

Контроль зависимостей между элементами структуры приложения

Юрий Солдаткин
Архитектор





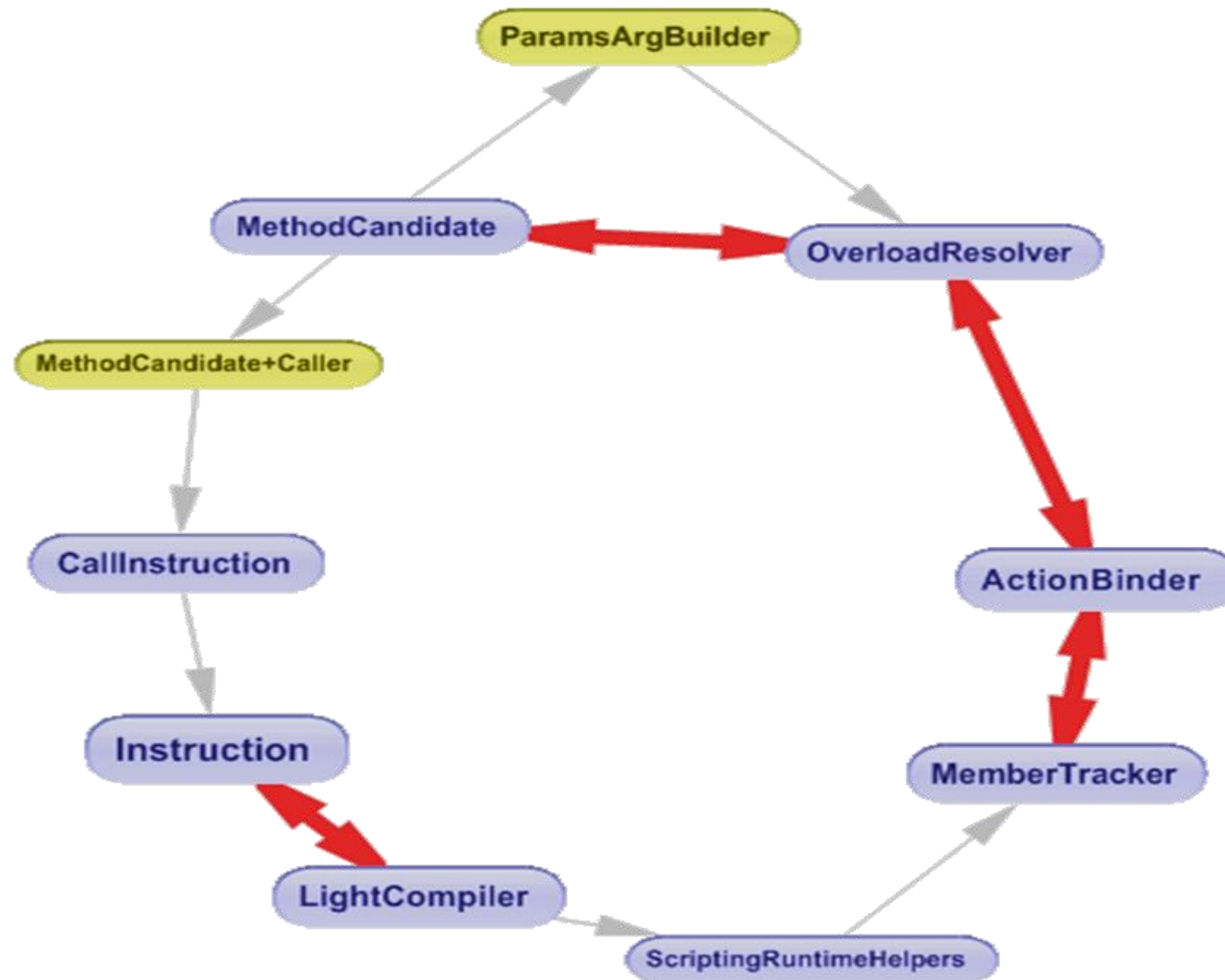


Request to merge `fix/task12345` into `master`

✓  **DonutProject/Donut.Web/Donut.Web.csproj**   5  1   

	1	+ <Project Sdk="Microsoft.NET.Sdk.Web">
2	2	
3	3	<PropertyGroup>
4	4	<TargetFramework>netcoreapp3.1</TargetFramework>
5	5	</PropertyGroup>
6	6	
	7	+ <ItemGroup>
	8	+ <PackageReference
		Include="Microsoft.EntityFrameworkCore.SqlServer" Version="3.1.0" />
	9	+ </ItemGroup>
	10	+

Циклические зависимости



Последствия попустительства

1

Небольшие по смыслу правки приводят к необходимости перелопачивания всего проекта

