BSDCan 2009 Sponsors

!

!

Thinking about thinking in code

George V. Neville-Neil

BSDCan 2009

Ottawa, Canada

What is this about?

BSDCan 2009

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 soooo 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 soooo 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 soooo 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?

BSDCan 2009

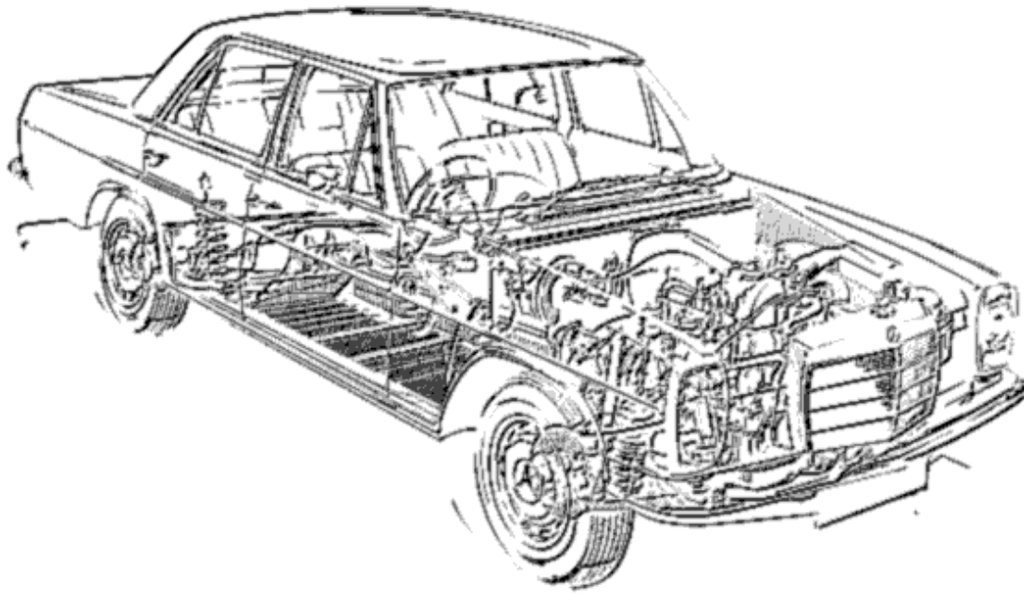
GM's Original Product

