digital consulting inc.

presents

SOFTWARE WORLD USA Presenting...

CASE WORLD, OBJEX & APPLICATION DEVELOPMENT WORLD

Volume II
Tuesday & Wednesday, March 1-2, 1994

This Book Belongs to:

Name: _____

Company: _____

Address: _____

_____

Phone: _____

Section C
Tuesday, March 1, 1994

11:00 am
Tracks

Transitioning to Object-Orientation

(A Twelve-Step Treatment Program)

SOFTWARE WORLD / USA

San Francisco

1 March 1994

Carl A. Argila, Ph.D., Inc.
SOFTWARE ENGINEERING CONSULTANT
P. O. Box 1219-B
Pico Rivera, CA 90660-1219
310-699-3196
carl@acm.org

Step 1: Accept the Inevitable

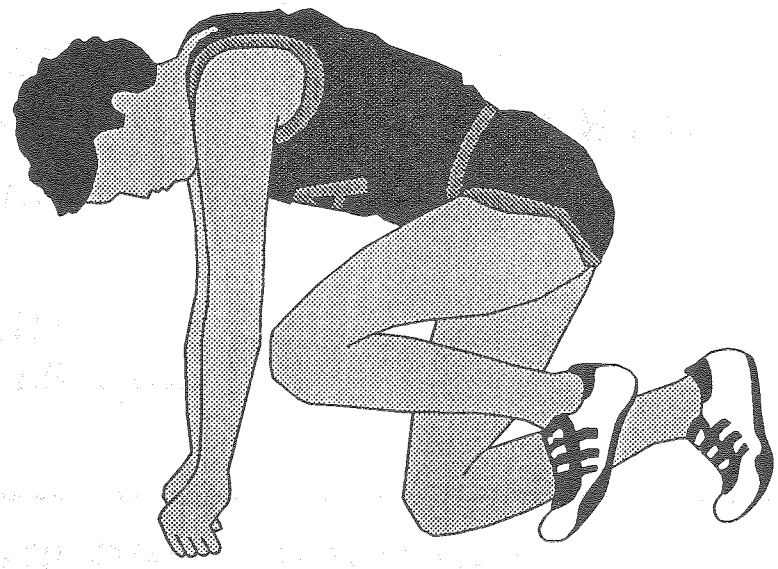
- Object-orientation will be the dominant software development paradigm of the future!
- There is the potential for very significant productivity and quality improvements.
- Vendor support is growing by leaps and bounds.
- Early adoptors are reporting success.
- There are serious standards efforts underway.
- There are de facto standards in OOPLs and emerging standards in OOA and OOD.

Management Imperatives

CHEAPER

BETTER

FASTER



Step 2: Understand, Understand, Understand

- Do you know why you're doing this???
  (It's re-use, stupid!!!)
- Understand your motivations.
  If you're just interested in short-term gain, you'll be disappointed!
- Understand the "paradigm switch."
  Traditional "top-down" functional decomposition vs. "middle-out" collaborations of objects.
- Understand "evolutionary" vs. "revolutionary" transition.
- ...Then, create a "manifesto" which presents a clear vision of where you want to be after you've completed this transition.

