Automatically Describing Program Structure and Behavior

Readability  Runtime Behavior  Documentation
Code is Difficult to Understand.
“Understanding code is by far the activity at which professional developers spend most of their time.”

Peter Hallam. *What Do Programmers Really Do Anyway?*  
Microsoft Developer Network (MSDN) – C# Compiler. Jan 2006
Maintenance accounts for about 70-90% of the total lifecycle budget of a software project.

```c
#include <math.h>
#include <sys/time.h>
#include <X11/Xlib.h>
#include <X11/keysym.h>

double L, o, P, t, T, Z, D = 1, d, s[999], E, h = 8, I, J, K, W[999], M, m, O, n[999], j = 33e - 3, i = 1E3, r, t, u, v, W, S = 74.5, l = 221, X = 7.26, a, B, A = 32.2, c, F, H;

int N, q, C, y, p, U;

Window z; char f[52];

main(){
    Display* e = XOpenDisplay(0);
    z = RootWindow(e, 0);
    for (;
        XMapWindow(e, z = XCreateSimpleWindow(e, z, 0, 0, 400, 400, 0, 0, WhitePixel(e, 0)),
            XClearWindow(e, z = XCreateGC(e, z, 0, 0), BlackPixel(e, 0)));
    for (;
        XSelectInput(e, z = XCreateKeyPressMask(e, z, 0, 0),
            XMapWindow(e, z));
        T = sin(O));
    for (;
        XPending(e);
        XEvent event;
        XNextEvent(e, &event);
        ++*((N = XLookupKeysym(&event.xkey, 0)) - IT? N-LT? UP-N? & E:
                J + u + 6h); --*(DNT-N? N-DT ?N==
                        KT?&u: & W:sh & 6j);
})

c += (I = M + 1, 1 + H + J + M + a X) * h; H = a r + v X - F * l + {E = 1.4 X 4.9 + l, t = T * m / 32 - 1 / T * 24 / S; K + F * M +
            h * 1 e4 / 1 - (T + E * 5 T E + 3 e 2 / S * X d * B A;
            a = 2.63 / 1 a d;
            X += (d = 1 - T / S * (1.9 + E + a * 6 + j / 1 e 3 - M v + A * Z) * l + K * W a +;
            sprintf(f, "%5d %3d %7d", p = l / 1.7, (C = 9 e 3 * 0 * 57.3) % 0 * 550, (int) i); d = t * (1.45 + 14 / t + X + 100 - 0.14) % 125 e 2 + F - 8 e 5 T * (1.47 * 1 - m 52 e 9 + d * t + 3.8 + u * 21 E / 1 e 2 W * 179 * v) / 2312; select(p = 0, 0, 0, 0, & G; v = (w * F - T * (1.6 + 1.08 + E + 9 - 25 - 25 - 11 u) / 0.174) * J; D = cos(o) + E = sin(o); })
```
Java
What does this print?

class Change {
    public static void main(String[] args) {
        System.out.println(2.00 - 1.10);
    }
}

*Adapted from Josh Bloch, Jeremy Manson*
What does this print?

class Change {
    public static void main(String[] args) {
        System.out.println(2.00 - 1.10);
    }
}

Output: 0.8999999999999999
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
What does this print?

```java
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
```

Output: 0.89999999999999991118215809299874766109466552734375
BigDecimal

public BigDecimal(double val)

Translates a double into a BigDecimal which is the **exact decimal representation of the double's binary floating-point value**. The scale of the returned BigDecimal is the smallest value such that \((10^{\text{scale}} \times \text{val})\) is an integer.

http://docs.oracle.com/javase/6/docs/api/java/math/BigDecimal.html
What we should have done

```java
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal("2.00");
        BigDecimal cost = new BigDecimal("1.10");
        System.out.println(payment.subtract(cost));
    }
}
```

Output: 0.90
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
The Rest of this Talk

Modeling Code Readability

Predicting Runtime Behavior

Synthesizing Documentation
```java
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
```
How can we tell if code is readable?
LIFE AND ADVENTURES
OF
NICHOLAS NICKLEBY
BY
CHARLES DICKENS

A REPRINT OF THE FIRST EDITION,
WITH THE ILLUSTRATIONS, AND AN INTRODUCTION,
BIOGRAPHICAL AND BIBLIOGRAPHICAL, BY
CHARLES DICKENS THE YOUNGER.

MACMILLAN AND CO., LIMITED
ST. MARTIN'S STREET, LONDON
1916
There once lived, in a sequestered part of the country of Devonshire, one Mr. Godfrey Nickleby: a worthy gentleman, who, taking it into his head rather late in life that he must get married, and not being young enough or rich enough to aspire to the hand of a lady of fortune, had wedded an old flame out of mere attachment, who in her turn had taken him for the same reason.

