Executable Texts

Fourth International Conference on Technology, Knowledge, & Society
Boston, Massachusetts

stuart mawler
smawler@gmail.com
Virginia Tech, Falls Church, Virginia
2008/01/18
Changing Perspectives on Software

- **Usual view**
  - Software running end product
    - e.g., Microsoft Word or PowerPoint

- **Alternative view**
  - Software as document, manuscript, corpus, or text
  - Consumed among communities of programmers
  - Continual historical archive, written and re-written
    - Programmer acts like “Talmudic” scholar

- **Method**
  - Comparing comments by two sub-communities
    - Linux Open Source Programmers working on the operating system kernel in C
    - Corporate programmers working on a core business application in COBOL
Some Patterns Emerge

- "Real" constraints
  - The artifacts are impacted by practical considerations outside of the programming task

- Social roles of executable texts
  - What the common laborer is doing, rather than those in charge
    - The lives and intentions of actual people are encapsulated in the comments
      - Even when created for "capitalist" ends, the personalities of the programmers show through

- "Programming archaeology" / Micro-histories
  - A single program is a form of virtual archaeology, peeling back the layers of modifications by various people for various projects
Linguistic Requirements

- How to define a comment...
  - in COBOL (corporate program)
    
    ```cobol
    00892 *----SET ADDRESS OF MESSAGE MAIN HEADER.                          PRGMNBR1
    00893 * PRGMNBR1
    ```

  - in C++ (online programming guide)
    
    ```cpp
    /******************************************************************/
    //Development By : Jigar Mehta
    //Date : [ & now() & ]
    /******************************************************************/
    ```

  - in C (Linux kernel)
    
    ```c
    /* Arch-specific enabling code. */
    ```
Technical Environment

- Different visual constraints
Cultural Constraints and Norms

- Accepted practices
  - Normative purpose for the comment
    - Power structure driving the norms
  - Content
  - Visual Impact

Purpose:
Identify “who” & “when”

Power:
Developed by a manager

Content:
Nothing about the code

Visual:
Easy to see when scanning

أعضاء تأسيسية ونظام
- ممارسة الموافقة
  - غرض تنظيمي للإشارة
    - هيكل القوة تحرك القيم
  - محتوى
  - مظهر تأثير

Purpose:
Identify “who” & “when”

Power:
Developed by a manager

Content:
Nothing about the code

Visual:
Easy to see when scanning
/* Comments have both 
* Form and Function, 
* which exist along 
* a continuum from 
* Normative to 
* Identity-oriented. 
*/

/* Each comment can be plotted 
* against these two axes. 
*/
"Good" Comments

/* Comments have both Form and Function, which exist along a continuum from Normative to Identity-oriented. */

/* Each comment can be plotted against these two axes. */
/* Comments have both Form and Function, which exist along a continuum from Normative to Identity-oriented. */

/* Each comment can be plotted against these two axes. */
Good Comments – One Example

Corporate Sample

- Written in COBOL
- Tells what is happening
- What has been changed
- How it was changed
- Does not tell “who” or “when”
- Does say “why”, subtly with “PSR3”
  - Shows much context, if you know the context...

- Just one of many forms a “good” comment may take
The Texts Speak for Themselves...

- Letting individual personality come through

/*
 * Select whether the frequency is to be controlled
 * and in which mode (PLL or FLL). Clamp to the operating
 * range. Ugly multiply/divide should be replaced someday.
 */ (time.c)

- Allowing comments to act as a “dialog”

if (IS_ERR(p)) { /* Should never happen since we send PATH_MAX */
    /* FIXME: can we save some information here? */
    audit_log_format(ab, "<too long>”);
} else
    audit_log_untrustedstring(ab, p);
kfree(path);
} (acct.c)
The Texts Speak for Themselves...

- Judging the programming history around one’s edit
  06073 * THAT CONCLUDES THE STRUCTURED COBOL PORTION OF THIS PGM...
  06074 * RETURN TO SPAGHETTI CODE!
  06075 GO TO 3159-CONTINUE-SPAGHETTI.

- Individualism in a limited medium
  00404 *RTR 8/22/94 /
  00405 05 WS-TRAN-CODE PIC X(02).
  00406 *RTR 8/22/94 \/

- Taking shots at other programmers
  02266 * 06/17/96 name 06/26/96 FIXED FOR PRT#960521.05.
  [... 5 lines removed ...]
  02272 *
  02273 *
  02274 *
  02275 *
  02276 *
  02277 *
  02278 *
  02279 *
  02280 *

  ALSO REMOVED THE CODE THAT APPEARED TO BE A RESULT OF A PROGRAMMER GUESSING HOW COBOL WORKED. THESE GUESSES CAUSED WARNINGS WHEN COMPILED, ADDED UNECESSARY OVERHEAD AND COMPLEXITY TO THE PROGRAM, AND OBSCURED THE LOGIC OF THE PGM.
“We”: Elevating the Discipline

* The “academic” lecture

```
/*
 * If we're in an interrupt or softirq, we're done
 * (this also catches softirq-disabled code). We will
 * actually run the softirq once we return from
 * the irq or softirq.
 *
 * Otherwise we wake up ksoftirqd to make sure we
 * schedule the softirq soon.
 */ (softirq.c)
```
"We": Enforcing Boundaries

