Developers Attentiveness to Example Usage

Ohad Barzilay, Blavatnik School of Computer Science, Tel-Aviv University

Amiram Yehudai, Blavatnik School of Computer Science, Tel-Aviv University

Orit Hazzan, Department of Education in Technology and Science, Technion

October 2010

HAoSE 2010:

The Second Workshop on Human Aspects of Software Engineering

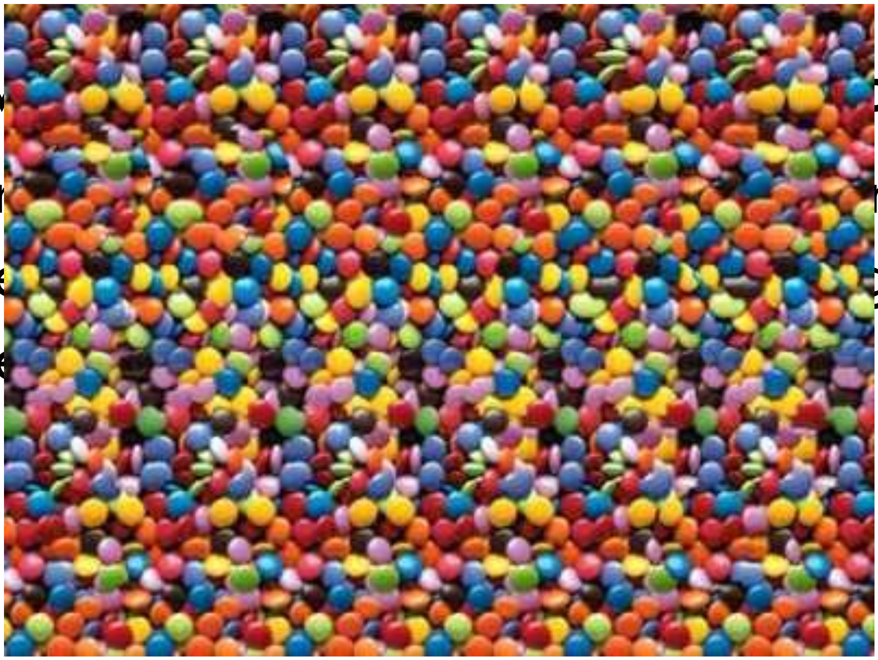
Agenda

- Research overview
- Example Attentiveness Observed
  - Context, Utilization, Scale
- Focus group case study
- Weaving example attentiveness into the software engineering ecosystem
- Summary

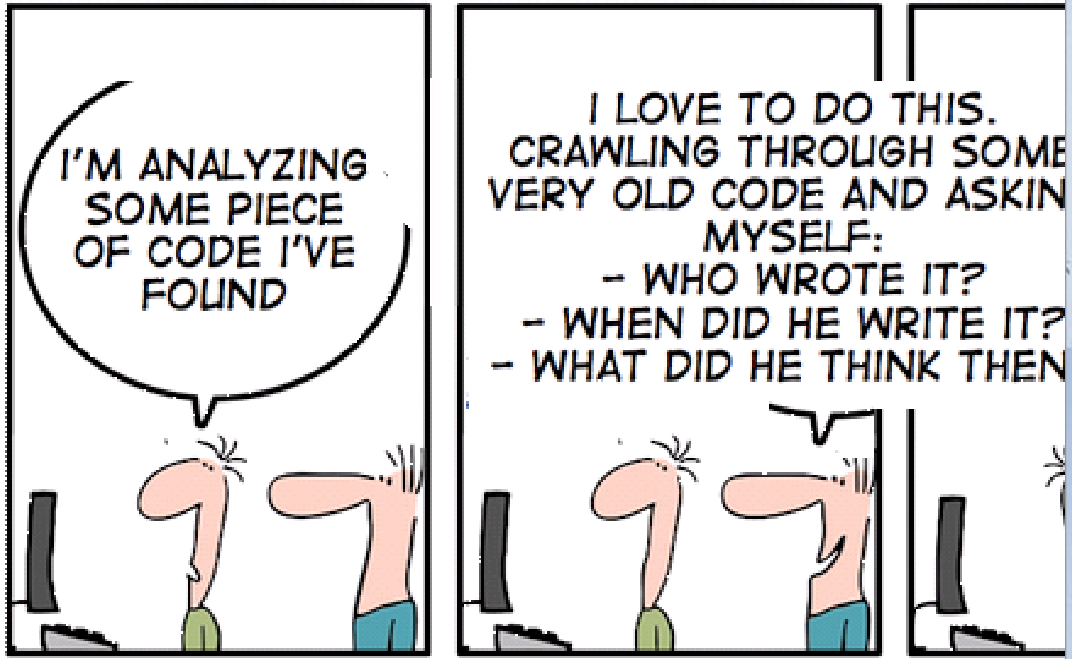
Introduction

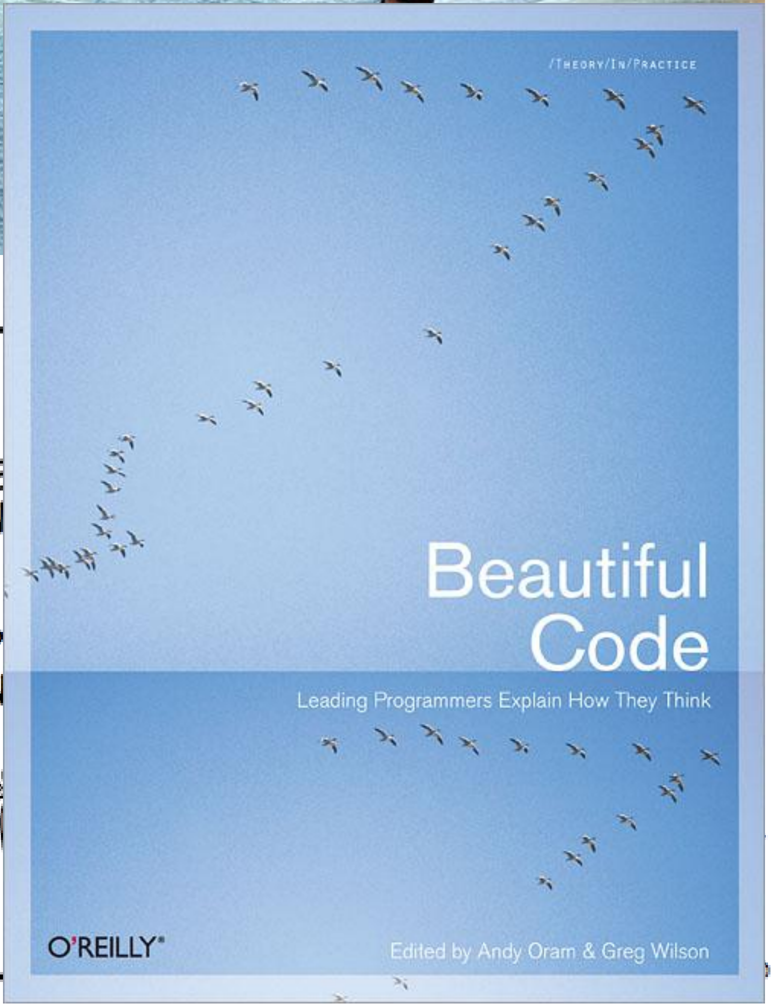
- We did not embark on this research with example usage in mind

