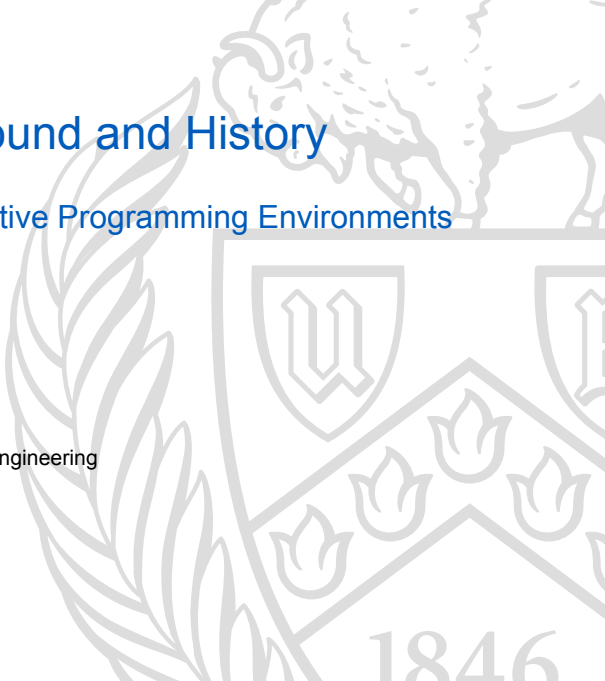
# Smalltalk: Background and History

## CSE 410/510 ETH: Interactive Programming Environments

Ethan Blanton

Department of Computer Science and Engineering

University at Buffalo



# The KiddiComp

Alan Kay proposed the KiddiComp while working on his Ph.D.

It focused on personal computing.
(Every person with their own computer, for their own needs.)

Kay believed that this vision had to start in childhood.

It was to be a computer usable by children.

When he moved to PARC, it became the DynaBook. [4]

# Contemporary Computing

The journey from KiddiComp to DynaBook was about 1968–1972.

At this time, computers were huge – think refrigerator.

The computer probably had hundreds of KB of RAM.
(A 1-bit megapixel display uses 128 kB.)

It ran at speeds in 100s of kHz or single-digit MHz.

It cost as much as a car or small house.

This was visionary work, not incremental.

# The Vision

Kay envisioned a connected, graphical computer.

Users would download information and software.

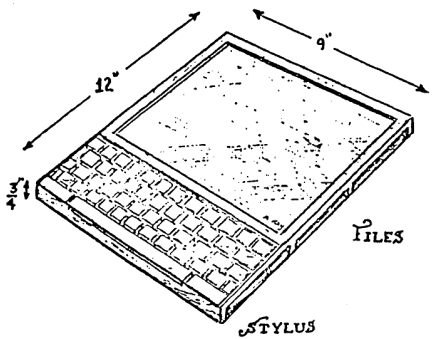
Software could communicate (*e.g.*, multiplayer games).

Everything was open to the user-as-programmer:
The user was expected to modify their environment.

He believed that children would take to this naturally.

He believed that adults would use it powerfully.

# The DynaBook




The DynaBook as rendered by Kay in “A Personal Computer...” [3]

# Limitations

Kay wanted:

- $500
- No more than 4 lbs.
- 1 MB of storage
- 300 kbps data transfer
- 4000 print quality characters on-screen
- Keyboard input

None of these but the keyboard were *quite* possible in 1972.
(At least, not together!)

# Smalltalk

*[W]hat remains [after eliminating large and "traditionally" extensible languages] is a chance to present to a user a very simpleminded language (which reveals the true state of programming semantics) that, nonetheless, is capable of a wide variety of expression. – Kay, "A Personal Computer ..." [3]*

*The name [was] a recation against the "IndoEuropean god theory" where systems were named Zeus, Odin, and Thor, and hardly did anything. I figured that "Smalltalk" was so innocuous a label that if it ever did anything nice people would be pleasantly surprised. – Kay, "The Early History of Smalltalk" [4]*

# Foundations

Early Smalltalk grew from Kay's interest in message passing.

He declared that the language had to fit on one sheet of paper.