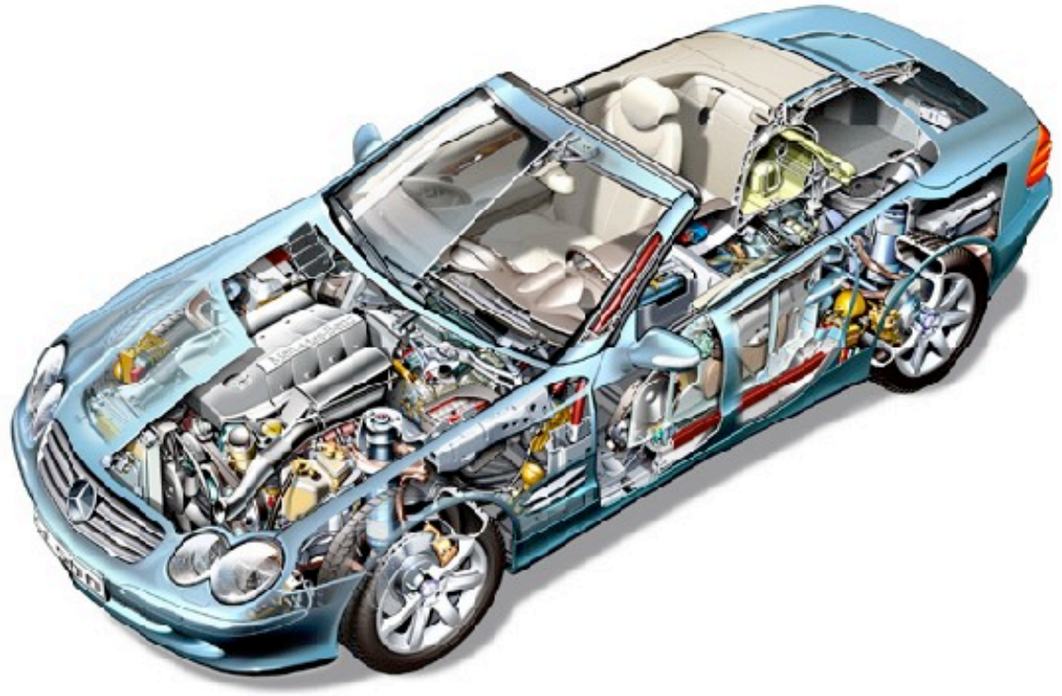
GM's Latest Product

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

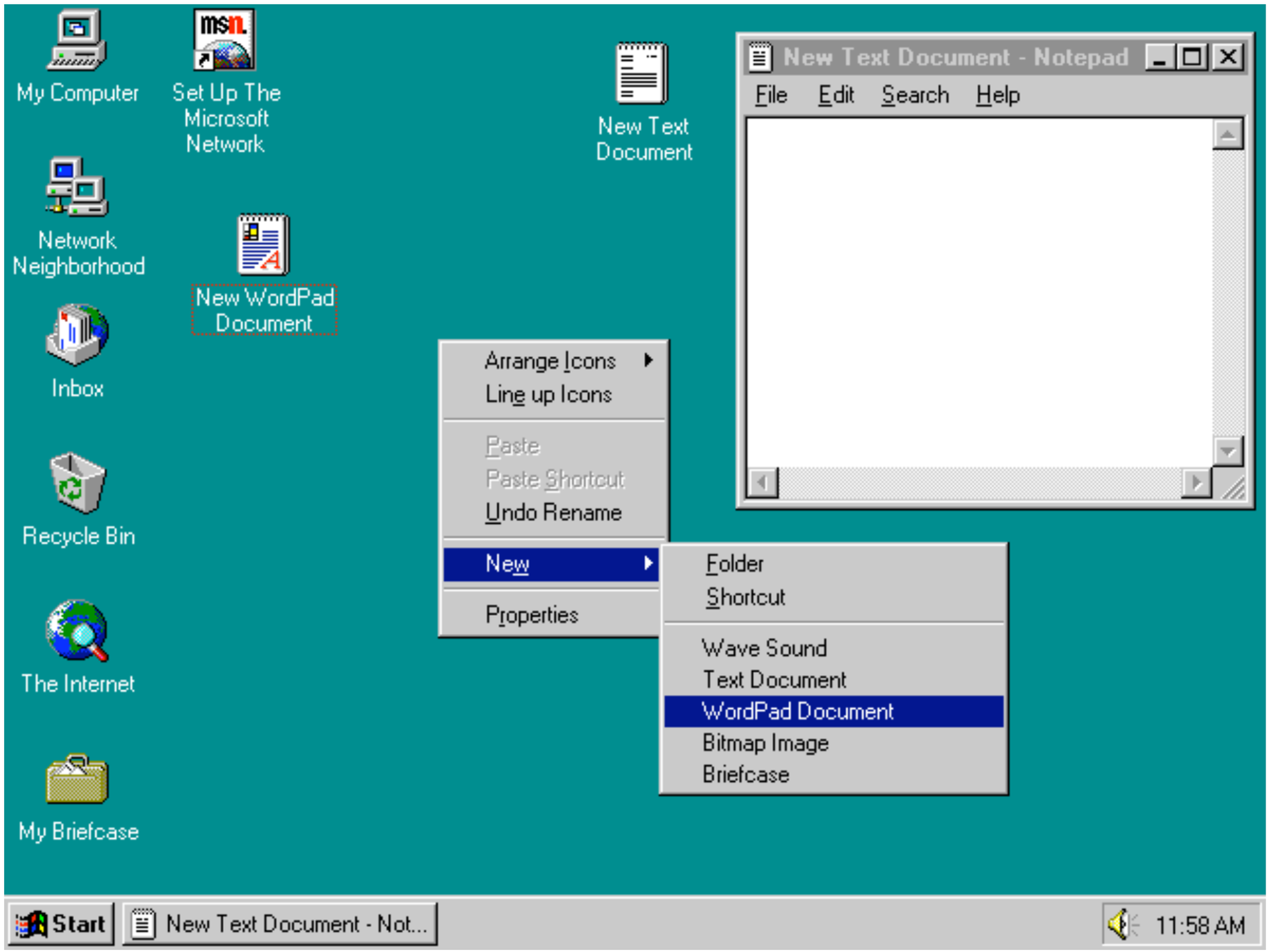
Automobile

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