- Our motiv ctivity
  - Refactor nformally for years be given a name [Opdyke



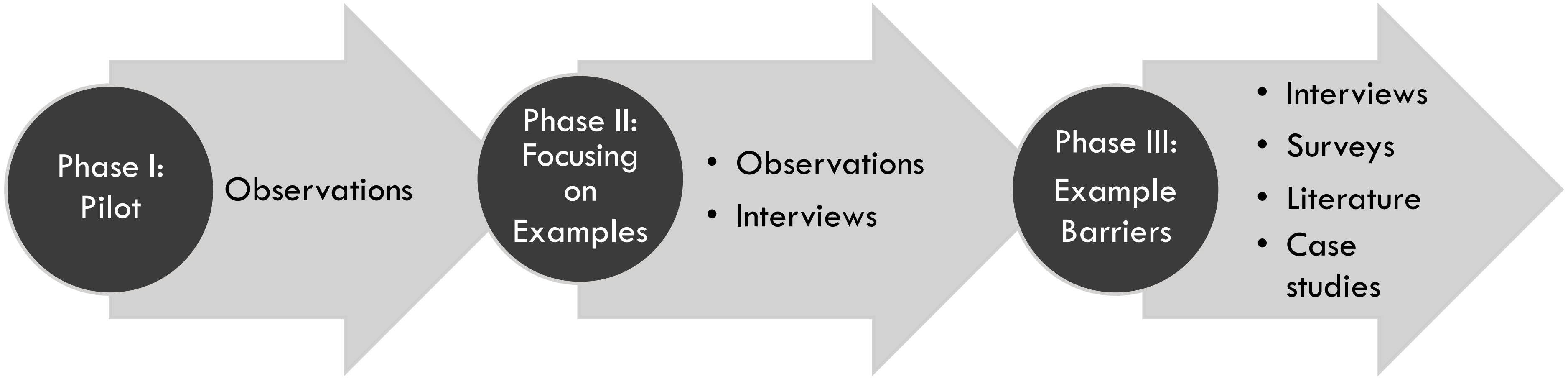
Looking for "The Next Refactoring"


I'M ANALYZING SOME PIECE OF CODE I'VE FOUND
I LOVE TO DO THIS. CRAWLING THROUGH SOME VERY OLD CODE AND ASKIN MYSELF: - WHO WROTE IT? - WHEN DID HE WRITE IT? - WHAT DID HE THINK THEN


/Theory/In/Practice
Beautiful Code
Leading Programmers Explain How They Think
O'REILLY®
Edited by Andy Oram & Greg Wilson

Research overview

- Iterative process
- Grounded theory



Examples and related areas

- We define example usage as using an already existing code fragment (the example) within a new context. For example:
  - Looking for code in the Internet
  - Examples that are part of the documentation
  - Examining code in the code base of the organization
  - And much more…

- There is some overlap with the following areas:
  - Reuse
  - Copy and paste
  - Patterns

