



IEEE Managers Column

January 1996

# Objects as Property

by [Brad Cox](#), ([bcox@virtualschool.edu](mailto:bcox@virtualschool.edu)), <http://virtualschool.edu>

---

*Let us consider the inherent properties of this irreducible essence of modern software systems: complexity, conformity, changeability, and invisibility.* [\[Brooks\]](#)

This article will reexamine Brooks' famous argument from an entirely new viewpoint and arrive at opposite conclusions. Whereas Brooks used the techno-centric perspective of the software industry's established paradigm, we'll step outside of this frame and examine the same issue from a human-centric viewpoint. We'll not focus our microscope on how we build software today. Rather we'll focus on the people, or more precisely on the incentive structures that might lead people and firms to build software in the future as mature engineering domains produce tangible goods [\[Gibbs\]](#).

The departure from Brooks' analysis starts by recognizing that the "irreducible essence" of software is *not* "complexity, conformity, changeability, and invisibility" at all. These aren't causes but mere symptoms of a deeper irreducible essence. Software's irreducible essence, and the cause of these symptoms, is that it is made of bits, unlike tangible goods which are made of atoms.

Atoms abide by physical laws, such as conservation of mass, that enforce the meanings of terms like buying, selling and owning. Since these laws do not apply to bits, software engineers must rely on laws of man to provide, at far higher costs, what the laws of nature give to our hardware colleagues for free.

## Wooden versus Electronic Pencils

This article will show how essential difference impacts the human systems that produce goods, and how this difference in human systems causes the symptoms known as the software crisis [\[Gibbs\]](#). The first step is to distinguish two possibilities. Are tangible products (hardware) intrinsically simpler than digital products such as software? Or is the difference in the *human* systems that produce them? To illustrate this, it is useful to pick what may at first seem to be an overly simple example by comparing an ordinary wooden pencil with its digital analog, a word processor.

Even this mundane object turns out to be overwhelmingly complex inside. For example [\[Read\]](#) describes pencil manufacturing in the following manner:

I, Pencil, simple though I appear to be, merit your wonder and awe, a claim I shall attempt to prove. In fact, if you can understand me--no, that's too much to ask of anyone--if you can become aware of the miraculousness which I symbolize, you can help

save the freedom mankind is so unhappily losing. I have a profound lesson to teach. And I can teach this lesson better than can an automobile or an airplane or a mechanical dishwasher because--well, because I am seemingly so simple.

Simple? Yet, not a single person on the face of this earth knows how to make me. Pick me up and look me over. Not much meets the eye--there's some wood, lacquer, the printed labeling, graphite lead, a bit of metal, and an eraser. Just as you cannot trace your family tree back very far, so is it impossible for me to name and explain all my antecedents. But I would like to suggest enough of them to impress upon you the richness and complexity of my background. [\[Read\]](#)

He goes on to expose the vast social system that produces and markets pencils; the office supply clerks, truckers, distributors, foresters, lumberjacks, millworkers, graphite, tin and copper miners, rape seed oil farmers, enamel manufacturers... and diverse web that supplies the tools, supplies, housing and food to support them. The complexity of this web far exceeds the comprehension of any individual. Yet this complexity is somehow encapsulated so thoroughly that the rest of us have the illusion that pencils are in any way "simple". By contrast, according to the project manager of one of Microsoft Word earlier versions, the complexity of this electronic pencil was borne by a team of just eight programmers.

Our task in this article is to understand how this human system works for tangible goods, why it collapses for digital goods, and how it might be resurrected in the future by basing ownership on useright, not copyright, as described in "Superdistribution: Objects as Property on the Electronic Frontier" [\[Cox\]](#).

## Somebody Else's Problem

The pencil example brings into focus a crucial but slippery distinction. It may be academically amusing to contrast the absolutely complexity of wooden and electronic pencils but this doesn't really move the ball ahead. What does is realizing that *encapsulated* complexity is no longer complexity at all. Its gone, buried forever in somebody else's problem.

Pencil makers hide complexity by giving their sub-suppliers a financial incentive to hide sub-system complexity for them via commercial exchange transactions. The exchange transaction gives the office supply store its incentive to keep office workers from having to build their own pencils. It gives millworkers their incentive to hide milling's complexity from the office supply store. It gives lumberjacks their incentive to hide the complexity of logging from the mill. And it gives miners, drillers and farmers their incentive to hide the complexity of their particular mine, well or farm from everyone else in the global structure of production.

It does doesn't matter whether wooden pencils are more or less complex than electronic pencils. What does matter is that pencil manufacturers, and tangible goods manufacturers in general, have developed a thriving system for hiding complexity from everyone else. They've evolved a thriving **human** system that does this so well that everyone else has the luxury of forgetting that complexity was ever even there.

And if that doesn't qualify as a silver bullet for the complexity of manufactured goods, I'd have to agree with Brooks that there really is no hope for us.