2

Код плохо тестируется

3

Изменение даже в одном проекте влечет перекомпиляцию большей части решения

Что контролировать

**Связи между
проектами**

**Связи между
пространствами
имен**

***Степень
связности
компонентов**

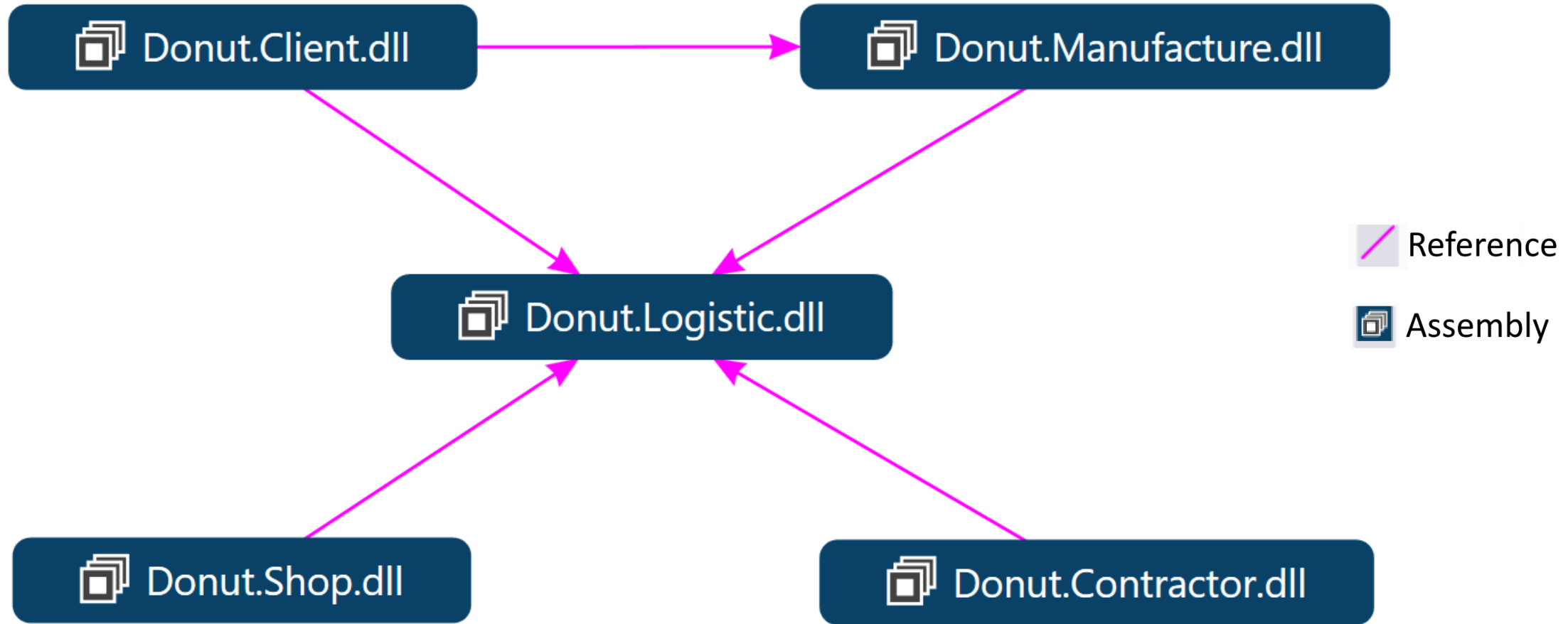
***Внешние
зависимости**

Бизнес-задача

CUSTIS®

