Intellectual property

- **protection mechanisms**
  - trade secrets
  - trademarks
  - patents
  - copyrights
  - licenses

- **standards and standardization**

- **open source / free software**

Trade secrets

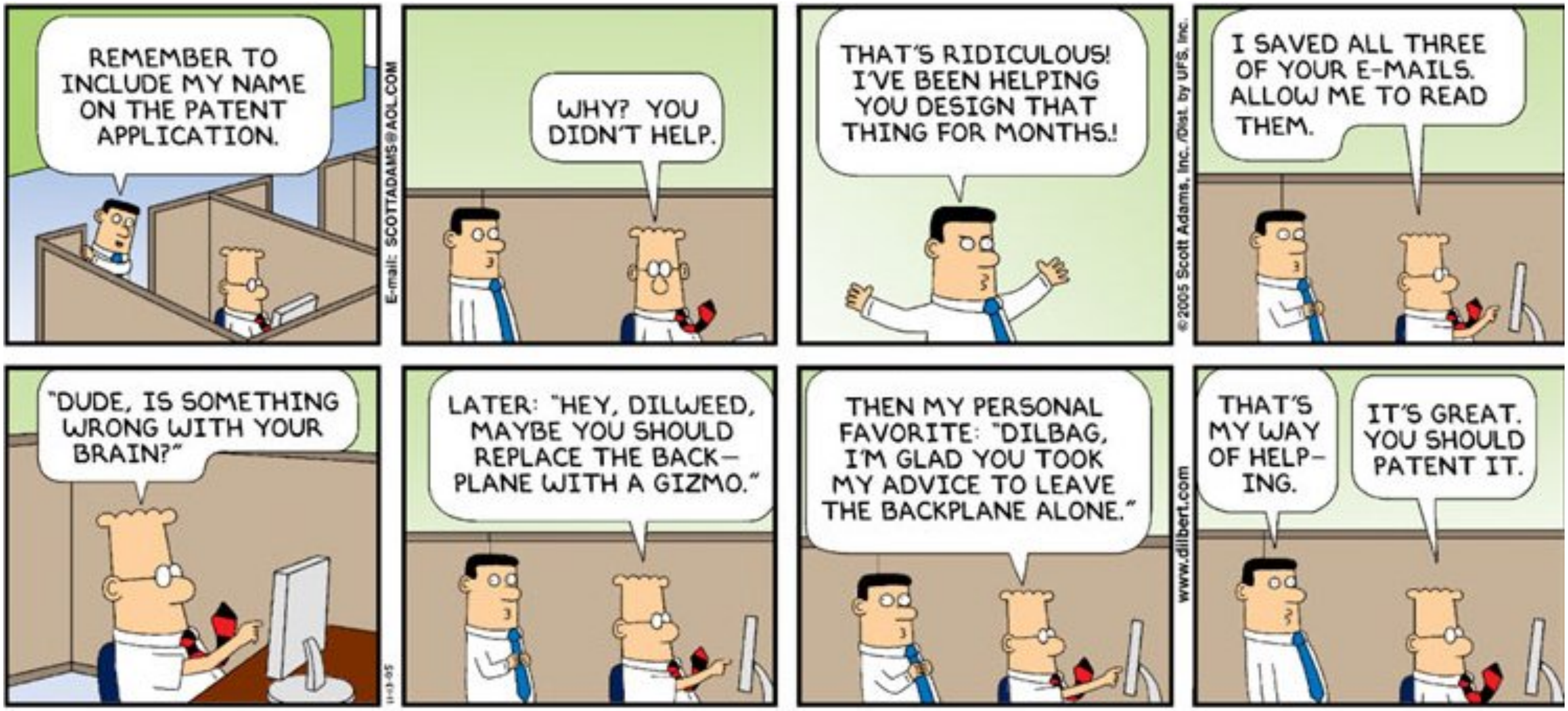
- **information is a secret held by its owner**
- **disclosed only under some kind of agreement**
  - e.g., "non-disclosure agreement" or NDA
- **no recourse if secrecy is lost**
- **often used to argue that information should not be made public**
  - voting machine
  - breathalyzer
  - ...

Patents

- **exclusive right to make, use or sell an invention in US**
- **valid for 20 years after filing**

- **requirements:**
  - statutory subject matter:
    process, machine, article of manufacture, composition of matter
  - novel
  - useful
  - unobvious to person having ordinary skill in the art
    at the time of filing

- **contents:**
  - abstract
  - drawings/diagrams
  - specifications (narrative description, preferred embodiment)
  - **claims**


"REMEMBER TO INCLUDE MY NAME ON THE PATENT APPLICATION.
WHY? YOU DIDN'T HELP.
THAT'S RIDICULOUS! I'VE BEEN HELPING YOU DESIGN THAT THING FOR MONTHS.!
I SAVED ALL THREE OF YOUR E-MAILS. ALLOW ME TO READ THEM.
"DUDE, IS SOMETHING WRONG WITH YOUR BRAIN?"
LATER: "HEY, DILWEED, MAYBE YOU SHOULD REPLACE THE BACK–PLANE WITH A GIZMO."
THEN MY PERSONAL FAVORITE: "DILBAG, I'M GLAD YOU TOOK MY ADVICE TO LEAVE THE BACKPLANE ALONE."
THAT'S MY WAY OF HELP–ING.
IT'S GREAT. YOU SHOULD PATENT IT.
E-mail: SCOTTADAMS@AOL.COM
©2005 Scott Adams, Inc./Dist. by UFS, Inc.
www.dilbert.com

© Scott Adams, Inc./Dist. by UFS, Inc.

