

BSDCan 2009 Sponsors



!

!

Thinking about thinking in code

George V. Neville-Neil

BSDCan 2009

Ottawa, Canada



What is this about?



What is this about?

- This is not about any particular BSD



What is this about?

- This is not about any particular BSD
- This presentation is about how we think about coding



What is this about?

- This is **not** about any particular BSD
- This presentation is about how we think about coding
- The opinions expressed are those of the author and do not reflect the opinions of any BSD project...



What is this about?

- This is **not** about any particular BSD
- This presentation is about how we think about coding
- The opinions expressed are those of the author and do not reflect the opinions of any BSD project...
 - Yes, this is a bit of a rant



A Challenge



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.

Bill Gates



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.

Bill Gates



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.

Bill Gates

People who build software are sooooo much more innovative than people who build cars.



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.

Bill Gates

People who build software are sooooo much more innovative than people who build cars.



A Challenge

If GM had kept up with technology like the computer industry has, we would all be driving \$25 cars that got 1000 MPG.

Bill Gates

People who build software are sooooo much more innovative than people who build cars.

Or are we?



Just how is GM doing?



How about Microsoft?



MSFT vs. GM Who's Better?



GM's Original Product



BSDCan 2009



GM's Latest Product



BSDCan 2009



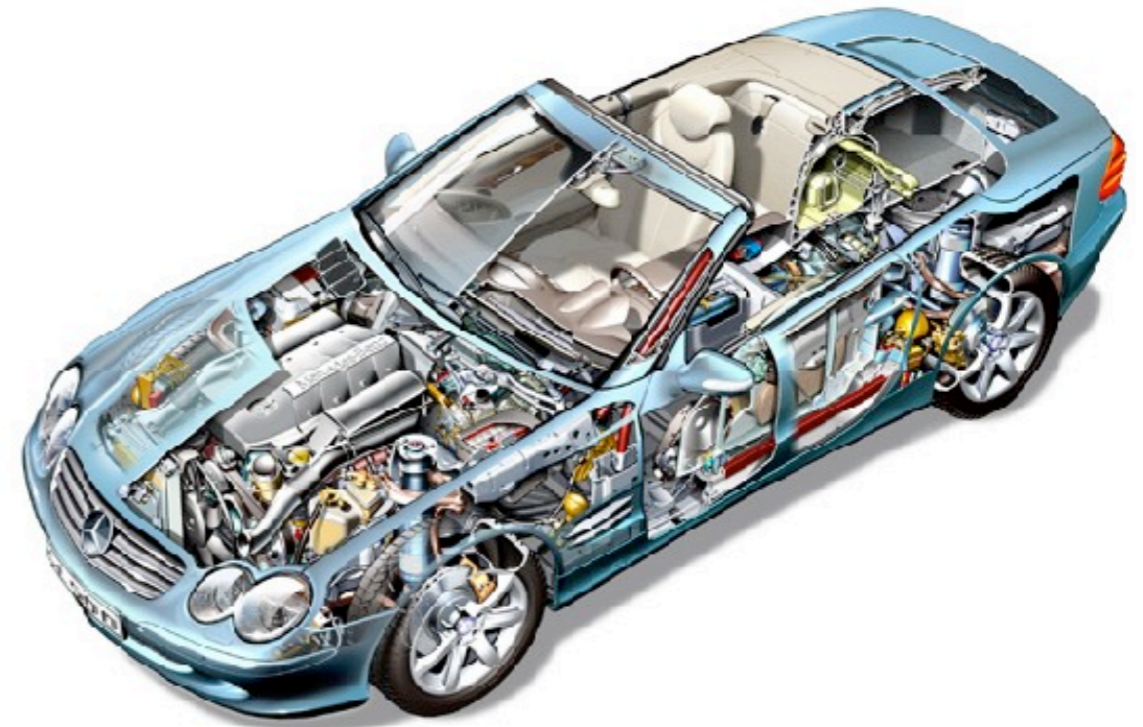
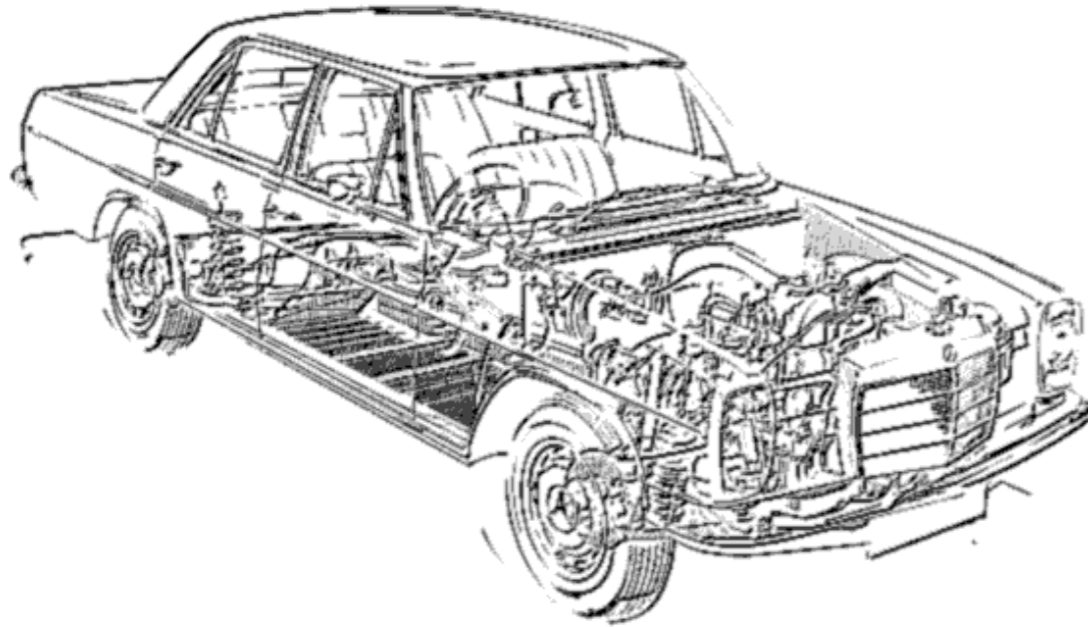
Sometimes you take a wrong turn



