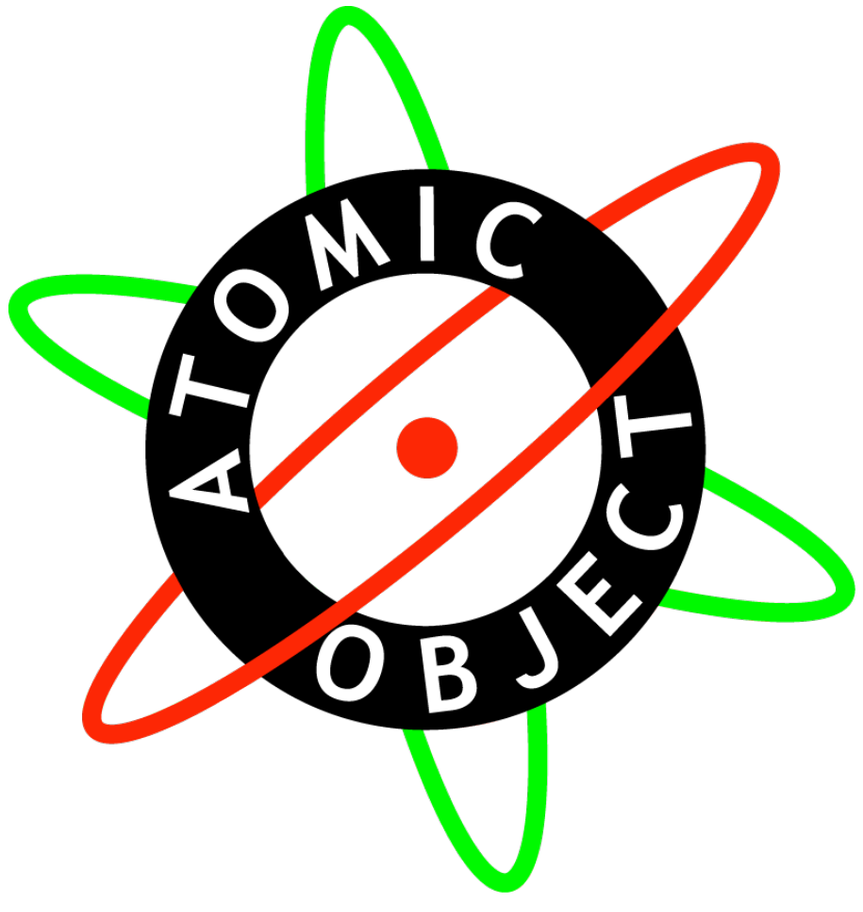
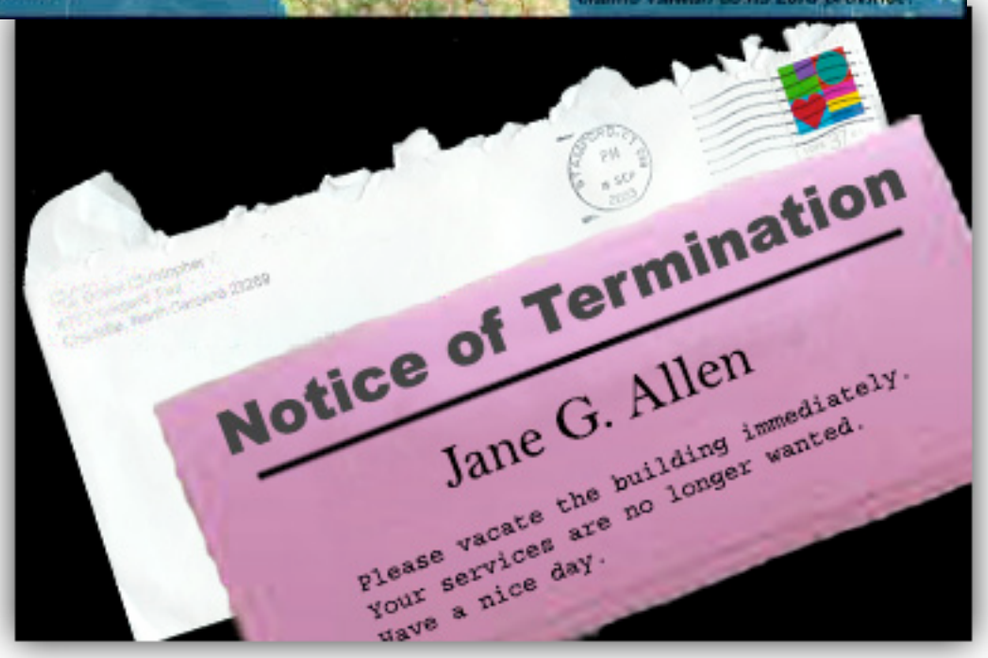
Fostering excellence in IT services through quality and innovation

Carl Erickson
Atomic Object





gloom and doom, unemployment, layoffs, India, China, cost-pressure on CEOs

Vernon Cycle

1. new product development
2. maturation
3. standardization

“Across the great divide”, David Alan Grier, George Washington U, IEEE Computer, July 2006

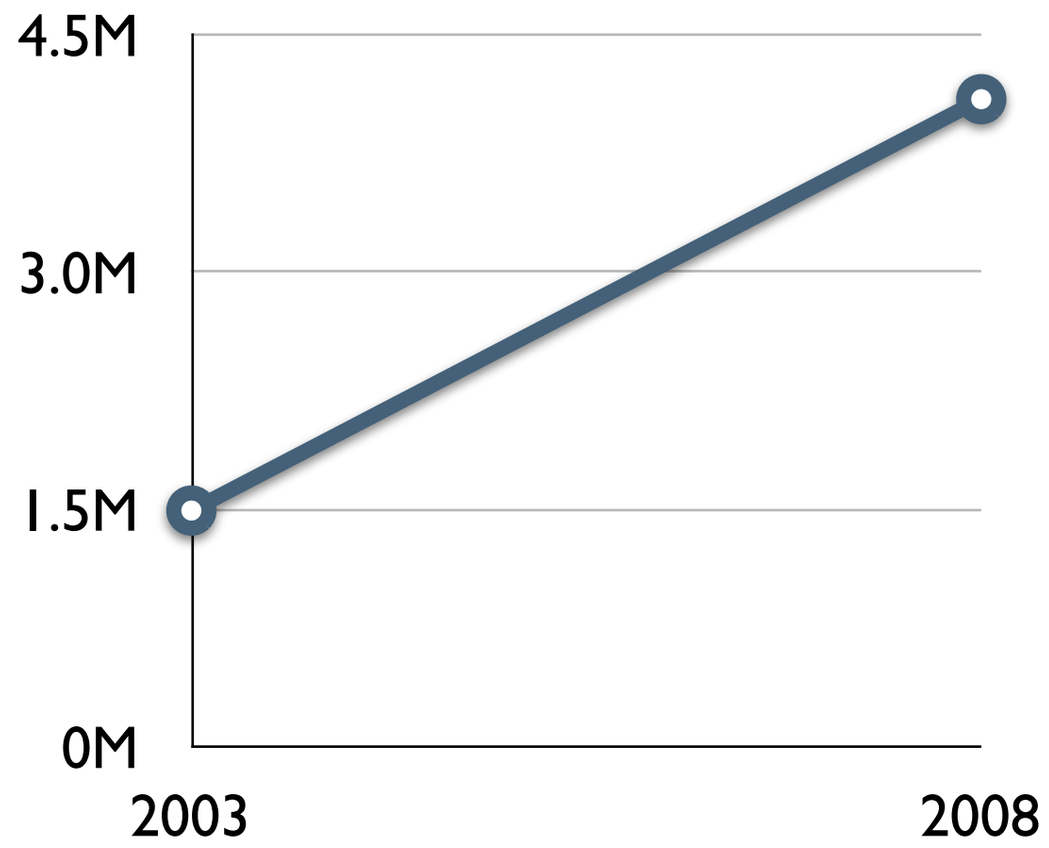
R. Vernon, "International investment and international trade in the product cycle," Quarterly Journal of Economics, 80 (1966) pp 190-207.