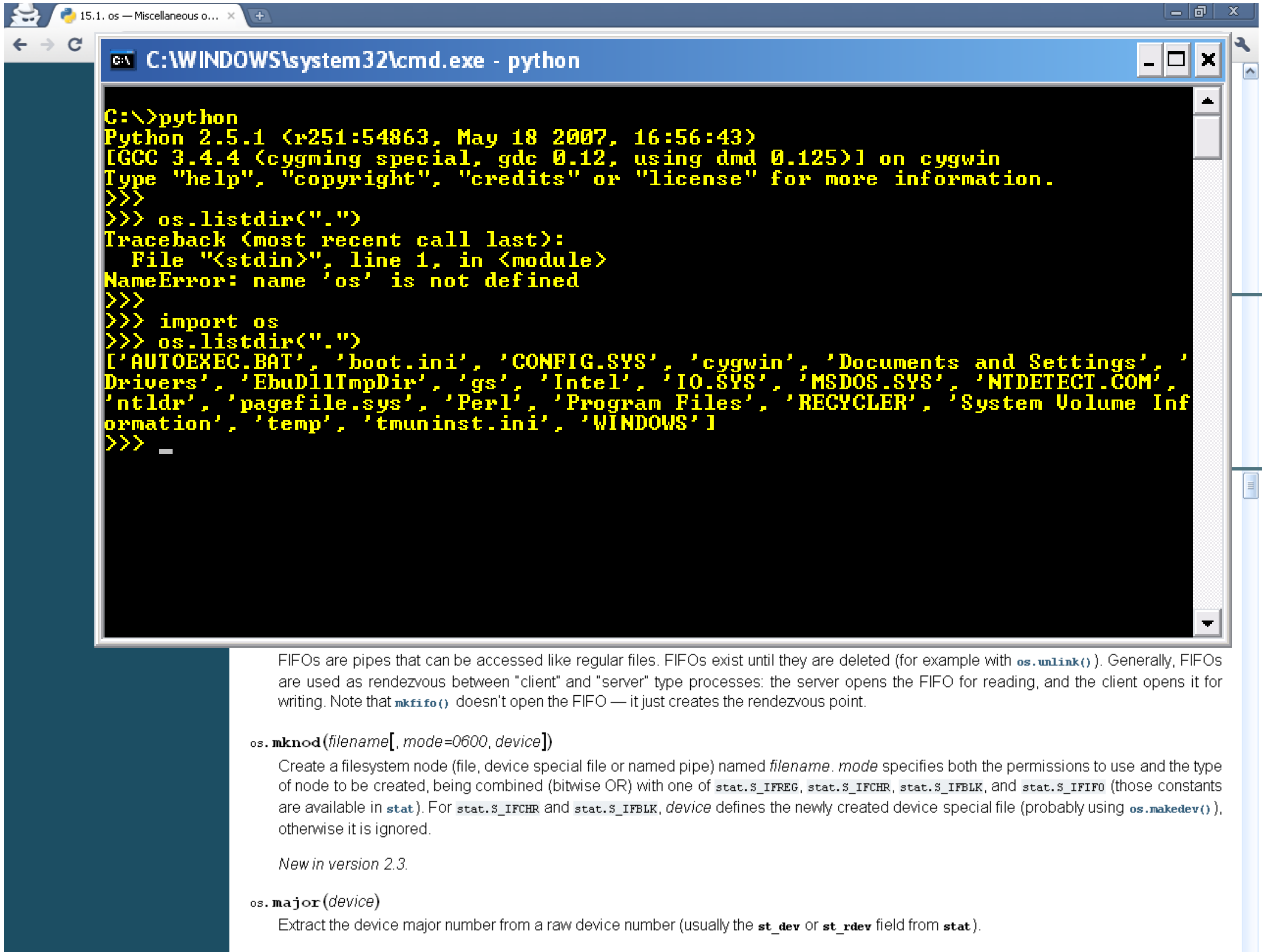
Example Attentiveness Observed

- Phase I:
  - 2 teams in 2 large world wide software companies
  - 1 month, 14 sessions of 2-3 hours each
  - Observed 10 developers

- In this talk we focus on one session (during phase I), and we interpret it from the perspective of example attentiveness

Example Attentiveness I

Only on Python



15.1. os — Miscellaneous o...
C:\WINDOWS\system32\cmd.exe - python

```
C:\>python
Python 2.5.1 (r251:54863, May 18 2007, 16:56:43)
[GCC 3.4.4 (cygming special, gdc 0.12, using dmd 0.125)] on cygwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> os.listdir(".")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'os' is not defined
>>>
>>> import os
>>> os.listdir(".")
['AUTOEXEC.BAT', 'boot.ini', 'CONFIG.SYS', 'cygwin', 'Documents and Settings', '
Drivers', 'EbuDllTmpDir', 'gs', 'Intel', 'IO.SYS', 'MSDOS.SYS', 'NTDETECT.COM',
'ntldr', 'pagefile.sys', 'Perl', 'Program Files', 'RECYCLER', 'System Volume Inf
ormation', 'temp', 'tmuninst.ini', 'WINDOWS']
>>> _
```

FIFOs are pipes that can be accessed like regular files. FIFOs exist until they are deleted (for example with `os.unlink()`). Generally, FIFOs are used as rendezvous between "client" and "server" type processes: the server opens the FIFO for reading, and the client opens it for writing. Note that `mkfifo()` doesn't open the FIFO — it just creates the rendezvous point.

`os.mknod`(*filename*[, *mode*=0600, *device*])

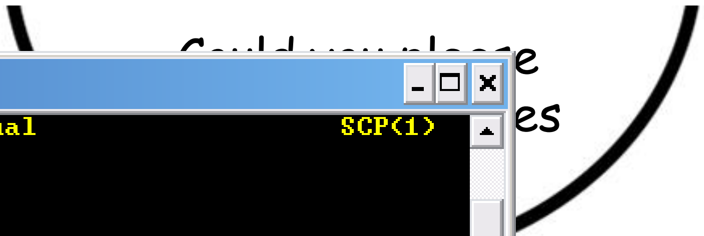
Create a filesystem node (file, device special file or named pipe) named *filename*. *mode* specifies both the permissions to use and the type of node to be created, being combined (bitwise OR) with one of `stat.S_IFREG`, `stat.S_IFCHR`, `stat.S_IFBLK`, and `stat.S_IFIFO` (those constants are available in `stat`). For `stat.S_IFCHR` and `stat.S_IFBLK`, *device* defines the newly created device special file (probably using `os.makedev()`), otherwise it is ignored.

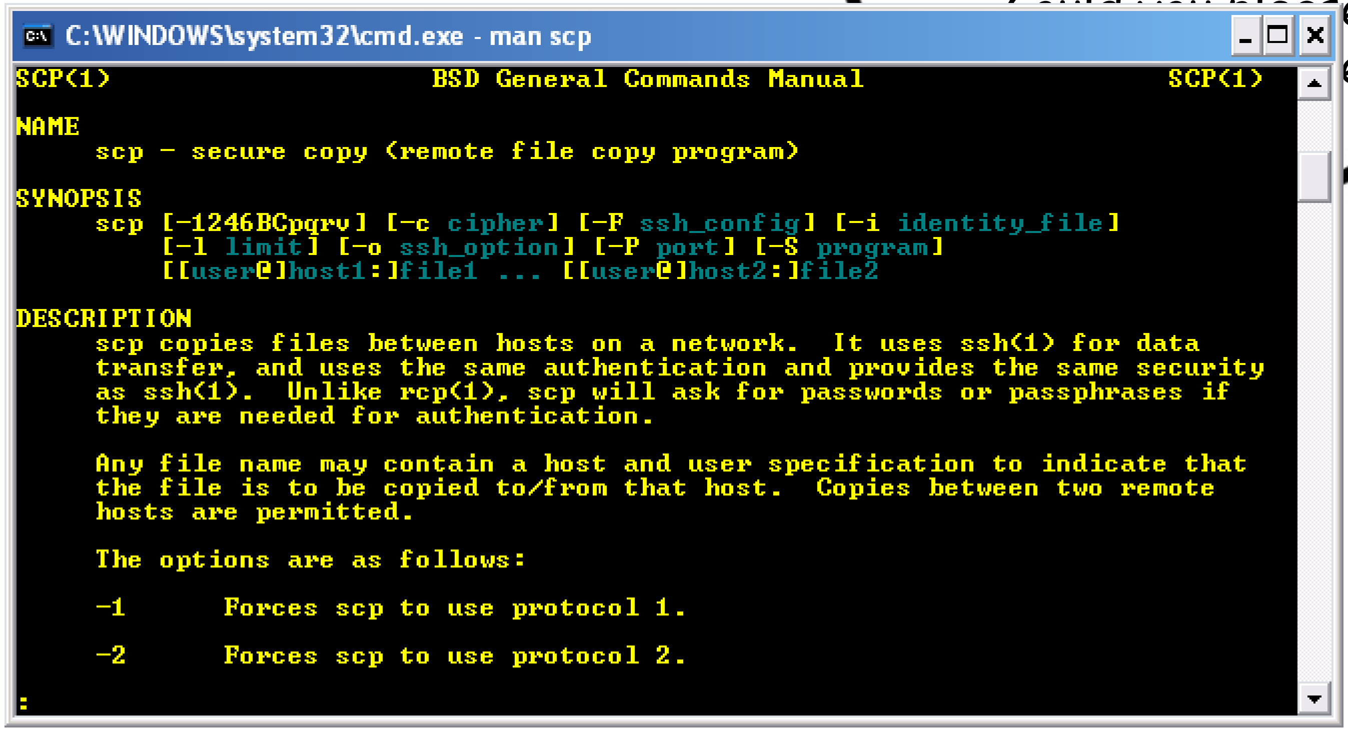
New in version 2.3.