"I do not like them in a box. I do not like with a fox. I do not like them in a house. I do not like them with a mouse. ..."
I do like them not in a box

There
Mr Godfrey Nickleby (gentleman) lived in part a sequestered country of Devonshire

who had wedded flame

and out of attachment more

who had taken him far part her

young enough to aspire to hand of lady of fortune

rich enough not to"
Flesch-Kincaid Readability

I do not like them in a box.
I do not like them with a fox.
I do not like them in a house.
I do not like them with a mouse.
I do not like them here or there.
I do not like them anywhere.
I do not like green eggs and ham.
I do not like them, Sam-I-am.
Flesch-Kincaid Readability

I do not like them in a box.
I do not like them with a fox.
I do not like them in a house.
I do not like them with a mouse.

Readability
- Passive Sentences: 0%
- Flesch Reading Ease: 100.0
- Flesch-Kincaid Grade Level: 0.0
Flesch-Kincaid Readability

0.0

15.3
Flesch-Kincaid Readability

0.0

10.0 DOD MIL-M-38784B

15.3
Flesch-Kincaid Readability

0.0

10.0

DOD MIL-M-38784B

12.1

15.3

GREEN EGGS AND HAM

By Dr. Seuss

Automatically Detecting Program Structure and Behavior

A Dissertation

Prepared by

The Faculty of the School of Engineering and Applied Sciences

Cornell University

in Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy in Computer Science

Robert R. Obus

May 2012

LIFE AND ADVENTURES OF NICHOLAS NICKLEBY

By CHARLES DICKENS

A NOVEL OF THE FIFTY YEARS

WITH THE ILLUSTRATIONS, AND AN INTRODUCTION

BY MANUS BRIDGES THE JOSHER.

MACMILLAN AND CO., LIMITED.
ST. MARTIN'S STREET, LONDON.
1841.
Can this work for code?

Research questions:

• To what extent do humans agree on what code is readable?
• Can we derive an accurate descriptive model for readability?
• Does the model correlate significantly with software quality?
/**
 * Computes factorial with recursion
 */
public int factorial( int integer )
{
    if( integer < 1 )
        return 0;

    if( integer == 1)
        return 1;

    return integer * factorial( integer - 1 );

Snippet Pack demo: 2 of 4
Vertical bands imply agreement

More Readable

Less Readable

120 human annotators

100 snippets
NOT SURE IF HIGH AGREEMENT

OR JUST RED-GREEN COLOR BLIND
Quantifying Agreement
Quantifying Agreement

Correlation Statistics

- Pearson’s $r$ – linear dependence
- Spearman’s $\rho$ – monotonic dependence
- Kendall’s $\tau$ – counts bubble sort operations
- Cohen’s $\kappa$ – nominal agreement
Quantifying Agreement

Correlation Statistics

- Pearson’s $r$ – linear dependence
- Spearman’s $\rho$ – monotonic dependence
- Kendall’s $\tau$ – counts bubble sort operations
- Cohen’s $\kappa$ –

Absolute agreement less important than relative agreement
Quantifying Agreement

Perfect Absolute Agreement

\[ \rho = 1 \]
Quantifying Agreement

Perfect Relative Agreement

\[ \rho = 1 \]
Quantifying Agreement

Absolute Disagreement

\[ \rho = -1 \]
Quantifying Agreement

Random Agreement

\[ \rho = 0 \]
Quantifying Agreement

“Strong” Agreement

\[ \rho > 0.5 \]
Quantifying Agreement With a Group
Quantifying Agreement With a Group

Apples and Oranges: An Empirical Comparison of Commonly Used Indices of Interrater Agreement
Allan P. Jones, Lee A. Johnson, Mark C. Butler and Deborah S. Main
Quantifying Agreement With a Group

Compute all pairwise correlations

\[ \rho_0 \rho_1 \rho_2 \rho_3 \ldots \rho_{n-1} \]
Quantifying Agreement With a Group

\[ \rho = \frac{\sum \rho_0 \rho_1 \rho_2 \rho_3 \ldots \rho_{n-1}}{n} \]

Return Average \( \rho \)
Spearman correlation between annotators

Annotators sorted by agreement
Building a Model
Flesch-Kincaid Readability

0.39 \left( \frac{\text{wordCount}}{\text{sentenceCount}} \right) + 11.8 \left( \frac{\text{syllableCount}}{\text{wordCount}} \right) - 15.59
Flesch-Kincaid Readability

\[ 0.39 \left( \frac{\text{wordCount}}{\text{sentenceCount}} \right) + 11.8 \left( \frac{\text{syllableCount}}{\text{wordCount}} \right) - 15.59 \]

\[ f(\vec{x}) = \beta_0 + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_n(x_n) \]
Flesch-Kincaid Readability

$$0.39 \left( \frac{\text{wordCount}}{\text{sentenceCount}} \right) + 11.8 \left( \frac{\text{syllableCount}}{\text{wordCount}} \right) - 15.59$$

$$f(\hat{x}) = \beta_0 + \beta_1 (x_1) + \beta_2 (x_2) + \cdots + \beta_n (x_n)$$

Features
Flesch-Kincaid Readability

\[
0.39 \left( \frac{\text{wordCount}}{\text{sentenceCount}} \right) + 11.8 \left( \frac{\text{syllableCount}}{\text{wordCount}} \right) - 15.59
\]

\[
f(\vec{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n
\]

Features

Weights
Features

Creativity / Intuition

Weights

Supervised Learning
- Regression
- Bayesian
- Neural Net
- SVM
- ...

Use training data from human study
class Change {

    //Computes 2.00 – 1.10
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal("2.00");
        BigDecimal cost = new BigDecimal("1.10");
        System.out.println(payment.subtract(cost));
    }
}
Potential Code Readability Features

class Change {

    //Computes 2.00 - 1.10
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal("2.00");
        BigDecimal cost = new BigDecimal("1.10");
        System.out.println(payment.subtract(cost));
    }
}

Line Length
class Change {

//Computes 2.00 - 1.10

public static void main(String[] args) {
    BigDecimal payment = new BigDecimal("2.00");
    BigDecimal cost = new BigDecimal("1.10");
    System.out.println(payment.subtract(cost));
}
}
Potential Code Readability Features

class Change {

    //Computes 2.00 - 1.10
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal("2.00");
        BigDecimal cost = new BigDecimal("1.10");
        System.out.println(payment.subtract(cost));
    }
}

Identifier Length
class Change {

    // Computes 2.00 - 1.10
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal("2.00");
        BigDecimal cost = new BigDecimal("1.10");
        System.out.println(payment.subtract(cost));
    }
}
Evaluation
Model agrees with humans as much as they agree with each other
Line Length, # of identifiers is important
Length of identifiers is not important.
Readability and Software Quality
Benchmarks
12 Open Source Sourceforge Projects, Over 2M LOC
Mature projects tend to be more readable
JUnit 3.0 -> 4.0
Readability Metric

• Metric (and source code) is freely available: arrestedcomputing.com/readability

• Has been used directly in several published papers.
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
Predicting Runtime Behavior
Approaches to Predicting Behavior

Dynamic Profiles
- Precise - full program path profiles
- Requires indicative workloads

Static Heuristics
- Cheap
- Only need program code
- Typically limited in scope
IF YOU REQUIRE WORKLOADS
YOU’RE GONNA HAVE A BAD TIME
Static Heuristics


Intra-class static path profiles

• Precision similar to a dynamic profiler
• Workloads not required
Key idea
public V put(K key, V value) {
    if (value == null)
        throw new Exception();

    if (count >= threshold)
        rehash();

    index = key.hashCode() % length;

    table[index] = new Entry(key, value);
    count++;

    return value;
}
```java
public V put(K key, V value) {
    if (value == null)
        throw new Exception();

    if (count >= threshold)
        rehash();

    index = key.hashCode() % length;

    table[index] = new Entry(key, value);
    count++;

    return value;
}
```

*from java.util.HashTable jdk6.0*
Static Path Profiling

Research questions:

• What static code features are predictive of path execution frequency?
• Can we derive an accurate descriptive model for runtime behavior?
Approach

• Build a descriptive model of path execution frequency
• Features: length of path, presence of exceptions, number of variables written ...
• Train and cross-validate on SPEC Java benchmarks
Choose **5%** of all paths and get **50%** of runtime behavior.

Ranking by our metric

Baseline: random ranking

**Time spent on paths**

**Percent of all paths selected**
Choose 1 path per method and get 94% of runtime behavior.

Baseline: random ranking

Ranking by our metric

Time spent on paths

Number of paths selected per method
Applications for Profiles

• Profile guided optimization
• Complexity/Runtime estimation
• Anomaly detection
• Significance of difference between program versions
• Prioritizing output from other analyses
• Documentation
Conclusion

- A formal model that statically predicts relative dynamic path execution frequencies
- The promise of helping other program analyses and transformations
import java.math.BigDecimal;

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(1.10);
        System.out.println(payment.subtract(cost));
    }
}
Documentation Synthesis
Classic Approaches to Understandability

Improving
- Code Reviews
- Training
- Languages
- Documentation

Compensating
- Testing
- Verification
- Other Program Analyses
Classic Approaches to Understandability

Improving

• Code Reviews
• Training
• Languages
• Documentation

Compensating

• Testing
• Verification
• Other Program Analyses

Doesn't solve underlying problem

Labor Intensive
Use these tools...

- Compensating
- Testing
- Verification
- Other Program Analyses
Use these tools...

Improving

• Code Reviews
• Training
• Languages
• **Documentation**

To do this.
The most significant barrier to code reuse is “software is too difficult to understand or is poorly documented.”

Source Code Documentation

• Describes some **important** aspect of the code in a way that’s **easier to understand**.

• Explanations/Summaries of Behavior
• Pre/Post Conditions, Caveats
• Usage Examples
• ...


Automatic Documentation

Program Code

Synthesis Tool

Documentation

When calling `LastPage format(String s)`
If s is not `null` and `s.split("[-]+")`.length != 2
return `s.split("[-]+")[0]` instead of `null`

When calling `EntryEditor getExtra()`
If `ed.getFieldName().equals("editor")`
call `contentSelectors .add(FieldContentSelector)`
Automatic Documentation

- Cheap
- Always up-to-date
- Complete
- Well-defined trust properties
- Structured (Searchable)
Exceptions

`IllegalStateException` thrown when `getLocation()` is not Europe


API Usage Examples

```java
Iterator iter = SOMETHING.iterator();
while( iter.hasNext() )
{
    Object o = iter.next();
    //Do something with o
}
```


Program Changes

When `arg0 == null` return `-1` instead of `arg0.toString()`

Exceptions

IllegalStateException thrown when
getLocation() is not Europe


API Usage Examples

Iterator iter =
    SOMETHING.iterator();
while( iter.hasNext() )
{
    Object o = iter.next();
    //Do something with o
}


Program Changes

When arg0 == null
    return -1 instead of
    arg0.toString()

“The greatest obstacle to learning an API ... is insufficient or inadequate examples”

Synthesizing API Examples

Research questions:

• What makes a good example?
• Can we create good examples automatically?
Sources of Examples

- stackoverflow
- JExamples
- msdn
- JavaDoc
Search-based examples

```java
BufferedReader reader = new BufferedReader(new InputStreamReader(page));
try {
    String line = reader.readLine();
    while (line != null) {
        if (line.matches(substituteWikiWord(wikiWord, newTopicPattern))) {
            Query: BufferedReader
        }
    }
}
```
FileOutputStream fos = new FileOutputStream("t.tmp");
ObjectOutputStream oos = new ObjectOutputStream(fos);
oos.writeInt(12345);
oos.writeObject("Today");
oos.writeObject(new Date());
oos.close();
Hand-Crafted Examples

Abstract Initialization

```java
int glyphIndex = ...;
GlyphMetrics metrics = GlyphVector.getGlyphMetrics(glyphIndex);
int isStandard = metrics.isStandard();
float glyphAdvance = metrics.getAdvance();
```

Query: java.awt.font.GlyphMetrics
Hand-Crafted Examples

```java
for(char c = iter.first();
    c != CharacterIterator.DONE;
    c = iter.next()) {
    processChar(c);
}
```

Query: `java.text.CharacterIterator`

```java
try {
    file.delete();
} catch (IOException exc) {
    // failed to delete, do error handling here
}
return FileVisitResult.CONTINUE;
```

Query: `java.nio.FileVisitor`
Our API Examples

```java
FileReader fReader; //initialized previously
BufferedReader br = new BufferedReader(fReader);
while (br.ready()) {
    String line = br.readLine();
    //do something with line
}
br.close();
```

Query: java.util.BufferedReader
Our API Examples

```java
FileReader fReader; //initialized previously
BufferedReader br = new BufferedReader(fReader);
while (br.ready()) {
    String line = br.readLine();
    //do something with line
}
br.close();
```

**Complete**

**Common variable names**

**Abstract Initialization**

**“Holes”**
Synthesis
Approach

• **Mine** usages from an existing program corpus
  – Similar to *Specification Mining*
• **Learn** common patterns
• **Abstract** representative examples
Software Corpus

Miner

Search Term

“BufferedReader”

Control Flow Graph Representation

<init>

ready

readLine

print

close

readLine

print
Software Corpus

Search Term

"BufferedReader"

Miner

Search Term

FileReader

argument

"foo.txt"

With Annotations Relevant to Documentation

argument

br

identifier

ready()

predicate

ready()

returns

String:

line

argument

line

argument

id_identifier

argument

init

ready

readLine

print

close
Many Concrete Uses
Multiple Common Patterns

BufferedReader br = new BufferedReader(new FileReader("foo.in"));
while( br.ready() )
{
    String s = br.readLine();
    //Do something with s
}

BufferedReader br = new BufferedReader(new FileReader("foo.in"));
String s;
while( ( s = br.readLine() ) != null )
{
    //Do something with s
}
Clustering: Group Similar Patterns

Many Concrete Uses

... Grouped Into A Few Different Patterns
Clustering

• k-medoids
• Distance metric captures difference in order of statements and types of objects
Abstraction

Many Concrete Examples

... Into One Abstract Example
Concrete

```java
if (iter.hasNext()) {
    set.add(iter.next());
}
```

Concrete

```java
if (iter.hasNext()) {
    print(iter.next());
}
```

**Abstraction Operator**

Least-upper-bound types

```java
if (iter.hasNext()) {
    Object o = iter.next();
    // Do something with o
}
```

Insert Hole
Iterator iter = set.iterator();
...

Concrete

Concrete

Iterator iter = list.iterator();
...

Concrete

Concrete

Iterator iter = SOMETHING.iterator();
...

Abstraction
Operator

Abstract
Initialization
Recap

Software Corpus

Search Term

Mine

Cluster

Abstract

Yield Well-Formed Source Code

Code Gen

```
Calendar calendar = Calendar.getInstance();
Date d = calendar.getTime();
//Do something with d
```
Examples

```java
Calendar calendar = Calendar.getInstance();
Date d = calendar.getTime();
//Do something with d
```

Query: java.util.Calendar
Examples

```java
String regex; //initialized previously
String input; //initialized previously
Pattern pattern = Pattern.compile(regex);
Matcher m = pattern.matcher(input);
//Do something with m
```

Query: java.util.regex.Pattern
Limitations

• Can’t always be perfectly precise
  – E.g., Aliasing, Types
  – Conservative analysis preserves correctness
• Common usage is not always best
  – E.g., poor exception handling
  – Guarantee representative examples
• Not all APIs have indicative patterns
• Some patterns are difficult to find
  – Message passing over network etc.
Evaluation
API Examples Study

java.util.StringTokenizer

If you had to use this class, which of these examples would you prefer?

Example A

```java
public void sendMessage(Message message, Address[] addresses) throws MessagingException, SendFailedException{
    if (!isConnected()){
        throw new MessagingException("not connected");
    }
    if (!(message instanceof MimeMessage)){
        throw new SendFailedException("only MimeMessages are supported");
    }
    MimeMessage mimeMessage = (MimeMessage) message;
```

Example B

```java
String str; //initialized previously
StringTokenizer st = new StringTokenizer(str);
while(st.hasMoreTokens()) {
    String s = st.nextToken();
    //Do something with s
}
```

Participant specifies preference
Comparison to Code Search
Comparison to Human-Written

![Bar chart comparing examples of Java packages to human-written samples. The chart shows the percentage of examples preferred by humans, with bars split to indicate neutrality and preference for 'Our Tool'. The sample sizes range from 16 to 138 examples.]

- java.awt: 80% Neutral, 20% Our Tool
- java.io: 90% Neutral, 10% Our Tool
- java.lang: 80% Neutral, 20% Our Tool
- java.text: 70% Neutral, 30% Our Tool
- java.util: 50% Neutral, 50% Our Tool
- ugrads (138): 40% Neutral, 60% Our Tool
- grads (16): 30% Neutral, 70% Our Tool
- all (154): 50% Neutral, 50% Our Tool
Use Cases
Warnings for Likely Mistakes

class Change {
    public static void main(String[] args) {
        BigDecimal payment = new BigDecimal(2.00);
        BigDecimal cost = new BigDecimal(2.00);
        System.out.println(payment);
    }
}
Auto Completion

```java
public class ReadFile {
    public void read(File fileToRead) {
        FileReader fReader = new FileReader(fileToRead);
    }
}
```
Conclusion
class Change {
    public static void main(String[] args) {
        BigDecimal p = new BigDecimal(2.00);
        System.out.println(p);
    }
}
Readability


Runtime Behavior


Documentation


http://arrestedcomputing.com
Thank You!
Questions?

Also ask me about:

• Documenting Exceptions
• Generating Commit Messages
• Conducting Human Studies
IF YOU DON'T LIKE TO ASK QUESTIONS

YOU SHOULD NOT HAVE COME TO A DISSERTATION DEFENSE