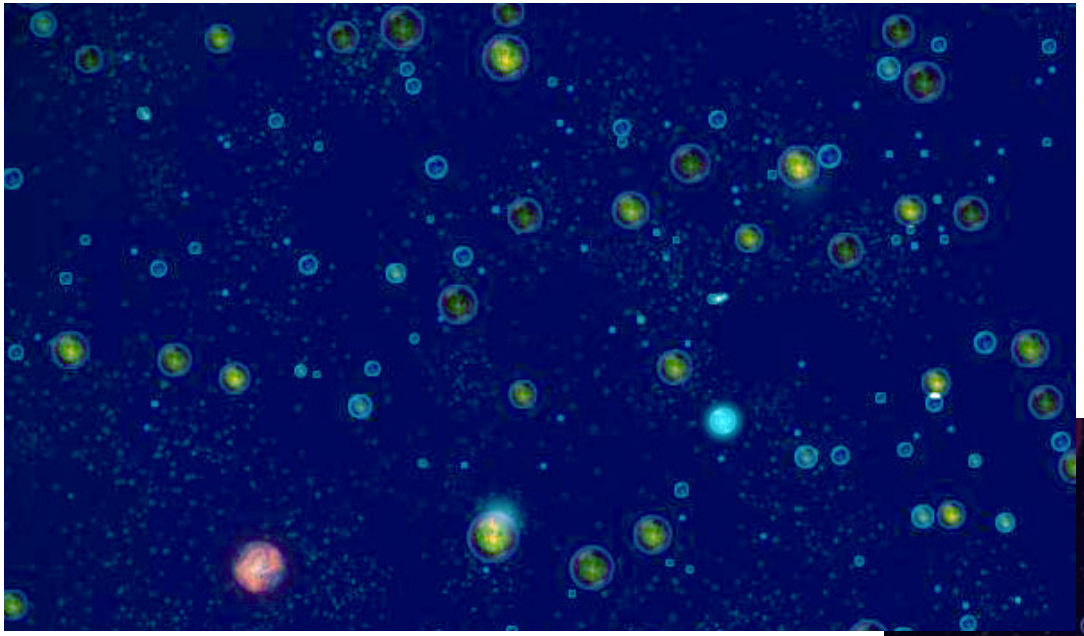
AI Developers Rant!

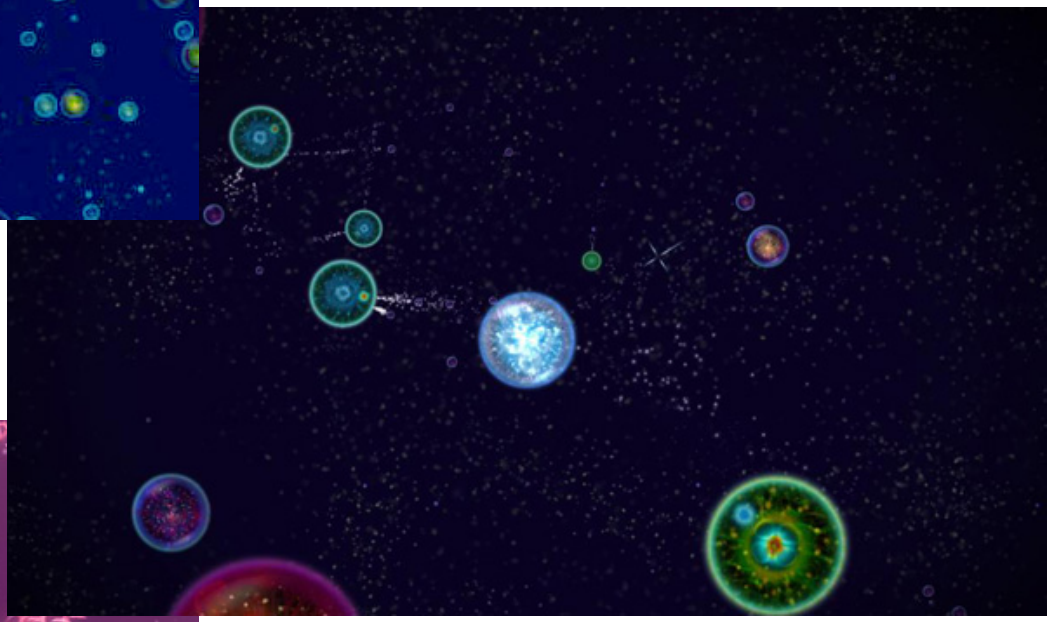
- Borut Pfeifer
  - Whither Innovation in Character AI?
- Adam Russell
  - Designing Sideways
- John Funge
  - Trials and Tribulations of AI Middleware
- Kevin Dill
  - Quit Being Afraid of the Flippin' Floats
- Steve Rabin
  - The Pain and Suffering of Custom Scripting Languages
- Dave Mark
  - No Respect! How I'm Really Annoyed at Game Reviewers