## Atoms Versus Bits

The important thing to notice here is that conservation of mass has *reach*. Since conservation of mass is endogenous, a property of the very atoms themselves, it applies not only to the pencils but to the subcomponents from which pencils are composed.

The effect is that every pencil in every customer's hand is connected to the vast global tree of human relationships that produced it. Conservation of mass guarantees that the faintest trace of graphite on paper draws the right amount of new production from the lower levels of the tree. The consumption of a pencil in the Denver Airport is communicated, through price signaling throughout the tree. Price signalling tells the graphite miner how much graphite to mine, the lumberjack how many trees to fell, the mill operator how many pencils to produce, and the store owner how many pencils to stock in the store.

This tree of relationships is a vast ecological system, exactly like a peach tree. The tree draws precisely the right amount of water and sunshine from nature and transforms them into precisely the right numbers, kinds and amounts of amino acids, sugars and proteins to support production of the tree's commercially significant result. In other words, this system is *recursive* in a particularly powerful way. It doesn't just work at the level of peaches, pencils and word processors; the levels outsiders think of as "commercially significant". It doesn't only apply to peaches but with precisely equal force down through the leaves, the twigs, the branches, the trunk and the roots.

However since electronic goods, don't abide by conservation of mass, the vessels inside our electronic pencil tree get vapor lock. So the tree dies. Or more precisely, such trees never evolve in the first place. The problem isn't superficial. Its not that we can't afford to buy reusable components, nor the infamous "Not Invented Here" syndrome. Its that we don't know **how** to do it without incurring crippling transaction costs. Handling multigranular property as it flows through elaborated structure of production chains is impossibly expensive when there's nothing more than the laws of man (i.e. lawyers) to enforce the meaning of terms like buy, sell or own.

Complexity is not a fundamental cause. Its merely a symptom of a much deeper disease. In fact, this disease is at the deepest level, so deep that it didn't even make it onto Brooks' list of software's essences. The disease is that software objects are made of bits whereas objects are made from atoms which abide by the fundamental physical laws that underlie commerce transactions in tangible goods. The software industry has never figured out how overcome this fundamental difference. This is why we perform so poorly in relation to those who have. Although we do give an occasional nod to the advantages of "software reuse", we've been unable to surmount the messy problematics of compensating those who provide software components to reuse.

Demand for computer software is extremely high these days. High prices encourage us to provide large granularity computer software like word processors. But the smaller granularity objects we might use to build them can be replicated without revenue to their owner each time an application that relies on them is sold. So there is no incentive to persuade people to populate the lower levels of the tree. We all naturally gravitate to the highest granularity levels where we can get paid. We all struggle to build large applications unaided, so structure of production trees never evolve. Since supporting levels don't evolve, the entire task falls to the team at the very top of what would otherwise be a cooperative human pyramid. And we all gather at software engineering conferences to complain of the crushing complexity of our task.

## Copyright versus Useright

Notice that the problem cannot be addressed at the level of large granularity computer applications. It involves attention to producing healthy roots, trunk and branches that support the production of

commercially significant fruit. However the fruit is the only level that is getting serious attention today. Copy protection technologies like dongles, shareware unlocking technologies, license servers, and encryption are all unigranular solutions that deal with objects large enough to stand on their own but nothing for the smaller-granularity components from which large-granularity objects might someday be composed. Yet large applications are the one level of the tree that least needs greater protection. After all, applications can be protected by simply attaching them paper and cellophane that can be bought and sold like cornflakes in an ordinary industrial age software store.

The path out of the crisis involves finding a viable replacement for the quantum event at the core of the economic tree that produces tangible goods like pencils. This quantum event is the commercial exchange transaction. Most commonly, this event relies on conservation of mass to define exchange in terms acquisition of atoms. But even in tangible domains, acquisition of atoms is not the only basis of owning. The path out of the software crisis involves another interpretation of owning, which is based on acquisition of use instead.

We don't generally think of these two meanings as separate because most common transactions bundle them together. When I buy an car, I acquire ownership rights to the physical atoms, and conservation of mass ensures that there's one less car for sale. A scarcity pulse travels throughout the car structure of production, renewing the market at each level and encouraging each level to produce more.

But I also acquire as part of my rights bundle the right to *use* the automobile as I please. I can drive it myself, drive it off a cliff, or sell it to someone else. My transaction conveyed not just copyright, enforced not by lawyers but by the natural laws that make tangible goods hard to replicate. Avis and Hertz shows that the right to *use* property can be bought and sold independently from who owns the physical atoms.

## Out of the Crisis

This distinction between copyright and useright points to where software's escape from the crisis might be launched. We buy and sell software as we buy and sell automobiles. And unlike atoms, bits are liabilities, not assets. Bits consume storage space that might be used for something else. And again unlike atoms, bits represent no cost to the provider, since their sale generates no scarcity pulse that requires the owner to buy more.