Operating System

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

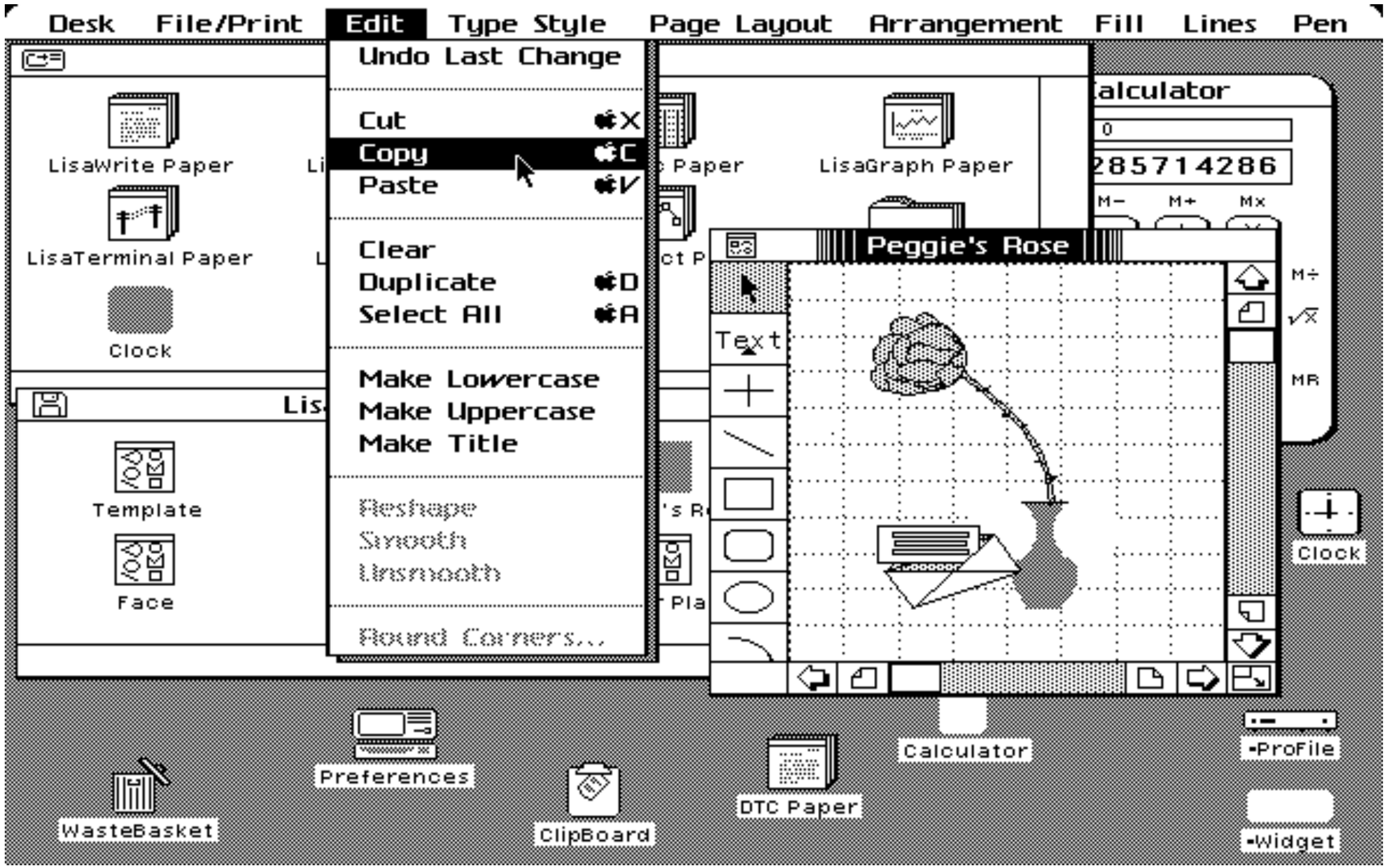
MSFT's Early Product



MSFT's Latest Product

BSDCan 2009

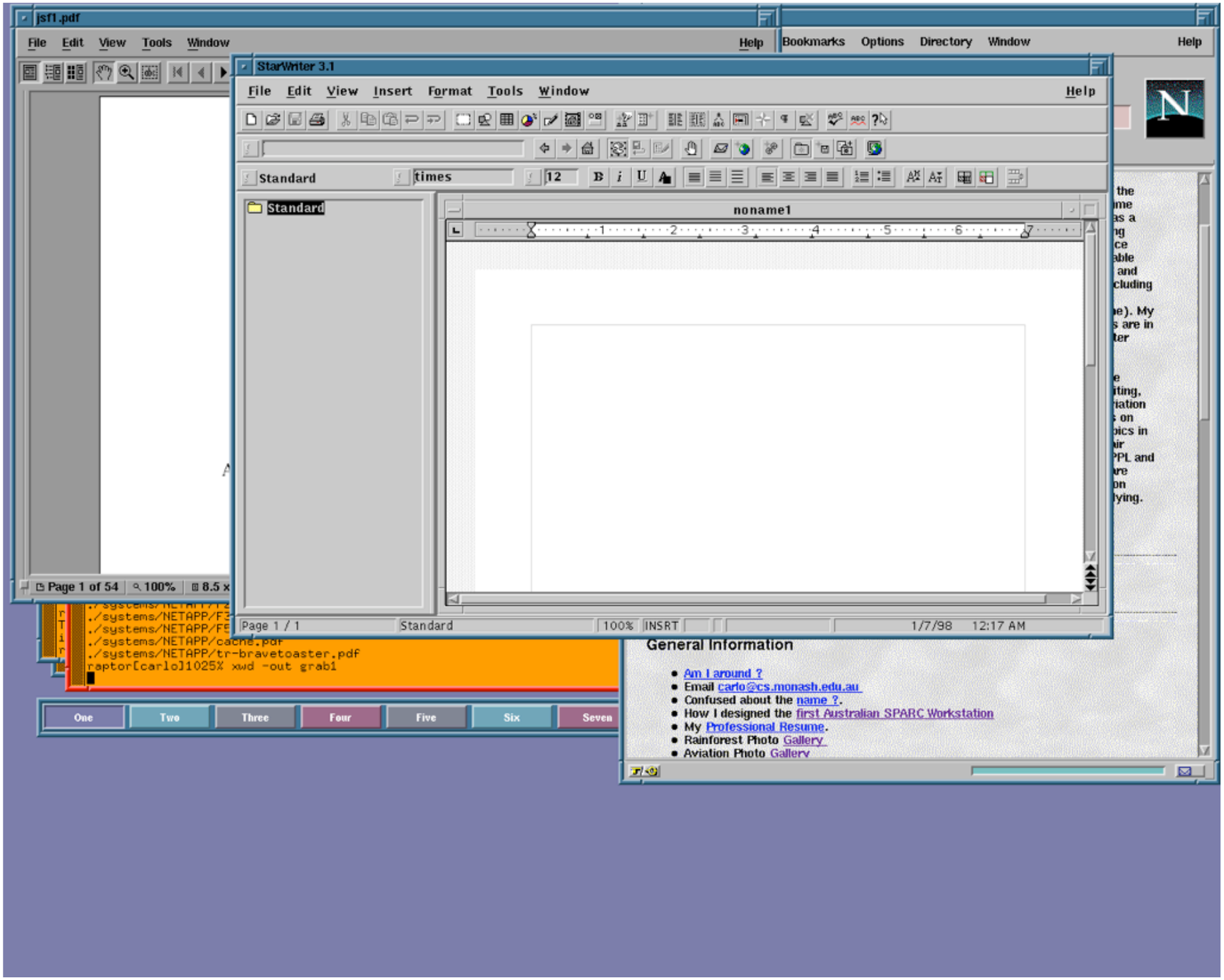
Early Innovation?



