Science is not Enough

On the Creation of Software

Very Confident | Richard P. Gabriel | 3 September 2009

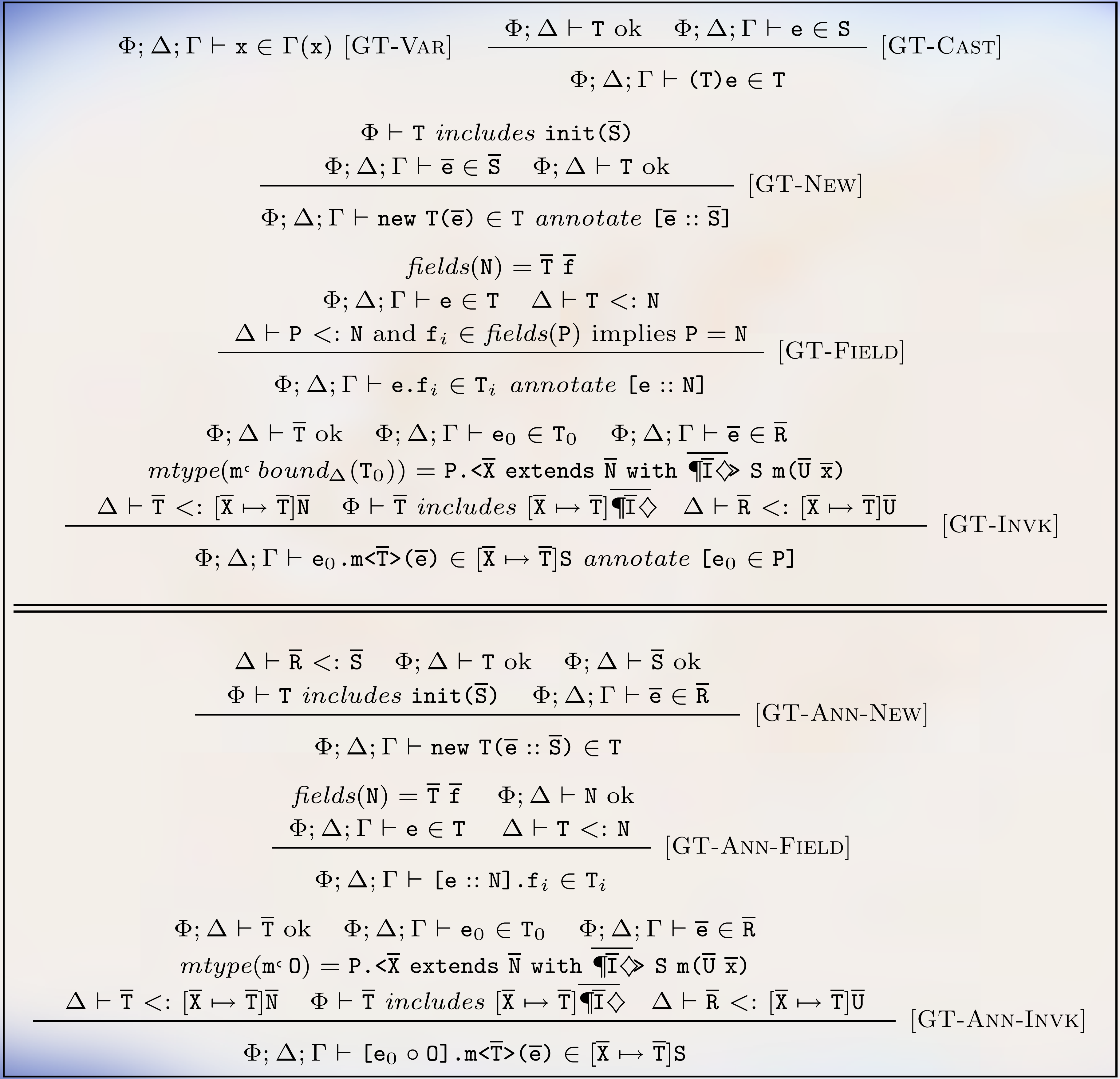
Only mystery enables us to live.

Only mystery.

–Federico Garcia Lorca

Prefrontal lightningbolt too lazy to chew the sphinx's loudest eyelash
Not even if it shushes you with a mast of sneers
Down which grateful bankvault-doors scamper
Because of a doublejointedness that glows in the dark
Like a soliloquy of walnuts
Numbed by beaks of headless measuringtape
So the lubriciousness can tower in peace
Like a buzzsaw trapped in a perfumery of shrugs
Lemon
Or lime
Only a maze can remember your hair of buttered blowguns

–Bill Knott, *Nights of Naomi*



$$\Phi;\Delta;\Gamma\vdash \texttt{x}\in\Gamma(\texttt{x})\ [\textsc{GT-Var}] \qquad \frac{\Phi;\Delta\vdash\texttt{T}\ \mathrm{ok}\quad \Phi;\Delta;\Gamma\vdash\texttt{e}\in\texttt{S}}{\Phi;\Delta;\Gamma\vdash(\texttt{T})\texttt{e}\in\texttt{T}}\ [\textsc{GT-Cast}]$$

$$\frac{\begin{array}{c}\Phi\vdash\texttt{T}\ \mathit{includes}\ \texttt{init}(\overline{\texttt{S}})\\ \Phi;\Delta;\Gamma\vdash\overline{\texttt{e}}\in\overline{\texttt{S}}\quad \Phi;\Delta\vdash\texttt{T}\ \mathrm{ok}\end{array}}{\Phi;\Delta;\Gamma\vdash\texttt{new T}(\overline{\texttt{e}})\in\texttt{T}\ \mathit{annotate}\ [\overline{\texttt{e}}::\overline{\texttt{S}}]}\ [\textsc{GT-New}]$$

$$\frac{\begin{array}{c}\mathit{fields}(\texttt{N})=\overline{\texttt{T}}\ \overline{\texttt{f}}\\ \Phi;\Delta;\Gamma\vdash\texttt{e}\in\texttt{T}\quad \Delta\vdash\texttt{T}<:\texttt{N}\\ \Delta\vdash\texttt{P}<:\texttt{N}\ \text{and}\ \texttt{f}_i\in\mathit{fields}(\texttt{P})\ \text{implies}\ \texttt{P}=\texttt{N}\end{array}}{\Phi;\Delta;\Gamma\vdash\texttt{e.f}_i\in\texttt{T}_i\ \mathit{annotate}\ [\texttt{e}::\texttt{N}]}\ [\textsc{GT-Field}]$$

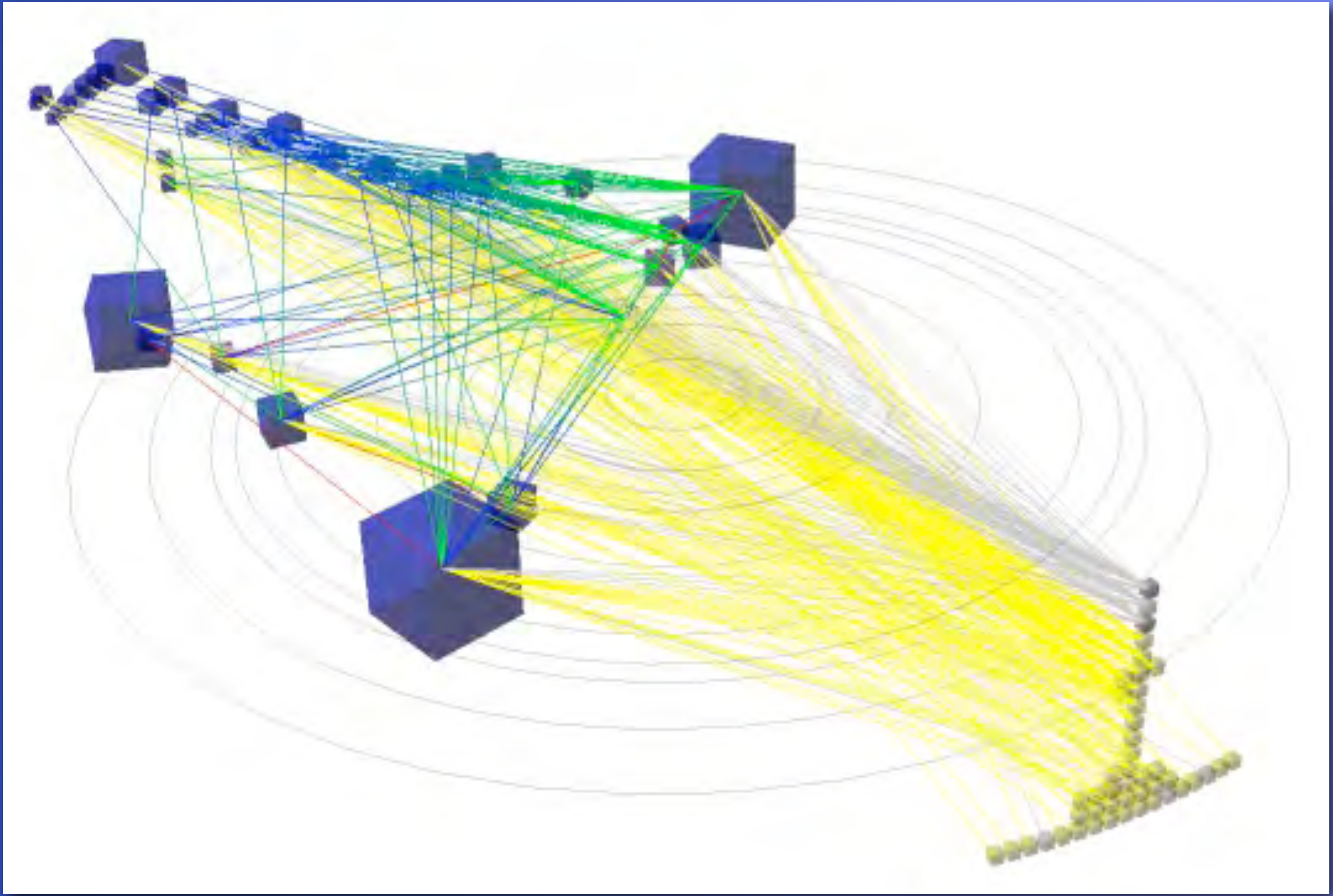
$$\frac{\begin{array}{c}\Phi;\Delta\vdash\overline{\texttt{T}}\ \mathrm{ok}\quad \Phi;\Delta;\Gamma\vdash\texttt{e}_0\in\texttt{T}_0\quad \Phi;\Delta;\Gamma\vdash\overline{\texttt{e}}\in\overline{\texttt{R}}\\ \mathit{mtype}(\texttt{m‹}\ \mathit{bound}_\Delta(\texttt{T}_0))=\texttt{P.<}\overline{\texttt{X}}\ \texttt{extends}\ \overline{\texttt{N}}\ \texttt{with}\ \overline{\P\texttt{I}}\diamondsuit\ \texttt{S m}(\overline{\texttt{U}}\ \overline{\texttt{x}})\\ \Delta\vdash\overline{\texttt{T}}<:[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\texttt{N}}\quad \Phi\vdash\overline{\texttt{T}}\ \mathit{includes}\ [\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\P\texttt{I}}\diamondsuit\quad \Delta\vdash\overline{\texttt{R}}<:[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\texttt{U}}\end{array}}{\Phi;\Delta;\Gamma\vdash\texttt{e}_0\texttt{.m<}\overline{\texttt{T}}\texttt{>}(\overline{\texttt{e}})\in[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\texttt{S}\ \mathit{annotate}\ [\texttt{e}_0\in\texttt{P}]}\ [\textsc{GT-Invk}]$$