In about 1972, he delivered that. [4]

His inspiration for this was McCarthy's Metacircular Evaluator. [6]

# Early Smalltalk

Kay credits Dan Ingalls and Adele Goldberg for driving early Smalltalk development.

Ingalls prototyped Smalltalk-72 in BASIC. [1]

This was somewhat of a surprise to Kay. [4]

He then implemented it on a Data General Nova assembly.

He credits his relative lack of language experience as valuable.
(But is quick to point out Kay's large experience!)

Once the interpreter was working, Diana Merry and Ted Kaehler did graphics.

# The Alto

Smalltalk-72 was rapidly ported to the Xerox Alto:

- 5 MHz microcode clock / 1 MHz memory clock [1]
- 606x808 1-bit display
- 2.5 MB removable hard disk pack
- 128 kB RAM
- Keyboard
- Mouse

This was initially in Nova emulation.

Later, some VM operations were moved into microcode.

# An Alto running Smalltalk

Photo by Ken Shirriff, https://www.righto.com/

# Early Smalltalk (cont'd)

Smalltalk-72 was influenced by Seymour Papert's Logo.[1, 7]

It used "to" to introduce functions, and had turtle drawing.

It also (unlike logo) used a variety of graphical characters:

- a pointing hand for quote (equivalent to lisp quote)
- ☺ for the turtle object
- an open eye to peek the input stream
- various arrows

It had functions as well as methods.

# The Turtle

```
to turtle var : pen ink width dir xor x y frame : f (
    CODE 21 'ϕgo⇒(draw a line of length :. ⇑SELF)
            ϕturn⇒(turn right :. ⇑SELF)
            ϕgoto⇒(draw a line to :x :y. ⇑SELF)'
    ϕpendn⇒(☞pen ← 1. ⇑SELF)
    ϕpenup⇒(☞pen ← 0. ⇑SELF)
    ϕink⇒(ϕ←. :ink. ⇑SELF)
    ϕwidth⇒(ϕ←. :width. ⇑SELF)
    ϕxor⇒(☞xor ← (ϕoff⇒(0) 1). ⇑SELF)
    ϕis⇒(ISIT eval)
    ϕhome⇒(☞x ← frame frmwd/2.
            ☞y ← frame frmht/2.
            ☞xf ← ☞yf ← 0. ☞dir←270. ⇑SELF)
    ϕerase⇒(frame fclear. ⇑SELF)
    ϕup⇒(☞dir ← 270. ⇑SELF)
    isnew⇒(☞ink ← ☞black. ☞pen ← ☞width ← 1. ☞xor ← 0.
            (ϕframe⇒(☞frame ← :) ☞frame ← f)
            ϕat⇒(:x. :y. ☞dir←270) SELF home) )
```

# Bytecode Interpretation

Smalltalk-72 used a partial bytecode interpreter.

Primitive operations (like arithmetic) invoked bytecodes.

Message sending invoked a bytecode.

This allowed some bytecodes to be implemented in microcode.

Code interpretation was in general textual.

# Smalltalk-76

Dan Ingalls rewrote Smalltalk, resulting in Smalltalk-76. [4]

It was simulated on, and then replaced, Smalltalk-74.

It moved to a full bytecode representation.

This full bytecode machine was very portable.

Almost all internal structures became message-passing objects.

Functions went away.

Keyword syntax was formally defined. [1]

# BitBlt

Dan Ingalls and Diana Merry introduced the BitBlt operator. [2]

It was added between Smalltalk-74 and Smalltalk-76.

BitBlt has several powerful features:

- It provides several graphical transforms
- It operates on a fixed-size window of data
- It can operate on an arbitrary bit-position

An arbitrary-size Conway's Game of Life board can be evolved using 60 BitBlt transforms.

It was used on many systems in the 80s and 90s.

# The Xerox NoteTaker

Smalltalk-76 was ported to the Xerox NoteTaker (Smalltalk-78). [1]

The NoteTaker was a portable computer, 8086, floppies only.

It did not have the parallel microarchitecture of the Alto.
(But it did have three 8086 processors!)

By rewriting some machine code in Smalltalk,
“the total machine kernal *[sic]* was reduced to 6K bytes”. [4]

This resulting Smalltalk was faster to interpret than the Alto.