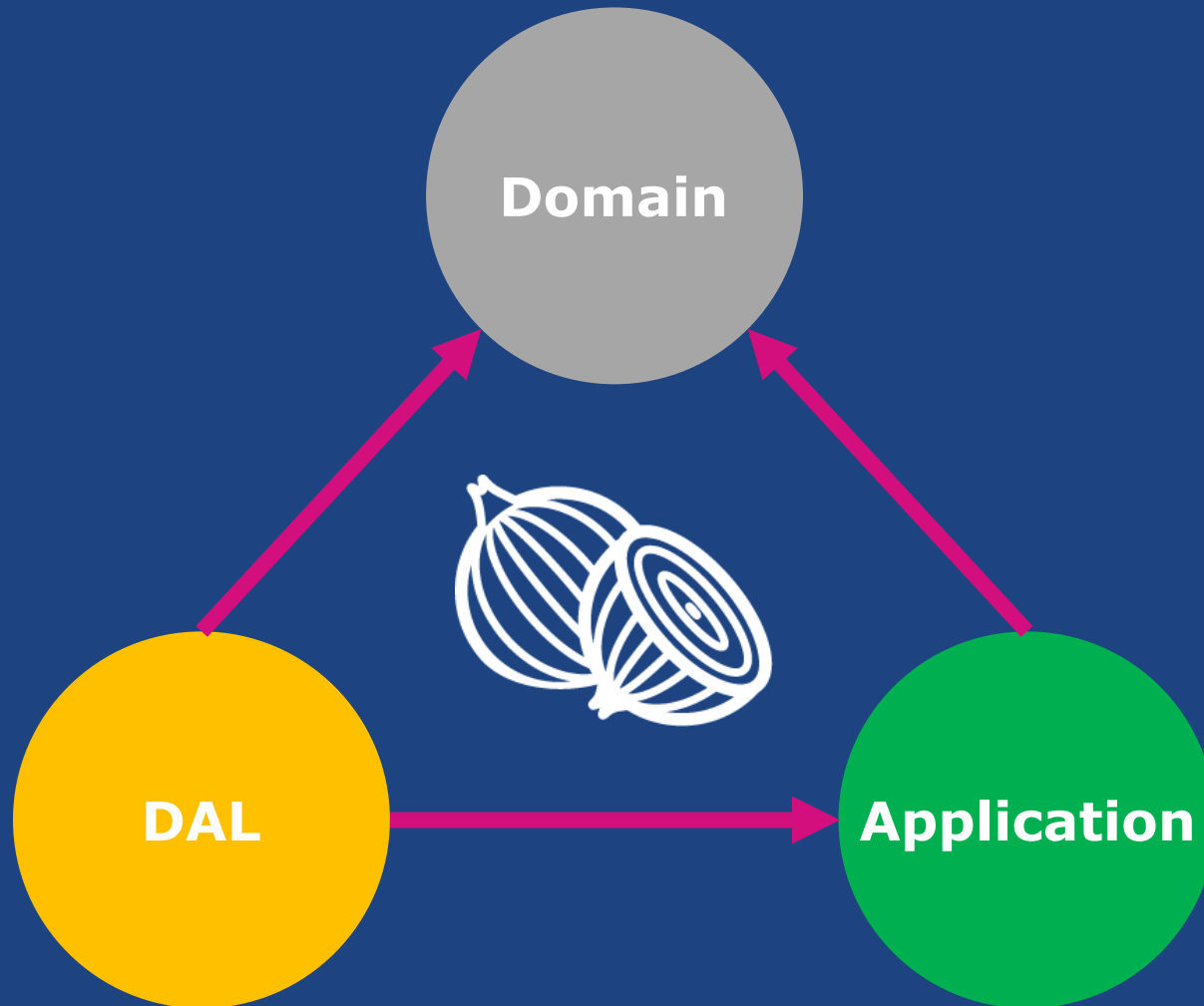


Domain-driven design

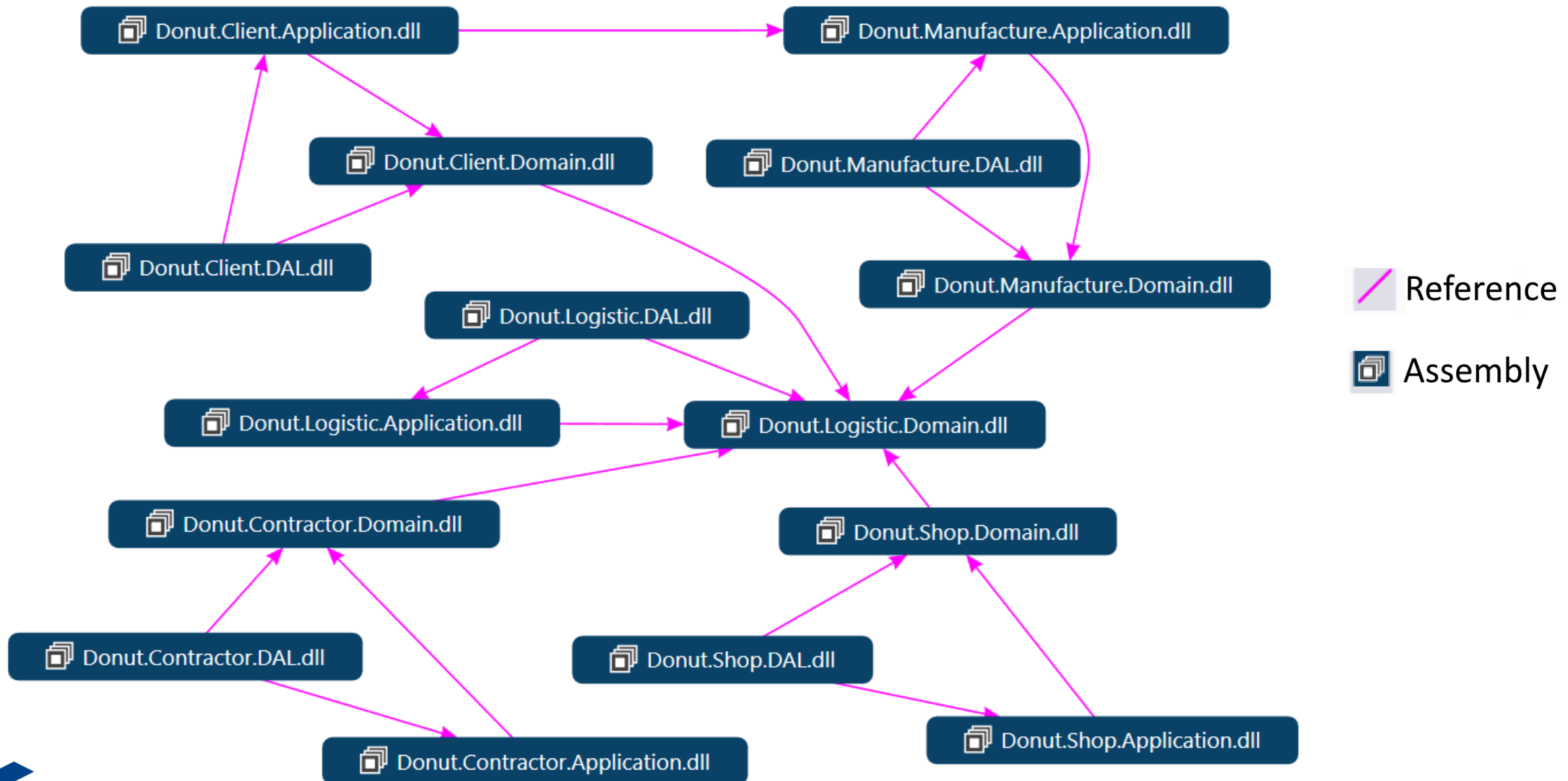


Многоуровневая архитектура

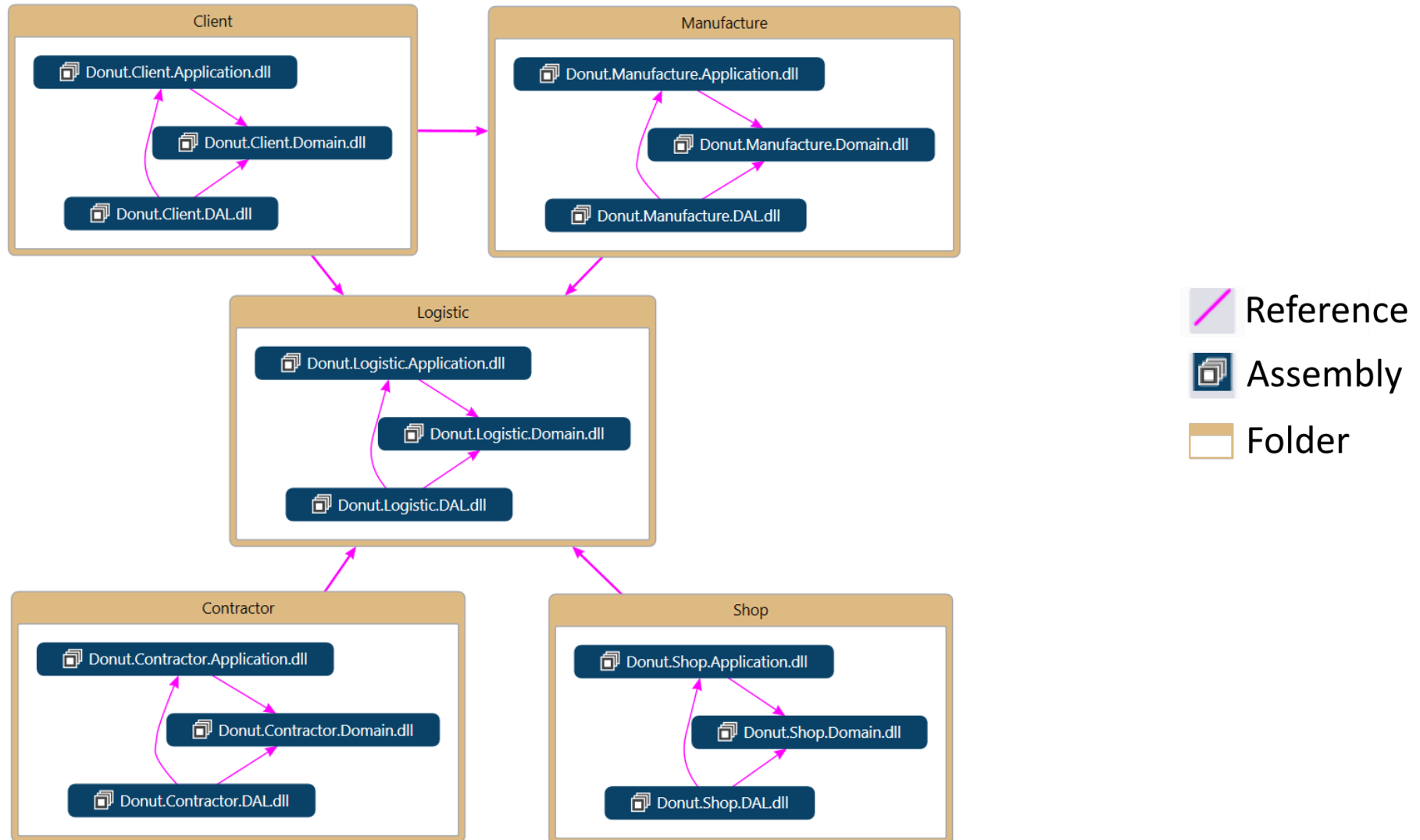
CUSTIS®