$$\frac{\begin{array}{c}\Delta\vdash\overline{\texttt{R}}<:\overline{\texttt{S}}\quad \Phi;\Delta\vdash\texttt{T}\ \mathrm{ok}\quad \Phi;\Delta\vdash\overline{\texttt{S}}\ \mathrm{ok}\\ \Phi\vdash\texttt{T}\ \mathit{includes}\ \texttt{init}(\overline{\texttt{S}})\quad \Phi;\Delta;\Gamma\vdash\overline{\texttt{e}}\in\overline{\texttt{R}}\end{array}}{\Phi;\Delta;\Gamma\vdash\texttt{new T}(\overline{\texttt{e}}::\overline{\texttt{S}})\in\texttt{T}}\ [\textsc{GT-Ann-New}]$$

$$\frac{\begin{array}{c}\mathit{fields}(\texttt{N})=\overline{\texttt{T}}\ \overline{\texttt{f}}\quad \Phi;\Delta\vdash\texttt{N}\ \mathrm{ok}\\ \Phi;\Delta;\Gamma\vdash\texttt{e}\in\texttt{T}\quad \Delta\vdash\texttt{T}<:\texttt{N}\end{array}}{\Phi;\Delta;\Gamma\vdash[\texttt{e}::\texttt{N}]\texttt{.f}_i\in\texttt{T}_i}\ [\textsc{GT-Ann-Field}]$$

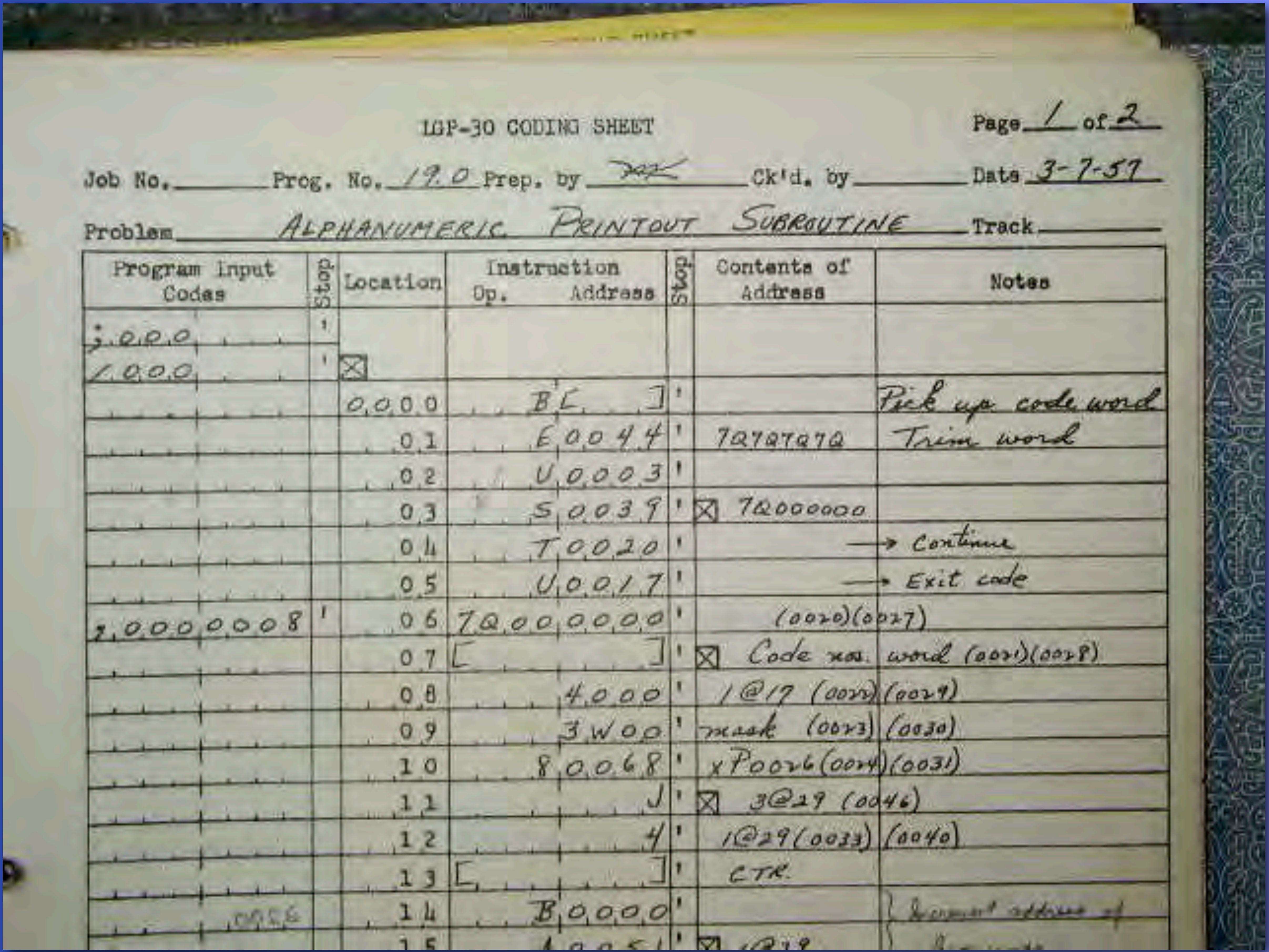
$$\frac{\begin{array}{c}\Phi;\Delta\vdash\overline{\texttt{T}}\ \mathrm{ok}\quad \Phi;\Delta;\Gamma\vdash\texttt{e}_0\in\texttt{T}_0\quad \Phi;\Delta;\Gamma\vdash\overline{\texttt{e}}\in\overline{\texttt{R}}\\ \mathit{mtype}(\texttt{m‹}\ \texttt{O})=\texttt{P.<}\overline{\texttt{X}}\ \texttt{extends}\ \overline{\texttt{N}}\ \texttt{with}\ \overline{\P\texttt{I}}\diamondsuit\ \texttt{S m}(\overline{\texttt{U}}\ \overline{\texttt{x}})\\ \Delta\vdash\overline{\texttt{T}}<:[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\texttt{N}}\quad \Phi\vdash\overline{\texttt{T}}\ \mathit{includes}\ [\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\P\texttt{I}}\diamondsuit\quad \Delta\vdash\overline{\texttt{R}}<:[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\overline{\texttt{U}}\end{array}}{\Phi;\Delta;\Gamma\vdash[\texttt{e}_0\circ\texttt{O}]\texttt{.m<}\overline{\texttt{T}}\texttt{>}(\overline{\texttt{e}})\in[\overline{\texttt{X}}\mapsto\overline{\texttt{T}}]\texttt{S}}\ [\textsc{GT-Ann-Invk}]$$