and you never recover...



Changing Internals of Autos



Are we assembly line programmers?

Automobiles

- Assembly Line
- Modular Design
- Unified UI
- Just In Time Delivery

Software

- Waterfall Model
- Modules
- Objects
- Components
- Re-use
- Design Patterns



Parts Used in Construction

Automobile

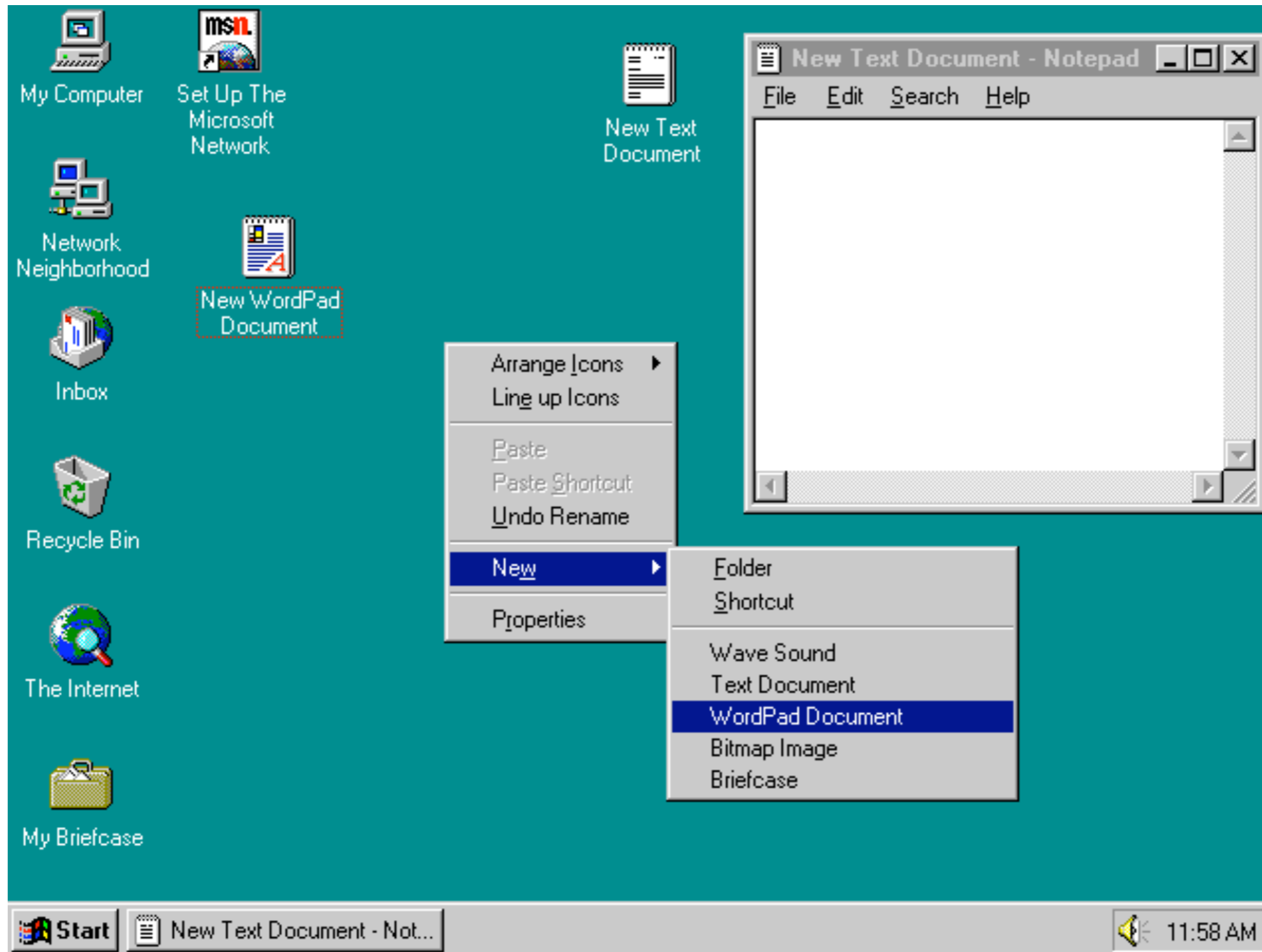
- Engine
- Brakes
- Drive Train
- Wheels
- Transmission
- Door Locks
- Exhaust System

