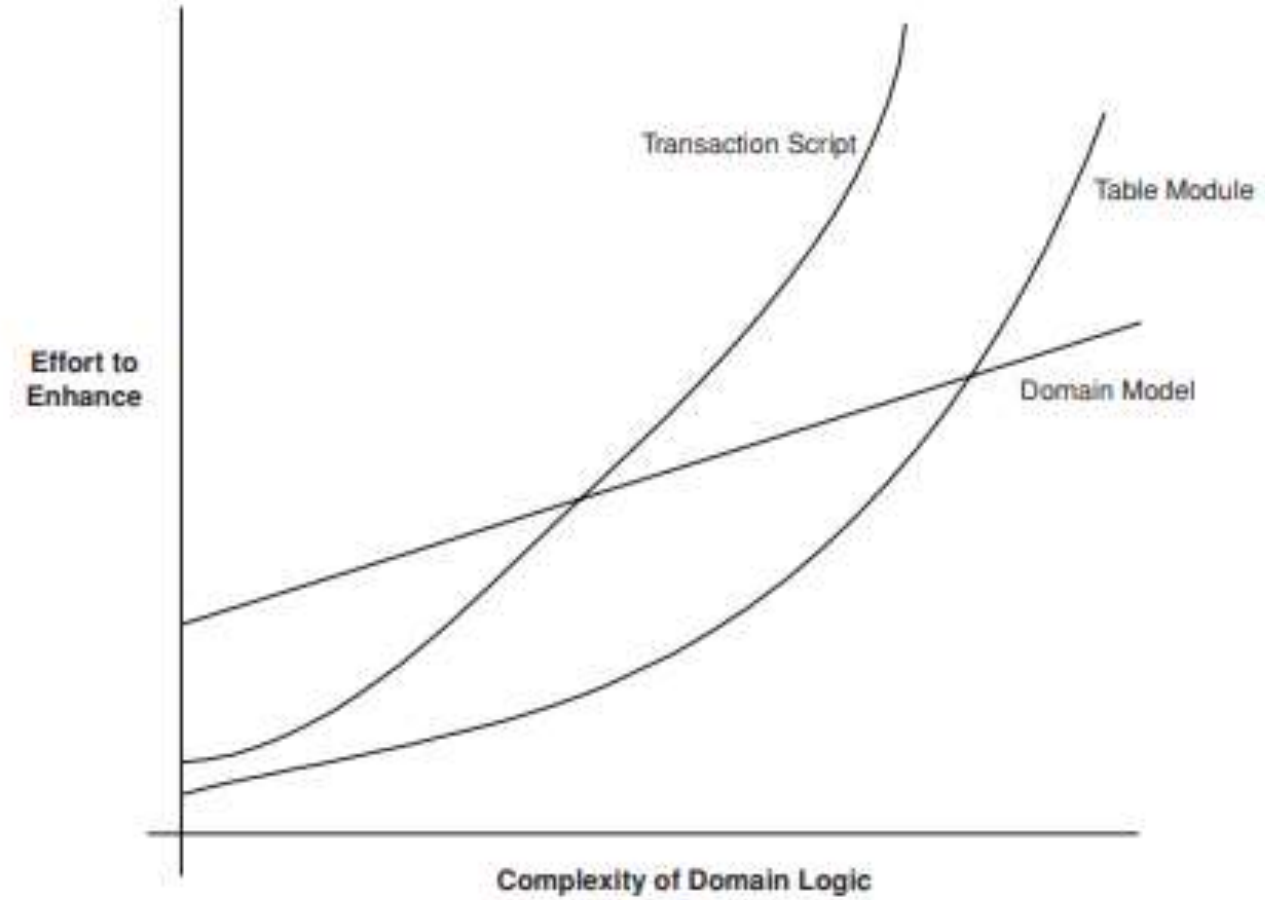
Модели данных
Бизнес-логика

Подходы к организации бизнес-логики

Согласно “Patterns of enterprise application architecture”, Martin Fowler:

- Transaction Script / Транзакционный сценарий
- Table Module / Табличный модуль
- Domain Model / Модель предметной области
  - Anemic Domain Model / Анемичная модель предметной области
  - Rich Domain Model / Насыщенная модель предметной области

Подходы к организации бизнес-логики



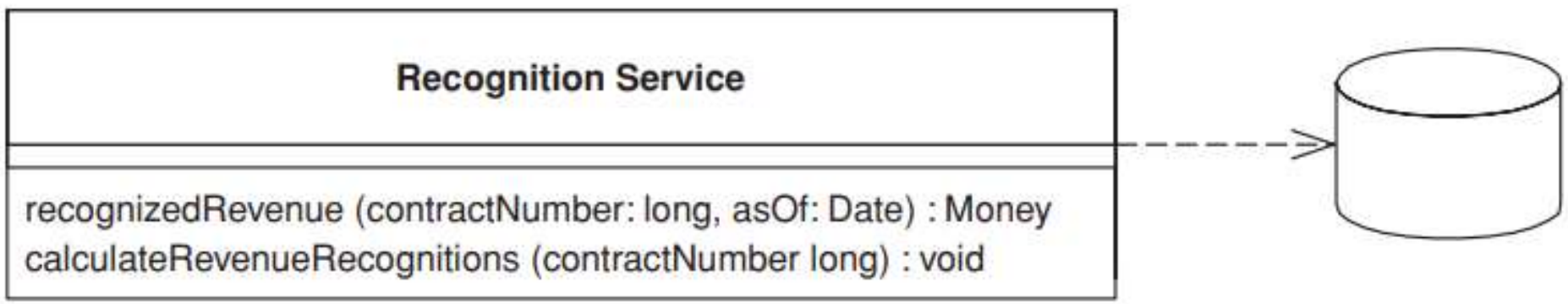
Transaction Script

Самый простой подход к организации бизнес-логики.

Бизнес-логика разбивается на процедуры, каждая из которых соответствует конкретному запросу, поступающему от слоя представления

Transaction Script

Organizes business logic by procedures where each procedure handles a single request from the presentation.



