An Experiment in Developing Small Mobile Phone Applications: Comparing On-Phone to Off-Phone Development

Tuan A. Nguyen,
Sarker T.A. Rumee,
Christoph Csallner
(University of Texas at Arlington)

Nikolai Tillmann
(Microsoft Research)

This material is based upon work supported by the National Science Foundation under Grants No. 1017305 and 1117369. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
New Development Style: On-Phone

```
action temperature_converter()
  var n := wall.ask_number("Enter temp for "...)
  var x := (n - 32) * 5 / 9
  wall.prompt("Temp in Celsius": x)
```

Semi-structured multi-touch IDE

Interpreter within IDE
Traditional Development Style: Off-Phone

Android as an example, but Windows Phone, iOS, etc. work similarly
Resulting Apps May Look Very Similar

Example: Convert degrees Fahrenheit to degrees Celsius

TouchDevelop

Android
On-Phone vs. Traditional Off-Phone Development

Research questions (RQ), expectations (E), and hypotheses (H)

RQ1: How large are TouchDevelop apps?
   E1: Tiny phone screen $\rightarrow$ expect most TouchDevelop apps to be small
   H1: TouchDevelop Apps Are Small

RQ2: For given task: TouchDevelop-LOC vs. Android-LOC
   E2: TouchDevelop specialized, assumes and hides details $\rightarrow$ expect TouchDevelop apps to be smaller
   H2: TouchDevelop-LOC < Android-LOC

RQ3: Programmer productivity: TouchDevelop vs. Android
   E3: Tiny phone screen, no keyboard, no mouse $\rightarrow$ expect TouchDevelop programmers to be less productive
   H3: TouchDevelop-Productivity < Android-Productivity
RQ1: How Large Are TouchDevelop apps?

Count LOC of all TouchDevelop apps in TouchDevelop cloud

Counting TouchDevelop LOC (and Android LOC)

Normalize TouchDevelop and Android apps
Do not count content of configuration files (XML, text, etc.)
RQ2 & RQ3: Experiment on Student Subjects

27 students of CSE 5324 software engineering class

25 MS CS + 1 MS CE + 1 MS SE, taught by Csallner, Rume: TA
In this class students (expect to) work on big Android team project (team = 5 students)
Experiment conducted toward end of semester → students have some Android experience

10 Samsung Focus Windows 7 Phones from Microsoft

University lab with > 17 lab PCs
Randomly assigned subjects to:
  10 WP7 phones with pre-installed TouchDevelop v2.4.0.0 beta
  17 lab PCs with pre-installed Eclipse + Android SDK v 1.6

One class period

10 min informed consent & phone loan forms + 60 min tasks + 10 min questionnaire
Stressed that participation does not influence grades, provided link to respective APIs
Individual development: Can consult samples, web, no other communication except with instructor/TAs
Windows Phone subjects

**Did not receive training in TouchDevelop**
- Received link to TouchDevelop website and 2-minute intro video

**Did 5-minute phone setup**
- Setup wireless internet connection
- Enter assigned fresh Windows Live ID
- Download & install TouchDevelop

**TouchDevelop comes with samples**

**Students not allowed to use another device**
- Simulate phone-only development
- Developed apps published to cloud by TA

Android lab PC subjects

**Had taught themselves Android for class project**

**Did 5-min PC setup**
- Download & install Android Development Tools
- Create & start virtual device (emulator)

**Wizard generates working “Hello World” program**

**E-mailed source of developed apps to TA**
- 2 subjects failed to do that
  - Left with 15 Android subjects
Subjects Had Little Prior TouchDevelop Knowledge

Asked for hours spent before experiment on learning TouchDevelop (TD), Android (An), Eclipse (Ec)

- Median #hours spent learning TouchDevelop = 0
- Median #hours spent learning Android = 30
- Median #hours spent learning Eclipse = 55
Prior LOC: TouchDevelop < Android < Java

Asking for LOC written before experiment in TouchDevelop (TD), Android (An), non-Android Java (J), C#

- Median TouchDevelop LOC written before experiment = 0
- Median Android LOC written before experiment = 250
- Median non-Android Java LOC written before experiment = 1,000
(P1) Any "Hello World" program that prints "CSE 5324" on the screen.
(P2) A program that takes as input an integer number representing degrees Fahrenheit, converts it to degrees Celsius (using the Fahrenheit to Celsius conversion rule: deduct 32, multiply by 5, then divide by 9), and prints the resulting value.
(P3) A tip calculator that takes as input two integer numbers A, B from the user and prints the value of A*B/100.
(P4) A program that takes as input an integer number and prints "even" if it is an even number and "odd" if it is an odd number.
(P5) A program that takes as input a string and a character, prints "contains" if the string contains the character or else prints "not in there".
(P6) A program that takes as input a string and prints out the string with first character in uppercase.
(P7) A program that prints the system’s current time as text.
(P8) A program that asks the user for a positive integer value n and prints odd numbers between 0 and n (including n if n is odd).
(P9) A program that takes as input a string that consists of numbers separated by commas. The program should output the numbers in increasing order.
(P10) A program that draws a circle on the screen.
(P11) A program that takes two strings as input and checks if they are equal.
RQ1: How Large Are TouchDevelop apps?
Result: TouchDevelop Apps Are Small

2,081 apps in the TouchDevelop cloud (17 Feb 2012)

Two largest apps of 1,742 and 1,675 LOC not shown

47% are ≤ 9 LOC
60% are ≤ 19 LOC
RQ2: For Given Task: TouchDevelop LOC vs. Android LOC
On Same Task: TD-LOC < Android-LOC

Correct solutions: TouchDevelop-LOC \approx 4 \text{ Android-LOC}

Correctness judged manually, width is proportional to \#correct solutions we received
TouchDevelop App Has Less Code

Example: Convert degrees Fahrenheit to degrees Celsius

TouchDevelop

```
action: temperature converter()
var n := wall - ask number("Enter temp for ")
var x := (n - 32) * 5 / 9
wall = prompt("Temp in Celsius" ... 1 x)
```

Android

```
package com.cse5324.p2;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class P2Activity extends Activity implements OnClickListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        t1 = (TextView) findViewById(R.id.textView2);
        e1 = (EditText) findViewById(R.id.editText);
        b1 = (Button) findViewById(R.id.button);
        b1.setOnClickListener(this);
    }

    public void onClick(View v) {
        // TODO Auto-generated method stub
        Double cel = Double.parseDouble(e1.getText().toString());
        cel = ((cel - 32) * 5 / 9);
        t1.setText(cel.toString());
    }
}
```

Less general, focused on basic mobile device functions
+ Less configuration
+ Less code
RQ3: Programmer Productivity: TouchDevelop vs. Android
TouchDevelop-Productivity > Android-Productivity

TouchDevelop subjects on average started and finished more apps

<table>
<thead>
<tr>
<th>System</th>
<th>Subjects</th>
<th>Apps / subject</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Some code</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>TouchDevelop</td>
<td>10</td>
<td>3.7</td>
<td>2.4</td>
<td></td>
</tr>
<tr>
<td>Android</td>
<td>15</td>
<td>3.2</td>
<td>1.7</td>
<td></td>
</tr>
</tbody>
</table>

Graph showing the percentage of subjects (%) for different apps (1 to 11) across Phone and PC platforms.
Why Were TD Subjects More Productive?

Possible explanations

**TouchDevelop is more focused on tasks**
- Less configuration and setup in TouchDevelop

**Modern language features: Type-inference**
- Convenient for small tasks

**Semi-structured TouchDevelop IDE**
- Why traditional IDEs do not have this feature?
  - Traditional development may benefit from semi-structured IDEs
Our subjects are not a random sample
Study may not generalize well to novice or hobbyist programmers world wide
Subjects all UTA students
Subjects self-selected graduate software engineer course that has a large Android project
But subjects were not aware of this experiment

Hands-off administration of tasks
Did not instruct subjects on how to work on the given tasks
Time limits for each task may produce results that are easier to compare, but prevents subjects from switching back to earlier tasks and reusing later solutions in earlier tasks

Results may not generalize to larger programs
Limited mobile phone screen size and the limited amount of time
Designed all tasks to be simple, small, and solvable with both Android and TouchDevelop
Program size increases → TouchDevelop may need more scrolling & navigation → lower productivity
Conclusions & Future Work
Programmers so far have written small TouchDevelop apps

Experiment comparing on-phone to off-phone development

Small programming tasks
Student subjects
Subject training: Android > TouchDevelop

→ TouchDevelop LOC < Android LOC

→ TouchDevelop productivity > Android productivity
Future Work

Why were TouchDevelop subjects more productive?
How large is the impact of the semi-structured IDE vs. the other TouchDevelop components? E.g.: Observe programmers on semi-structured IDE vs. on un-structured version of same IDE

What happens for larger programs?
Challenging to write large programs on small screen

Are TouchDevelop maintainers also more productive?
Observe subjects as they add/change a small feature in an third-party TouchDevelop app

Are TouchDevelop testers also more productive?
How do you debug and test code on a phone?
Technical paper

“An experiment in developing small mobile phone applications comparing on-phone to off-phone development”
By Tuan A. Nguyen, Sarker T.A. Rumee, Christoph Csallner, and Nikolai Tillmann.

Corpus of TouchDevelop apps,
Experiment tasks & resulting TouchDevelop & Android apps,
Questionnaire, tools:

http://cseweb.uta.edu/~tuan/tdexp/
More Details
Background of This Experiment

Csallner’s programming experience mostly Java

Teaches project-based Software Engineering course
Team project, each team develops piece of software, covers major development phases
Current hot topic: Mobile app development skills
Pick Android as it is a major platform and matches instructor’s Java experience

TouchDevelop comes along
Radical new approach to mobile app development

What are the trade-offs?
Counting LOC: Logical Source Statements

Example: Input an integer number and print "even" if it is an even number and "odd" if it is an odd number.

TouchDevelop LSS: 8

```plaintext
action main()
  var x := wall -> ask number,
  "Enter an integer"...
  while x ≥ 1 do
    x := x - 2
    if x ≠ 0 then
      "Odd integer" -> post to wall
    else
      "Even integer" -> post to wall
```

Android LSS: 25

```plaintext
package com.cs5324.p4;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class PAActivity extends Activity implements OnClickListener {
  /** Called when the activity is first created. */
  TextView t1;
  EditText e1;
  Button b1;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    t1 = (TextView) findViewById(R.id.textView);
    e1 = (EditText) findViewById(R.id.editText);
    b1 = (Button) findViewById(R.id.button);
    b1.setOnClickListener(this);
    
    // TODO Auto-generated method stub
    Double cel = Double.parseDouble(e1.getText().toString());
    if (cel % 2 = 0) {
      t1.setText("Number is even");
    } else {
      t1.setText("Number is odd");
    }
  }
```
Correct Apps Per Subject

Higher percentage of TouchDevelop subjects finish 1, 2, 3 apps
Higher percentage of Android subjects finish 4 apps
Subject Reusing Android Apps Between Tasks

“Hello world” template (left) vs. “Print CSE 5324” (right)

```java
package utsa.edu.cse5324.sample;
import android.app.Activity;
import android.os.Bundle;

public class HelloWorldActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}

package com.cse5324;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class P1Activity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView Label = (TextView) findViewById(R.id.label);
        Label.setText("CSE 5324");
    }
}
```

- Lookup text view and change text to “CSE 5324”
- Could have implemented this change without changing code, just replace “Hello World” string in XML configuration file
Subject Reusing Android Apps Between Tasks

°F to °C conversion (left) vs. Tip Calculator (right)

<table>
<thead>
<tr>
<th>Package</th>
<th>Code Snippet</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.cse5324.p2</td>
<td>import android.app.Activity; import android.os.Bundle; import android.view.View; import android.view.View.OnClickListener; import android.widget.Button; import android.widget.EditText; import android.widget.TextView; public class P2Activity extends Activity implements OnClickListener { /** Called when the activity is first created. */ TextView t1; EditText e1; Button b1; @Override public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.main); t1 = (TextView) findViewById(R.id.textView2); e1 = (EditText) findViewById(R.id.editText1); b1 = (Button) findViewById(R.id.button1); b1.setOnClickListener(this); } public void onClick(View v) { // TODO Auto-generated method stub Double cel = Double.parseDouble(e1.getText().toString()); cel = ((cel - 32) * 5) / 9; t1.setText(cel.toString()); } }</td>
<td>Requires addition to XML configuration file (not shown)</td>
</tr>
<tr>
<td>com.cse5324.p3</td>
<td>import android.app.Activity; import android.os.Bundle; import android.view.View; import android.view.View.OnClickListener; import android.widget.Button; import android.widget.EditText; import android.widget.TextView; public class P3Activity extends Activity implements OnClickListener { /** Called when the activity is first created. */ EditText e1, e2; @Override public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.main); Button b1; t1 = (TextView) findViewById(R.id.textView2); e1 = (EditText) findViewById(R.id.editText1); e2 = (EditText) findViewById(R.id.editText2); b1 = (Button) findViewById(R.id.button1); b1.setOnClickListener(this); } public void onClick(View v) { // TODO Auto-generated method stub Double result = (Double.parseDouble(e1.getText().toString()) * Double.parseDouble(e2.getText().toString())) * 100; t1.setText(result.toString()); } }</td>
<td>Modify result expression</td>
</tr>
</tbody>
</table>
Prior Experience vs. Success in Experiment

Relation between app success and prior experience was much stronger for Android subjects

Number of apps implemented with desired functionality vs. Prior experience across programming languages (kLOC)
How many of the following have you done before this exercise?

Lines of code written, counting all programming languages: ________
(do not include plain html, but include JavaScript, C, C++, C#, Java, etc.)
Lines of (non-Android) Java code written: ________
Lines of C# code written: ________
Lines of Java for Android code written: ________
Lines of TouchDevelop code written: ________
Hours spent working with Eclipse (write Java code, etc..): ________
Hours spent learning TouchDevelop (watch video, read website, api, etc.): ________
Hours spent learning Android (watch video, read website, api, etc.): ________
In completing this exercise, which problems did you encounter?

Preparing the IDE, emulator, etc.:
Developing particular apps:
Loading apps into the device:
Other (please elaborate):

In completing this exercise, which sources did you use (web sites, etc.)?

Samples that were part of the tool:
Official API documentation:
Examples found on the web:
Other (please elaborate):
Comparing these sources with other documentation you have used in the past, how useful were the sources you used in this experiment?

Samples that were part of the tool:
Official API documentation:
Examples found on the web:
Other (please elaborate):

Which aspects of this exercise did you particularly enjoy?

Please let us know any additional comments you may have.
Sources Used By Subjects During Experiment

TouchDevelop subjects mainly used code samples
Android subjects: API sources and web sources

<table>
<thead>
<tr>
<th>Source used</th>
<th>Android (%)</th>
<th>TouchDevelop (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code samples</td>
<td>13</td>
<td>80</td>
</tr>
<tr>
<td>API</td>
<td>53</td>
<td>0</td>
</tr>
<tr>
<td>Web sources</td>
<td>53</td>
<td>20</td>
</tr>
<tr>
<td>Other sources</td>
<td>7</td>
<td>20</td>
</tr>
</tbody>
</table>
Acknowledgements and Credits

Microsoft Research Connections
For providing Windows Phones for the duration of the semester

National Science Foundation
This material is based upon work supported by the National Science Foundation under Grants No. 1017305 and 1117369. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Tuan ("Tony")
A. Nguyen
University of
Texas
Arlington

Sarker T.A.
Rumee
University of
Texas
Arlington

Nikolai
Tillmann
Microsoft
Research
Redmond