Operating System

- Kernel
- Scheduler
- Memory
- VM
- I/O
- Filesystem
- Sockets



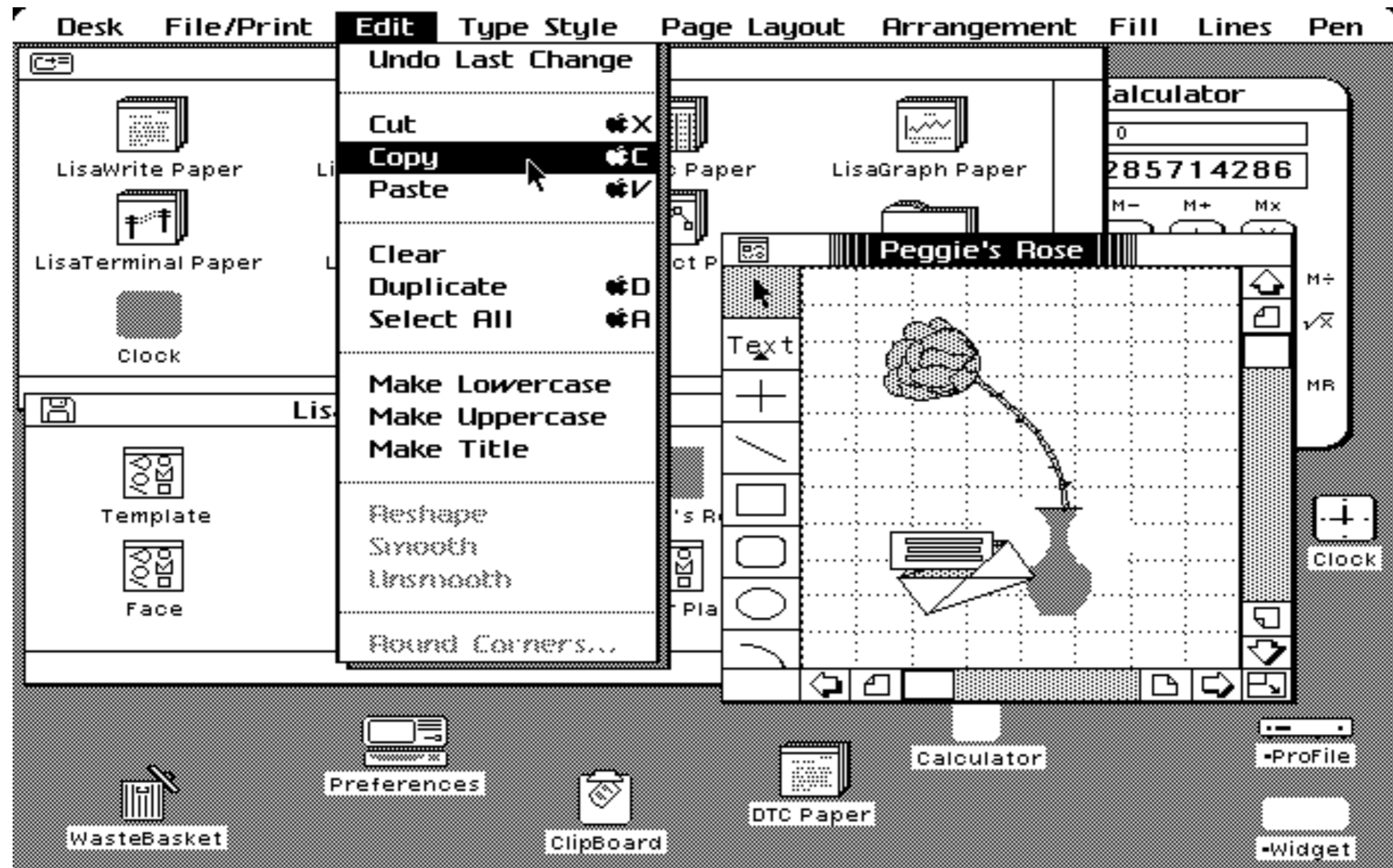
MSFT's Early Product



MSFT's Latest Product



