Refactoring

Kim Brugger

A well-written program is its own Heaven;

a poorly-written program is its own Hell.

-The Tao of Programming

Refactoring: Outline

Introduction

How to refactor

When to refactor

Example

Hands on

Reflection

Intro: Software package XXX

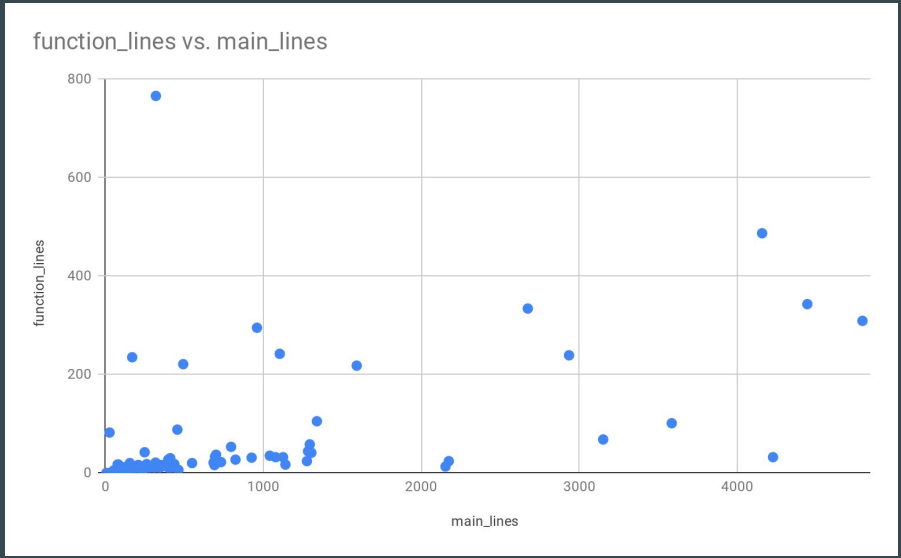
~100 perl script

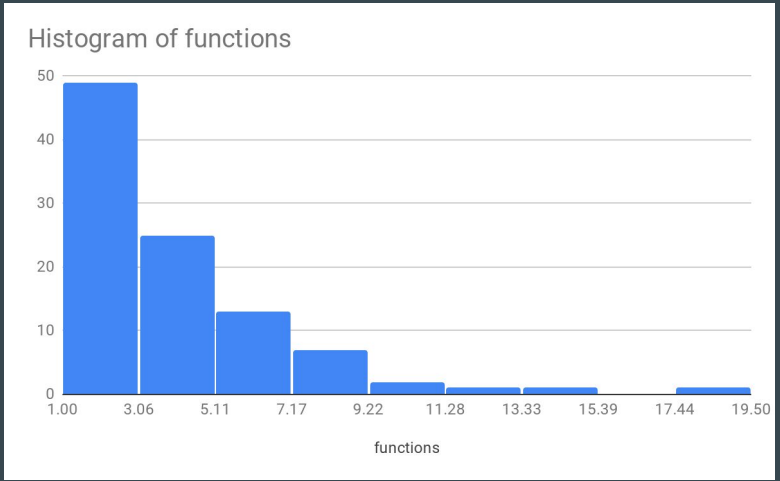
A lot of “dead” code

Duplicate functions in scripts:

Function name	Copies
mysystem	72
split_line_ref	57
split_line	33
a2file	32

Intro: Software package XXX

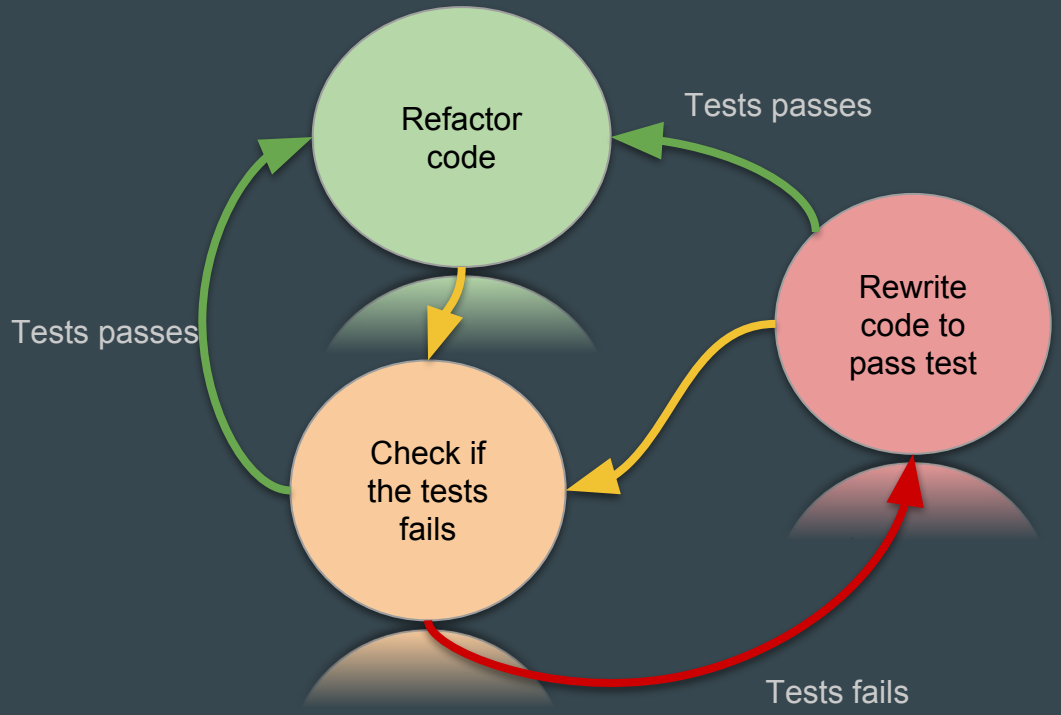




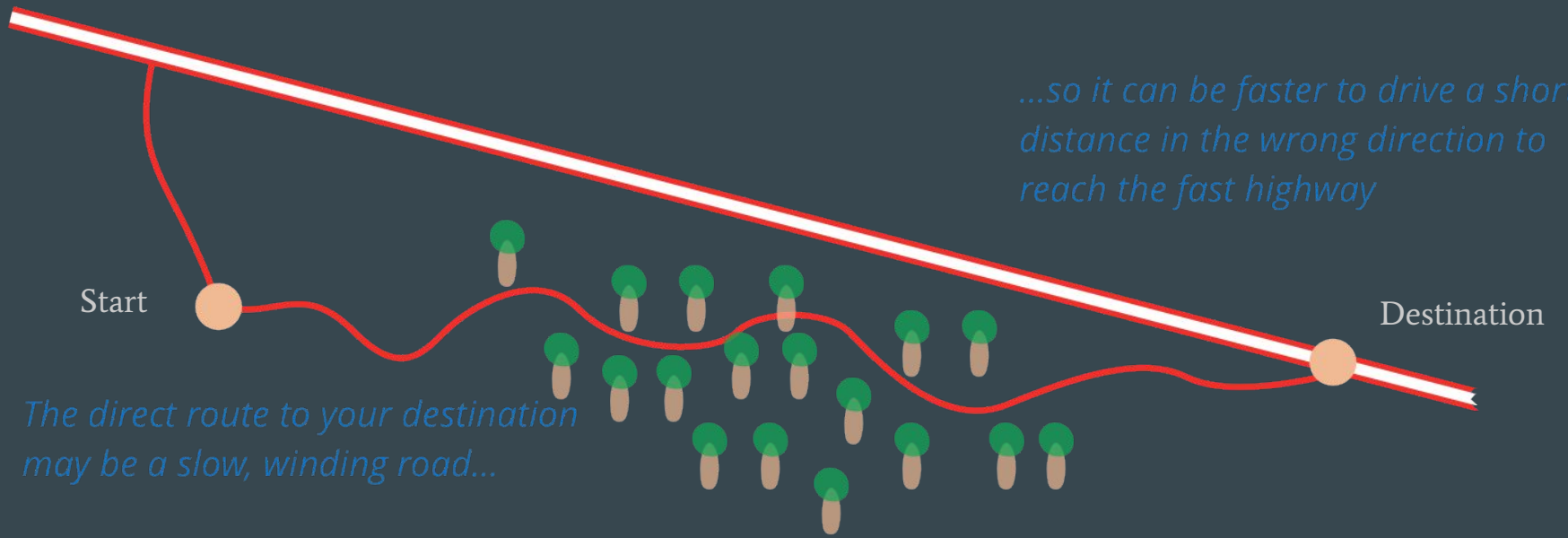
How: What refactoring feels like

How: Refactoring according to your boss

How: Refactoring life cycle

