Gendarme

Szabály alapú statikus forráskód-verifikáció

Zakál Dávid
zakal@aut.bme.hu

Áttekintés

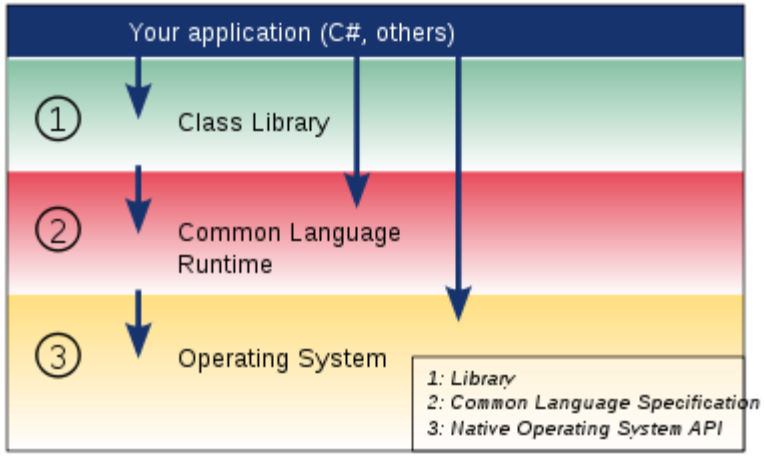
- Bevezetés
- Gendarme működése
- Konkrét példák
- Konklúziók

Gendarme: bevezetés 1.

- Egy kiterjeszthető, szabály alapú ellenőrző eszköz .NET komponensek ellenőrzéséhez
- Olyan gyakori hibaminták keresése, amiket a fordítók és IDE-k általában nem keresnek
- Ellenőrzés bemenete: CIL köztes nyelvű kód
  - ISO/IEC 23271:2006 illetve ECMA-335 szabványban definiált
  - Kimondottan továbbfordításra, nem pedig közvetlen végrehajtásra tervezett nyelv (elméleti alapja a kiértékelő verem, regisztereket nem ismer)

Gendarme: bevezetés 2.

- Gendarme kezdetek: Sebastien Pouliot (Mono biztonsági audit) és Aaron Tomb (Google Summer of Code)
- *Mono*: CIL kódot futtató, nyílt forrású runtime és keretrendszer; gyakorlatilag egy több platformra elérhető .NET implementáció
  - Megbízható, nagy teljesítményű, jó MS kompatibilitás
  - Mono project: jóval több mint framework
  - Ximian, Novell, jelenleg pedig Xamarin gondozza



A Gendarme eszköz

- *Metaadat-bejárás* alapú statikus elemző (nincs komolyabb vezérlésáramlás és adatáramlás analízis)
- *Mono.Cecil* könyvtárra épül (CIL formátum magas szintű manipulációja)
- Az antimintákat és hibamintákat leíró szabályokat CIL szerelvényekből (assembly) veszi
  - Kiterjeszthetőség, rugalmasság
- Felhasználói interfészek
  - Konzol alkalmazás
  - Grafikus front-end, varázsló
  - Más alkalmazásokba integrálva (CruiseControl.NET, NAnt, stb.)

Szabályok 1.

- Szakterület-specifikus (domain-specific) programozási antiminták, ahol a szakterület a .NET (pontosabban CIL illetve CLI) platform
- Gyakori tévedések, feledékenység okozta mulasztások, valamint (látszólag) kozmetikai apróságok
  - Vannak látszólag lényegtelen (nem hibát leíró) szabályok is, de hosszú távú SW karbantarthatóság ill. evolúció szempontjából számítanak (gyengébb *change tolerance*)
  - Akár tervezési antimintákra (*design antipatterns*) is lehet következtetni a jelentésekből
  - Pl. az *AvoidLackOfCohesionOfMethodsRule* és az *AvoidSpeculativeGeneralityRule* jelezheti a *Poltergeist* szabványos antiminta előfordulását

Szabályok 2.