Transaction Script. Пример

```
1   using CodeParadise.Money;
2   using System;
3   using System.Data;
4   using System.Data.SQLite;
5
6   namespace PoEAA_TransactionScript
7   {
8       public class Gateway
9       {
10          private const string FindRecognitionsStatement =
11              @"
12                  SELECT Amount FROM RevenueRecognitions
13                  WHERE Contract = $contractId AND RecognizedOn <= $beforeDate
14              ";
15
16          private const string FindContractStatement =
17              @"
18                  SELECT * FROM Contracts c, Products p
19                  WHERE c.Id = $contractId AND c.product = p.Id
20              ";
21
22          private const string InsertRecognitionsStatement =
23              @"
24                  INSERT INTO RevenueRecognitions VALUES ($contractId, $amount, $recognizedOn)
25              ";
```

Transaction Script. Пример

```
27          public DataTable FindRecognitionsFor(int contractId, DateTime beforeDate)
28          {
29              var result = new DataTable();
30              using var connection = DbManager.CreateConnection();
31              connection.Open();
32              var command = connection.CreateCommand();
33              command.CommandText = FindRecognitionsStatement;
34              command.Parameters.AddWithValue("$contractId", contractId);
35              command.Parameters.AddWithValue("$beforeDate", beforeDate);
36
37              using(var sqlDataAdapter = new SQLiteDataAdapter(command))
38              {
39                  sqlDataAdapter.Fill(result);
40              }
41
42              return result;
43          }
```

Transaction Script. Пример

```
45          public DataTable FindContract(int contractId)
46          {
47              var result = new DataTable();
48              using var connection = DbManager.CreateConnection();
49              connection.Open();
50              var command = connection.CreateCommand();
51              command.CommandText = FindContractStatement;
52              command.Parameters.AddWithValue("$contractId", contractId);
53
54              using(var sqlDataAdapter = new SQLiteDataAdapter(command))
55              {
56                  sqlDataAdapter.Fill(result);
57              }
58
59              return result;
60          }
```

Transaction Script. Пример

```
62          public void InsertRecognitions(int contractId, Money amount, DateTime recognizedOn)
63          {
64              using var connection = DbManager.CreateConnection();
65              connection.Open();
66              var command = connection.CreateCommand();
67              command.CommandText = InsertRecognitionsStatement;
68              command.Parameters.AddWithValue("$contractId", contractId);
69              command.Parameters.AddWithValue("$amount", amount.Amount);
70              command.Parameters.AddWithValue("$recognizedOn", recognizedOn);
71              command.ExecuteNonQuery();
72          }
```

Transaction Script. Пример

```
1   using CodeParadise.Money;
2   using System;
3
4   namespace PoEAA_TransactionScript
5   {
6       class RecognitionService
7       {
8           public Money RecognizedRevenue(int contractNumber, DateTime beforeDate)
9           {
10              Money result = Money.Dollars(0m);
11              Gateway db = new Gateway();
12              var dt = db.FindRecognitionsFor(contractNumber, beforeDate);
13              for (int i = 0; i < dt.Rows.Count; ++i)
14              {
15                  var amount = (decimal) dt.Rows[i]["Amount"];
16                  result += Money.Dollars(amount);
17              }
18
19              return result;
20          }
```

Transaction Script. Пример

```
22          public void CalculateRevenueRecognitions(int contractId)
23          {
24              Gateway db = new Gateway();
25              var contracts = db.FindContract(contractId);
26              Money totalRevenue = Money.Dollars((decimal) contracts.Rows[0]["Revenue"]);
27              DateTime recognitionDate = (DateTime) contracts.Rows[0]["DateSigned"];
28              string type = contracts.Rows[0]["Type"].ToString();
29
30              if(type == "S")
31              {
32                  Money[] allocation = totalRevenue.Allocate(3);
33                  db.InsertRecognitions(contractId, allocation[0], recognitionDate);
34                  db.InsertRecognitions(contractId, allocation[1], recognitionDate.AddDays(60));
35                  db.InsertRecognitions(contractId, allocation[2], recognitionDate.AddDays(90));
36              }
37              else if(type == "W")
38              {
39                  db.InsertRecognitions(contractId, totalRevenue, recognitionDate);
40              }
41              else if(type == "D")
42              {
43                  Money[] allocation = totalRevenue.Allocate(3);
44                  db.InsertRecognitions(contractId, allocation[0], recognitionDate);
45                  db.InsertRecognitions(contractId, allocation[1], recognitionDate.AddDays(30));
46                  db.InsertRecognitions(contractId, allocation[2], recognitionDate.AddDays(60));
47              }
48          }
```

Transaction Script. Пример

```
RecognitionService service = new RecognitionService();

// database product
service.CalculateRevenueRecognitions(1);
var databaseRevenue = service.RecognizedRevenue(1, new System.DateTime(2020, 1, 25));
Console.WriteLine($"database revenue before 2020-01-25 = {databaseRevenue.Amount}");

// spreadsheet product
service.CalculateRevenueRecognitions(2);
var spreadsheetRevenue = service.RecognizedRevenue(2, new System.DateTime(2020, 6, 1));
Console.WriteLine($"spreadsheet revenue before 2020-06-01 = {spreadsheetRevenue.Amount}");

 // word processor product
service.CalculateRevenueRecognitions(3);
var wordProcessorRevenue = service.RecognizedRevenue(3, new System.DateTime(2020, 9, 30));
Console.WriteLine($"word processor revenue before 2020-09-30 = {wordProcessorRevenue.Amount}");
```

Transaction Script. Пример с использованием паттерна Команда

```
public interface ICommand<in T>
{
    void Execute(T model);
}
```

```
public class TransactionScriptModel
{
    public Guid Id { get; set; }
    public string Status { get; set; }
}
```

Transaction Script. Пример с использованием паттерна Команда

```
public class TransactionScript<T>
{
    private ICommand<T> _command;

    public void SetCommand(ICommand<T> command)
    {
        _command = command;
    }

    public void ExecuteCommand(T model)
    {
        _command.Execute(model);
    }
}
```

