CS207 #1, 25 Sep 2009

Gio Wiederhold

http://infolab.stanford.edu/people/gio.html

Gates B12

Syllabus:

The order is flexible

1. Why should software be valued?
2. Open source software. Scope. Theory and reality
3. Principles of valuation. Cost versus value.
4. Market value of software companies.
5. Intellectual capital and property (IP).
6. The role of patents, copyrights, and trade secrets.
7. Life and lag of software innovation.
8. Sales expectations and discounting of future income.
9. Alternate business models.
10. Selling or Licensing SW.
11. Separation of use rights from the property itself.
12. Risks when outsourcing and offshoring development.
13. Effects of using taxhavens to house IP.
14. How to grow a software company: organic or by acquisitions

Topics 2009

For a motivation see Jeff Hawkins: What I wish I'd learned in college
<A HREF="http://ecorner.stanford.edu/authorMaterialInfo.html?mid=2289">
Slides from all lectures:
Why should software be valued? Open source software, theory and reality. Scope.
<A HREF="http://infolab.stanford.edu/pub/gio/2009/CS207-1.pdf">
Intellectual capital and property (IP). Principles of valuation.
<A HREF=" http://infolab.stanford.edu/pub/gio/2009/CS207-2.pdf">
Cost versus value. Market value of software companies. Sales expectations and discounting,.
<A HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-3.pdf>
Alternate business models.
<A HREF=" http://infolab.stanford.edu/pub/gio/2009/CS207-4.pdf">
Life and lag of software innovation
<HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-5.pdf>
The role of patents, copyrights, and trade secrets. Managing IP.
<A HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-6.pdf>
Off shoring (Prof. Amar Gupta)
<A HREF=http://infolab.stanford.edu/pub/gio/2009/Stanford-Nov09.pdf>
Licensing. Separation of use rights from the property itself. Offshoring alternatives. Risks.
<A HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-7.pdf>
Effects of using taxhavens to house IP.
<A HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-8.pdf>
Acquisitions and growth .
<A HREF=http://infolab.stanford.edu/pub/gio/2009/CS207-9.pdf>

Course Info

Meets weekly, Fridays 2:15pm, Gates B12.

Me: Gio Wiederhold, Prof. Emeritus, Gates 436, hours by appointment gio@cs.stanford.edu

For course updates and references see

https://cs.stanford.edu/wiki/cs207/

Grading: 1 unit P/F for attendance

if a class is missed: 1 page report on related topic

Optional: directed reading graded units for a relevant report

about 3 pages, draft by 18 Nov 2009, on-line.

Background

Two aspects to Software Economics

1. Minimizing the cost of building effective SW

   Much literature exists, taught as part of SW engineering

   Factors

   1. Well educated people → **you**
   2. Good languages → expressive and constraining
   3. Good methods → Waterfall, Spiral, Rapid prototyping, Scrum, Extreme programming, Agile processes.

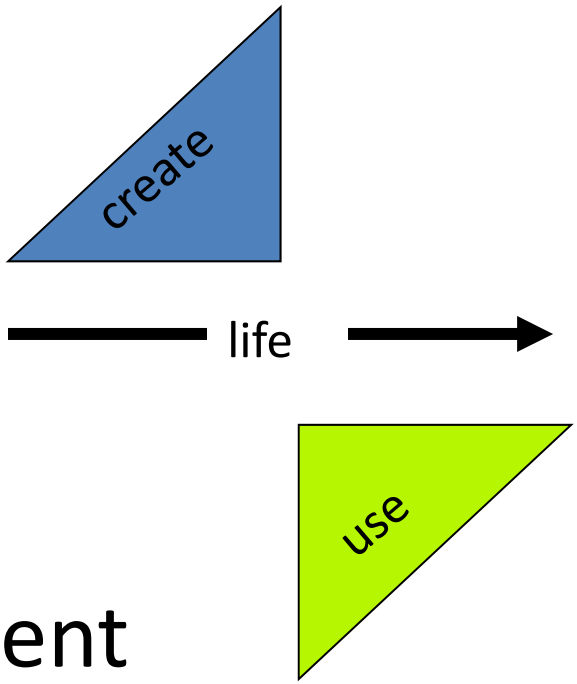
   And when the work is done

2. Maximizing the benefits of the SW *the topic of CS207*

Current State

1. Software producers traditionally care about
   - Cost of writing software
   - Time to complete products
   - Capabilities
2. When the value is a concern
   - Business people
   - Economists
   - Lawyers
   - Promoters

} inconsistent



What is the problem?