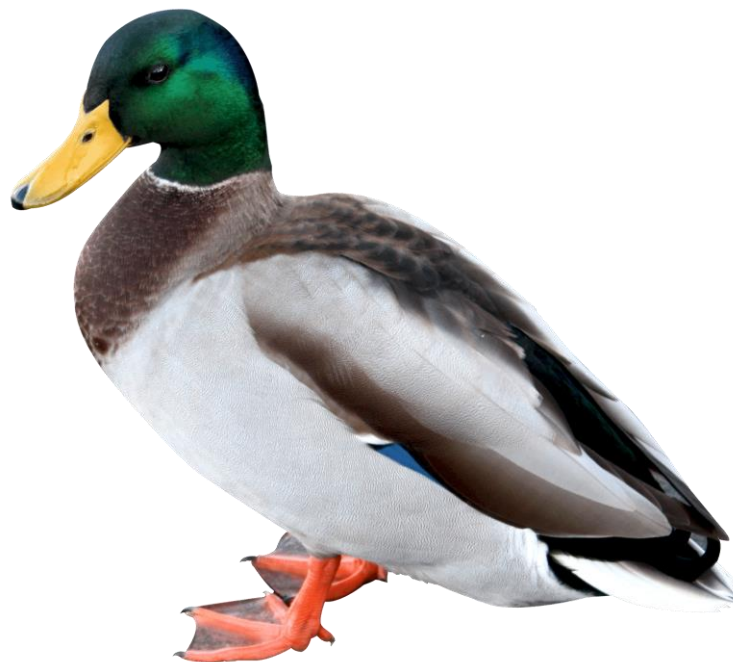
Обновленная схема проектов



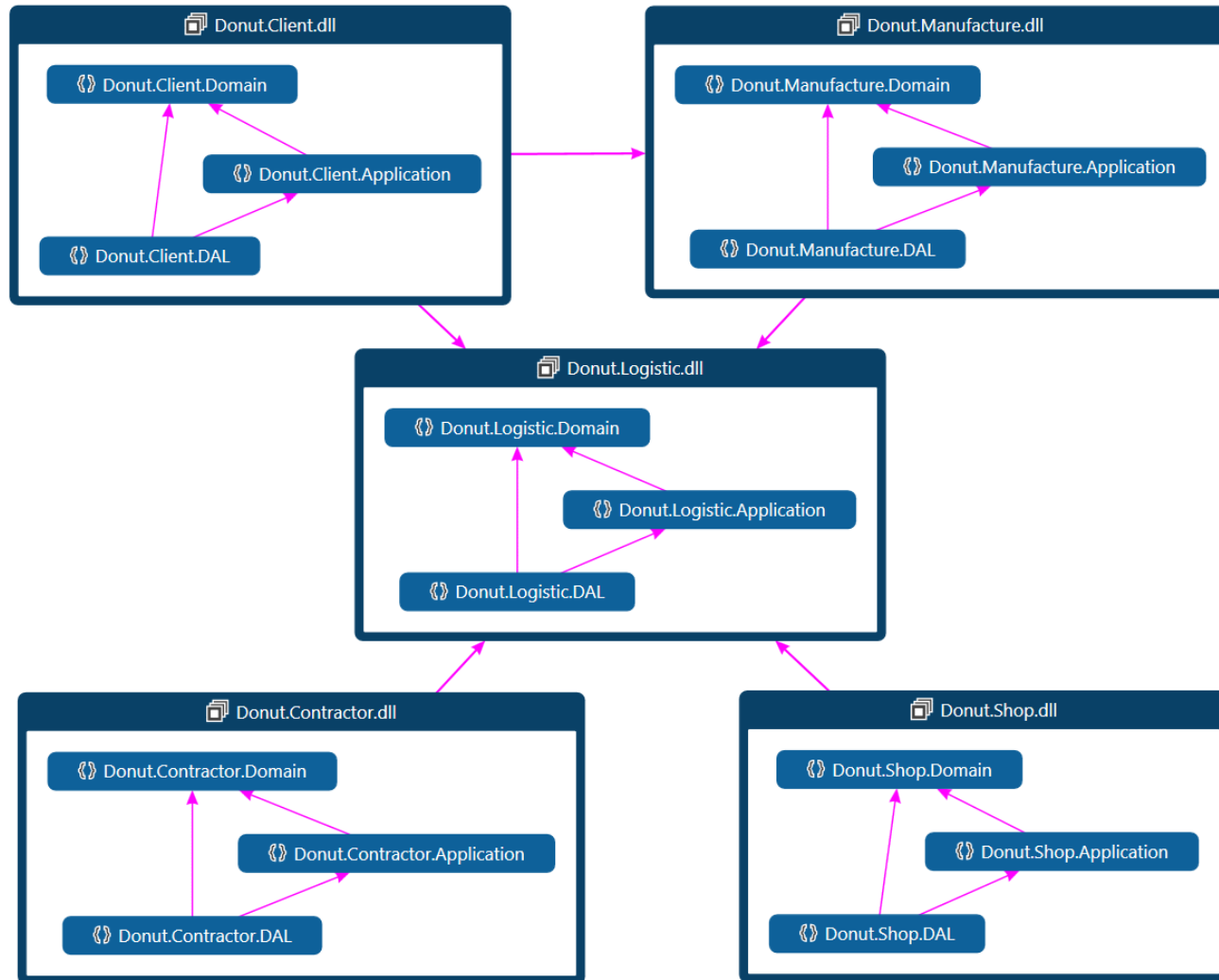
Структура – не только проекты






Альтернатива?