–Eric Allen, et al



–source code visualization

–Jackson Pollock



LGP-30 CODING SHEET

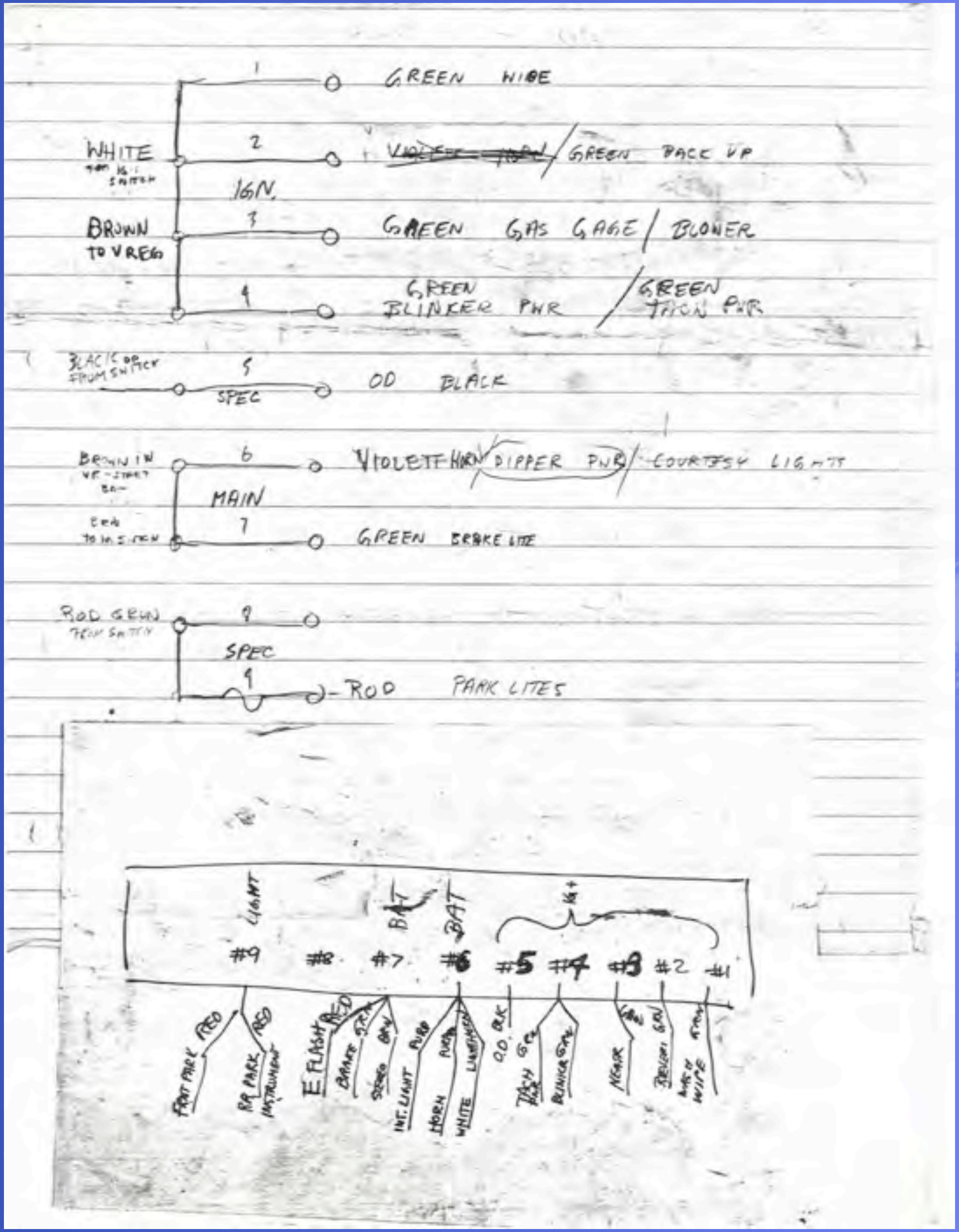
Page 1 of 2

Job No. ______ Prog. No. 19.0 Prep. by [illegible] Ck'd. by ______ Date 3-7-57

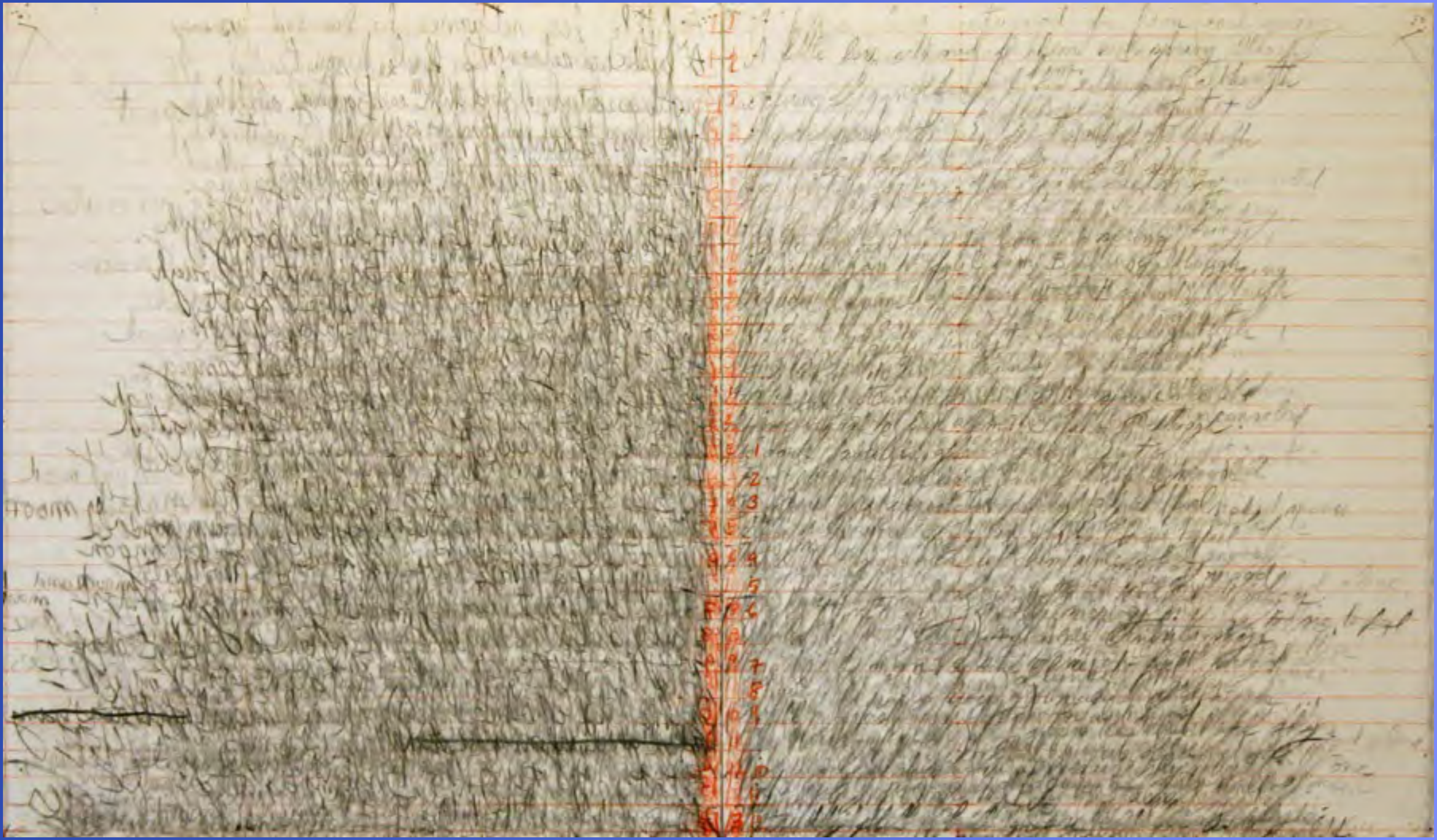
Problem ALPHANUMERIC PRINTOUT SUBROUTINE Track ______

Program Input Codes	Stop	Location	Instruction Op. Address	Stop	Contents of Address	Notes
;0000	'					
/0000	'	☒				
		0000	B[ ]	'		Pick up code word
		01	E0044	'	7Q7Q7Q7Q	Trim word
		02	U0003	'		
		03	S0039	'	☒ 7Q000000	
		04	T0020	'		→ Continue
		05	U0017	'		→ Exit code
;00000008	'	06	7Q000000	'	(0020)(0027)	
		07	[ ]	'	☒ Code nos.	word (0021)(0028)
		08	4000	'	1@17 (0022)	(0029)
		09	3W00	'	mask (0023)	(0030)
		10	80068	'	xP0026 (0024)	(0031)
		11	J	'	☒ 3@29 (00	46)
		12	4	'	1@29 (0033)	(0040)
		13	[ ]	'	CTR.	
0056		14	B0000	'		} Increment address of
		15	[illegible]	'	☒ 1@19	[illegible]

–coding sheet


1
GREEN WIPE
WHITE
FROM IG. SWITCH
2
~~VIOLET HORN~~ / GREEN BACK UP
IGN.
BROWN
TO V REG
3
GREEN GAS GAGE / BLOWER
4
GREEN BLINKER PWR / GREEN TACH PWR
BLACK OR BLK FROM SWITCH
5
SPEC
OD BLACK
BROWN IN
6
VIOLET HORN / DIPPER PWR / COURTESY LIGHTS
MAIN
RED TO IGN SWITCH
7
GREEN BRAKE LITE
ROD GRN FROM SWITCH
8
SPEC
9
ROD PARK LITES
LIGHT
BAT
BAT
IG+
#9
#8
#7
#6
#5
#4
#3
#2
#1
FRONT PARK RED
RED
RR PARK
INSTRUMENT
E FLASH RED
BRAKE GRN
STEREO
BRN
INT. LIGHT
PURP
HORN
PURP
WHITE
LAMINATION
OD BLK
TACH PWR
GRN
BLINKER GRN
HEATER
GRN
BLOWER GRN
WASH WIPE

–messy notes



–Sanda Illescu

```
void Step1() {
   selectDrawing(firstDrawing);
   if (isNewDrawing(firstDrawing))
      writePoem(firstDrawing);
   else
      rewriteLessPoem(firstDrawing);
   if (!oneLineLeft(firstDrawing))
      Step2(firstDrawing);
}

void Step2(int currentDrawing) {
   selectDrawing(++currentDrawing);
   if (isNewDrawing(currentDrawing)) {
      writePoem(currentDrawing);
      Step1();
   } else {
      rewriteLessPoem(currentDrawing);
      Step2(currentDrawing);
   }
}
```

–Kevin Sullivan, *Java implementation of Sanda's process*

Science: Traditional View

Scientists
create knowledge
study the world as it is
are trained in scientific method
use explicit knowledge
are thinkers

Engineering: Traditional View

Engineers

apply that knowledge

seek to change the world

are trained in engineering method

use tacit knowledge

are doers

Art: Traditional View

Artists
create artifacts
study the world as it appears
are trained in a creational skill
use aesthetics
are doers

The “Cleanliness” of Science

- The Demarcation Problem
- Positivism
- Elitist authoritarianism
- Epistemological
- Anarchism
- Inductivism
- Conventionalism
- Falsification
- Research programmes

Demarcation

Demarcation problem: When is one theory better than another, where science is at one end of the scale and pseudoscience at the other.