Say you create some software and ship it on a CD to a company that sells software.

- Let's assume they get the exclusive right to the SW.
  What should the selling company pay you?
  1. The cost of the CD and mailing it? about **$10.-**?
  2. The amount it cost you to write the SW:
     5 months at $10,000/month = **$50,000.-** ?
  3. Half of their sales that year *(~ 50% is their cost of selling)* :
     50% of 10,000 copies at $49.99 = **$250,000.-** ?
  4. 50% of their $2M lifetime sales = **$1,000,000.-** ?
- How does what you get affect your obligations?

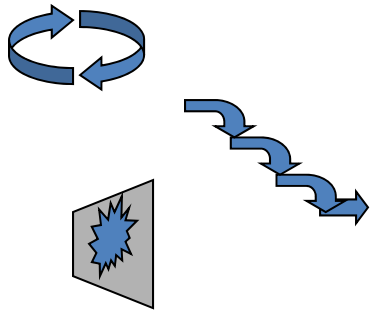
Why is value a Concern

- Making decisions about creative tradeoffs
  - Elegance versus functionality
  - Rapid generation versus maintainability
  - Careful specification versus flexibility
- *Dealing with customers*

  Dijkstra model: *for self-satisfaction*

  Engineering model: *formal process driven*

  Startup model: *see if it sticks to the wall*



- Gain respect: *know what you are doing*

CS versus other professions

- Architects *of buildings*

  Know if they are designing public housing or a castle

    That helps specify the type of furnishing and fixtures: *zinc / nickel*

- Car Designers ~1M/year


/year

  Know if they are designing a people's car or a Siddeley

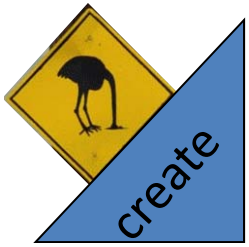
    That helps specify the level of sound insulation and parts' life time

- Software engineers

  Don't consider if the software will be widely used,

    Bugs, when encountered by many customers, are costly

  May spend much time refining software that will be used rarely



Why now

Worrying about economics is a sign of a maturing field

Phases:

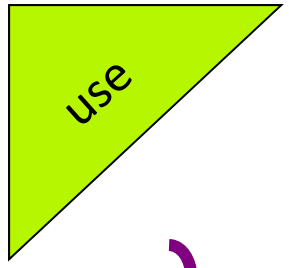
1. Get new stuff to work
2. Getting adequate performance
3. Get it to be sufficiently reliable to be useful
4. Get it into routine production
5. Increase capacity
6. Make it safe
7. Make it affordable

Why me

US Treasury concern:

- Much software is being exported as part of **offshoring** (offshore outsourcing)
- It is typically property – i.e., *protected*
- If it is not valued correctly – i.e., *too low*
  1. Loss of income to the creators *in the USA*
  2. And loss of taxes *to the US* treasury
  3. Excessive profits *kept external to the USA*
  4. Increased motivation for external investment

Value depends on use



When the value is a concern

- Business people
  - Income from sales or businesses improvements
  - Price or license determination
- Economists
  - Effects on national productivity
- Lawyers
  - Settlement of disputes and infringements
- Promoters
  - Motivating investments

inconsistent

1987 Quote

"Some day, on the corporate balance sheet, there will be an entry which reads, 'Information';

for in most cases the information is more valuable than the hardware which processes it."

-- *Grace Murray Hopper* 1906-1992

Rear Adm., US Navy, 1943-1986.

Early Univac programmer, *when they cost > $1,000,000*

ntributor to the development of COBOL

guage and compiler

given away at no cost to Univac purchasers

Open Source software?

Should software should be a free good?

Implicit in that view is that government, universities, and foundations should pay for software development, rather than the users.

1. *Programmers are creative artists, creating beauty and benefits for all of Mankind !*

vs.

2. Software is an industry.

SW revenue is $121B per year in the U.S. alone, well over 1% of the US GDP.

Non-software companies spend yet more for business-specific software.

Over 4.8 million people are employed in IT, earning nearly $333B annually.

- It is unlikely that universal free software is an achievable and even a desirable goal.

