SZOFTVERMETRIKÁK

A minőségbiztosítás informatikája

Gégény Dávid - KHIWFS

„You can't manage what you can't control, and you can't control what you don't measure.”

- Tom DeMarco

Szoftvermetrikák

- A metrikák számszerűsítk egy folyamat vagy termék minőségét
- Fontos a fejleszthetőség
- Objektív eredményt adnak
- Lehetővé teszik az összehasonlítást
- A metrikák vizsgálata a minőség-ellenőrzés és minőségbiztosítás részét képezi

Szoftvermetrikák osztályozása

- Folyamatmetrikák
  - pl. átlagos javítási idő
- Termékmetrikák
  - Pl. kódsorok száma, ciklomatikus komplexitás

Folyamatmetrikák

- Fejlesztés hatékonyságát mérő
  - új kódsorok száma havonta
    - ne ezzel figyeljük a produktivitást (copy-paste)
  - új funkciópontok száma havonta
  - stb.
- Termékminőségre vonatkozó
  - Bugok száma kódsoronként
  - átlagos javítási idő
  - átlagos meghibásodási idő
  - két meghibásodás között eltelt átlagos idő
  - hiba valószínűsége kérés esetén

Metrikák szerepe

- Kockázatelemzés
- Kockázatcsökkentés
- Erőforrás-igény, költségigény becslése
- Szoftverminőség meghatározása

Termékmetrikák

- Méretmetrikák
- Öröklődési metrikák
- Komplexitás
- Kohézió
- Csatolás
- Bad Smell és klón metrikák
- forráskód minősége (pl. szabványsértések száma)

Méretmetrikák

- Kódsorok száma (LOC)
  - a rendszerhibákkal kapcsolatot feltételeztek
  - de függ
    - a programozási nyelvtől
    - a programozási stílustól
    - a rendszer típusától
- Hasznos kódsorok száma (ELOC)
  - Nem üres és nem komment sorok
- Osztályok száma (NCL)
- Attribútumok/metódusok száma (NA/NM)

OO metrikák

- Objektumorientált programozáshoz tartozó metrikák
- Öröklési fa mélysége (DIT)
  - az osztálytól a gyökérhez vezető út hossza a fában
- Osztályra adott válaszok száma (RFC)
  - azon metódusok száma, amelyet válaszként hívhat egy másik osztály hatására
- Specializációs index

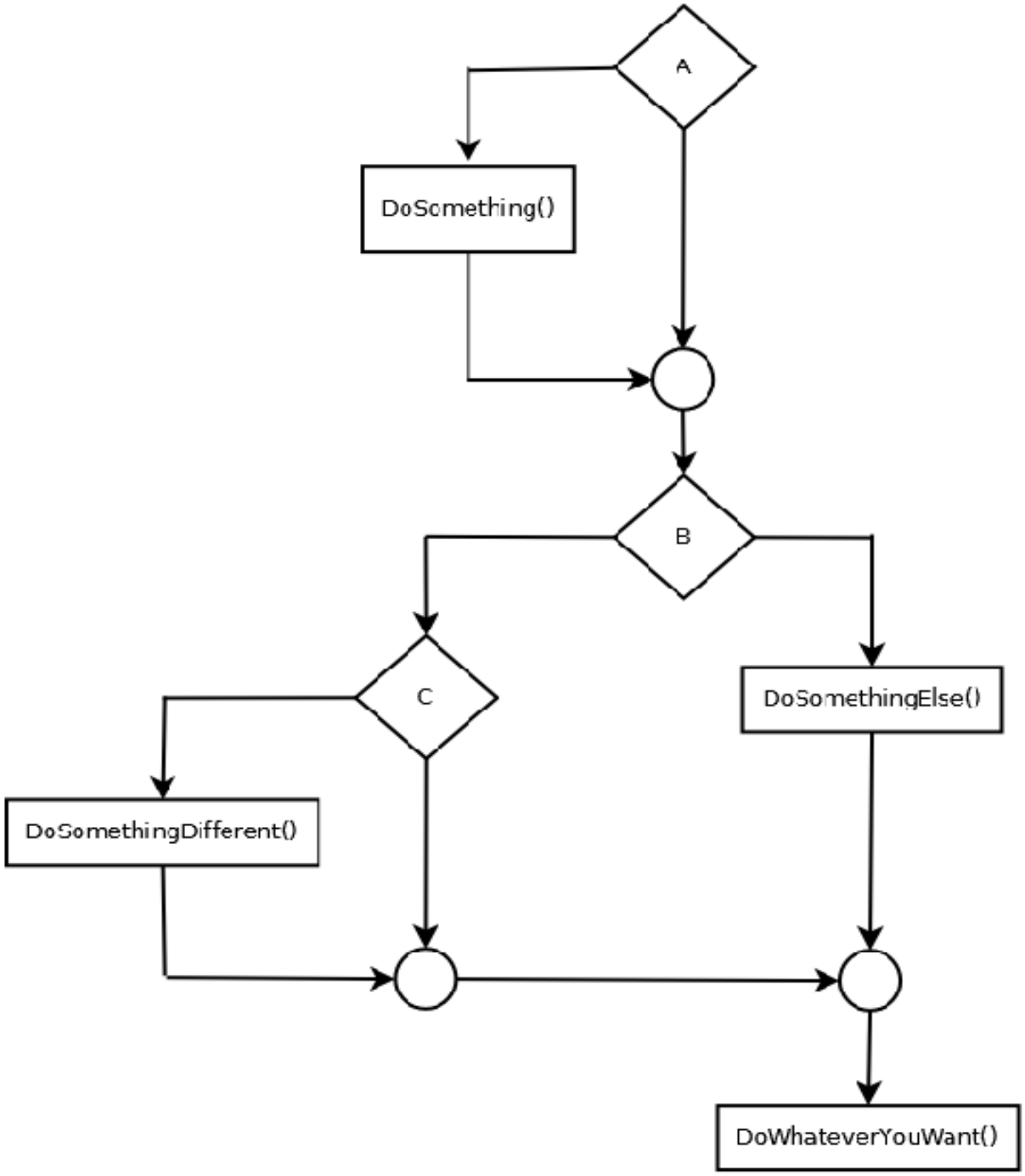
Komplexitás (McCabe)

