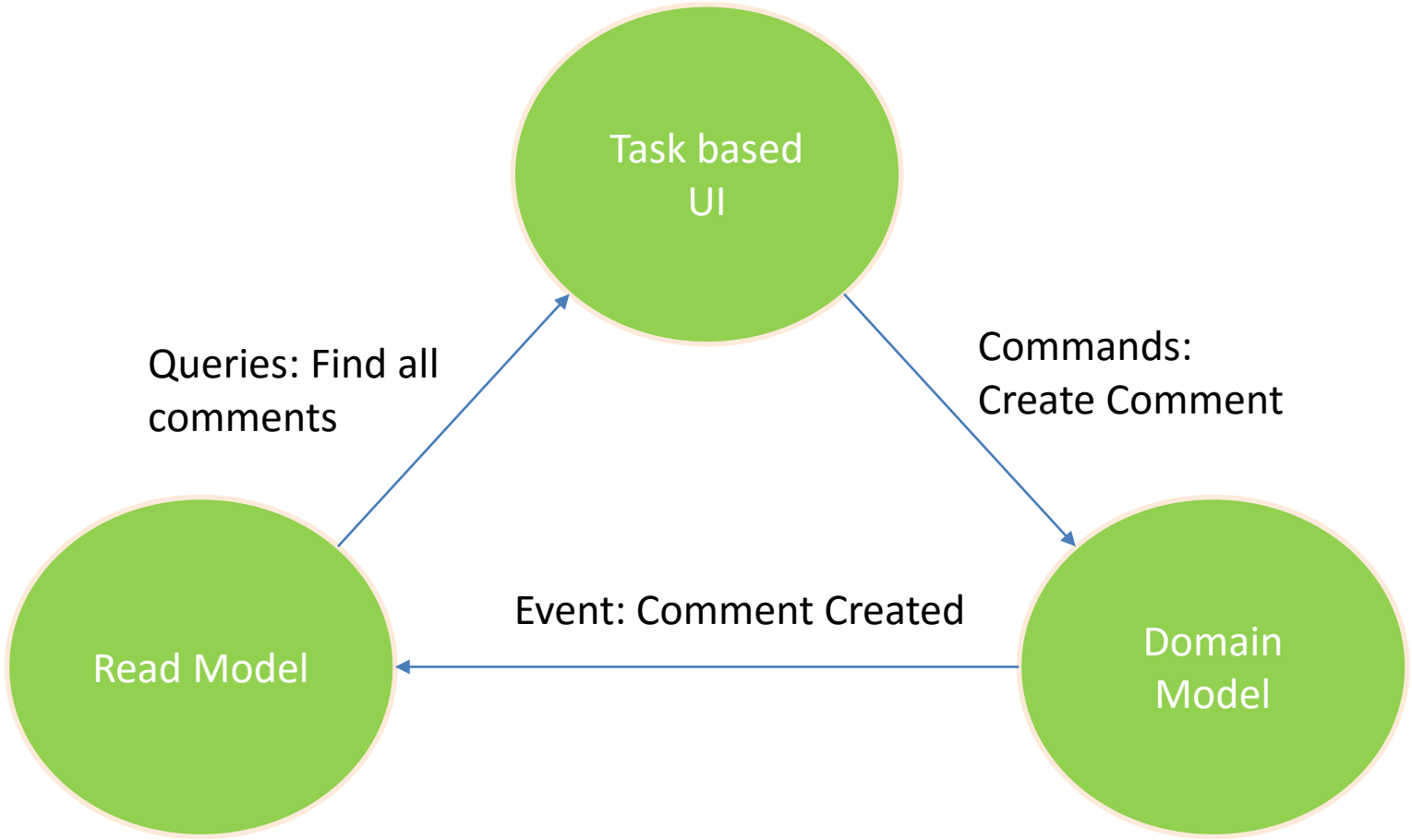
```
//EF requires a parameter-less constructor, but it can be private
private Comment()
{

}

//By requiring the rest of the application to only call this constructor,
//we can ensure we have a valid initialisation of an instance
public Comment(string correlationId, string content)
{
    this.SetCommentContent(correlationId, content);

    CommentCreatedEvent change = new CommentCreatedEvent(this);

    this.ApplyChange(change);
}
```

Obrada događaja

```
public class CommentCreatedEventHandler : DomainEventHandler<CommentCreatedEvent>
{
    public CommentCreatedEventHandler()
        : this(ServiceLocator.Current.TryGet<IUnitOfWork>(Extensions.GetReadRepositoryName<Comment>()))...

    public CommentCreatedEventHandler(IUnitOfWork unitOfWork)
        : base(unitOfWork)...

    protected override void Handle(CommentCreatedEvent change, InvocationContext context)
    {
        Comment item = change.Data;

        this.UnitOfWork.Set<Comment>(item, context);

        this.UnitOfWork.Commit();
    }
}
```

Demo

Kružna arhitektura



Prednosti i mane

PREDNOSTI

- Komande su glagoli i podržavaju uvek samo jedan scenario
- Query Store – može da bude NoSQL
- Proširivost
- Performantnost
- Lako se testira
- Lako se integriše

MANE

- Teško je održavati sinhronizaciju između više skladišta podataka
- Puno novih koncepata
- Puno novih odluka koje treba doneti
- Limitirana praksa

Resursi

- https://msdn.microsoft.com/en-us/library/jj554200.aspx
  By Patterns & Practices
- http://www.codemag.com/article/1411071
  By Leonardo Esposito
- http://martinfowler.com/bliki/CQRS.html
  By Martin Fowler
- http://codebetter.com/gregyoung/2012/09/09/cqrs-is-not-an-architecture-2/
  By Greg Young

– http://www.udidahan.com
  By Udi Dahan

Pitanja i odgovori

- Slobodno se prijavite, Smartwave trenutno nudi besplatno „beta“ verziju za testiranje i evaluaciju

  http://smartwavebeta.cloudapp.net/

- Kod sa predavanja biće dostupan na:

  https://github.com/milanskoric

PARTNERI KONFERENCIJE

Microsoft

Singidunum University

SPONZORI

ComTrade

acer

MDS

ewe comp

OBLAK TEHNOLOGIJE

MERA Software Services, Inc.

Merit SOLUTIONS

Hosted by EUnet eunethosting.com Hyper-V Cloud

JET BRAINS

Coca-Cola Hellenic Passion for Excellence

PRIJATELJI KONFERENCIJE

MSCommunity.ba

VEEAM IT JUST WORKS!™

CityExpert

TARABICA#
IT CONFERENCE
BELGRADE