- 254 szabály (jelenleg), 17 kategóriába szedve
- Különböző *súlyosság*
- Előfordulás érzékelése esetén *konfidencia*
- Kategóriák
  - *BadPractice, Concurrency, Correctness, Design, Design.Generic, Design.Linq, Exceptions, Interoperability, Maintainability, Naming, Performance, Portability, Security, Security.Cas, Serialization, Smells, Ui*

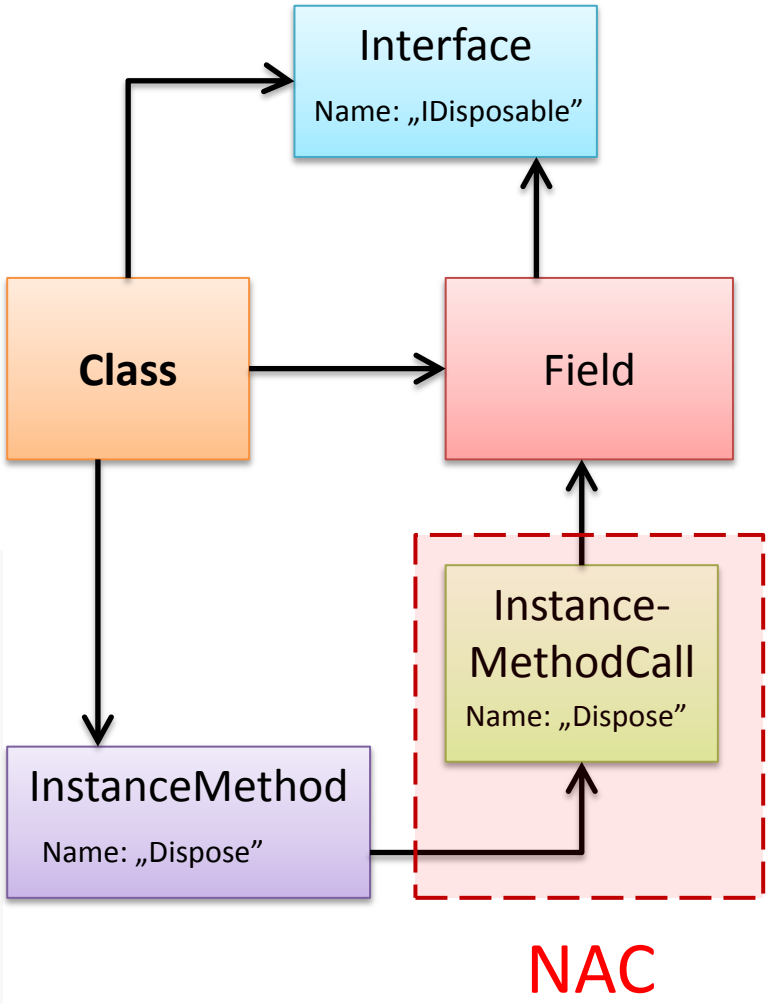
Szabály példa 1.

- Egyszerű szakterület-sp. szabály:
  DisposableFieldsShouldBeDisposedRule
- Szabály megvalósítása:
  DisposableFieldsShouldBeDisposedRule.cs,
  300 LOC

Lehetséges találat környezete a kódban:

```
class DoesNotDisposeMember : IDisposable {
    byte[] buffer;
    IDisposable field;

    public void Dispose ()
    {
        buffer = null;
        // field is not disposed
    }
}
```



Szabály példa 2.