So why don't we just give them away? What people want is the ability to **use software functionality and not the bits themselves. So why not base revenue collection on use of the functionality instead?**

**This shift of the ownership issue from copyright to useright immediately puts the whole issue on ground where technology can apply. Although software is unable to monitor its *acquisition*, it is trivially able to monitor its *use*. Like this:**

```
if (query()) {      /* is this a paid up customer? */
    ...            /* deliver requested service */
    commit();       /* record successful delivery */
} else {           /* deny further service */
    ...
}
```

**These query and commit calls rely on infrastructure that doesn't exist now, but infrastructure can be built easily. Obviously it needs to be tamper-resistant and protective of privacy. But such issues are simply a matter of engineering and entirely within the computer industry's**

capabilities.

## Endogeneous versus Exogenous Enforcement

Notice that the software industry relies entirely on exogeneous techniques for protecting ownership today. For example, encryption, shrinkwrap, dongles and software stores are all external to the software itself. However invocation metering is an endogeneous approach, exactly as conservation of mass is an endogeneous property of tangible goods. Invocation metering instructions convey the same recursive properties to software that conservation of mass conveys to commerce in tangible goods.

When a large object runs on an end-user's PC, the user will be charged for the use this application, but *the application's owner* is charged for his application's use of the reusable subcomponents it contains. Metering of invocation could then carry energy from the top to the roots of robust digital structure of production trees, exactly as conservation of mass conveys scarcity in one direction (and revenue in the other) in manufacturing.

## References

[Brooks]: [No Silver Bullet; Essence and Accidents of Software Engineering](#); Fred Brooks;

[Gibbs]: Software's Chronic Crisis; Trends In Computing, by W. Wayt Gibbs; Scientific American; September 1994; Page 86

[Read] "I, Pencil; My family tree as told to Leonard E. Read" by Leonard E Read, available online at </mon/Economics/ReadIPencil.html>

[Cox] Brad Cox; "Superdistribution: Objects as Property on the Electronic Frontier; Addison Wesley Publishing Company; digital and paper editions at <http://www.virtualschool.edu/mon>.

## Acknowledgements

This article was adapted from [No Silver Bullet Reconsidered](#); American Programmer Magazine; November 1995 and "Superdistribution: Objects as Property on the Electronic Frontier"; Addison Wesley Publishing Company.

---

X-Sender: rspa@pop.connix.com  
Mime-Version: 1.0  
Date: Tue, 16 Jul 1996 16:56:35 -0500  
To: Brad Cox  
From: pressman@rspa.com (Roger Pressman)  
Subject: Re: A guest column for IEEE software

Brad,

Thanks for the rapid positive response. It was, in fact, your article in AP that reminded me to put you on the list of potential contributors to \*Manager\*. I enjoyed it the first time I read it. It's exactly the type of material we need!

Two things: (1) "No Silver Bullet Reconsidered" is about 4,000 words in

length; we max out at about 2,250. (2) by policy we do not use previously published reprints in the Manager column ... however, adapted material from published articles is fine.

Here's what I recommend:

The basic thrust of the piece is encapsulated by your comment:

>This paper will reexamine Brooks' argument from a different perspective to  
>arrive at precisely opposite conclusions. Whereas he adopted the  
>techno-centric perspective of the software industry's established paradigm,  
>we'll step outside of this frame to examine it from a human-centric viewpoint.  
>We'll not focus our microscope on how we build software today. We'll focus  
>instead on the people, or more precisely on the incentive structures that  
>might lead people and firms to build software in the future as mature  
>engineering domains build baggage carts and airplanes today.

I would suggest an edit of the article to reduce its length but maintain its thrust. Although it's all very good stuff, I suspect you could remove refs to the Denver airport and the lengthy discussion of the genesis of the pencil without damage. In addition, it should be possible to abbreviate some of your discussion without losing its intent. Overall, the goal is a column of 2,000 words.

In that way, we could publish a 'new' column that has been 'adapted' from the AP piece, rather than a reprint, which we don't do.

If this is acceptable, let's go with "No Silver Bullet Reconsidered" in a modified form.

I had you slotted for the July, 97 issue. However, if you have the time to do an edit and get me the draft before 1 Oct 96, I'd actually like to juggle and editorial calendar and lead off the year (January, 1997 issue) with your piece. Let me know what you think.

Best regards and thanks,

Roger

-----  
Roger S. Pressman, Ph.D.  
R.S. Pressman & Associates, Inc.  
620 East Slope Drive  
Orange, CT 06477 USA  
voice: 203.795.5044  
fax: 203.799.1023  
e-mail: [pressman@rspa.com](mailto:pressman@rspa.com)  
WWW: <http://www.rspa.com>

[Virtual School](#)

[Middle of Nowhere](#)

[Brad Cox](#)