

Объектно-ориентированные CASE-технологии Язык UML.

2. Мета модель языка UML

Романов Владимир Юрьевич,
Московский Государственный Университет им. М.В.Ломоносова
Факультет Вычислительной Математики и Кибернетики
vromanov@cs.msu.su,
romanov.rvy@yandex.ru

UML – Unified Modeling Language.

Унифицированный язык моделирования

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Стандарт на язык моделирования разработанный консорциумом фирм Object Management Group:

<http://www.omg.org>

- Стандартизация языка UML консорциумом OMG:

<http://www.omg.org/uml>

<http://www.uml.org/>

- Текущие версии стандарта доступные для свободного скачивания:

<http://www.omg.org/spec/>

UML – Unified Modeling Language.

Стандарты связанные с языком UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

MODELING AND METADATA SPECIFICATIONS

UML, MOF, CWM, XMI Specifications

SPECIFICATION	acronym	topical area / domain	Document #
Action Language for Foundational UML	ALF	modeling	ptc/2012-08-43
Common Terminology Services 2	cts2	modeling	formal/2012-11-01
Common Warehouse Metamodel	CWM	data warehousing, modeling	formal/2003-03-02
CWM Metadata Interchange Patterns	MIPS	data warehousing, modeling	formal/2004-03-25
Diagram Definition	DD	modeling	formal/12-07-01
Essence - Kernel and Language for Software Engineering Methods	Essence	modeling	ptc/2013-06-08
Interaction Flow Modeling Language	IFML	modeling	ptc/2013-03-08
Meta Object Facility Core	MOF	modeling	formal/2011-08-07
Model Driven Message Interoperability	MDMI	modeling	formal/2010-03-01
Model-level Testing and Debugging	MLTD	modeling	ptc/2007-05-14
MOF Model to Text Transformation Language	MOFM2T	modeling	formal/2008-01-16
MOF Query / View / Transformation	QVT	modeling	formal/2011-01-01
MOF Support for Semantic Structures	SMOF	modeling	formal/2013-04-02
MOF 2 Facility and Object Lifecycle	MOFFOL	modeling	formal/2010-03-04
MOF 2 Versioning and Development Lifecycle	MOFVD	modeling	formal/2007-05-01
Object Constraint Language	OCL	modeling	formal/2012-01-01
OMG Systems Modeling Language	SysML	modeling	formal/2012-06-01
Ontology Definition Metamodel	ODM	modeling	formal/2009-05-01
Reusable Asset Specification	RAS	modeling	formal/2005-11-02
Semantics of a Foundational Subset for Executable UML Models	FUML	modeling	formal/2011-02-01
Service oriented architecture Modeling Language	SoaML	modeling	formal/2012-05-01
Software Process Engineering Metamodel	SPEM	modeling	formal/2008-04-01
Unified Modeling Language	UML	modeling	formal/2011-08-05, formal/2011-08-06
UML Simplified (UML 2.5)	UML	modeling	ptc/2012-10-24
UML Human-Usable Textual Notation	HUTN	modeling	formal/2004-08-01
MOF 2 XMI Mapping	XMI	modeling	formal/2011-08-09

UML – Unified Modeling Language.

Инфраструктура и суперструктура языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Текущая версия языка UML:
<http://www.omg.org/spec/UML/2.4.1/>
- Инфраструктура (база для последующего расширения):
<http://www.omg.org/spec/UML/2.4.1/Infrastructure/PDF>
- Суперструктура (+ элементы для моделирования конструкций языков программирования):
<http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF>
- XML Metadata Interchange (XMI) формат для обмена моделями инструментами:
<http://www.omg.org/spec/XMI/2.4.1/>
- Спецификация суперструктуры в формате XMI
<http://www.omg.org/spec/UML/20110701/Superstructure.xmi>

UML – Unified Modeling Language.

Стандарт для обмена UML-диаграммами

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

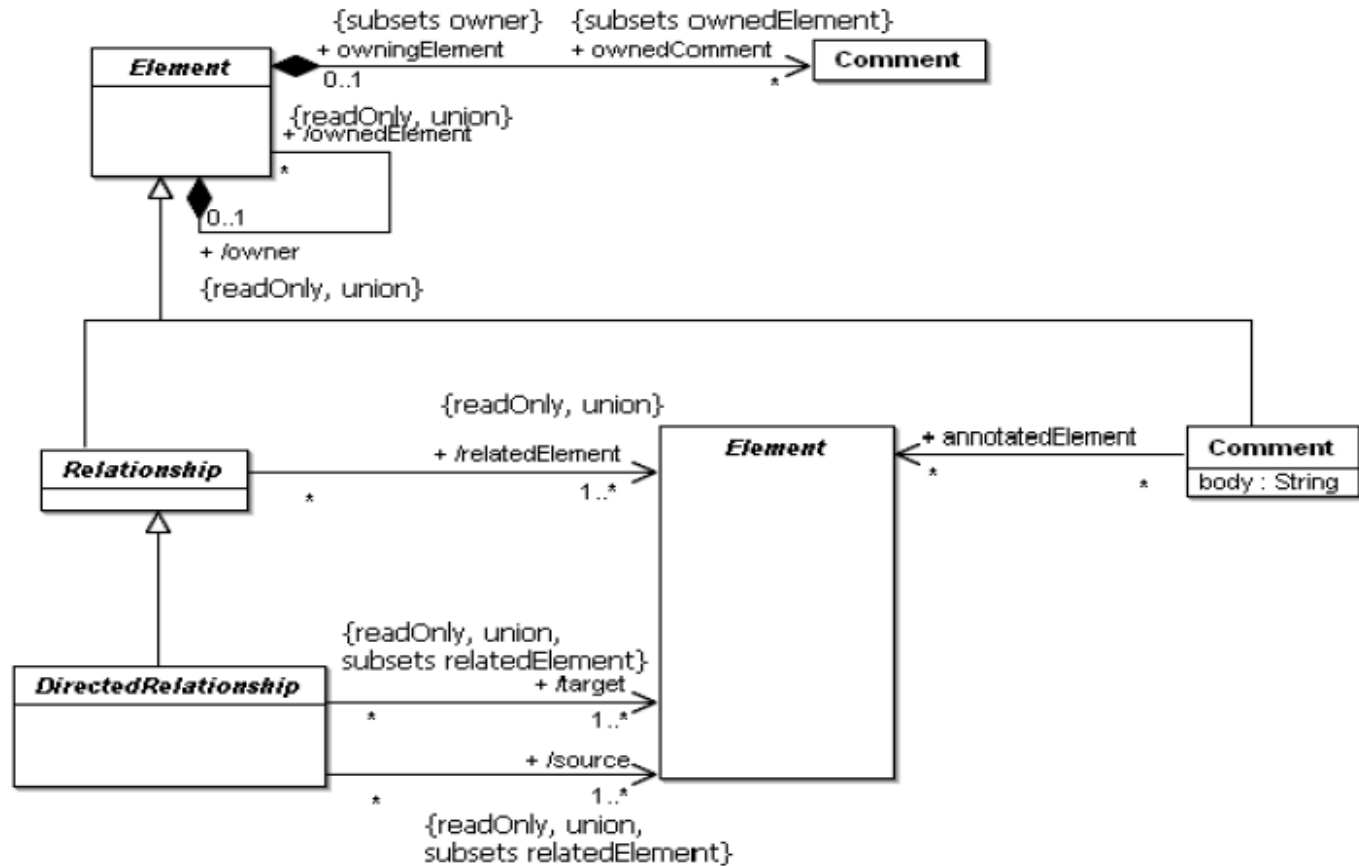
- **Diagram Definition (DD)**
<http://www.omg.org/spec/DD/1.0/>
- XMI of the Diagram Graphics v1.0 package
<http://www.omg.org/spec/DD/20110901/DG.cmoF>
- XMI of the Diagram Interchange v1.0 package
<http://www.omg.org/spec/DD/20110901/DI.cmoF>
- XMI of the Diagram Common v1.0 package
<http://www.omg.org/spec/DD/20110901/DC.cmoF>

UML – Unified Modeling Language.

Фрагмент спецификации метамодели UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



UML – Unified Modeling Language.

Объектный язык ограничений (OCL)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Object Constraint Language (OCL)
<http://www.omg.org/spec/OCL/2.3.1/>
- Спецификация языка OCL на языке XML
<http://www.omg.org/spec/OCL/20090501/OCL.cmo>

UML – Unified Modeling Language.

Фрагмент спецификации метамодели OCL

МГУ им. М.В.Ломоносова. Факультет ВМК.

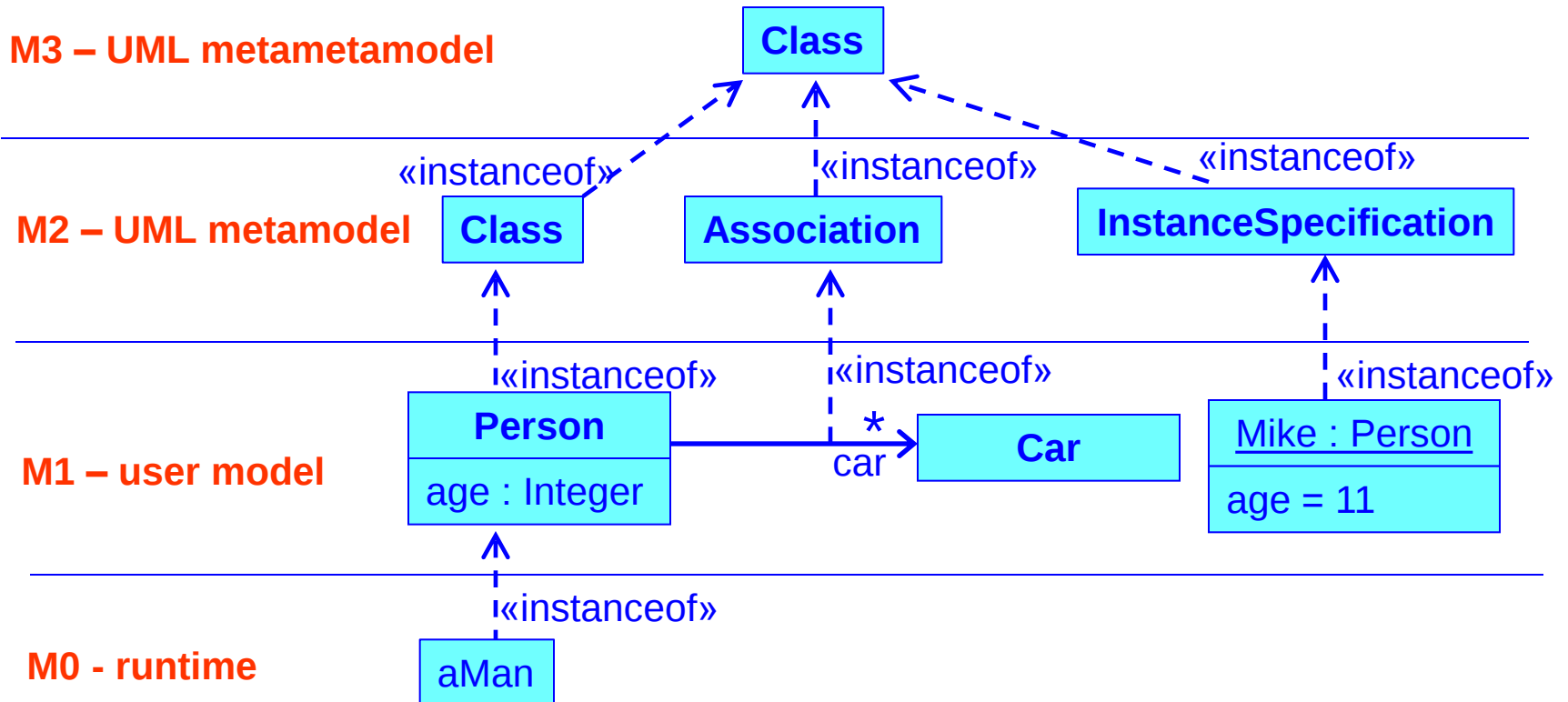
Романов Владимир Юрьевич ©2025

- **if** memberEnd->size() > 2 **then** ownedEnd->*includesAll*(memberEnd)
- parents()->*select*(oclIsKindOf(Association)).oclAsType(Association)-> *forAll*(p |
p.memberEnd->size() = **self**.memberEnd->size())
- *Sequence*{1..**self**.memberEnd->size()}->
 forAll(i |
 self.general->*select*(oclIsKindOf(Association)).oclAsType(Association)
 ->
 forAll(ga |
 self.memberEnd->
 at(i).type.conformsTo(ga.memberEnd->at(i).type)))

4-х уровневая иерархия метамodelей

МГУ им. М.В.Ломоносова. Факультет ВМК.

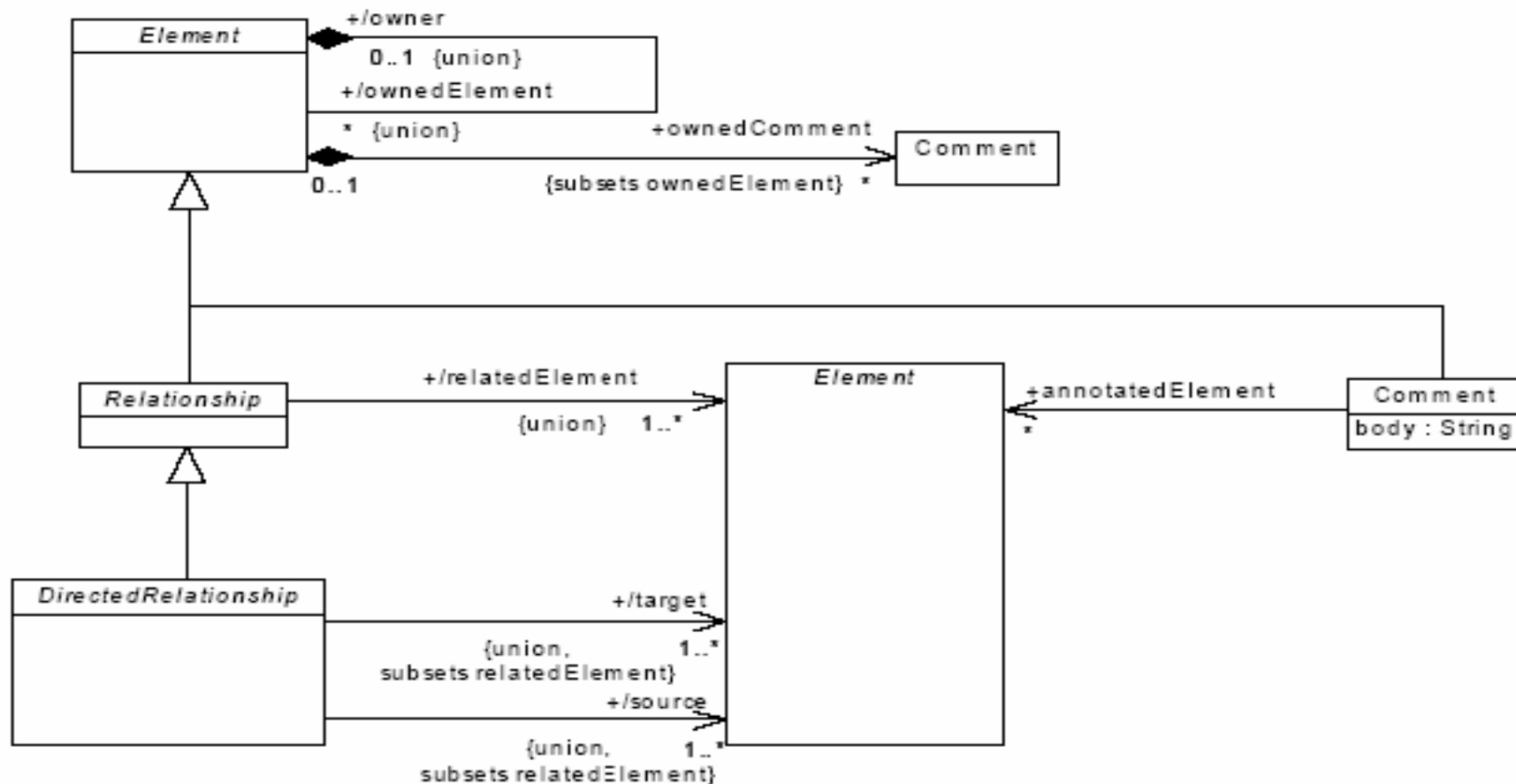
Романов Владимир Юрьевич ©2025



Ядро метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

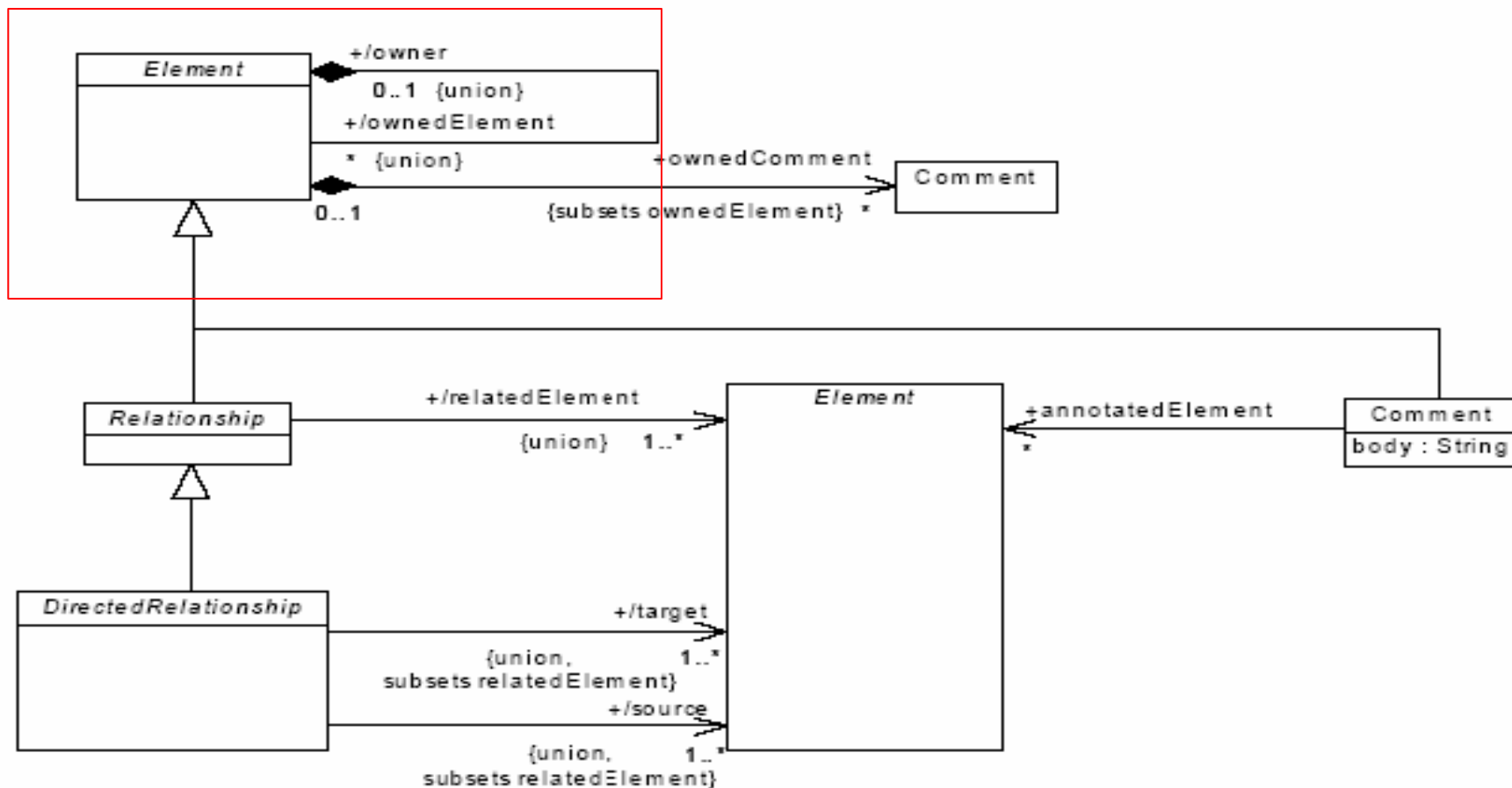
Романов Владимир Юрьевич ©2025



Ядро метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

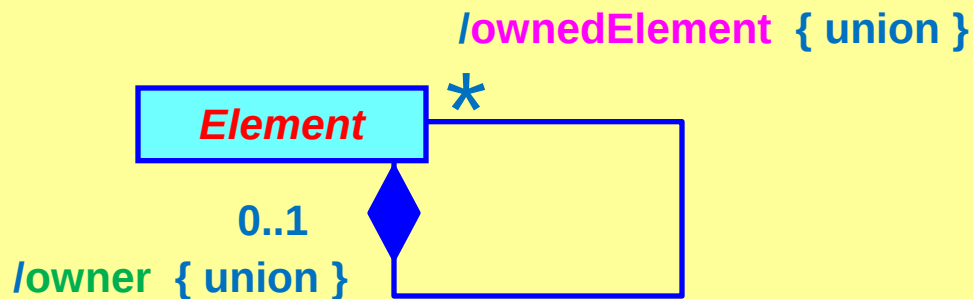
Романов Владимир Юрьевич ©2025



Моделирование отношения ассоциации собственник ----- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Element - абстрактный базовый класс для всех классов метамодели.

Атрибуты и отношения этого класса наследуются всеми классами метамодели.

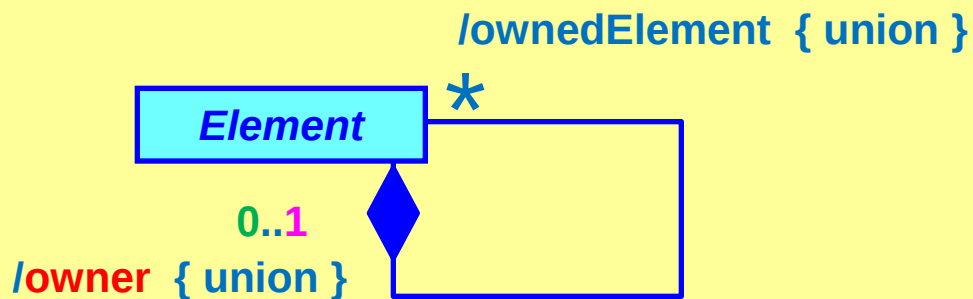
В частности, всеми классами наследуется и отношение ассоциации:
собственник (**owner**) ----- собственность (**ownedElement**)

Моделирование отношения ассоциации собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Окончание ассоциации с ролью **owner**



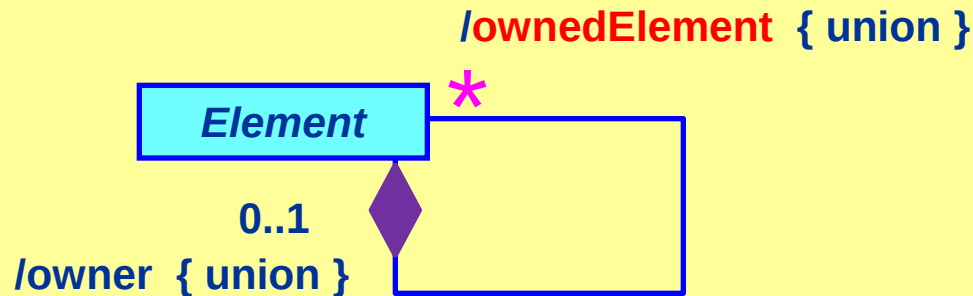
У каждого элемента модели **может быть не более одного** собственника (**owner**).

Моделирование отношения ассоциации собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Окончание ассоциации с ролью **ownedElement**



У каждого элемента модели **может быть** (по умолчанию нижняя граница от нуля) **неограниченное количество** собственных элементов (**ownedElement**).

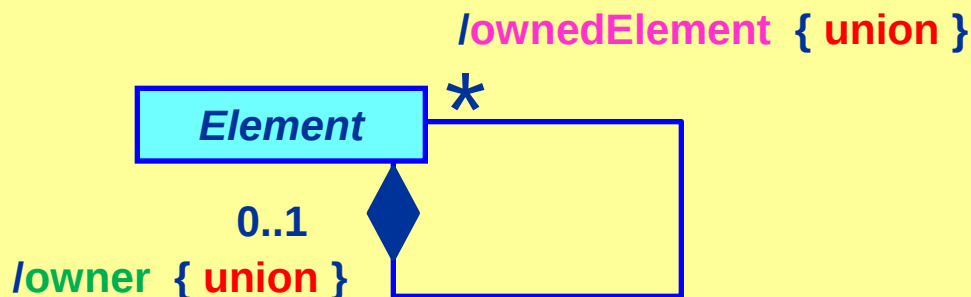
Время жизни собственного элемента (**ownedElement**) **совпадает** с временем жизни элемента-собственника (**owner**).

Моделирование отношения ассоциации собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Свойство **union** на окончаниях ассоциации



В классах-потомках класса **Element** могут быть свойства этих классов:

{ subsets owner }

и / или

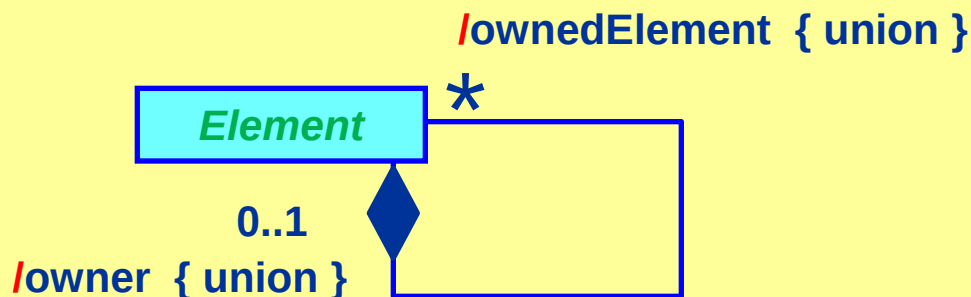
{ subsets ownedElement }

Моделирование отношения ассоциации собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Свойство / (*derived*) на окончаниях ассоциации

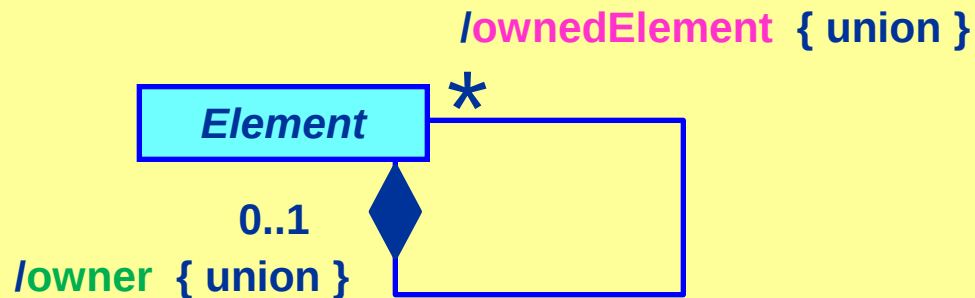


Свойство / (*derived*) на окончаниях ассоциации означает, что собственники и собственные элементы не хранятся в экземпляре класса **Element**, а получаются *объединением (union)* элементов в классах-потомках класса **Element**.