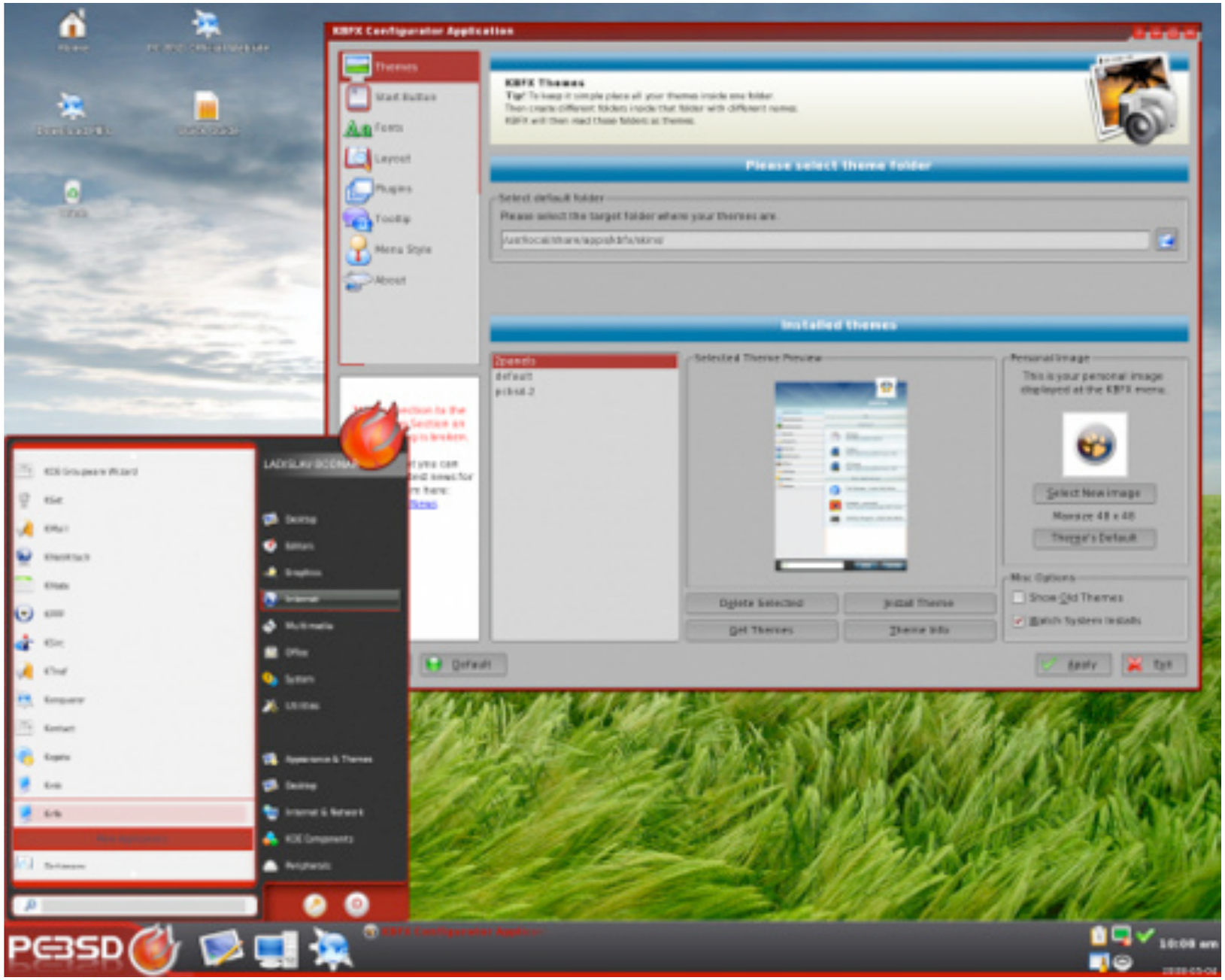
Still thinking differently?

