Embedded Software Licensing?

Prof. Philip Koopman
Carnegie Mellon University
koopman@cmu.edu
October 26, 2000
Personal Background

- **Experience:**
  - U.S. Navy computer system integration
  - Embedded CPU designer (Harris Corp.)
  - Embedded commercial applications R&D (United Technologies)
  - Next-generation cell phone services (Gravitate Inc.)
  - Research & teaching in embedded systems at Carnegie Mellon

- **Ph.D. in Computer Engineering**
  - Books, technical papers, *etc.*
  - 20 U.S. patents
  - Embedded products in current volume production
Embedded software licensing is going to be a mess

- Current attempts to say “software is different” may lead to undermining consumer protection beyond desktop computing

Fundamental problems:

- The concept of a purely “embedded” computer is obsolete
- The concept of saying “software is different” is unworkable
- Consumer choice in license acceptance is endangered
The Way The World Used To Be

- **Embedded systems were anything not in a computer equipment space (a “machine room”)**
  - Custom software with a single purpose, often mission-critical
  - Computers added to products to provide enhanced functionality
  - Products were expected to work regardless of whether they had software or not

- **“General purpose” computers were in office buildings**
  - Used a general purpose operating system (Unix, Windows)
  - Increasingly, not expected to really work all the time
    - Notion of “good enough” to reduce time to market
    - Critical applications used special techniques, not off-the-shelf software
The Way The World Is Becoming

- **Embedded systems are becoming “computers”**
  - Cell phones with built-in Web browsers
  - Car computers that phone for help when an airbag deploys
  - Thermostat that sends e-mail and serves web pages
  - The “internet microwave oven” (yes, this is real)
  - Windows CE – for embedded, but also for handheld computers

- **“Computers” are becoming embedded**
  - Home PC to control household appliances
  - “Auto-PC” – a “real computer” permanently installed in a car
  - “Embedded Windows NT” (slimmed-down Windows NT)
  - PCs used for embedded applications
UCITA Includes Embedded Computers

- **Wording of UCITA fails to exclude embedded computers**
  - The section that seems intended to exclude them won’t stand up to technical scrutiny
  - Even if it were to stand up, it could easily be worked around

- **UCITA official comments don’t exclude them either**
  - Examples given don’t match actual technology facts
  - Exclusion arguments don’t trace back to UCITA wording

- **Proposed UCC Article 2 wording doesn’t do it either**
  - Proposed UCC is slightly stronger in attempting exclusion
  - But doesn’t deal with the reality of convergence of embedded and mainstream computing
These Are Both Web Servers

- Which one is a “computer”?
  - They’re both “computers”, even if one is in a thermostat.

http://www-ccs.cs.umass.edu/~shri/iPic.html/
Conclusion:
The term “embedded” isn’t useful for determining which licensing rules to apply.
“Software” Isn’t Just Spreadsheets

- Operating systems are going everywhere
  - Embedded systems have Unix & Windows operating systems
  - Very soon, essentially every car will have a commercially produced operating system (as opposed to proprietary ones)

- It’s easy to migrate hardware functions into software
  - If we make a product look like a “computer,” does the software no longer have to work?

- How do you know if software is “embedded”?
  - Should it matter if the very same software is running inside a PC or a dishwasher?
Is This “Embedded Software”?

- Single purpose computer:
  Automatic speech translation: English · Croatian

- PC hardware running Windows 95 and off-the-shelf speech software
Conclusion:

Arguing “software is different” will distort engineering tradeoffs in embedded product design.
Licenses For Embedded Systems?

- **Current protection based on patents**
  - In embedded systems, *functionality* is what matters, not “software” vs. “hardware” (in fact, they can be equivalent)
  - Functionality can be patented, and has been for decades
  - Now, software can be patented too

- **Encouraging embedded software licensing is potentially dangerous**
  - Currently, embedded software is not considered “special”
    - This moderates the rate of introducing new features
    - This is one of the few forces acting to moderate the software safety problem (we’re still struggling with how to measure “software safety”)
  - Do you really want embedded software to be as robust as current desktop software?
Would *You* Drive A Car In Which:

“THE SOFTWARE is provided ‘AS IS’ and with all faults. THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY, AND EFFORT (INCLUDING LACK OF NEGLIGENCE) IS WITH YOU.”

*(You will.)*
Company A: (License wording available on the Web)
- Any use constitutes agreement
- No Warranties; “As is” and with all faults and any negligence
- Any user of product of which it is a component must agree
- Reverse engineering prohibited

Company B: (License wording available on the Web)
- Same as above, PLUS
- Leasing or sale of software prohibited; can’t leave country
- “Bugs are likely”
- But, warrants it will work per documentation for 6 months

Company C: flatly refused to provide EULA
Will Consumers Have A Choice?

- **Theory is that consumers can pick appropriate license**
  - Look at licensing terms before purchase (perhaps on web)
  - Marketplace presumably will force reasonable license terms

- **But what if there is no choice?**
  - All operating system vendors seem to have similar approaches
  - Complex products such as cars will have many components
    - If any single OS is in any component of different vehicles, the same EULA applies!
    - It would be no surprise if only one or two operating systems dominate within a few years
  - Even if only non-embedded software “is different”, vendors will have huge incentive to make their products be non-“embedded”
Conclusion:

Current approaches to software licensing will jeopardize consumer protection and choice for embedded systems.
Conclusions

◆ Fundamental problems:
  • “Embedded” computers and “computers” are converging
    – Any potentially useful definition can be discredited or circumvented
  • The concept of saying “software is different” is dangerous
    – Converting complexity into software instead of hardware is easy
  • Consumers will be hurt by licensing embedded software
    – This is already happening; it just hasn’t reached high market penetration yet

◆ Embedded software licensing is going to be a mess
  • UCITA/UCC wording requires significant fixes; may be unfixable
  • Even if “embedded” can be excluded from UCITA, there will be compelling incentive to make everything look like a non-embedded “computer”