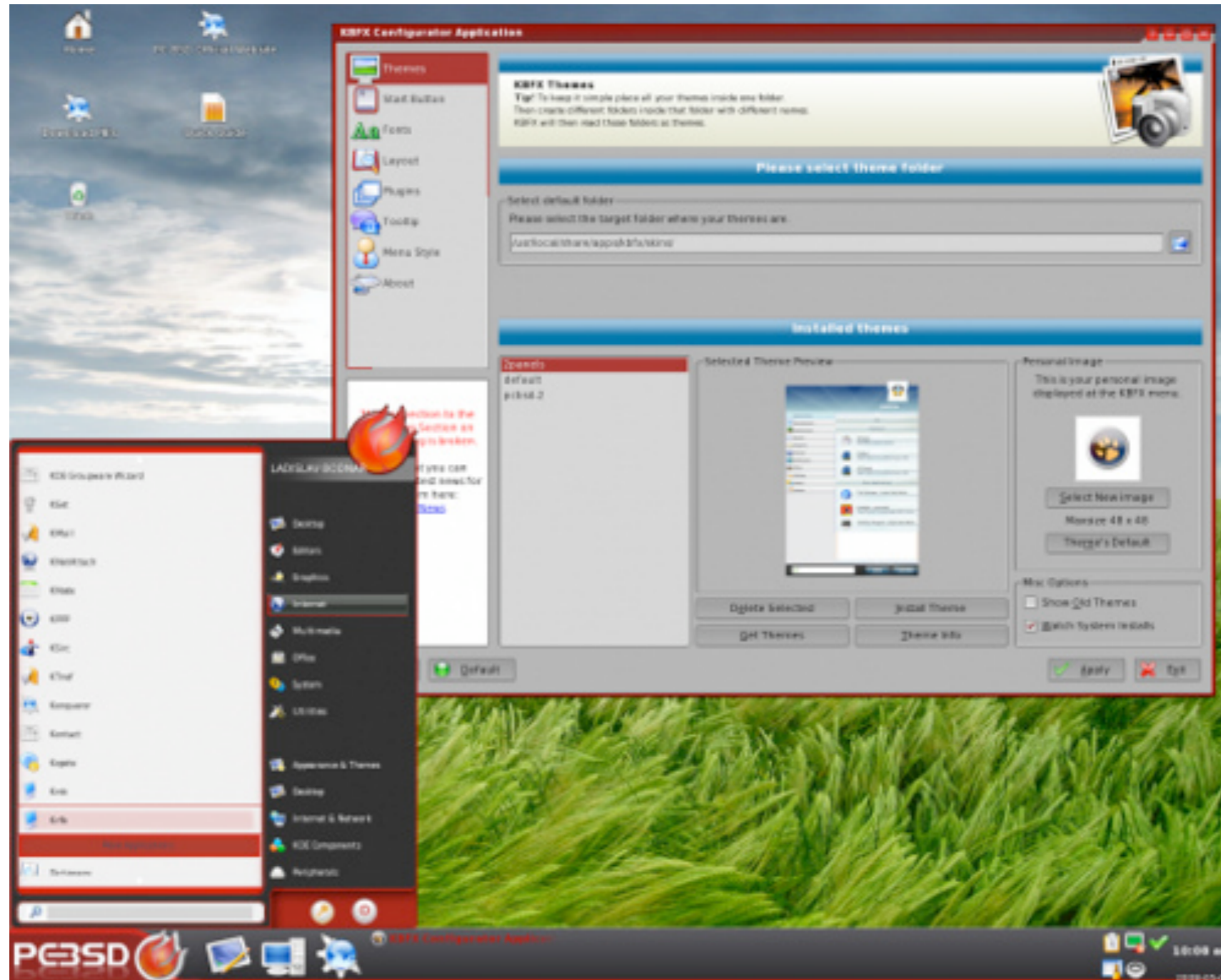
Early Innovation?



Still thinking differently?



One of our recent products



Never forget your history!