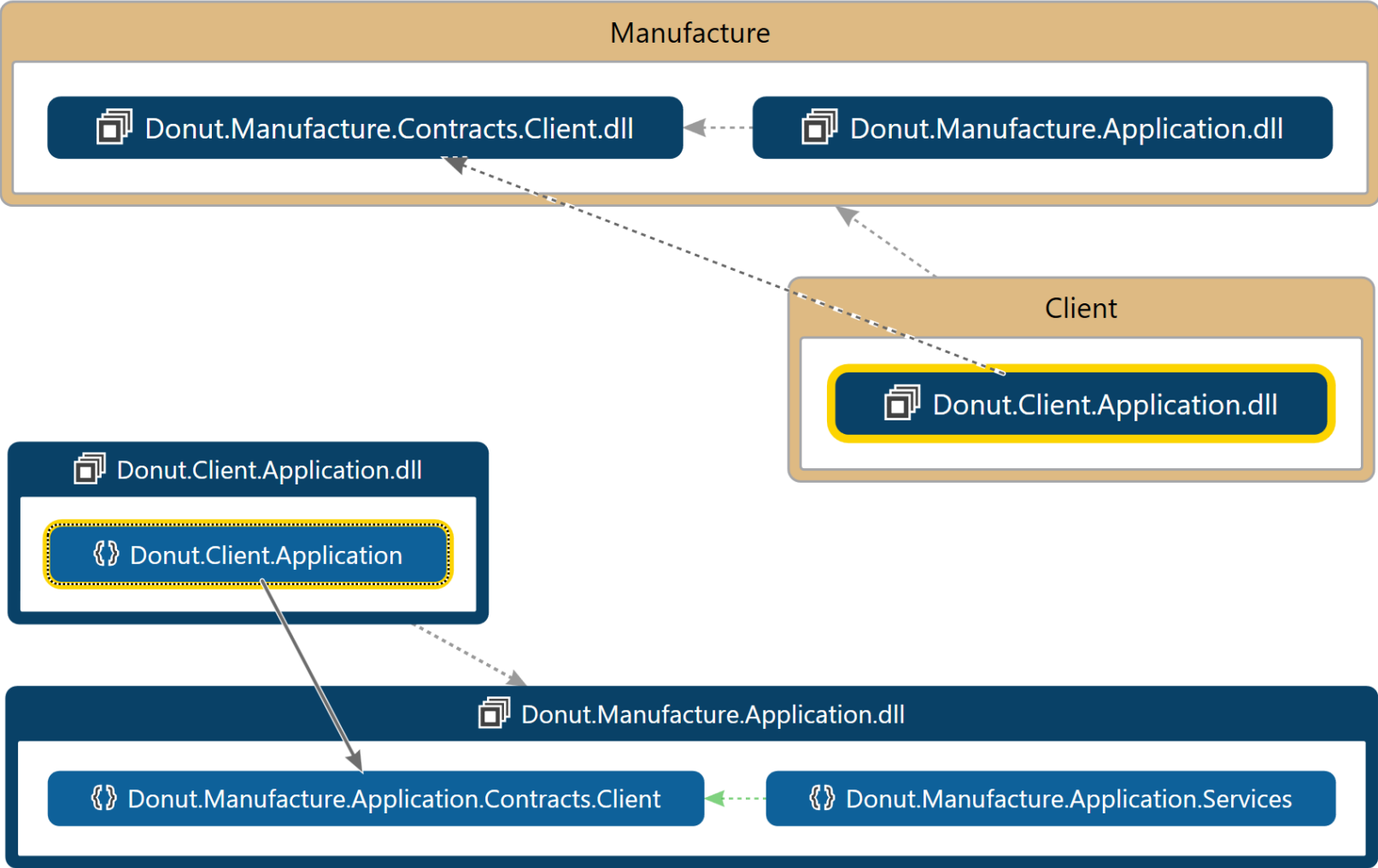
Пространства имен



-  Reference
-  Assembly
-  Namespace



Контракты



Как контролировать



NDepend



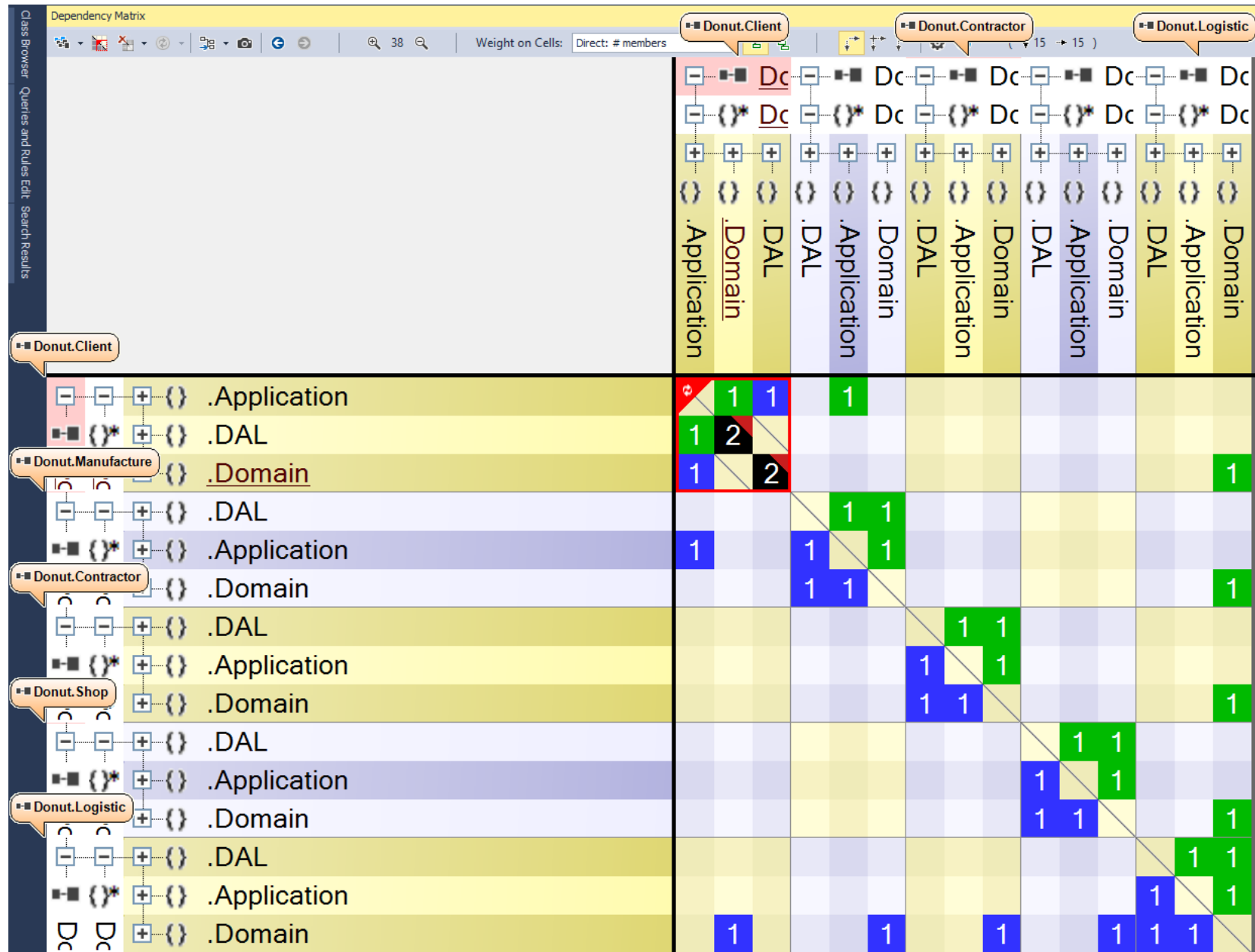
**Visual
Studio
Enterprise**



**Велосипед
(Roslyn)**

NDepend

- **Визуализация**
- **CQLinq**
- ***Степень связности внутри и между сборками/пространствами имен**
- ***Внешние зависимости**