- Szabály implementációja

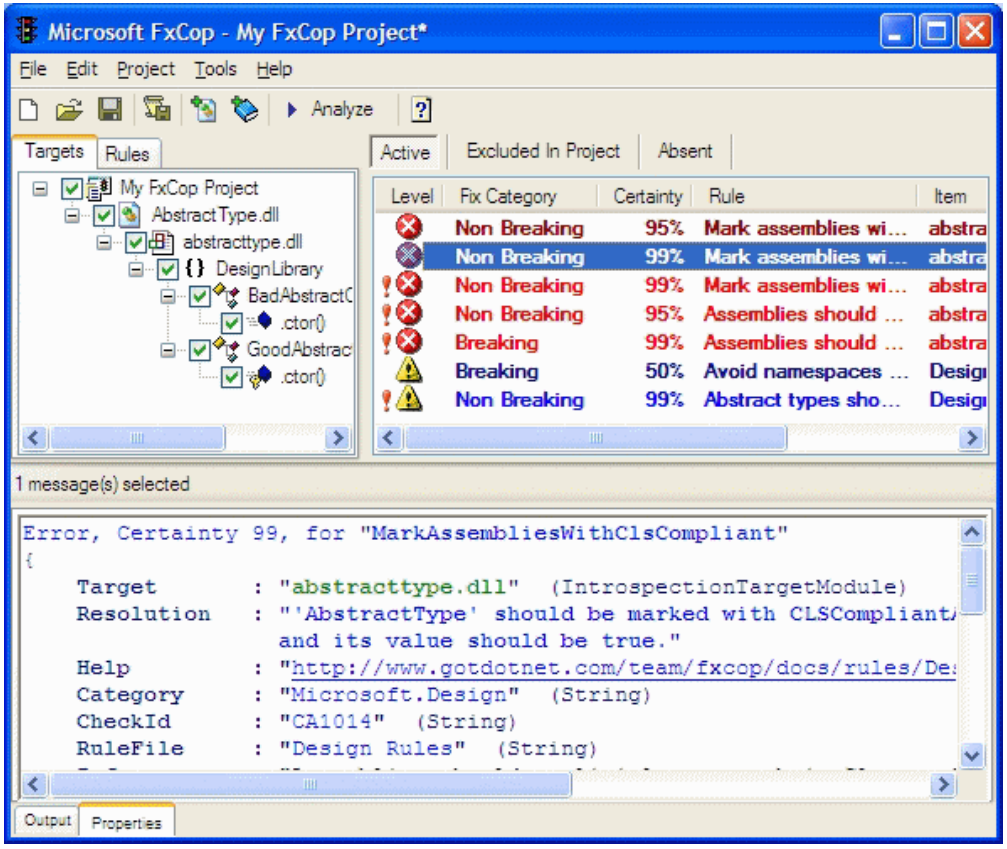
```
using Mono.Cecil;
using Mono.Cecil.Cil;

using Gendarme.Framework;
using Gendarme.Framework.Engines;
using Gendarme.Framework.Helpers;
using Gendarme.Framework.Rocks;

namespace Gendarme.Rules.Correctness
{
    [Problem("This type contains disposable field(s) which aren't disposed.")]
    [Solution("Ensure that every disposable field(s) is correctly disposed.")]
    [FxCopCompatibility("Microsoft.Usage", "CA2213:DisposableFieldsShouldBeDisposed")]
    public class DisposableFieldsShouldBeDisposedRule : Rule, ITypeRule
    {
        private List<FieldDefinition> disposeableFields = new List<FieldDefinition>();
        private static MethodDefinition GetNonAbstractMethod(TypeReference type,
            MethodSignature signature)...
        public RuleResult CheckType(TypeDefinition type)...
        private void CheckBaseDispose(TypeDefinition type, MethodDefinition implicitDisposeMethod,
            MethodDefinition explicitDisposeMethod)...
        private void CheckIfBaseDisposeIsCalled(MethodDefinition method, MemberReference baseMethod)...
        static readonly MethodSignature DisposeBool = new MethodSignature("Dispose", "System.Void",
            new string[] { "System.Boolean" });
        private void CheckIfAllFieldsAreDisposed(MethodDefinition method,
            ICollection<FieldDefinition> fields)...
        private static void ProcessMethod(MethodDefinition method,
            ICollection<FieldDefinition> fieldsToDispose)...
    }
}
```

Kitérő: Microsoft FxCop

- Gendarme-hoz hasonló, elsősorban .NET osztálykönyvtárakhoz szánják



Szabály feldolgozása 1.

- 1. lépés: szabályok betöltése a szerelvényekből

```
private void LoadRulesFromAssembly(string assemblyName)
{
    AssemblyName aname = AssemblyName.GetAssemblyName(Path.GetFullPath(assemblyName));
    Assembly a = Assembly.Load(aname);
    foreach (Type t in a.GetTypes())
    {
        if (t.IsAbstract || t.IsInterface)
            continue;

        if (t.FindInterfaces(RuleTypeFilter, "Gendarme.Framework.IRule").Length > 0)
        {
            rules.Add((IRule)Activator.CreateInstance(t));
        }
    }
    assembly_rules = rules.OfType<IAssemblyRule>();
    type_rules = rules.OfType<ITypeRule>();
    method_rules = rules.OfType<IMethodRule>();
}
```

Szabály feldolgozása 2.