Transaction Script. Пример с использованием паттерна Команда

```
public class InsertCommand : ICommand<TransactionScriptModel>
{
    private readonly string _connectionString;

    public InsertCommand(string connectionString)
    {
        _connectionString = connectionString;
    }
    public void DoStuff(TransactionScriptModel model)
    {
        if (model == null || model.Status == "Fake")
        {
            throw new Exception("Bad data");
        }
    }
```

Transaction Script. Пример с использованием паттерна Команда

```
public void Execute(TransactionScriptModel model)
{
    using (IDbConnection db = new SqlConnection(_connectionString))
    {
        db.Open();
        using (IDbTransaction transaction = db.BeginTransaction(IsolationLevel.ReadUncommitted))
        {
            try
            {
                db.CreateCommand().Transaction = transaction;
                var sqlQuery = "INSERT INTO TransactionScript (Id,Status) VALUES(@Id, @Status)";
                db.Execute(sqlQuery, model, transaction);
                DoStuff(model);
                transaction.Commit();
            }
            catch
            {
                transaction.Rollback();
                throw;
            }
        }
    }
}
```

Transaction Script. Пример с использованием паттерна Команда

```
public class TransactionScriptTests
{
    private string connString =
        "Data Source=localhost;Initial Catalog=test;Integrated Security=True;MultipleActiveResultSets=True";

    [Fact]
    public void ExecuteInsertCommand_WithValidData_ShouldInsertEntityToDb()
    {
        var ts = new TransactionScript<TransactionScriptModel>();
        ts.SetCommand(new InsertCommand(connString));
        ts.ExecuteCommand(new TransactionScriptModel
        {
            Status = "Done",
            Id = Guid.NewGuid()
        });
    }

    [Fact]
    public void ExecuteInsertCommand_WithBadStatus_ShouldThrowException()
    {
        var ts = new TransactionScript<TransactionScriptModel>();
        ts.SetCommand(new InsertCommand(connString));
        Assert.Throws<Exception>(() => ts.ExecuteCommand(new TransactionScriptModel
        {
            Status = "Fake",
            Id = Guid.NewGuid()
        }));
    }
}
```

Transaction Script. Преимущества и недостатки

Преимущества:

- Простота
- Каждая транзакция независима

Недостатки:

- При увеличении сложности бизнес-логики приводит к непомерному росту класса, дублированию кода
- Сложно рефакторить

Table Module

Прямое сопоставление класса таблице в базе данных. Экземпляр такого класса (или статический класс) соответствует таблице ЦЕЛИКОМ.

Table Module неплохо работает с Record Set

Table Module

A single instance that handles the business logic for all rows in a database table or view.