CQLinq

```
warnif count > 0
```

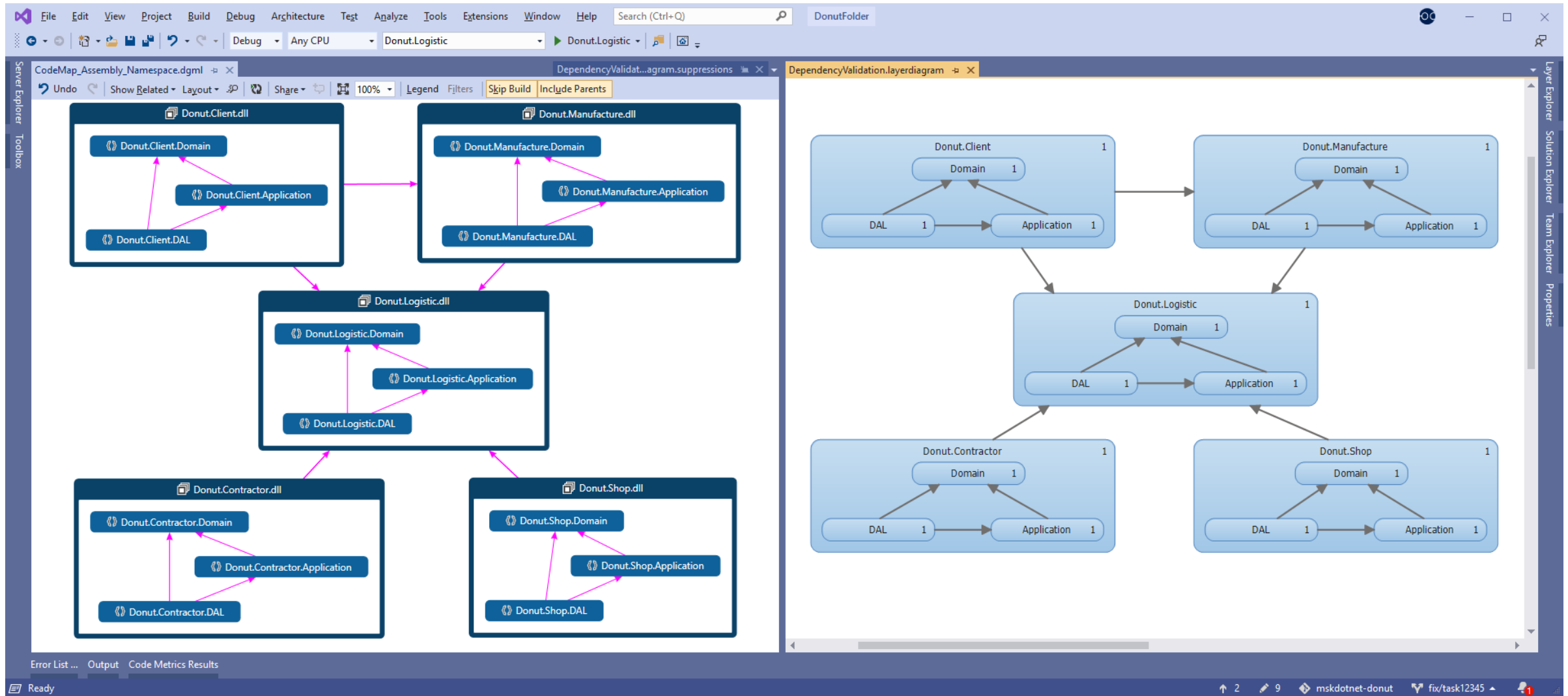
```
let uiTypes = Application.Types.UsingAny(Assemblies.WithNameIn(  
    "PresentationFramework", "System.Windows", "System.Windows.Forms", "System.Web"))  
let dbTypes = ThirdParty.Assemblies.WithNameIn("System.Data",  
    "EntityFramework", "NHibernate").ChildTypes()
```

```
from uiType in uiTypes.UsingAny(dbTypes)  
let dbTypesUsed = dbTypes.Intersect(uiType.TypesUsed)  
let dbTypesAndMembersUsed = dbTypesUsed.Union(  
    dbTypesUsed.ChildMembers().UsedBy(uiType))
```

```
select new {  
    uiType,  
    dbTypesAndMembersUsed,  
    Debt = dbTypesAndMembersUsed.Count().ToMinutes().ToDebt(),  
    Severity = Severity.High  
}
```