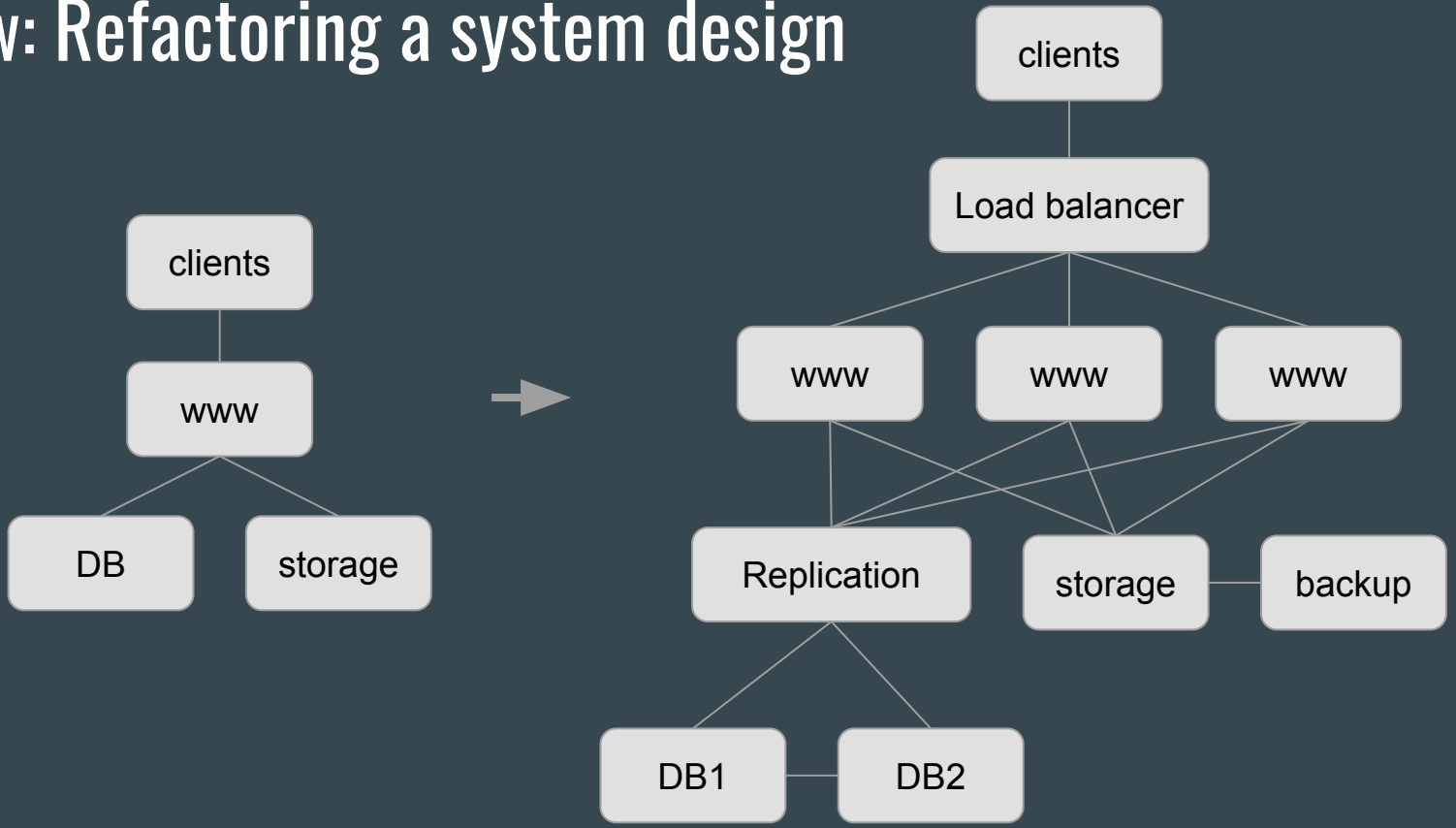


When: Preparatory Refactoring

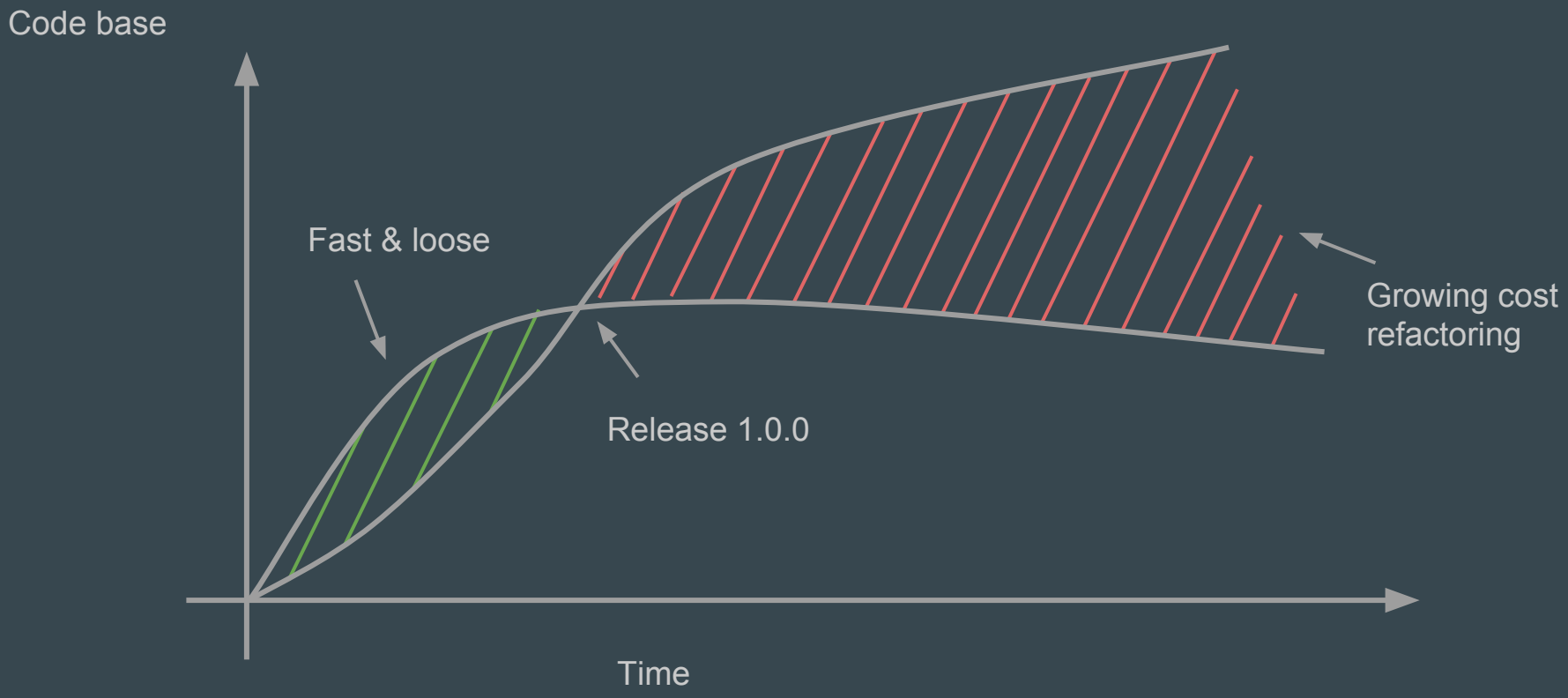


https://www.martinfowler.com/articles/preparatory-refactoring-example.html

How: Refactoring a system design



When: best time to refactor



Example: refactoring

```
l = 11
[    print( "".join([" "]*int((l+1 - line*2)/2-1)), end ="") or  \
 print( "".join(["*"]*(--l+1 - int(2*(l+1-line *2)/2-1))), end ="", sep=".o0O0o.") or \
      print( "".join([" "]*int((l+1 - line*2)/2-1)))  \
      for line in range(0, int(length/2)+1)]
```

