Task 3: Relation among (us) classes

Depth of the inheritance tree (DIT)

What is it?

- refers to the number of levels in a class hierarchy
- each level represents a class that inherits from another class
- it refers to the distance between a class and the root of the inheritance tree

Depth of the inheritance tree (DIT)

How do you measure it?

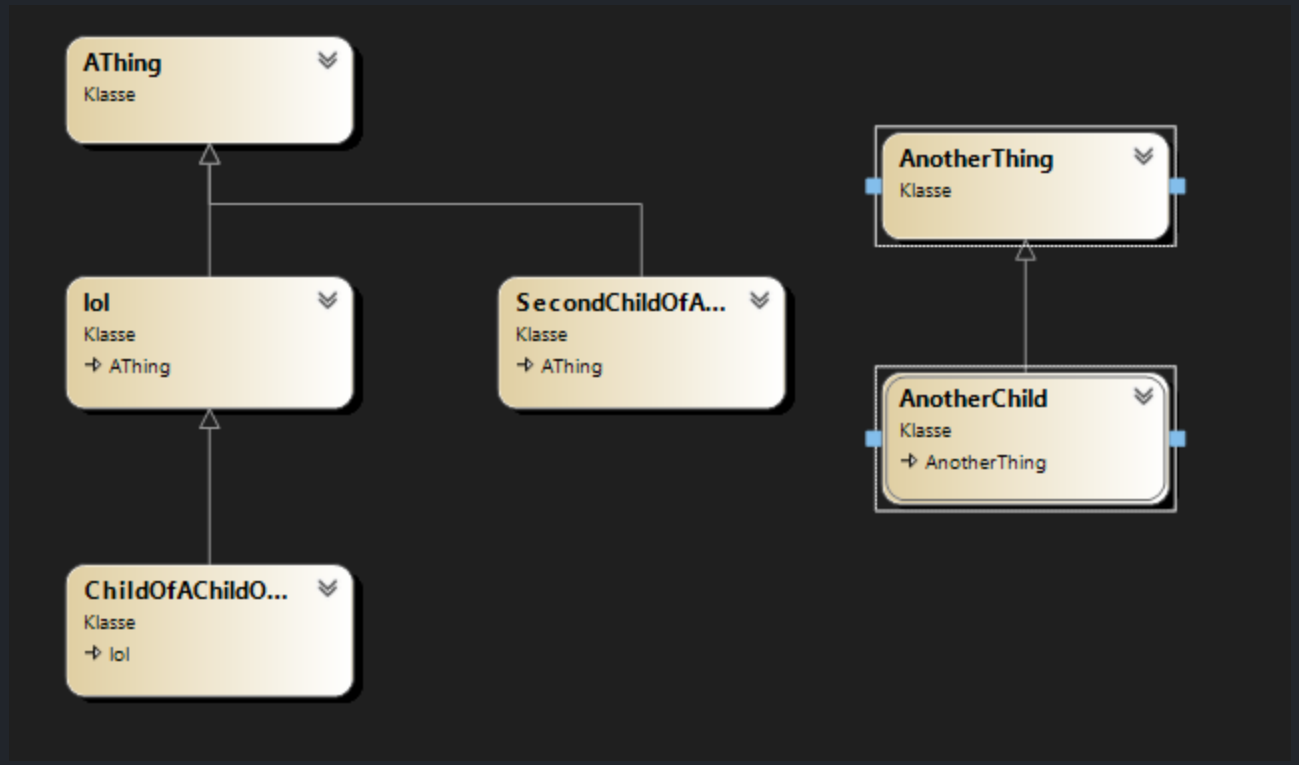
- Class diagramm → many IDEs have tools to generate class diagrams (e. g. Class Designer for Visual Studio)
- Tools that analyze the class hierarchy
- Manually by counting the level of inheritance for each class

Depth of the inheritance tree (DIT)

What is good?

- Low level of inheritance is good
- A high depth of inheritance can lead to a complex and rigid class hierarchy
- Changes to one class can have cascading effects on multiple other classes

Depth of the inheritance tree (DIT)



Number of children (NOC)

What is that?

- Number of immediate subclasses that inherit from one class
- They measure the width of the inheritance tree
- DIT and NOC are strongly correlated

Number of children (NOC)

What is that?

- Higher value => higher reusability of the class
- But: Risk of improper abstraction and misuse of subclassing

- The importance of testing increases as well

Number of children (NOC)

How to calculate them?

1. Identify the parent class or module for which you want to calculate the NOC.
2. Count the number of immediate sub-classes or sub-modules that inherit from the parent class or module.
3. This count is the NOC for the parent class or module.

=> NOC for a parent class or module

Number of children (NOC)

How to calculate them?

```
2 Verweise
public class ParentClass {
    // Stuff
}

0 Verweise
public class Child : ParentClass {
    // Stuff
}

0 Verweise
public class AnotherChild : ParentClass {
    // Stuff
}
```

1

2

=> NOC (ParentClass) = 2

Number of children (NOC)

How to calculate them?