Что моделирует отношение ассоциации собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
// Java
package casetool.graph;

public class Node {}
```

```
// Java
package casetool.graph;
```

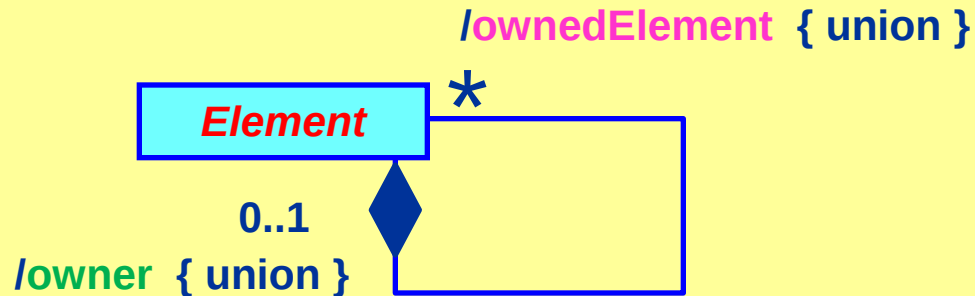
```
// Java
package casetool.graph;

public class Node {
    String name;
}
```

Программный интерфейс для отношения собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
public static void dump(Element element, int level) {
    String blanks = String.format("%" + 5*level + "s" , " ");

    for (Element owned : element.getOwnedElements()) {
        System.out.println(blanks + owned);

        if (!owned.getOwnedElements().isEmpty())
            dump(owned, level + 1);
    }
}
```

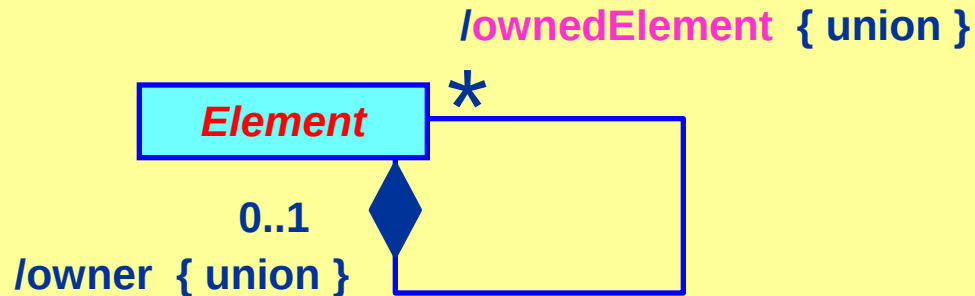
Дамп UML-модели на консоль

```
org.eclipse.uml2.uml.internal.impl.PackageImpl@7bfd270d (name: signature, visibility:
<unset>) (URI: null)
    org.eclipse.uml2.uml.internal.impl.ClassImpl@290f0613 (name:
SignatureReader, visibility: <unset>) (isLeaf: false, isAbstract: false,
isFinalSpecialization: false) (isActive: false)
        org.eclipse.uml2.uml.internal.impl.ElementImportImpl@22a79bc
(alias: <unset>, visibility: public)
            org.eclipse.uml2.uml.internal.impl.PropertyImpl@61d729ab (name:
a, visibility: private) (isLeaf: true) (isStatic: false) (isOrdered: false, isUnique: true,
isReadOnly: false) (aggregation: none, isDerived: false, isDerivedUnion: false, isID: false)
                org.eclipse.uml2.uml.internal.impl.OperationImpl@7c6c1995
(name: SignatureReader, visibility: public) (isLeaf: false, isStatic: false, concurrency:
sequential, isAbstract: false) (isQuery: false)
                    org.eclipse.uml2.uml.internal.impl.ParameterImpl@1d03c504
(name: p0, visibility: <unset>) (isOrdered: false, isUnique: true, direction: in, effect:
<unset>, isException: false, isStream: false)
                        org.eclipse.uml2.uml.internal.impl.OperationImpl@627b987d
(name: accept, visibility: public) (isLeaf: false, isStatic: false, concurrency: sequential,
isAbstract: false) (isQuery: false)
                            org.eclipse.uml2.uml.internal.impl.ParameterImpl@2058690e
(name: p0, visibility: <unset>) (isOrdered: false, isUnique: true, direction: in, effect:
<unset>, isException: false, isStream: false)
```

Программный интерфейс для отношения собственник ---- собственность

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
public static void dump(Element element, int level) {
    String blanks = String.format("%" + 5*level + "s", " ");

    for (Element owned : element.getOwnedElements()) {
        System.out.println(blanks + owned.getClass());

        if (!owned.getOwnedElements().isEmpty())
            dump(owned, level + 1);
    }
}
```

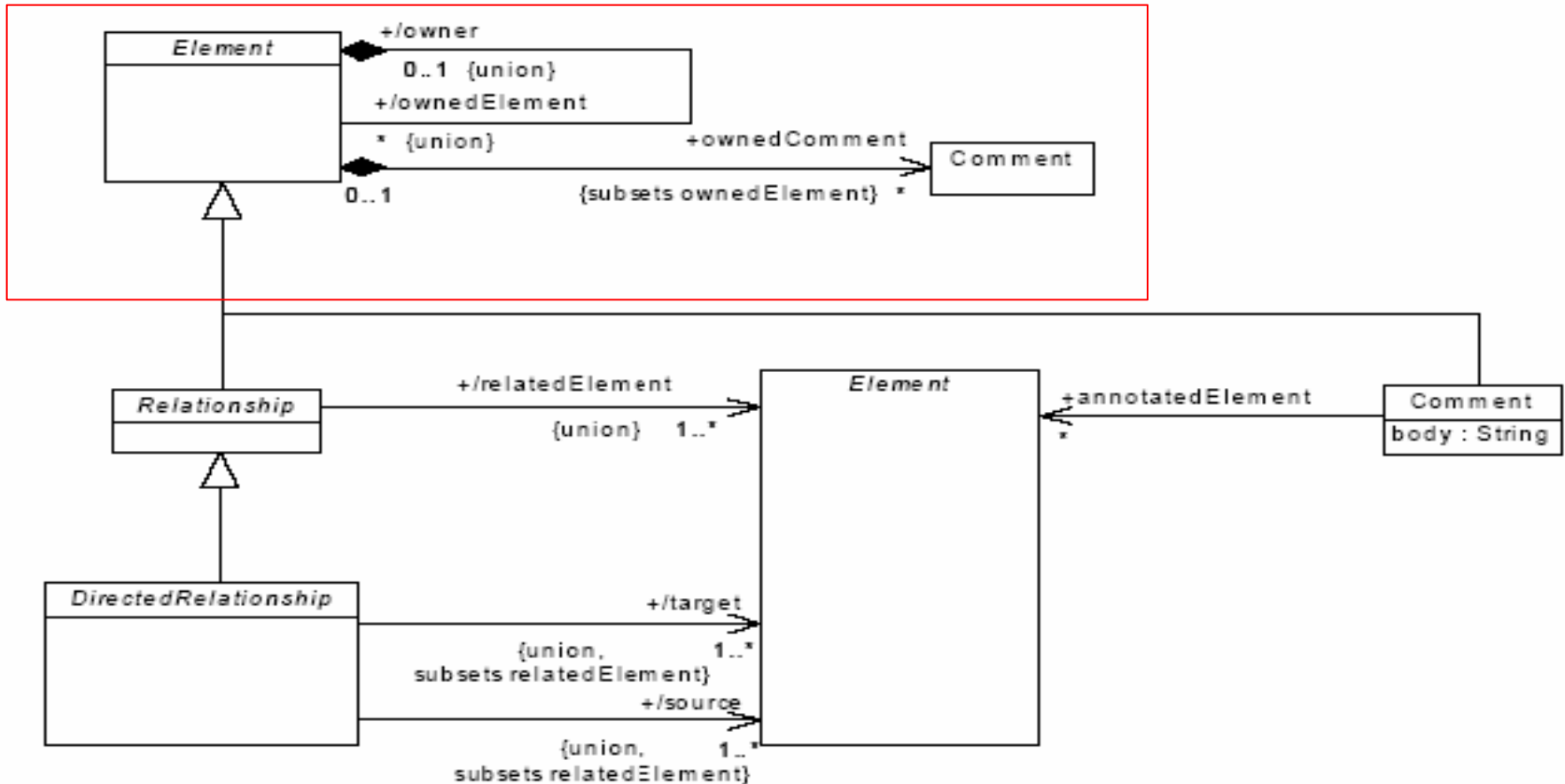
Дамп UML-модели на консоль

```
class org.eclipse.uml2.uml.internal.impl.PackageImpl
  class org.eclipse.uml2.uml.internal.impl.PackageImpl
    class org.eclipse.uml2.uml.internal.impl.PackageImpl
      class org.eclipse.uml2.uml.internal.impl.InterfaceImpl
      class org.eclipse.uml2.uml.internal.impl.InterfaceImpl
    class org.eclipse.uml2.uml.internal.impl.ClassImpl
    class org.eclipse.uml2.uml.internal.impl.ClassImpl
    class org.eclipse.uml2.uml.internal.impl.ClassImpl
    class org.eclipse.uml2.uml.internal.impl.InterfaceImpl
```

Ядро метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

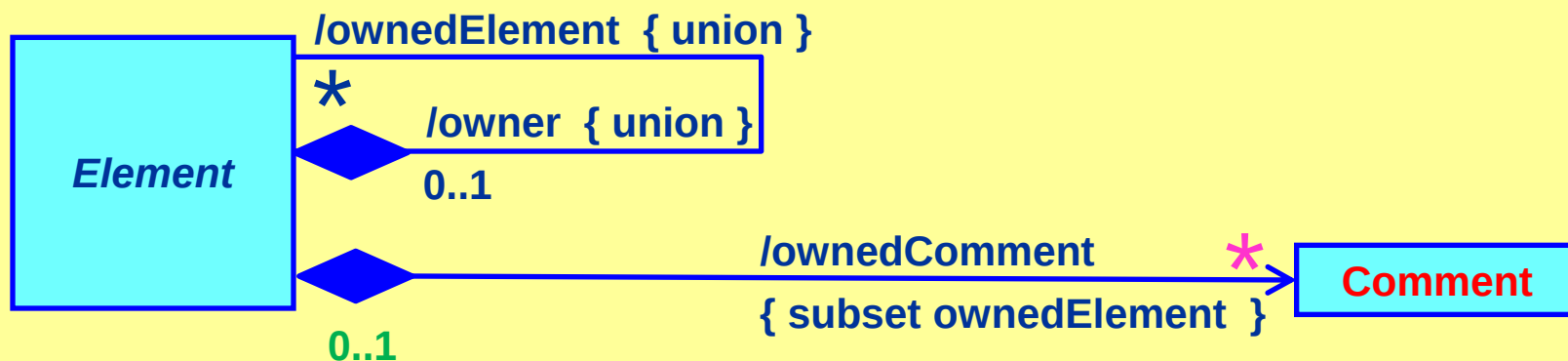
Романов Владимир Юрьевич ©2025



Моделирование комментариев к элементам модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Comment - конкретный класс для комментирования экземпляров всех классов метамодели.

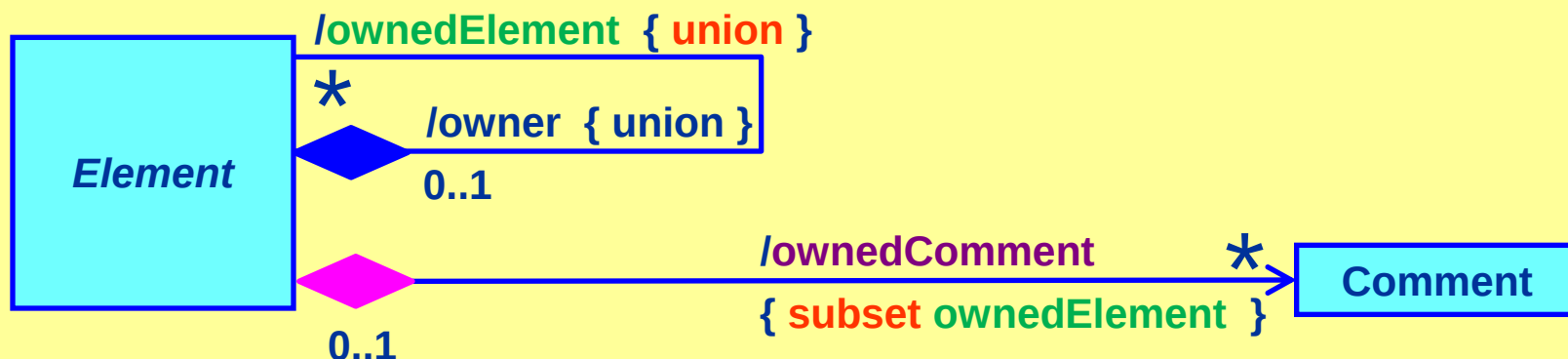
У каждого экземпляра элемента модели может быть **неограниченное количество** экземпляров комментариев.

Каждый комментарий может относиться **не более чем к одному** элементу модели.

Моделирование комментариев к элементам модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Время жизни комментария **совпадает** со временем жизни комментируемого элемента.

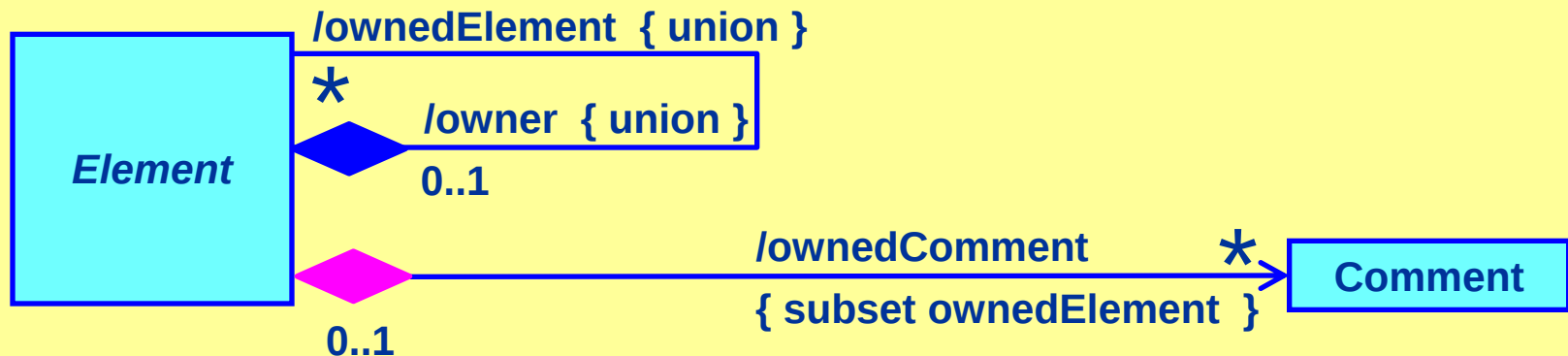
Собственный комментарий (**ownedComment**) является подмножеством (**subset**) объединения (**union**) **ownedElement**.

Это означает, что при запросе с помощью метода **getOwnedElements()** всех элементов, которыми владеет данный элемент, в объединение **ownedElement** попадут и все комментарии для этого элемента.

Моделирование комментариев к элементам модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



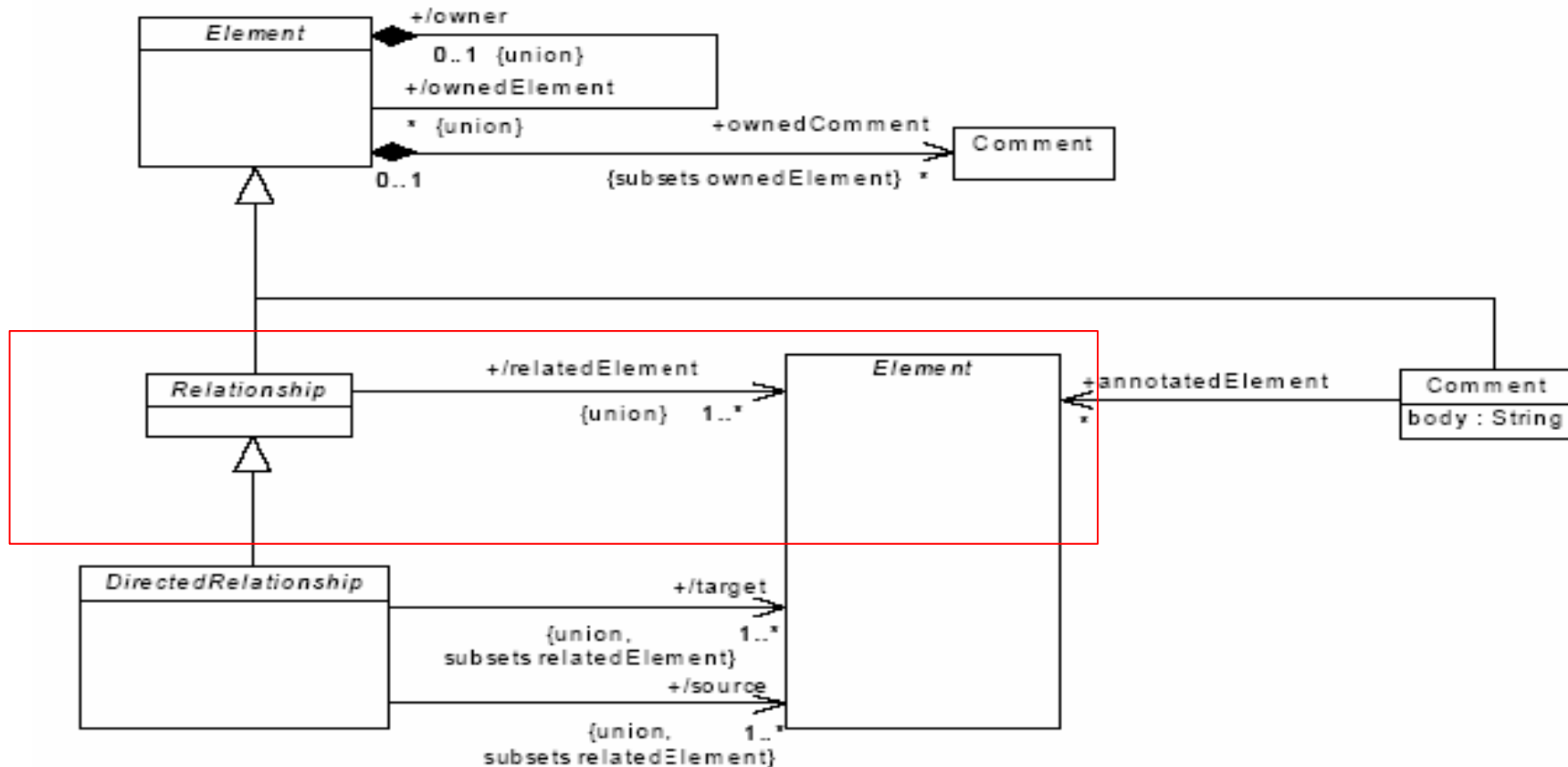
Время жизни комментария **совпадает** со временем жизни комментируемого элемента.

Поэтому для каждого элемента модели существует метод **`createOwnedComment()`**.

Программный интерфейс для работы с комментариями элементов модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

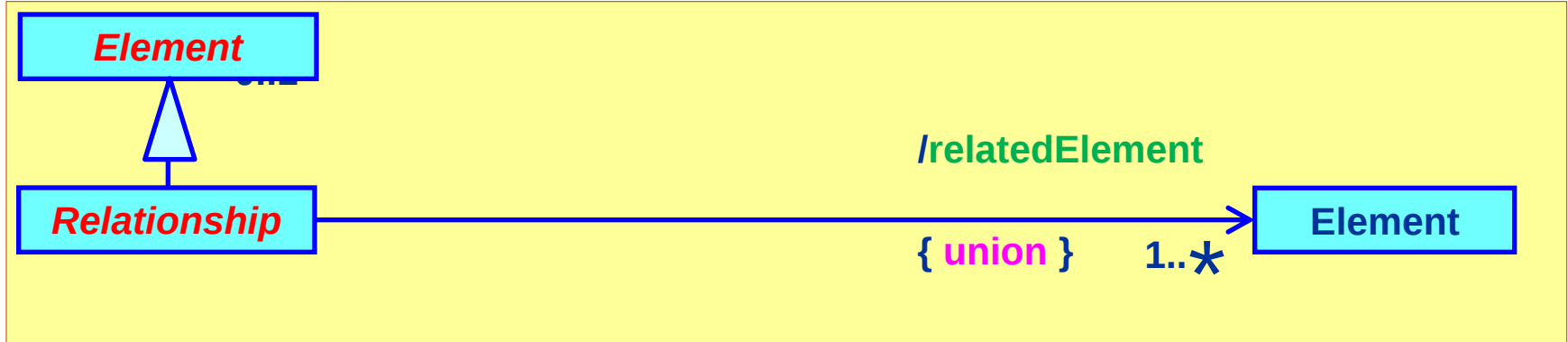
Романов Владимир Юрьевич ©2025



Моделирование отношений между элементами UML-модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Relationship – абстрактный базовый класс для всех отношений в UML- модели.

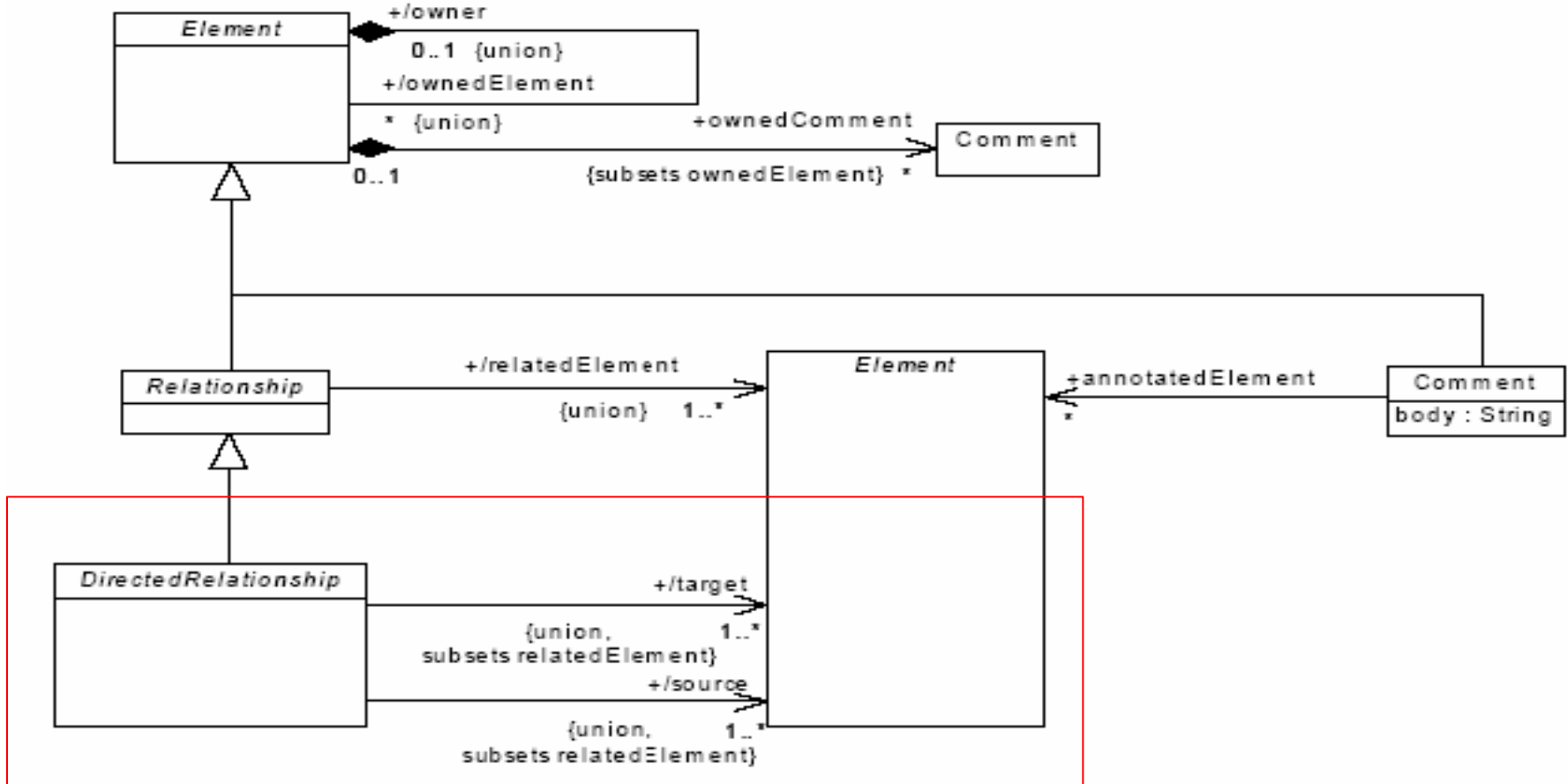
Каждое отношение может связывать 1 и более элементов. Если множественность 1, то отношение элемента с сами собой.

union – означает, что можно считать все связанные отношением элементы модели с помощью метода **getRelatedElement()**.

Программный интерфейс для работы с комментариями элементов модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

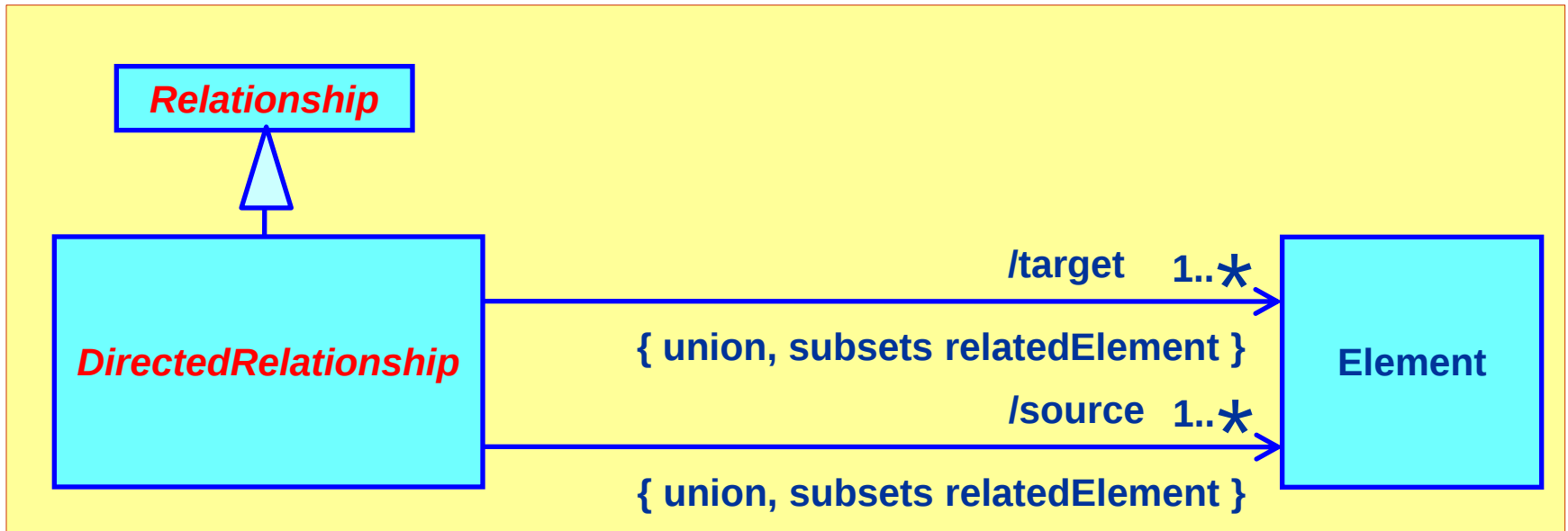
Романов Владимир Юрьевич ©2025



Моделирование отношений между элементами UML-модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

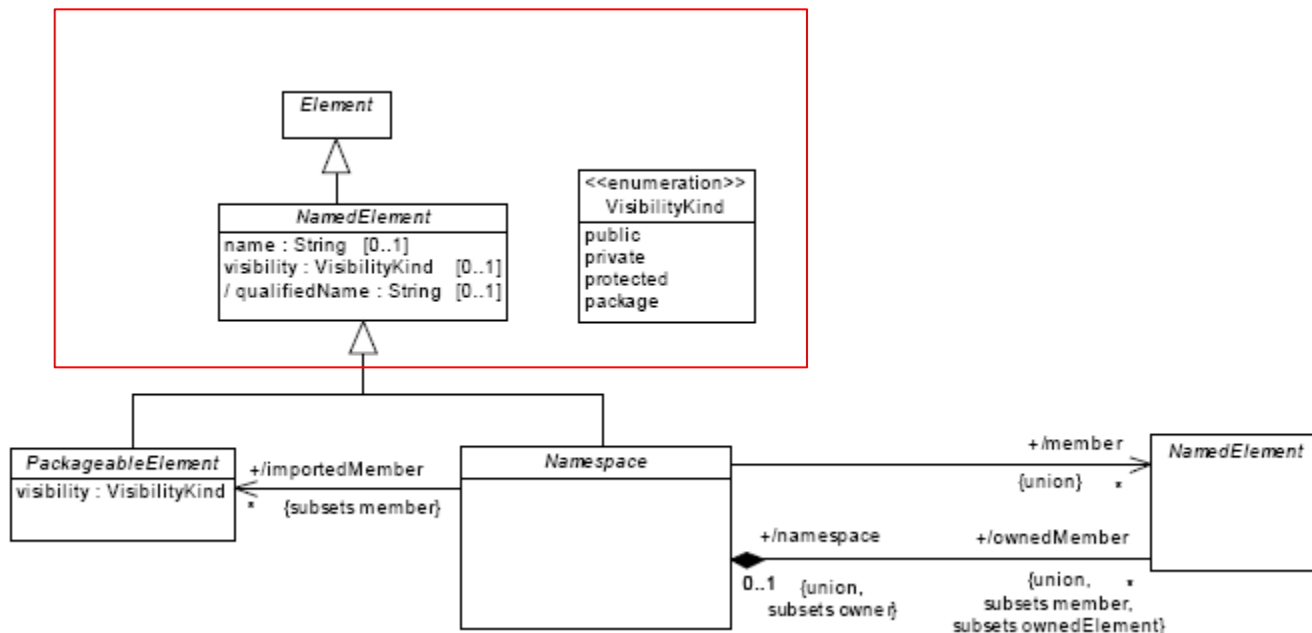


DirectedRelationship – абстрактный базовый класс для всех направленных отношений в UML- модели.

Представление пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

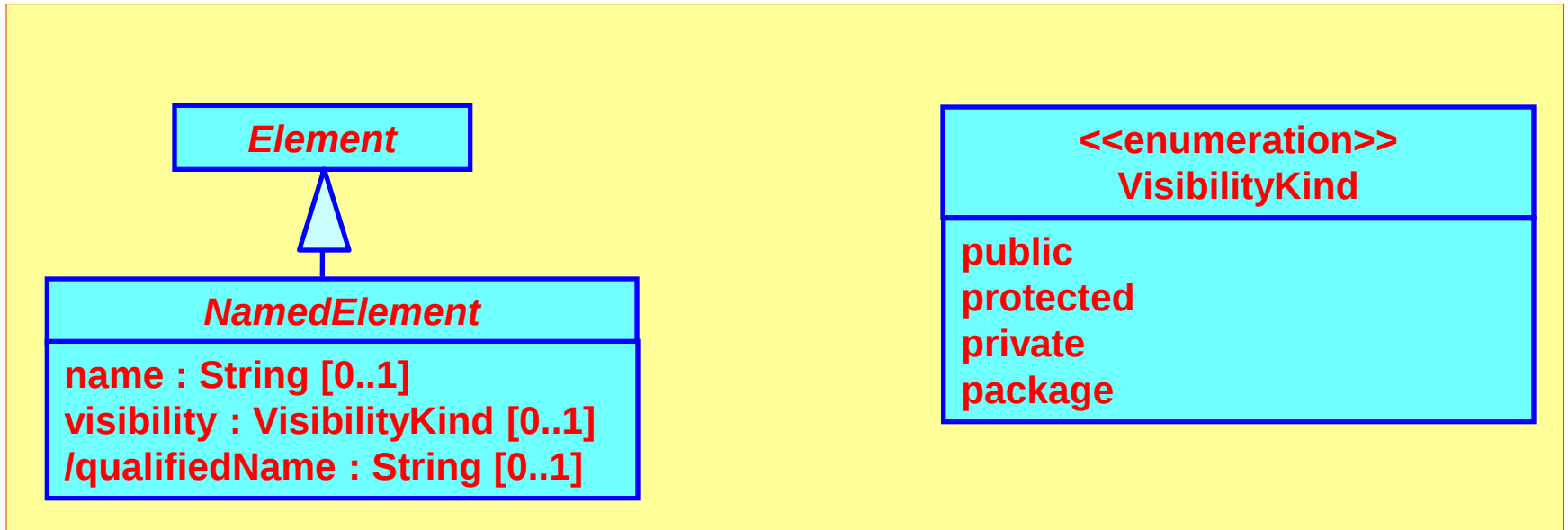
Романов Владимир Юрьевич ©2025



Моделирование именованных элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



/qualifiedName — означает, что интерфейсе есть метод **getQualifiedName()** позволяющий считать квалифицированное имя именованного элемента модели (имя состоящее из цепочки охватывающих именованный элемент пространств имен).

Вывод именованных элементов модели

```
public static void dumpNamed (Element element, int level) {
    if (!(element instanceof NamedElement)) return;

    NamedElement ne = (NamedElement) element;

    String kind = "";
    if (ne instanceof Class)    kind = "class";
    if (ne instanceof Interface) kind = "interface";
    if (ne instanceof Package)  kind = "package";

    if (!kind.isEmpty())
        out.format("%10s: %s %n", kind, ne.getQualified_name());

    for (Element owned : ne.getOwnedElements()) {
        if (!(owned instanceof NamedElement))
            continue;

        if (!owned.getOwnedElements().isEmpty())
            dumpNamed(owned, level + 1);
    }
}
```

Вывод именованных элементов модели

```
package: Guava
  package: Guava::com
    package: Guava::com::google
      package: Guava::com::google::common
        package: Guava::com::google::common::annotations
          interface: Guava::com::google::common::annotations::Beta
          interface: Guava::com::google::common::annotations::GwtCompatible
          interface: Guava::com::google::common::annotations::GwtIncompatible
          interface: Guava::com::google::common::annotations::VisibleForTesting
            package: Guava::com::google::common::base
              class: Guava::com::google::common::base::Absent
              class: Guava::com::google::common::base::Optional
              class: Guava::com::google::common::base::AbstractIterator$1
              class: Guava::com::google::common::base::AbstractIterator
              class: Guava::com::google::common::base::Ascii
```

Вывод классификаторов в поток вывода