The screenshot displays a classic Macintosh-style graphical user interface with several overlapping windows:

- System Browser:** A window at the top left showing a hierarchical tree of system classes and methods. The tree includes categories like 'Collections-Sequential', 'Collections-Text', 'Collections-Array', 'Collections-Stream', 'Collections-Support', 'Graphics-Primitive', 'Graphics-Display', 'Graphics-Media', and 'Graphics-Paths'. A table below the tree lists methods such as 'Interval', 'LinkedList', 'MappedCollection', 'OrderedCollection', and 'SortedCollection'. A 'collect:' window is also visible, showing methods like 'do:', 'do:andBetweenDo:', 'promoteFirstSuchT', 'reverse', 'reverseDo:', and 'select:'.
- Code Editor:** A window titled 'Form Editor' containing Lisp code for a 'collect:' function:

```
collect: aBlock  
  "Evaluate aBlock with each of my elements as the argument. Collect  
  resulting values into a collection that is like me. Answer with  
  collection. Override superclass in order to use add:, not ofPut:  
  
  | newCollection |  
  newCollection ← self species new.  
  self do: [:each | newCollection add: (aBlock value: each)].  
  newCollection
```
- User Interrupt:** A window showing a stack of messages:

```
Paragraph>>characterBlockAtPoint:  
Paragraph>>mouseSelect:to:  
CodeController(ParagraphEditor)>>processRedButton  
CodeController(ParagraphEditor)>>processMouseButtons  
CodeController(ParagraphEditor)>>controlActivity  
CodeController(Controller)>>controlLoop
```
- controlActivity:** A window showing the following code:

```
controlActivity  
  self scrollbarContainsCursor  
  ifTrue:  
    [self scroll]  
  ifFalse:  
    [self processKeyboard] ifNotNil:  
      self processMouse
```
- File List:** A window showing a list of files:

```
File List  
[<Robson>SF]*  
[file] <Robson>SF>Screenform.st  
[file] <Robson>SF>Screenform.text  
[file] <Robson>SF>Screenform.Changes.st  
[file] <Robson>SF>WordGraphics.form
```
- Diagram:** A technical drawing of a bolt and nut, labeled 'Fig. 1' in a decorative font.
- Other Windows:** A 'blueButton' window with coordinates '31@507 corner: 63@770'. A 'Screenform' window showing a 'printRectangle:' message with coordinates '(30@5 extent: 674@790)' and an 'onFileNamed:' message for 'ExampleScreen.press'. A '(Form readFrom: "FilledSkate.form") edit' message is also visible.



Well, perhaps that's just UIs



Well, perhaps that's just UIs

- Users force us to provide these types of systems.



Well, perhaps that's just UIs

- Users force us to provide these types of systems.
- Paradigm shift is hard.



Well, perhaps that's just UIs

- Users force us to provide these types of systems.
- Paradigm shift is hard.
- If we make it too different it won't sell.



Well, perhaps that's just UIs

- Users force us to provide these types of systems.
- Paradigm shift is hard.
- If we make it too different it won't sell.
- We're just implementing the marketing spec!