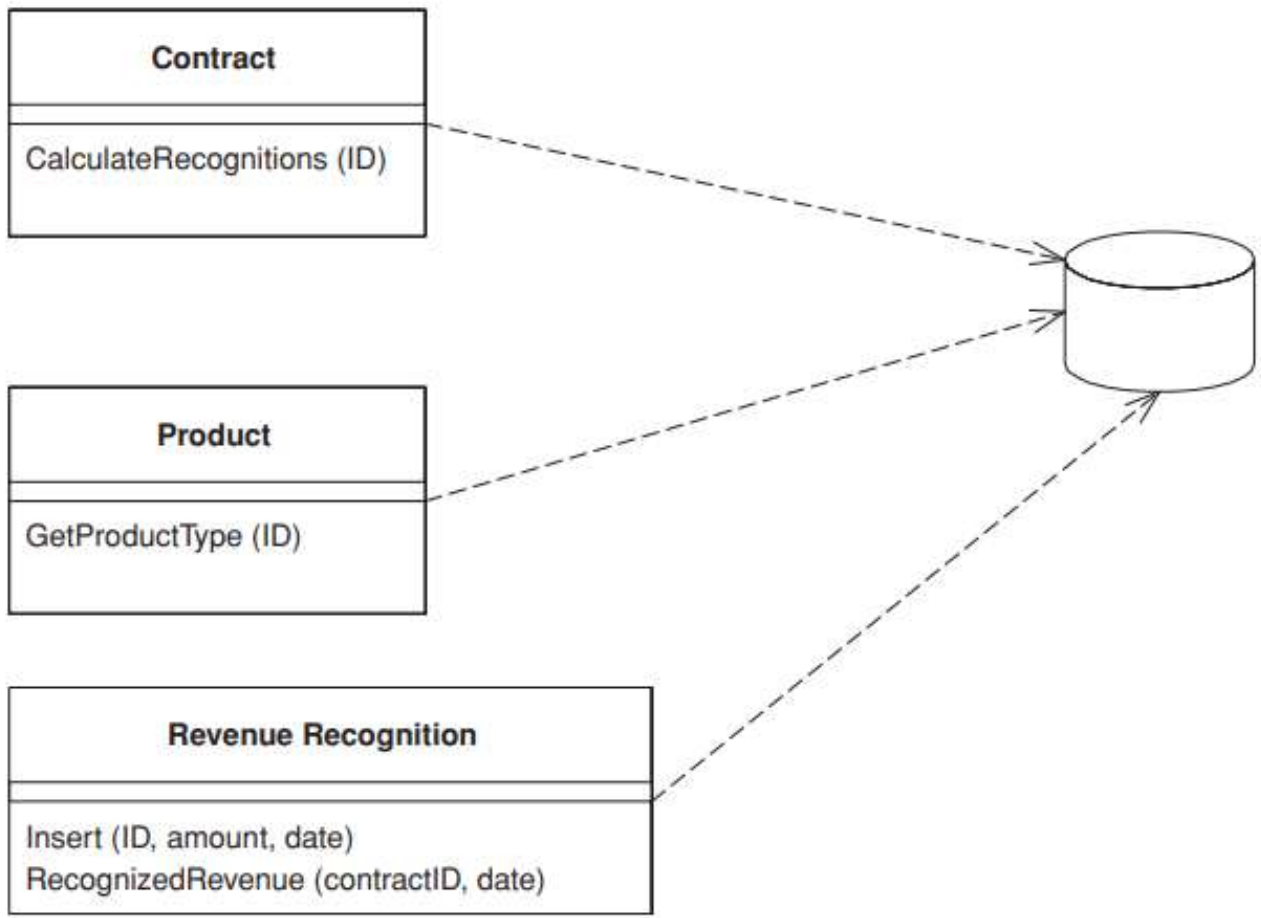


Table Module. Пример

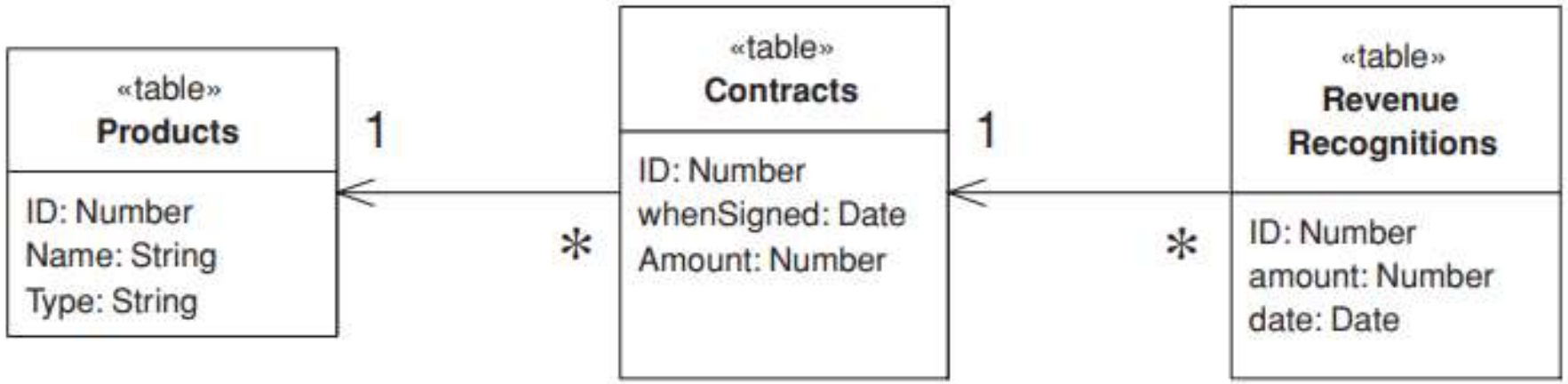


Figure 9.6 *Database schema for revenue recognition.*

Table Module. Пример

```
using System.Data;

namespace PoEAA_TableModule
{
    abstract class TableModule
    {
        protected DataTable Table;

        protected TableModule(DataSet ds, string tableName)
        {
            Table = ds.Tables[tableName];
        }
    }
}
```

Table Module. Пример

```
using System;
using System.Data;

namespace PoEAA_TableModule
{
    public enum ProductType
    {
        W,
        S,
        D
    };

    class Product : TableModule
    {
        public Product(DataSet tableDataSet) : base(tableDataSet, "Products")
        {
        }

        public DataRow this[int key]
        {
            get
            {
                string filter = $"Id = {key}";
                return Table.Select(filter)[0];
            }
        }

        public ProductType GetProductType(int prodId)
        {
            string typeCode = (string) this[prodId]["Type"];
            return (ProductType) Enum.Parse(typeof(ProductType), typeCode);
        }
    }
}
```

Table Module. Пример

```
1   using System;
2   using System.Data;
3
4   namespace PoEAA_TableModule
5   {
6       class RevenueRecognition : TableModule
7       {
8           private static int _id = 1;
9           private static readonly object IdLock = new object();
10          public RevenueRecognition(DataSet ds) : base(ds, "RevenueRecognitions")
11          {
12          }
13
14          public int Insert(int contractId, decimal amount, DateTime date)
15          {
16              DataRow newRow = Table.NewRow();
17              int id = GetNextId();
18              newRow["Id"] = id;
19              newRow["Contract"] = contractId;
20              newRow["Amount"] = amount;
21              newRow["RecognizedOn"] = date;
22              Table.Rows.Add(newRow);
23
24              return id;
25          }
26
```

Table Module. Пример

```
27          public decimal RecognizedRevenue(int contractId, DateTime asOf)
28          {
29              string filter = $"Contract = {contractId} AND RecognizedOn <= #{asOf:d}#";
30              DataRow[] rows = Table.Select(filter);
31              decimal result = 0m;
32              foreach(DataRow row in rows)
33              {
34                  result += (decimal) row["Amount"];
35              }
36
37              return result;
38          }
39
40          private static int GetNextId()
41          {
42              lock (IdLock)
43              {
44                  return _id++;
45              }
46          }
47      }
48  }
```

Table Module. Пример

```
1  using System;
2  using System.Data;
3
4  namespace PoEAA_TableModule
5  {
6      class Contract : TableModule
7      {
8          public Contract(DataSet ds) : base(ds, "Contracts")
9          {
10         }
11
12         public DataRow this[int key]
13         {
14             get
15             {
16                 string filter = $"Id = {key}";
17                 return Table.Select(filter)[0];
18             }
19         }
```

Table Module. Пример

```
21          public void CalculateRecognitions(int contractId)
22          {
23              DataRow contractRow = this[contractId];
24              decimal amount = (decimal) contractRow["Revenue"];
25              RevenueRecognition rr = new RevenueRecognition(Table.DataSet);
26              Product prod = new Product(Table.DataSet);
27              int prodId = GetProductId(contractId);
28
29              var productType = prod.GetProductType(prodId);
30
31              if (productType == ProductType.W)
32              {
33                  rr.Insert(contractId, amount, GetWhenSigned(contractId));
34              }
35              else if (productType == ProductType.S)
36              {
37                  DateTime signedDate = GetWhenSigned(contractId);
38                  decimal[] allocation = Allocate(amount, 3);
39                  rr.Insert(contractId, allocation[0], signedDate);
40                  rr.Insert(contractId, allocation[1], signedDate.AddDays(60));
41                  rr.Insert(contractId, allocation[2], signedDate.AddDays(90));
42              }
43              else if (productType == ProductType.D)
44              {
45                  DateTime signedDate = GetWhenSigned(contractId);
46                  decimal[] allocation = Allocate(amount, 3);
47                  rr.Insert(contractId, allocation[0], signedDate);
48                  rr.Insert(contractId, allocation[1], signedDate.AddDays(30));
49                  rr.Insert(contractId, allocation[2], signedDate.AddDays(60));
50              }
51          }
```