```
public static void dumpClassifiers (Element element) {
    if (!(element instanceof NamedElement)) return;
    NamedElement ne = (NamedElement) element;

    String kind = "";
    if (ne instanceof Class)    kind = "class";
    if (ne instanceof Interface) kind = "interface";
    if (ne instanceof Enumeration) kind = "enum";

    if (!kind.isEmpty())
        out.format("%s %s { %n} %n%n", kind, ne.getName());

    for (Element owned : ne.getOwnedElements()) {
        if (!(owned instanceof NamedElement))
            continue;

        if (!owned.getOwnedElements().isEmpty())
            dumpClassifiers(owned);
    }
}
```

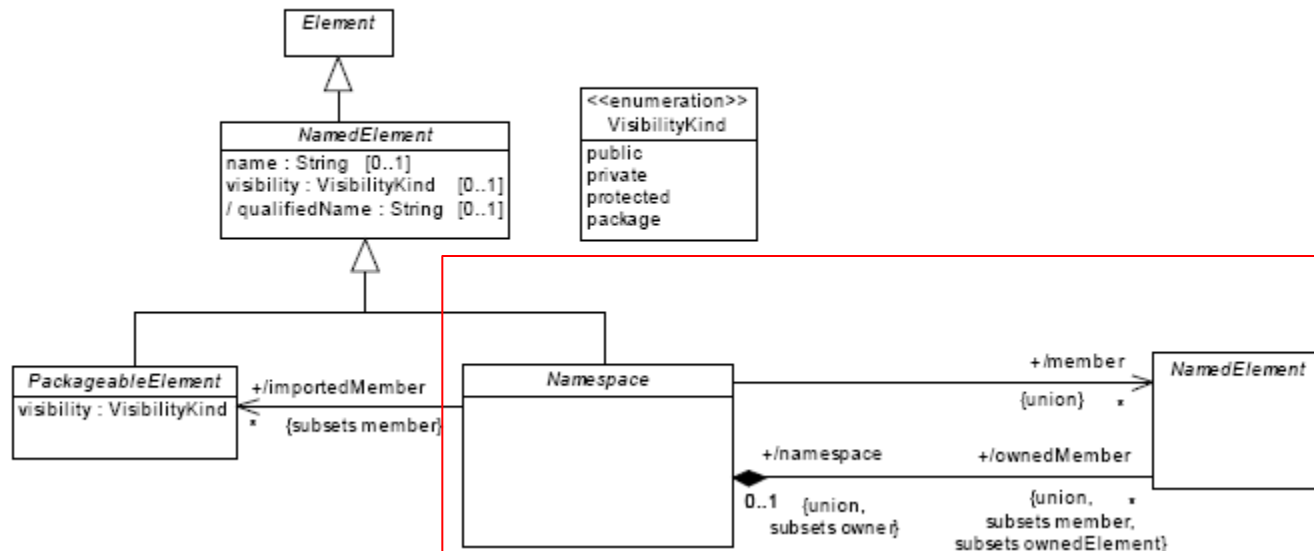
Вывод классификаторов модели

```
interface Beta {  
}  
  
interface GwtCompatible {  
}  
  
interface GwtIncompatible {  
}  
  
interface VisibleForTesting {  
}  
  
class Absent {  
}  
  
class Optional {  
}  
  
class AbstractIterator$1 {  
}  
  
enum AbstractIterator$State {  
}
```

Представление пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

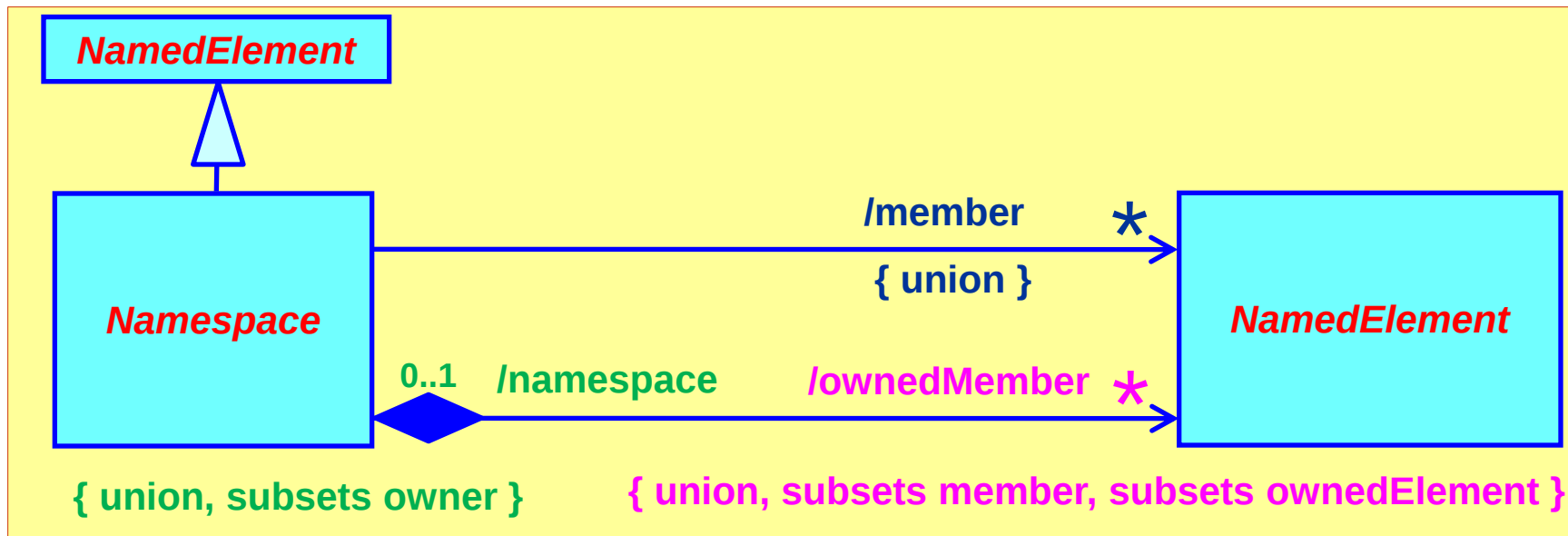
Романов Владимир Юрьевич ©2025



Моделирование пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



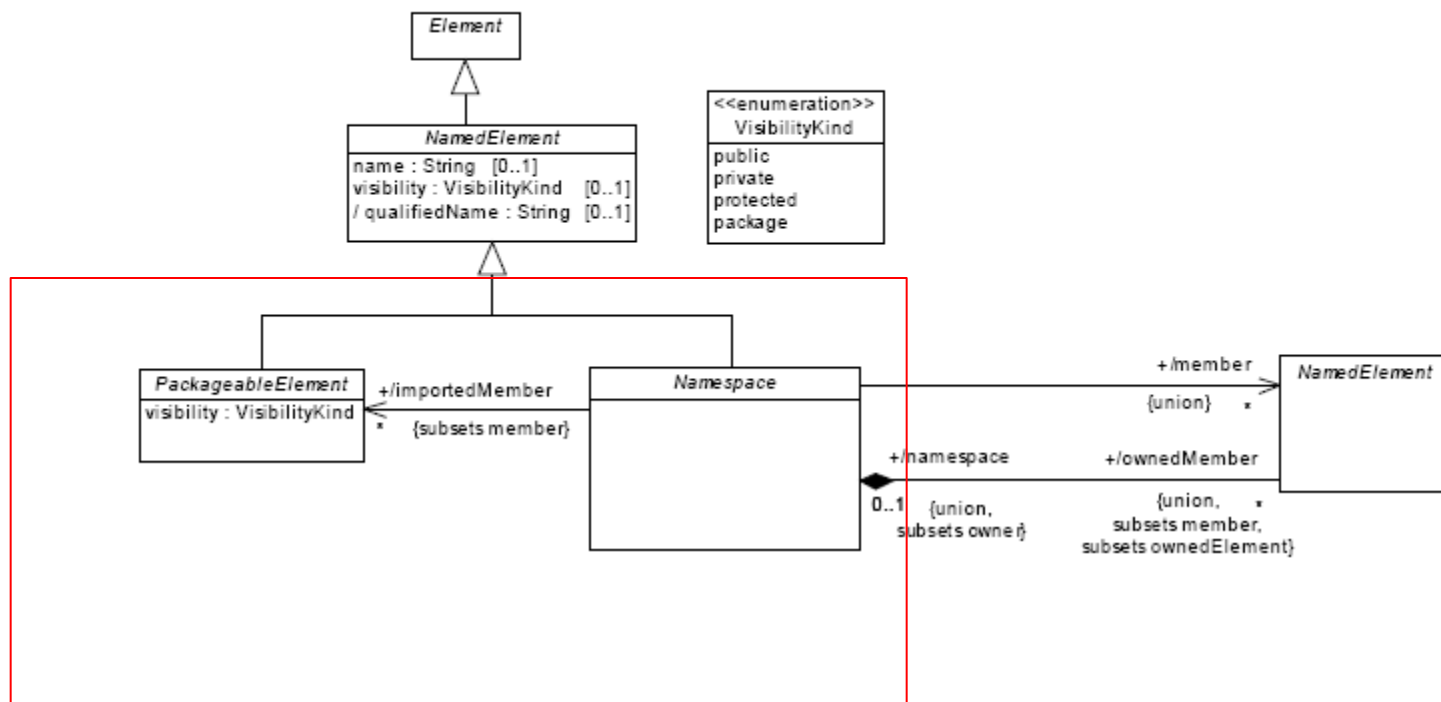
/member – собственные элементы, идентифицируемые элементы, импортированные элементы и унаследованные элементы

/ownedMember – собственные элементы

Представление пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

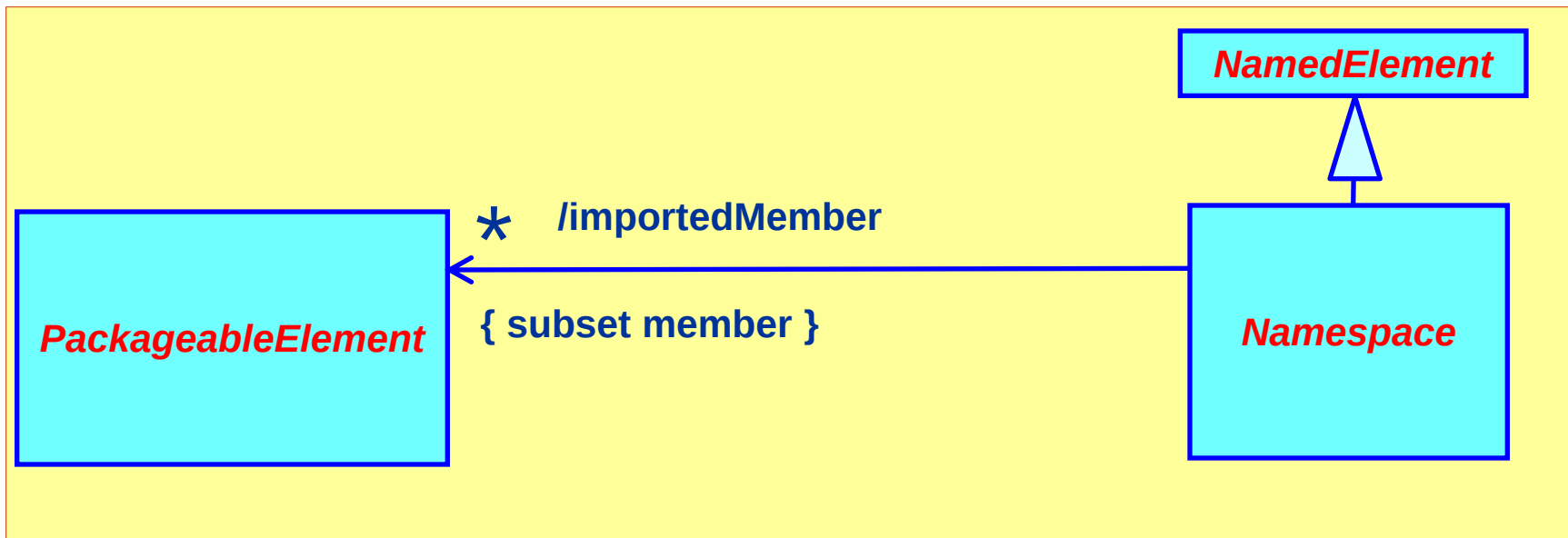
Романов Владимир Юрьевич ©2025



Моделирование пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



/importedMember – импортированные элементы:

- импортированные по отдельности (**ElementImport**)
- импортированные часть импорта всего пакета (**PackageImport**)

Вывод пакета в поток вывода

```
public static void dumpPackages (Element element) {  
    if (!(element instanceof Package)) return;  
    Package p = (Package) element;  
  
    out.format("package %s { %n", p.getName());  
  
    for (NamedElement member : p.getOwnedMembers())  
        dumpClassifier(member);  
  
    for (NamedElement member : p.getOwnedMembers()) {  
        if (!(member instanceof Package))  
            continue;  
  
        if (!member.getOwnedElements().isEmpty())  
            dumpPackages(member);  
    }  
    out.format("} %n%n");  
}
```

Вывод классификатора в поток вывода

```
private static void dumpClassifier (NamedElement ne) {  
    String kind = "";  
    if (ne instanceof Class)    kind = "class";  
    if (ne instanceof Interface) kind = "interface";  
    if (ne instanceof Enumeration) kind = "enum";  
  
    if (!kind.isEmpty())  
        out.format(" %s %s { %n } %n", kind, ne.getName());  
}
```

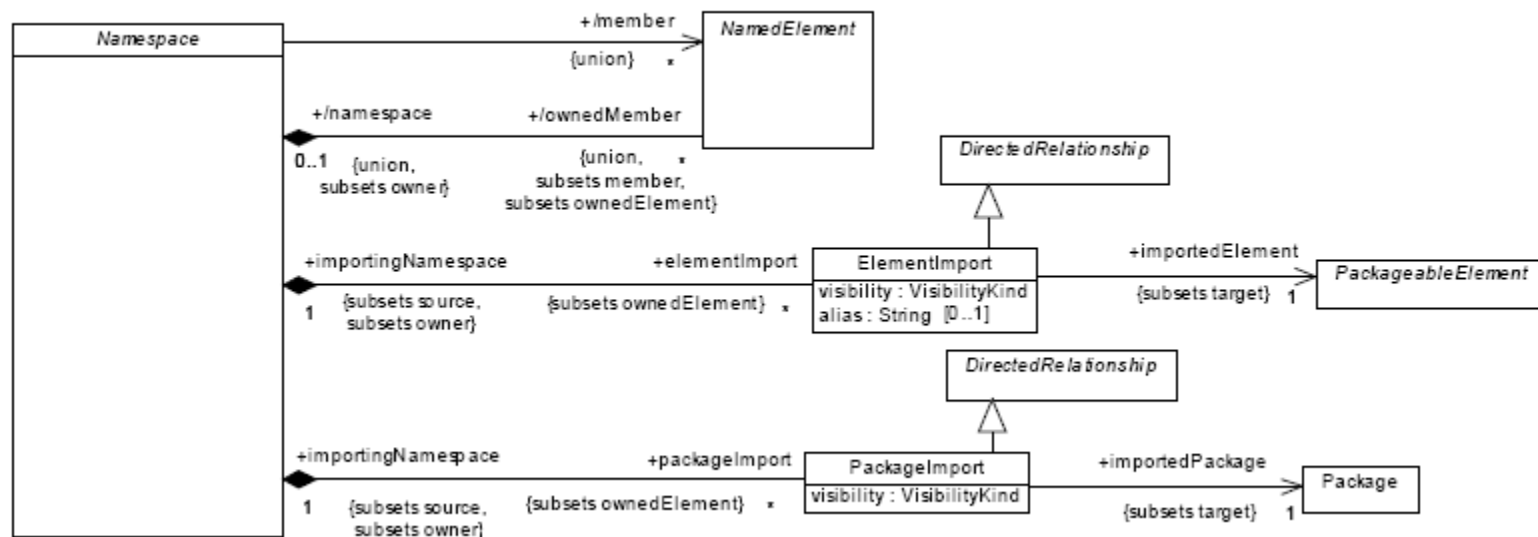
Вывод пакетов и классификаторов модели

```
package Guava {  
  package com {  
    package google {  
      package common {  
        package annotations {  
          interface Beta {  
          }  
          interface GwtCompatible {  
          }  
          interface GwtIncompatible {  
          }  
          interface VisibleForTesting {  
          }  
        }  
      }  
    }  
  }  
  package base {  
    class Absent {  
    }  
    class Optional {  
    }  
  }  
}
```

Импорт в пространства имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

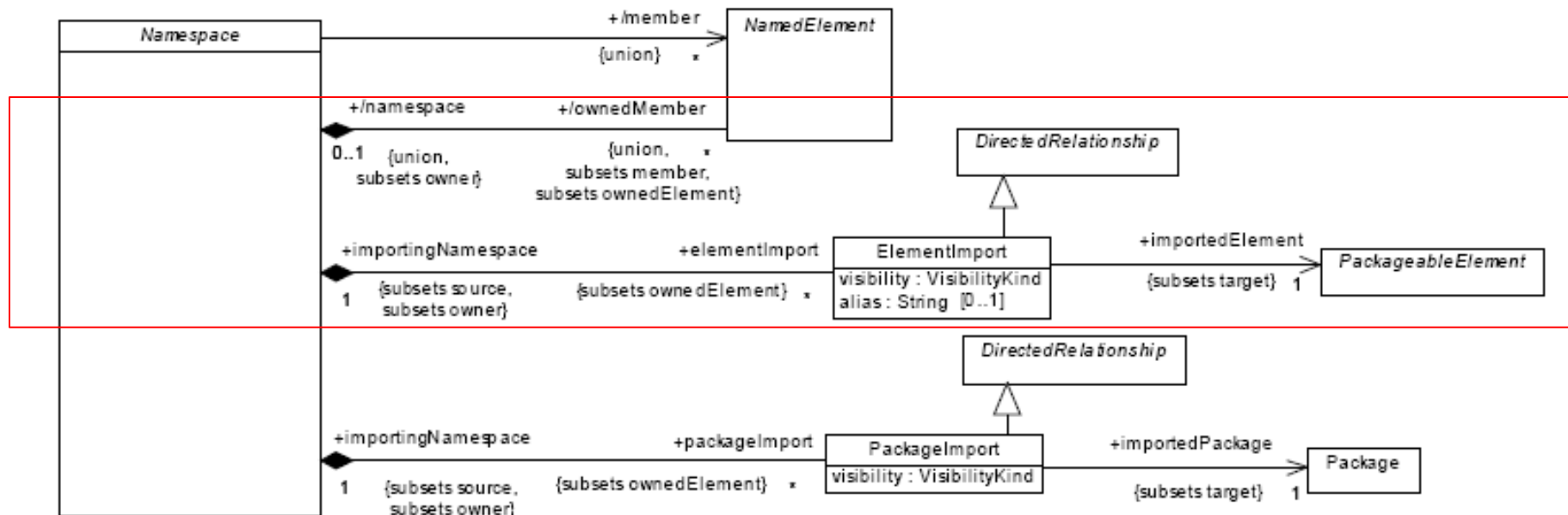
Романов Владимир Юрьевич ©2025



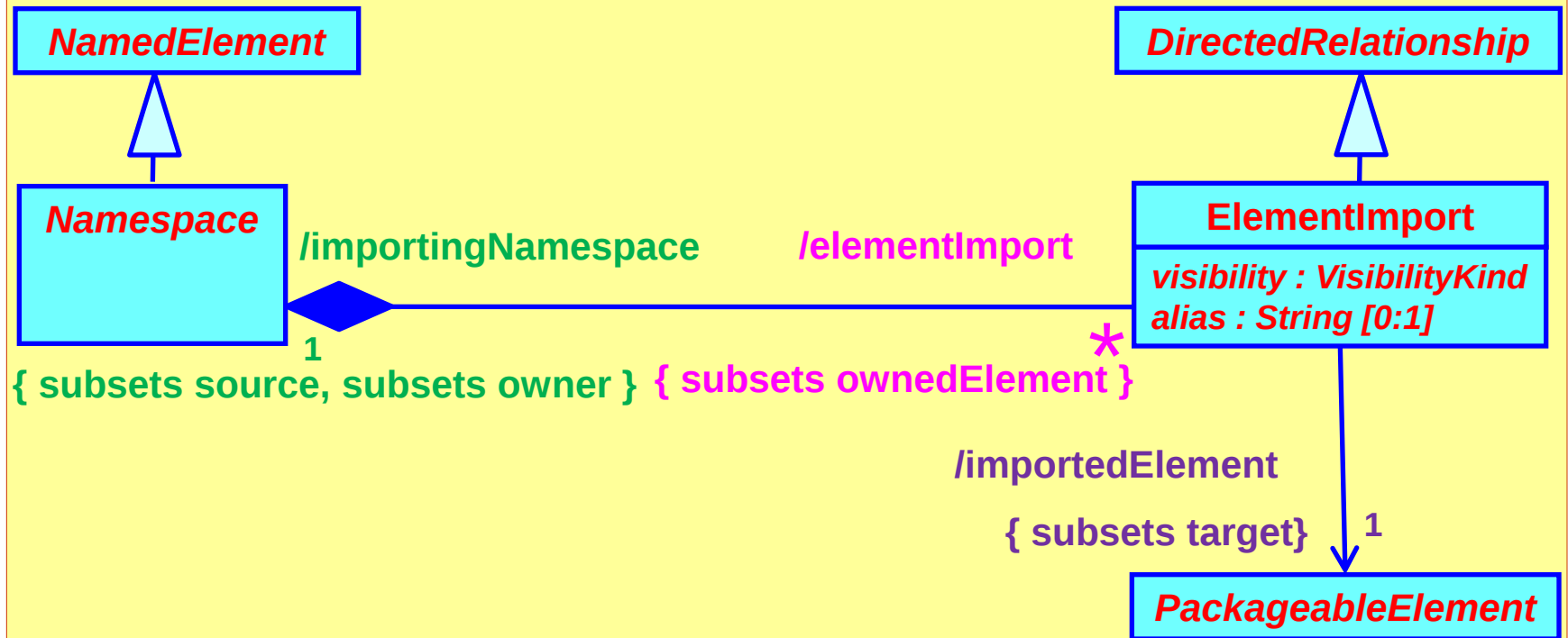
Импорт в пространства имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



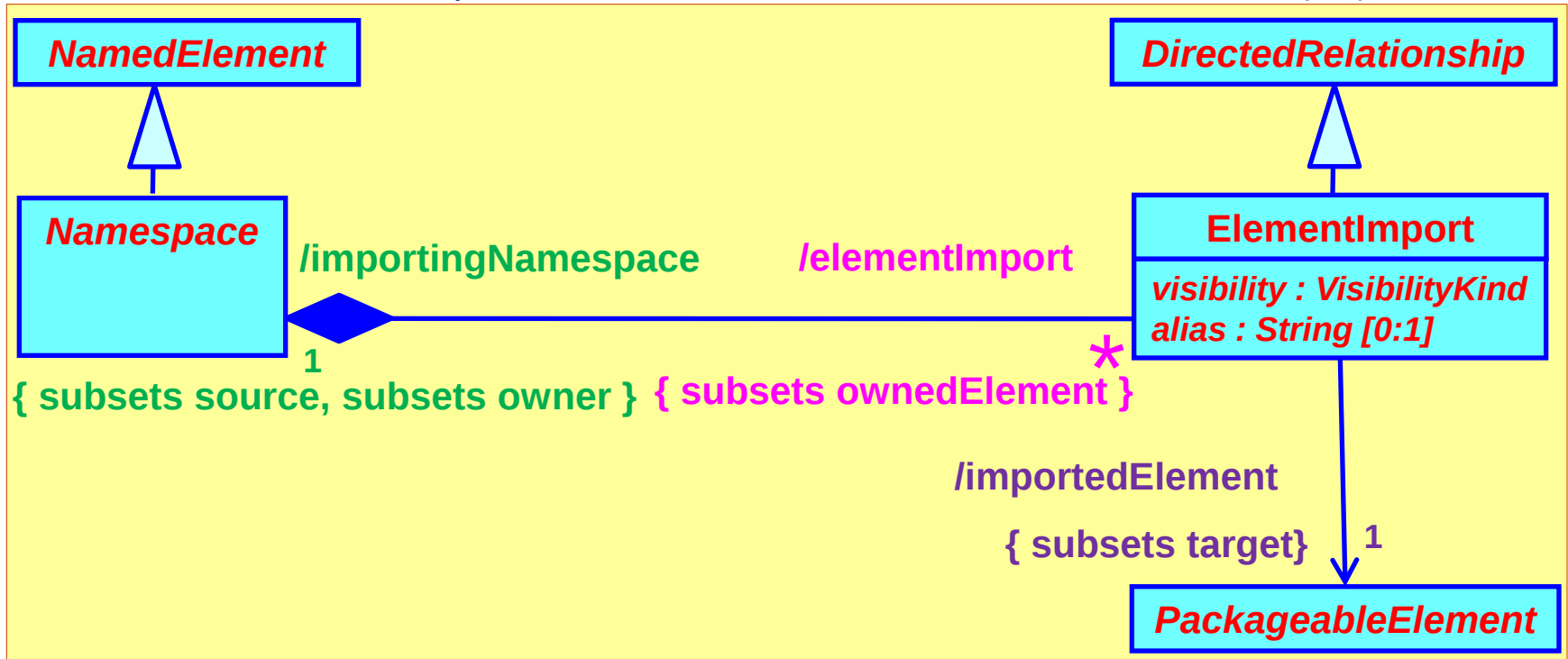
Моделирование импорта элементов в метамодели языка UML



Моделирование импорта элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



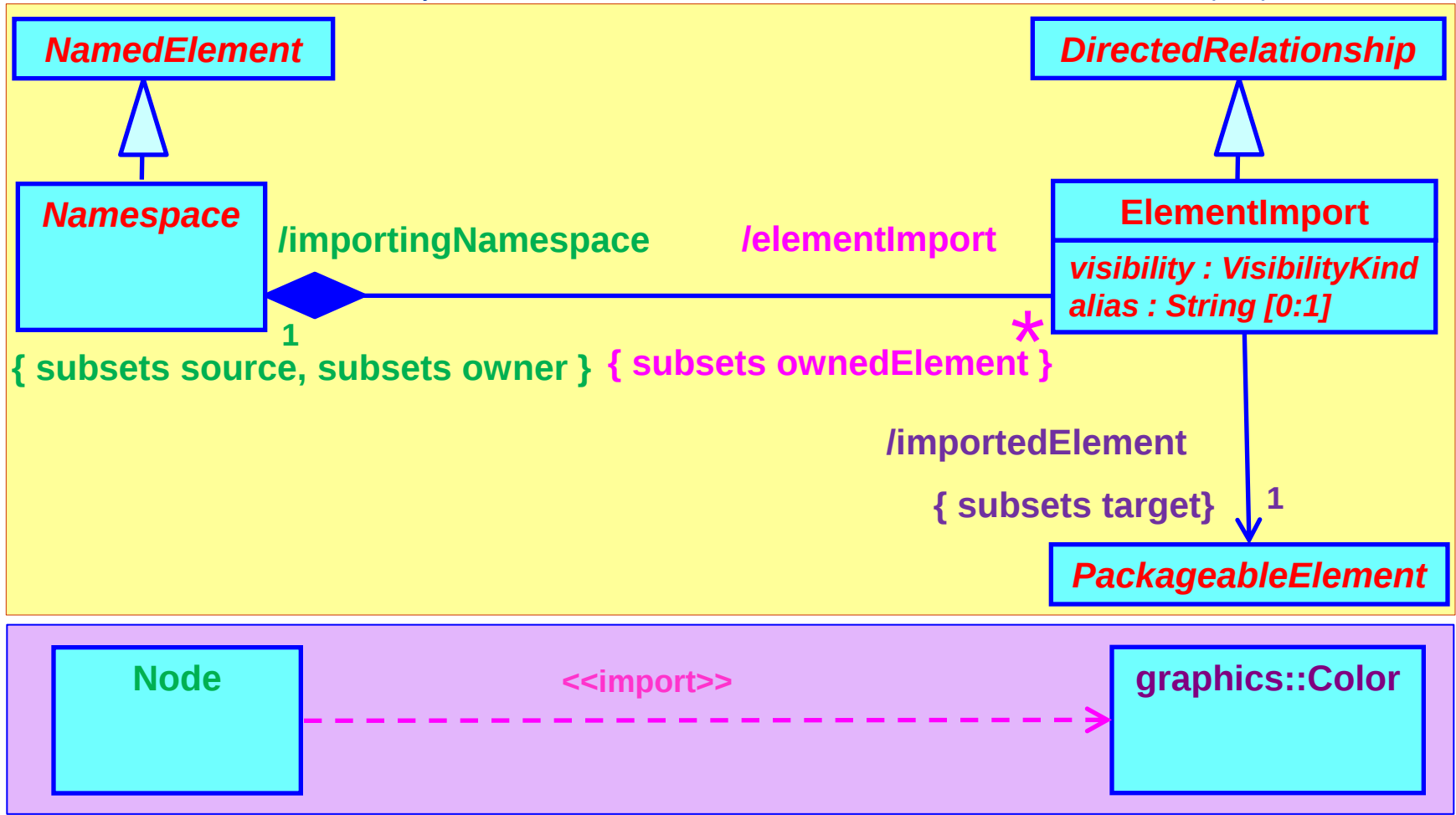
```
import graphics.Color;