1. new product development - market uncertainty, need for effective communication between market and company
   product created and produced in close proximity to the knowledge that creates it
2. maturation - demand and production expands, exports begin
   made in any technically sophisticated country
3. standardization - uncertainty gone, margins lower, cost more important
   can be made in any modern country, hence lowest manufacturing cost wins

Overblown Fear

1. The numbers
2. Nature of our work
3. Types of innovation

Service jobs offshored



Economist, Oct 7, 2006 – global competition for talent, fear of globalization overblown

McKinsey Global Institute study

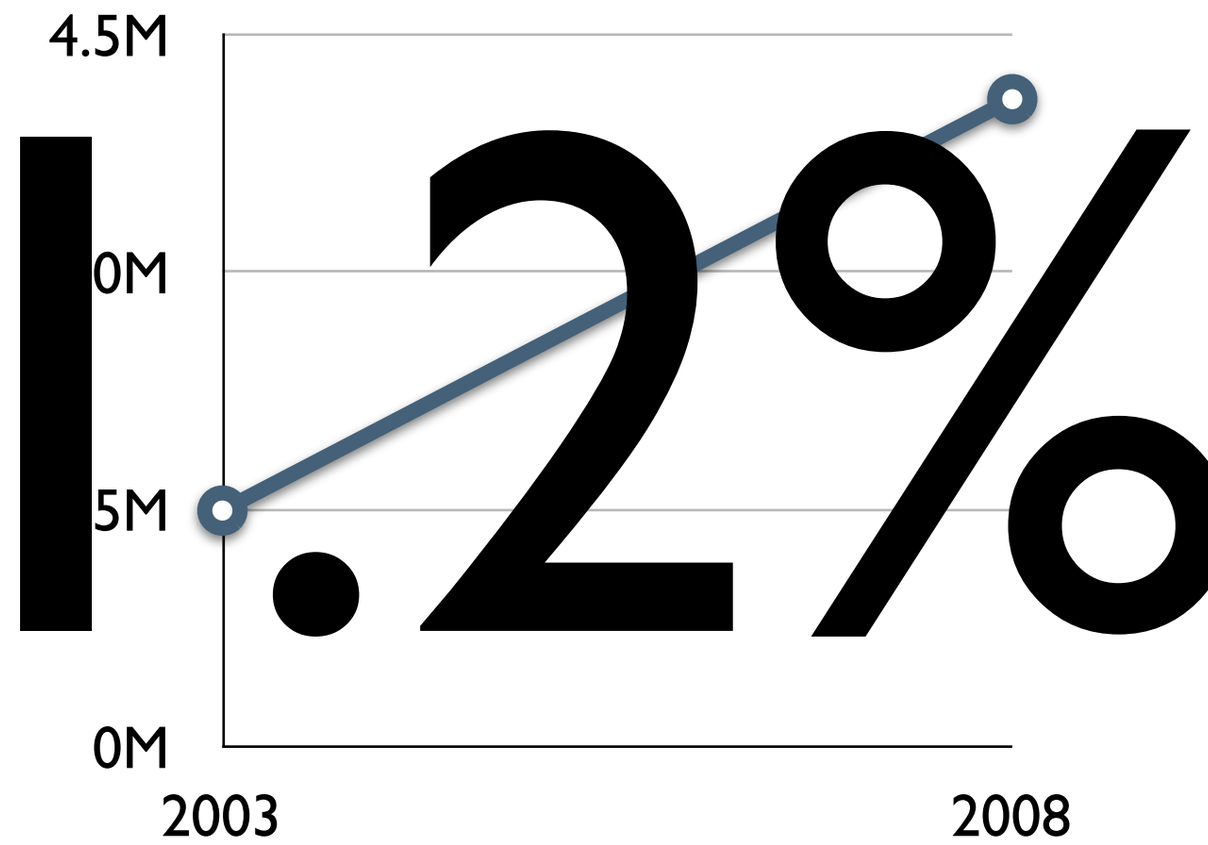
service jobs worldwide which moved offshore:

- from 1.5M in 2003 to 4.1M in 2008

seems scary, but

- that's only 1.2% of the demand for labor in the developed world

Service jobs offshored



Economist, Oct 7, 2006 - global competition for talent, fear of globalization overblown

McKinsey Global Institute study

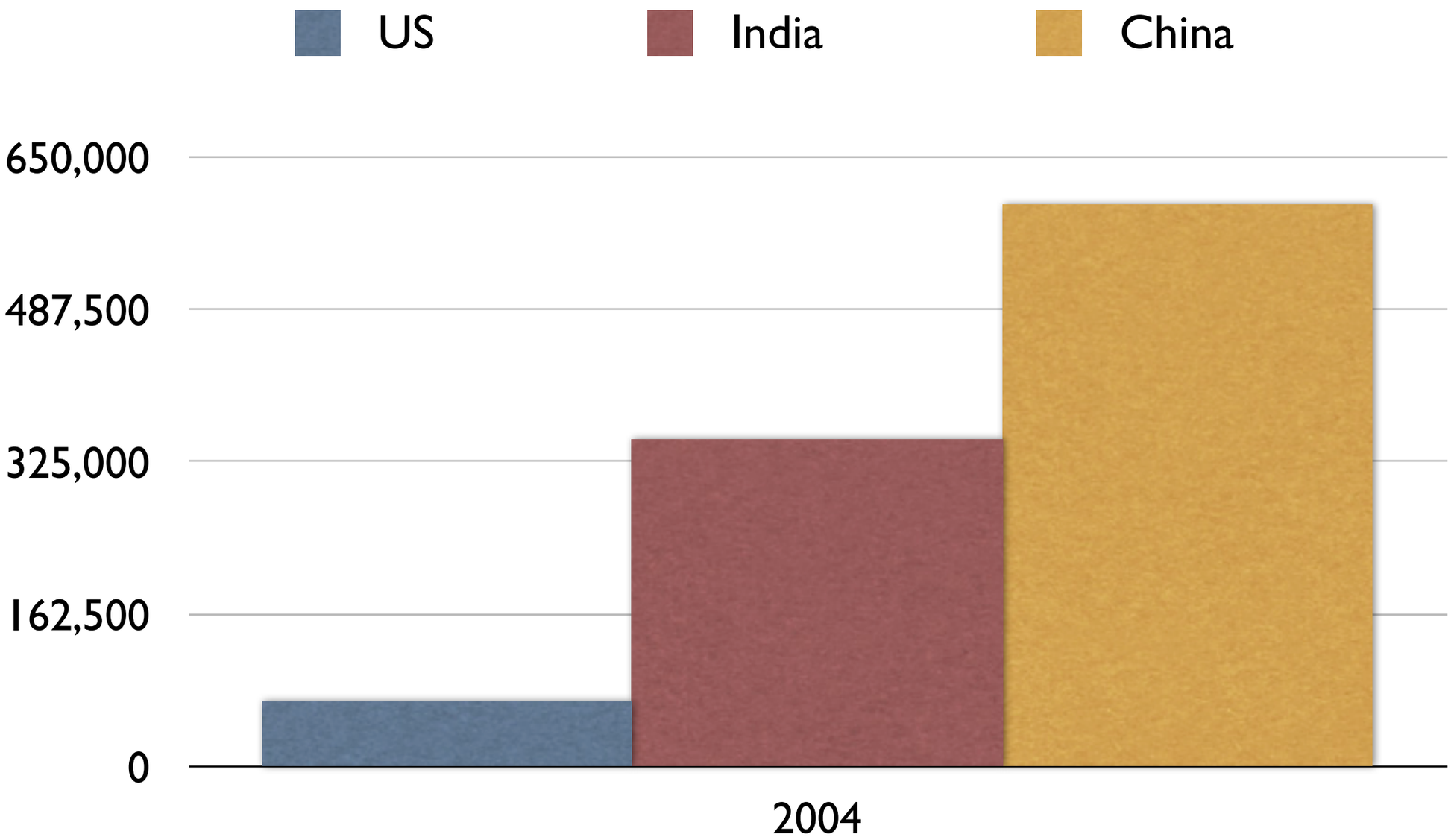
service jobs worldwide which moved offshore:

from 1.5M in 2003 to 4.1M in 2008

seems scary, but

that's only 1.2% of the demand for labor in the developed world

Engineering Graduates

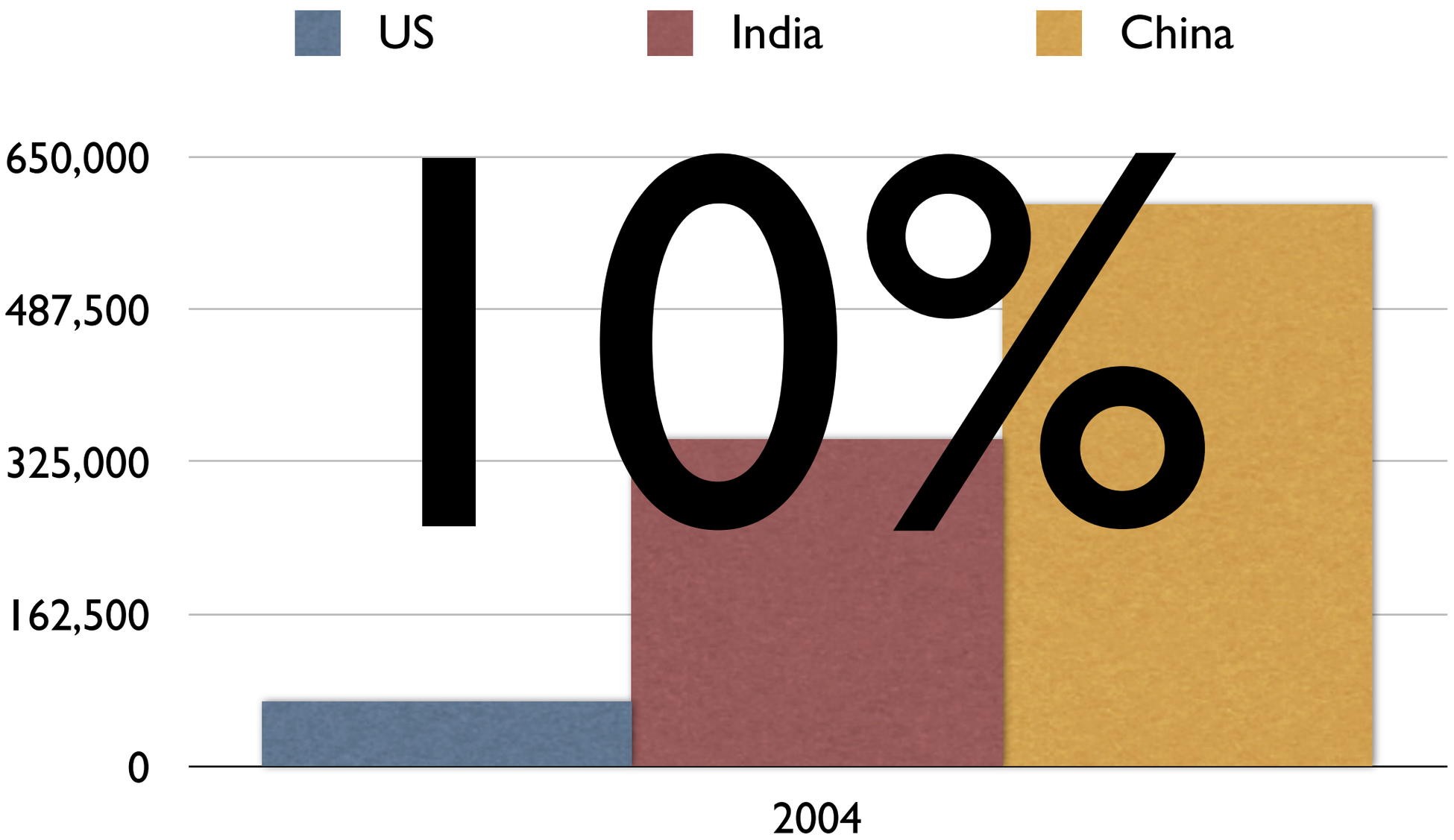


Business Week, December 2005
engineering graduates
China 600k, India 350k, US 70k

But, according to McKinsey,

quality of education, infrastructure, language barriers means only 10% of Chinese engineering graduates are equipped to work for a Western multinational

Engineering Graduates

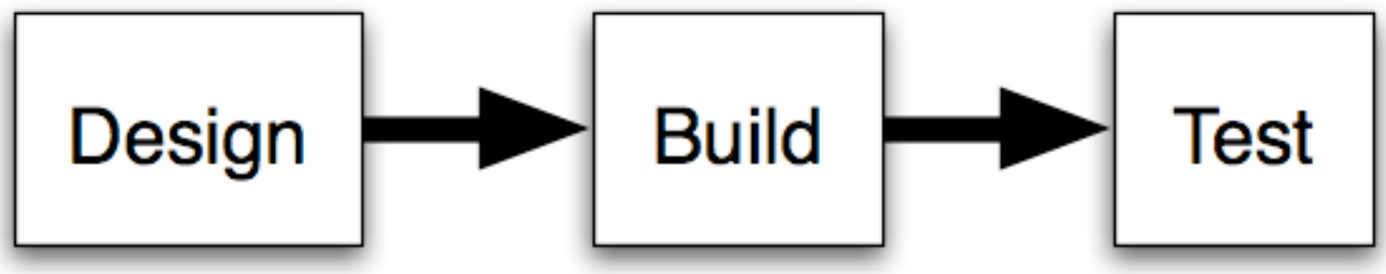


Business Week, December 2005
engineering graduates
China 600k, India 350k, US 70k

But, according to McKinsey,

quality of education, infrastructure, language barriers means only 10% of Chinese engineering graduates are equipped to work for a Western multinational

Software as Widget?

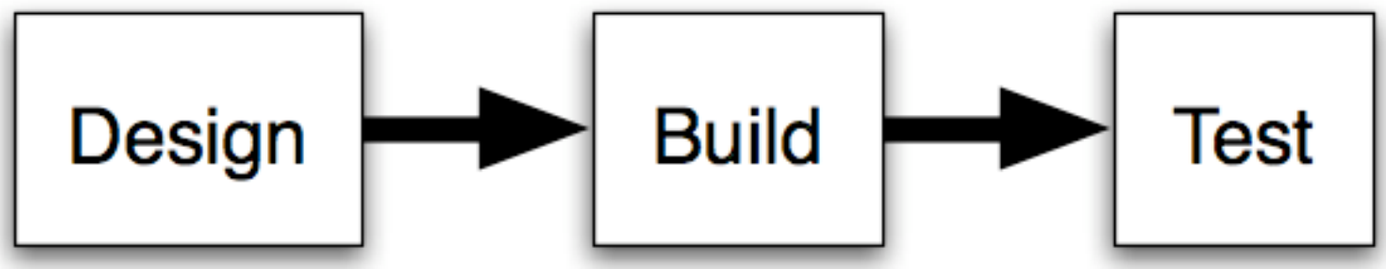


For widgets, design is complex, time-consuming, people-centric, creative
but the cost is amortized over many widgets

For widgets, build/test is also time-consuming, and potentially expensive

For software...

Software as Widget?

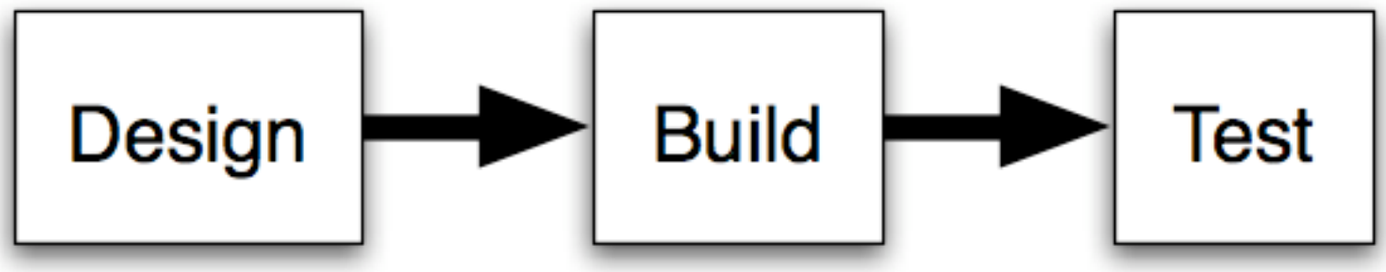


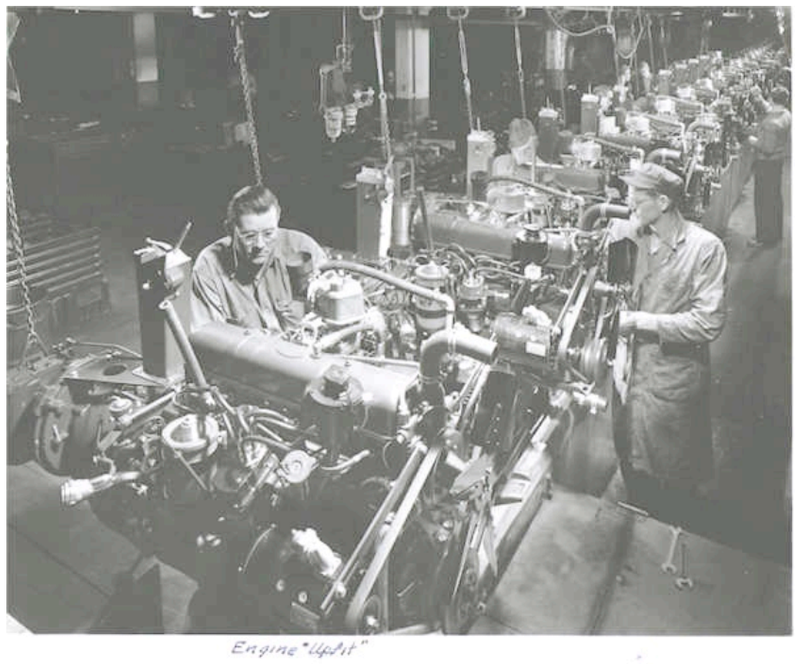
For widgets, design is complex, time-consuming, people-centric, creative
but the cost is amortized over many widgets

For widgets, build/test is also time-consuming, and potentially expensive

For software...