Positivism

Positivism: “the view that serious scientific inquiry should not search for ultimate causes deriving from some outside source but must confine itself to the study of relations existing between facts which are directly accessible to observation.”

Logical positivism: “Philosophy should provide strict criteria for judging sentences true, false, [or] meaningless.”

–Wikipedia

Elitist Authoritarianism

The idea that the demarcation problem is for a jury of accepted scientists to judge; that is, judge which work is scientific and which isn't.

Inductivism

- A statement is scientific only if it is provable from facts
- Need to go from a fact (ths is a computer) to a proposition ("this is a computer")
- Need to show that something true in a particular set of spaces and times (inverse square gravitation law) is true everywhere, all the time

Probabilism

- A statement is scientific only if it can be shown to be probable (and therefore one scientific theory is “better” than another if it is more probable).
- Because the universe is large, most probabilities are small.

Conventionalism

- The idea that a scientific theory is a convention or a means people agree to in order to make certain calculations.
- Ptolemaic crystal spheres provide an accurate predictive framework.
- Clumsy conventions (Ptolemy) are replaced by simpler ones (Copernicus)

Falsification

- A statement is scientific if an experiment can be stated that could falsify it.
- Suppose a statement requires a measurement made by an instrument to be falsified, then
  - there needs to be a theory of how the instrument works and what it measures (conventionalism)
  - if the instrument provides a reason to throw out the statement, then there needs to be a analysis of whether there are potential disturbing factors (*ceteris paribus*).

Falsification

At any given time, every theory has numerous "anomalies" that are considered either the result of "disturbing factors" or things to be cleared up later.

That is, every theory is heavily patched.

Research Programmes

- A research programme is a series of "progressive" theories; a theory is progressive (over the last) when:
  - it leads to new predictions
  - none of its bold predictions are falsified
  - it is not "patched" by ad hoc statements, but instead retains its hard core and adjusts its protective belt

Epistemological Anarchism

- There are no demarcation lines.
- There is no such thing as progress, only fashion.
- Anything goes.
- Science as propaganda.

Bruno Latour

- Scientific “fact” is determined by a jury.
- Science is the set of things constructed in a laboratory.
- Scientists “vote” on what’s good science through references.
- And lots of things I don’t understand.

Art & Engineering Precede Science

...our testaments to physical work are so often focused on the values such work exhibits rather than on the thought it requires. It is a subtle but pervasive omission....It is as though in our cultural iconography we are given the muscled arm, sleeve rolled tight against biceps, but no thought bright behind eyes, no image that links hand and brain.

–Mike Rose, *The Mind at Work*

Art & Engineering Precede Science

Skilled manual labor entails a systematic encounter with the material world, precisely the kind of encounter that gives rise to natural science. From its earliest practice, craft knowledge gas entailed knowledg of the "ways" of one's materials—that is, knowledge of their nature, acquired through disciplined perception.

–Matthew B. Crawford, *Shop Class as Soulcraft*

Art & Engineering Precede Science

...in areas of well-development craft practices, technological developments typically preceded and gave rise to advances in scientific understanding, not vice versa.

–Matthew B. Crawford, *Shop Class as Soulcraft*

Art & Engineering Precede Science

The steam engine is a good example. It was developed by mechanics who observed the relations between volume, pressure, and temperature. This was at a time when theoretical scientists were tied to the caloric theory of heat, which later turned out to be a conceptual dead end.

–Matthew B. Crawford, *Shop Class as Soulcraft*

Science & Engineering: Realistic View

Scientists	Engineers
create knowledge	create knowledge
are problem driven	are problem driven
seek to understand and explain	seek to understand and explain
design experiments to test theories	design devices to test theories
prefer abstract knowledge	prefer contingent knowledge
but rely on tacit knowledge	but rely on tacit knowledge

Discovery

In the scientist's house are many mansions....Outsiders often regard science as a sober enterprise, but we who are inside see it as the most romantic of all callings. Both views are right. The romance adheres to the processes of scientific discovery, the sobriety to the responsibility for verification.

Histories of science put the spotlight on discovery....The story of scientific progress reaches its periodic climaxes at the moments of discovery....In the philosophy of science, all the emphasis is on verification, on how we can tell the true gold of scientific law from the fool's gold of untested fantasy. In fact. it is still the majority view among philosophers of science that only verification is a proper subject of inquiry, that nothing of philosophical interest can be said about the process of discovery.

–Langley, Simon, Bradshaw, and Zytkow, *The Mind at Work*

Discovery

...But we believe that science is also poetry, and—perhaps even more heretical—that discovery has its reasons, as poetry does. However romantic and heroic we find the moment of discovery, we cannot believe either that the events leading up to that moment are entirely random and chaotic or that they require genius that can be understood only by congenial minds. We believe that finding order in the world must itself be a process impregnated with purpose and reason.

Art as Exploration & Discovery

- Exploration: “some combination of premeditated searching and undisciplined, perhaps only partly conscious, rambling.” Defocused attention; flat associative hierarchies.
- Exploration is assertive action in the face of uncertain assumptions, often involving false starts, missteps, and surprises.
- Discovery: “a reckless encounter with the unexpected.” Focus; concentration; persistence. “If we persist [in exploration], we discover.”

–Peter Turchi, *Maps of the Imagination*

- Understand: Only after discovery can the work be properly structured, can the selection and organization of the significant moments of time take place.

If we attempt to map the world of a story before we explore it, we are likely either to (a) prematurely limit our exploration, so as to reduce the amount of material we need to consider, or (b) explore at length but, recognizing the impossibility of taking note of everything, and having no sound basis for choosing what to include, arbitrarily omit entire realms of information. The opportunities are overwhelming.

–Peter Turchi, *Maps of the Imagination*

Artistic Creation

Artistic creation is a voyage into the unknown. In our own eyes, we are off the map. The excitement of potential discovery is accompanied by anxiety, despair, caution, perhaps, perhaps boldness, and, always, the risk of failure. Failure can take the form of our becoming hopelessly lost, or pointlessly lost, or not finding what we came for (though that last is sometimes happily accompanied by the discovery of something we didn't anticipate, couldn't even imagine before we found it). We strike out for what we believe to be uncharted waters, only to find ourselves sailing in someone else's bathtub. Those are the days it seems there is nothing new to discover but the limitations of our own experience and understanding.

–Peter Turchi, *Maps of the Imagination*

Artistic Creation

Some of the oldest stories we know, including creation myths, were attempts to make sense of the world. Those early storytellers invented answers to the mysteries all around them. Why does the rain come? Why does it stop? If a child is created by two adults, from where did the first two adults originate? What is the earth like beyond what we have seen, and beyond what the people we know have seen? What lies beyond the stars?

–Peter Turchi, *Maps of the Imagination*

Artistic Creation

Sometimes it's very tempting to be satisfied with what's easy, particularly if people tell you it's good....What's essential is to work without any preconception whatever, without knowing in advance what the picture is going to look like....It is very, very important to avoid all preconception, to try to see only what exists...to translate one's sensation.

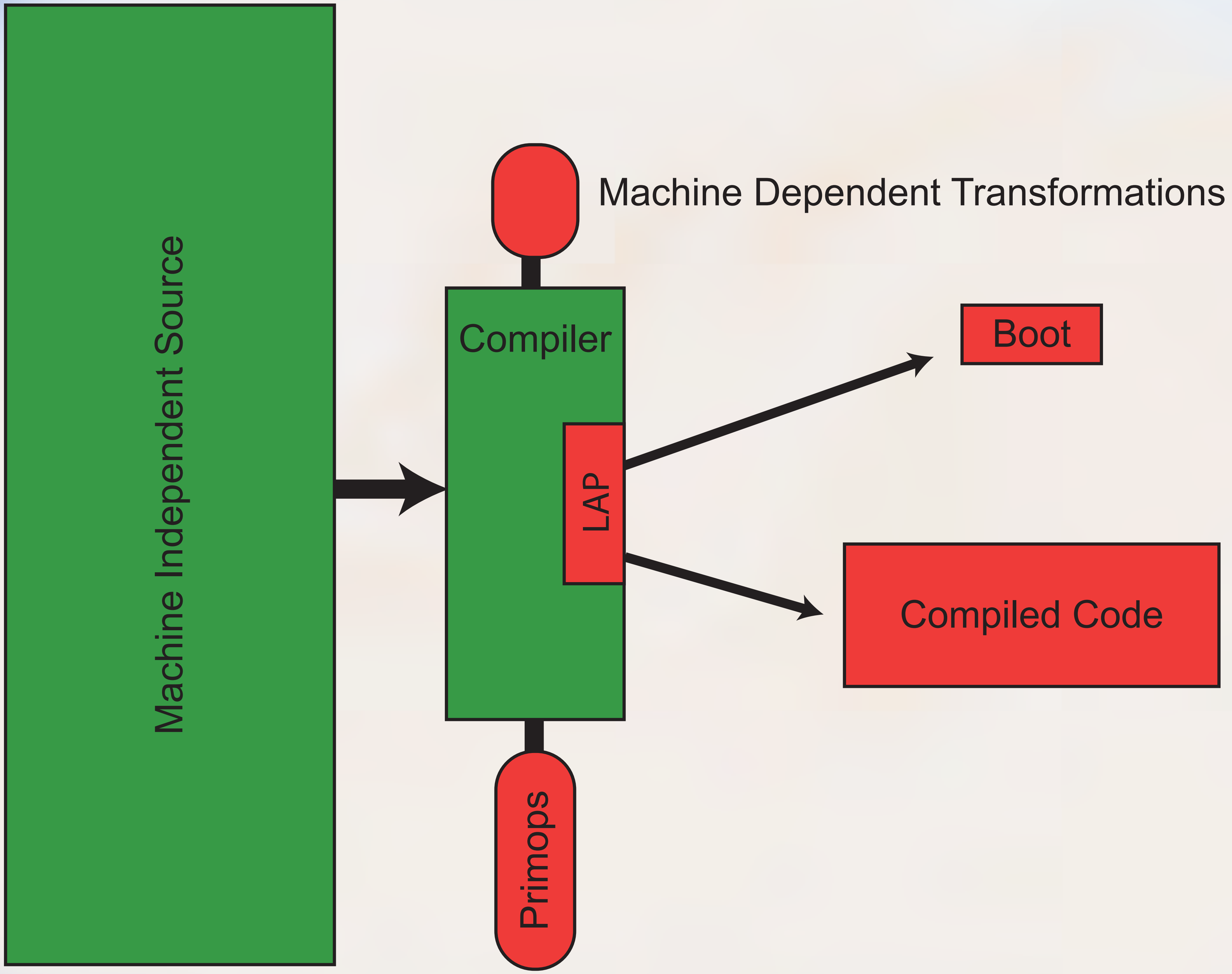
–Alberto Giacometti, on painting James Lord