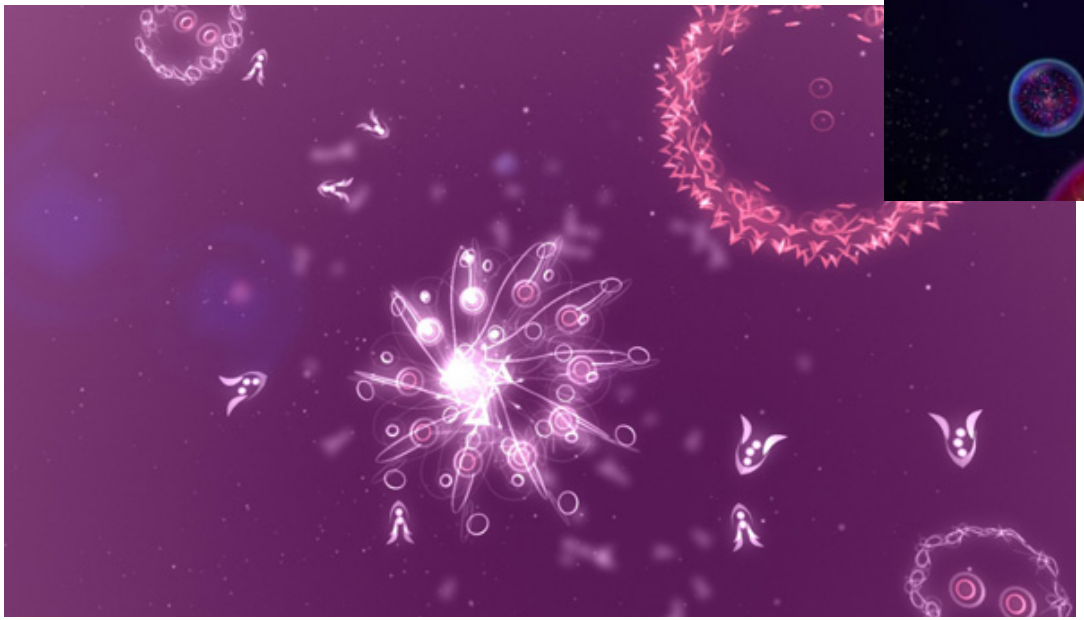
Whither Innovation in Character AI?

Borut Pfeifer
Plush Apocalypse Productions


2005!!!