Software as Widget?



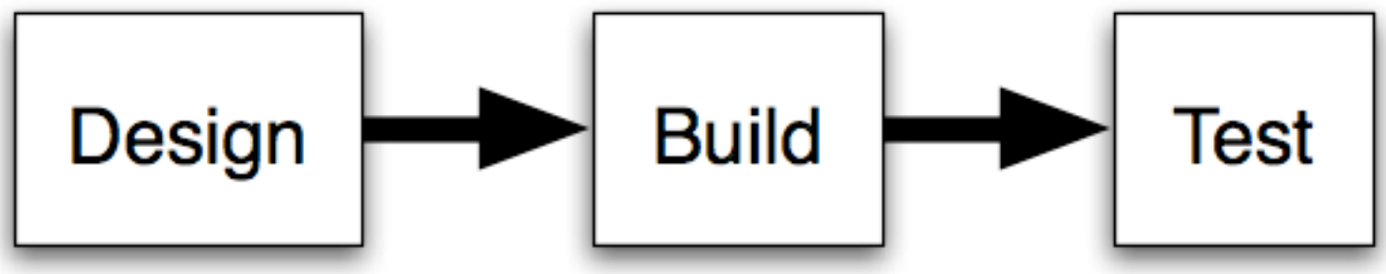


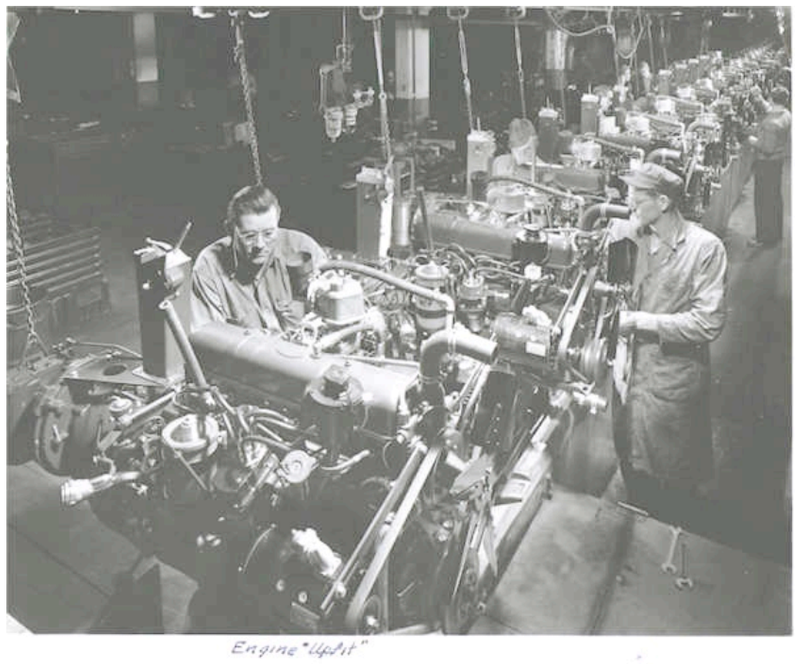
For widgets, design is complex, time-consuming, people-centric, creative
but the cost is amortized over many widgets

For widgets, build/test is also time-consuming, and potentially expensive

For software...

Software as Widget?





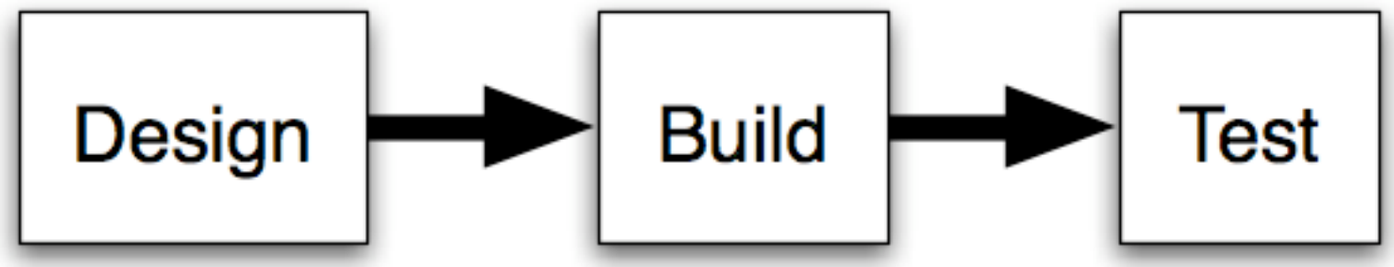
For widgets, design is complex, time-consuming, people-centric, creative
but the cost is amortized over many widgets

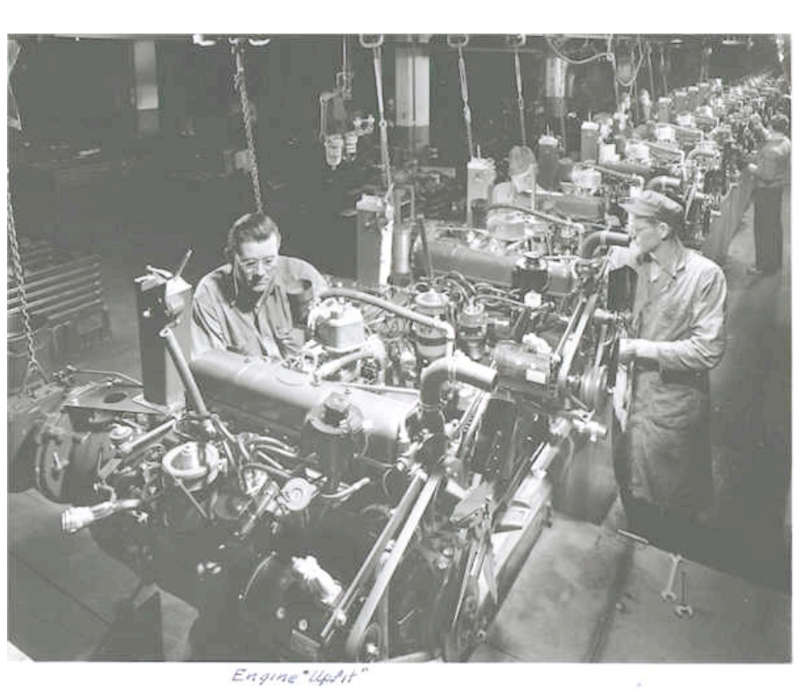
For widgets, build/test is also time-consuming, and potentially expensive

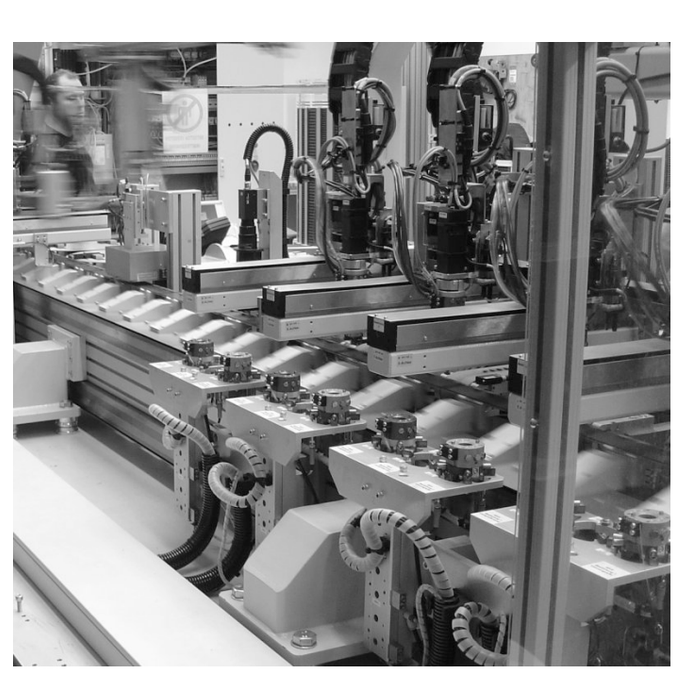
For software...

building is cheap, effortless (compiler, assembler, linker)

Software != Widget







Creating software

but design and test is expensive

design actually encompasses many activities (architecture, specification, prototyping, design, coding, testing, …)

design happens throughout the life of the project

so if the designers and the coders are different people, coders will be changing design