Step 3: Assess Your Assets

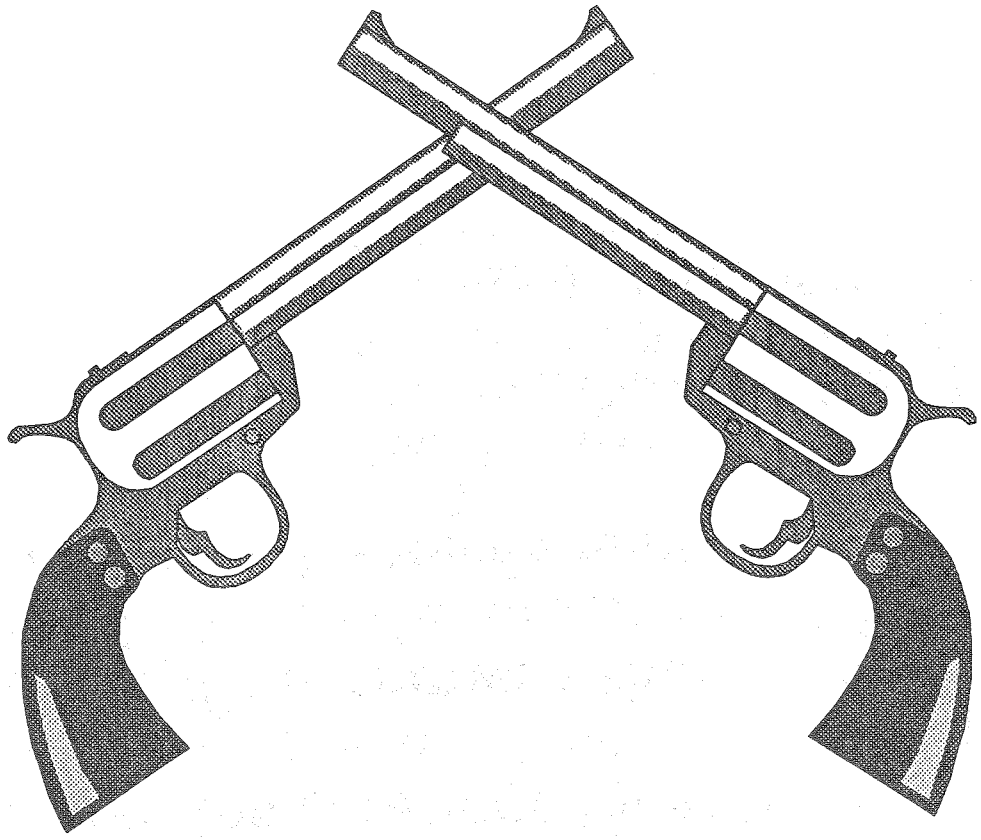
- Assess your software development process.
  Establish some sense of where your software development process is "at."
  Distinguish between *work-products* and *artifacts*.
  What artifacts does your staff create? How do those artifacts become deliverable work-products?
- Which of your artifacts can you salvage as you transition to object-orientation?
- Assess your peopleware assets.
  What artifacts does each person create and how will that skill support your new software development approach?
- Base your transition plans on your assets!

"Peopleware"

Powerful programming languages and fast compilers do not produce good software. Advanced development methods and software engineering practices may help, but offer no guarantees. In the real world of software and applications development, even the most rapid of rapid prototyping takes time, and even mildly sophisticated systems need the contributions of multiple developers. Under these real circumstances, how the human resources of programming are organized and managed become crucial factors in the succes or failure of development projects. Only good people, well organized and well managed to enhance their productivity and the quality of their work, can produce good software.

Larry L. Constantine
Lucy A. D. Lockwood
Orchestrating Project Organization and Management
Communications of the ACM
October 1993

Techno-Cowboys



Techno-Cowboys

A cohesive team cannot tolerate extremist mavericks. One of our most difficult realizations was that some talented individuals cannot flourish in a team-oriented environment.

Risa B. Hyman
Creative Chaos in High-Performance Teams: An Experience Report.
Communications of the ACM
October 1993.

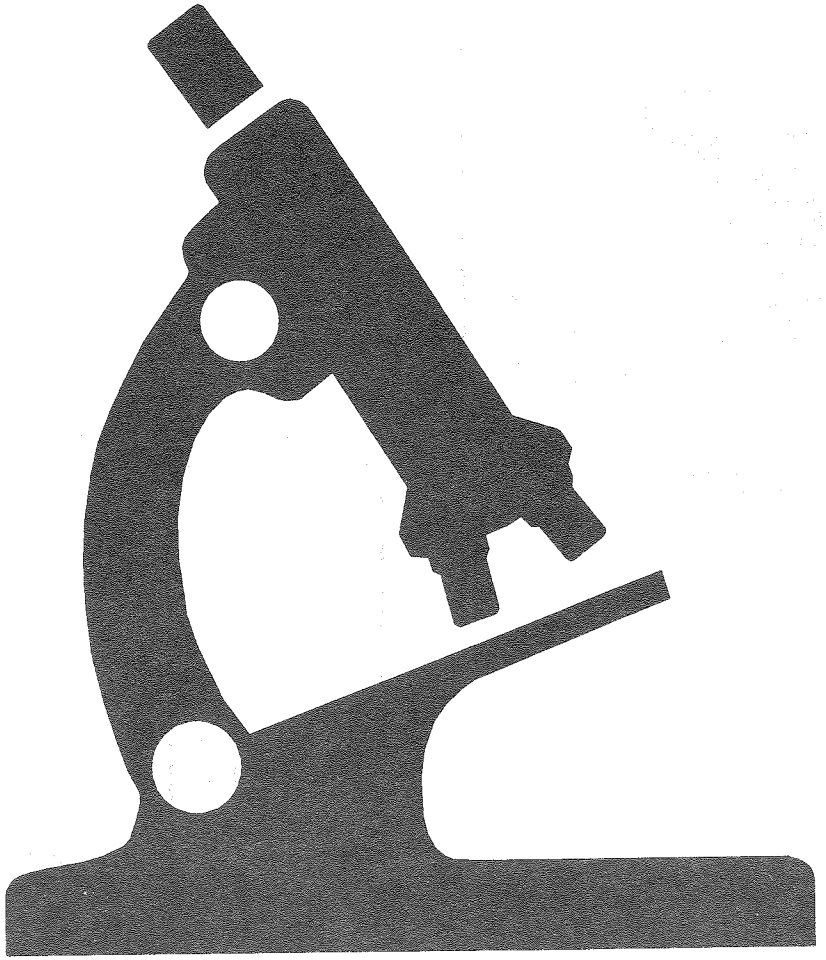
Step 4: Identify a "Symbioject"