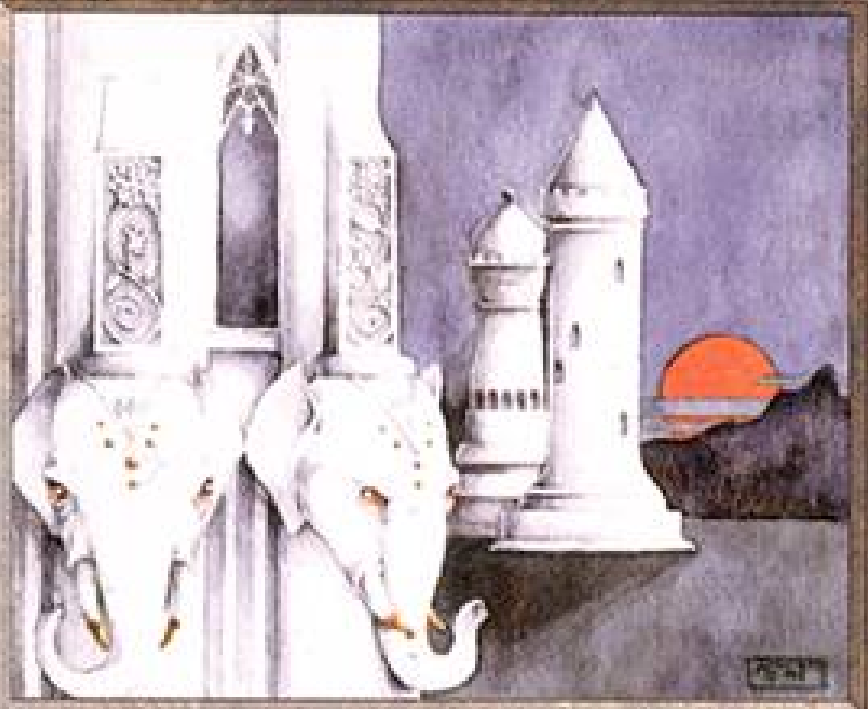






Ivory Tower

1



Artifact

During your upkeep phase, gain 1 life for each card in your hand above four.

Valuing scholarship above all else, the inhabitants of the Ivory Tower reward those who sacrifice power for knowledge.