`os.major`(*device*)

Extract the device major number from a raw device number (usually the `st_dev` or `st_rdev` field from `stat`).

Example Attentiveness I
Only on Python





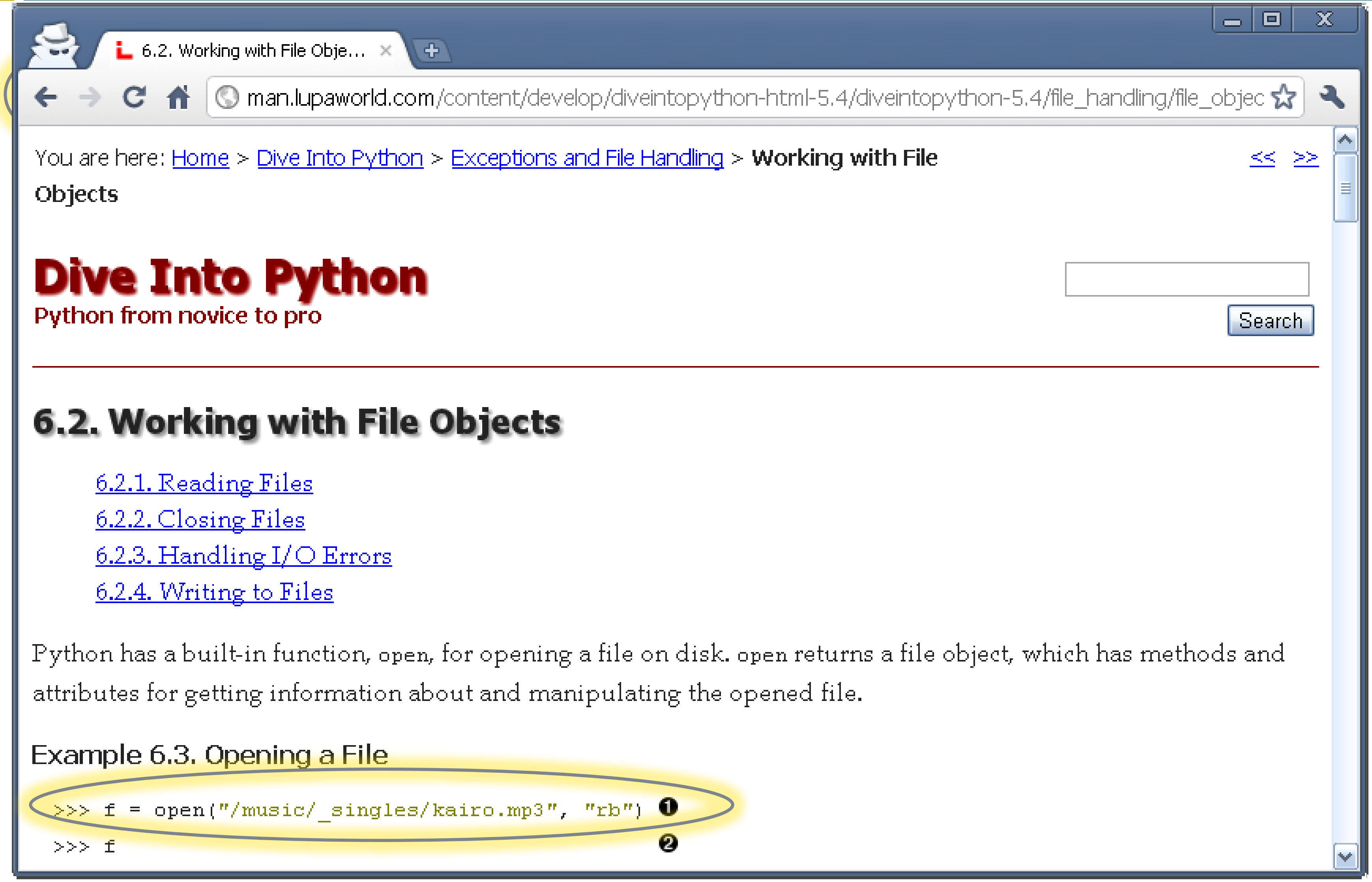
```
SCP(1)                    BSD General Commands Manual                    SCP(1)

NAME
     scp - secure copy (remote file copy program)

SYNOPSIS
     scp [-1246BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
         [-l limit] [-o ssh_option] [-P port] [-S program]
         [[user@]host1:]file1 ... [[user@]host2:]file2

DESCRIPTION
     scp copies files between hosts on a network.  It uses ssh(1) for data
     transfer, and uses the same authentication and provides the same security
     as ssh(1).  Unlike rcp(1), scp will ask for passwords or passphrases if
     they are needed for authentication.

     Any file name may contain a host and user specification to indicate that
     the file is to be copied to/from that host.  Copies between two remote
     hosts are permitted.

     The options are as follows:

     -1      Forces scp to use protocol 1.

     -2      Forces scp to use protocol 2.

:
```

Example Attentiveness I
Only on Python

- Ben used Google to search for the Python method but didn't use it for the SCP syntax

- Possible reasons:
  - He wasn't attentive for this option
  - He might think that short search-write cycles are unique for Python
  - Might be related to the way he learned Python
  - Maybe other reasons?

Example Attentiveness II
Read Only


6.2. Working with File Obje…
man.lupaworld.com/content/develop/diveintopython-html-5.4/diveintopython-5.4/file_handling/file_objec
Search

You are here: Home > Dive Into Python > Exceptions and File Handling > **Working with File Objects** << >>

Dive Into Python

Python from novice to pro

6.2. Working with File Objects

Python has a built-in function, `open`, for opening a file on disk. `open` returns a file object, which has methods and attributes for getting information about and manipulating the opened file.