Art:
Realistic View

Artists
create knowledge, language, alternative reality
are image or “piece” driven
seek to understand and explain
design devices to explore the world and the self
prefer contingent knowledge
but rely on tacit knowledge

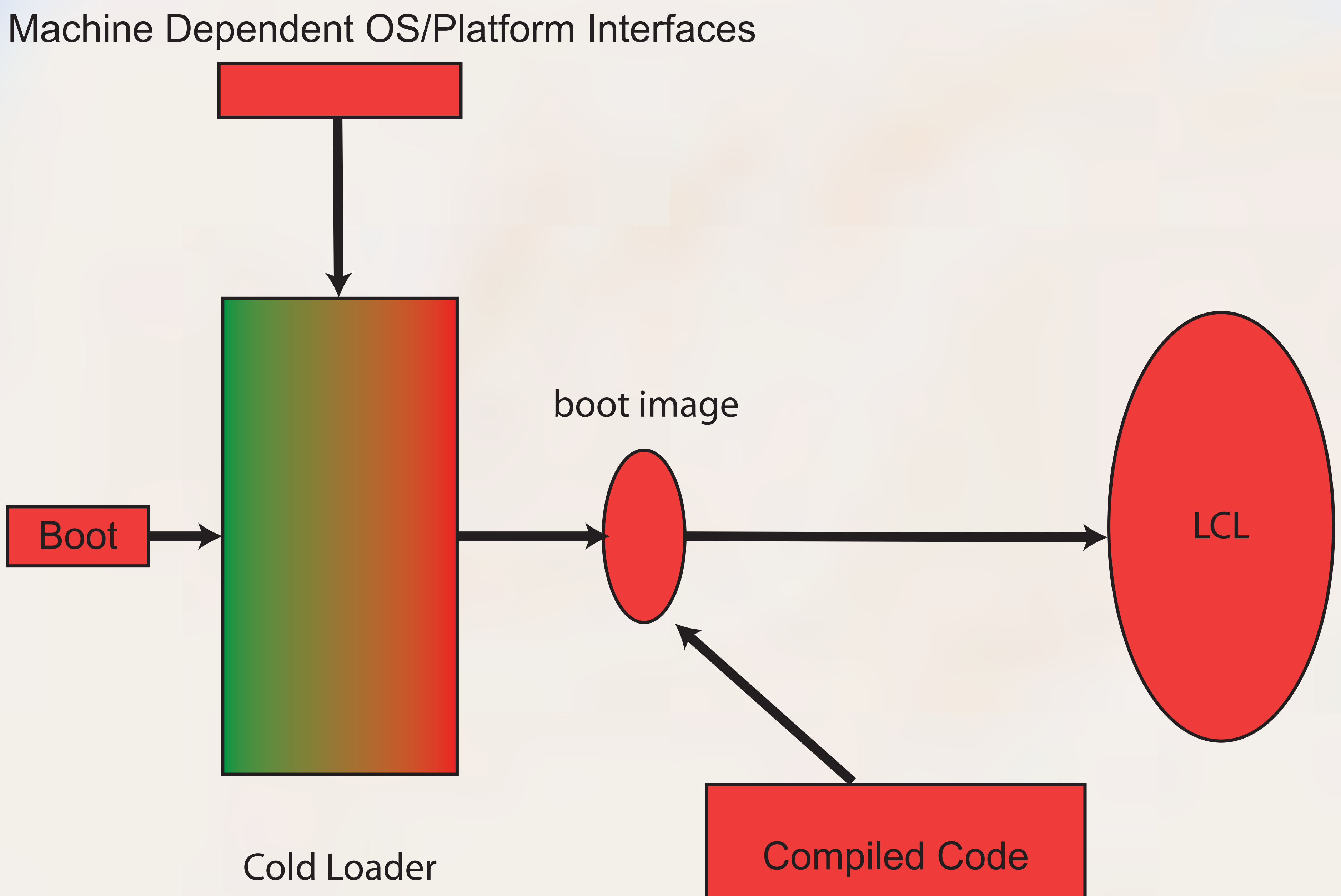
My Jaunt into SPLs: 1984

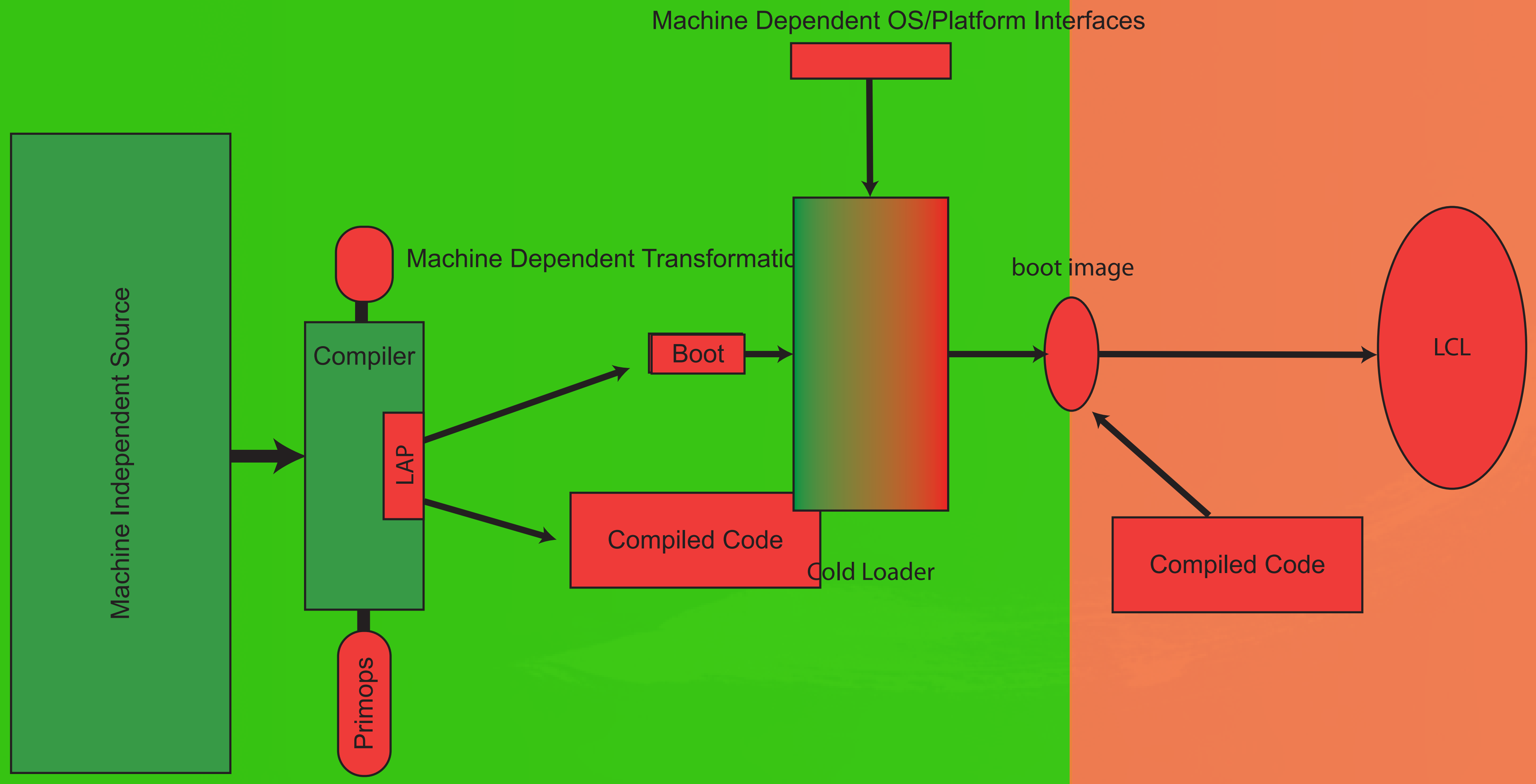
- Lucid Common Lisp
- Extensive runtime including
  - memory management (ephemeral GC)
  - threading / scheduling
  - graphics & window system (!)
  - programming environment including editors, debuggers and tracing (á la AOP)
  - foreign function interfaces (C, C++, Fortran, Cobol)
- 30 different platforms (including OSs)
  - mainframes, workstations, minicomputers


Machine Independent Source
Machine Dependent Transformations
Compiler
LAP
Boot
Compiled Code
Primops

```
(defun car (x) (car x))
```

primops includes operands that enable low-level things, like the garbage collector, to be written in Lisp


Machine Dependent OS/Platform Interfaces
boot image
Boot
LCL
Compiled Code
Cold Loader


Machine Dependent OS/Platform Interfaces
Machine Independent Source
Machine Dependent Transformatio
Compiler
LAP
Primops
Boot
boot image
LCL
Compiled Code
Cold Loader
Compiled Code

Development Platform

Target Platform

Dynamically Retargetable Compiler

```
(compile-file "foo" :target 'prime)
(compile-file "foo" :target 'sun3)
(compile-file "foo" :target 'apollo)
```

Target Model

- whether or not there is a hardware stack pointer and a push instruction
- whether data movement can be memory /memory, or is limited to memory/register (as in RISC machines)
- how registers are grouped into classes (such as ‘address’ and ‘data’)
- how the component slots of a stack frame are accessed
- how other locations are accessed, including local variables, special value cells of dynamic variables, and other stack frame slots
- what side effects the instructions and pseudo-instructions have.