Software architects

Game Designers

Storytellers

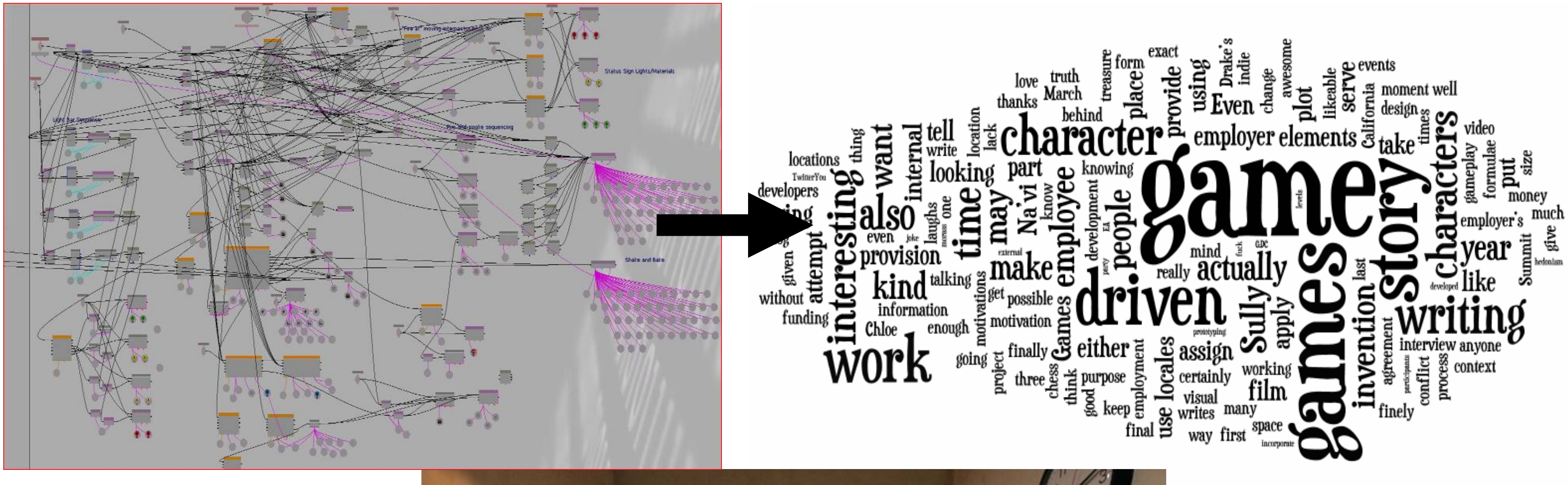
Technical animators

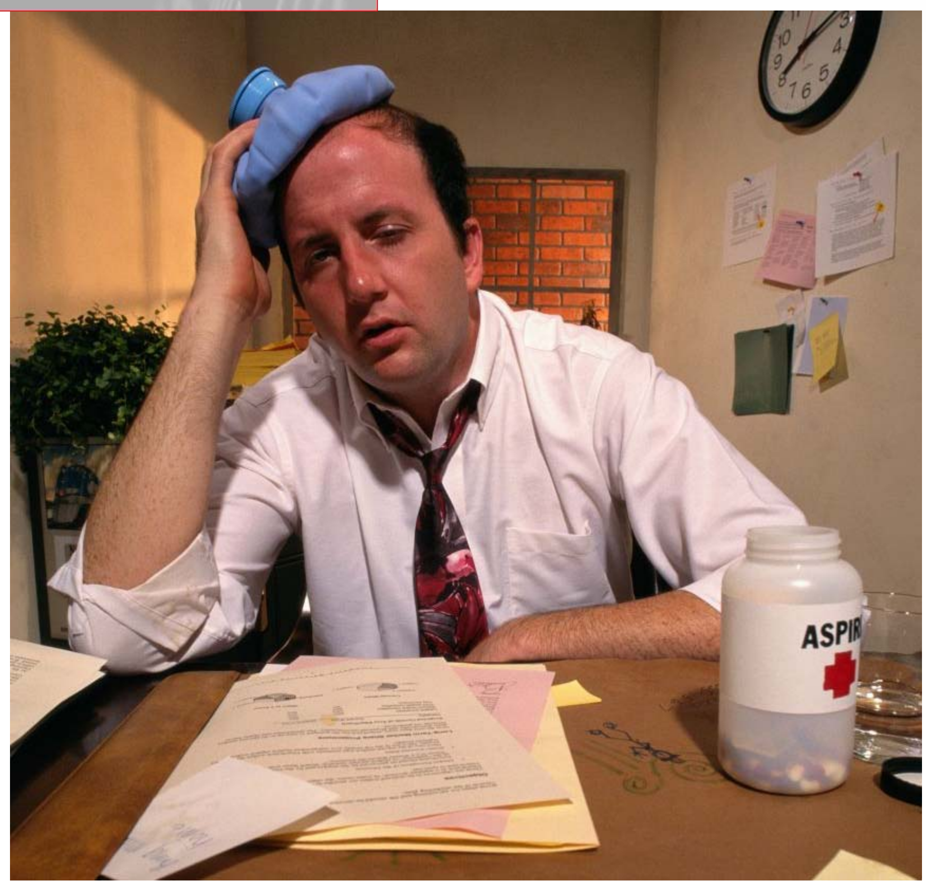
Marketers

Info-vis designers

Producers

Teachers


game
games
driven
story
character
work
writing
interesting
characters
also
time
kind
make
employee
people
invention
actually
year
provision
want
looking
employer
elements
Sully
film
take