Example: What could be improved here

```
l= 11

for line in range(l*0, int(l /2)+1):

    left   = "".join([" "]*int((l+1 - line*2)      /2-1  ))
    r="".join([" "]*int((--l+1-line*2)/2-1))
    middle = "".join(["*"]*(l+1 - int(2*(l+1-line*2)/2-1)))
    print(r+middle+left)
```

Refactoring your turn (30 min)

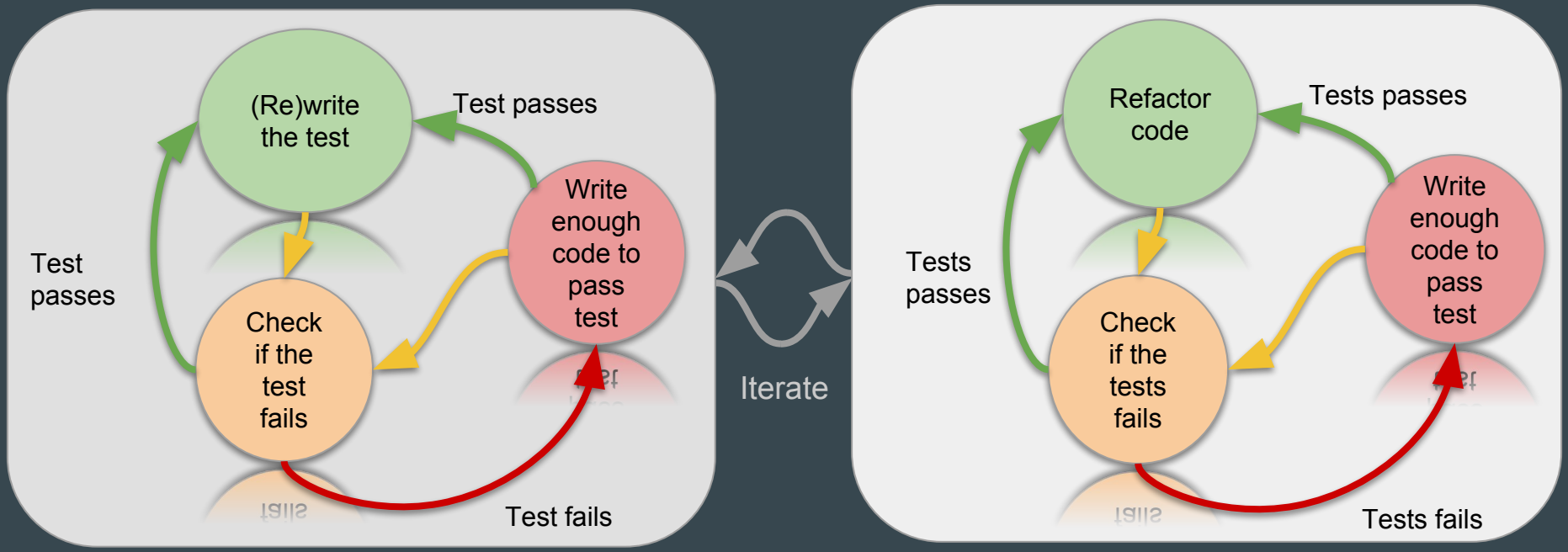
```
s = "Thanks for noticing me - Eeyore".lower()
r = 10
while True:
    l = [" "] * len( s )
    for j in range(1, len( s ) - 1 ):
        if (s[j-1]==s[j+1]):
            l[j] = "*"
    print("".join( l ))
    s = "".join( l )

    r-=1
    if ( not r  ):
        break
```

How would unit testing help, and how would you do it in this example?

Example: my refactoring

```
def make_pattern(text:str, loops:int=10):
    """ takes a text and place a star if the neighbouring letters are identical, generate loops lines """

    # Make text lower case
    text = text.lower()

    text_length = len( text )

    for i in range(0, loops ):

        # Make an empty list the same length as the text
        chars = [" "] * text_length

        for pos in range(1, text_length - 1 ):

            if text[pos-1]==text[pos+1]:
                chars[ pos ] = "*"


        text = "".join( chars )

        print( text )
```

Reflection: TTD & refactoring life cycle



Implement according to requirements

Improve code wo/ changing functionality

Refactoring: reflection & thoughts

Do you refactor code?

How to preserve software integrity when refactoring

How to handle legacy code?

Do you have code you don’t dare touch?

IDE’s helps you save time.