

Decremental Development

Regaining Simplicity in Software

Prof. Peter Sommerlad

HSR - Hochschule für Technik Rapperswil
Institute for Software

Oberseestraße 10, CH-8640 Rapperswil

peter.sommerlad@hsr.ch

<http://ifs.hsr.ch>

<http://wiki.hsr.ch/PeterSommerlad>



TWEAKFEST

-
- Microsoft releases Windows 2020 consisting of fewer source code lines than Windows 95.
 - This Windows 2020 is bug free, fully automatically tested, easily adaptable by its users and runs the software required by those users easily on the hardware they already possess.
 - And, BTW, it is released in the year 2019.

Vision

Open Source version



- In 2015 a Linux version is released that has fewer source code lines than Linux 1.0 (1994)
- This Linux version is bug free, fully automatically tested, easily adaptable by its users and runs the software required by all users easily on the hardware they already chose to run it on.

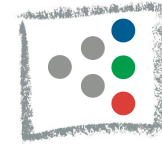
Decremental Development



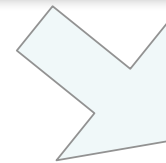
INSTITUTE
FOR
SOFTWARE

- **Reduce software size TO 10%**
 - while keeping required functionality
 - while improving its quality
 - while improving its design

One means of reduction -> choice of language

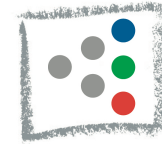


```
using System;  
  
class HelloWorld  
{  
    public static int Main(String[] args)  
    {  
        Console.WriteLine("Hello, World!");  
        return 0;  
    }  
}
```



puts "Hello, World!"

Hiding + Knowledge



```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
for i in 0..a.length-1 do
  for j in i+1...a.length do
    if (a[i] > a[j]) then
      x = a[i]
      a[i] = a[j]
      a[j] = x
    end
  end
end
puts "a:"
puts a
```

Hiding + Knowledge



```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then
```

```
      x = a[i]  
      a[i] = a[j]  
      a[j] = x
```

```
    end
```

```
  end
```

```
end
```

```
puts "a:"
```

```
puts a
```

Hiding + Knowledge



```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then
```

```
      x = a[i]  
      a[i] = a[j]  
      a[j] = x
```

```
    end
```

```
  end
```

```
end
```

```
puts "a:"
```

```
puts a
```

```
def swap a, i, j  
  x = a[i]  
  a[i] = a[j]  
  a[j] = x  
end
```

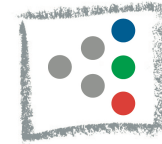
```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then  
      swap(a, i, j)
```

```
    end
```

```
  end
```

```
end
```

Hiding + Knowledge



```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then
```

```
      x = a[i]  
      a[i] = a[j]  
      a[j] = x
```

```
    end
```

```
  end
```

```
end
```

```
puts "a:"
```

```
puts a
```

```
def swap a, i, j  
  a[i], a[j] = a[j], a[i]  
end
```

```
end
```

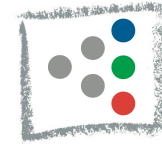
```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then  
      swap(a, i, j)
```

```
    end
```

```
  end
```

```
end
```

Use existing stuff from libraries -> Knowledge



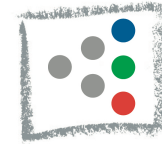
```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
def swap a, i, j
  a[i],a[j] = a[j], a[i]
end

for i in 0..a.length-1 do
  for j in i+1...a.length do
    if (a[i] > a[j]) then
      swap(a, i, j)
    end
  end
end

puts "a:"
puts a
```

Use existing stuff from libraries -> Knowledge

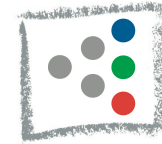


```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
def swap a, i, j  
  a[i],a[j] = a[j], a[i]  
end
```

```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then  
      swap(a, i, j)  
    end  
  end  
end  
puts "a:"  
puts a
```

Use existing stuff from libraries -> Knowledge



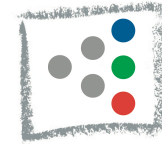
```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
def swap a, i, j
  a[i],a[j] = a[j], a[i]
end

for i in 0..a.length-1 do
  for j in i+1...a.length do
    if (a[i] > a[j]) then
      swap(a, i, j)
    end
  end
end

puts "a:"
puts a
```

Use existing stuff from libraries -> Knowledge

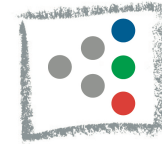


```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
def swap a, i, j  
  a[i],a[j] = a[j], a[i]  
end
```

```
for i in 0..a.length-1 do  
  for j in i+1...a.length do  
    if (a[i] > a[j]) then  
      swap(a, i, j)  
    end  
  end  
end  
puts "a:"  
puts a
```

Use existing stuff from libraries -> Knowledge



```
a=[42, 1, 7, 2, 34, 64, 29, 2]
```

```
def swap a, i, j
  a[i],a[j] = a[j], a[i]
end

for i in 0..a.length-1 do
  for j in i+1...a.length do
    if (a[i] > a[j]) then
      swap(a, i, j)
    end
  end
end

puts "a:"
puts a
```

Use existing stuff from libraries -> Knowledge



INSTITUTE
FOR
SOFTWARE

```
a=[42, 1, 7, 2, 34, 64, 29, 2]  
puts "a:"  
puts a.sort
```

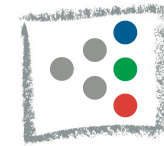
Famous Quotes by Tony Hoare



- Inside every large program, there is a small program trying to get out.
- There are two ways of constructing a software design:
 - one way is to make it so simple that there are obviously no deficiencies, and
 - the other way is to make it so complicated that there are no obvious deficiencies.
- The first method is far more difficult.

Proposal

Decremental Development



INSTITUTE
FOR
SOFTWARE

- **Create Tools for automated Refactoring**
 - for languages lacking support, i.e.,
 - C/C++, PL/1, COBOL(?)
 - javascript, php, groovy
- **Develop new Approaches for higher-level Software Simplification**
 - beyond Refactoring
 - i.e., detecting potential for simplification
- **Increase Valuation of Simplicity**
 - as a software design goal
 - articles, presentations, case studies

Why the need for public funding?

- **Popular Development Environments are “free”**
 - MS VS, Eclipse, Netbeans, ...
 - some even Open Source
 - no real money making from development tools
- **Results can benefit many**
 - shared interest
 - hard to get a “really interested sponsor”
- **Dissemination of an “idea”**
 - a lot of “unlearning” required
 - ongoing presence of theme (media)

Why we need Decremental Development



- **Problems solved by Software increase**
 - more problems
 - larger & more complex
- **“Good-enough” quality often isn’t**
 - when deployed (Beta-Release)
 - while maintained (updates breaking stuff)
- **Useful Software is used longer than intended**
 - pro-active maintenance often neglected
 - repeated bug-hunt-fix-patch deteriorates quality

Decremental Development

- **Reduce software size TO 10%**
 - while keeping required functionality
 - while improving its quality
 - while improving its design
- **This is not a dream, but requires hard work.**
 - We need to start paying attention in the small!