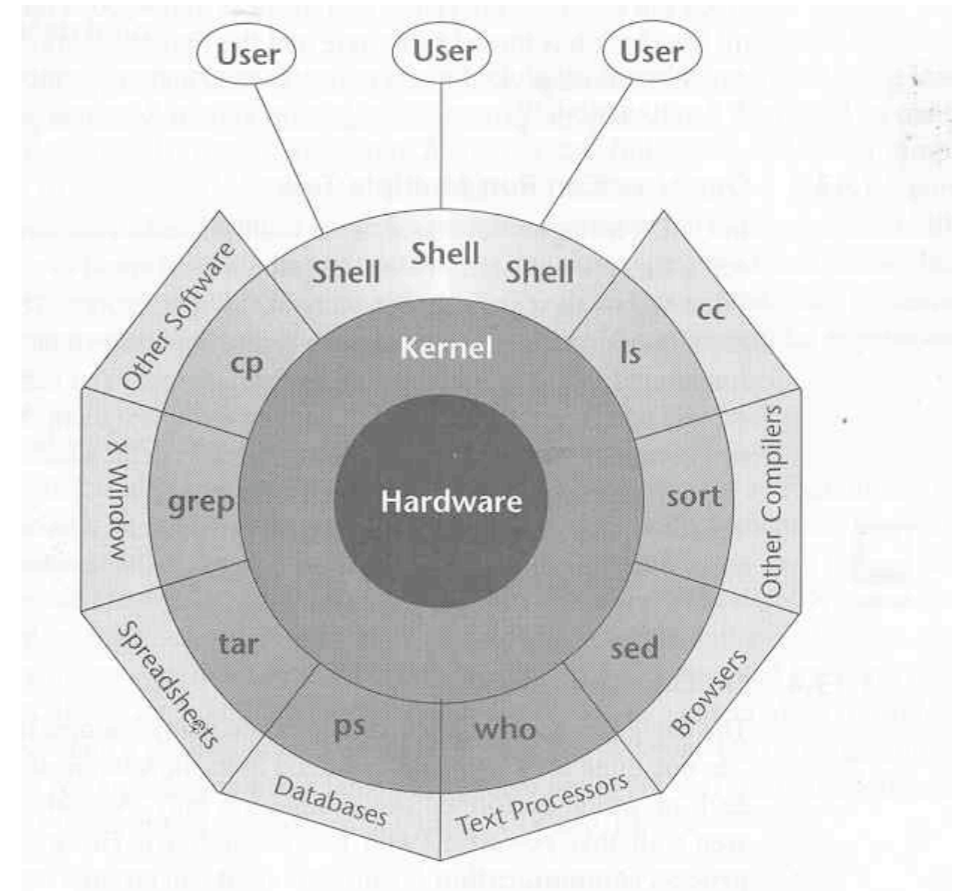
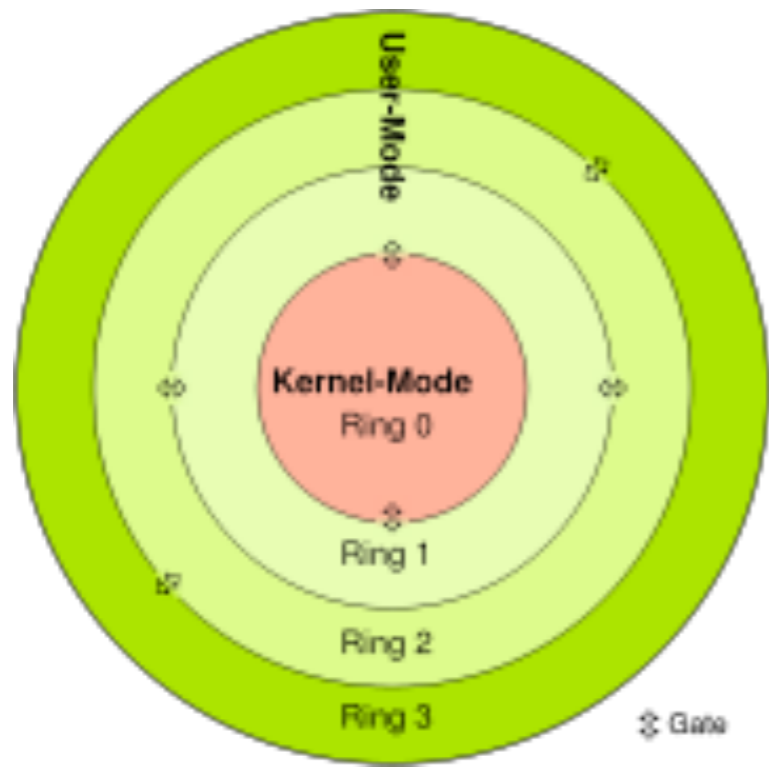


Well, perhaps that's just UIs

- Users force us to provide these types of systems.
- Paradigm shift is hard.
- If we make it too different it won't sell.
- We're just implementing the marketing spec!
- What's under the hood is totally different!



Changing Internals of OSs



How Different Are We?

- BSD
 - C
 - Kernel
 - VM
 - Processes
 - Threads
 - Events UI
- Linux
 - C
 - Kernel
 - VM
 - Processes
 - Threads
 - Events UI
- Windows
 - C
 - Kernel
 - VM
 - Processes
 - Threads
 - Events UI
- Mac OS
 - C/C++
 - uKernel ?
 - VM
 - Processes
 - Threads
 - Events UI



Why do we fail to provide new models?



Why do we fail to provide new models?

- What we were taught?



Why do we fail to provide new models?

- What we were taught?
- What the people want?



Why do we fail to provide new models?

- What we were taught?
- What the people want?
- What we want?



Why do we fail to provide new models?

- What we were taught?
- What the people want?
- What we want?
- These are the best models?



Why do we fail to provide new models?

- What we were taught?
- What the people want?
- What we want?
- These are the best models?
- Our languages and environment dictate our models?



How do most programmers program?



How do most programmers program?

- Text structured in various (non)sensical ways



How do most programmers program?

- Text structured in various (non)sensical ways
- Edit, Compile, Link, Run, Debug cycle



How do most programmers program?

- Text structured in various (non)sensical ways
- Edit, Compile, Link, Run, Debug cycle
- Edit, run, edit, run, edit, run



How do most programmers program?

- Text structured in various (non)sensical ways
- Edit, Compile, Link, Run, Debug cycle
- Edit, run, edit, run, edit, run
- Three basic types of languages



But what are your options?



But what are your options?

- Algol



But what are your options?

- Algol
 - Procedural



But what are your options?

- Algol
 - Procedural
- Lisp



But what are your options?

- Algol
 - Procedural
- Lisp
 - Functional



But what are your options?

- Algol
 - Procedural
- Lisp
 - Functional
- Prolog



But what are your options?

- Algol
 - Procedural
- Lisp
 - Functional
- Prolog
 - Insanity



How we think in code matters!