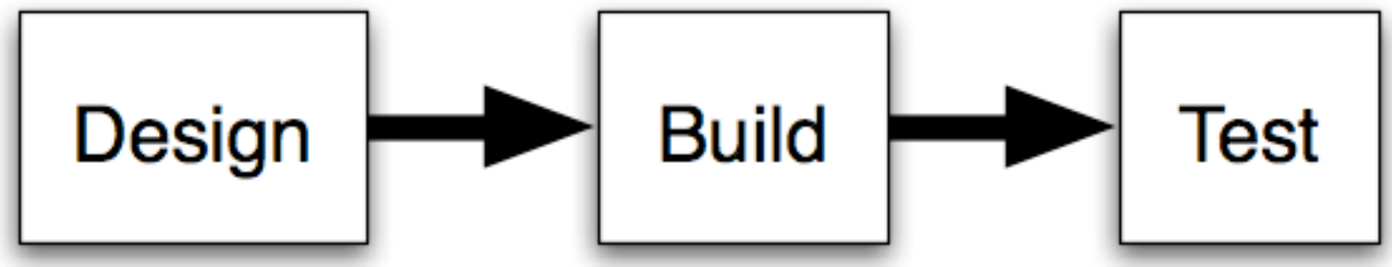
source code expresses the true and ultimate design - it is therefore a design artifact

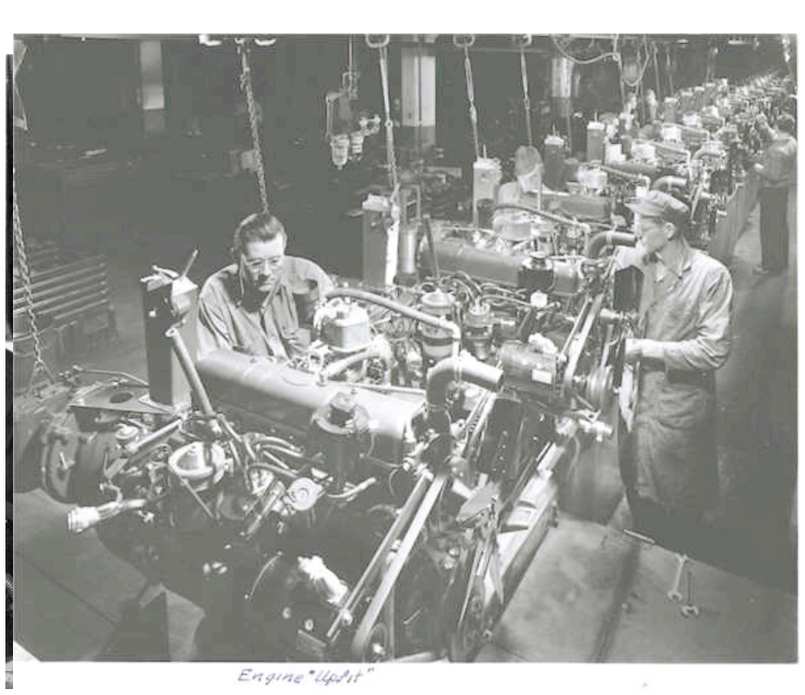
Creating good software requires skills and experience (craftsmanship)
Creating good software predictably over time requires some process (engineering)

Essay by Jack Reeves from The C++ Journal in 1982 sums up these ideas very nicely. See Robert Martin's Agile Software Development book for a copy.

The obvious conclusion is that the Vernon Cycle doesn't apply to custom software development, particularly for new products

Software != Widget





Creating software

but design and test is expensive

design actually encompasses many activities (architecture, specification, prototyping, design, coding, testing, …)

design happens throughout the life of the project

so if the designers and the coders are different people, coders will be changing design

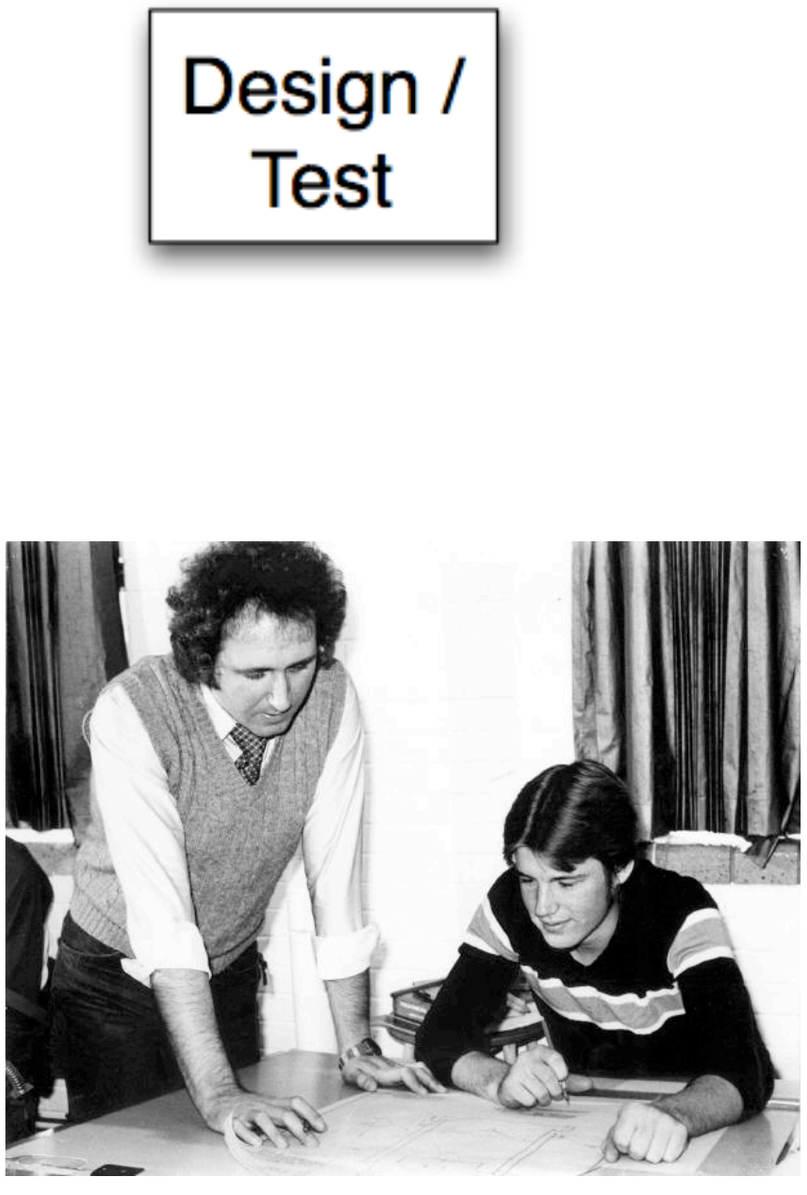
source code expresses the true and ultimate design - it is therefore a design artifact

Creating good software requires skills and experience (craftsmanship)
Creating good software predictably over time requires some process (engineering)

Essay by Jack Reeves from The C++ Journal in 1982 sums up these ideas very nicely. See Robert Martin's Agile Software Development book for a copy.

The obvious conclusion is that the Vernon Cycle doesn't apply to custom software development, particularly for new products

Software != Widget



Creating software

but design and test is expensive

design actually encompasses many activities (architecture, specification, prototyping, design, coding, testing, …)

design happens throughout the life of the project

so if the designers and the coders are different people, coders will be changing design

source code expresses the true and ultimate design - it is therefore a design artifact

Creating good software requires skills and experience (craftsmanship)
Creating good software predictably over time requires some process (engineering)

Essay by Jack Reeves from The C++ Journal in 1982 sums up these ideas very nicely. See Robert Martin's Agile Software Development book for a copy.

The obvious conclusion is that the Vernon Cycle doesn't apply to custom software development, particularly for new products

Up vs Downstream

"Venturesome Consumption, Innovation and Globalization"
Amar Bhide of Columbia University
The Economist, July 29, 2006

Too much attention paid to upstream innovation: invention, engineering and science

Too little attention to downstream innovation: marketing, customization, productizing

downstream sort of innovation is more complex, more valuable, and stays closer to where it occurs

experience is starting to show up...

Experience

"GSD: Not a business necessity, but a march of folly"
Girish Seshagiri, IEEE Software, Sep/Oct 2006

- "The biggest driver of software costs is poor quality." quoting a paper from the Center for eBusiness@MIT
- "'Follow the sun' model is essentially a quick-and-dirty strategy that converts a schedule problem into a quality disaster."
- GSD favors waterfall methodology

Experience

"Lessons from Offshore Outsourcing"
Bhat, Gupta, Murthy, IEEE Software, Sep/Oct 2006

- shared goal
- shared culture
- shared process
- shared responsibility
- trust

studied the requirements engineering aspect of software development

root cause analysis of real-world case studies found these strategic success factors

So who has the natural advantage in providing the best software development service?