class Node {
    Color borderColor;
}
```

Моделирование импорта элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

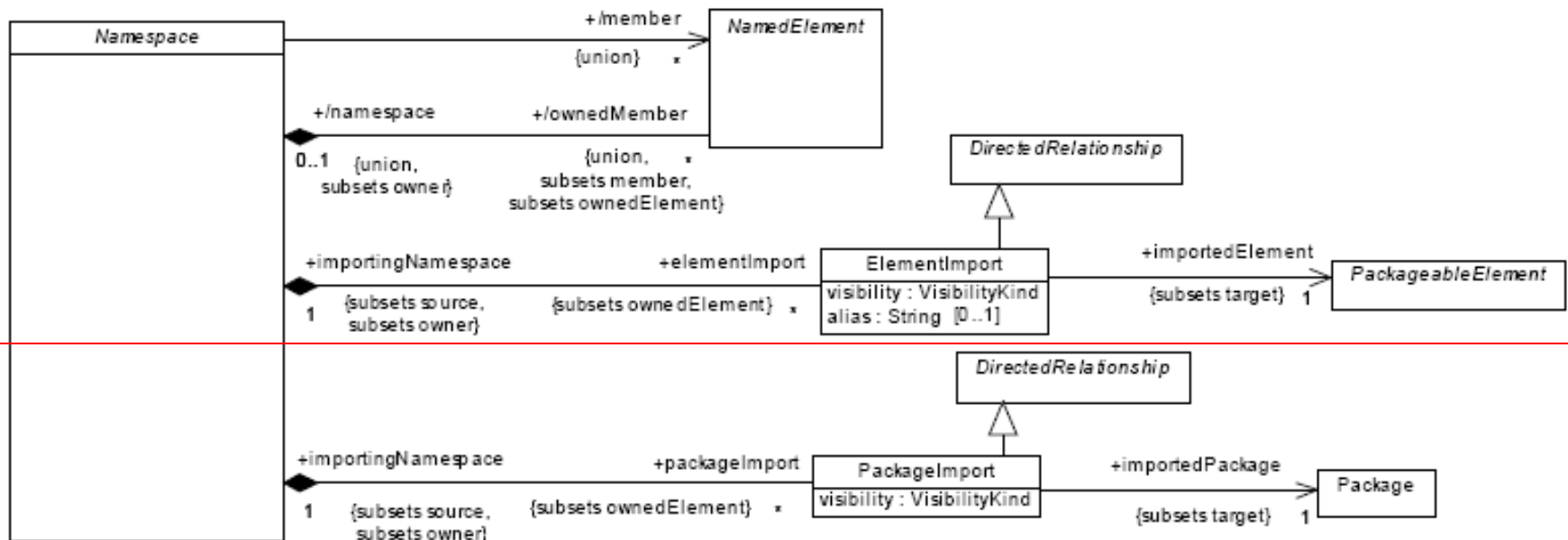
Романов Владимир Юрьевич ©2025



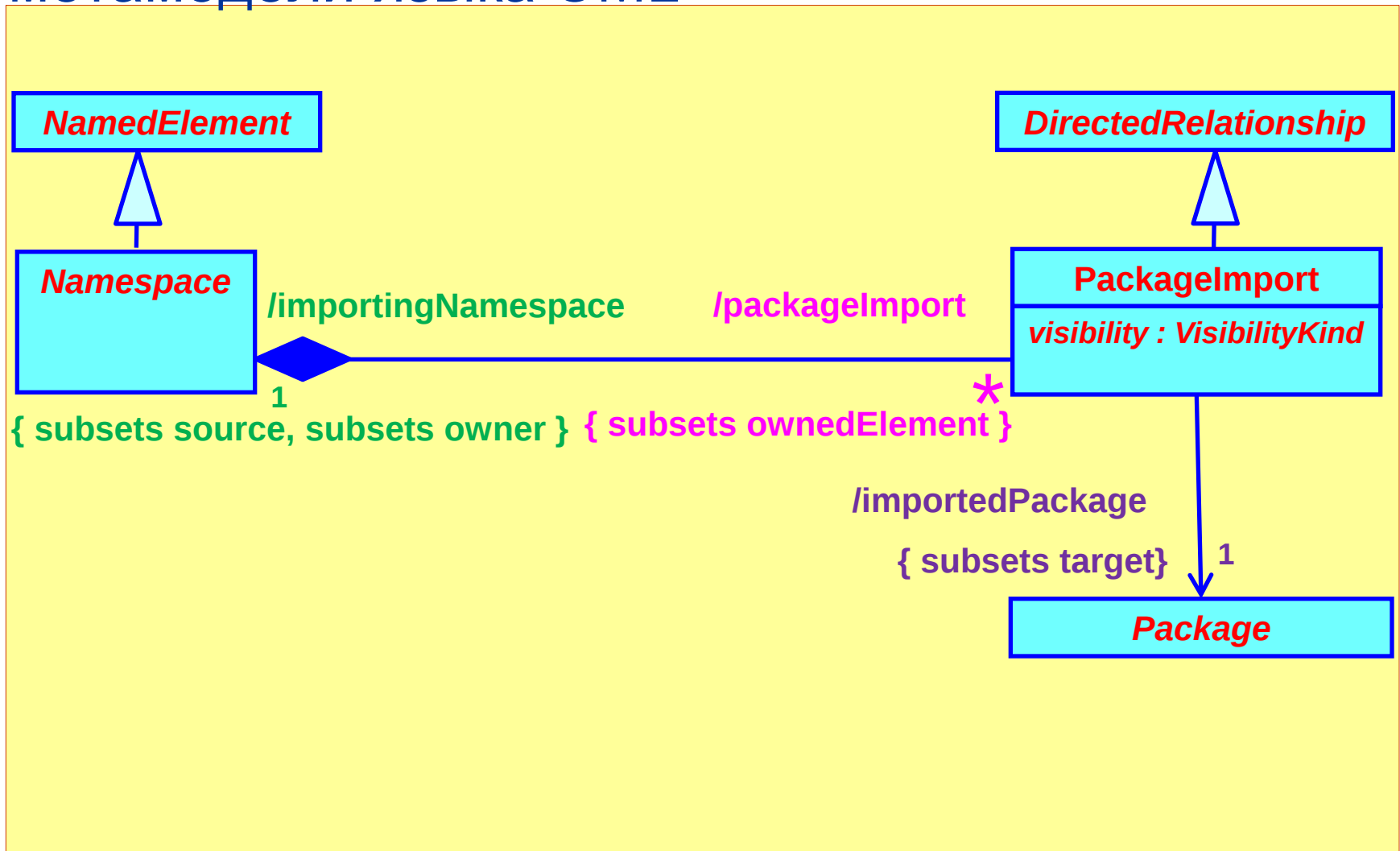
Импорт в пространства имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



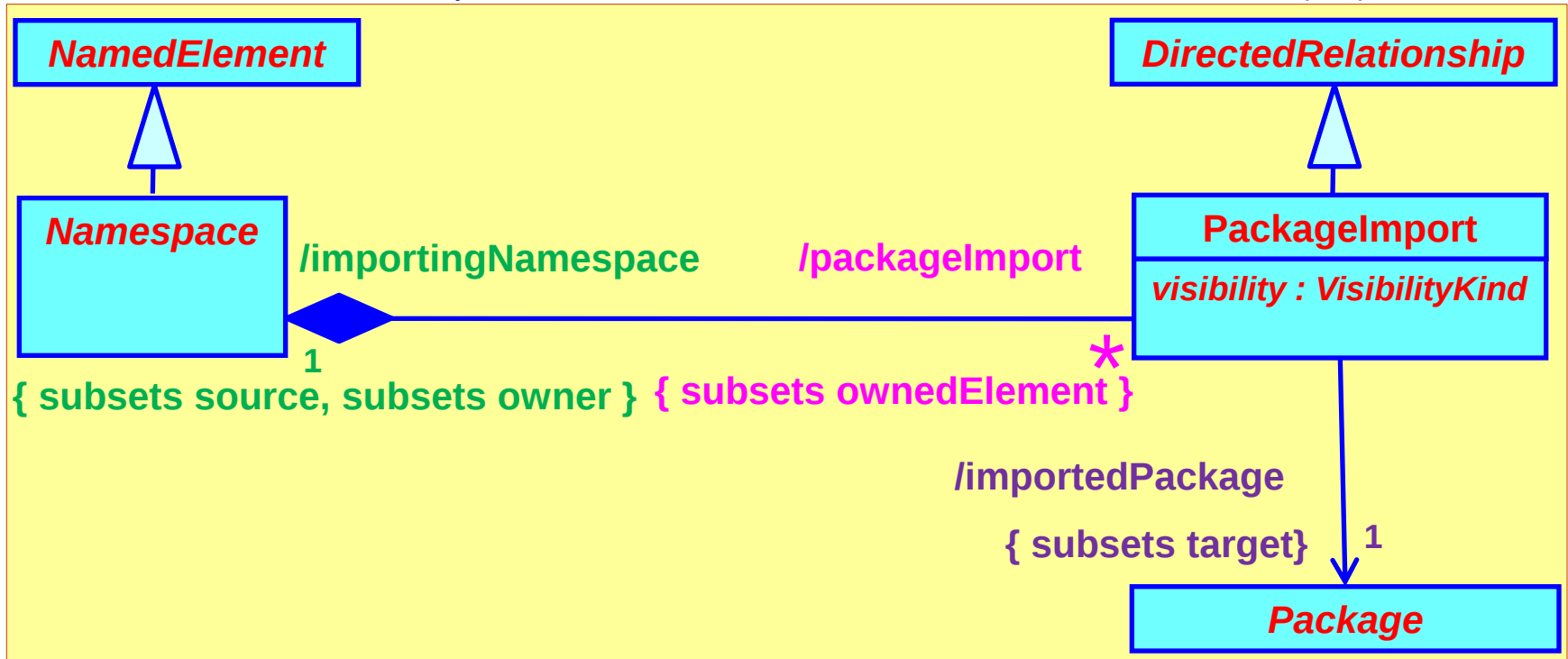
Моделирование импорта элементов в метамодели языка UML



Моделирование импорта элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

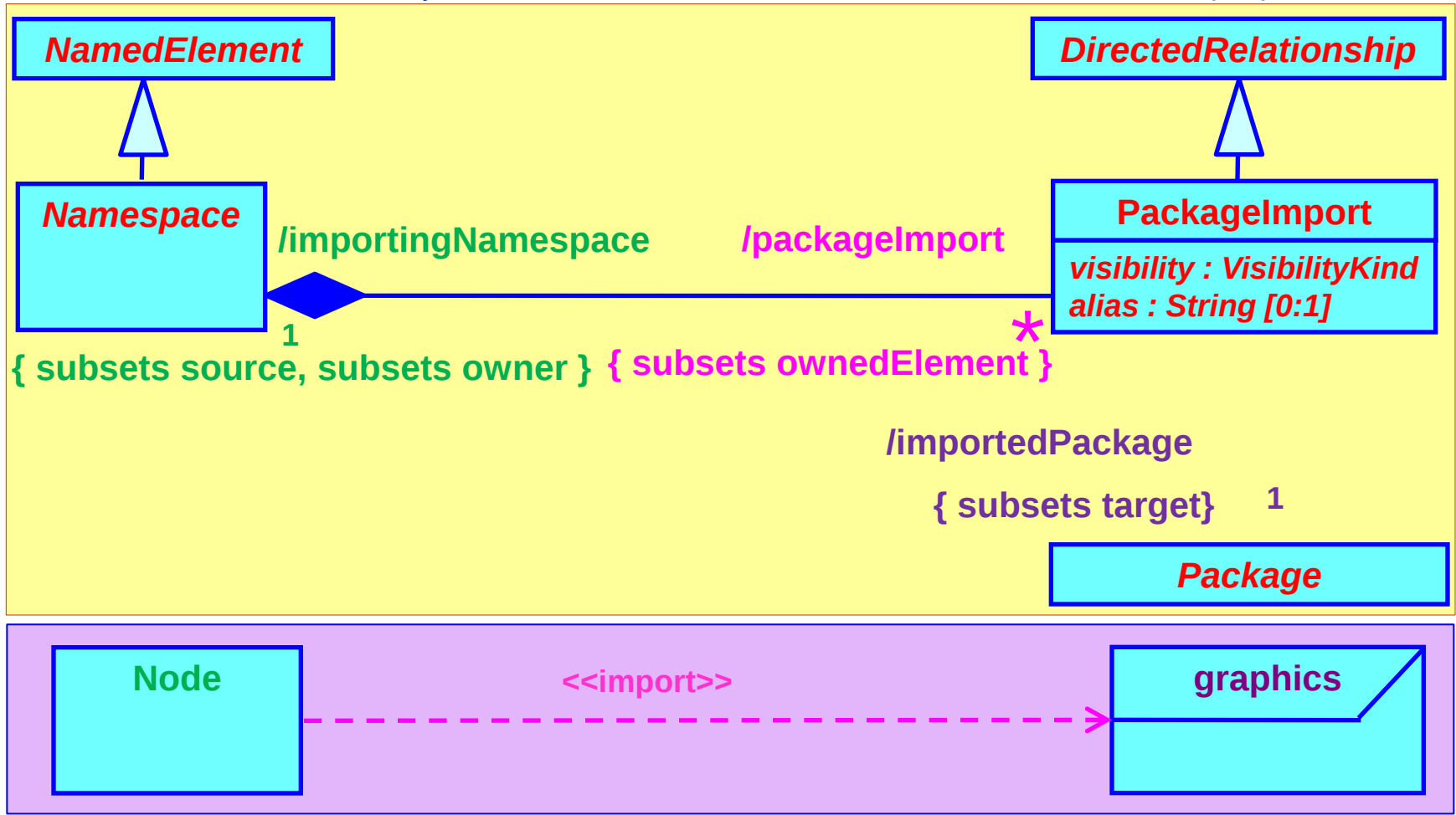
Романов Владимир Юрьевич ©2025



Моделирование импорта элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

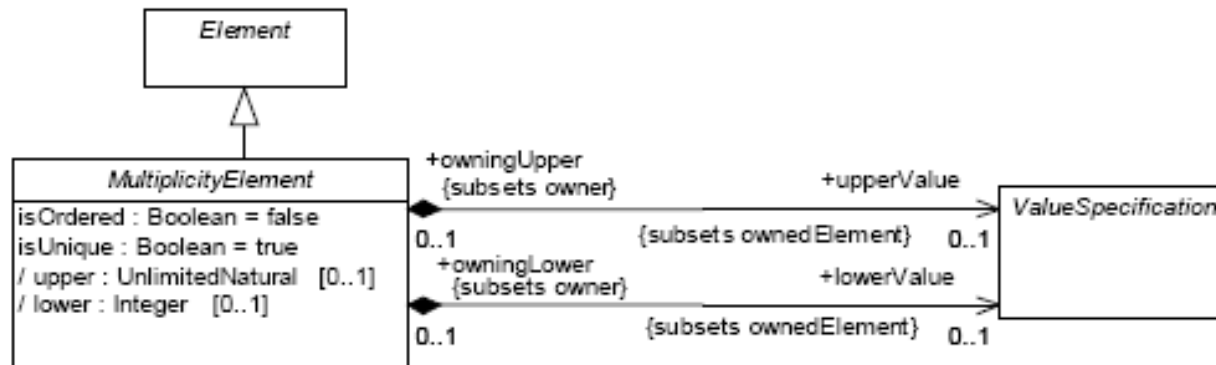
Романов Владимир Юрьевич ©2025



Представление множественных значений в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

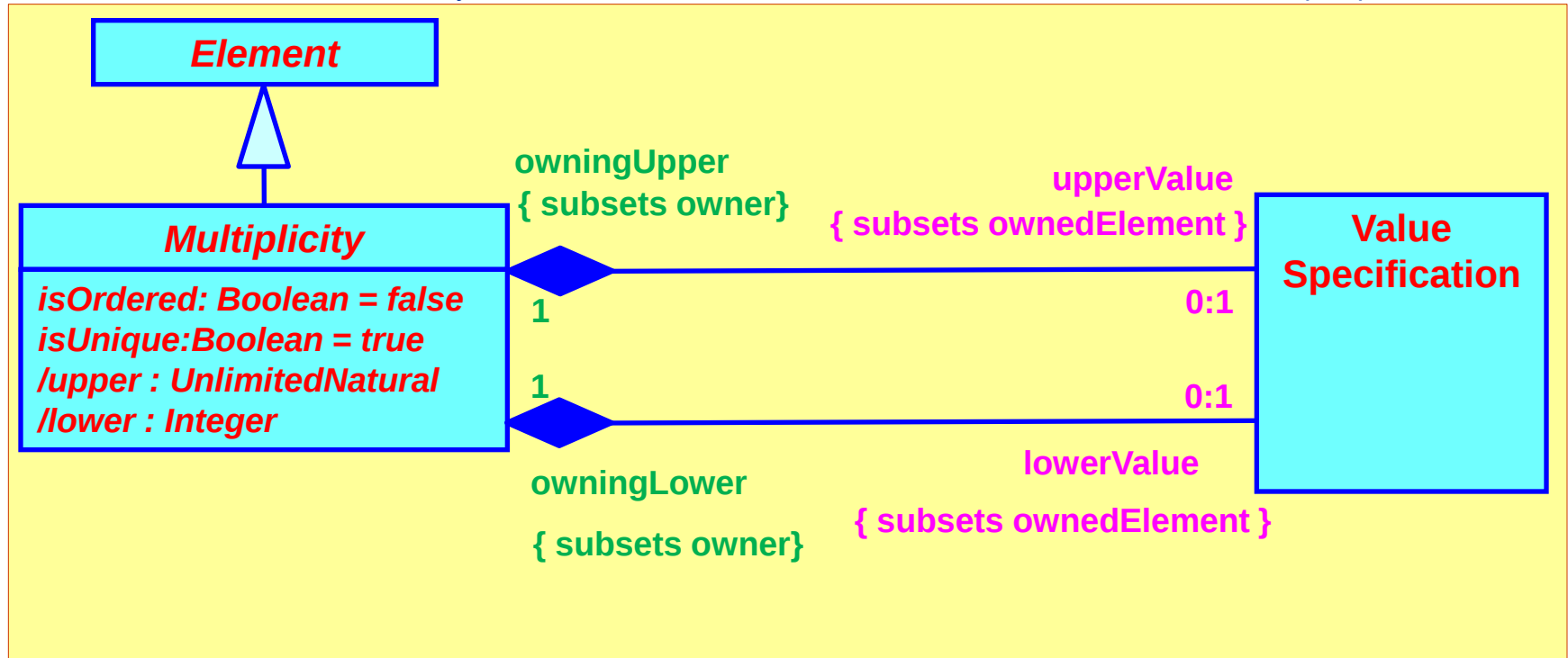
Романов Владимир Юрьевич ©2025



Моделирование множественных значений в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



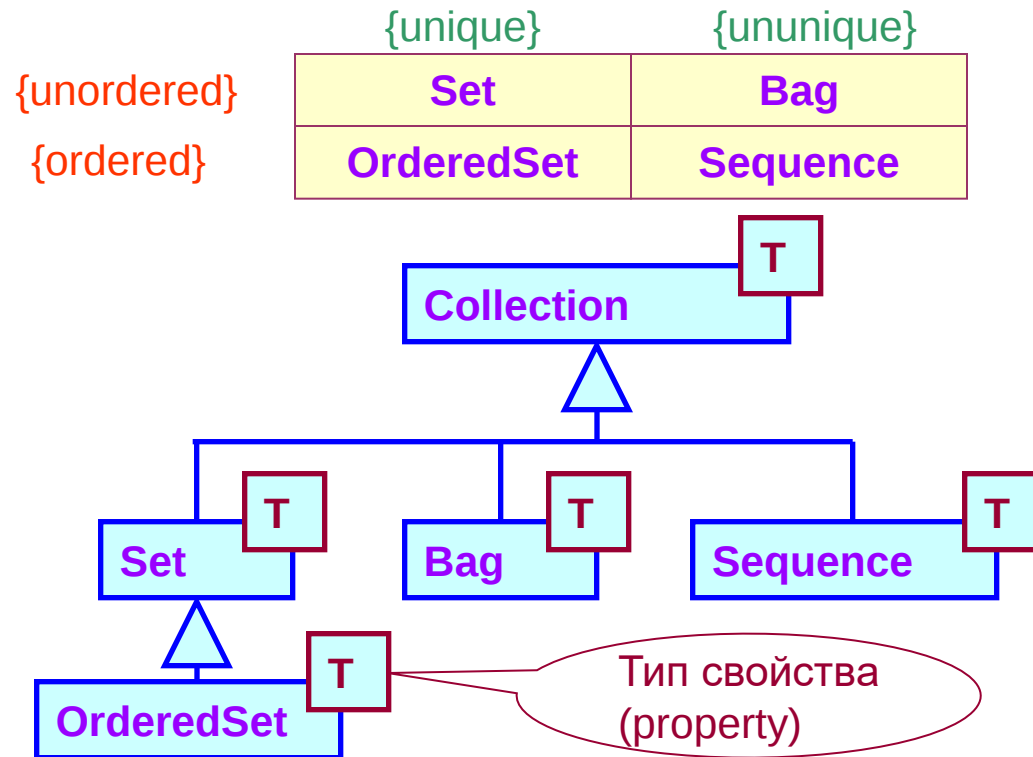
Отношение ассоциации.

Уникальность, упорядоченность и язык OCL

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

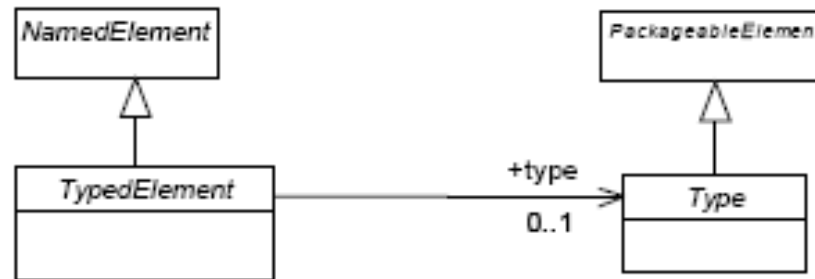
OCL коллекции



Типы и типизированные элементы в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

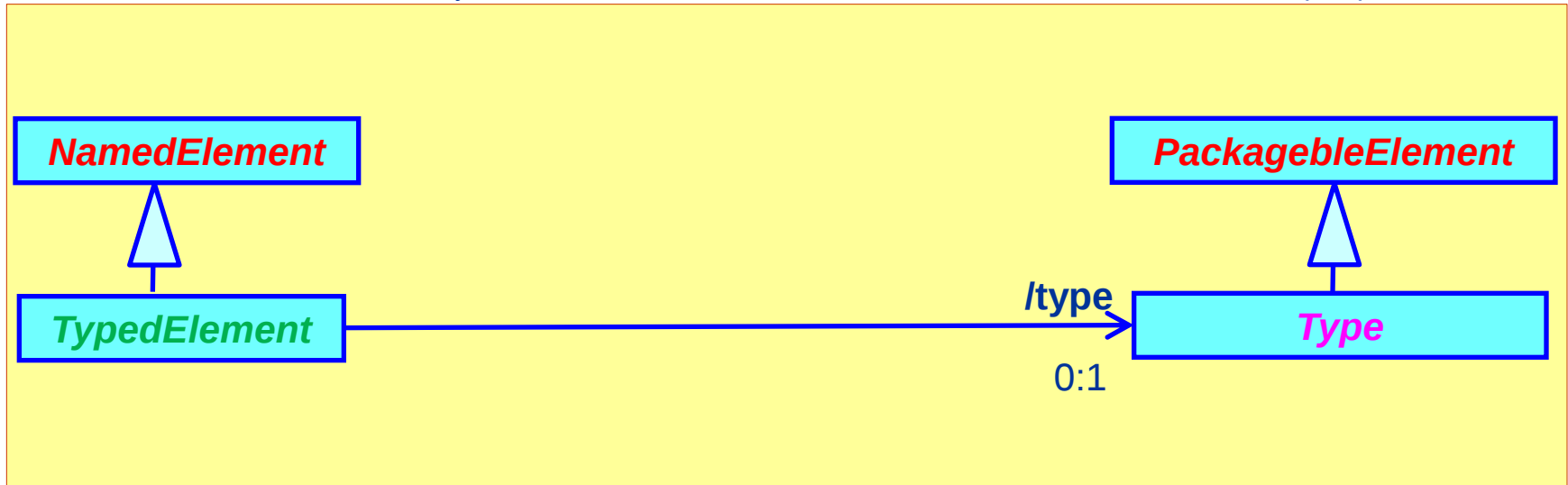
Романов Владимир Юрьевич ©2025



Моделирование импорта элементов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
import graphics.*;

class Node {
    Color borderColor;
    void draw(Graphics graphics) {
    }
}
```

Генерация текста на языке JAVA

UML → JAVA
на JAVA

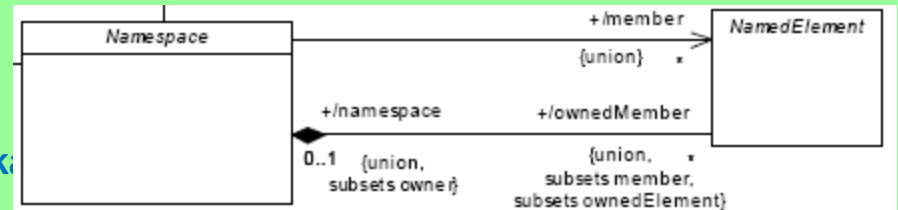
Генерация классификаторов пакета на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
package uml2java;
```

```
public static void generatePackage (Package package_, String packageDir) {  
    //  
    // Генерируем классификаторы текущего пакета.  
    //  
    for (NamedElement m : package_.getOwnedMembers()) {  
        // Внешний тип неизвестного вида (класс, интерфейс, перечисление?)  
        if ( m.hasKeyword("unknown") )  
            continue;  
  
        if (m instanceof Class)  
            generateClass ((Class) m, packageDir);  
        else  
        if (m instanceof Interface)  
            generateInterface ((Interface) m, packageDir);  
        else  
        if (m instanceof Enumeration)  
            generateEnumeration ((Enumeration) m, packageDir);  
        else  
        if (m instanceof Package)  
            generatePackage((Package) m, packageDir + "/" + m.getName());  
    }  
}
```



Генерация класса

Генерация текстов класса на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public static void generateClass (Class cls, String packageDir) {  
    makePackageDir(packageDir);
```

```
    PrintStream ps = null;
```

```
    try { ps = new PrintStream(packageDir + "/" + cls.getName() + ".java"); }  
    catch (FileNotFoundException e) { return; }
```

```
    genPackage(class_, ps);
```

```
    genImports(class_, ps);
```

```
    String modifiers = visibilityToJava (cls.getVisibility());
```

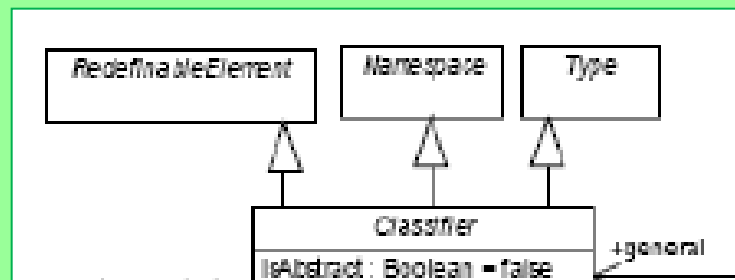
```
    if (cls.isAbstract())  
        modifiers += "abstract ";
```

```
    if (cls.isLeaf())  
        modifiers += "final ";
```

```
    ps.format("%n%sclass %s ", modifiers, cls.getName());  
    // ...
```

NamedElement	
name : String	{0..1}
visibility : VisibilityKind	{0..1}
/ qualifiedName : String	{0..1}

<<enumeration>> VisibilityKind	
public	
private	
protected	
package	



Генерация текстов класса на языке Java

```
genParents(cls, ps);  
  
ps.format("{ %n");  
  
genFields(cls, ps);  
  
genMethods(cls, ps);  
  
ps.format("} %n");  
  
ps.close();  
}
```

Короткое квалифицированное имя Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private static String getShortName(NamedElement named) {  
    String longName = named.getQualifiedName().replace("::", ".");  
  
    int k = longName.indexOf('.');  
  
    return longName.substring(k + 1);  
}
```

NamedElement	
name : String	{0..1}
visibility : VisibilityKind	{0..1}
/ qualifiedName : String	{0..1}

Генерация пакета для класса Java

```
private static void genPackage (Class cls, PrintStream ps) {  
    Package nearestPackage = cls.getNearestPackage();  
  