How we think in code matters!

- Sloppy languages lead to sloppy code



How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always



How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always
- Unsafe languages lead to unsafe code



How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always
- Unsafe languages lead to unsafe code
 - Almost always



How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always
- Unsafe languages lead to unsafe code
 - Almost always
- Confusing languages lead to confusing code



How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always
- Unsafe languages lead to unsafe code
 - Almost always
- Confusing languages lead to confusing code
 - Almost always

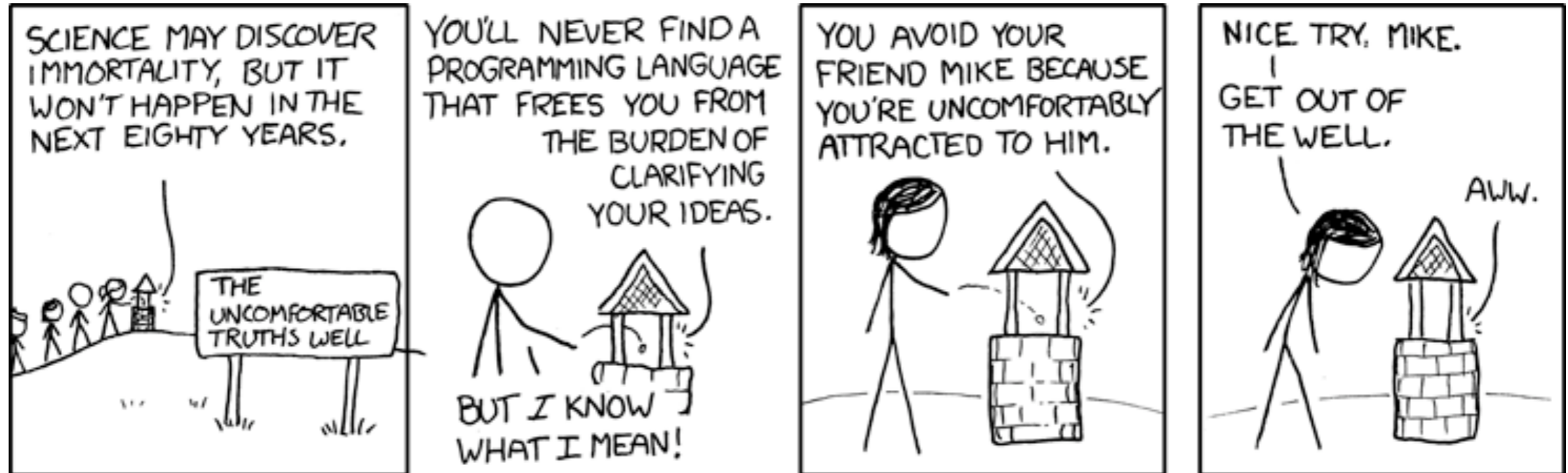


How we think in code matters!

- Sloppy languages lead to sloppy code
 - Almost always
- Unsafe languages lead to unsafe code
 - Almost always
- Confusing languages lead to confusing code
 - Almost always
- Lowering the barrier to entry has lowered the quality of code



How many words is a comic worth?



Programmers learn what they live



Programmers learn what they live

- If a programmer reads good code she will write good code



Programmers learn what they live

- If a programmer reads good code she will write good code
- If a programmer sees good abstractions he will write good abstractions



Programmers learn what they live

- If a programmer reads good code she will write good code
- If a programmer sees good abstractions he will write good abstractions
- If a programmer works with good programmers her code will improve



Programmers learn what they live

- If a programmer reads good code she will write good code
- If a programmer sees good abstractions he will write good abstractions
- If a programmer works with good programmers her code will improve
- If a programmer works with poor programmers his code will suffer



Programmers learn what they live

- If a programmer reads good code she will write good code
- If a programmer sees good abstractions he will write good abstractions
- If a programmer works with good programmers her code will improve
- If a programmer works with poor programmers his code will suffer
- Don't read crap code!



Programmers learn what they live

- If a programmer reads good code she will write good code
- If a programmer sees good abstractions he will write good abstractions
- If a programmer works with good programmers her code will improve
- If a programmer works with poor programmers his code will suffer
- Don't read crap code!
- Don't hang out with crap programmers!



Everything has been discovered!



Everything has been discovered!

- “Everything that can be invented has been invented.” Charles Duell (?)



Everything has been discovered!

- “Everything that can be invented has been invented.” Charles Duell (?)
- Turns out not to be true



Will the real Charles Duell...



Will the real Charles Duell...

- "Our future progress and prosperity depend upon our ability to equal, if not surpass, other nations in the enlargement and advance of science, industry and commerce. To invention we must turn as one of the most powerful aids to the accomplishment of such a result."