- Ciklomatikus komplexitás
- Gráfmetrika
- Folyamatábrán (programgráfon) használjuk
- E – N + 2P
  - E – élek száma
  - N – csúcsok száma
  - P – összefüggő komponensek száma (általában egy)
- Minél nagyobb, annál nehezebb tesztelni vagy megérteni a kódot
- Az idő- és tárbonyolultság szintén fontos szerepet játszik (ordó szimbolika)

```
if (A)
{
    DoSomething();
}

if (B)
{
    if (C)
    {
        DoSomethingDifferent() { }
    }
}
else
{
    DoSomethingElse();
}

DoWhateverYouWant();
```



No. of Nodes: 10
No. of Edges: 12
No. of components: 1

Cyclomatic complexity:
$12 - 10 + 2 \cdot 1 = 4$

WMC

- Weighted Methods per Class
- A tartalmazott metódusok McCabe komplexitásainak összege
- Magas érték tervezési hibára utalhat

LCOM

- Lack of Cohesion on Methods
- Azt méri, mennyire állnak kapcsolatban egymással az osztály metódusai
- Magas érték jobb enkapszulációt jelent
- Az alacsony érték tervezési hibára vagy magas komplexitásra utalhat

CBO

- Coupling Between Object classes
- Az osztályok mennyire állnak egymással kapcsolatban (pl. metódusokon keresztül)
- Az osztály által használt más osztályok száma (metódus, attribútum vagy öröklés által)
- A magas érték jelentése
  - rossz enkapszuláció
  - magas hibaszám
  - rossz tesztelhetőség
  - érzékenység a változásra
- Nagyon magas a korreláció a hibaszámmal

Fan-in, fan-out

- Fan-in – hívó modulok száma
- Fan-out – hívott modulok száma
- Henry és Kafura komplexitás
  - Komplexitás = hossz • (fan-in • fan-out)^2
- Card és Glass komplexitás
  - strukturális komplexitás: S(i) = fan-out^2(i)
  - adatkomplexitás: D(i) = v(i) / (fan-out(i) + 1)
    - v(i) – I/O paraméterek száma
  - rendszer komplexitás: C(i) = S(i) + D(i)

Klón metrikák

- „Copy-paste használat”
- Nehezebb változtatni vagy megérteni a kódot
- Felesleges kódrészek kerülhetnek bele
- CCL – klónosztályok
- CI – klónpéldányok
- CC – klón-lefedettség
  - klónként azonosított kódrészek aránya
  - magas érték több hibát, nehezebb érthetőséget eredményez

```
static void Main(string[] args)
{
    //declare array f as double array and
    //assign initial values to double arrays a, b, c, d, e

    for (int i = 0; i < a.Length; i++)
    {
        f[i] = 2 * a[i] * a[i] + 3;
    }
    DoSomething(f);

    for (int i = 0; i < b.Length; i++)
    {
        f[i] = 2 * b[i] * b[i] + 3;
    }
    DoSomething(f);

    for (int i = 0; i < c.Length; i++)
    {
        f[i] = 2 * c[i] * c[i] + 3;
    }
    DoSomething(f);

    for (int i = 0; i < d.Length; i++)
    {
        f[i] = 2 * d[i] * d[i] + 3;
    }
    DoSomething(f);

    for (int i = 0; i < e.Length; i++)
    {
        f[i] = 2 * e[i] * e[i] + 3;
    }
    DoSomething(f);
}
```

```
static double[] CalculateFunction(double[] array)
{
    double[] result = new double[array.Length];

    for (int i = 0; i < array.Length; i++)
    {
        result[i] = 2 * array[i] * array[i] + 3;
    }

    return result;
}

static void Main(string[] args)
{
    //declare array f as double array and
    //assign initial values to double arrays a, b, c, d, e

    f = CalculateFunction(a);
    DoSomething(f);

    f = CalculateFunction(b);
    DoSomething(f);

    f = CalculateFunction(c);
    DoSomething(f);

    f = CalculateFunction(d);
    DoSomething(f);

    f = CalculateFunction(e);
    DoSomething(f);
}
```

Származtatott metrikák (ISO/IEC 9126)

- Funkcionalitás
- Megbízhatóság
- Használhatóság
- Hatékonyság
- Karbantarthatóság
- Hordozhatóság

Elemzés

- A meghatározott metrikákat elemezni kell
- Baseline értékek szükségesek
  - nagyszámú mintából számított átlag
- szakértői tudást igényelhet

Bad Smell

- Adatosztály (csak adattagok, esetleg getter, setter)
- Feature envy (más osztály adattagjaira jobban koncentrál, mint a sajátjára)
- Nagyméretű osztály
- Lusta osztály (a szülő, gyerek, vagy a hívó végzi az összes feladatát)
- Hosszú metódus
- Hosszú paraméterlista

Köszönöm a figyelmet!