    // Короткое квалифицированное имя  
    // (без имени модели в начале имени).  
    String qName = getShortName(nearestPackage);  
  
    ps.format("package %s; %n%n", qName);  
}
```

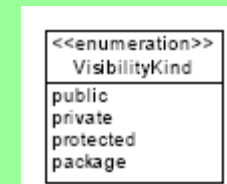
Генерация текстов класса на языке Java.

Получение текста для видимости элемента.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

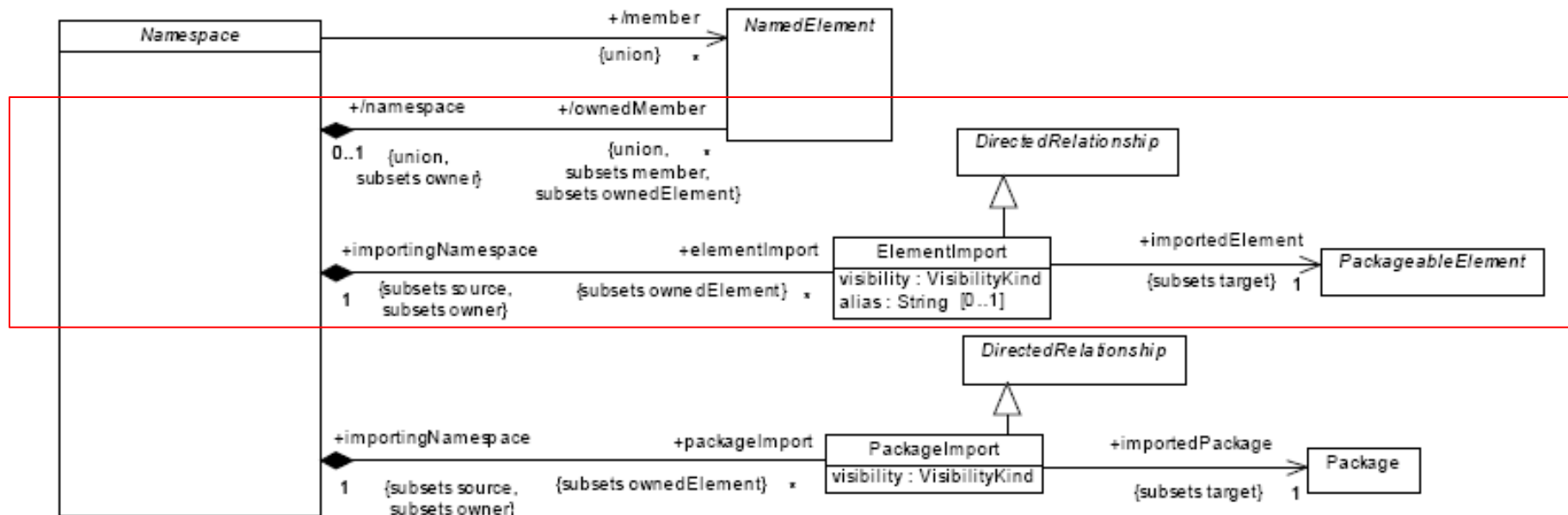
```
/**
 * Получить ключевое слово Java для представления видимости в модели UML.
 * @param visibility значение видимости
 * @return ключевое слово Java
 */
public static String visibilityToJava (VisibilityKind visibility) {
    return visibility == VisibilityKind.PACKAGE_LITERAL
        ? ""
        : visibility.getName() + ' ';
}
```



Импорт в пространства имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

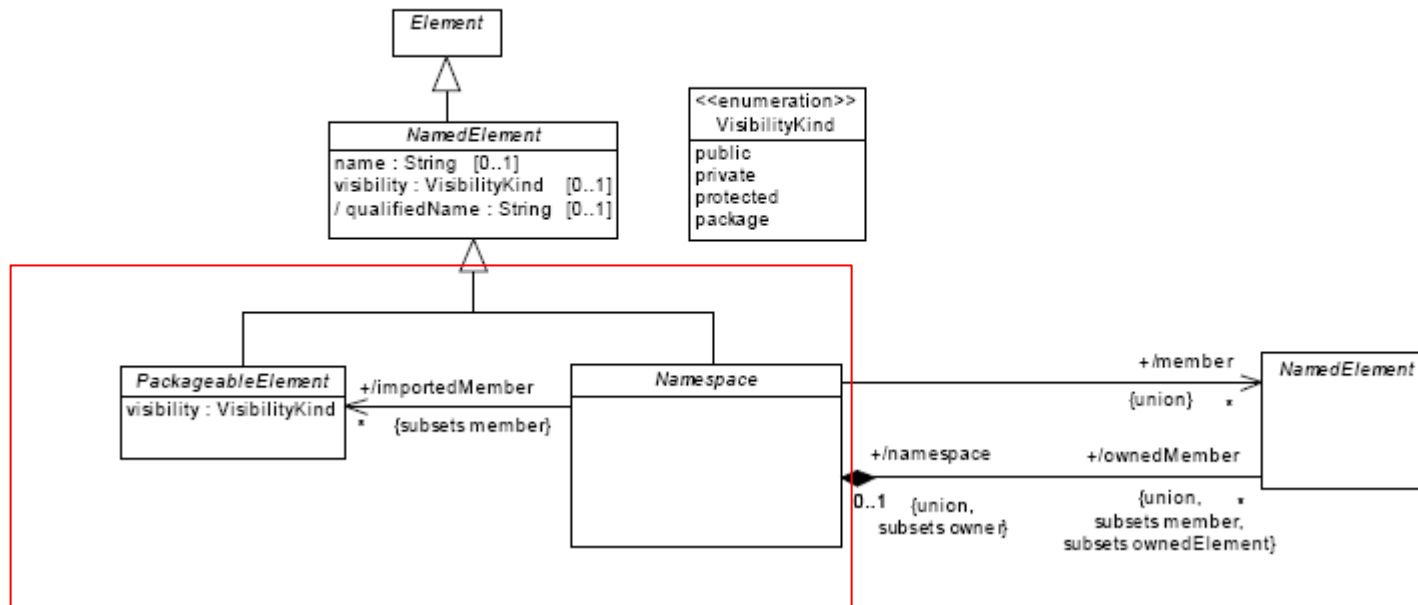
Романов Владимир Юрьевич ©2025



Представление пространств имен в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Генерация текстов класса на языке Java.

Генерация импортов класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

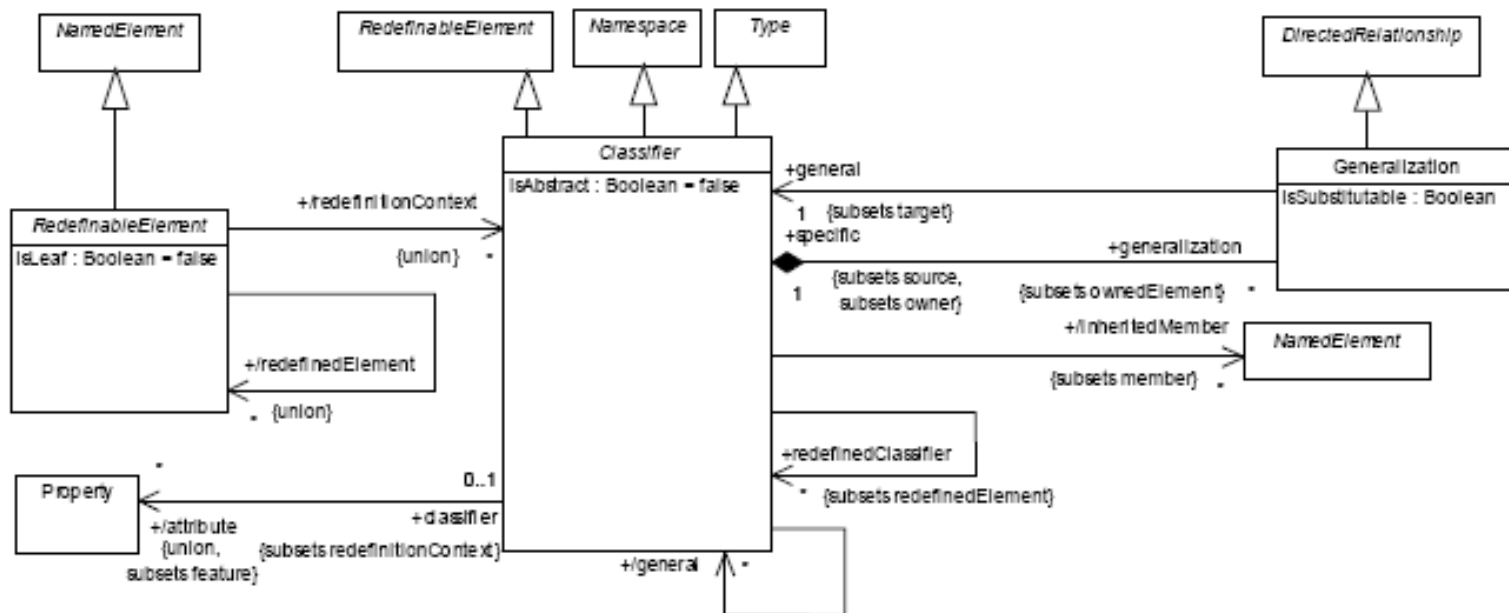
```
private static void genImports (Classifier classifier, PrintStream ps) {  
    for (PackageableElement ie : classifier.getImportedMembers()) {  
        String imported = getShortName(ie);  
  
        if (! imported.startsWith("java.lang."))  
            ps.format("import %s; %n", imported);  
    }  
}
```



Классификаторы в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

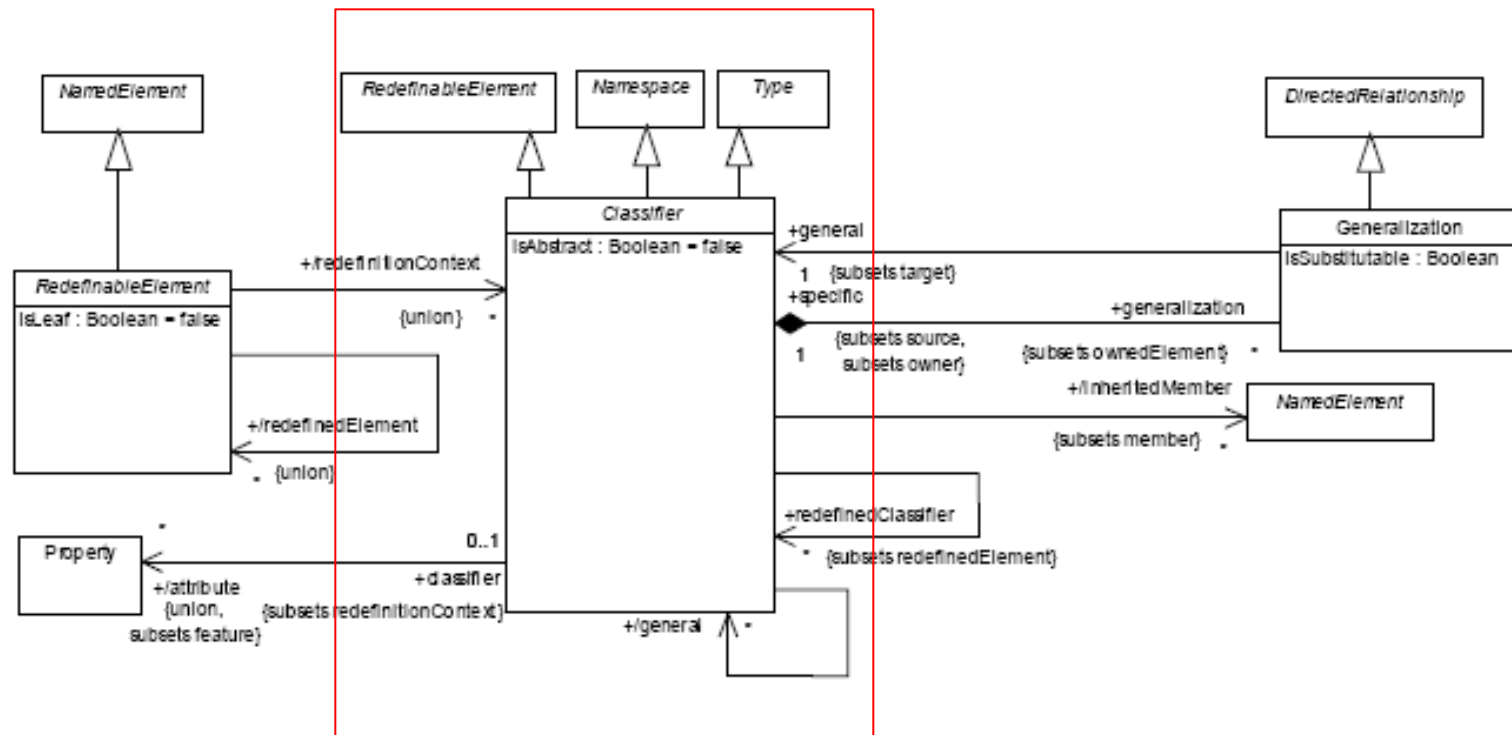
Романов Владимир Юрьевич ©2025



Классификаторы в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК. Романов Владимир Юрьевич ©2025

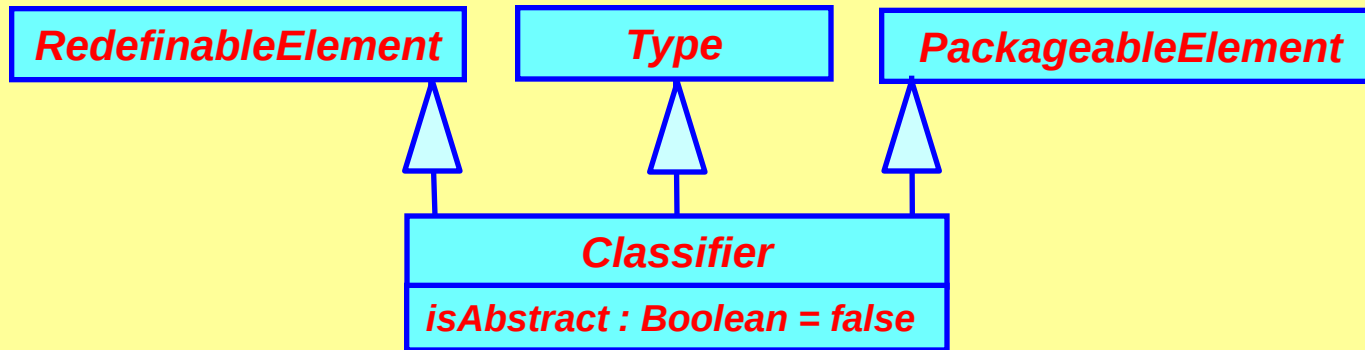
МГУ им. М.В.Ломоносова. Факультет ВМК. Романов Владимир Юрьевич ©2025



Моделирование классификаторов метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

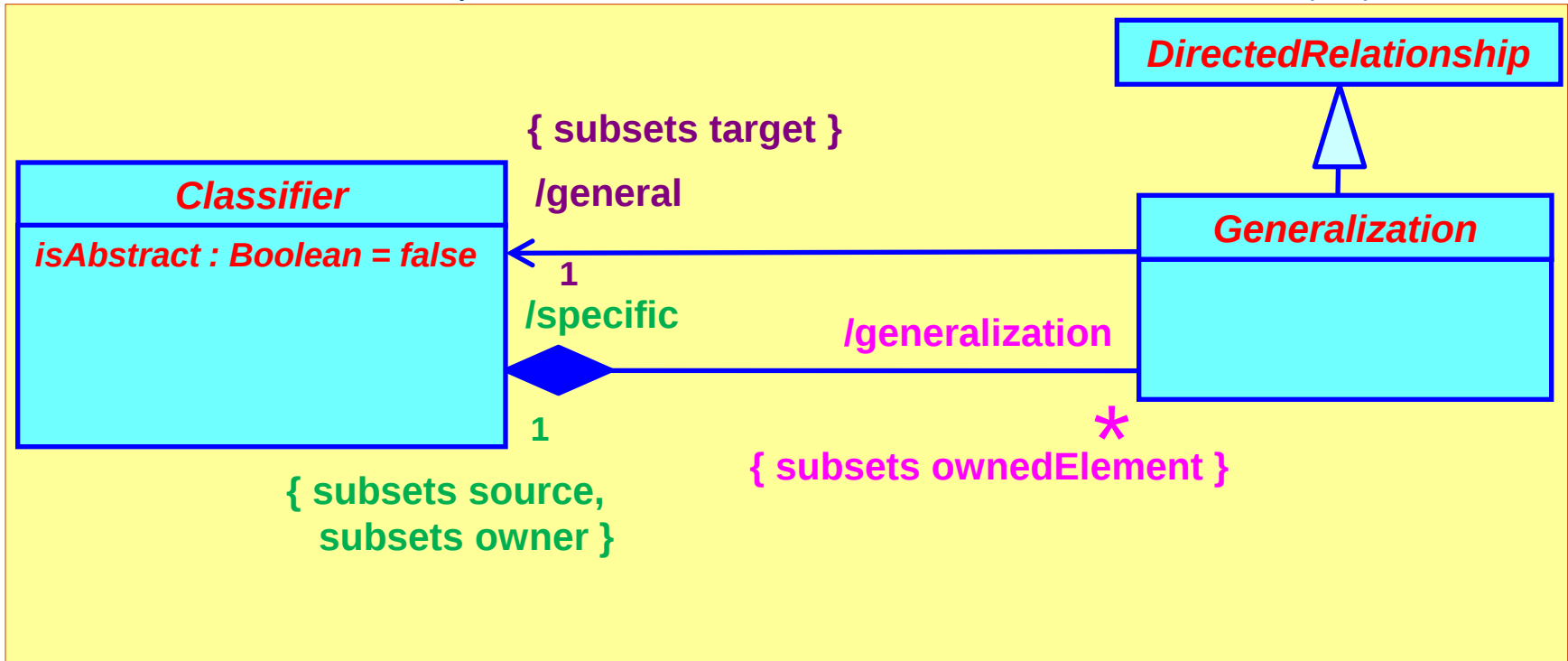
Романов Владимир Юрьевич ©2025



Моделирование отношения наследования (обобщения) метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

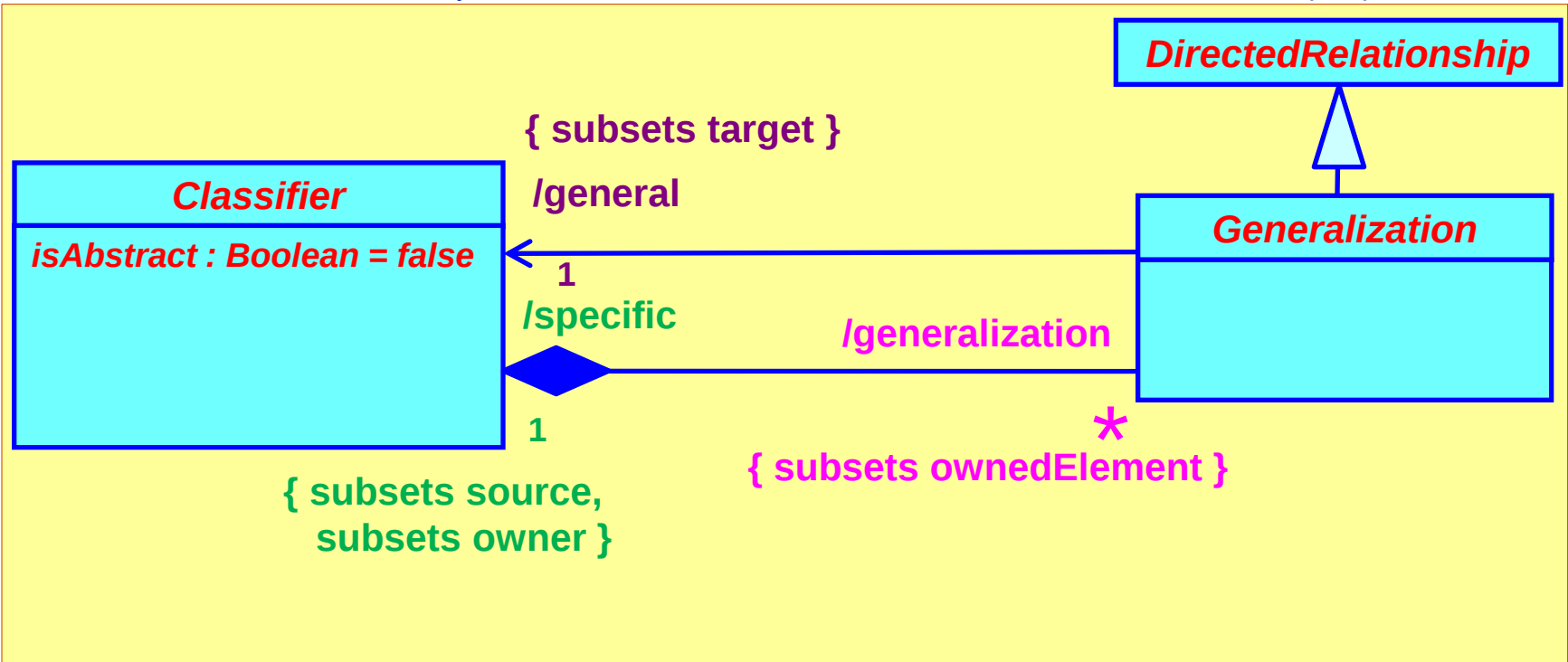


```
interface FlyingFish extends Swimming, Flying {  
  
}
```

Моделирование отношения наследования (обобщения) метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
class Node extends View {  
  
}
```

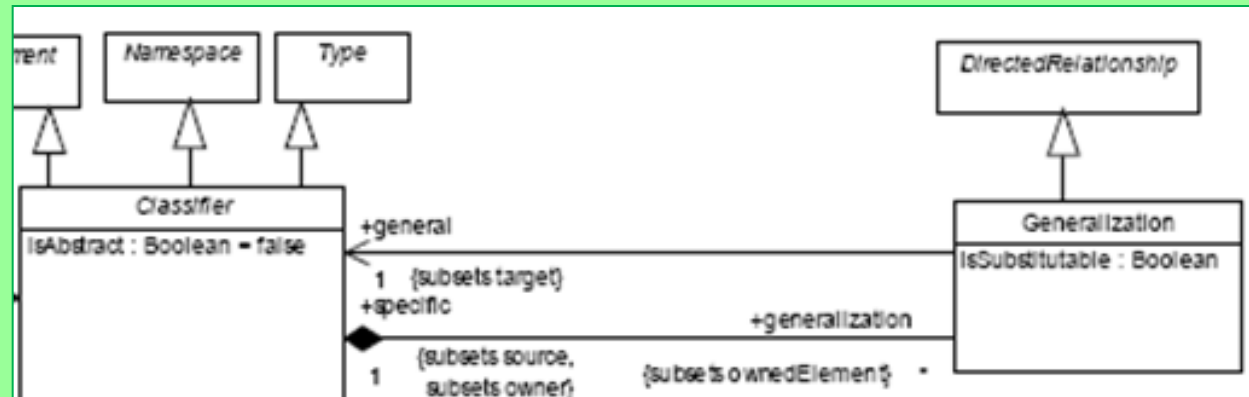
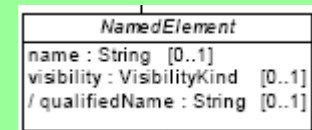
Генерация текстов класса на языке Java.

Генерация предков класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

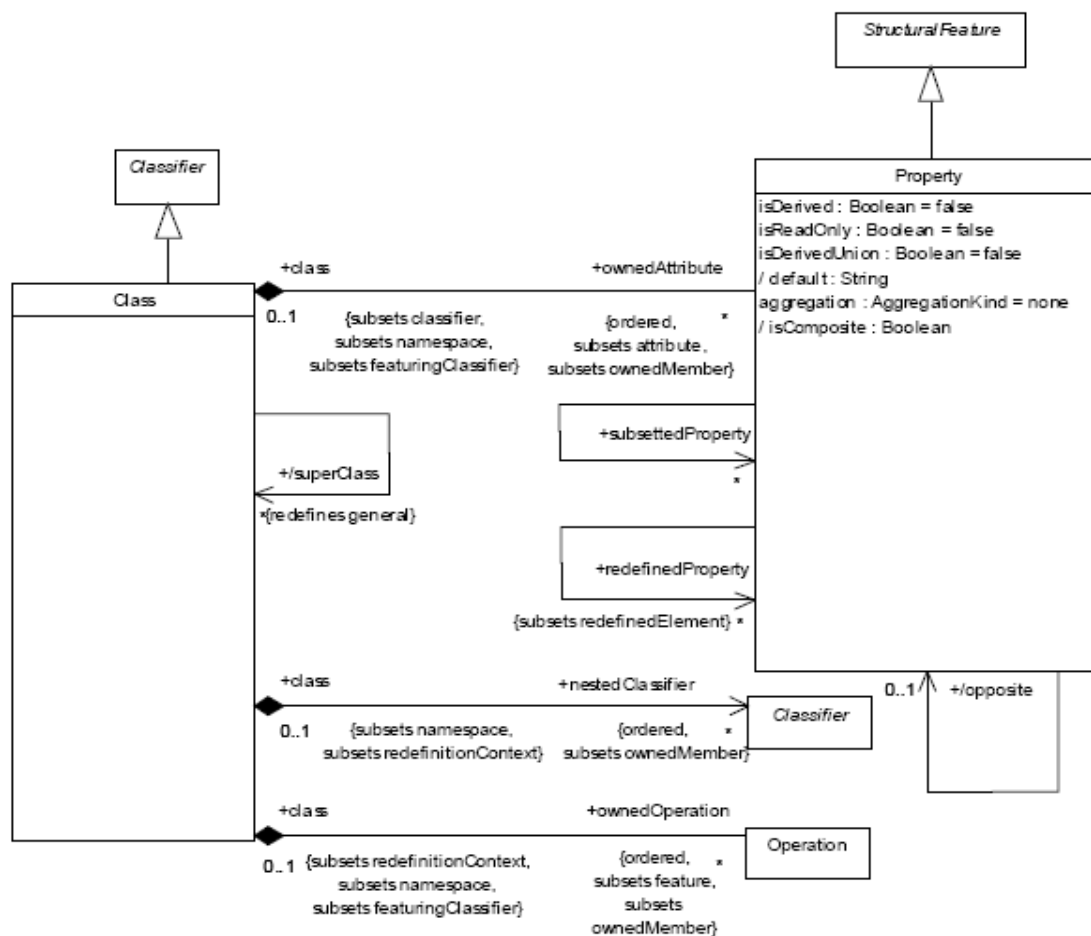
```
private static void genParents (Class cls, PrintStream ps) {  
    for (Generalization g : cls.getGeneralizations()) {  
        Classifier parent = g.getGeneral();  
  
        String parentQName = parent.getQualifiedName();  
        if (parentQName.endsWith("java::lang::Object"))  
            return;  
  
        ps.format(" extends %s %n", parent.getName());  
        return;  
    }  
}
```



Классы, их свойства и операции в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

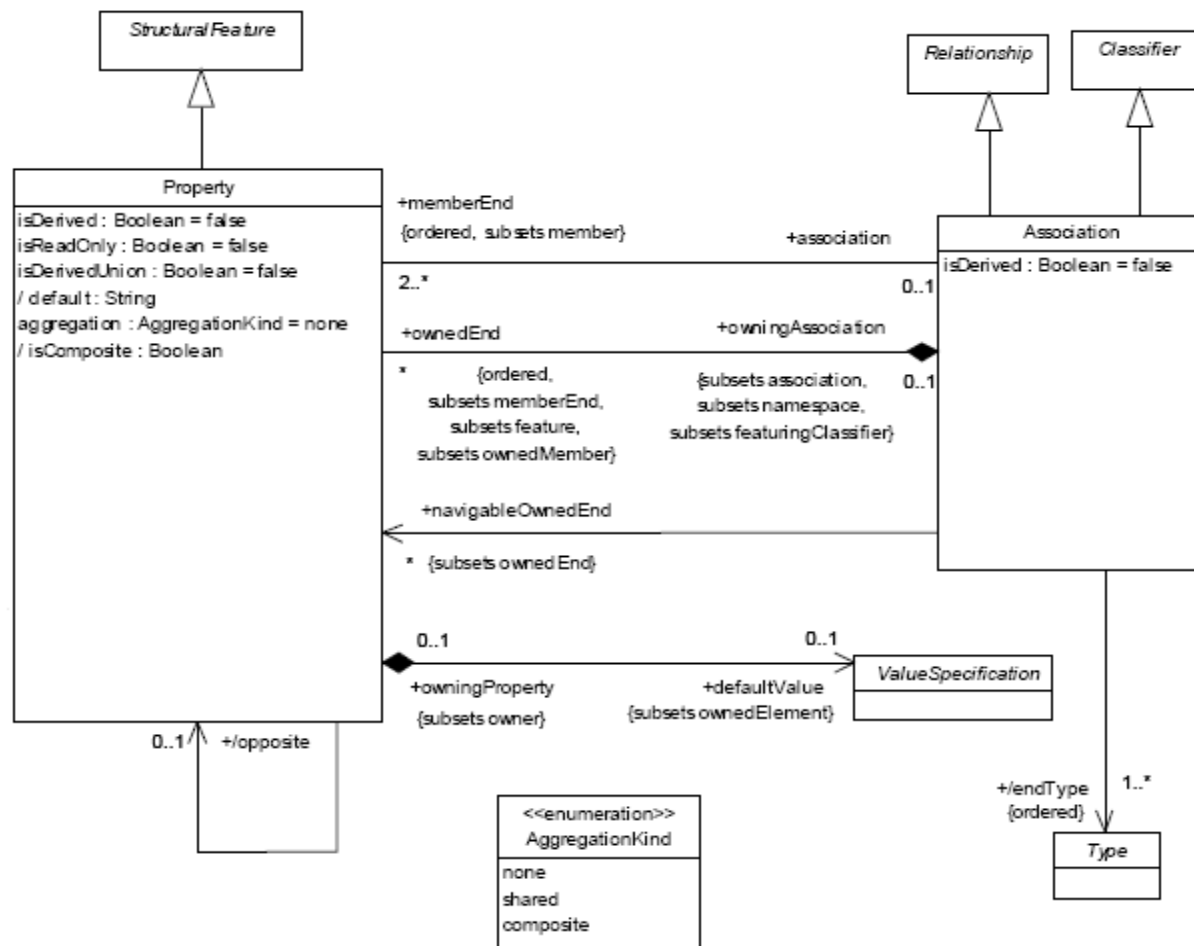
Романов Владимир Юрьевич ©2025



Отношения ассоциации между свойствами в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



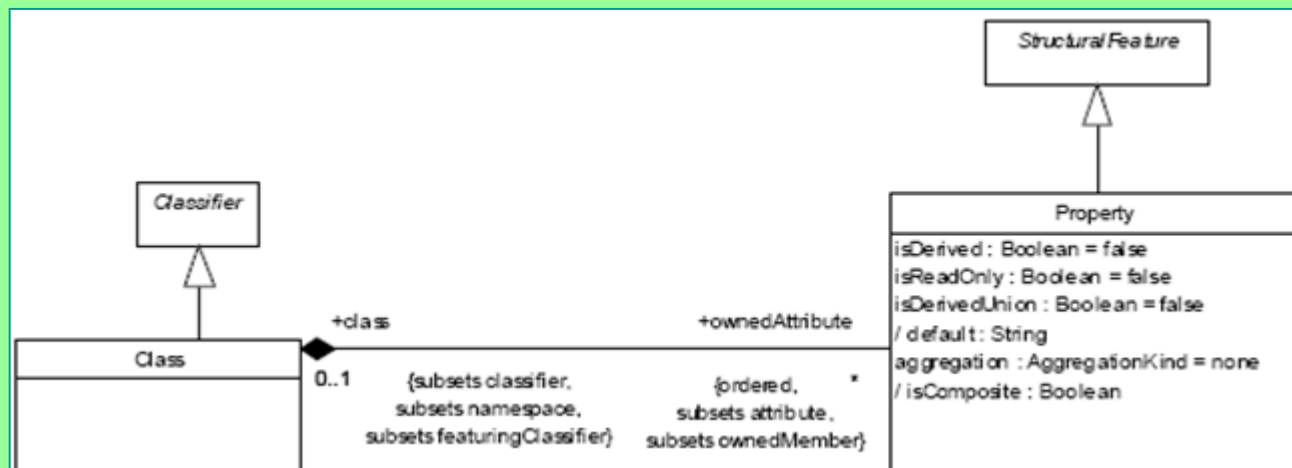
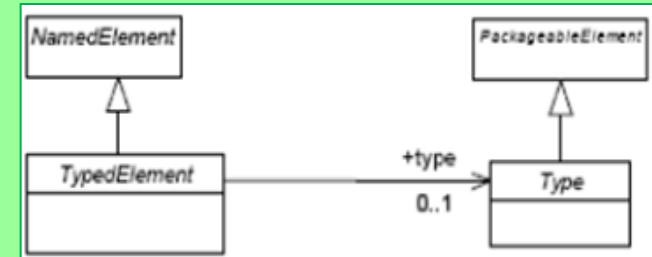
Генерация текстов класса на языке Java.