- It's foolhardy to introduce any radically new or different technology into a "mission critical" project without first completely understanding all aspects of the technology and its ramifications for the project.
- Pilot projects are excellent vehicles for learning a new technology, but frequently get thrown out after they're completed.
- Initiate a pilot project which has a symbiotic relationship with a mission critical project -- this can actually be a component of a mission critical project which can be comfortably addressed as a quasi-pilot project.

Step 5: Establish Meaningful Metrics

- If you don't know what's being done you can't measure. If you can't measure you can't predict. If you can't predict you can't control. If you can't control you can't manage!
- Establish realistic, meaningful artifact based (not workproduct based) metrics. Then use them!
- Establish criteria, based on these metrics, for performance and project success.

Micro-Metrics



Micro-Monitoring

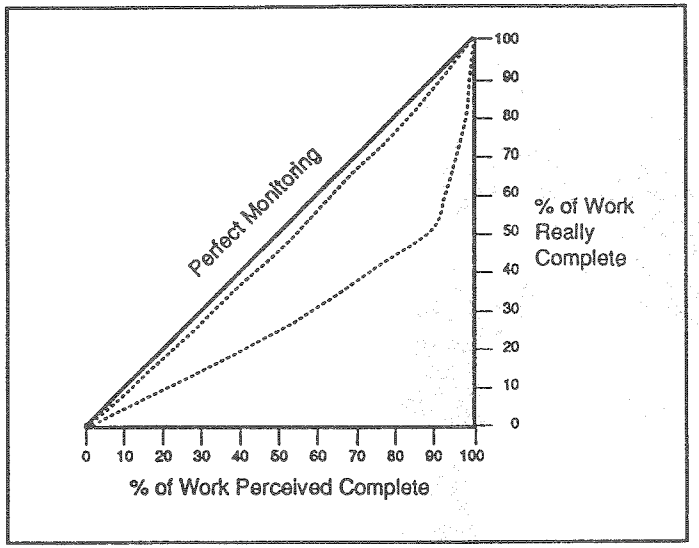


Figure 8: Progress ramps of commercial software projects

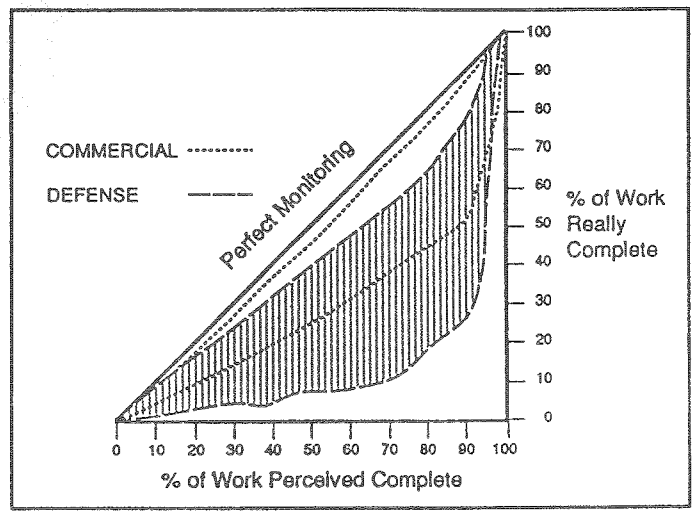


Figure 9: Progress ramps

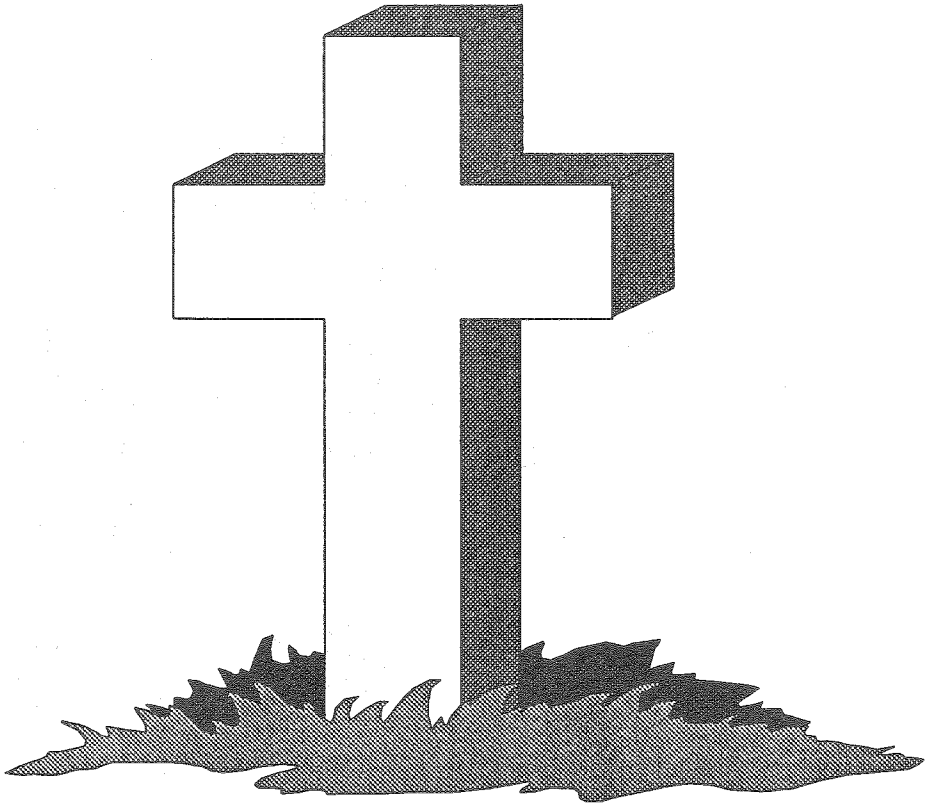
From: *Swords and Plowshares: The Rework Cycles of Defense and Commercial Software Development Projects*
American Programmer, May 1993.

Step 6: Plan for "The Games"

- "The Games" will be played...whether or not you want them to be played...whether or not you choose to play..."The Games" will be played! So plan for them!!!
- Creative Avoidance:
  "We can't do this...why...why...we don't have the right CASE tool...the right methodology...the right (fill in the blank)"
- Malicious Compliance:
  "You want objects? I'll give you objects!!!"
- The Potemkin Syndrome

Beware of "The Potemkin Syndrome"

Potemkinism can kill...



Your Project!

Your Company!!!

Step 7: Plan for N-Squared

- Murphy's Law No. 382: If two complex computer technologies can possibly interact, they will interact, and they will do so in the most obscure and difficult manner possible.
- New Application + New Methodology + New CASE Tool + New Platform + New O/S + New DBMS + New ... = New Job !
- Limit the new technology introduced into your first object oriented project.

Step 8: "Jump Start" the Learning Curve

- Recognize the learning curve. There will be a significant learning curve as you transition into objectorientation.

- Provide your staff with proper training and support. It's an investment in the future success of your projects. Remember: "In the long run it's the stingy man who winds up spending the most" (Klick & Klack the "Car Guys").

- Types of Training:
  Just-in-Time vs. Just-too-Late

Step 9: Get Help for Your First Information Model

- That first information model is REALLY important -- and it's REALLY easy to botch up.
- Your first information model will be the foundation for your reuse library and lots of other things will be built upon it.