BSDCan 2009

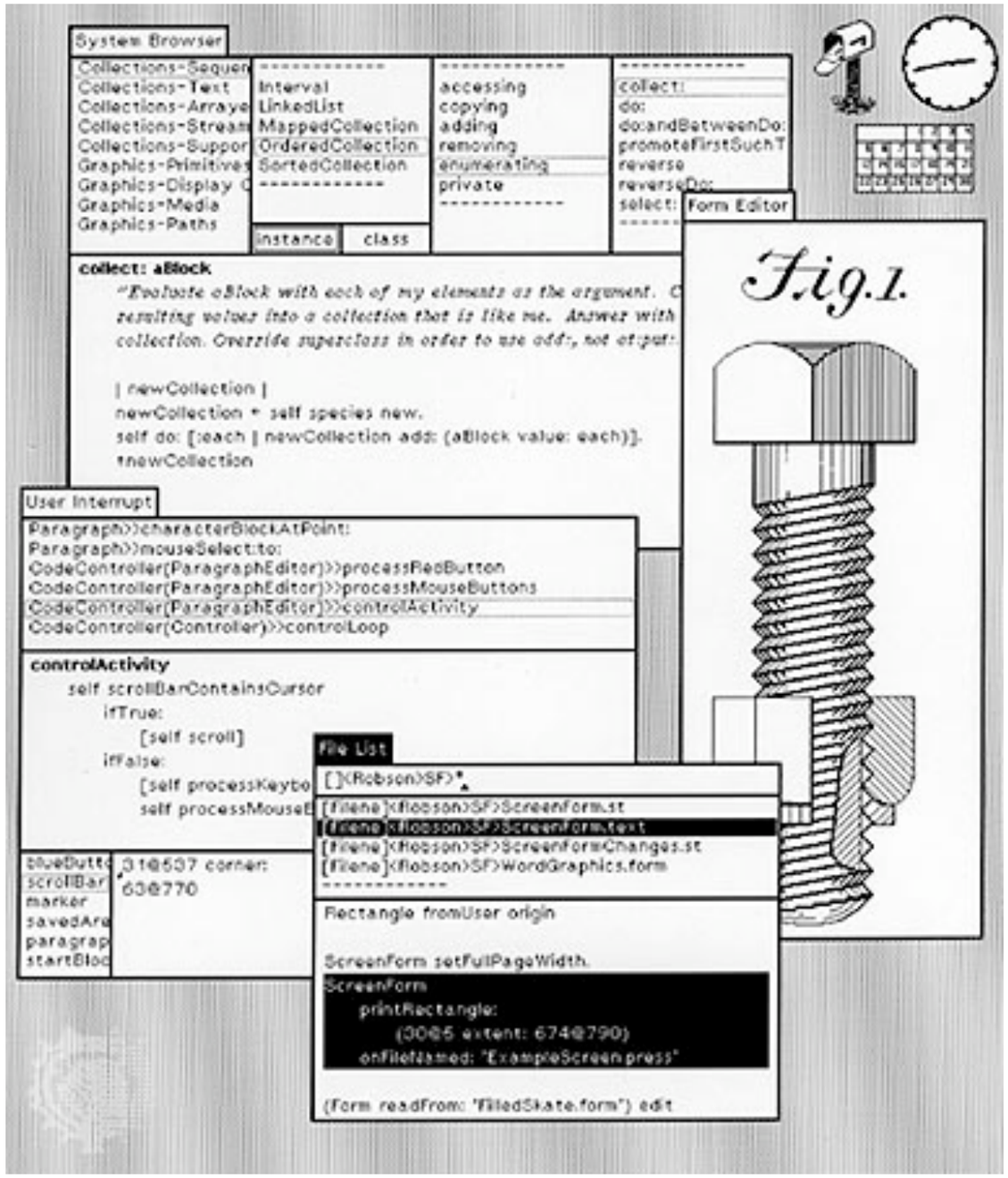
One of “our” early products



One of our recent products



Never forget your history!



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.

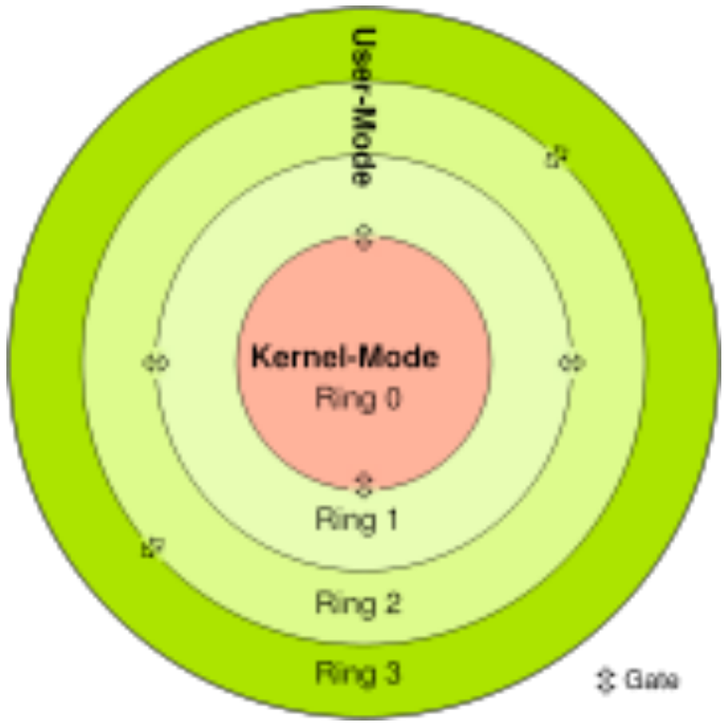