Генерация полей класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

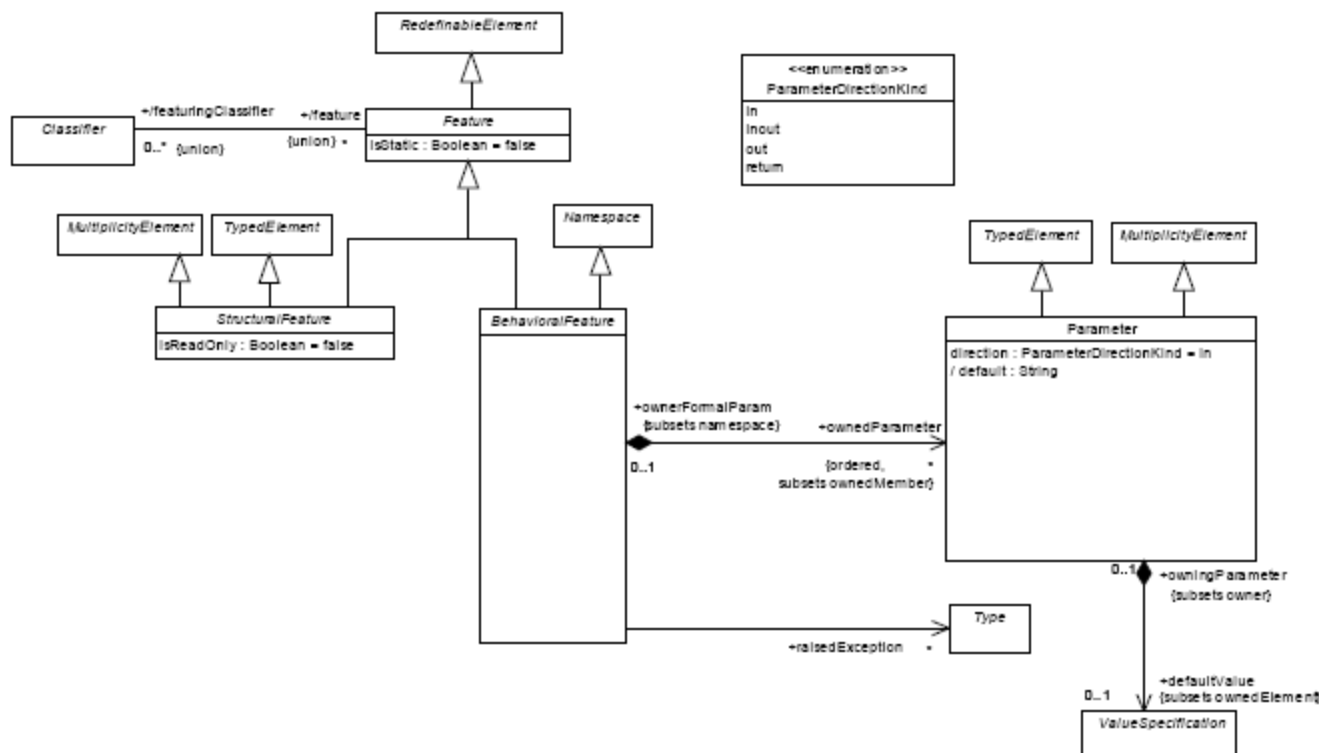
```
private static void genFields (Class cls, PrintStream ps) {  
    for (Property f : cls.getOwnedAttributes()) {  
        String typeName = f.getType().getName();  
  
        String modifiers = visibilityToJava(f.getVisibility());  
        if (f.isStatic())  
            modifiers += "static";  
  
        String fieldName = f.getName();  
        ps.format("%s %s %s;%n", modifiers, typeName , fieldName );  
    }  
}
```



Структура и поведение классификаторов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

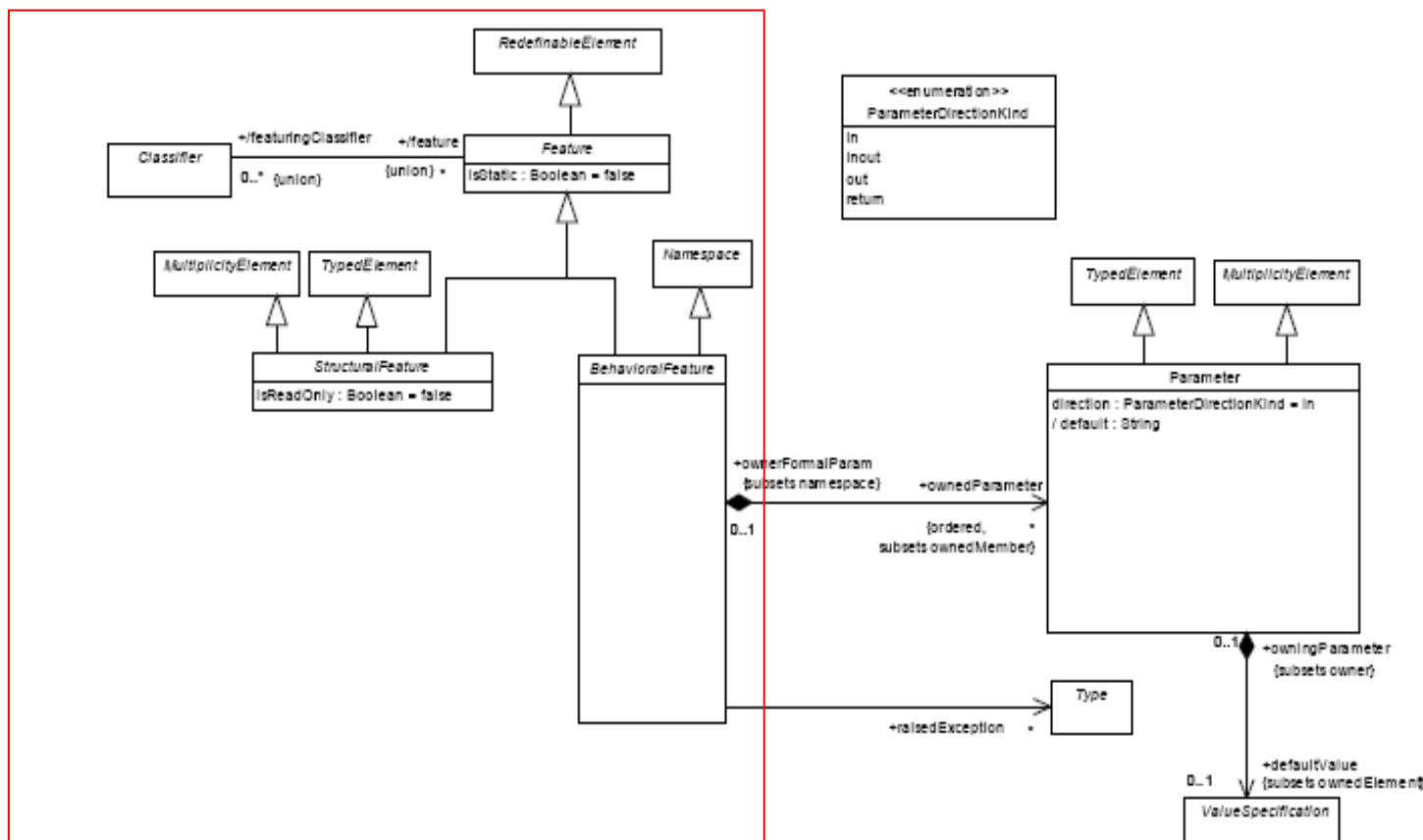
Романов Владимир Юрьевич ©2025



Структура и поведение классификаторов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

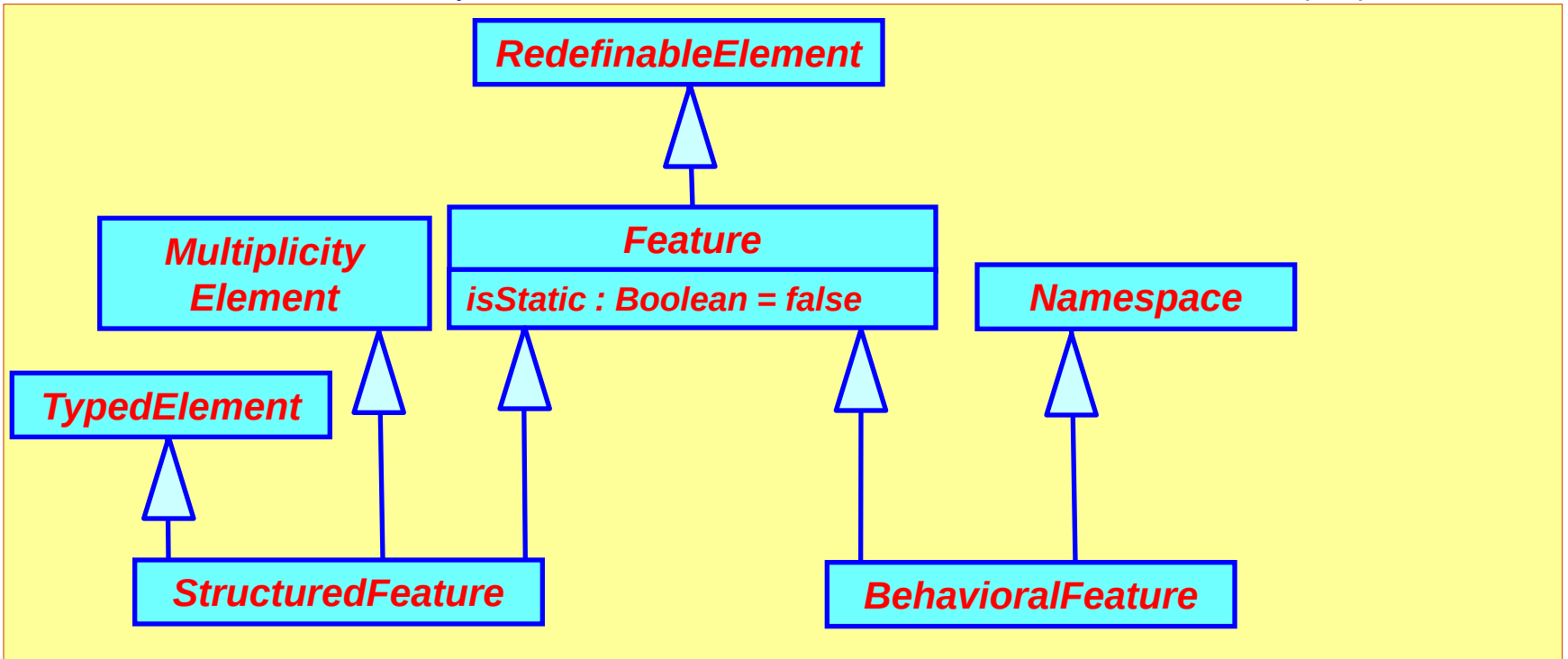
Романов Владимир Юрьевич ©2025



Моделирование возможностями в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

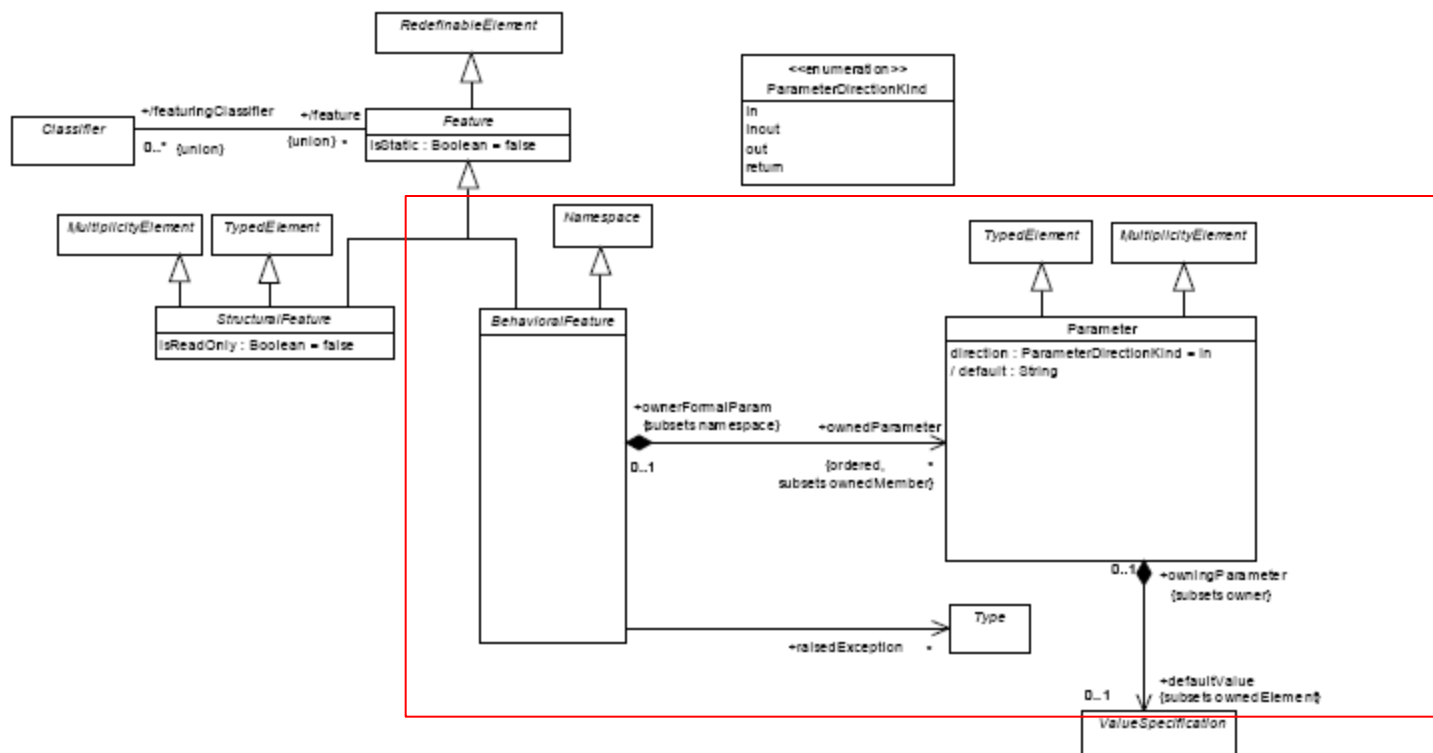
Романов Владимир Юрьевич ©2025



Структура и поведение классификаторов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

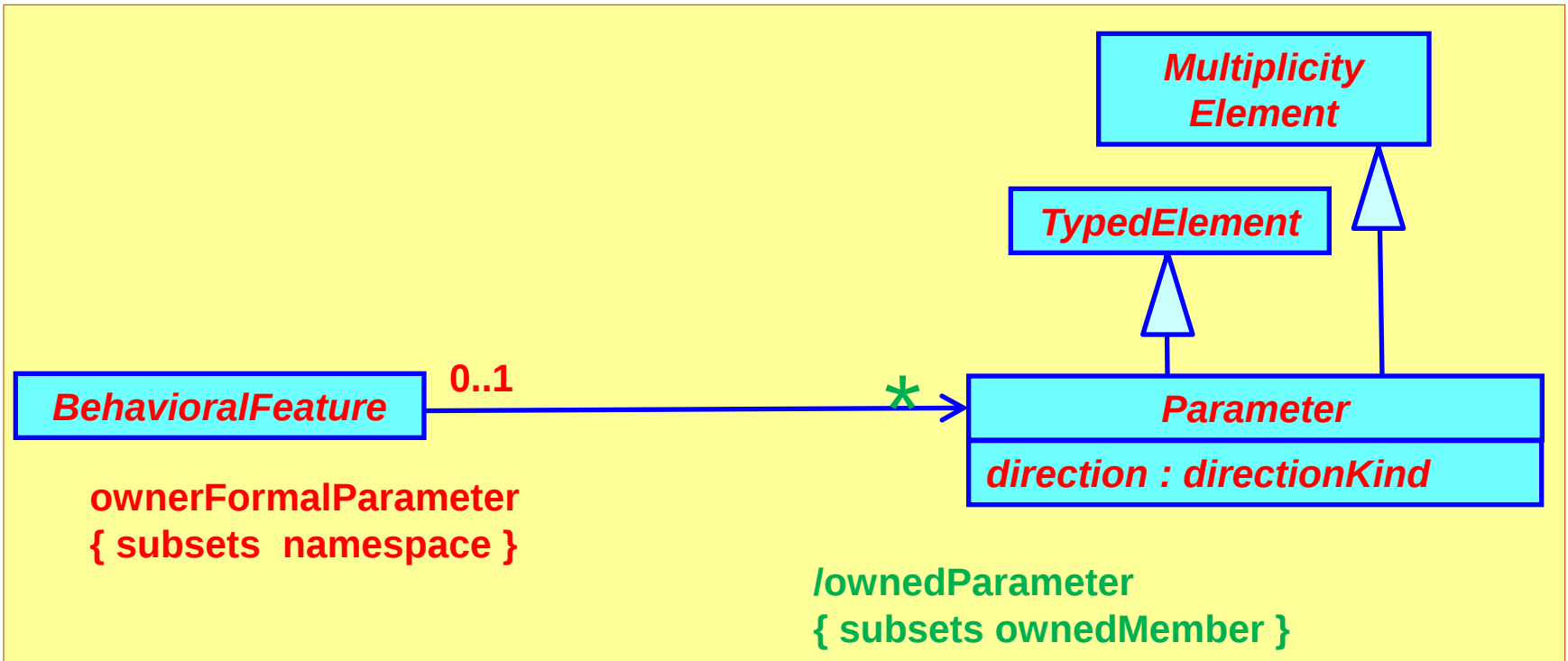
Романов Владимир Юрьевич ©2025



Моделирование параметров в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

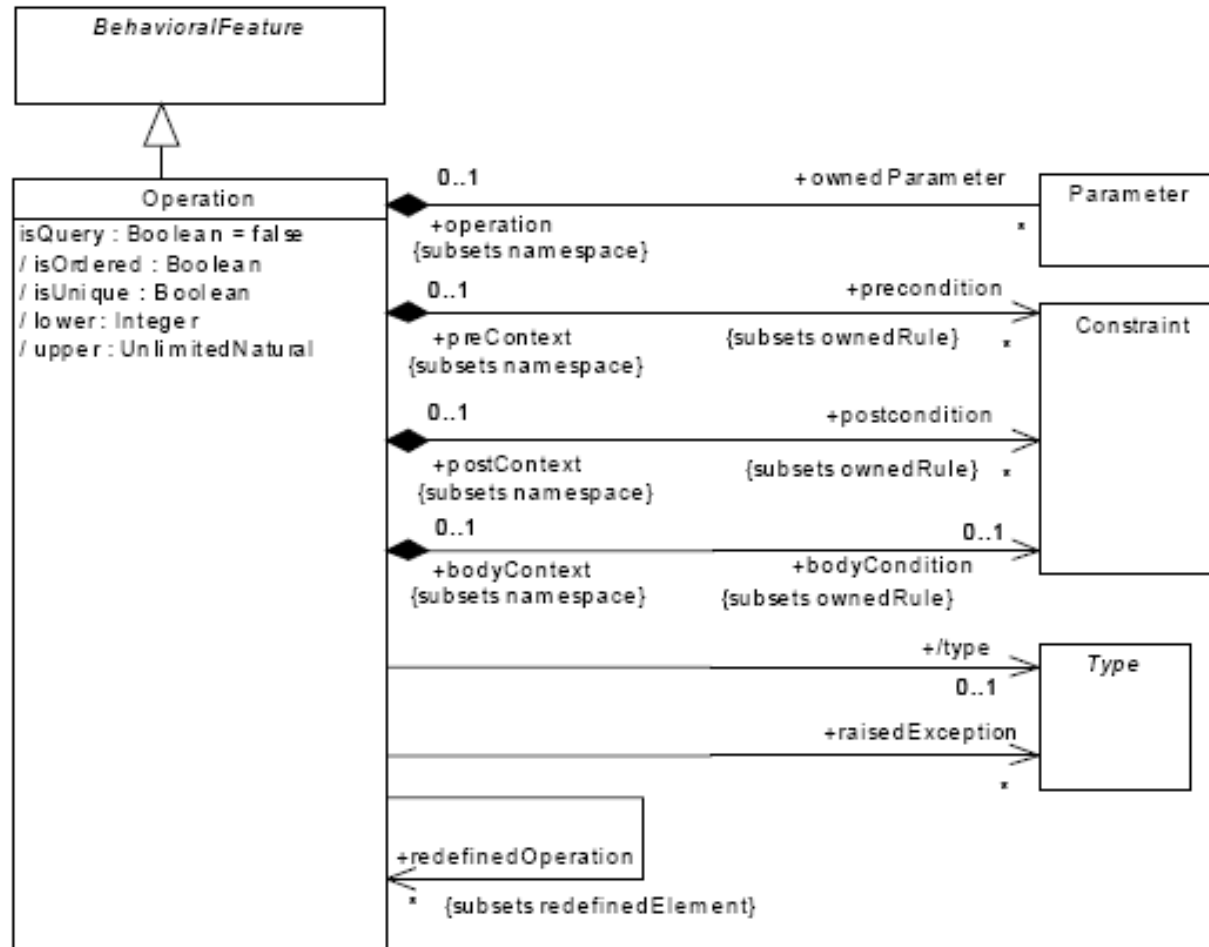
Романов Владимир Юрьевич ©2025



Операции классификаторов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

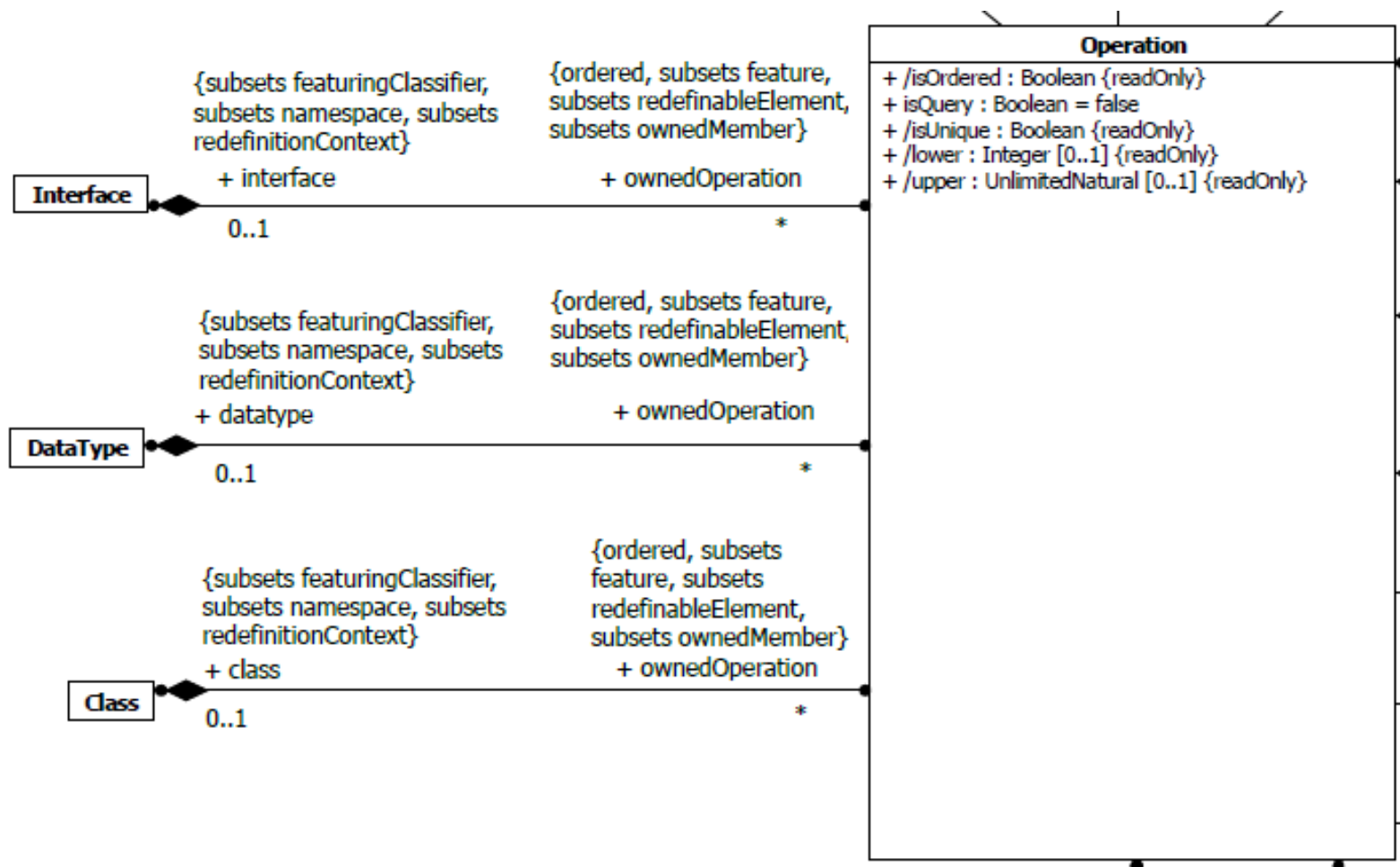
Романов Владимир Юрьевич ©2025



Операции классификаторов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Генерация текстов класса на языке Java.

Генерация методов класса (1)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private static void genMethods (Class cls, PrintStream ps) {  
  
    for (Operation op : cls.getOwnedOperations()) {  
        String operationName = op.getName();  
  
        String modifiers = visibilityToJava(op.getVisibility());  
  
        if (op.isStatic())  
            modifiers += "static";  
  
        //...
```

Генерация текстов класса на языке Java.

Генерация методов класса (2)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
//...
```

```
String returns = "void";  
String parameters = "";
```

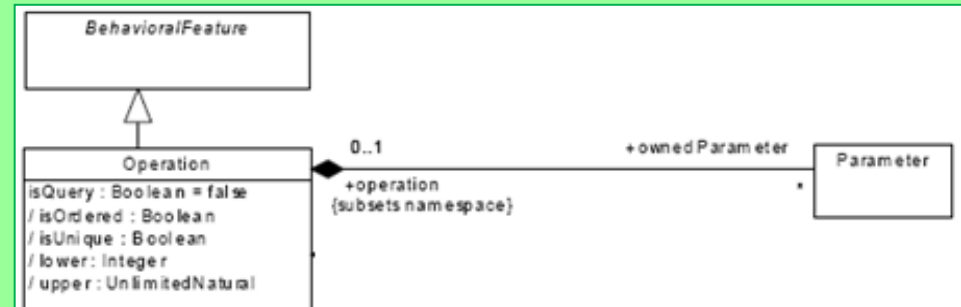
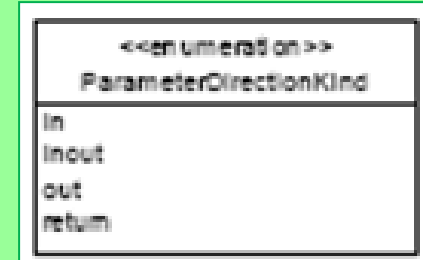
```
for(Parameter p : op.getOwnedParameters()) {  
    String typeName = p.getType().getName();
```

```
    if (p.getDirection() == ParameterDirectionKind.RETURN_LITERAL) {  
        returns = typeName;  
        continue;  
    }
```

```
    if (!parameters.isEmpty())  
        parameters += ", ";
```

```
    parameters += typeName + " " + p.getName();
```

```
}  
//...
```



Генерация текстов класса на языке Java.

Генерация методов класса (3)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
//...

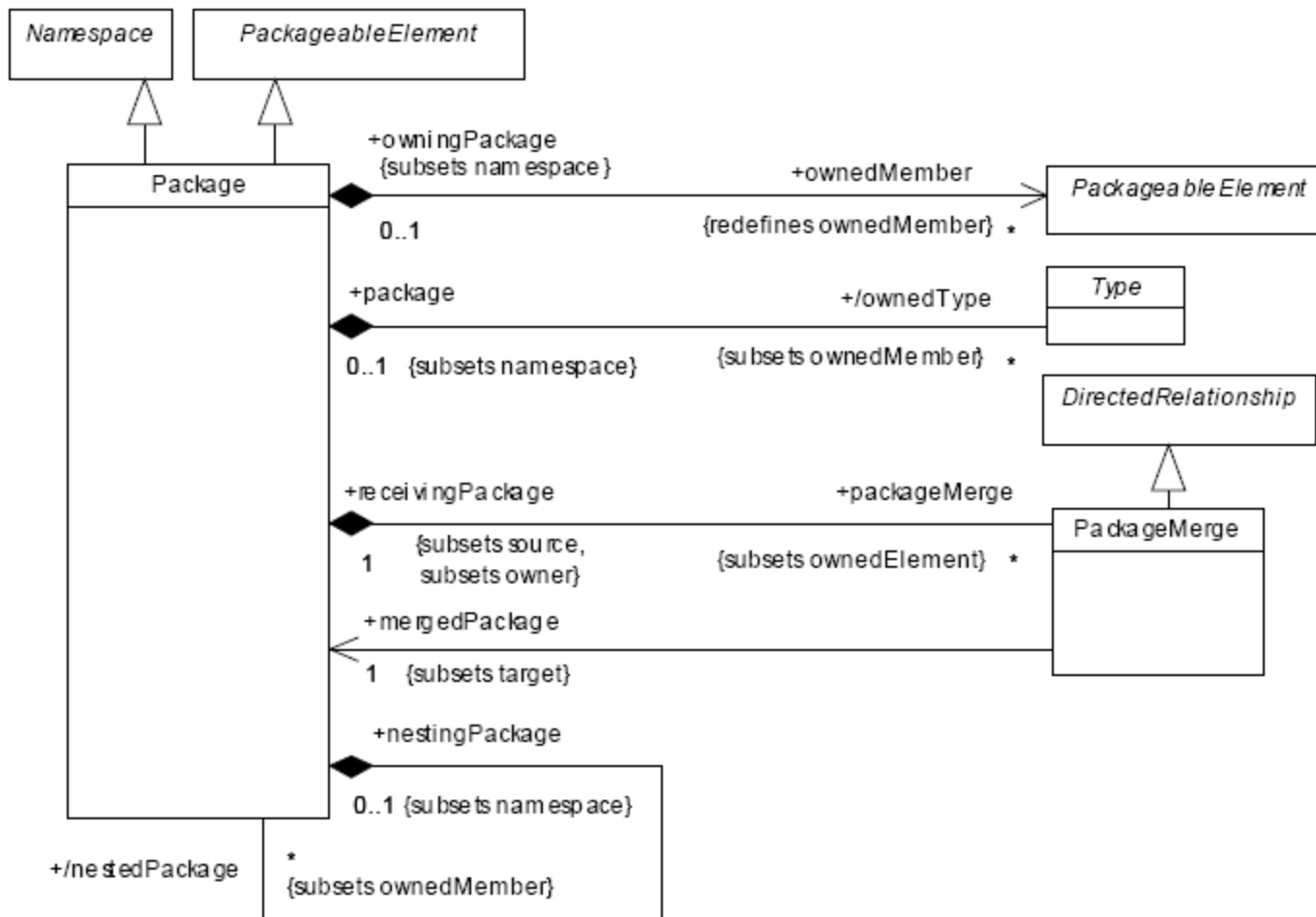
String tail = op.isAbstract() ? ";" : "{ \n    }";