Where can you add a little…

Drama

Emotional state

Expressiveness

Systems feedback

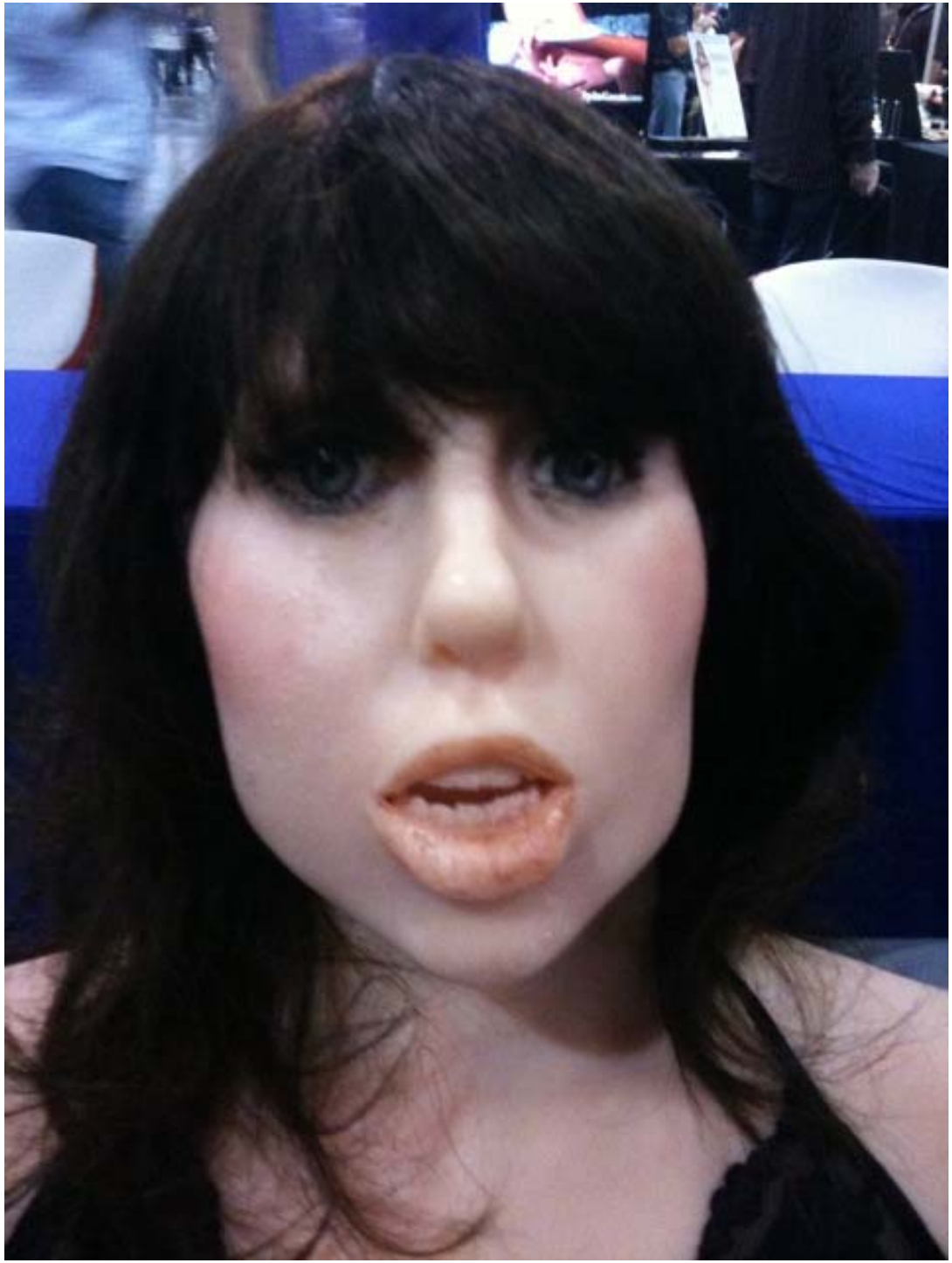
Player modeling

NPC Lifespan!

Behavioral Adverbs

Declarative authoring

Experience management



Designing sideways

Adam Russell

B-Block Studios / University of Derby

When working with designers...

There are two options.

1. Bottom-up

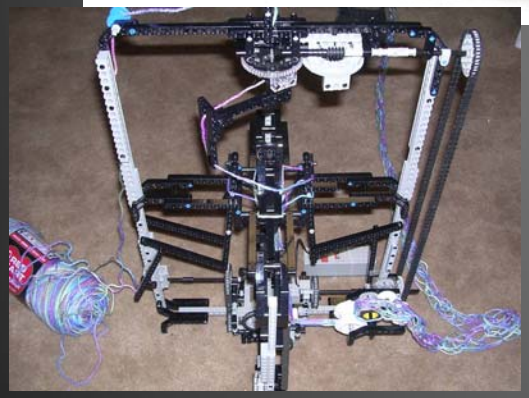
Bottom-up design helps us reuse content.

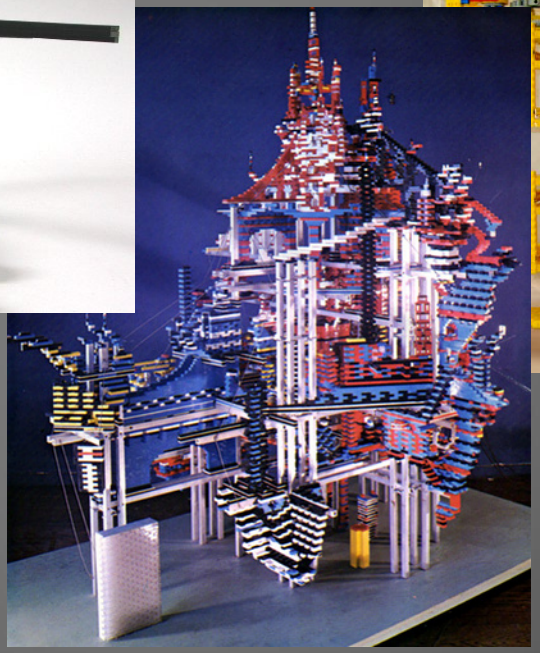
It scales well.