Well, perhaps that's just UIs

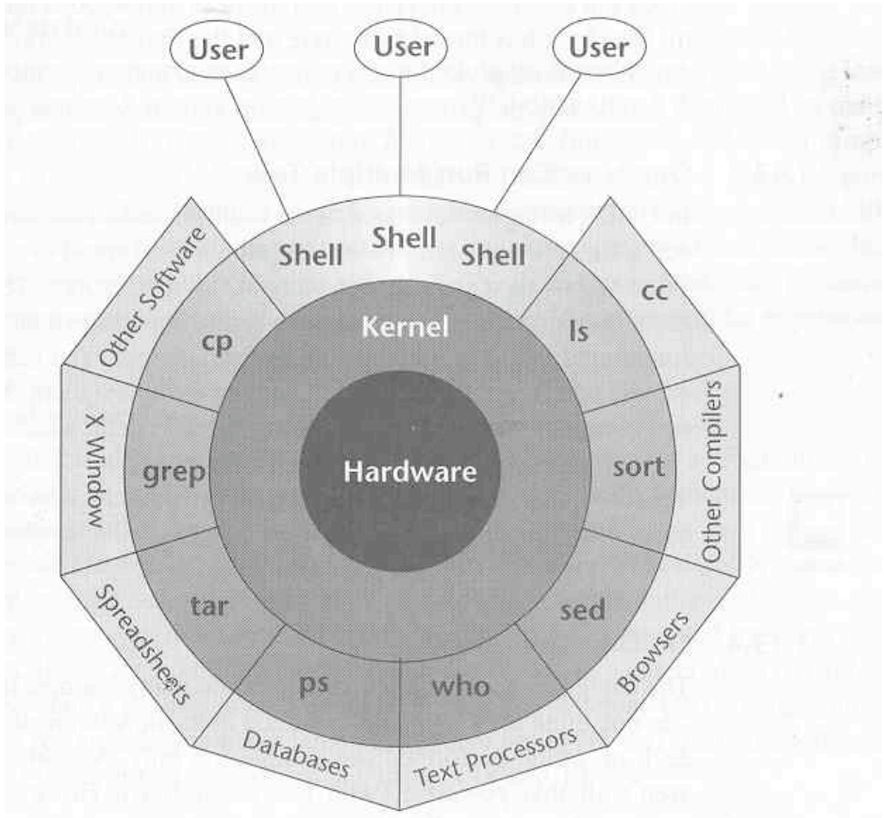
- Users force us to provide these types of systems.
- Paradigm shift is hard.
- If we make it too different it won't sell.
- We're just implementing the marketing spec!