Table Module. Пример

```
53          private decimal[] Allocate(decimal amount, int by)
54          {
55              decimal lowResult = amount / by;
56              lowResult = decimal.Round(lowResult, 2);
57              decimal highReult = lowResult + 0.01m;
58              decimal[] results = new decimal[by];
59              int remainder = (int) amount % by;
60              for (int i = 0; i < remainder; ++i)
61              {
62                  results[i] = highReult;
63              }
64
65              for (int i = remainder; i < by; ++i)
66              {
67                  results[i] = lowResult;
68              }
69
70              return results;
71          }
72
73          private DateTime GetWhenSigned(int contractId)
74          {
75              return (DateTime)this[contractId]["DateSigned"];
76          }
77
78          private int GetProductId(int contractId)
79          {
80              return (int)this[contractId]["Product"];
81          }
82      }
83  }
```

Table Module. Пример

```
1   using System;
2   using System.Data;
3
4   namespace PoEAA_TableModule
5   {
6       class Program
7       {
8           static void Main(string[] args)
9           {
10              // mock Result Set
11              DataSet ds = new DataSet();
```

Table Module. Пример

```
13              DataTable productTable = new DataTable("Products");
14              productTable.Columns.Add("Id", typeof(int));
15              productTable.Columns.Add("Name", typeof(string));
16              productTable.Columns.Add("Type", typeof(string));
17              productTable.Rows.Add(1, "Code Paradise Database", "D");
18              productTable.Rows.Add(2, "Code Paradise Spreadsheet", "S");
19              productTable.Rows.Add(3, "Code Paradise Word Processor", "W");
```

Table Module. Пример

```
21              DataTable contractTable = new DataTable("Contracts");
22              contractTable.Columns.Add("Id", typeof(int));
23              contractTable.Columns.Add("Product", typeof(int));
24              contractTable.Columns.Add("Revenue", typeof(decimal));
25              contractTable.Columns.Add("DateSigned", typeof(DateTime));
26              contractTable.Rows.Add(1, 1, 9999, new DateTime(2020,1,1));
27              contractTable.Rows.Add(2, 2, 1000, new DateTime(2020, 3, 15));
28              contractTable.Rows.Add(3, 3, 24000, new DateTime(2020, 7, 25));
```

Table Module. Пример

```
30              DataTable revenueRecognitionsTable = new DataTable("RevenueRecognitions");
31              revenueRecognitionsTable.Columns.Add("Id", typeof(int));
32              revenueRecognitionsTable.Columns.Add("Contract", typeof(int));
33              revenueRecognitionsTable.Columns.Add("Amount", typeof(decimal));
34              revenueRecognitionsTable.Columns.Add("RecognizedOn", typeof(DateTime));
35
36              ds.Tables.Add(productTable);
37              ds.Tables.Add(contractTable);
38              ds.Tables.Add(revenueRecognitionsTable);
```

Table Module. Пример

```
42              // calculate recognized revenues
43              Contract contract = new Contract(ds);
44
45              // database product
46              contract.CalculateRecognitions(1);
47              var databaseRevenue = new RevenueRecognition(ds).RecognizedRevenue(1, new DateTime(2020, 1, 25));
48              Console.WriteLine($"database revenue before 2020-01-25 = {databaseRevenue}");
49
50              // spreadsheet product
51              contract.CalculateRecognitions(2);
52              var spreadsheetRevenue = new RevenueRecognition(ds).RecognizedRevenue(2, new DateTime(2020, 6, 1));
53              Console.WriteLine($"spreadsheet revenue before 2020-06-01 = {spreadsheetRevenue}");
54
55              // word processor product
56              contract.CalculateRecognitions(3);
57              var wordProcessorRevenue = new RevenueRecognition(ds).RecognizedRevenue(3, new DateTime(2020, 9, 30));
58              Console.WriteLine($"word processor revenue before 2020-09-30 = {wordProcessorRevenue}");
59          }
60      }
61  }
```