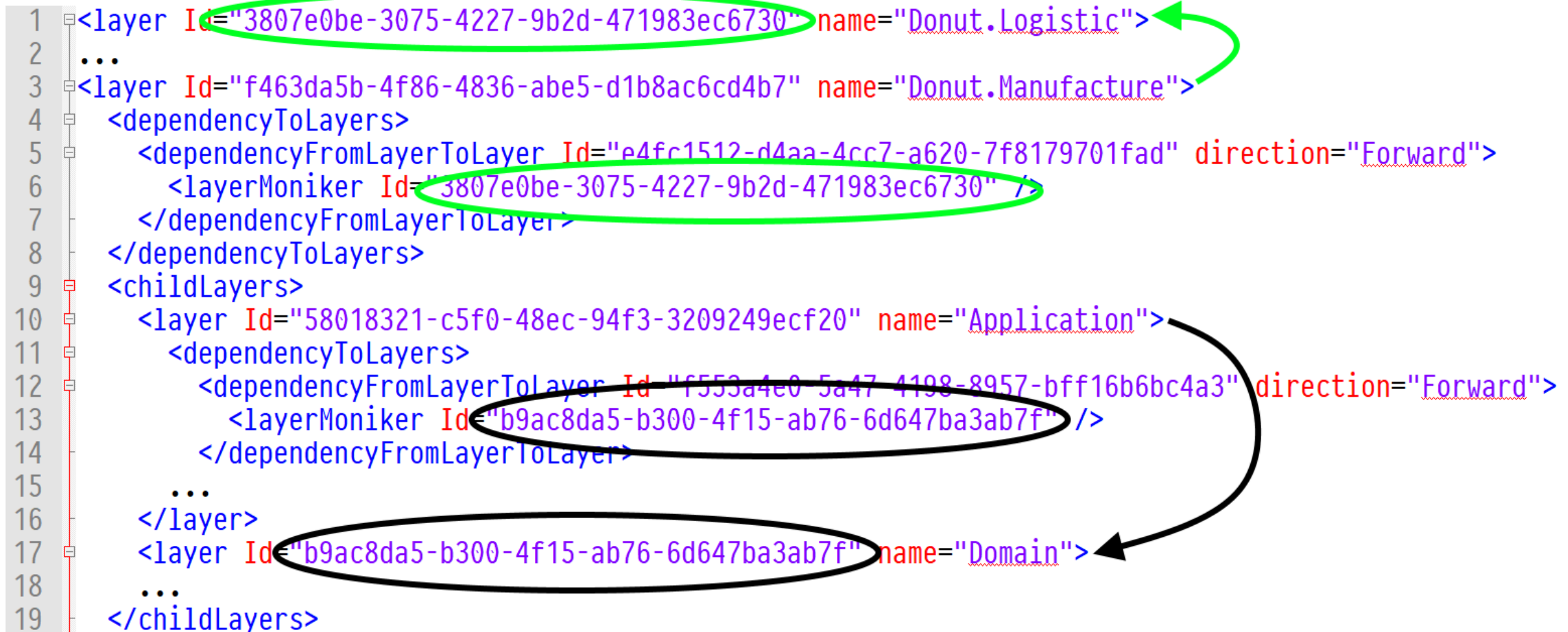
Visual Studio Enterprise

- Низкий порог освоения
- Лучшая интеграция в процесс
- Ограниченные возможности
- *Степень связности внутри и между сборками/пространствами имен



Под капотом

```
1 <layer Id="3807e0be-3075-4227-9b2d-471983ec6730" name="Donut.Logistic">
2 ...
3 <layer Id="f463da5b-4f86-4836-abe5-d1b8ac6cd4b7" name="Donut.Manufacture">
4   <dependencyToLayers>
5     <dependencyFromLayerToLayer Id="e4fc1512-d4aa-4cc7-a620-7f8179701fad" direction="Forward">
6       <layerMoniker Id="3807e0be-3075-4227-9b2d-471983ec6730" />
7     </dependencyFromLayerToLayer>
8   </dependencyToLayers>
9   <childLayers>
10    <layer Id="58018321-c5f0-48ec-94f3-3209249ecf20" name="Application">
11      <dependencyToLayers>
12        <dependencyFromLayerToLayer Id="f553a4e0-5a47-4198-8957-bff16b6bc4a3" direction="Forward">
13          <layerMoniker Id="b9ac8da5-b300-4f15-ab76-6d647ba3ab7f" />
14        </dependencyFromLayerToLayer>
15      ...
16    </layer>
17    <layer Id="b9ac8da5-b300-4f15-ab76-6d647ba3ab7f" name="Domain">
18      ...
19    </childLayers>
```



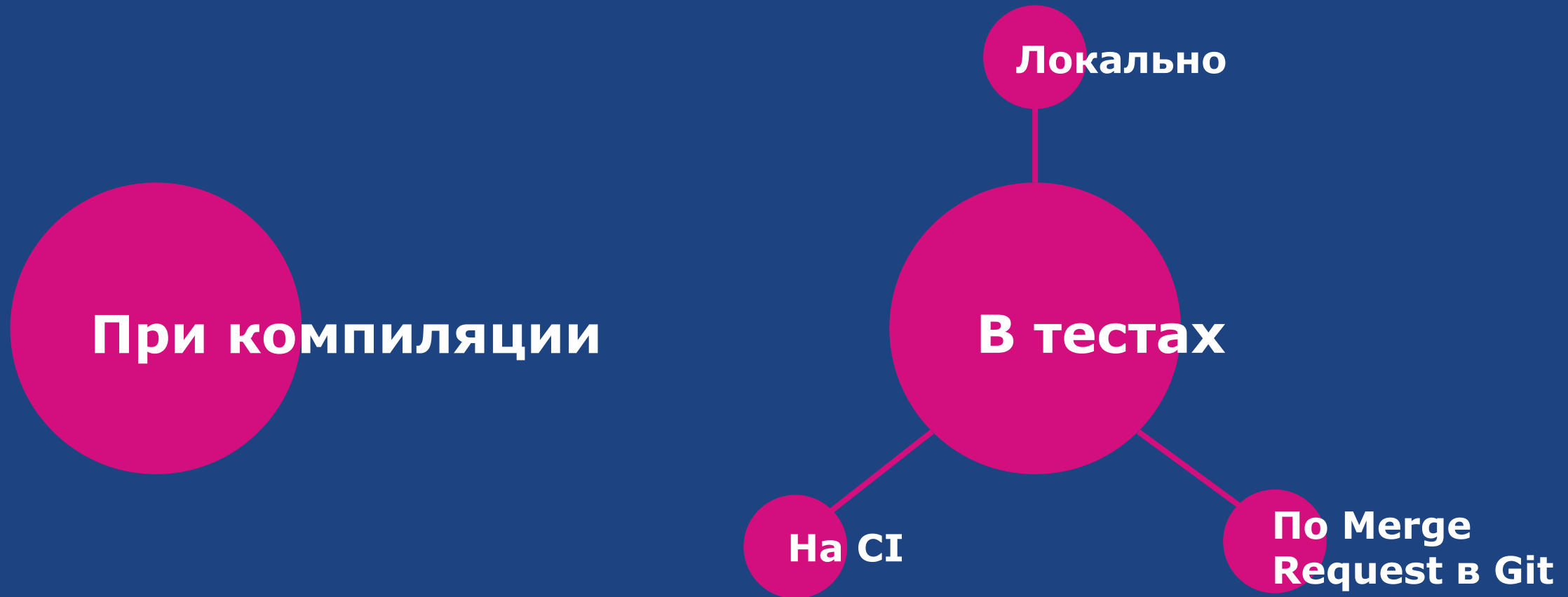


Сравнение

CUSTIS®

	NDepend	VSE	Roslyn
Проверка при компиляции	+/-	+	+
Визуализация	+	+	-
Оценка степени связности	+	+/-	+
Контроль внешних зависимостей	+	+/-	+
Гибкость задания правил	+	-	+

Когда контролировать



Наш опыт

- **Последовательное разделение большого legacy-приложения**
 - Поддержка двумя независимыми командами
 - Контроль только новых проблем
- **Выделение контрактных сборок**

Спасибо за внимание!
Вопросы?

Юрий Солдаткин
Архитектор