Charles Duell



Stop Re-inventing the Wheel



Stop Re-inventing the Wheel

- There does not need to be another list implementation



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree
 - Dictionary



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree
 - Dictionary
 - Locking API



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree
 - Dictionary
 - Locking API
 - > Spin Lock



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree
 - Dictionary
 - Locking API
 - > Spin Lock
- Stop fighting over the scraps



Stop Re-inventing the Wheel

- There does not need to be another list implementation
- No one needs another hash table
- Tree
 - Dictionary
 - Locking API
 - > Spin Lock
- Stop fighting over the scraps
- Stop re-implementing your school projects



Invent Some New Wheels



Invent Some New Wheels

- UNIX should not be the end of the family tree



Invent Some New Wheels

- UNIX should not be the end of the family tree
 - No matter how much we love BSD



Invent Some New Wheels

- UNIX should not be the end of the family tree
 - No matter how much we love BSD
- Can you imagine if Vista is what our grandchildren will use?



Invent Some New Wheels

- UNIX should not be the end of the family tree
 - No matter how much we love BSD
- Can you imagine if Vista is what our grandchildren will use?
- MacOS X? (XI?)



Invent Some New Wheels

- UNIX should not be the end of the family tree
 - No matter how much we love BSD
- Can you imagine if Vista is what our grandchildren will use?
- MacOS X? (XI?)
- Linux?



How to Find New Wheels



How to Find New Wheels

- Read some papers



How to Find New Wheels

- Read some papers
 - Read the abstracts first



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python
 - Haskell



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python
 - Haskell
 - C++



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python
 - Haskell
 - C++
 - ...



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python
 - Haskell
 - C++
 - ...
- Don't specialize too much



How to Find New Wheels

- Read some papers
 - Read the abstracts first
 - Ignore most of them, most are pointless
- Look at code you wouldn't normally touch
- Learn a computer language you don't like
 - Scheme
 - Perl
 - Python
 - Haskell
 - C++
 - ...
- Don't specialize too much
- Lose your religion!



You must be willing to



You must be willing to

- break things



You must be willing to

- break things
- look stupid



You must be willing to

- break things
- look stupid
- be wrong



You must be willing to

- break things
- look stupid
- be wrong
- learn from others



All Programmers Have 2 Enemies



All Programmers Have 2 Enemies

- Hubris



All Programmers Have 2 Enemies

- Hubris
- Not starting projects



All Programmers Have 2 Enemies

- Hubris
- Not starting projects
- Never finishing a project



All Programmers Have 2 Enemies

- Hubris
- Not starting projects
- Never finishing a project
 - OK, all programmers have 3 enemies



All Programmers Have 2 Enemies

- Hubris
- Not starting projects
- Never finishing a project
 - OK, all programmers have 3 enemies
 - Off by one errors are a programmer's 3rd enemy



All Programmers Have 2 Enemies

- Hubris
- Not starting projects
- Never finishing a project
 - OK, all programmers have 3 enemies
 - Off by one errors are a programmer's 3rd enemy
 - > Counting from 0



Your time is limited



Your time is limited

- You are only going to live to 72



Your time is limited

- You are only going to live to 72
 - On average



Your time is limited

- You are only going to live to 72
 - On average
- You have to work to eat



Your time is limited

- You are only going to live to 72
 - On average
- You have to work to eat
- Most people only ever have 3 original ideas in their life



Your time is limited

- You are only going to live to 72
 - On average
- You have to work to eat
- Most people only ever have 3 original ideas in their life
 - Not 3 good ideas, just 3 original ideas



Your time is limited

- You are only going to live to 72
 - On average
- You have to work to eat
- Most people only ever have 3 original ideas in their life
 - Not 3 good ideas, just 3 original ideas
- You can easily waste your time on the wrong ideas



Your time is limited

- You are only going to live to 72
 - On average
- You have to work to eat
- Most people only ever have 3 original ideas in their life
 - Not 3 good ideas, just 3 original ideas
- You can easily waste your time on the wrong ideas
- Find people who will honestly tell you if your ideas are crap



Places to start



Places to start

- Safe and powerful programming languages



Places to start

- Safe and powerful programming languages
- Anything that measurably reduces complexity



Places to start

- Safe and powerful programming languages
- Anything that measurably reduces complexity
- Visualization Tools



Places to start

- Safe and powerful programming languages
- Anything that measurably reduces complexity
- Visualization Tools
- Real Software Re-use



Places to start

- Safe and powerful programming languages
- Anything that measurably reduces complexity
- Visualization Tools
- Real Software Re-use
- Novel ways of organizing data



Places to start

- Safe and powerful programming languages
- Anything that measurably reduces complexity
- Visualization Tools
- Real Software Re-use
- Novel ways of organizing data
- Solve the dependency problem



Thank you for your time



Get back to work!