ps.format("%n    %s %s %s(%s) %s %n",
          modifiers, returns, operationName, parameters, tail);
}
}
```

Представление пакетов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

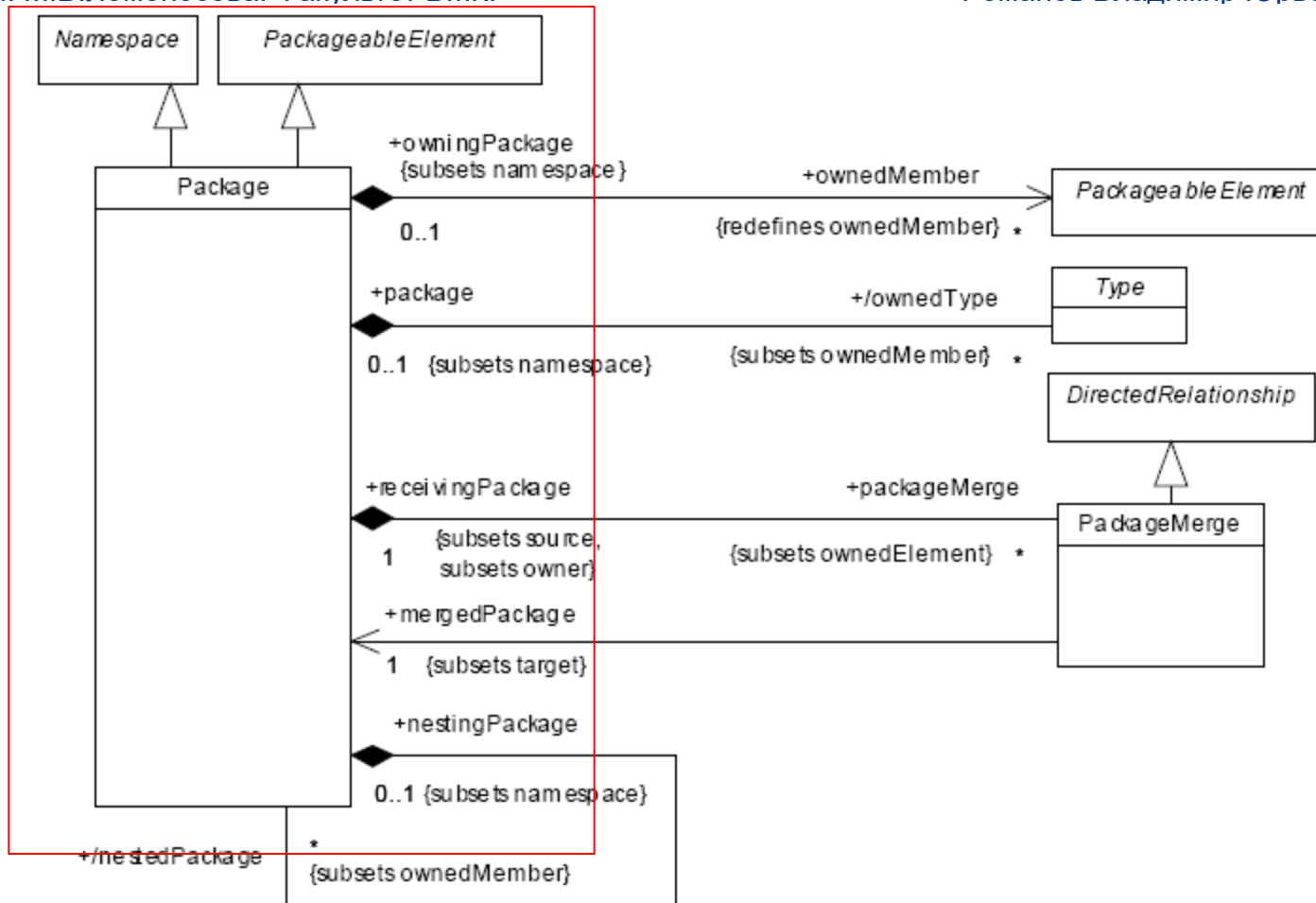
Романов Владимир Юрьевич ©2025



Представление пакетов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

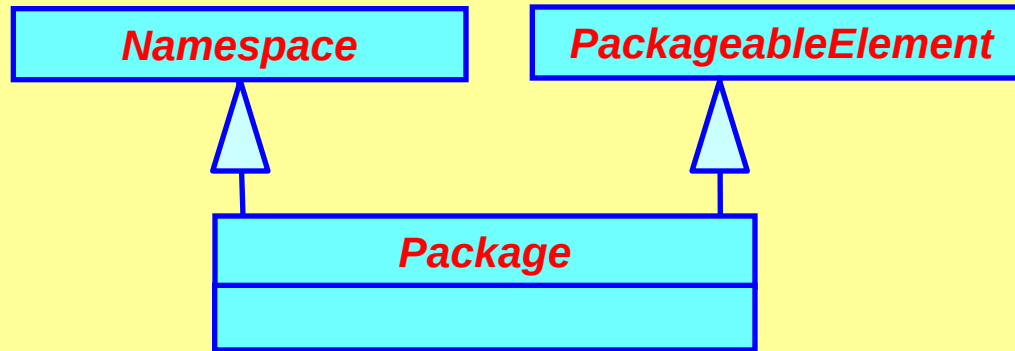
Романов Владимир Юрьевич ©2025



Моделирование классификаторов метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

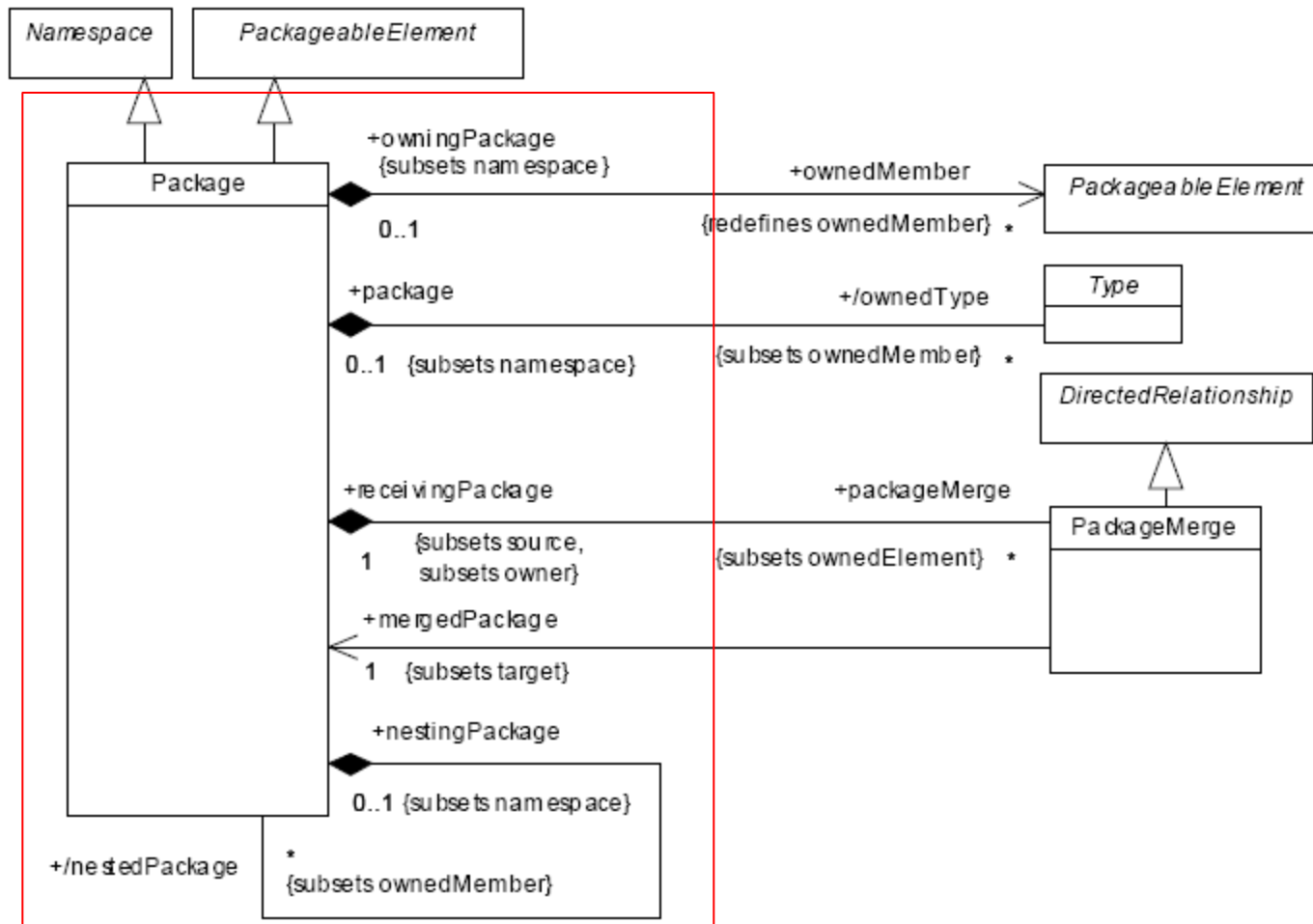
Романов Владимир Юрьевич ©2025



Представление пакетов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

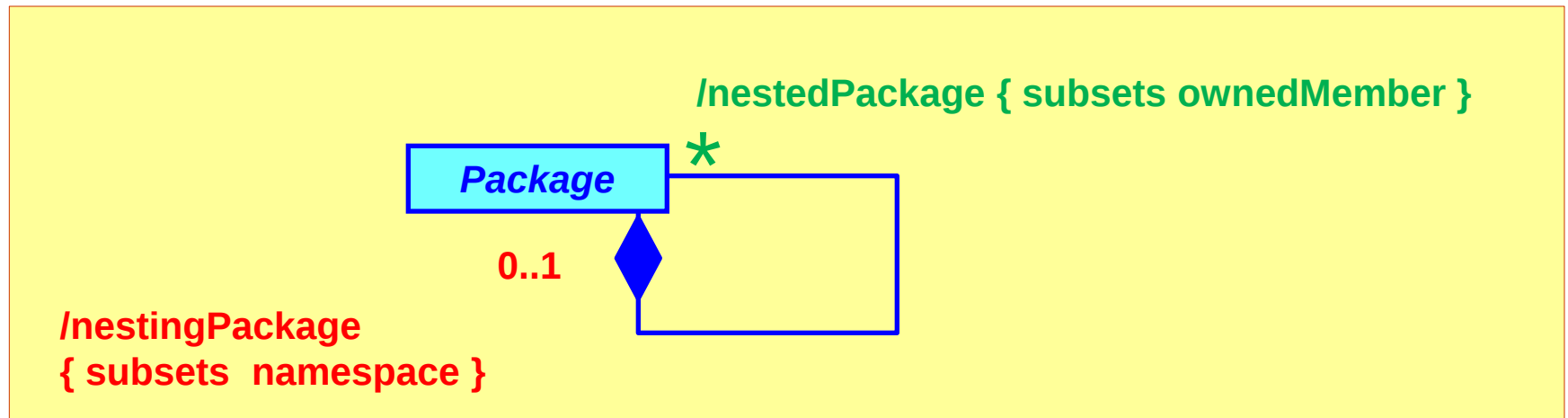
Романов Владимир Юрьевич ©2025



Моделирование отношения между пакетами

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

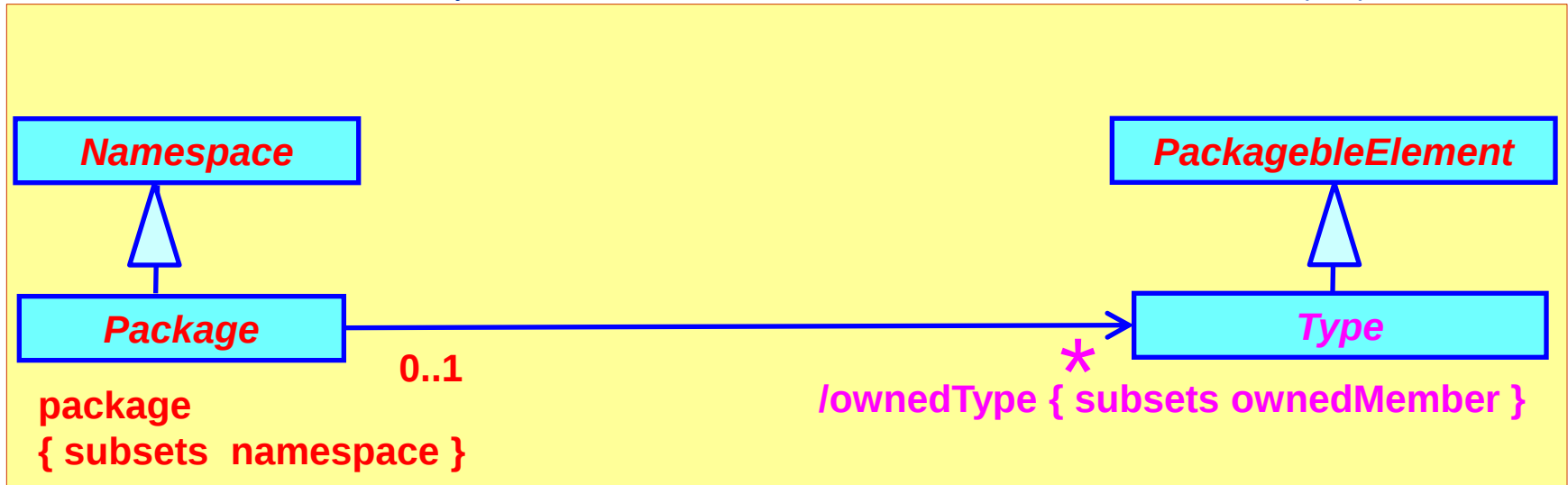


```
package graphics.core;
```

Моделирование пакетов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



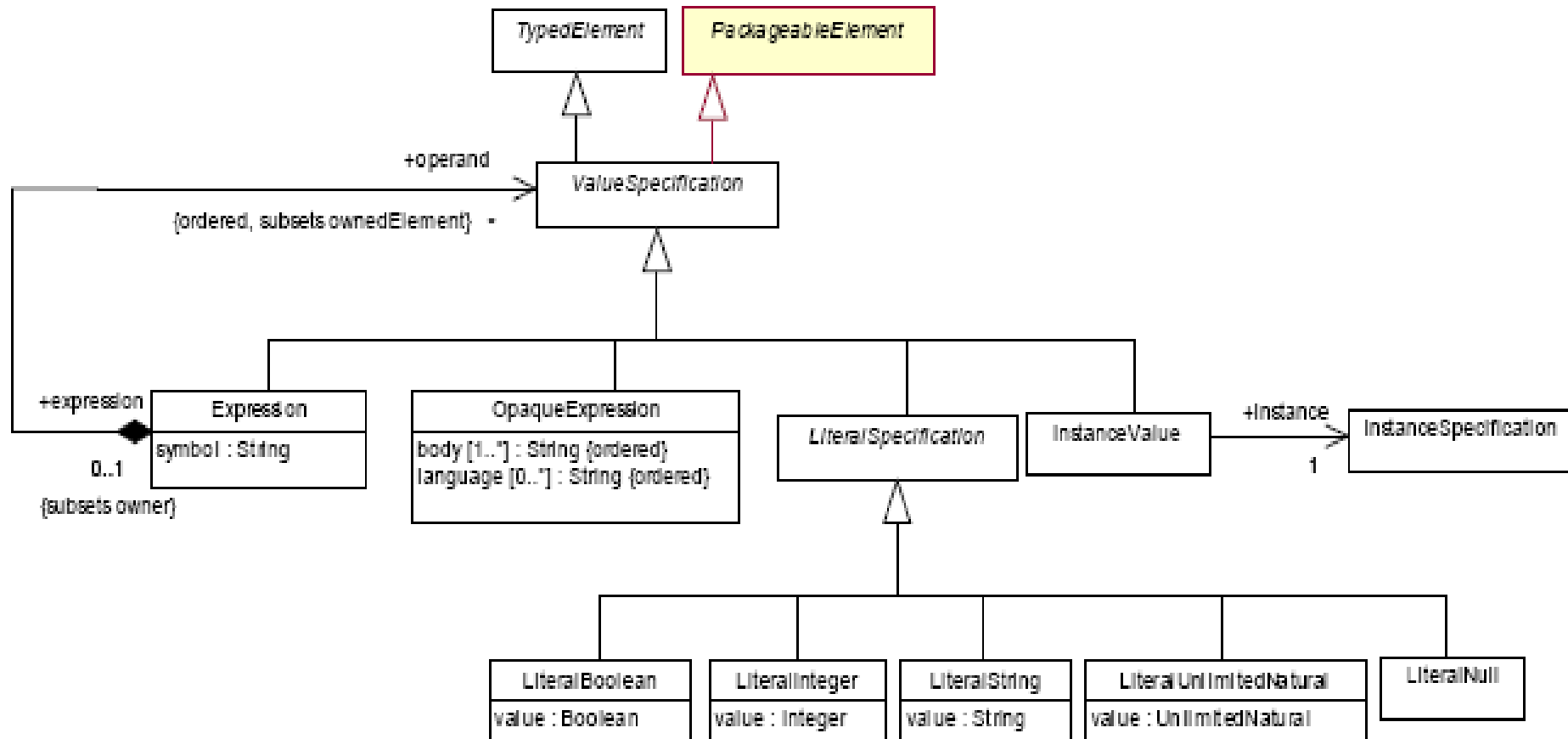
```
package graph;

class Node {
}
```

Представление значений в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

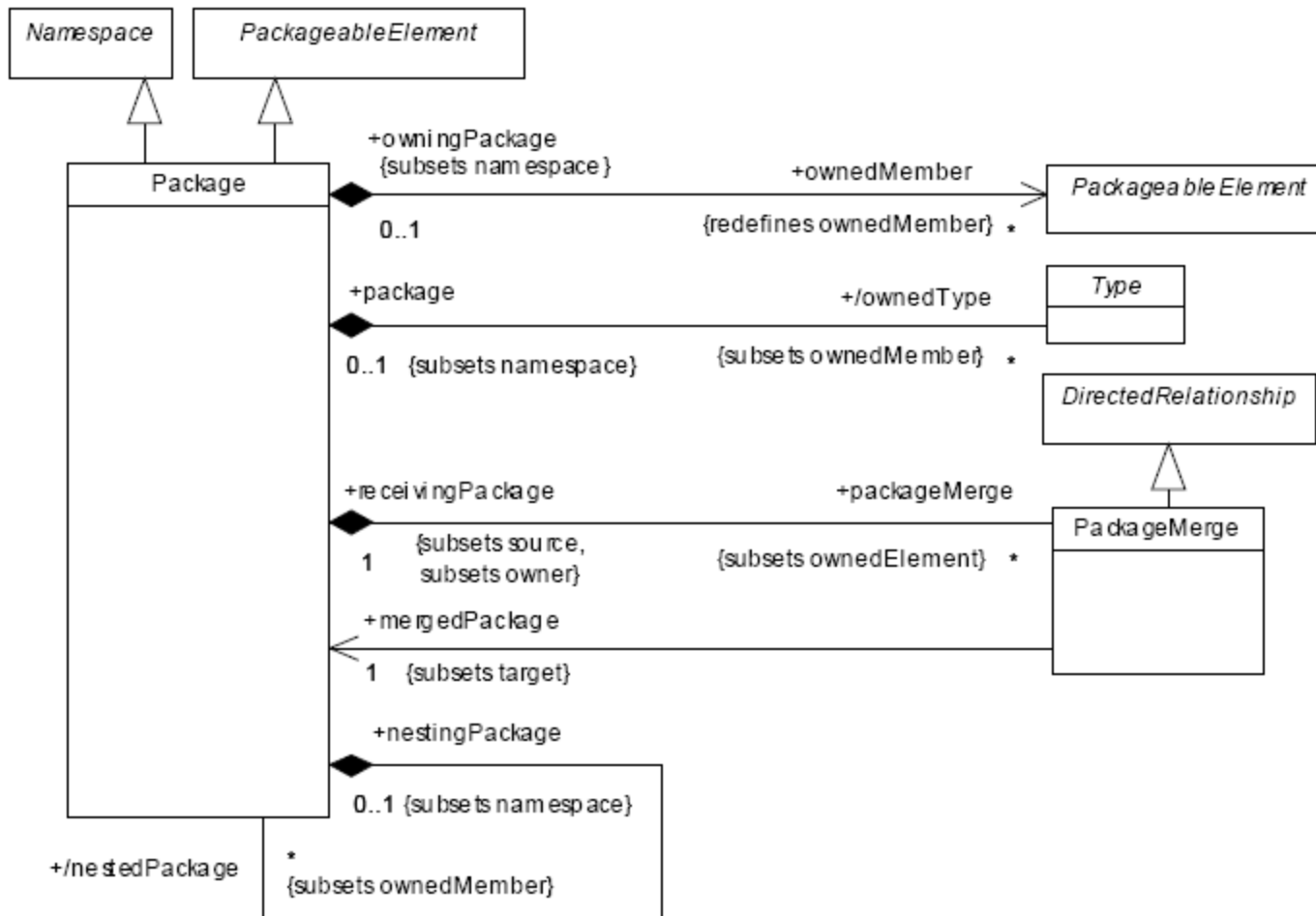
Романов Владимир Юрьевич ©2025



Представление пакетов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

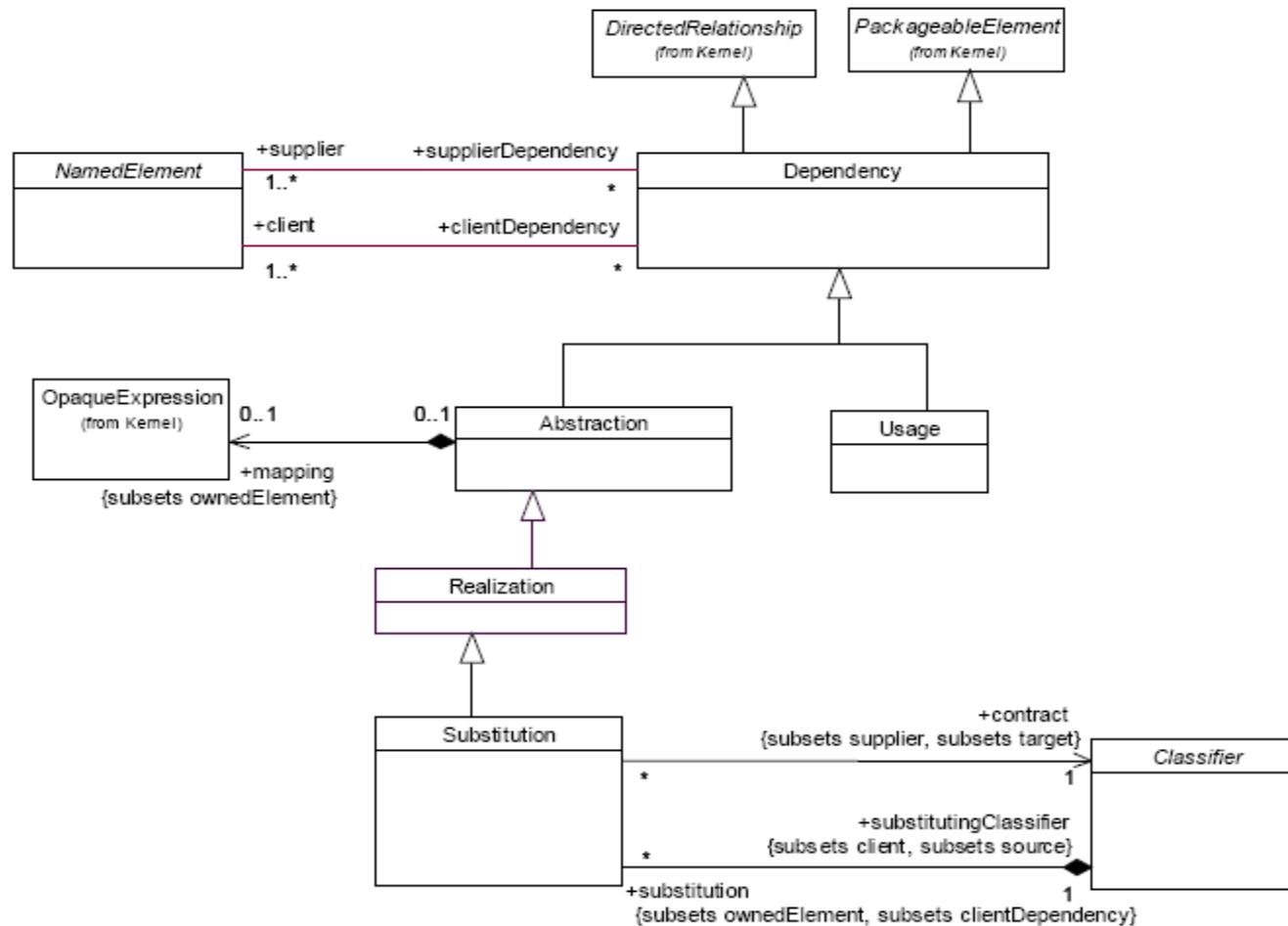
Романов Владимир Юрьевич ©2025



Представление отношений зависимости в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

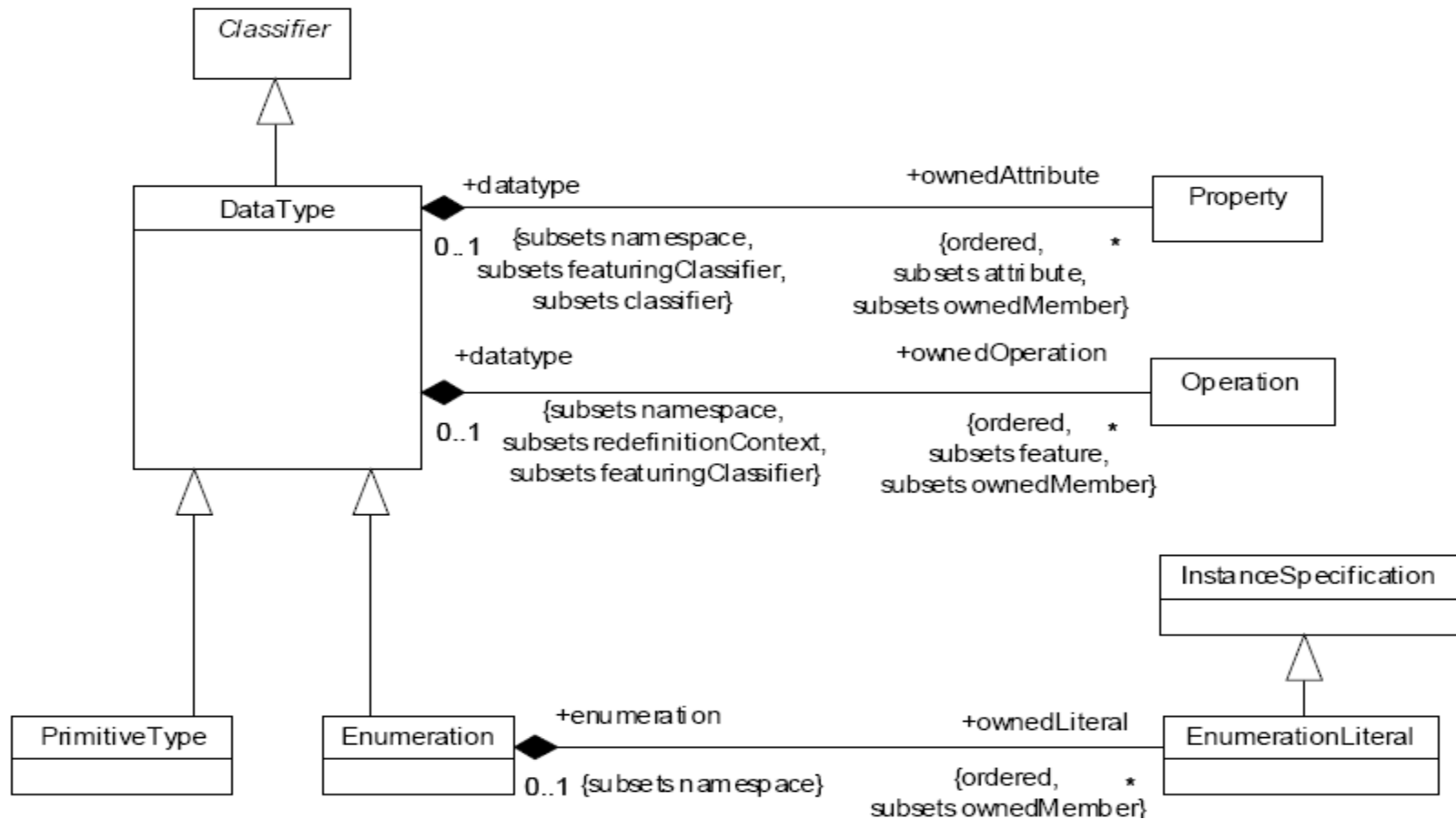
Романов Владимир Юрьевич ©2025



Представление типов данных и перечислений в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

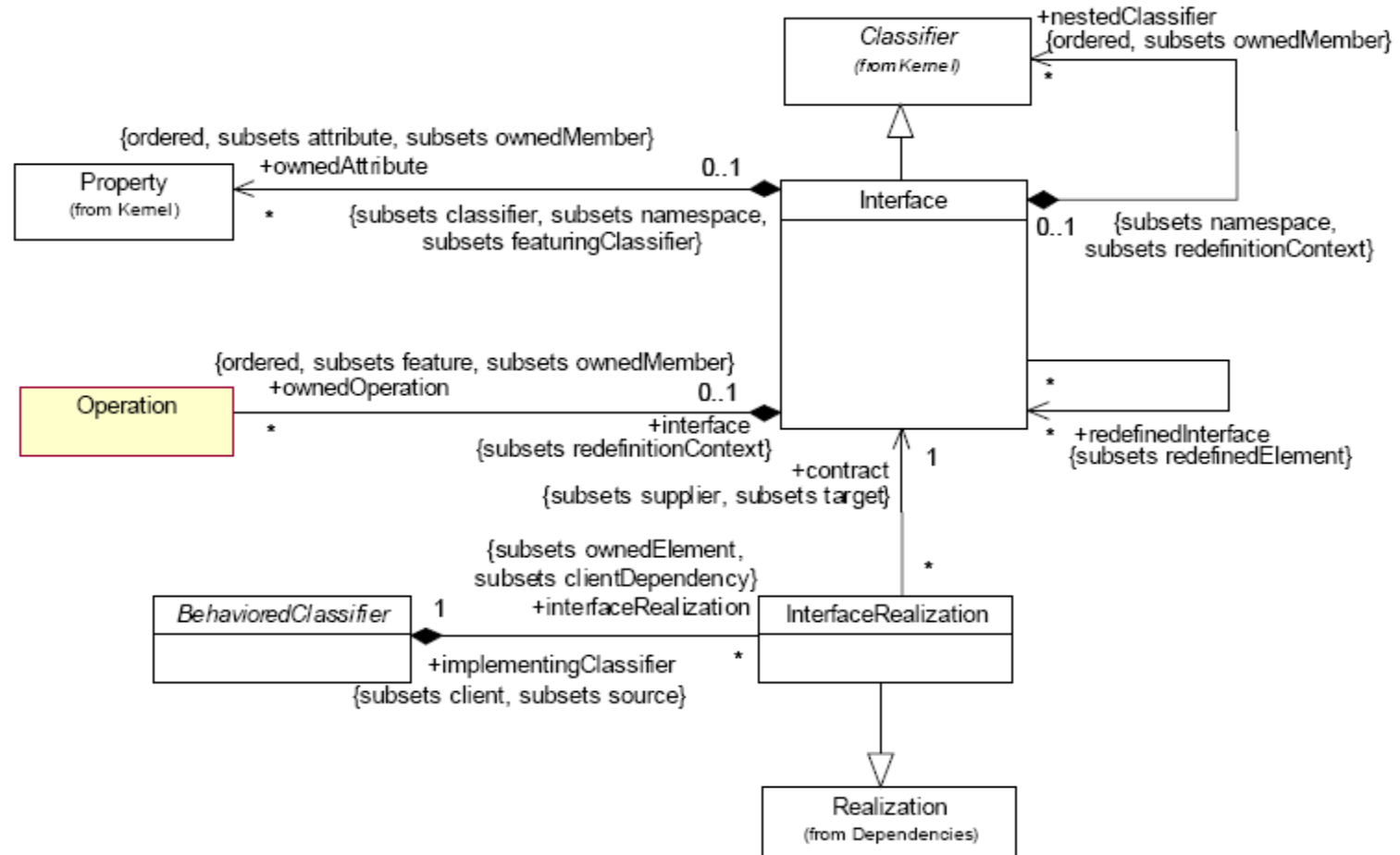
Романов Владимир Юрьевич ©2025



Представление интерфейсов в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Генерация текстов класса на языке Java.

Генерация реализуемых классом интерфейсов

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public static void generateClass (Class class_, String packageDir) {  
    // ...  
    genPackage(class_, ps);  
    genImports(class_, ps);  
  
    String modifiers = visibilityToJava(class_.getVisibility());  
    if (class_.isAbstract())  
        modifiers += "abstract ";  
    if (class_.isLeaf())  
        modifiers += "final ";  
  
    ps.format("%n%sclass %s ", modifiers, name);  
    genParents(class_, ps);  
  
    genInterfaces(class_, ps);  
  
    ps.format("{ %n");  
        genFields(class_, ps);  
        genMethods(class_, ps);  
    ps.format("} %n");  
    // ...  
}
```

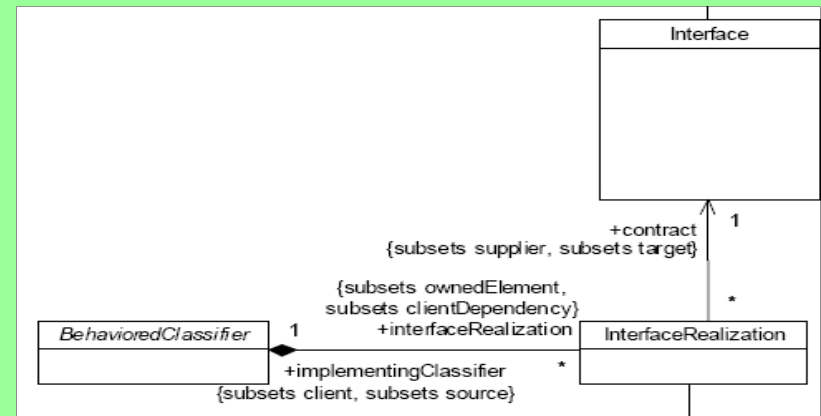
Генерация текстов класса на языке Java.