Target Model

- the primary tagging scheme—where do the tags go?
- the stack frame layout—which slots hold machine-dependent overhead, which hold local variables, what information is passed during function call?
- which primitive operations will be open-coded into native machine instructions, and which will be coded as machine language subroutines?

Compiler Machines

- constant values
- compiler macros
- primops and function descriptors
- opcode-macros
- operand-decoders
- register-descriptors
- LAP-methods

Hierarchy of Compiler Machines

$$\begin{tikzcd}
 & \text{CommonMachine} \arrow[dl] \arrow[d] \arrow[dr] & \\
\text{MC68000} \arrow[d] & \text{SPARC} & 370 \\
\text{MC68020} & &
\end{tikzcd}$$

Lisp in Lisp

- about 500,000 lines of code
- 4% is machine dependent
- porting to a novel machine / OS combination takes about a month
- 65% of the compiler is machine independent
  - 5% for tags
  - 30% for CPU

Invented Languages

The 900 languages, over 900 years, we do have evidence for [sic] suggest that the urge to invent languages is as old and persistent as language itself.

–Arika Okrent, *In the Land of Invented Languages*

Invented Languages

It is at least as old and persistent as the urge to complain about language. The primary motivation for inventing a new language has been to improve upon natural language, to eliminate its design flaws, or rather the flaws it has developed for lack of conscious design. Looked at from an engineering perspective, language ***is*** *a kind of disaster.*

–Arika Okrent, *In the Land of Invented Languages*

Advantages of Natural Languages

*...natural language and the messy qualities that give it so much flexibility and power, and that make it so much more than a simple communication device. The ambiguity and lack of precision allow it to serve as an instrument of thought **formulation**, of experimentation and discovery. We don't have to know exactly what we mean before we speak; we can figure it out as we go along. Or not.*

–Arika Okrent, *In the Land of Invented Languages*

Advantages of Natural Languages

*At the same time natural language still works as an instrument of thought transmission, one that can be **made** extremely precise and reliable when we need it to be, or left loose and sloppy when we can't spare the time or effort.*

–Arika Okrent, *In the Land of Invented Languages*

```
void foo(int x[], int y, int z)
{
 if (z > y + 1)
 {
 int a = x[y], b = y + 1, c = z;
 while (b < c)
 {
 if (x[b] <= a) b++; else {
 int d = x[b]; x[b] = x[--c];
 x[c] = d;
 }
 }
 int e = x[--b]; x[b] = x[y];
 x[y] = e; foo(x, y, b);
 foo(x, c, z);
}
```

and this does what?

```
void quicksort(int array[], int begin, int end)
{
 if (end > begin + 1)
 {
 int pivot = array[begin],
 l = begin + 1, r = end;
 while (l < r)
 {
 if (array[l] <= pivot)
 l++;
 else
 swap(&array[l], &array[--r]);
 }
 swap(&array[--l], &array[beg]);
 sort(array, begin, l);
 sort(array, r, end);
 }
}
```

add some art

```
better !pout !cry
better watchout
lpr why
santa claus <north pole >town

cat /etc/passwd >list
ncheck list
ncheck list
cat list | grep naughty >nogiftlist
cat list | grep nice >giftlist
santa claus <north pole > town

who | grep sleeping
who | grep awake
who | egrep 'bad|good'
for (goodness sake) {be good}
```

add a lot of art

Stopping by http://babelfish.altavista.com on a Snowy Evening

Here is a task whose outcome is certain:
Thinking of someone’s forest
and then thinking whether this forest is that someone’s.
And as for his house (I’ve picked this up):
it is certainly located in town.

I am stopped here paying attention to the snow above,
observing the trees filling in above the snow.
My eye finds comfort in this.

As for my horse, he strangely and narrowly stops.
I am small, me and the small end of the tree both agree.
To the horse, we are stopped between a farm and the frozen sea.
This evening is the strangest and the darkest of the year, the horse must think.

His harness bells are his only user interface.
These bells are installed to a flange by some wiring, and so
he gives the flange a shock, vibrating the wires,
thereby jolting the bells (giving them a restlessness)
in order to pose me a question:
Is there some kind of mistake here?
Surely a certain error exists.
He is a small horse.

There is only one other sound,
a different sound like a clay tone,
but only to the extent of a thin layer or a languid ribbon
forming a closed loop: the sweepback of a light breeze
over downy soft flakes—a simple, easy wind;
flakes like cotton wool or hair
or a rag for cleaning, which is the same thing.
Or maybe it sounds like this:
khlop!

(I am excited by this.)
Woods are attractive. Likable. Lovable, even.
Or sometimes—obscure. One of the trees
is dark and from a place which is deep.
And you know what they say: Dark and deep are deep.

But I am held to obligations which I must maintain.
Before I sleep I must resume my outward journey.
(And other unspecified things of the same class.)

–written with the use of translation-based defamiliarization

Stopping By Woods on a Snowy Evening

Whose woods these are I think I know.
His house is in the village though;
He will not see me stopping here
To watch his woods fill up with snow.

My little horse must think it queer
To stop without a farmhouse near
Between the woods and frozen lake
The darkest evening of the year.

He gives his harness bells a shake
To ask if there is some mistake.
The only other sound's the sweep
Of easy wind and downy flake.

The woods are lovely, dark and deep.
But I have promises to keep,
And miles to go before I sleep,
And miles to go before I sleep.

–Robert Frost, *New Hampshire*

Ark!!!

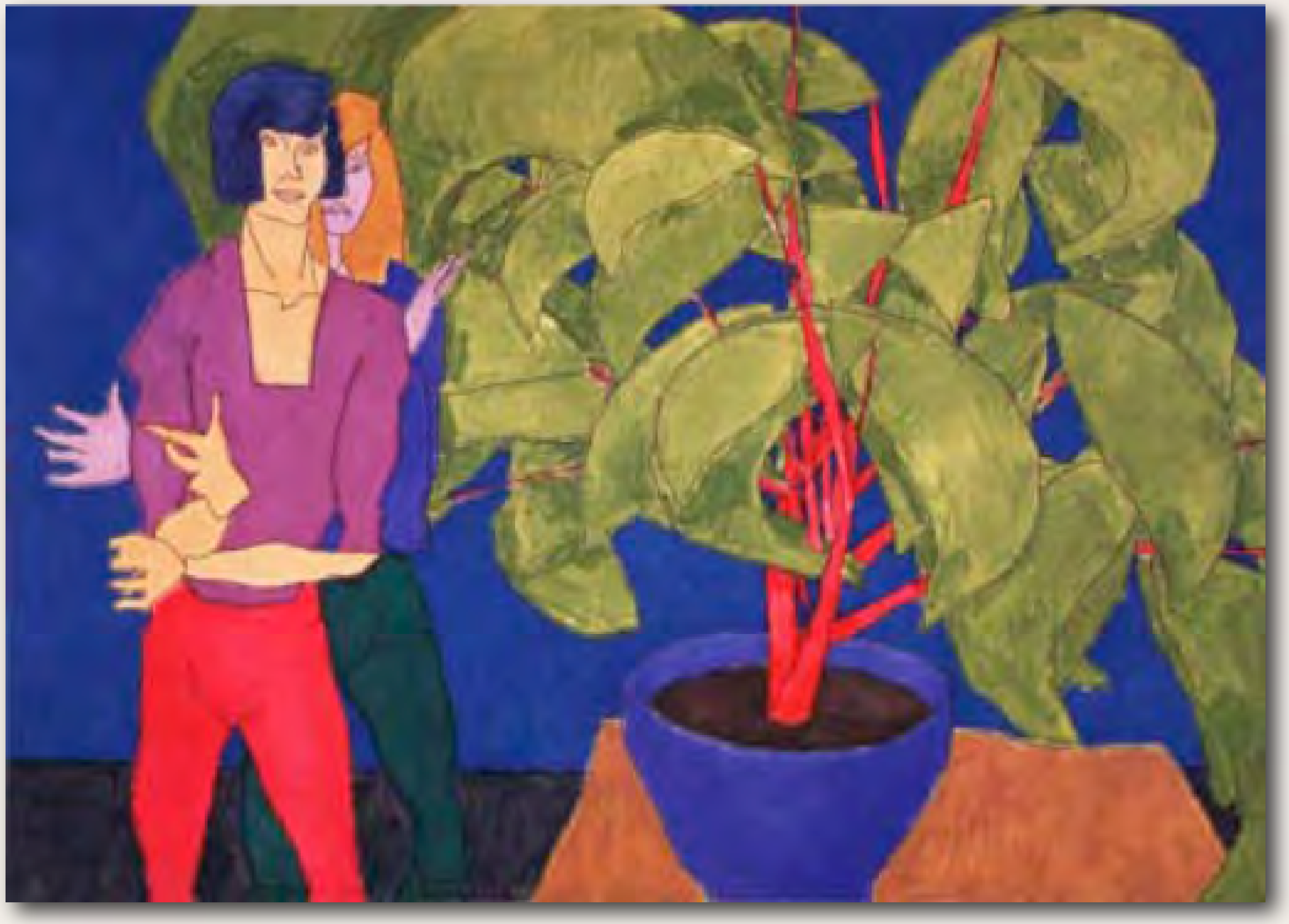
Shem raised the rope
and forced his bulky frame through
chapped his knees when they allowed
a braying ass through the holy gates
but it’s holy shit fire downtown

his oiled black hair glistened in the sun
as the ass was led around the path
toward the pinnacle of the secret
of the hiding place of the Holy Ark