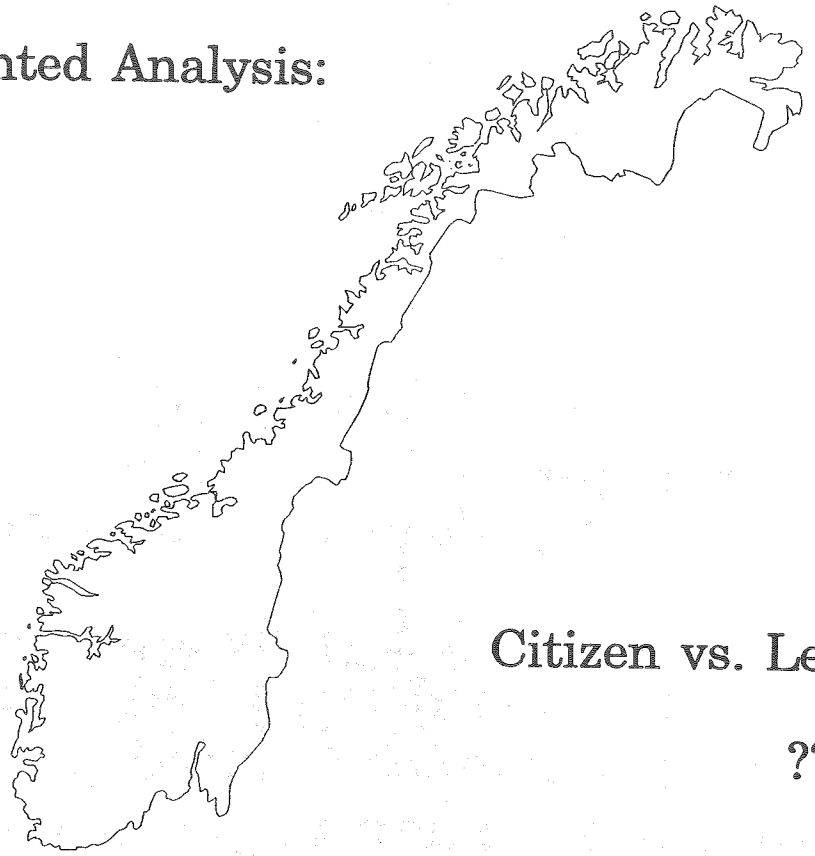
Carl A. Argila, Ph.D., Inc.
SOFTWARE ENGINEERING CONSULTANT

REUSEFUL Not Just REUSABLE

...for watering flowers in the rain.

Norwegian Department of Social Security

Object-Oriented Analysis:



Citizen vs. Legislative Rule

???

Step 10: "Time-Box" Your First Project

- It's really easy to let your first object-oriented project get out of hand. You must "time-box" your first project, reducing the complexity and/or scope of the project as necessary...but not the delivery date!
- Getting that first project out on schedule is a real morale booster, establishes credibility, creates political capital, etc.

Step 11: Begin That Re-Use Library

- Mechanisms establish the basic mechanism for re-use within your corporate environment. Remember that re-use library doesn't start with objects! It start's with a corporate commitment that the NEXT project will re-use objects.
- Your corporate culture may need to change to accommodate re-use. Individual project managers may have to be directed to "re-use" from "a higher authority!"
- Motivators establish real motivators for re-use. How about royalty payments to class developers? Bonuses to re-use users? And special incentives to project managers?
- Measuring re-use Measuring the reusable portion of a delivered software product should be a part of every project's "vital stats."

Step 12: Conduct a Serious Postpartum/Postmortem

- What major organizational changes should be considered at this point???
- Shake up the organization: Ultimately your organization will be split into the "object builders" (the "software factory") and the "system builders." Will your organization be able to adapt?

Seek out the Messengers, but...

...PLEASE, Never, Never Shoot Them!!!

How Will You Know it When You Get There???

- You will have established two distinct classes of people: the "object builders" and the "system builders." More and more this will become part of your "corporate culture."
- You will know, unhesitatingly, what everyone has done, is doing and will do on every project day. And you will use that knowledge to predict your performance on this and future projects.
- You will have in place the kernel of a re-use library and will be planning future system based on this re-use library. Like your new organizational structure, re-use is becoming part of your "corporate culture."
- You'll feel comfortable about introducing object-orientation into larger and more critical projects.