Example 6.3. Opening a File

```
>>> f = open("/music/_singles/kairo.mp3", "rb")  ❶
>>> f                                            ❷
```

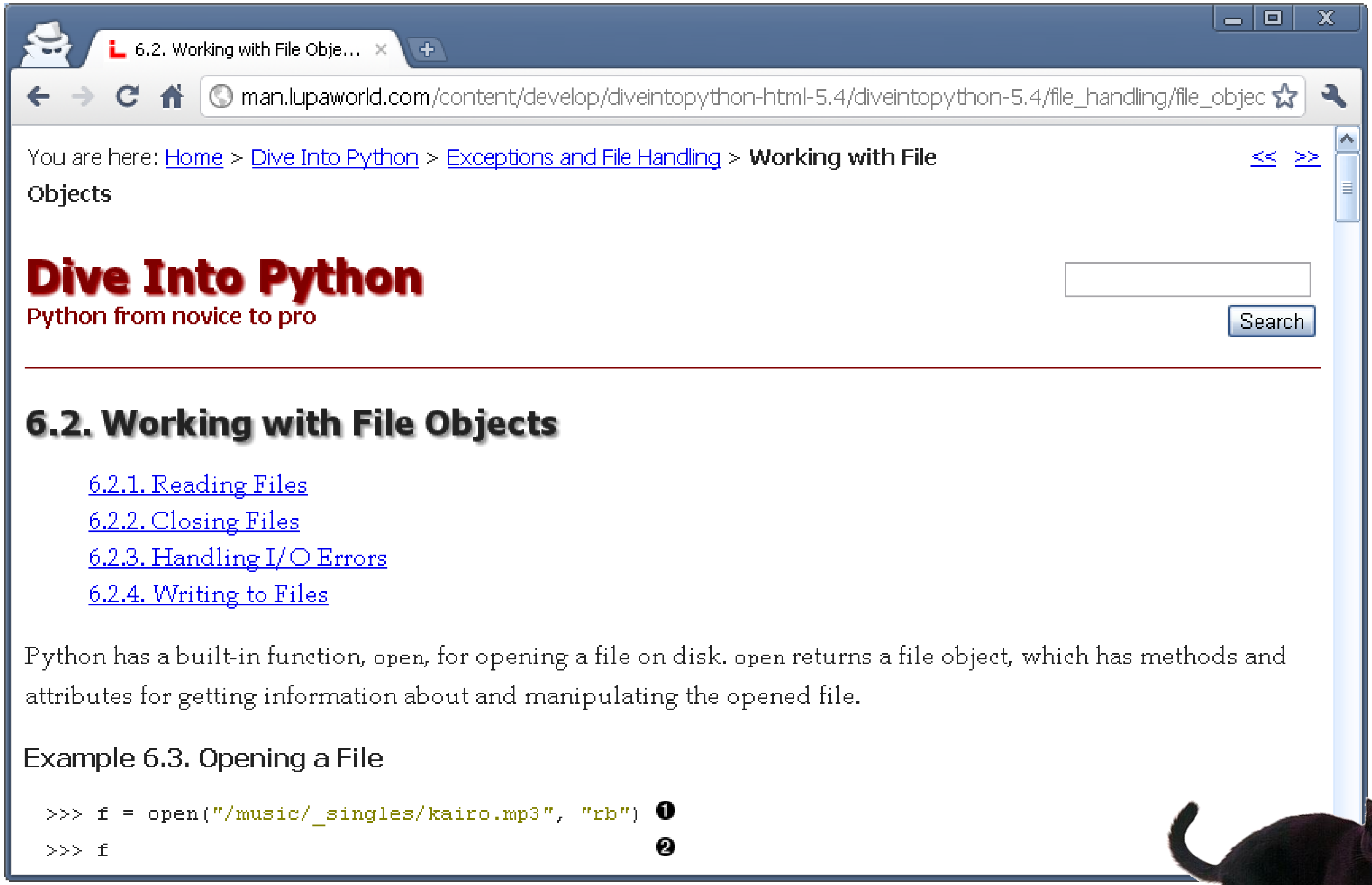
Example Attentiveness II
Read Only

- Ben found an example that matches his purpose, however he avoided copying and pasting it but rather typed the code by himself
- This resulted in several errors, which were later fixed.

- Avoiding copy and paste:
  - Ben associates example with learning. He is not attentive for using also the example code
  - Typing the code by oneself gives him or her additional opportunity to review the code, memorize it and better understand it
  - Ben might consider copy and paste harmful

Example Attentiveness III

Think Big


6.2. Working with File Obje...
man.lupaworld.com/content/develop/diveintopython-html-5.4/diveintopython-5.4/file_handling/file_objec
Search

You are here: Home > Dive Into Python > Exceptions and File Handling > **Working with File Objects** << >>

Dive Into Python

Python from novice to pro

6.2. Working with File Objects

Python has a built-in function, open, for opening a file on disk. open returns a file object, which has methods and attributes for getting information about and manipulating the opened file.

Example 6.3. Opening a File

```
>>> f = open("/music/_singles/kairo.mp3", "rb")  ❶
>>> f                                             ❷
```

Example Attentiveness III **Think Big**

- Ben looked each part of its task independently, and did not look for example of the whole scenario

- Possible reasons:
  - Ben started to code only after he has already decomposed the problem in his head
    - He was locked into a mindset, certain level of abstraction and is not attentive for other courses of action
  - Ben may associate example usage with certain scale and is not attentive using examples otherwise
  - Ben may not be aware of the **existence** of such examples
  - Ben is missing advanced **retrieval techniques** (not aware to their existence)