Table Module. Пример

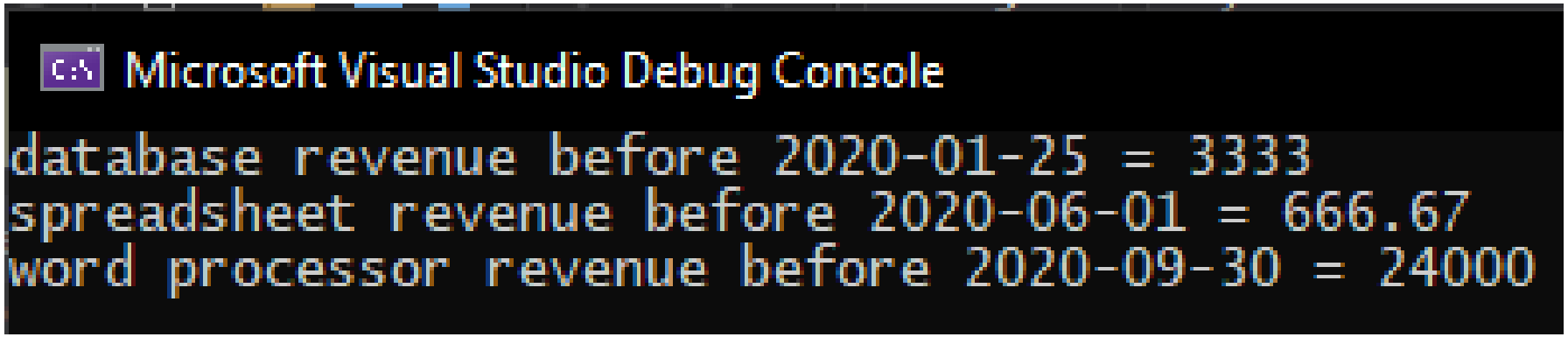


Table Module. Несколько Table Module работают с одним Record Set

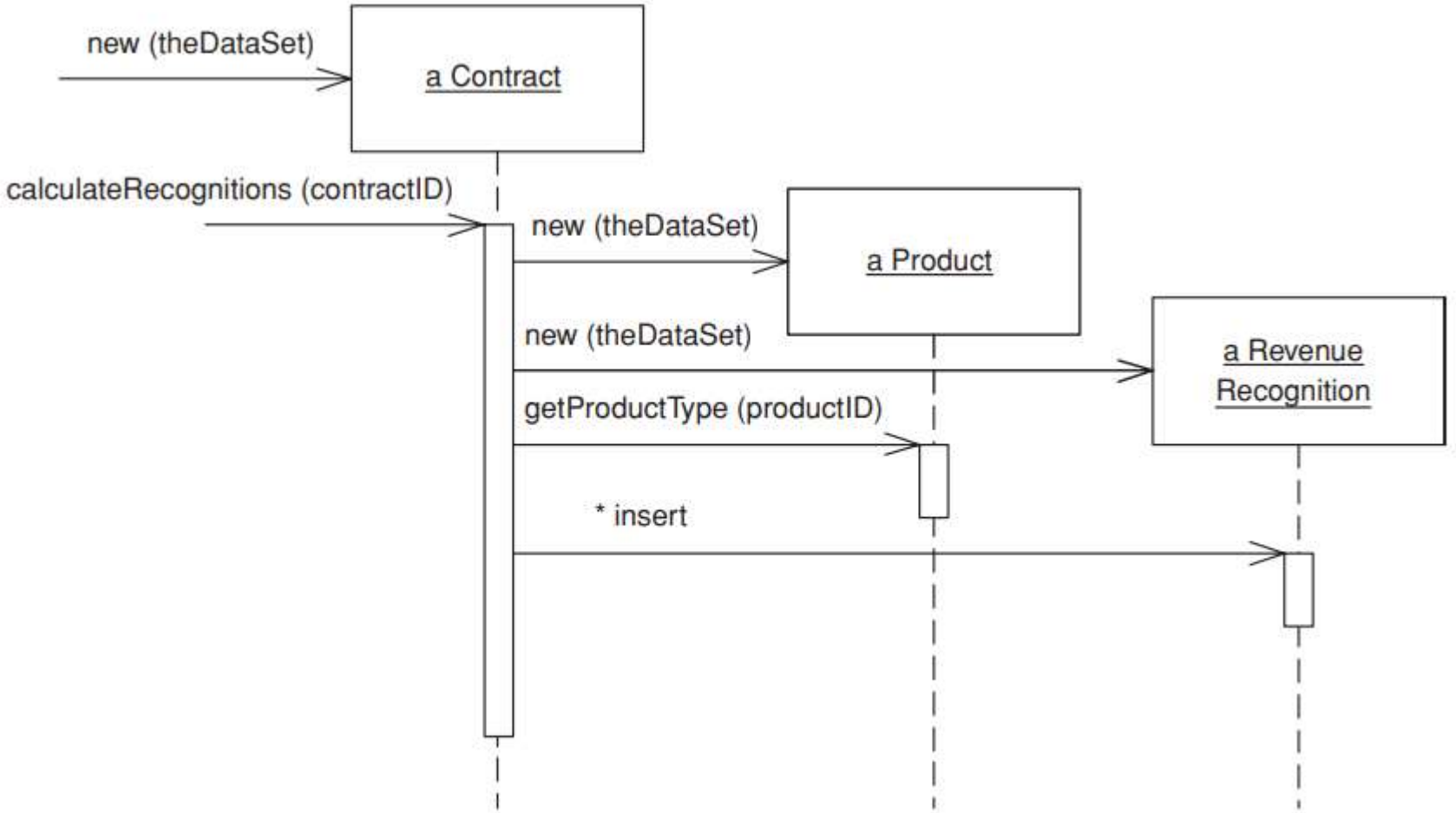


Figure 9.4 *Several* Table Modules *can collaborate with a single* Record Set (508).