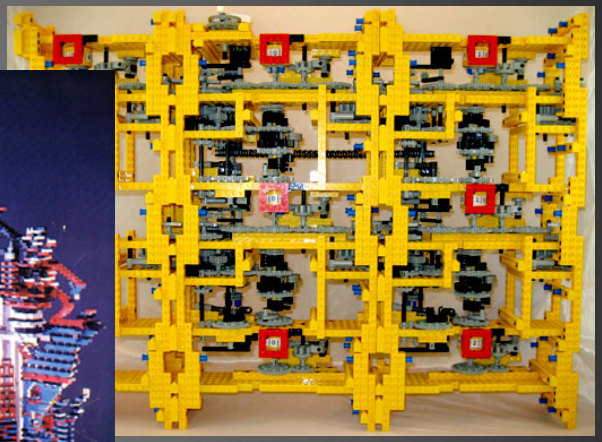
Bottom-up design gives the player more control.

It supports creative play.

Bottom-up design allows new combinations of existing rules.







It feels open-ended.

Bottom-up scenarios are all variations on a theme.

This can get boring.

Bottom-up scenarios are also hard to control.

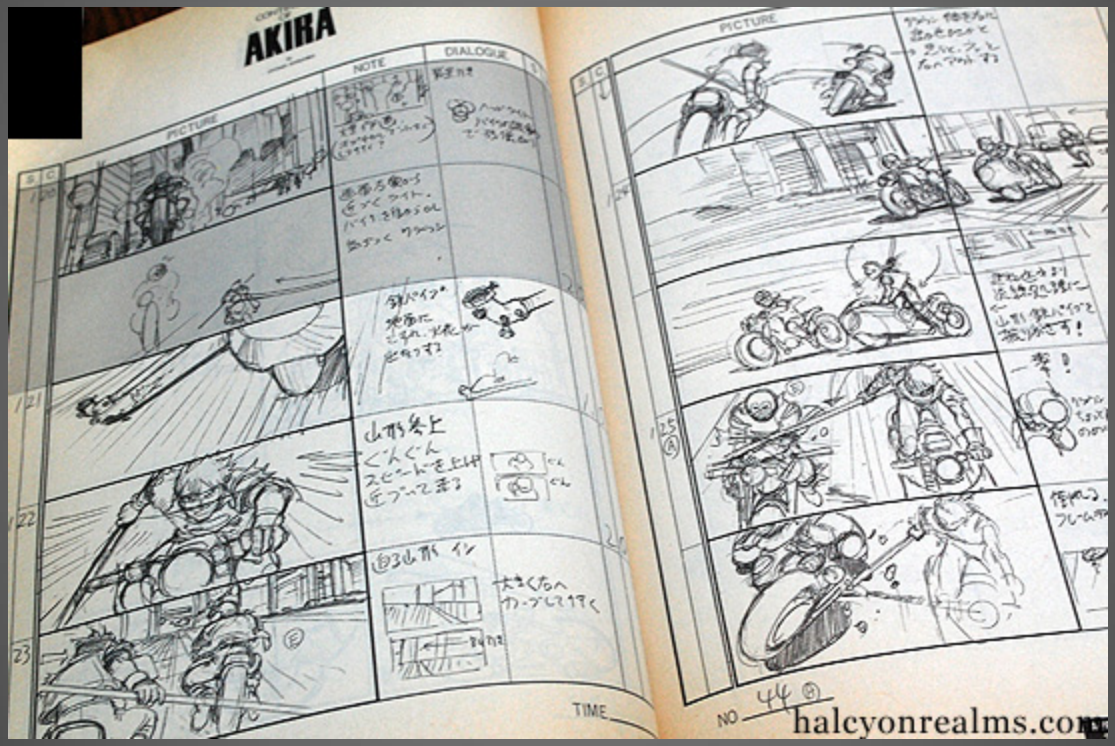
This frustrates designers.