Noah bitched and moaned about the count
so Yaphet used the pinnacle of a technical
split legged capture bomb (holy fuck)
this kicked all kinds of ass

Ham completely destroyed his bitch ass

–a sought poem or *flarf*



Two Friends with Potted Plant, 1991, Aaron

Experimental vs Conceptual

- This is a distinction like that between artistic and scientific
- Or iterative versus waterfall

Experimental Artists

Artists who have produced experimental innovations have been motivated by aesthetic criteria: they have aimed at presenting visual perceptions. Their goals are imprecise, so their procedure is tentative and incremental. The imprecision of their goals means that these artists rarely feel they have succeeded, and their careers are consequently often dominated by the pursuit of a single objective.

–David W. Galenson, *Old Masters and Young Geniuses*

Conceptual Artists

In contrast, artists who have made conceptual innovations have been motivated by the desire to communicate specific ideas or emotions. Their goals for a particular work can usually be stated precisely, before its production, either as a desired image or a desired process for the work's execution. Conceptual artists consequently make detailed preparatory sketches of plans for their paintings.

–David W. Galenson, *Old Masters and Young Geniuses*

Experimental Artists

For experimental artists, planning a painting is unimportant. The subject selected might be simply a convenient object of study, and frequently the artist returns to work on a motif he has used in the past. Some experimental painters begin without a specific subject in mind, preferring instead to let the subject emerge as they work. Experimental painters rarely make elaborate preparatory sketches. Their most important decisions are made during the working stage. The artist typically alternates between applying paint and examining the emerging image; at each point, how he develops the image depends on his reaction to what he sees.

–David W. Galenson, *Old Masters and Young Geniuses*

Conceptual Artists

For conceptual artists, planning is the most important stage. Before he begins working, the conceptual artist wants to have a clear vision either of the completed work or of the process that will produce it. Conceptual artists consequently often make detailed preparatory sketches or other plans for a painting. With the difficult decisions already made in the planning stage, working and stopping are straightforward. The artist executes the plan and stops when he has completed it.

–David W. Galenson, *Old Masters and Young Geniuses*

Conceptual Artists

...extreme practitioners...make all the decisions for a work before beginning it. It is unclear, however, if this is literally possible. There are artists who came close to it, and perhaps achieved it, during the 1960s, by making plans for their work and having these plans executed by others.

–David W. Galenson, *Old Masters and Young Geniuses*

Professional vs Compulsive Programmers

- Joseph Weizenbaum
- *Computer Power and Human Reason*
- 1976
- Eliza
- Weizenbaum hated Stallman

The ordinary [professional] programmer will...generally do lengthy preparatory work, such as writing and flow diagramming, before beginning work with the computer itself. His sessions with the computer may be comparatively short. He may even let others do the actual console work. He develops his program slowly and systematically. When something doesn't work, he may spend considerable time away from the computer framing careful hypotheses to account for the malfunction and designing crucial experiments to test them....When he has finally composed the program he set out to produce, he is able to complete a sensible description of it and turn his attention to other things.

–Jospeh Weizenbaum, *Computer Power and Human Reason*

The compulsive programmer is usually a superb technician,...one how knows every detail of the computer he works on, its peripheral equipment, the computer's operating system, etc....He can write small subsystem programs quickly, that is, in one or two sessions of, say, 20 hours each....His main interest...is in very large, very ambitious systems of programs.... [T]he systems he undertakes to build have very grandiose but extremely imprecisely stated goals. Some examples...are: new computer languages to facilitate man-machine communication; a general system that can be taught to play any board game; a system to make it easier for computer experts to write super-systems.

–Jospeh Weizenbaum, *Computer Power and Human Reason*

The compulsive programmer...calls what he does "hacking."...He cannot set before himself a clearly defined long-term goal and a plan for achieving it, for he has only technique, not knowledge. He has nothing he can analyze or synthesize; in short, he has nothing to form theories about....

(It has to be said that not all hackers are pathologically compulsive programmers. Indeed, were it not for the often, in its own terms, highly creative labor of people who proudly claim the title "hacker," few of today's sophisticated computer time-sharing systems, computer language translators, computer graphics systems, etc., would exist.)

–Jospeh Weizenbaum, *Computer Power and Human Reason*

	Theory	Experiment
Science	mathematicians Einstein waterfall	Craftspeople Darwin Engineers complexity science
Art	Warhol Picasso	1970s hackers Cézanne agile?

more dimensions?

	Theory	Experiment
Science		
Art	Software Project	

more dimensions?

“Intellectuals” cannot observe art in action

If you give a somewhat complex knot to an adult who has developed all the relevant cognitive skills and discipline, and ask them to undo it, they don’t jump into action immediately. First they hold it carefully, turn it this way and that, looking at the knot from each side, until they understand the structure of the thing itself. Then a plan begins to form: “if I loosen this strand, it will release that one, and then I’ll be able to pass this through that loop,” and so on. If they’ve understood the structure correctly, and formed the plan based on that understanding, then the knot falls apart, and is no more.

–Malcolm Sparrow, being wrong while talking about *The Character of Harms: Operational Challenges in Control*

“Intellectuals” cannot observe art in action

By contrast, if you give the same knot to a child who hasn’t developed the same skills or mental habits, watch what they do. They jump into action immediately. They don’t pay such close attention to the structure of the thing. They pull and tug and probably make things worse.

–Malcolm Sparrow, being wrong while talking about *The Character of Harms: Operational Challenges in Control*

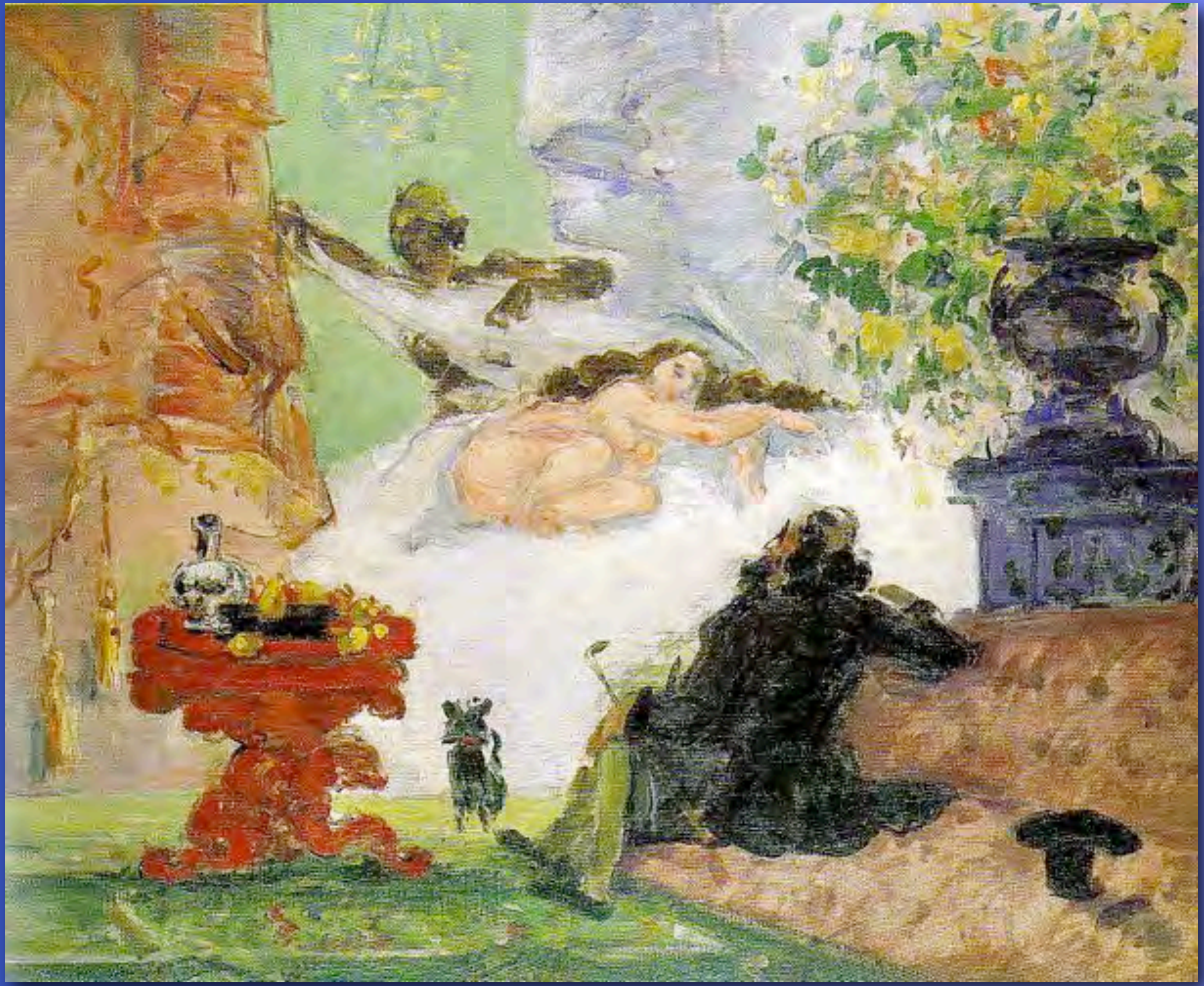
How Knots are Really Untied

Are you looking for an easier way to untie jammed-up knots? Most people know the standard method of looking for “ears” or “collars” of rope in the knot to shift. What if you don't have that option, or it just isn’t working? One other method is to quickly and firmly twist the parts of the rope just outside the knot back and forth as you push in slack.