Table Module. Взаимодействие с Table Data Gateway

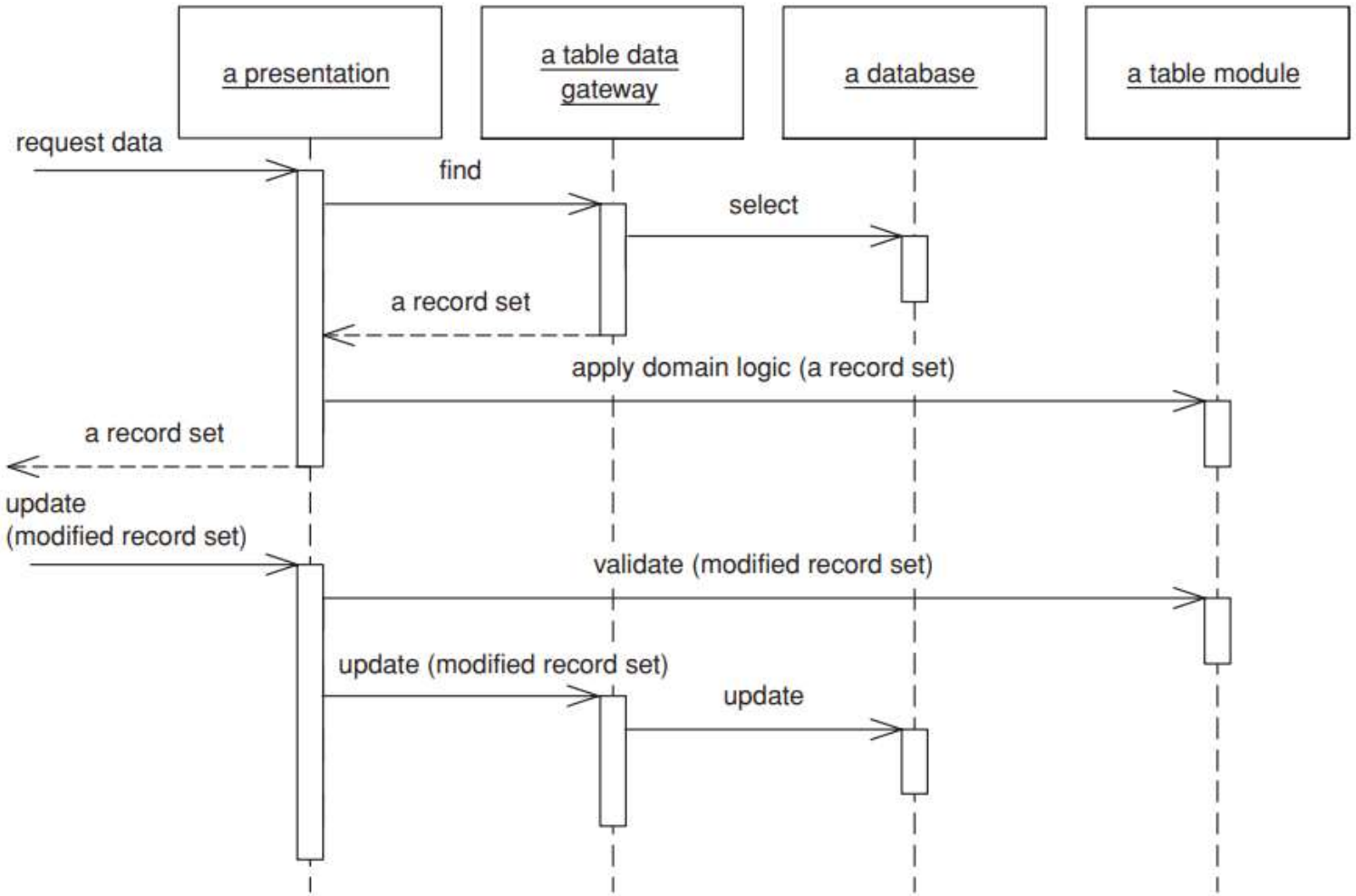


Figure 9.5 *Typical interactions for the layers around a* Table Module.

Table Module. Преимущества и недостатки

Преимущества:

- Лёгкая интеграция с реляционной БД
- Код лучше структурирован, чем в Transaction Script

Недостатки:

- Не выйдет использовать всю мощь объектов: полиморфизм, прямая композиция между классами таблиц
- Придётся всё время иметь дело с идентификаторами

Domain Model

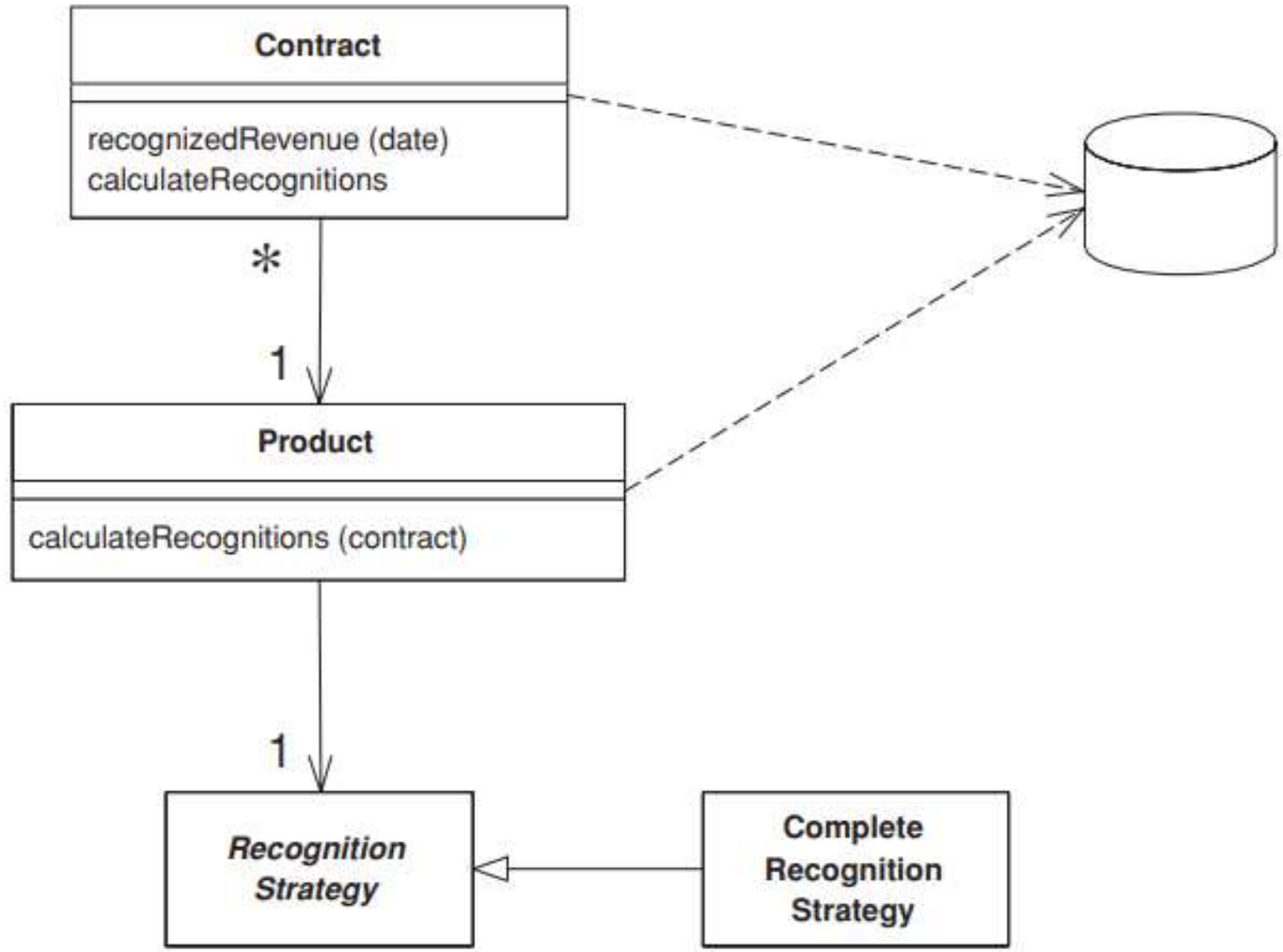
Выделяются объекты, соответствующие объектам предметной области.