- 2. lépés: szabályok futtatása

```
public virtual void Run()
{
    RunnerEventArgs runner_args = new RunnerEventArgs(this);
    foreach (AssemblyDefinition assembly in assemblies)
    {
        currentTarget = (IMetadataTokenProvider)assembly;
        runner_args.CurrentAssembly = assembly;
        OnAssembly(runner_args);

        foreach (ModuleDefinition module in assembly.Modules)
        {
            currentTarget = (IMetadataTokenProvider)module;
            runner_args.CurrentModule = module;
            OnModule(runner_args);

            foreach (TypeDefinition type in module.GetAllTypes())
            {
                currentTarget = (IMetadataTokenProvider)type;
                runner_args.CurrentType = type;
                OnType(runner_args);

                foreach (MethodDefinition method in type.Methods)
                {
                    currentTarget = (IMetadataTokenProvider)method;
                    runner_args.CurrentMethod = method;
                    OnMethod(runner_args);
                }
            }
        }
    }
}
```

Szabály feldolgozása 3.

- Pl. szabályok alkalmazása .NET típuson

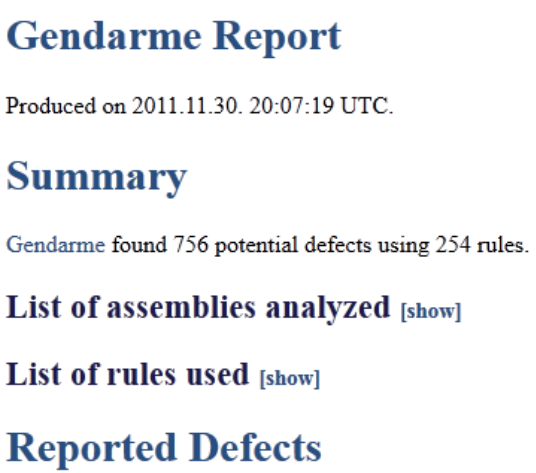
```
protected virtual void OnType(RunnerEventArgs e)
{
    OnEvent(AnalyzeType, e);

    foreach (ITypeRule rule in type_rules)
    {
        // ignore the rule on some user defined types
        if (IgnoreList.IsIgnored(rule, e.CurrentType))
            continue;

        currentRule = rule;
        rule.CheckType(e.CurrentType);
    }
}
```

Szabály feldolgozása 4.

- 3. lépés: jelentés véglegesítése
- Kimeneti célok
  - stdout
  - XML
  - HTML



Gendarme Report

Produced on 2011.11.30. 20:07:19 UTC.

Summary

Gendarme found 756 potential defects using 254 rules.

List of assemblies analyzed [show]

List of rules used [show]

Reported Defects

Table of contents

Hatékonyság növelése

- Mérnöki megoldás a metaadat alapú bejárás teljesítményének javítására: *OpCodeEngine*
  - Metódusonkénti bitmaszk a metódus által használt CIL utasításkódokról
  - Segíti a szabályok illesztését a metódusokra a szabályok feltételeinek hatékony ellenőrzésével

```
public class OpCodeEngine : Engine
{
    void OnMethodBody(object sender, EngineEventArgs e)
    {
        MethodBody body = (sender as MethodBody);
        MethodDefinition method = body.Method;

        OpCodeBitmask mask = new OpCodeBitmask();
        foreach (Instruction ins in body.Instructions)
        {
            mask.Set(ins.OpCode.Code);
        }
        bitmasks.Add(method, mask);
    }
}
```

Konklúziók

- Gendarme: FOSS, több platform, CIL formátum
- Nem ad igazolt helyességet, hanem ellenjavallt mintákat és gyakori hibákat keres
  - Nem teljes
  - …de: kiváló .NET szakterület-specifikus checklist
  - Nem feltétlenül helyes (konfidencia mérték)
  - …de: hasznos ajánlások (gyors manuális vizsgálathoz)
- Működés elve: *metaadat* alapú statikus analízis
- Infrastruktúra: *Cecil* alapú, szabály-szerelvényeket tölt be
- Bővíthetőség: saját, privát szabályok lehetősége
  - További (igazi) szakterület-specifikus szabályok lehetősége (DSL)

Köszönöm a figyelmet!