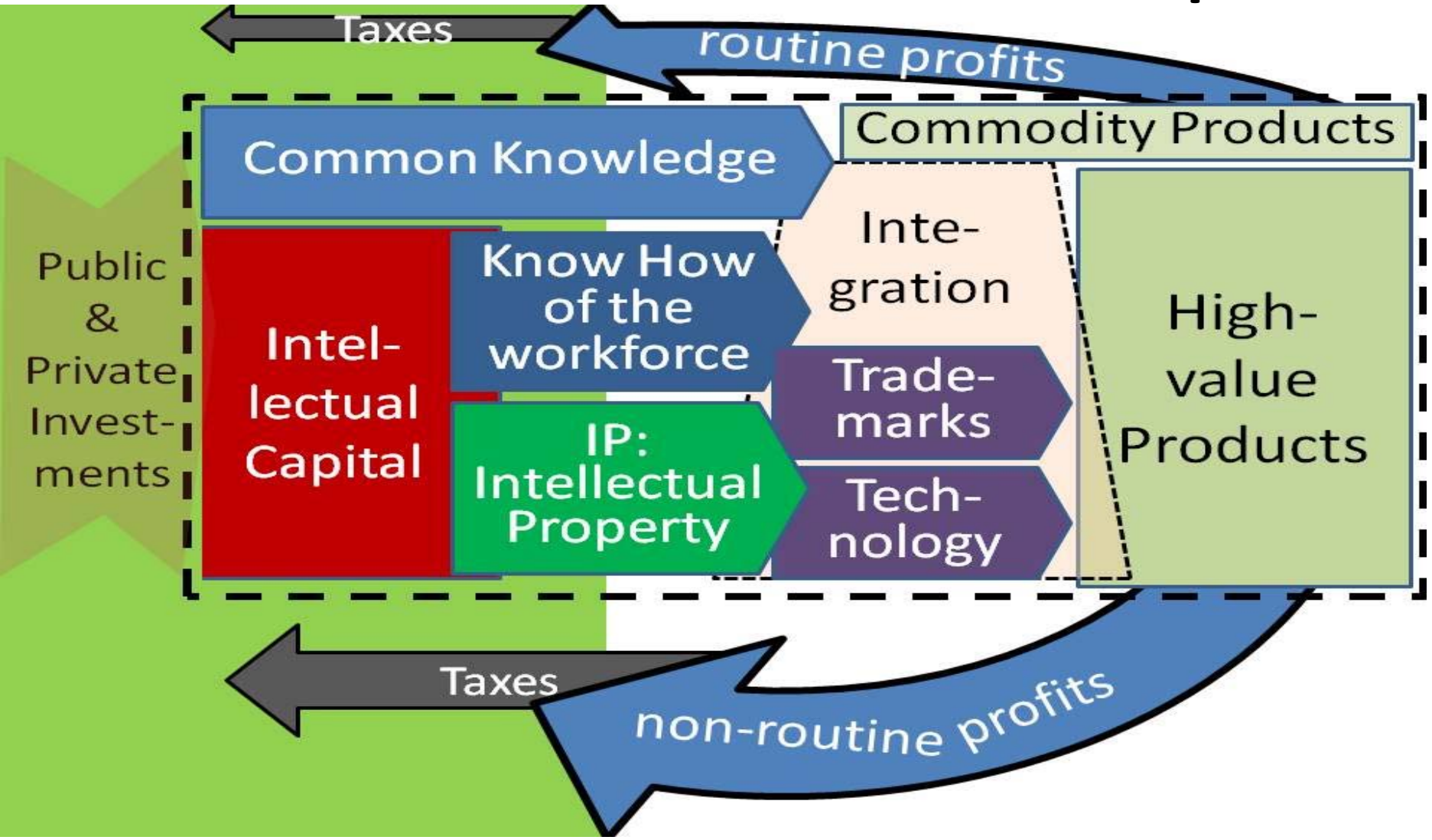
Open Source Practice

- Appropriately, open source initiatives actually focus on software that deserves wide public use and should be freely available to students and innovators, as *editors, compilers, and operating systems*.
- Much open source software is incorporated into Commercial software, that is not made freely available*,*
  - *even if it should be.*

What's left to value?

- Common software that is sold or licensed
- Software that enables Internet Services
- Software that is written inside companies to improve their business
- Software purchased from vendors by companies to improve their business
- Software purchased from vendors by government to improve its operations
  - Military, Social Security, IRS, Healthcare, . . .

Economic Loop



Next Week

- How to value software
  - What is valuable in software
  - Where does the value derive from

- Questions?
  - Email to Gio@cs.stanford.edu