Let's look at what the global alternatives are for building software...

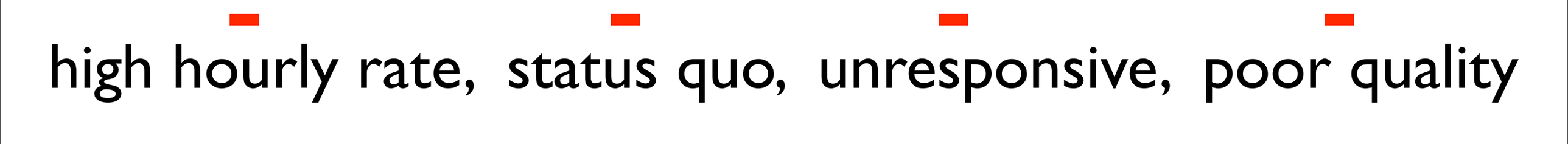
Global Alternatives

Suppose this is what you offer your company, or your customers.

Hardly a compelling choice.

Chad is speaking at the November XP West Michigan meeting

Global Alternatives



Suppose this is what you offer your company, or your customers.

Hardly a compelling choice.

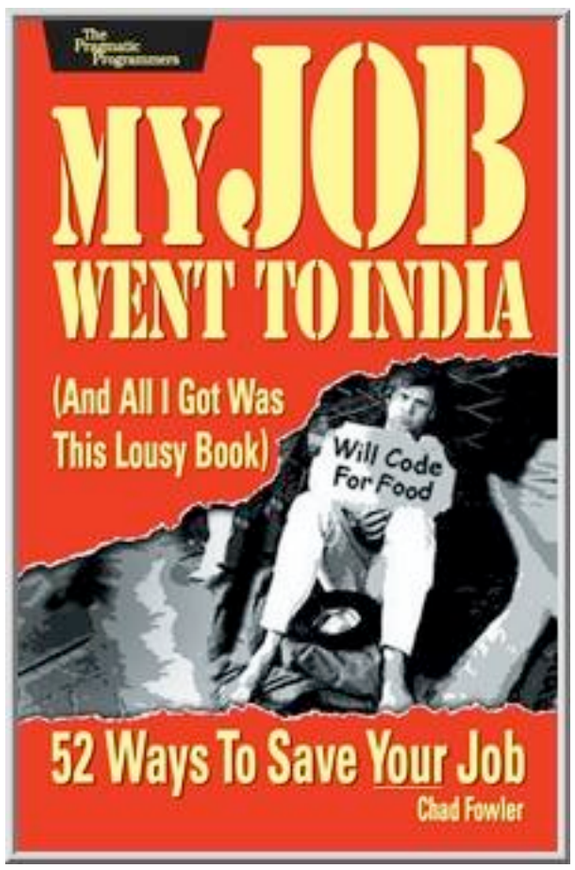
Chad is speaking at the November XP West Michigan meeting

Global Alternatives

high hourly rate, status quo, unresponsive, poor quality

"If a company is going to create a rut and pay humans to fall into it, it makes sense to pay as little as possible. The average software developer is a lemming, marching with his or her eyes to the ground toward an unknown goal."

-- Chad Fowler



Suppose this is what you offer your company, or your customers.

Hardly a compelling choice.

Chad is speaking at the November XP West Michigan meeting

Global Alternatives

the offshore alternative is similar, but less costly per hour

What's compelling is the third alternative...

Conclusion: we can compete globally, we have many natural advantages
but it's no longer a foregone conclusion we'll get the work

local excellence is required to compete globally

quality + innovation == excellence

At Atomic we call the combination of quality and innovation .... software craftsmanship

Global Alternatives

high hourly rate, status quo, unresponsive, poor quality

Uncompetitive local

low hourly rate, status quo, unresponsive, poor quality

the offshore alternative is similar, but less costly per hour

What's compelling is the third alternative...

Conclusion: we can compete globally, we have many natural advantages
but it's no longer a foregone conclusion we'll get the work

local excellence is required to compete globally

quality + innovation == excellence

At Atomic we call the combination of quality and innovation .... software craftsmanship

Global Alternatives

high hourly rate, status quo, unresponsive, poor quality

Uncompetitive local

low hourly rate, status quo, unresponsive, poor quality

Uncompetitive offshore

the offshore alternative is similar, but less costly per hour

What's compelling is the third alternative...

Conclusion: we can compete globally, we have many natural advantages
but it's no longer a foregone conclusion we'll get the work

local excellence is required to compete globally

quality + innovation == excellence

At Atomic we call the combination of quality and innovation .... software craftsmanship

Global Alternatives

the offshore alternative is similar, but less costly per hour

What's compelling is the third alternative...

Conclusion: we can compete globally, we have many natural advantages
    but it's no longer a foregone conclusion we'll get the work

local excellence is required to compete globally

    quality + innovation == excellence

At Atomic we call the combination of quality and innovation .... software craftsmanship

Global Alternatives

high hourly rate, status quo, unresponsive, poor quality

Uncompetitive local

low hourly rate, status quo, unresponsive, poor quality

Uncompetitive offshore

high hourly rate, innovative, responsive, high quality

the offshore alternative is similar, but less costly per hour

What's compelling is the third alternative...

Conclusion: we can compete globally, we have many natural advantages
but it's no longer a foregone conclusion we'll get the work

local excellence is required to compete globally

quality + innovation == excellence

At Atomic we call the combination of quality and innovation .... software craftsmanship

Quality

1. Process
2. Testing
3. Design

many dimensions: i'll hit three briefly, and describe specific development practices

Rant:
these aren't debatable anymore
these aren't only for certain industries, or certain methodologies, or certain team or company sizes
abandon whatever excuses you've been using and figure these three out
    customers shouldn’t accept anything less
    developers shouldn’t work to any lower standard

my promise: master these and everybody - company, customer, developer, manager, user - will be better off

Process

- Frequent, regular delivery of working, complete, tested software
- Story-driven development
- Engaged customer

regular delivery - no technical debt, no confusion about status or progress

story driven - building software from customer priorities, letting business steer dev

engaged customer - feedback, requirements resolution

Testing

- Think of testing as a development activity
- Practice TDD
- Automate your tests

testing as dev activity – design, requirements, refactoring, done by developers

TDD – write a test, watch it fail, write the code, watch it pass, refactor

to eliminate the possibility of buildup of technical debt

automation – executable proof of being done, being right, running in regression mode

Note: this isn't the whole story on testing; this is a way of looking at testing you might not be familiar with.

Design

- Study design principles and patterns
- Strengthen your design skills

becoming a better designer - experience, examples, mentoring

this is hard, takes years, requires other people

are there any shortcuts?...

The Secret to Becoming a Better Designer

becoming a better designer sounds great, but how?

by far and away the most valuable thing you could learn from this talk

will be revealed shortly...

What's one thing that a software craftsperson can't easily do with this code?

and with this code?

```
public void addOdxFile(String fileName) {

      String path = getPath(fileName);
      Vector logicalBlocks = sigView.getLogicalBlocks();

      LogicalBlockWithSignature lBlock;
      long dBlockStart;
      long dBlockSize;
      long lBlockStart;
      long lBlockSize;

      Vector v;
      Vector dataBlocks;
      SessionDescription sd;
      DataBlock dBlock;

      v = ODX.loadFile(fileName);

      for (int i = 0; i < 1; i++) {    // Use only the first session
            sd = (SessionDescription) v.elementAt(i);
            dataBlocks = sd.session.dataBlocks;
            for (int b = 0; b < dataBlocks.size(); b++) {
                  dBlock = (DataBlock) dataBlocks.elementAt(b);
                  dBlockStart = Long.parseLong(dBlock.startAddress.toString(), 16);
                  dBlockSize = Long.parseLong(dBlock.endAddress.toString(), 16) - dBlockStart;

               for (int lb = 0; lb < logicalBlocks.size(); lb++) {
                  lBlock = (LogicalBlockWithSignature) logicalBlocks.elementAt(lb);
                  lBlockStart = Long.parseLong(lBlock.addr, 16);
                  lBlockSize = Long.parseLong(lBlock.size, 16);
                  if (lBlockStart + lBlockSize <
                     dBlockStart || lBlockStart > dBlockStart + dBlockSize)
                        continue;
                  lBlock.setSignature(dBlock.signature.toString().trim().toUpperCase());
                  break;
               }

               addFile(path + getName(dBlock.flashData.fileName.toString()));
         }
         setChanged();
         notifyObservers(sd.partnumber.toString());
   }
```

And finally what about this reasonably simple, clean-looking, readable Java?

it's got interesting logic that certainly needs testing

but it's also got a couple of crucial design flaws that makes it almost as hard to test as the previous examples

- violates Law of Demeter
- coupled to the filesystem

Now you're ready for the secret...