It suffered from slow graphics and no graphics microcode.

# Smalltalk-80

Smalltalk-80 was a gradual evolution of Smalltalk-76.

The language remained essentially the same.

It was polished and documented for external release.

Almost all modern Smalltalks are based on Smalltalk-80.

Squeak contains code in its image from pre-Smalltalk-80!

*From [Smalltalk-76] forward, we never built Smalltalk from [source], but instead [wrote] out a mutated copy of the system. – Dan Ingalls, "The Evolution of Smalltalk" [1]*

# The Residential Environment

Smalltalk was conceptualized as residential from the beginning. [1]

Smalltalk-72 saved the entire 128 K of Alto memory to disk.

Later everything-is-an-object Smalltalk saved object states.

This latter representation is today's Smalltalk "image."

# Steve Jobs and Smalltalk (anecdote)

Steve Jobs saw Smalltalk during his (in)famous visit to PARC.

He didn't like Smalltalk's jumpy line-by-line scrolling.

Dan Ingalls wrote smooth scrolling on the fly. [1]

(Everyone was blown away. [4])

# Squeak

Squeak is the Smalltalk that wasn't possible in the 1970s.

It has graphics, sound, animation, large memory space, …

Alto Smalltalk had to choose (*e.g.*) fast BitBlt or sound.

# Alan Kay's Perspective

For Alan Kay, it was all about teaching children.

He says over and over that that was the value and reward. [4]

At some point he and the LRG became disillusioned of it:
Sure, some kids were learning, but not all.

He never lost interest or hope, but it was clear it wasn't done.

# Other Perspectives

Dan Ingalls and others loved the dynamic environment. [1]

Many comparisons are made to INTERLISP and other Lisps.

A lot of emphasis is placed on changing the running system.

The idea of personal computing motivated all of them.

# What Is an Object?

Kay remarks that C++ *et. al.* colonized objects.

His objects are dynamic and loosely coupled.

They communicate only by passing messages.

Both Kay and Ingalls remark that they may not even reside on the same system!

Smalltalkers do not view C++/Java/*etc.*as truly OO.

# Personal Computing

Kay called the Apple Macintosh “the first personal computer good enough to criticize”. [5]

He called Microsoft’s Tablet PC “the first Dynabook-like computer good enough to criticize.”. [5]

(Those were a long time in coming!)

He’s still waiting for something as powerful and flexible.

# Some Smalltalk Firsts

Smalltalk introduced the following concepts:

- *Everything* is an object
- Overlapping windows
- Pop-up menus
- BitBlt transforming graphics operations

# References I

## Optional Readings


[1] Daniel Ingalls. “The evolution of Smalltalk: From Smalltalk-72 through Squeak”. In: *Proceedings of ACM Programming Languages History of Programming Languages* 4 (2020). URL: https://dl.acm.org/doi/10.1145/3386335.

[2] Daniel H. H. Ingalls. “The Smalltalk Graphics Kernel”. In: *Byte* 6.8 (Aug. 1981), pp. 168–194.

[3] Alan C. Kay. “A Personal Computer for Children of All Ages”. In: *Proceedings of the ACM Annual Conference* 1 (1972). URL: https://dl.acm.org/doi/10.1145/800193.1971922.

[4] Alan C. Kay. “The Early History of Smalltalk”. In: *ACM SIGPLAN Notices* 28.3 (Mar. 1993), pp. 1–54. URL: https://dl.acm.org/doi/10.1145/155360.155364.

[5] Steven Levy. “Bill Gates Says, Take This Tablet”. In: *Newsweek* (Apr. 2001). URL: {https://www.newsweek.com/bill-gates-says-take-tablet-150561}.

# References II

[6] John McCarthy et al. *LISP 1.5 Programmer's Manual*. Second Edition. The MIT Press, 1985. ISBN: 0-262-13011-4.

[7] Seymour Papert. "Teaching Children Thinking". In: *Programmed Learning and Educational Technology* (1972). DOI: 10.1080/1355800720090503.

# License



To retrieve a copy of this material, or related materials, see `https://www.cse.buffalo.edu/~eblanton/`.