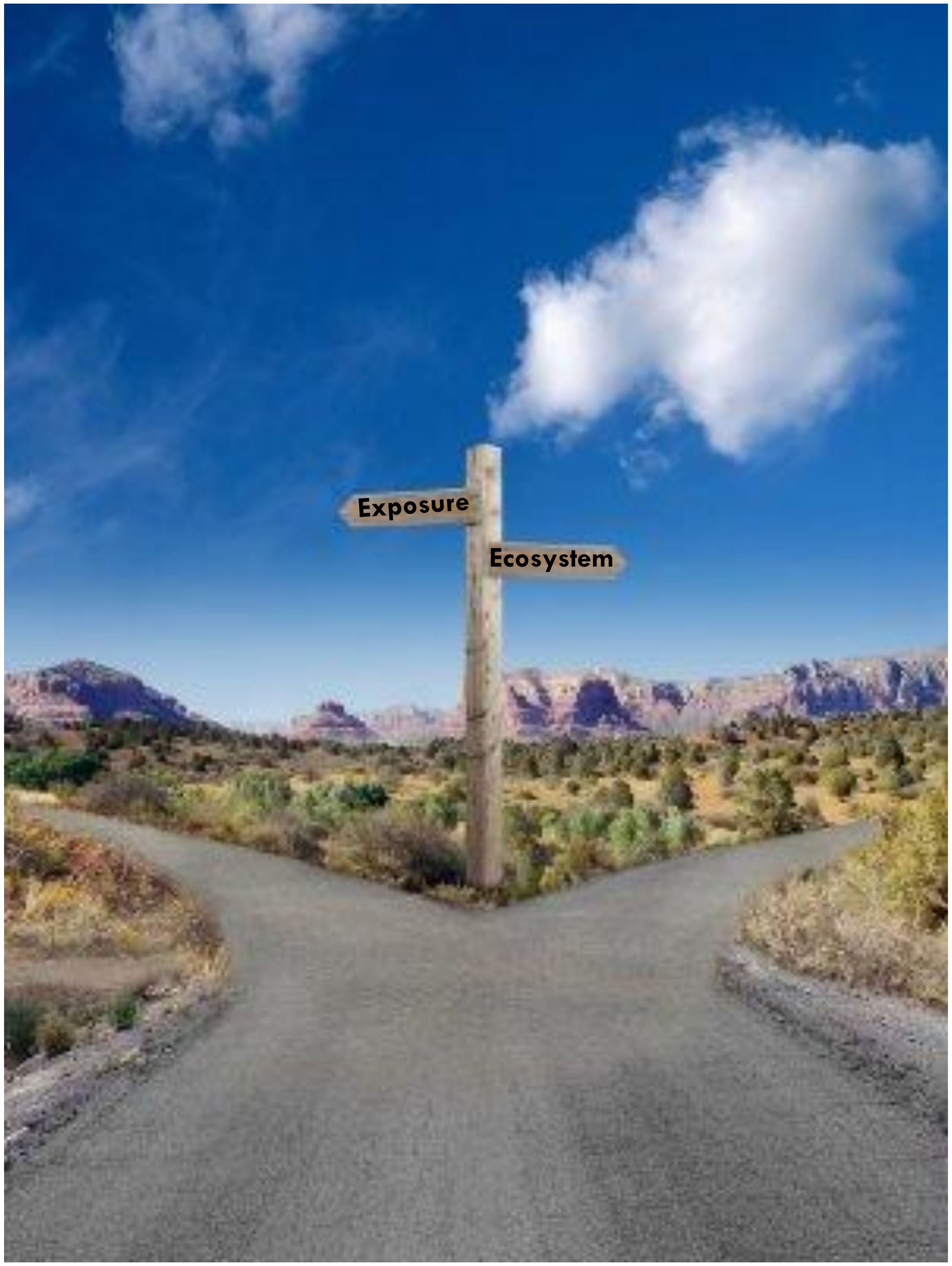
Developers attentiveness to example usage

- This session demonstrates 3 dependencies of example attentiveness:
  - Context
  - Utilization
  - Scale

- Would drawing Ben's attention for new opportunities of example usage change his behavior in the future?

Addressing Attentiveness Issues

- Two alternatives:
  - Take a proactive approach to increase developers awareness and attentiveness
  - Build an ecosystem in which theses issues are already weaved-in



Focus group case study

- Discussing example usage with developers
- Part of the mechanism we establish to collect industry feedback
- 20 developers
- Agenda:
  - Research review
  - Discussion
  - Reflection questionnaires
- A follow up questionnaire (after 3 months)

Focus group case study

- Research Review
  - Potential **benefits** of the systematic use of examples
  - **Barriers** preventing example usage from being applied more extensively
  - Example related **techniques**

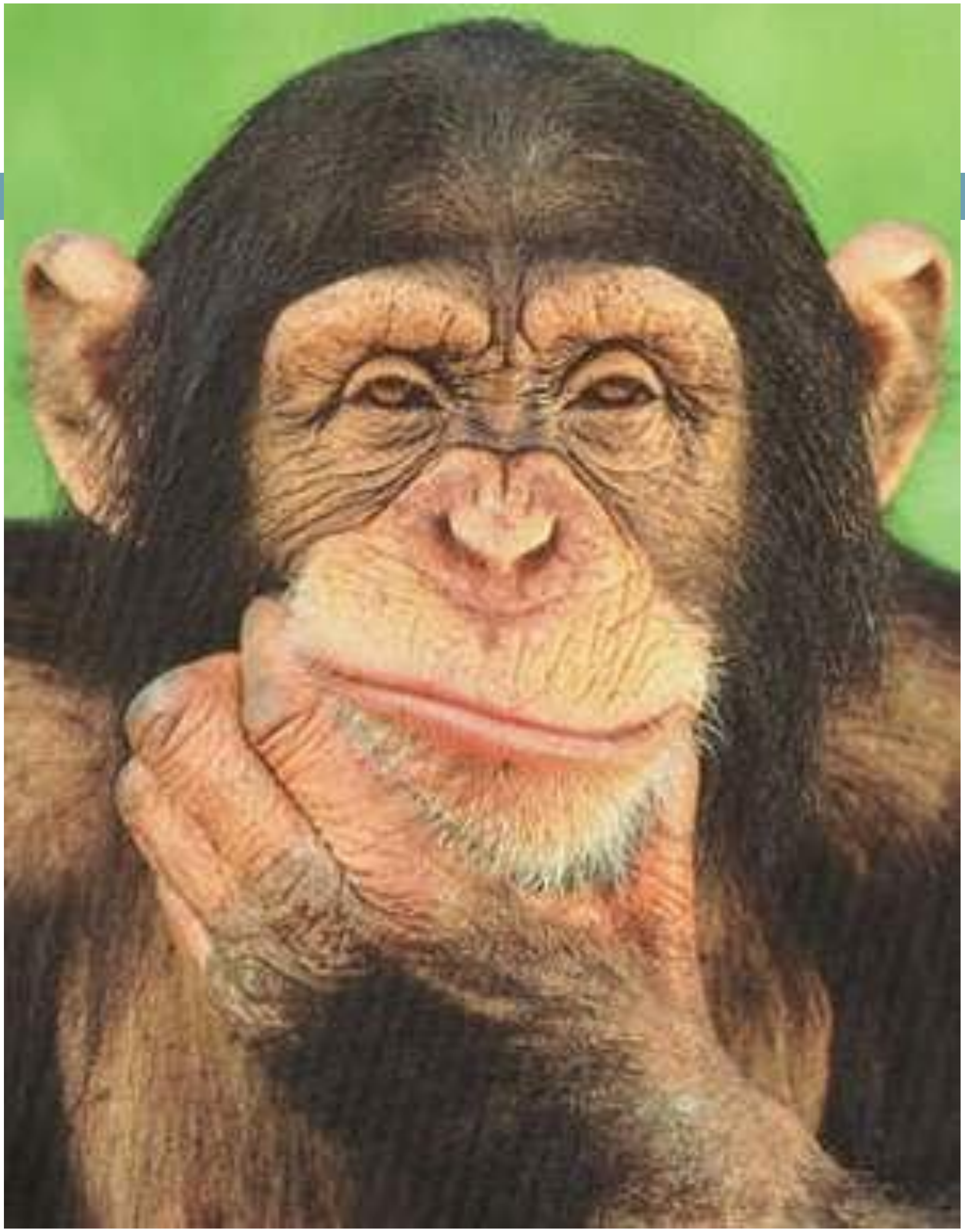
- In the reflection questionnaires we asked the participants:
  - Whether they **use** examples in their work
  - Whether they are **in favor** of using examples
  - Whether they were **influenced** by the session
  - How they estimate the session will affect their work **in the future**