How will I write code to test this?

Wrestle with this question every day, every hour, with every feature you implement

Wrestling with and answering this question will push you to more testable designs.

More testable design is better design.

The quest for testability will lead you to design principles, patterns, and understanding.

Global Competition

high hourly rate, innovative, responsive, high quality

Here's where we want to be. This is the competitive local alternative.

The keys to this are: Quality and Innovation

Innovation

If a company sets its main focus on innovation and marketing, it will significantly outperform every other competitor in its industry.

-- Peter Drucker

(from Curtis Gray in Business Review West Michigan, June 15, 2006)

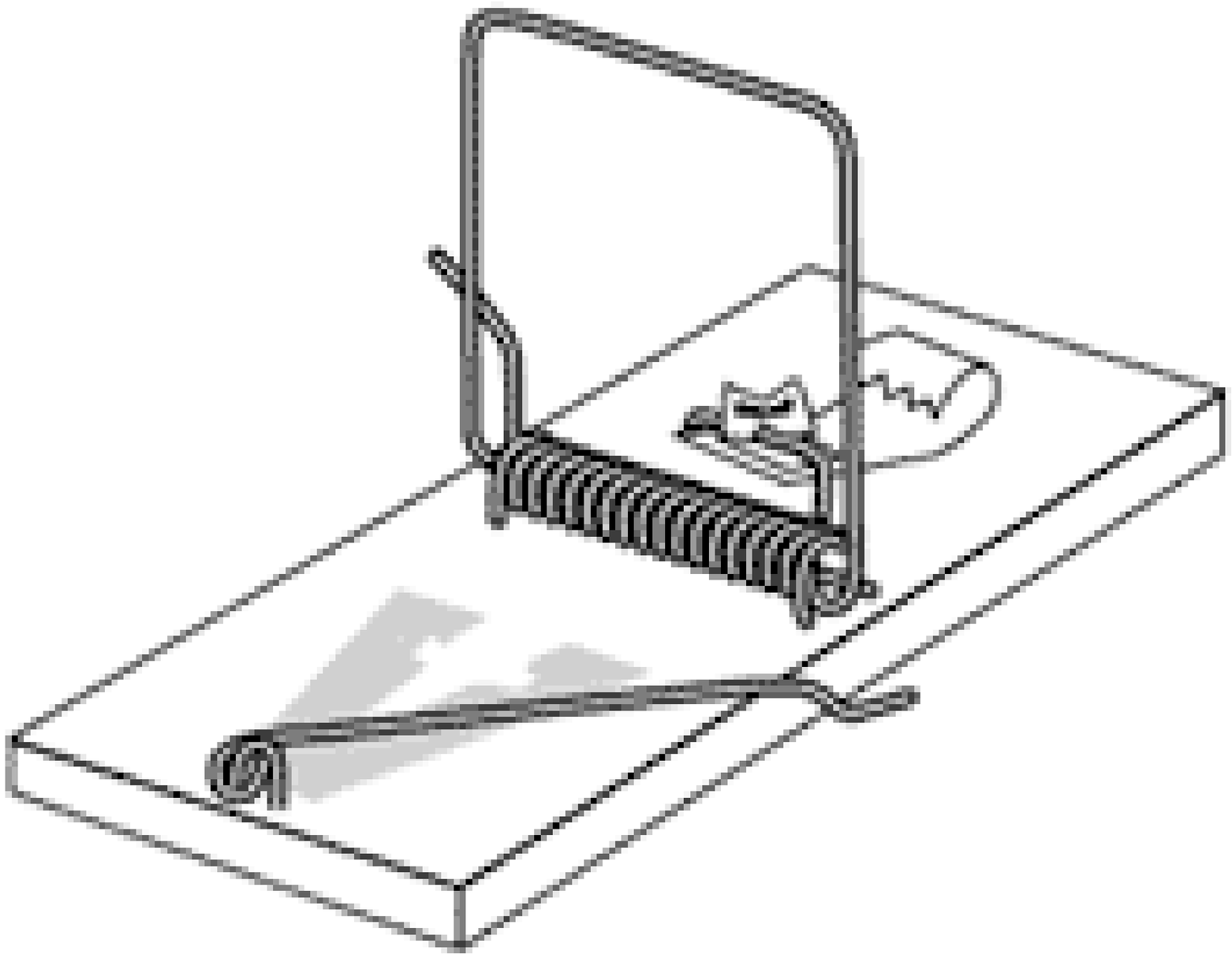
The more a company's revenue comes from new products, the more profitable it is.

Amazon: 165,635 book hits

Google: 212,000,000 page hits

Innovation is a hot topic
Innovation is scene as essential to business success

What does innovation mean for software developers?



classic view: lone genius, better mousetrap

what does this say about software development when you’re not a software company?

more and more mousetraps have software in them

SEI at Carnegie Mellon claims software content of products increasing 10x every 5 years

IT Service Innovation

1. "supporting the business's innovation agenda"

vs

"supporting the business"

Michael Schrage, CIO magazine, August 15, 2006

2. Getting better at crafting software

1. Getting closer to the business
2. Getting better at building software

Peter Coffee at this year's Agile Conference: know the business, know how to innovate products and services with technology. Don't just be the person they tell to build stuff.

At Atomic we spend a lot of time and effort on the second innovation

Necessity & Sufficiency

- Passing Operating Systems is *necessary* for graduation, but not *sufficient*

  (since Data Structures, Networks, etc, are also required)

- Satisfying all course requirements is *sufficient* for graduation, though individually any given elective is not *necessary*

To use some terms from logic, I believe we've found not only a necessary, but a sufficient condition for software development innovation.

A sufficient condition is a strong statement. It's what gets you what you want.

Here's an example to easily understand necessary and sufficient:
operating systems class at GVSU

First, the necessary condition...

Smart, creative people

Necessary condition: smart, creative people

These are the ones I have the privilege of working with and learning from

Smart and creative isn’t sufficient. The key ingredient you have to have to guarantee innovation is...

Smart, creative people *who care*

People who don't care...

Sufficient condition

People who care are obviously a good thing, but why can I claim that's sufficient for innovation?

Consider the opposite case to see why...

People who don't care
- are happy with the status quo
- don't feel pressure to do better
- withhold creative energy and commitment

This doesn't lead to innovation. It leads to TV watching.

My view is that...

Smart, creative people *who care*

People who don't care...

- are satisfied with the status quo
- don't feel pressure to do better
- withhold their creative energy
- keep their commitment low

Sufficient condition

People who care are obviously a good thing, but why can I claim that's sufficient for innovation?