How Knots are Really Untied

- *You don’t sit there analyzing the problem. You start playing with it to see if there is an easy way.*
- *You begin to push back on the tight areas to loosen their hold on the rope.*
- *You feel for what is moving, what is releasing.*
- *You make mistakes. One unwinding may only create another knot.*
- *You persist.*
- *You keep pushing and pulling until you get a release.*
- *You use one release to get the next release.*
- *You continue until the knot is completely released.*

Paul Cezanne



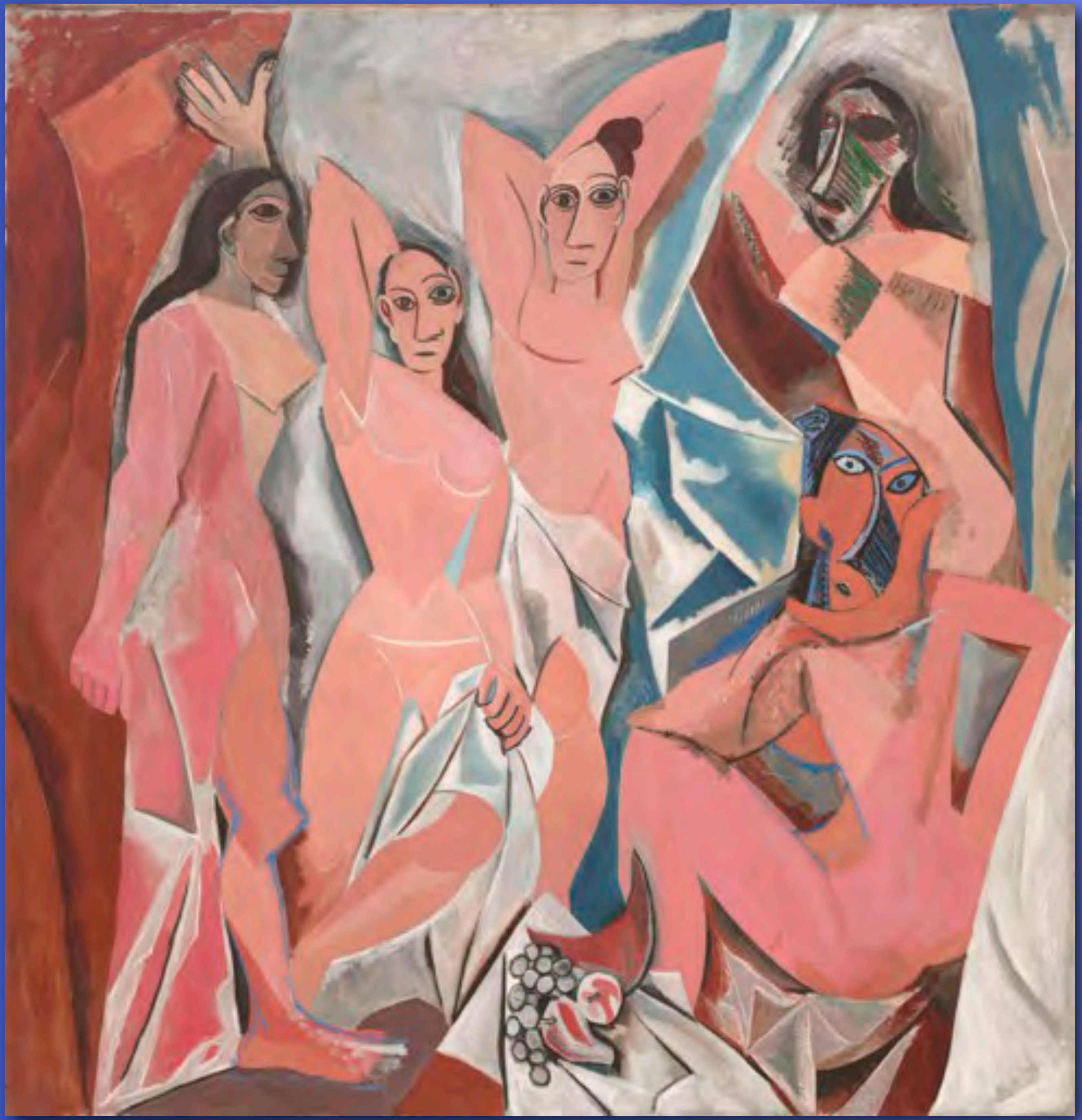
A Modern Olympia

Paul Cézanne

For him as I understand his work, the ultimate synthesis of a design was never revealed in a flash; rather he approached it with infinite precautions, stalking it, as it were, now from one point of view, now from another....For him the synthesis was an asymptote toward which he was forever approaching without ever quite reaching it; it was a reality, incapable of complete realization.

–Roger Fry, *Cézanne*

Pablo Picasso


Les Demoiselles d’Avignon

Pablo Picasso

I can hardly understand the importance given to the word "research" in connection with modern painting. To find, is the thing....

When I paint my object is to show what I have found, not what I am looking for....

I have never made trials or experiments. Whenever I had something to say, I have said it in the manner in which I have felt it ought to be said.

–Picasso, 1923 *interview*

Pablo Picasso cannot observe art in action

*During the winter 1906–7, he filled a series of sketchbooks with the preparatory studies for Les Demoiselles d'Avignon.... Historian William Rubin estimated that Picasso made more than 400 studies for the Demoiselles, "a quantity of preparatory work...without parallel, for a single picture, in the entire history of art." **

–David W. Galenson, *Old Masters and Young Geniuses*

"I have never made trials or experiments."

* Rubin, Seckel, & Cousins, *Les Demoiselles d'Avignon*

Scientific Processes

Scientific processes, whether for product development, manufacturing, or service delivery, seek to minimize variances in task execution through the imposition of tight constraints, measurement systems, and feedback control.

–Gabriel, Griswold, Notkin, & Sullivan, *Hybrid Art-Science Software Development Processes*, an unpublished (and unpublishable) manauscript

Scientific Processes

Thousands of manufacturing companies have achieved tremendous improvements in quality and efficiency by copying the Toyota Production System, which combines rigorous work standardization with approaches such as just-in-time delivery of components and the use of visual controls to highlight deviations. Process standardization also has permeated nearly every service industry, generating impressive gains.

–Robert M. Hall & M. Eric Johnson,
When Should a Process be Art, Not Science?

Artistic Processes

An artistic process is one that tolerates or even welcomes a variety of inputs and works to produce the best (or at least an acceptable) result given the inputs and the situations encountered while executing the process. In particular, an artistic process can adapt to very poorly stated requirements or even no requirements at all aside from the production of something of value. Artistic processes can endeavor to invent and blaze new trails. In many cases the value of the outcome of an artistic process will be determined only after the product has been produced, though it's often possible to make some judgments along the way.

–Gabriel, Griswold, Notkin, & Sullivan, *Hybrid Art-Science Software Development Processes*, an unpublished (and unpublishable) manauscript

Scientific Processes

The idea that some processes should be allowed to vary flies in the face of the century-old movement toward standardization. Process standardization is taught to MBAs, embedded in Six Sigma programs, and practiced by managers and consultants worldwide.

–Robert M. Hall & M. Eric Johnson,
When Should a Process be Art, Not Science?

Manufacturing Metaphor

The process regime was born of the software crisis at a time when even large software systems were built one line of code at a time. With some logic it established roles and behaviors rooted in a manufacturing metaphor, where software processes are analogous to manufacturing processes, programmers are analogous to assembly-line workers, and the ultimate product is lines of code.

–Wallnau, Hissam, & Seacord, *Building Systems from Commercial Components*

Programmers’ Cycle

If we look closely at how software developers spend their time, we see that they do these things in sequence: {analyze–code–build–test}. First they figure out how they are going to address a particular problem, then they write code, then they do a build and run the code to see if it indeed solves the problem, and finally, they repeat the cycle. Many times. This is how programmers work.

–Mary Poppendieck, *Lean Design,*
http://www.poppendieck.com/design.htm

Programmers' Cycle

An interesting thing about software development is that this cycle: {analyze–code–build–test}, occurs both in the large and in the small. Every large section of software will pass through this cycle (many times), but so will every small section of code. A developer may go through these steps several times a day, or even, many times per hour. Generally there is no particular effort, nor any good reason, to get the code exactly right the first time. Try it, test it, fix it is a far more efficient approach to programming than perfection in the first draft.

–Mary Poppendieck, *Lean Design,*
http://www.poppendieck.com/design.htm

Programmers' Cycle

The reason a manufacturing metaphor does not work for development is because development is not sequential, it is a cycle of discovery. The {analyze–code–build–test} cycle is meant to be repeated, not to happen only once. Further, as ideas and information move through this cycle, two things must be assured. First, information must not be lost through handoffs, and second, feedback from the cycle must be as short as possible.

–Mary Poppendieck, *Lean Design,*
http://www.poppendieck.com/design.htm

Science is not Enough

On the Creation of Software