NOC for a whole software:

1. Count the NOC for each parent class/module
2. Aggregate them

Number of children (NOC)

Automation with Software possible?

Yes.

For example with:

- SonarQube (Open Source)
- Understand (Commercial)

Method Inheritance Factor (MIF)

"Measure the level of inheritance of methods in all classes"

$$\frac{\text{Number of Inherited Methods}}{\text{Total Number of available Methods}}$$

- Level of Reuse
- Assessment in Testing needed

Method Inheritance Factor (MIF)

```
1 Verweis
class BaseClass {
    1 Verweis
    public virtual void Foo() { }

    0 Verweise
    public virtual void Bar() { }
}
```

```
1 Verweis
class DerivedClass : BaseClass {
    0 Verweise
    public void Baz() { }

    0 Verweise
    public void FooBar() { }
}
```

Number of inherited Methods = 2
Total number of Methods = 4

MIF = 2/4 = 0.5

Method Inheritance Factor (MIF)

```
using System.Reflection;

float totalMethods = 0;
float inheritedMethods = 0;

Type? type = typeof(DerivedClass);
while (type != null) {
    totalMethods += type.GetMethods().Length;
    inheritedMethods += type
        .GetMethods(BindingFlags.Instance | BindingFlags.NonPublic | BindingFlags.Public | BindingFlags.DeclaredOnly)
        .Length;
    type = type.BaseType;
}

float mif = inheritedMethods / totalMethods;
Console.WriteLine($"Inherited Methods: {inheritedMethods}");
Console.WriteLine($"Total Methods: {totalMethods}");
Console.WriteLine($"Method Inheritance Factor (MIF) = {mif}");
```

Method Inheritance Factor (MIF)

```
1   using System.Reflect
2
3   float totalMethods =
4   float inheritedMetho
5
6   Type? type = typeof(
7   while (type != null)
8       totalMethods +=
9       inheritedMethods
10          .GetMethods(
11          .Length;
12      type = type.Base
13  }
14
15  float mif = inherite
16  Console.WriteLine($"
17  Console.WriteLine($"
18  Console.WriteLine($"
19
20
```

Microsoft Visual Studio-Debugging-Konsole

```
Inherited Methods: 10
Total Methods: 20
Method Inheritance Factor (MIF) = 0,5

C:\Users\Service\Desktop\Programmieren\c#\TestApp\TestApp\
"0" beendet.
Um die Konsole beim Beenden des Debuggens automatisch zu
"Konsole beim Beenden des Debuggings automatisch schließe
Drücken Sie eine beliebige Taste, um dieses Fenster zu sc
```

```
dingFlags.DeclaredOnly)
```

Response for a class (RFC)

The total number of methods that can potentially be executed in response to a message received by an object of a class.

Given a class, its RFC is the addition of:

- ClassMethod elements
- Method elements
- References to any Method/ClassMethod (including self calls)

On next example, RFC is 7. The elements for the RFC are remarked:

```
ClassMethod CreateProjection(cls As %String, ByRef params) As %Status
{
    set ns=$namespace
    new $namespace
    znspace "%SYS"

    if ('##class(Security.Applications).Exists(..#CSPAPP)) {
        do ##class(Security.System).GetInstallationSecuritySetting(.security)
        set cspProperties("AutheEnabled") = $select((security="None"):64,1:32)
        set cspProperties("NameSpace") = ns
        set cspProperties("Description") = ..#CSPAPPDESCRIPTION
        set cspProperties("DispatchClass") = ..#ROUTER
        write !, "Creating WEB application """_..#CSPAPP_"""..."
         $$$ThrowOnError(##class(Security.Applications).Create(..#CSPAPP, .cspProperties))
        write !, "WEB application """_..#CSPAPP_""" created."

        if ##class(%Studio.General).GetWebServerPort(,,,.url) {
             write !, "You can now open it with a link: "_url_$p(..#CSPAPP,"/",2,*)_"/"
        }
    } else {
        write !, "WEB application """_..#CSPAPP_""" already exists, so it is ready to use."
    }
     Quit $$$OK
}

/// This method is invoked when a class is 'uncompiled'.
ClassMethod RemoveProjection(cls As %String, ByRef params, recompile As %Boolean) As %Status
{
    new $namespace
    znspace "%SYS"

    if (##class(Security.Applications).Exists(..#CSPAPP)) {
        w !, "Deleting WEB application """_..#CSPAPP_"""..."
        do ##class(Security.Applications).Delete(..#CSPAPP)
        w !, "WEB application """_..#CSPAPP_""" was successfully removed."
    }
     QUIT $$$OK
```

Response for a class (RFC)

Automatic Measuring

- In Sonar, from the project dashboard

average RFC for the Apache commons-lang project: