Do Now:

1. Say hello to your neighbor!
2. Think of an OS you use. Discuss what you use it for and how you trust it.

CS111 Lecture
OS Trust in Context

Benjamin Xie, Ph.D.
Embedded Ethics Fellow
benjixie@stanford.edu | benjixie.com

made with William Grant Ray III, Xiyu Zhang, Liana Keesing, Swayam Parida, Prof. Nick Troccoli, Prof. John Ousterhout
year = ORIGINYEAR; /* = 1980 */

while (days > 365) {
    if (IsLeapYear(year)) {
        if (days > 366) {
            days -= 366;
            year += 1;
        }
    }
    else {
        days -= 365;
        year += 1;
    }
}

The following code handles the clock driver for a device. This code contains a bug that affects the device one day every four years. Discuss with your neighbor why.
Edge cases, y’all

year = ORIGINYEAR; /* = 1980 */

while (days > 365) {
    if (IsLeapYear(year)) {
        if (days > 366) {
            days -= 366;
            year += 1;
        }
    } else {
        days -= 365;
        year += 1;
    }
}

No else condition
=> stuck in while loop on last day of leap year (until days > 366)
The day the Zune stood still

On 31 Dec 2008 (last day of leap year), Zune 30s froze.

Solution: let battery run out and bug will disappear next day 🙁

Microsoft discontinued Zunes less than 4 years later…
What is needed to build a trustworthy OS?
How does the deployment context matter?
Trusting software is extending agency

- *agency*: our capacity to take actions that align with our goals
- “when we trust, we try to make something a part of our agency... To unquestioningly trust something is to let it in—to attempt to bring it inside one’s practical functioning.”
- Example: glucose monitoring

CT Nguyen: *Trust as an unquestioning attitude*
Risk: Agential Gullibility

- Trusting more than warranted
- Difficult to b/c software changes, hard to inspect
- Example: glucose monitoring issues w/ Android update

Android 13: Dangerous disconnections to blood glucose meters

Simon Lüthje · 17, February 2023
Three paths to trust

1. Assumption: trust absent any clauses to warrant it
   a. E.g. using unknown third party library b/c deadline nearing

2. Inference: reputation is based on past performance, characteristics, institutions
   a. Some weaker (e.g. trust in brands or affiliation)
   b. Some stronger (e.g. past performance)
   c. Trust in prior versions of software

3. Substitution: structural arrangements that partly replace need for trust
   a. Often involves separation of code, responsibilities
   b. E.g. user permissions of file system, keeping personal info off work accounts, devices

Paul B. de Laat: How can contributors to open-source communities be trusted? On the assumption, inference, and substitution of trust
So many kinds of OS!

Proprietary
- iOS
- Windows
- SteamOS
- chromeOS

Open Source
- Android
- Chrome OS
- Linux

Embedded Systems (e.g. IoT)
- VxWorks
- freeRTOS
- HelenOS

Research & Edu
- ReactOS
As a systems programmer, what is your responsibility as someone designing and maintaining operating systems?
today in privacy policies

- We retain your device/IP data for as long as we need it to ensure that our systems are working appropriately, effectively and efficiently.

**Personal Data of Children**

As noted in the Terms of Use, we do not knowingly collect or solicit Personal Data about children under 13 years of age; if you are a child under the age of 13, please do not attempt to register for or otherwise use the App. Use of the app under the age of 13, but no Personal Data about the child is collected. If we learn we have collected Personal Data from a child under 13 years of age, we will delete that information as quickly as possible. If you have concerns about how your child's information is being used or if you would like to request that we delete it, please contact us.

---

**ars technica**

Apple TV apparently requires iCloud terms and conditions to be accepted using a device running iOS 16 or iPadOS 16.

Apple TV users are now required to accept terms on iPhone or iPad.

---

**Tech Crunch**

6:20 PM • May 15, 2023
Dimensions of Context

**Stakeholders**
- Types: Direct, indirect
- Other considerations: Non-targeted use, changing hands, one person multiple roles

**Values**
- Explicit vs collateral
- Value tensions
- Different perceptions of same value

**Pervasiveness**
- How widespread?
- For what? (personal, critical infrastructure)
- Connected w/ what?
- Cultural and political implications?
- Crossing national boundaries?

**Time**
- Support duration
- Obsolescence
- Reappropriation
Dimensions of Context

Stakeholders
Types: Direct, indirect
Other considerations: Non-targeted use, changing hands, one person multiple roles

Pervasiveness
How widespread?
For what? (personal, critical infrastructure)
Connected w/ what?
Cultural and political implications?
Crossing national boundaries?

Values
Explicit vs collateral
Value tensions
Different perceptions of same value

Time
Support duration
Obsolescence
Reappropriation
**Stakeholders**

**Direct stakeholders**: directly interact w/ system
- users
- app developers
- system programmers

**Indirect stakeholders**: affected by system w/o directly using it
- customers
- patients
- everyone?!

**Additional considerations**:
- **non-targeted use**: tech not always used the way designers intended
- **changing hands**: stewardship of OS handed off between systems or organizations
- **one person, multiple roles**: same person can be direct and indirect stakeholder
**Example: Therac 25**

- Radiation therapy machine
- 1985-87: 6 patient deaths (overdoses of radiation)
- dual-mode: switch between by
  - low energy electron: topical cancer
  - high energy X-ray beams: deep cancers

<table>
<thead>
<tr>
<th>Therac-6</th>
<th>Therac-20</th>
<th>Therac-25</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Photon-mode</td>
<td>- <strong>Dual-mode</strong></td>
<td>- Dual-mode</td>
</tr>
<tr>
<td>- Hardware safety features</td>
<td>- Hardware safety features</td>
<td>- Some hardware safety features replaced with software</td>
</tr>
</tbody>
</table>
Race condition in Therac-25

<table>
<thead>
<tr>
<th>PATIENT NAME: John</th>
<th>BEAM TYPE: E</th>
<th>ENERGY (KeV):</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREATMENT MODE: FIX</td>
<td>ACTUAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PRESCRIBED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNIT RATE/MINUTE</td>
<td>0.000000</td>
<td>0.000000</td>
<td></td>
</tr>
<tr>
<td>MONITOR UNITS</td>
<td>200.000000</td>
<td>200.000000</td>
<td></td>
</tr>
<tr>
<td>TIME (MIN)</td>
<td>0.270000</td>
<td>0.270000</td>
<td></td>
</tr>
<tr>
<td>GANTRY ROTATION (DEG)</td>
<td>0.000000</td>
<td>0.000000</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>COLLIMATOR ROTATION (DEG)</td>
<td>359.200000</td>
<td>359.200000</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>COLLIMATOR X (CM)</td>
<td>14.200000</td>
<td>14.200000</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>COLLIMATOR Y (CM)</td>
<td>27.200000</td>
<td>27.200000</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>WEDGE NUMBER</td>
<td>1.000000</td>
<td>1.000000</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>ACCESSORY NUMBER</td>
<td>0.000000</td>
<td>0.000000</td>
<td>VERIFIED</td>
</tr>
</tbody>
</table>

DATE: 2012-04-16
TIME: 11:48:56
OPR ID: 033-tfs3p
REASON: OPERATOR
COMMAND: AUTO
Not considering context, poor SWE

Poor software engineering practices

1. all code written by single programmer
2. no formal software specifications
3. no testing strategy
4. no external review
5. uninformative error messages

Failure to consider stakeholders:

Direct stakeholders: medical technician, programmer, service technician

Indirect stakeholders: patients

Non-targeted use: technicians worked too quickly

Changing hands: used buggy software from previous versions (which had redundant hardware safety)
Modern medical software undergo formal verification

Aijaz Fatima
Dimensions of Context

**Stakeholders (direct, indirect)**
- Types: Direct, indirect
- Other considerations: Non-targeted use, changing hands, one person multiple roles

**Values**
- Explicit vs collateral
- Value tensions
- Different perceptions of same value

**Pervasiveness**
- How widespread?
- For what? (personal, critical infrastructure)
- Connected w/ what?
- Cultural and political implications?
- Crossing national boundaries?

**Time**
- Support duration
- Obsolescence
- Reappropriation
Pervasiveness

- How widespread is use?
- For what? (personal, recreation, critical infrastructure)
- Crossing national boundaries (different rules, customs, infrastructure)?
- Connected w/ what?
- Cultural and political implications?
# Dimensions of Context

<table>
<thead>
<tr>
<th>Stakeholders</th>
<th>Pervasiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Types: Direct, indirect</td>
<td>How widespread?</td>
</tr>
<tr>
<td>Other considerations: Non-targeted use, changing hands, one person multiple roles</td>
<td>For what? (personal, critical infrastructure)</td>
</tr>
<tr>
<td></td>
<td>Connected w/ what?</td>
</tr>
<tr>
<td></td>
<td>Cultural and political implications?</td>
</tr>
<tr>
<td></td>
<td>Crossing national boundaries?</td>
</tr>
</tbody>
</table>

## Values
- Explicit vs collateral
- Value tensions
- Different perceptions of same value

## Time
- Support duration
- Obsolescence
- Reappropriation
Values

- Explicit values: values designers intended to design for
- Collateral values: side effects of design decisions
- Value tensions: one value in a technology challenges another value
- Perceptions of a value: stakeholders have different perceptions of definition of a specific value
  - e.g. privacy as control over own information vs being left alone
Bundling: A growing value tension
Bundling: A growing value tension

- Printers preventing use of cartridges from other manufacturers
- **HP customer support**: “purpose of dynamic security feature is to protect HP’s innovations and intellectual property”
- included in security updates
- **explicit values**: security
- **collateral values**: profit, quality,
- **value tension**: (lack of) sustainability
- **perceptions of values**: network security vs security of intellectual property
### Dimensions of Context

<table>
<thead>
<tr>
<th>Stakeholders</th>
<th>Pervasiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Types: Direct, indirect</td>
<td>How widespread? For what? (personal, critical infrastructure)</td>
</tr>
<tr>
<td>Other considerations: Non-targeted use, changing hands, one person multiple</td>
<td>Connected w/ what? Cultural and political implications? Crossing national</td>
</tr>
<tr>
<td>roles</td>
<td>boundaries?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Values</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit: intentionally designed for (e.g. privacy, trust, adaptability, performance)?</td>
<td>Support duration Obsolescence Reappropriation</td>
</tr>
</tbody>
</table>
Dimensions of Context

**Stakeholders**
- Types: Direct, indirect
- Other considerations: Non-targeted use, changing hands, one person multiple roles

**Values**
- Explicit vs collateral
- Value tensions
- Different perceptions of same value

**Pervasiveness**
- How widespread?
- For what? (personal, critical infrastructure)
- Connected w/ what?
- Cultural and political implications?
- Crossing national boundaries?

**Time**
- Support duration
- Obsolescence
- Reappropriation
Time

- Support duration (long-term support)
- Obsolescence: manage end of support?
- Reappropriation: how OS reappropriated in novel way?
- Choice not to use (or stop using)?
FreeRTOS communicating trust

“de facto standard for microcontrollers and small microprocessors”

Why FreeRTOS?

Trusted kernel
With proven robustness, tiny footprint, and wide device support, the FreeRTOS kernel is trusted by world-leading companies as the de facto standard for microcontrollers and small microprocessors.

Accelerate time to market
With detailed pre-configured demos and Internet of Things (IoT) reference integrations, there is no need to determine how to setup a project. Quickly download, compile, and get to market faster.

Broad ecosystem support
Our partner ecosystem provides a breadth of options including community contributions, professional support, as well as integrated IDE and productivity tools.

Predictability of long term support
FreeRTOS offers feature stability with long term support (LTS) releases. FreeRTOS LTS libraries come with security updates and critical bug fixes for two years. Maintained by AWS for the benefit of the FreeRTOS community.
The original smartwatch... lives?
## Kicking a Pebble (OS) down the road...

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>Raised $10.3 million on Kickstarter</td>
</tr>
<tr>
<td>2015</td>
<td>Pebble raises $20.3 mil from 75k backers on Kickstarter</td>
</tr>
<tr>
<td>2016</td>
<td>Pebble company shut down, IP sold to Fitbit</td>
</tr>
<tr>
<td>2016</td>
<td>Rebble community founded</td>
</tr>
<tr>
<td>2018</td>
<td>Official Pebble support ended</td>
</tr>
<tr>
<td>2017</td>
<td>Rebble Alliance founded, creates RebbleOS</td>
</tr>
</tbody>
</table>

Pebble OS (proprietary, built upon FreeRTOS)

Pebble OS is a proprietary operating system built upon FreeRTOS, which is described as a "de facto standard for microcontrollers and small microprocessors".

**Pebble OS**

- **Pebble OS**: A proprietary operating system built upon FreeRTOS.
  - "De facto standard for microcontrollers and small microprocessors".
Pebble is dead! Long live Pebble!

- 2021: Fitbit acquired by Google
- 2021: Pebble app removed from iOS App Store
- 2022: New Pebble Android app (for 64 bit only Android OS)
  - signed w/ official Pebble keys

Pebble, the OG smartwatch that may never die, updated to work with Pixel 7

Deep inside Google, a signed update makes the smartwatch 64-bit-Android ready.

KEVIN PURDY - 10/28/2022, 9:27 AM
Dimensions of Context

**Stakeholders**
Types: Direct, indirect
Other considerations: Non-targeted use, changing hands, one person multiple roles

**Pervasiveness**
How widespread?
For what? (personal, critical infrastructure)
Connected w/ what?
Cultural and political implications?
Crossing national boundaries?

**Values**
Explicit vs collateral
Value tensions
Different perceptions of same value

**Time**
Support duration
Obsolescence
Reappropriation
1. Trust amongst tech **users, app developers, and system programmers** is intertwined.
2. Trust is about **extending agency** (“unquestioning attitude”).
3. Trust emerges through **assumption, inference, substitution**.
4. Can **design ways to substitute** some need to trust.

**Stakeholders**
Direct & indirect, Non-targeted use, changing hands, one person multiple roles

**Values**
Explicit vs collateral, value tensions, Different perceptions of same value

**Pervasiveness:** How widespread? For what?
Connected w/ what? Cultural, political, national boundaries

**Time**
Support duration, obsolescence, reappropriation
Turing Award Lecture

Reflections on Trusting Trust

To what extent should one trust a statement that a program is free of Trojan horses? Perhaps it is more important to trust the people who wrote the software.

Ken Thompson

Introduction

I thank the ACM for this award. I can’t help but feel that I am receiving this honor for timing and serendipity, programs. I would like to present to you the cutest program I ever wrote. I will do this in three stages and try to bring it together at the end.

Trust and OS in Context

1. Trust amongst tech **users, app developers, and system programmers** is intertwined
2. Trust is about **extending agency** ("unquestioning attitude")
3. Trust emerges through **assumption, inference, substitution**
4. Can **design ways to substitute** some need to trust

**Stakeholders**
Direct & indirect, Non-targeted use, changing hands, one person multiple roles

**Values**
Explicit vs collateral, value tensions, Different perceptions of same value

**Pervasiveness:** How widespread? For what? Connected w/ what? Cultural, political, national boundaries

**Time**
Support duration, obsolescence, reappropriation

Benjamin Xie, Ph.D.  |  Embedded Ethics Fellow  |  benjixie@stanford.edu  |  benji.phd