- A follow up (after 3 months)
  - Did the talk (or completing the questionnaire) affect your awareness to reuse and example usage? If yes, in what way?
  - Have you incorporated any new techniques or practices in your work with respect to example usage? If yes - which?

Reflection

- A ‘**reflective practitioner**’ [Schön, 1983,1987] is someone who, at regular intervals, looks back at the work done, and the work process, and considers how they can be improved

- **reflective practitiones** are not happy to carry on at the current standard, they want to improve



Summary of reflection questionnaires

- 7 of the 15 subjects stated that the session increased their level of **awareness** to **new opportunities** for example usage
  - Corresponding to the 3 types of awareness discussed earlier

- 4 of the 8 other subjects mentioned that following the session, they had some **new ideas about example usage** that they considered using in their work

New Opportunities

"Till now I used examples only 'as an inspiration'. Following the talk I would start using the **example code** as well.".

Selective example usage

- In another case study we identified additional types of selective example usage (though some of them are not related to attentiveness)

- The main variability factors are:
  - Reusing the example code or not
  - Developer's mental state (development mode)
  - Example size
  - Example source
  - Learning factor

- Indeed, the first 3 factors correspond to the 3 attentiveness aspects presented above: example context, scale and utilization.

Conceptualization and abstraction

"The talk helped [me] formalize some ideas I already had. I had been a learner-by-example for years but, as you know, putting a name to something makes it much more real and relevant."

4 of the 15 subjects addressed the **conceptualization** and **abstraction** of example usage in software development

Conceptualization and abstraction

- These quotes (and others) suggest addressing example usage as an **abstract fundamental software activity** and not merely as a **programming technique**

- The focus group participants consider examples in a wider context:
  - Developer productivity
  - Development speed
  - Code quality

- The focus group participants consider examples for:
  - Documentation purposes
  - Client training
  - Example-aware development process

Conceptualization and abstraction

- Implication on the nature of software development
  - "...at the end of the day, we are all merely plumbers...."
- To exploit to full potential of the example usage concept we propose to weave it into the **software engineering ecosystem**
- We demonstrate this idea using the **refactoring** concept

Refactoring Revisited

much more than a a fundamental activity

- **Refactoring** is ~~a disciplined technique~~ for restructuring an existing body of code, altering its internal structure without changing its external behavior [www.refactoring.com/]