Генерация реализуемых классом интерфейсов

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private static void genInterfaces (Class cls, PrintStream ps) {  
    String interfaces = "";  
  
    for (InterfaceRealization ir : cls.getInterfaceRealizations()) {  
        Interface implementedInterface = ir.getContract();  
  
        if (interfaces.isEmpty())  
            interfaces = "implements ";  
        else interfaces += ", ";  
  
        interfaces += implementedInterface.getName();  
    }  
    ps.format(" %s ", interfaces);  
}
```



Генерация интерфейса

Генерация текстов интерфейса на языке Java

```
public static void generateInterface(Interface cls, String packageDir) {
    makePackageDir(packageDir);

    PrintStream ps = null;
    try { ps = new PrintStream(packageDir + "/" + cls.getName() + ".java"); }
    catch (FileNotFoundException e) { return; }

    genPackage(cls, ps);

    genImports(cls, ps);

    String modifiers = visibilityToJava (cls.getVisibility());

    ps.format("%n%sinterface %s ", modifiers, cls.getName());

    genParents(cls, ps);

    ps.format("{ %n");
        genFields(cls.getOwnedAttributes(), ps);
        genMethods(cls.getOwnedOperations(), ps);
    ps.format("} %n");

    ps.close();
}
```

Генерация текстов класса на языке Java.

Генерация полей класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private static void genFields (List<Property> attributes, PrintStream ps) {  
    for (Property f : attributes) {  
        String typeName = f.getType().getName();  
  
        String modifiers = visibilityToJava(f.getVisibility());  
        if (f.isStatic())  
            modifiers += "static";  
  
        String fieldName = f.getName();  
        ps.format("%s %s %s;%n", modifiers, typeName , fieldName );  
    }  
}
```

Генерация текстов класса на языке Java.

Генерация методов класса (1)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private static void genMethods (List<Operation> operations, PrintStream ps) {  
  
    for (Operation op : operations) {  
        String operationName = op.getName();  
  
        String modifiers = visibilityToJava(op.getVisibility());  
  
        if (op.isStatic())  
            modifiers += "static";  
  
        //...  
    }  
}
```

Генерация текста на языке JAVA

UML → JAVA
на KOTLIN

Генерация классификаторов пакета на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
fun Package.generatePackage(packageDir: String) {  
    makePackageDir(packageDir)
```

ownedMembers

```
.filter { !it.hasKeyword("unknown") }
```

```
.forEach {
```

```
    when (it) {
```

```
        is Class -> it.generateClass(packageDir)
```

```
        is Interface -> it.generateInterface(packageDir)
```

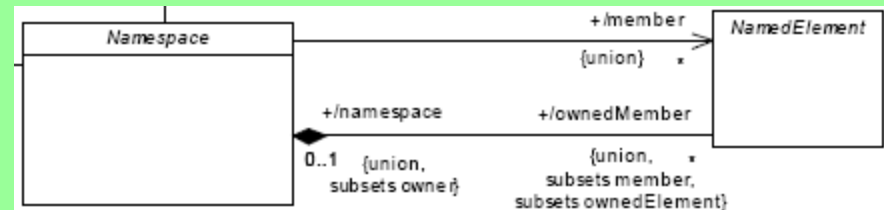
```
        is Enumeration -> it.generateEnumeration(packageDir)
```

```
        is Package -> it.generatePackage("$packageDir/${it.name}")
```

```
    }
```

```
}
```

```
}
```



Генерация классификаторов пакета на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
fun <T> Iterable<T>.joinToString
    (separator: CharSequence = ", ",
     prefix: CharSequence = "",
     postfix: CharSequence = "",
     limit: Int = -1,
     truncated: CharSequence = "...",
     transform: ((T) -> CharSequence)? = null)
    : String

people.joinToString(" ") { p: Person -> p.name }

people.joinToString(prefix="(", postfix=")")
    { p: Person -> "${p.name}${p.age}" }
```

Короткое квалифицированное имя Java и видимость именованного элемента

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val newLine: CharSequence = "\n"
```

```
private val VisibilityKind.asJava  
    get() = if (this == VisibilityKind.PACKAGE_LITERAL) "" else "$literal "
```

```
private val NamedElement.javaName: String  
    get() {  
        val longName = qualifiedName.replace("::", ".")  
        val k = longName.indexOf('.')  
        return longName.substring(k + 1)  
    }
```

<<enumeration>> VisibilityKind	
public	
private	
protected	
package	

NamedElement	
name : String	[0..1]
visibility : VisibilityKind	[0..1]
/ qualifiedName : String	[0..1]

Генерация класса

Генерация текстов класса на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
fun Class.generateClass(packageDir: String) {  
    val ps = createJavaFile(packageDir, name) ?: return
```

```
    ps.format("$packageAsJava%n%n")  
    ps.println(importsAsJava)
```

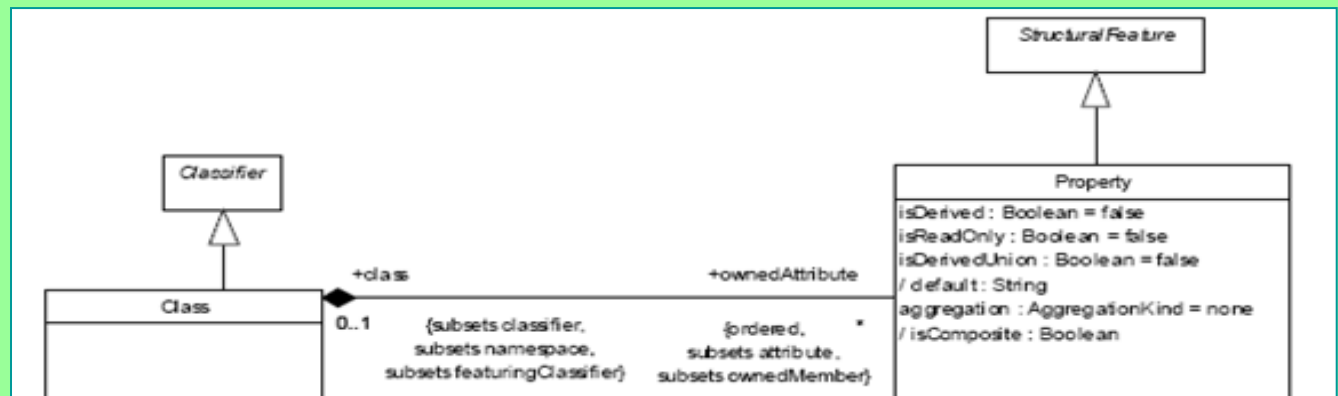
```
    ps.format("%n${modifiers}class $name$parentsAsJava$interfacesAsJava {%n%n")
```

```
    ownedAttributes.forEach { ps.println(it.propertyAsJava) }  
    ownedOperations.forEach { ps.println(it.operationAsJava) }
```

```
    ps.println("}")  
    ps.close()
```

```
}
```

NamedElement	
name : String	{0..1}
visibility : VisibilityKind	{0..1}
/ qualifiedName : String	{0..1}



Генерация пакета и импортов для класса на языке Java

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Classifier.packageAsJava
  get() = "package ${nearestPackage.javaName};"

private val Classifier.importsAsJava
  get() = importedMembers
    .map { "import ${it.javaName};" }
    .filter { !it.startsWith("import java.lang") }
    .joinToString(newLine)
```



Получение модификаторов класса на языке Java

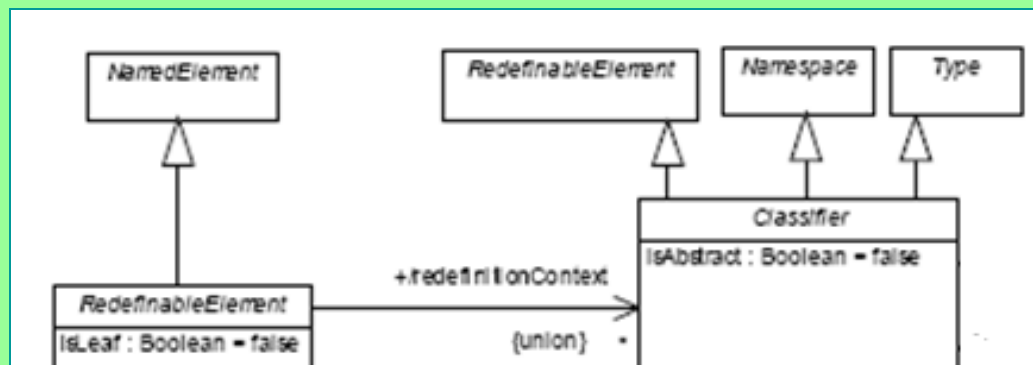
МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Class.modifiers: String
get() {
    var modifiers = visibility.asJava
    if (isAbstract) modifiers += "abstract "
    if (isLeaf) modifiers += "final "
    return modifiers
}
```

NamedElement	
name : String	[0..1]
visibility : VisibilityKind	[0..1]
/ qualifiedName : String	[0..1]

<<enumeration>> VisibilityKind	
public	
private	
protected	
package	



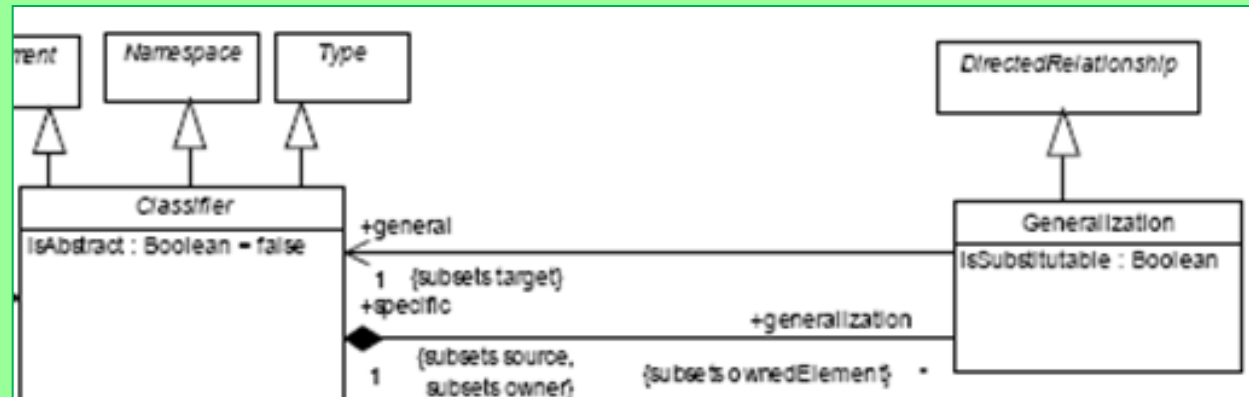
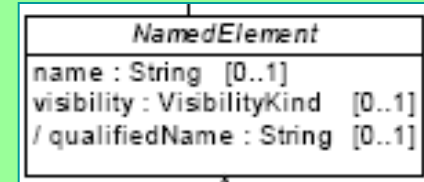
Генерация текстов класса на языке Java.

Генерация предков класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Classifier.parentsAsJava: String
get() {
    val parents = generalizations
        .map { it.general }
        .filter { !it.javaName.endsWith("java.lang.Object") }
        .joinToString { it.name }
    return if (parents.isNotEmpty()) " extends $parents" else ""
}
```



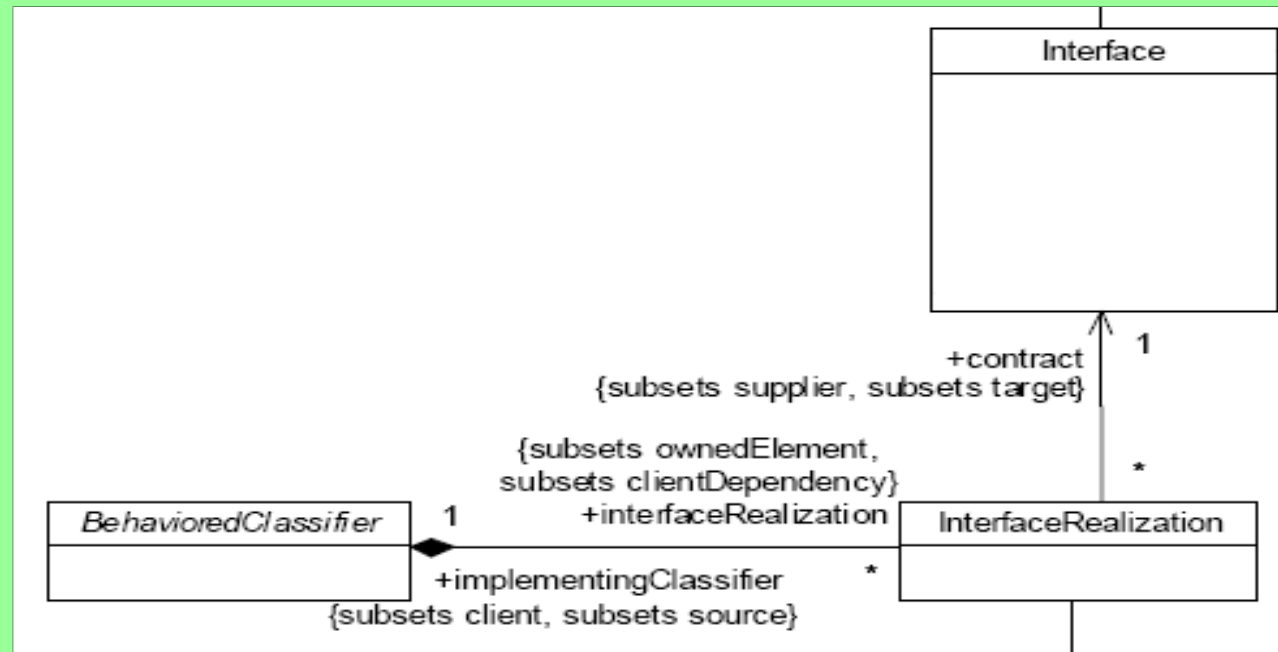
Генерация текстов класса на языке Java.

Генерация реализуемых классом интерфейсов

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Class.interfacesAsJava: String
get() {
    val implemented = interfaceRealizations.joinToString { it.contract.name }
    return if (implemented.isNotEmpty()) " implements $implemented" else ""
}
```



Генерация текстов класса на языке Java.

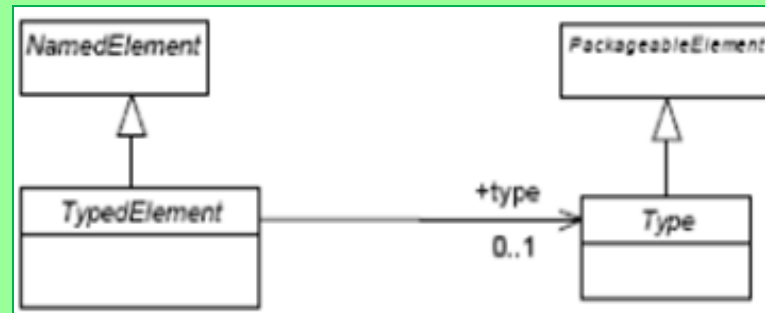
Генерация полей класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Property.propertyAsJava  
get() = "$modifiers${type.name} $name;"
```

```
private val Property.modifiers: String  
get() {  
    var modifiers = visibility.asJava  
    if (isStatic) modifiers += "static "  
    return modifiers  
}
```



Генерация текстов класса на языке Java.

Генерация методов класса (1)

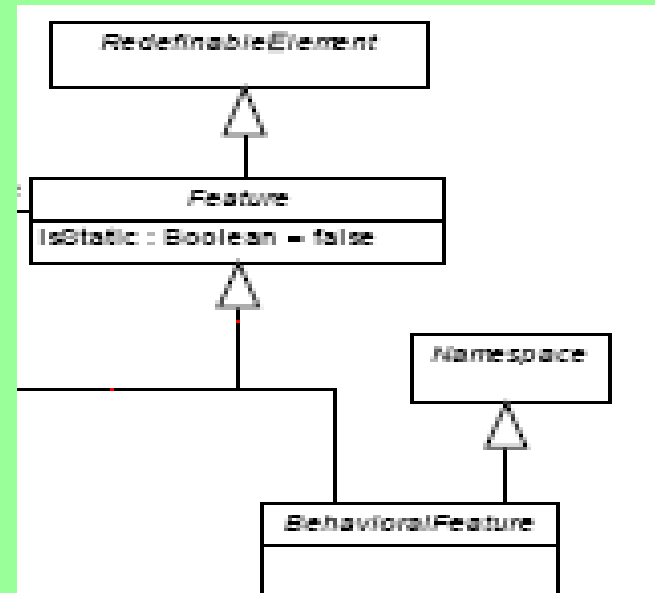
МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Operation.operationAsJava: String
get() {
    val returns = returnResult?.type?.name ?: "void"
    val tail = if (isAbstract) ";" else " {$newLine}$newLine"

    return "$modifiers$returns $name$parameters$tail"
}
```

```
private val Operation.modifiers: String
get() {
    var modifiers = visibility.asJava
    if (isStatic) modifiers += "static "
    if (isAbstract) modifiers += "abstract "
    return modifiers
}
```



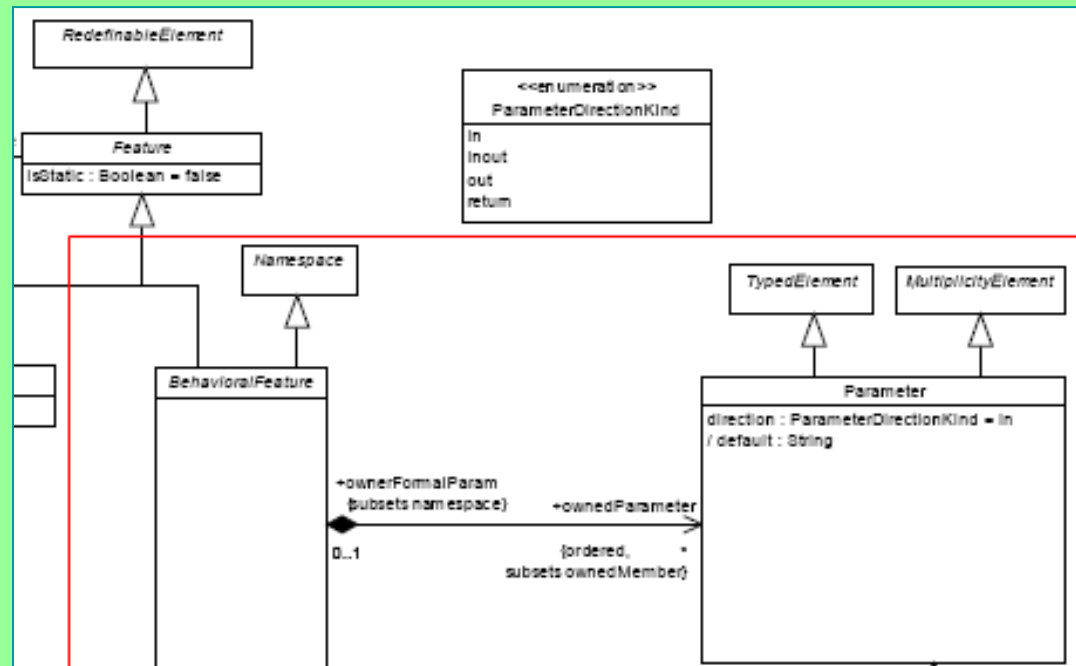
Генерация текстов класса на языке Java.

Генерация методов класса (2)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private val Operation.parameters
get() = ownedParameters
    .filter { it.direction != RETURN_LITERAL }
    .joinToString(prefix = "(", postfix = ")")
    { "${it.type.name} ${it.name}" }
```



Генерация интерфейса

Генерация текстов интерфейса на языке Java

```
fun Interface.generateInterface(packageDir: String) {
    val ps = createJavaFile(packageDir, name) ?: return

    ps.format("$packageAsJava%n")
    ps.println(importsAsJava)

    ps.format("%n${modifiers}interface $name$parentsAsJava {%n%n")

    ownedAttributes.forEach { ps.println(it.propertyAsJava) }
    ownedOperations.forEach { ps.println(it.operationAsJava) }
    ps.println("}")
    ps.close()
}

private val Interface.modifiers: String
    get() {
        var modifiers = visibility.asJava
        if (isLeaf) modifiers += "final "
        return modifiers
    }
```

Генерация перечислений

Генерация текстов интерфейса на языке Java

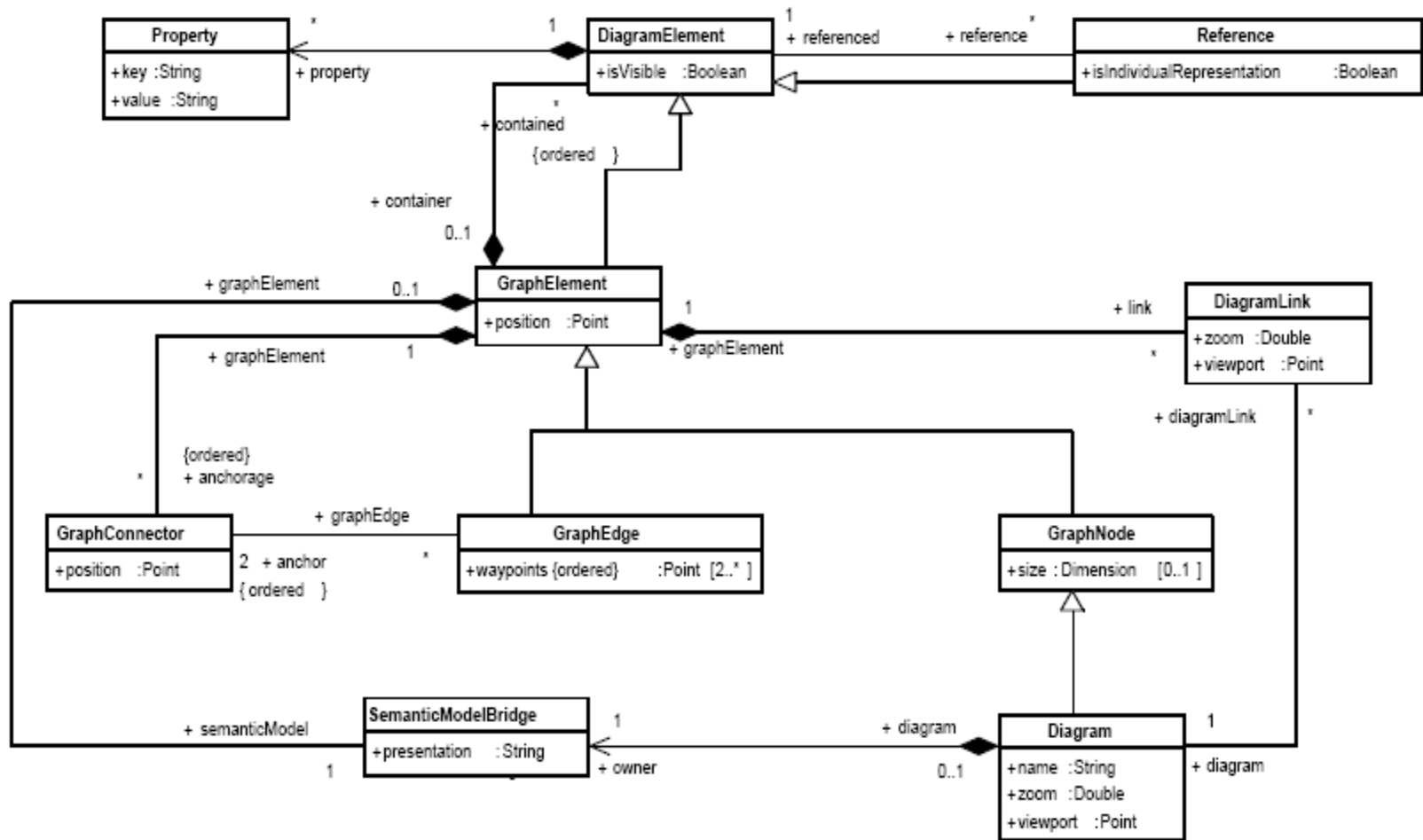
```
fun Enumeration.generateEnumeration(packageDir: String) {  
    val ps = createJavaFile(packageDir, name) ?: return  
  
    ps.format("$packageAsJava%n")  
    ps.println(importsAsJava)  
  
    ps.format("%n${modifiers}enum $name$parentsAsJava {%n%n")  
  
    ownedAttributes.forEach { ps.println(it.propertyAsJava) }  
    ownedOperations.forEach { ps.println(it.operationAsJava) }  
    ps.println("}")  
    ps.close()  
}
```

Представление диаграмм в метамодели языка UML

Представление основных элементов диаграмм в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

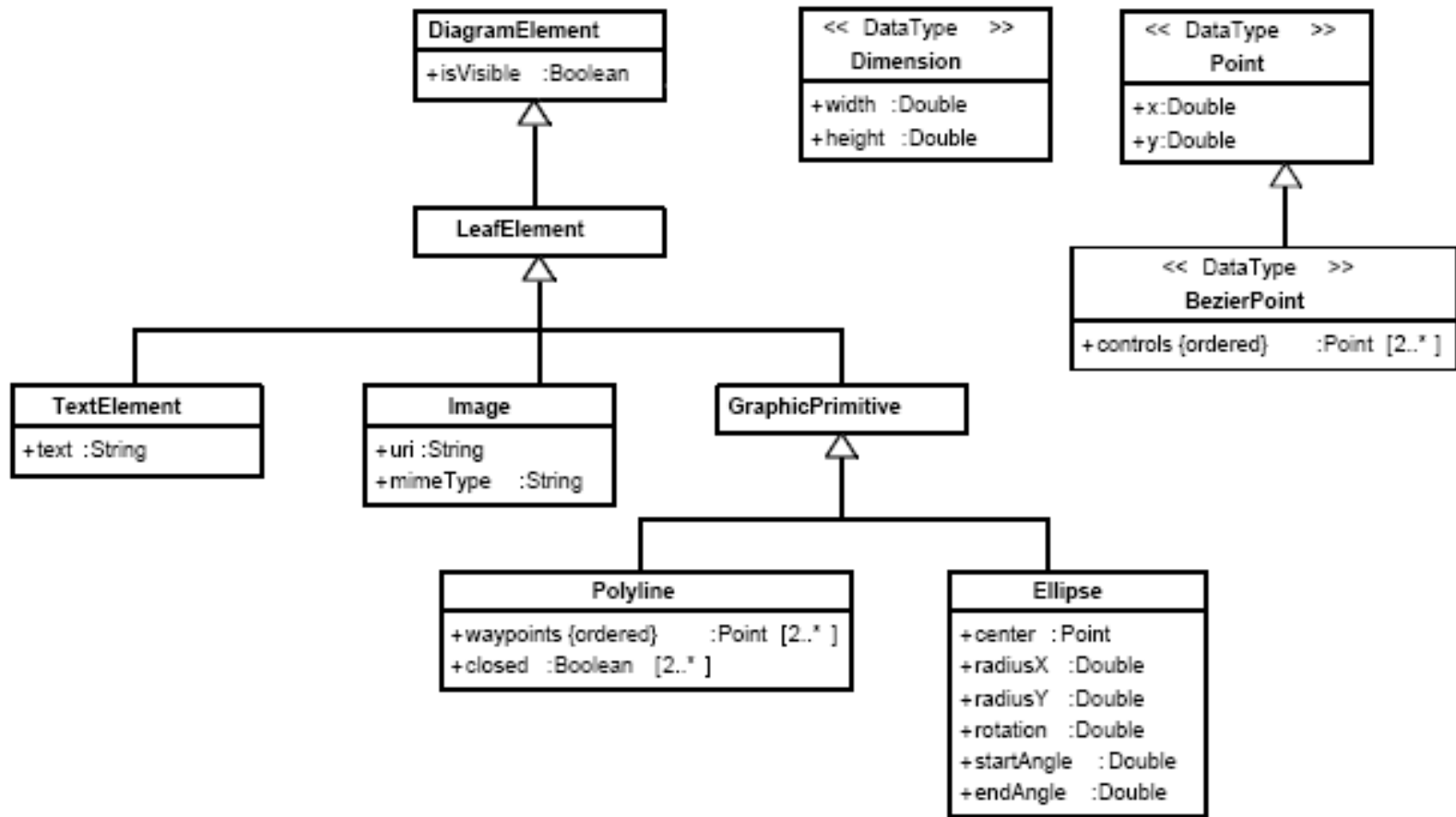
Романов Владимир Юрьевич ©2025



Представление листовых элементов диаграмм в метамодели языка UML

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Виды семантических связей между элементами UML диаграммы и UML модели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