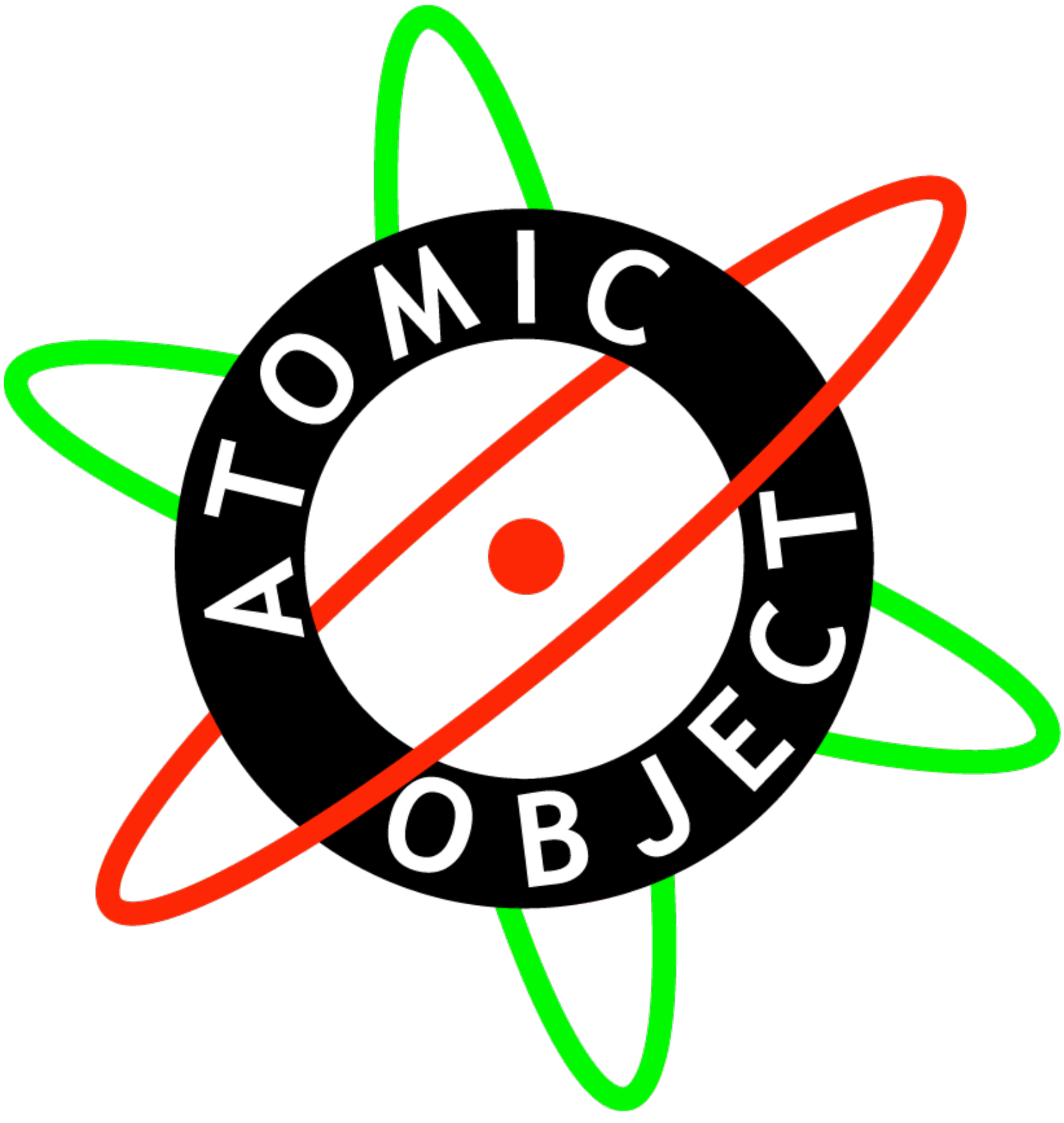
Consider the opposite case to see why...

People who don't care
are happy with the status quo
don't feel pressure to do better
withhold creative energy and commitment

This doesn't lead to innovation. It leads to TV watching.

My view is that...

Innovation is a natural consequence of a combination of attitude and circumstance.

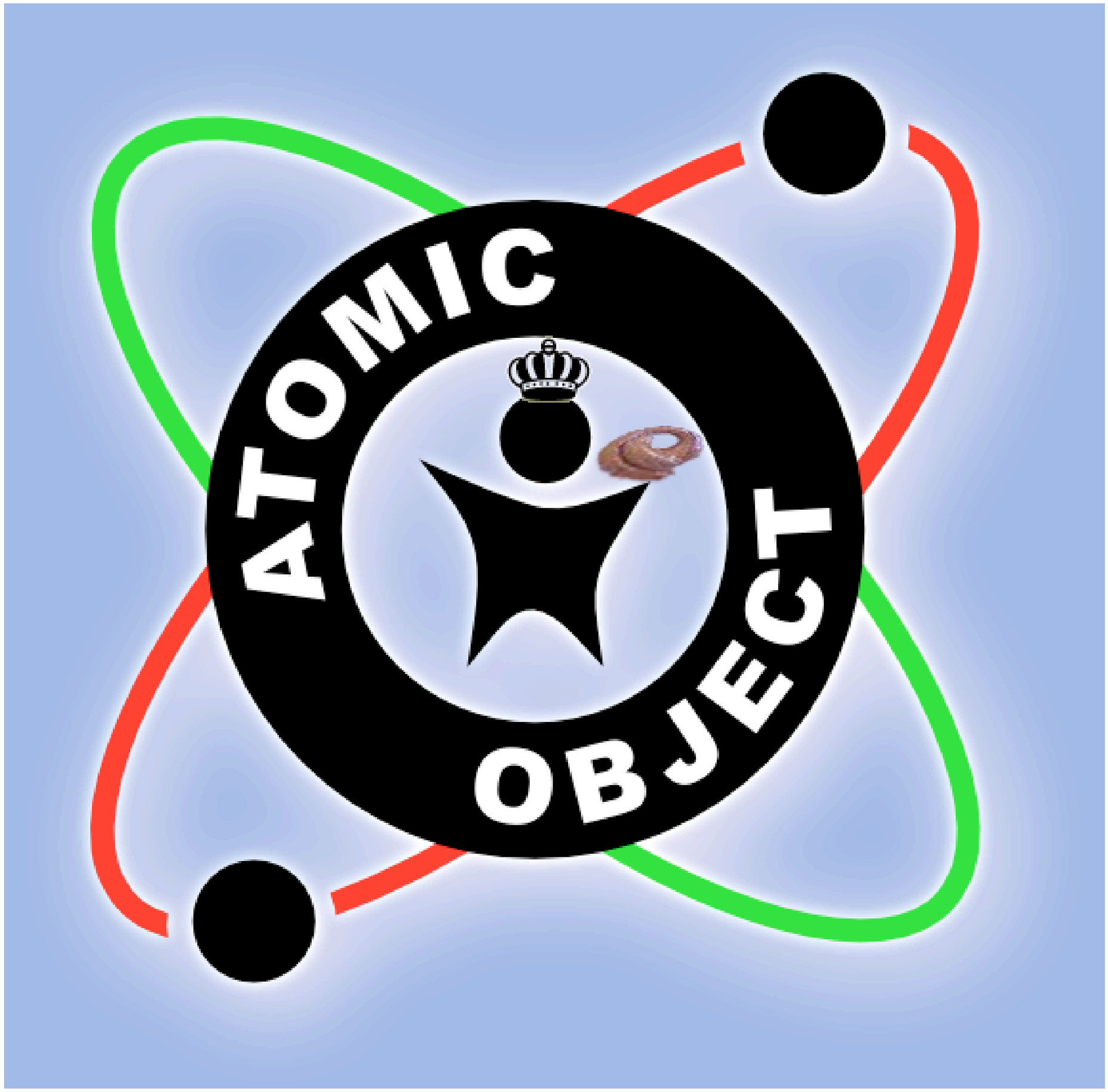


A year or so ago we spent some time deciding what the non-negotiables were for Atomic to hire someone.

On the short list was “they must care”. Only since this is an internal document it’s actually known by the more colloquial “must give a shit”.

That in turn inspired one of our guys to craft our secret, internal logo, rarely seen in public...

Thanks for listening, resources on our new website including these notes...



A year or so ago we spent some time deciding what the non-negotiables were for Atomic to hire someone.

On the short list was “they must care”. Only since this is an internal document it’s actually known by the more colloquial “must give a shit”.

That in turn inspired one of our guys to craft our secret, internal logo, rarely seen in public...

Thanks for listening, resources on our new website including these notes...

Custom Software Development, Test Driven Development, Agile ...Extreme Programming, Contract Programming | atomicobject.com

http://atomicobject.com/

Google

UU/IT/UpCERG Punchit PlanIt ToDo Sugar Contacts Wiki CampFire Phoenix WB Breast Reduc... for Review

Atomic PunchIt | Atomic PlanIt | SugarCRM - ... | SugarCRM - ... | Website > To... | http://georgi... | Custom Softw... | Royalty free s...

Atomic Object

Contact Us
Atomic Spin Blog

Home

- SERVICES
- WHY CHOOSE US
- PORTFOLIO
- COMPANY
- PRESS CENTER
- WHAT'S AN ATOMIC OBJECT?
- RESOURCE LAB

We build ridiculously good custom software for startups, huge companies, and everyone in-between.

Our testing and process fanaticism will win you over. Your software will have precious few bugs and will be delivered on time and within budget. No kidding.

Core Idea: Software Testing

5,317 tests at a click of a button? It's just one way our world class development process will give you joy. Why Choose Us...

Get Atomic Powered Custom Software

Our business nucleus is software development, but that's only one of the ways we can help you. Services...

Our Molecule & Software Craftsmanship

We're better today at our craft than yesterday, and we'll be even better tomorrow. Meet Our People...

Nuclear Waste? Nope. Sustainability.

We do way more than recycle. Our Green building, Agile trade association, and industry & community contributions are about Sustainability...

Atomic Spin WEBLOG

Latest News

Rails testing best practices at RubyConf*MI
Aug 26, 2006
Karlin Fox presented our best testing practices for Rails development at the first Ruby Conf MI

SD Best Practices 2006
Aug 03, 2006
Carl Erickson to speak at SD Best Practices 2006 in Boston

Innovation Essay in Business Review Western Michigan
Jul 20, 2006
Business Review Western Michigan publishes Carl's essay on innovation