Описываются отношения между такими объектами, соответствующие отношениям между объектами реального мира

Решение технологических вопросов, таких как хранение, безопасность, управление транзакциями, как правило, выносится за пределы слоя бизнес-логики

Domain Model

An object model of the domain that incorporates both behavior and data.



Domain Model

Выделяют два варианта Domain Model:

- Rich Domain Model - данные и поведение инкапсулируются внутри объектов предметной области
- Anemic Domain Model - в объектах предметной области инкапсулируются только данные, поведение же выносится в слой сервисов, расположенный поверх слоя предметной области

Domain Model

Rich Domain Model лучше работает в связке с Data Mapper

Anemic Domain Model больше подходит Active Record

Rich vs. Anemic

Rich Domain Model:

- Реализует возможности ООП по полной
- Проектировать и разрабатывать труднее (зато потом жить проще)
- Довольно часто классы получаются большими

Rich vs. Anemic

Anemic Domain Model:

- Отклоняется от принципов ООП - т.к. объекты предметной области в случае анемичной доменной модели не имеют поведения, то мы вступаем в противоречие с базовой идеей ООП - иметь данные и методы их обработки в одном месте
- Проектировать и разрабатывать проще (в начале)
- Простота автоматического создания объектов на основе схемы хранилища (БД)
- Лучше переиспользуемость

Domain Model. Пример

```
1   using CodeParadise.Money;
2   using System;
3
4   namespace PoEAA_DomainModel
5   {
6       class RevenueRecognition
7       {
8           private readonly Money _amount;
9           private readonly DateTime _date;
10
11          public RevenueRecognition(Money amount, DateTime date)
12          {
13              _amount = amount;
14              _date = date;
15          }
16
17          public Money GetAmount()
18          {
19              return _amount;
20          }
21
22          public bool IsRecognizableBy(DateTime asOf)
23          {
24              return asOf.CompareTo(_date) >= 0;
25          }
26      }
27  }
```

Domain Model. Пример

```
1   using CodeParadise.Money;
2   using System;
3   using System.Collections.Generic;
4
5   namespace PoEAA_DomainModel
6   {
7       class Contract
8       {
9           private readonly List<RevenueRecognition> _revenueRecognitions = new List<RevenueRecognition>();
10
11          private readonly Product _product;
12          private readonly Money _revenue;
13          private readonly DateTime _whenSigned;
14          private readonly int _id;
15          private static int _commonId = 1;
16          private static readonly object IdLock = new object();
17
18          public Contract(Product product, Money revenue, DateTime whenSigned)
19          {
20              _product = product;
21              _revenue = revenue;
22              _whenSigned = whenSigned;
23              _id = GenerateNewId();
24          }
```

Domain Model. Пример

```
26          private static int GenerateNewId()
27          {
28              // todo db generate auto increment id
29              lock (IdLock)
30              {
31                  return _commonId++;
32              }
33          }
34
35          public Money RecognizedRevenue(DateTime asOf)
36          {
37              Money result = Money.Dollars(0m);
38              _revenueRecognitions.ForEach(x =>
39                  {
40                      if (x.IsRecognizableBy(asOf))
41                      {
42                          result += x.GetAmount();
43                      }
44                  });
45
46              return result;
47          }
```

Domain Model. Пример

```
49          public Money GetRevenue()
50          {
51              return _revenue;
52          }
53
54          public DateTime GetWhenSigned()
55          {
56              return _whenSigned;
57          }
58
59          public void AddRevenueRecognition(RevenueRecognition revenueRecognition)
60          {
61              // todo
62              // db insert
63
64              // after db insertion, add it to list
65              _revenueRecognitions.Add(revenueRecognition);
66          }
67
68          public void CalculateRecognitions()
69          {
70              _product.CalculateRevenueRecognitions(this);
71          }
72      }
73  }
```

Domain Model. Пример

```
1   namespace PoEAA_DomainModel
2   {
3       abstract class RecognitionStrategy
4       {
5           public abstract void CalculateRevenueRecognitions(Contract contract);
6       }
7   }
```

Domain Model. Пример

```
1   namespace PoEAA_DomainModel
2   {
3       class CompleteRecognitionStrategy : RecognitionStrategy
4       {
5           public override void CalculateRevenueRecognitions(Contract contract)
6           {
7               contract.AddRevenueRecognition(new RevenueRecognition(contract.GetRevenue(), contract.GetWhenSigned()));
8           }
9       }
10  }
```

Domain Model. Пример

```
using CodeParadise.Money;

namespace PoEAA_DomainModel
{
    class ThreeWayRecognitionStrategy : RecognitionStrategy
    {
        private readonly int _firstRecognitionOffset;
        private readonly int _secondRecognitionOffset;

        public ThreeWayRecognitionStrategy(int firstRecognitionOffset, int secondRecognitionOffset)
        {
            _firstRecognitionOffset = firstRecognitionOffset;
            _secondRecognitionOffset = secondRecognitionOffset;
        }

        public override void CalculateRevenueRecognitions(Contract contract)
        {
            Money[] allocation = contract.GetRevenue().Allocate(3);
            contract.AddRevenueRecognition(new RevenueRecognition(allocation[0], contract.GetWhenSigned()));
            contract.AddRevenueRecognition(new RevenueRecognition(allocation[1], contract.GetWhenSigned().AddDays(_firstRecognitionOffset)));
            contract.AddRevenueRecognition(new RevenueRecognition(allocation[2], contract.GetWhenSigned().AddDays(_secondRecognitionOffset)));
        }
    }
}
```

Спасибо за внимание!