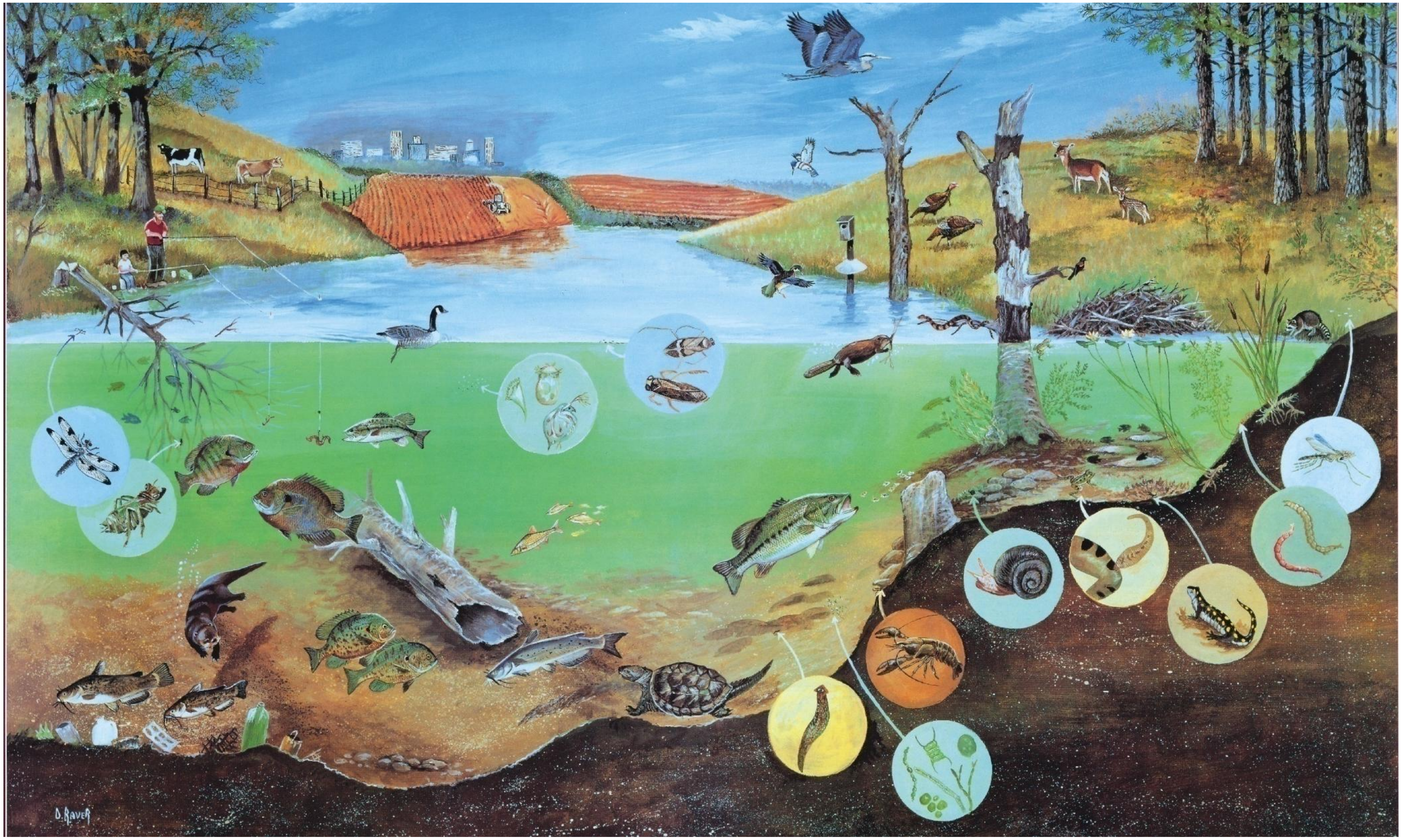
Appreciating *Refactoring*

The mere **identification of refactoring** promoted the following important processes:

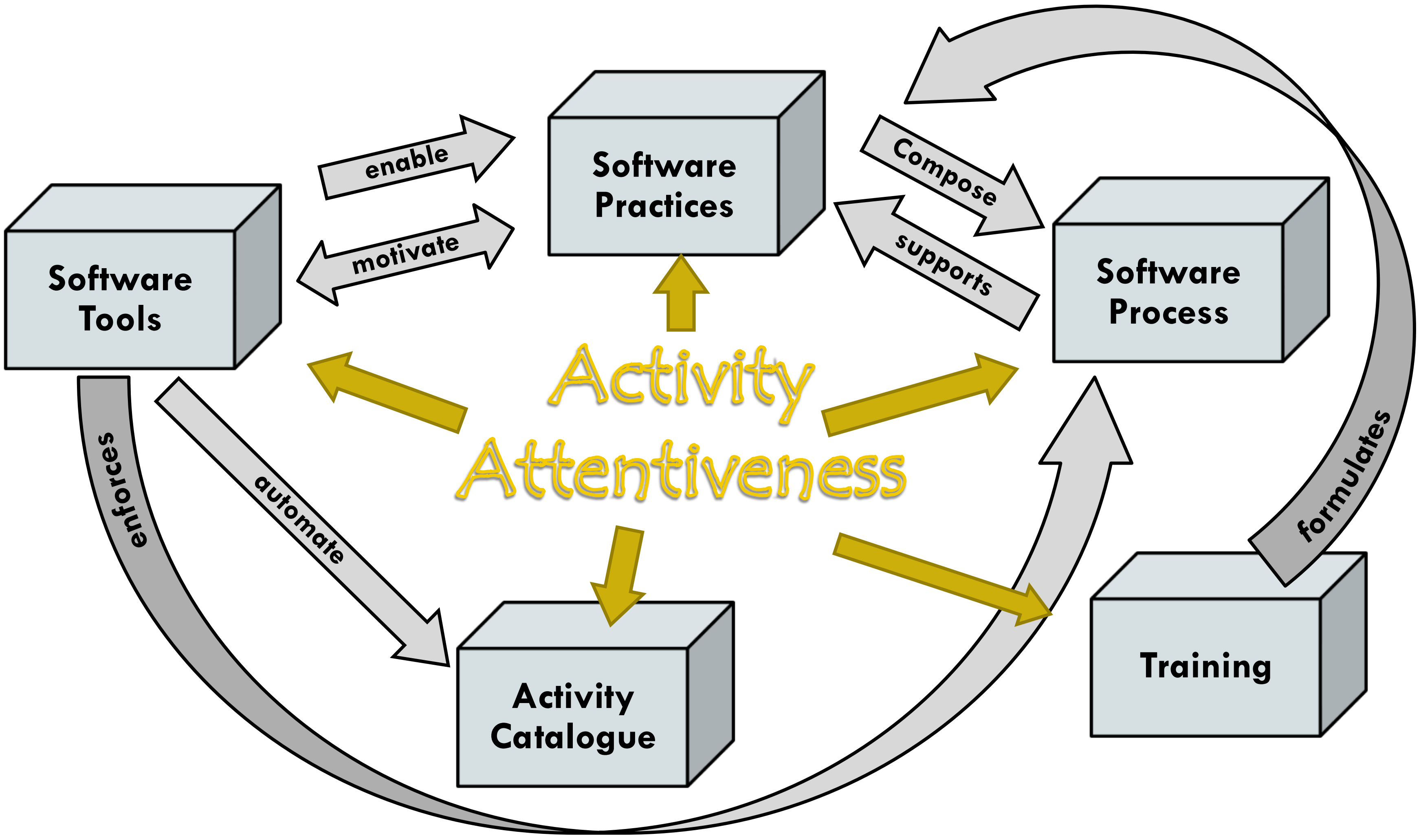
- Provided **name** and **definition** for the activity
- Laid the foundations for others to build a **catalogue**
- Enabled the development of **software tools**
- Promoted new **coding practices**
- Influenced the **development process**

These various aspects serve as an **ecosystem** that exploits the use of refactoring **systematically** and **methodically** to leverage its full potential and eliminate its **pitfalls** and **deficiencies**

Ecosystem



The Software Development Ecosystem

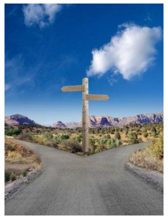


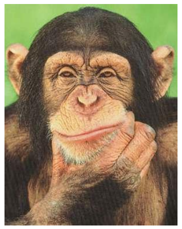
Future work: Example Embedding

- **Example Embedding** is the **notion** of using an already existing code fragment (the example) within a new context
- Investigate ways in which examples could be used more **systematically** more **extensively** and more **effectively** to exploit their full potential
- Further probe whether **productivity** benefits from using examples **habitually** and **correctly** in **example supportive environment**

Summary

- Example Attentiveness Observed
  - Context, Utilization, Scale
- Focus group case study
- Weaving example attentiveness into the software engineering ecosystem





Thank You

Discussion