- **Boundaries with users**

  /* We only trust the superuser with rebooting the system. */
  if (!capable(CAP_SYS_BOOT))
    return -EPERM;
  
  /* For safety, we require “magic” arguments. */
  if (magic1 != LINUX_REBOOT_MAGIC1 ||
      (magic2 != LINUX_REBOOT_MAGIC2 &&
       magic2 != LINUX_REBOOT_MAGIC2A &&
       magic2 != LINUX_REBOOT_MAGIC2B &&
       magic2 != LINUX_REBOOT_MAGIC2C))
    return -EINVAL;

- **With outsiders**

  /*
  * setuid() is implemented like SysV with SAVED_IDS
  * 
  * Note that SAVED_ID's is deficient in that a setuid root program
  * like sendmail, for example, cannot set its uid to be a normal
  * user and then switch back, because if you're root, setuid() sets
  * the saved uid too. If you don't like this, blame the bright people
  * in the POSIX committee and/or USG. Note that the BSD-style setreuid()
  * will allow a root program to temporarily drop privileges and be able to
  * regain them by swapping the real and effective uid.
  */
  */(sys.c)
"We": Linking Programmer & System

- First: Anthropomorphize the System

```c
/* Some compilers disobey section attribute on statics when not initialized -- RR */ (softirq.c)
```

- Then becoming part of the system is "natural"

```c
/*
 * We're trying to get all the cpus to the average_load, so we don't want to push ourselves above the average load, nor do we wish to reduce the max loaded cpu below the average load, as either of these actions would just result in more rebalancing later, and ping-pong tasks around. Thus we look for the minimum possible imbalance.
 * Negative imbalances (*we* are more loaded than anyone else) will be counted as no imbalance for these purposes -- we can't fix that by pulling tasks to us. Be careful of negative numbers as they'll appear as very large values with unsigned longs.
 */ (sched.c)
```
## Samples Compared

<table>
<thead>
<tr>
<th></th>
<th>Linux kernel:</th>
<th>Corporate sample:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Files</td>
<td>52</td>
<td>7</td>
</tr>
<tr>
<td>Total Lines</td>
<td>41,505</td>
<td>28,304</td>
</tr>
<tr>
<td>Lines with Comments</td>
<td>5711</td>
<td>6294</td>
</tr>
<tr>
<td>Percentage of Comments to Total Lines</td>
<td>13.75%</td>
<td>22.24%</td>
</tr>
<tr>
<td>Lines with “we” Construction</td>
<td>738</td>
<td>36</td>
</tr>
<tr>
<td>Percentage of Total Lines with “we”</td>
<td>1.78%</td>
<td>0.13%</td>
</tr>
<tr>
<td>Percentage of Comment Lines with “we”</td>
<td>12.92%</td>
<td>0.57%</td>
</tr>
</tbody>
</table>
Identities and Knowledge

- Comments are both normative & identity-oriented
  - Structure (form & function)

<table>
<thead>
<tr>
<th></th>
<th>Linux kernel</th>
<th>Corporate sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programmer Community</td>
<td>Collegial; Cooperative;</td>
<td>Hierarchical; Judgmental</td>
</tr>
<tr>
<td></td>
<td>Collective</td>
<td></td>
</tr>
<tr>
<td>Association with the Machine</td>
<td>High</td>
<td>“just a job”</td>
</tr>
<tr>
<td>Sense of Self</td>
<td>Closely tied to Program</td>
<td>“just a job”</td>
</tr>
</tbody>
</table>

- Commenting is a critical element of
  - Group / Community identity
  - Personal identity (it is “performative”)
  - Historical continuity (both “project” and “community”)

"just a job"
Making it Relevant

- Pragmatic / business implications
  - “There is only one accurate form of system documentation: the source code” – anonymous developer
    - The conventional wisdom may be applicable to organizational factors and the source code (including comments)
      - Window into group dynamics
      - Potentially a chart of organizational health

- Research
  - Valuable and largely untapped research archive
  - Apply the above conventional wisdom to a host of factors
    - Add more time dimensions
      - Program and comment change over time
      - Changes in style, tone, and cultural references
    - Add more samples
A Note of Caution!

- How to Write Unmaintainable Code
  - Counter-normative
  - Shows how comments “should” be written

---

“Any fool can tell the truth, but it requires a man of some sense to know how to lie well.”
~ Samuel Butler (1835 - 1902)

“Incorrect documentation is often worse than no documentation.”
~ Bertrand Meyer
A Contrary View

- Does the “We” Construction really indicate collegiality?
- The Commentator – a satirical faux comment generator

```
Personality Controls

Language
- English

Options
- Profanity
- Religious references
- Drug references
- I33t speak
- Use TODO in empty blocks

FUD Factor
EFF Microsoft
Humor Level
Dykstra Ballmer
Verbosity
RTFM IBM
Self-importance
L. Wall M. Fluery
Relevance
SCO Chicks
Bitterness
Green DeathMarch
```