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!

BSDCan 2009

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

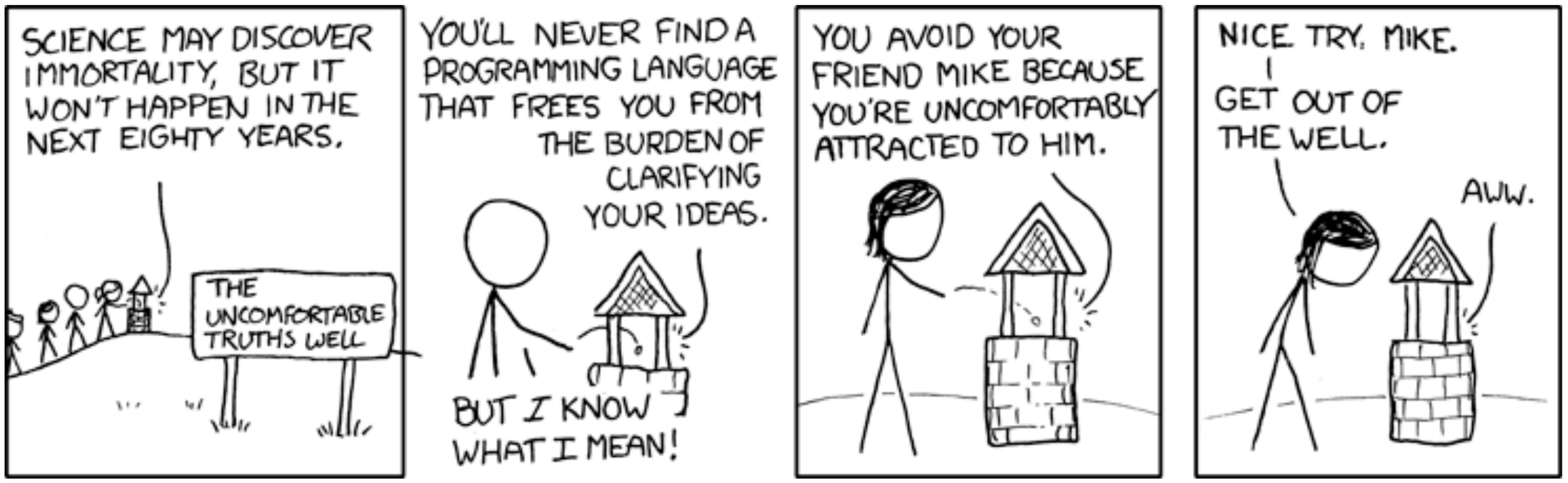
How we think in code matters!

- 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

BSDCan 2009

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...

BSDCan 2009

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

BSDCan 2009

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

BSDCan 2009

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

BSDCan 2009

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

BSDCan 2009

Get back to work!