Patent issues for software

- **what is patentable? (statutory subject matter)**
  - software itself
  - business methods

    (whether implemented in software or not?)

- **what's novel?**
  - how new and unobvious does something have to be?

- **policy questions**
  - what should be patentable?
  - are patents (and patent trolls) impeding progress?
  - is the 20-year term too long?

Copyright

- **protects expression, not idea**
- **duration used to be 17 years + one renewal**
- **now life + 70 years, or 95 years for commercial works**
  - (the "Mickey Mouse Protection Act", 1998)
- **"fair use" permits limited copying under some circumstances**
  - criticism, comment, scholarship, research, news reporting, teaching
- **uncertain what fair use really is -- case by case decisions**
- **considerations:**
  - purpose and character of the use
  - nature of the copyrighted work
  - amount and substantiality of the portion used
  - effect of the use on potential market or value of the copyrighted work
- **recent copyright laws may prevent some fair uses**
  - can't decrypt to make excerpt for teaching or criticism
  - can't reverse engineer to make copies in different media

Copyright issues in software

- **code**
  - theft in commercial setting
  - plagiarism in academic setting
- **visual appearance, "look and feel", etc., of a program**
- **interfaces vs implementations**
- **reverse engineering?**
  - clean room implementation
- **copyright or patent?**
  - which is appropriate to protect specific piece of software?

Licenses

- **an agreement (e.g., contract) that allows a particular use of some software**
  - that might otherwise be a violation of copyright, patent, etc.

- **are shrinkwrap and clickwrap licenses valid and enforceable?**

- **is licensing replacing purchase?**

- **are warranty and liability disclaimers for software valid?**

Standards and standardization

- **standard: technical specification sufficiently precise that it ensures independent implementation, uniformity, interoperability, ...**
  - physical measurements: length, weight, time, chemical composition, ...
  - mechanical properties: plugs & sockets, CD/DVD dimensions, …
  - electrical properties: voltage, frequency, …
  - software: character sets, programming languages, operating system interfaces, data formats, information exchange protocols, ...
- **standardization: process of establishing a specification**
  - usually involves competing entities, so tradeoffs are needed between mutual benefit and competitive advantage
  - often international (e.g., ISO: International Organization for Standardization)
- **de facto vs de jure standards**
  - de facto: Windows, Office, Flash, PDF, ...
  - de jure: ASCII, Unicode, major programming languages, ...

Open source / free software

- **source code: instructions in a readable programming language**
  - usually has significant commercial value

    e.g., Windows, Office, TurboTax, Photoshop, …
  - usually proprietary, secret, not revealed

    even if compiled version is given away (e.g., iTunes, Internet Explorer)

- **"open source": source code is available, can be use, copied and modified**
  - a reaction to restrictions on proprietary code
  - promoted by Free Software Foundation, other open source projects & groups
- **various kinds of licenses determine what can be done with it**
  - mainly concerned with keeping source code open enough that others can continue to build on it and improve it
  - prevents anyone from taking it private / proprietary
- **a viable threat to proprietary software in important areas**

Free Software Foundation (Richard Stallman, MIT, ~1985)

- **plan to build an operating system and all supporting software**
  - "GNU" -- "GNU's not Unix"
- **started non-profit organization called the Free Software Foundation**
- **wanted source code to be released so that it could not be made proprietary, would remain free forever**
  - "free" as in "free speech", not "free beer"
    - ok to charge for distribution, support, etc.
- **source released under copyright agreement that requires that any subsequent distribution be covered by the same agreement**
- **GNU GPL (General Public License): "copyleft"**
  - full permission to use, copy, modify, distribute modifications
  - copies, derivative works, etc., must have the same terms if distributed
  - copies, etc., must have the same license attached to them
  - NO permission to add further restrictions; explicitly forbidden
- **source code has to be freely available**
  - can't "take it private"

Fundamental Software Ideas

- **algorithm: sequence of precise, unambiguous steps**
  - performs some task and terminates
  - based on defined basic / primitive operations
  - describes a computation independent of implementation details
- **programming language:**
  - grammar, syntax, and semantics for expressing computation
    - notation is important
- **program: algorithms implemented in a programming language**
- **compilers, interpreters: programs that convert from the high level language used by people to a lower level**
  - a compiler is a program that writes a program
  - an interpreter also acts as a computer so the program can be run
- **libraries and components: programs written by others**
  - packaged in a form that can be used in a new program
- **abstraction, layers, interfaces, virtualization**
  - hiding details, pretending to be something else
- **bugs: the need for absolute precision**
  - cover all cases, cope with failures and misuse