2. Top-down

Top-down design can be used to control complex scenes.

It supports narrative.

Top-down design allows us to plan the mode of presentation.



It supports a cinematic style.

Top-down design delivers more unique moments.

It provides a rich experience.

Top-down design reduces potential for content reuse.

This doesn’t scale well.

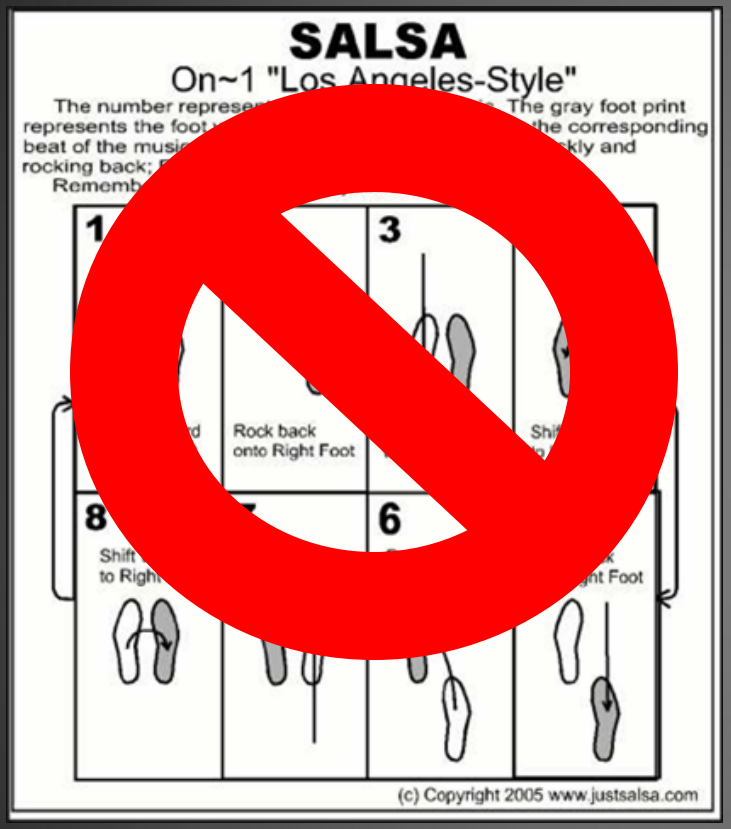
Top-down design gives a similar experience each time.

This reduces replay value.

Bottom-up design isn't enough.

We sometimes need servants.

Top-down design isn't enough.



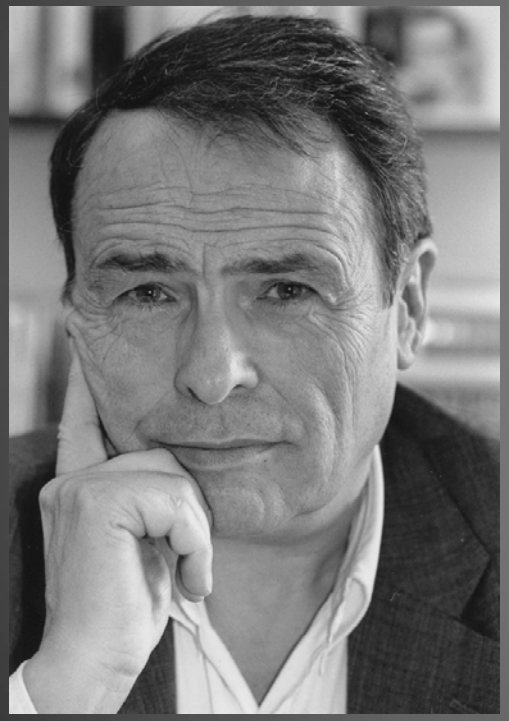
We must support free expression.

It’s not always clear which approach is better.

We really need to deliver both.

3. Sideways

Sideways design parallels the sociological concept of *Habitus*.



Freedom within structural constraints.

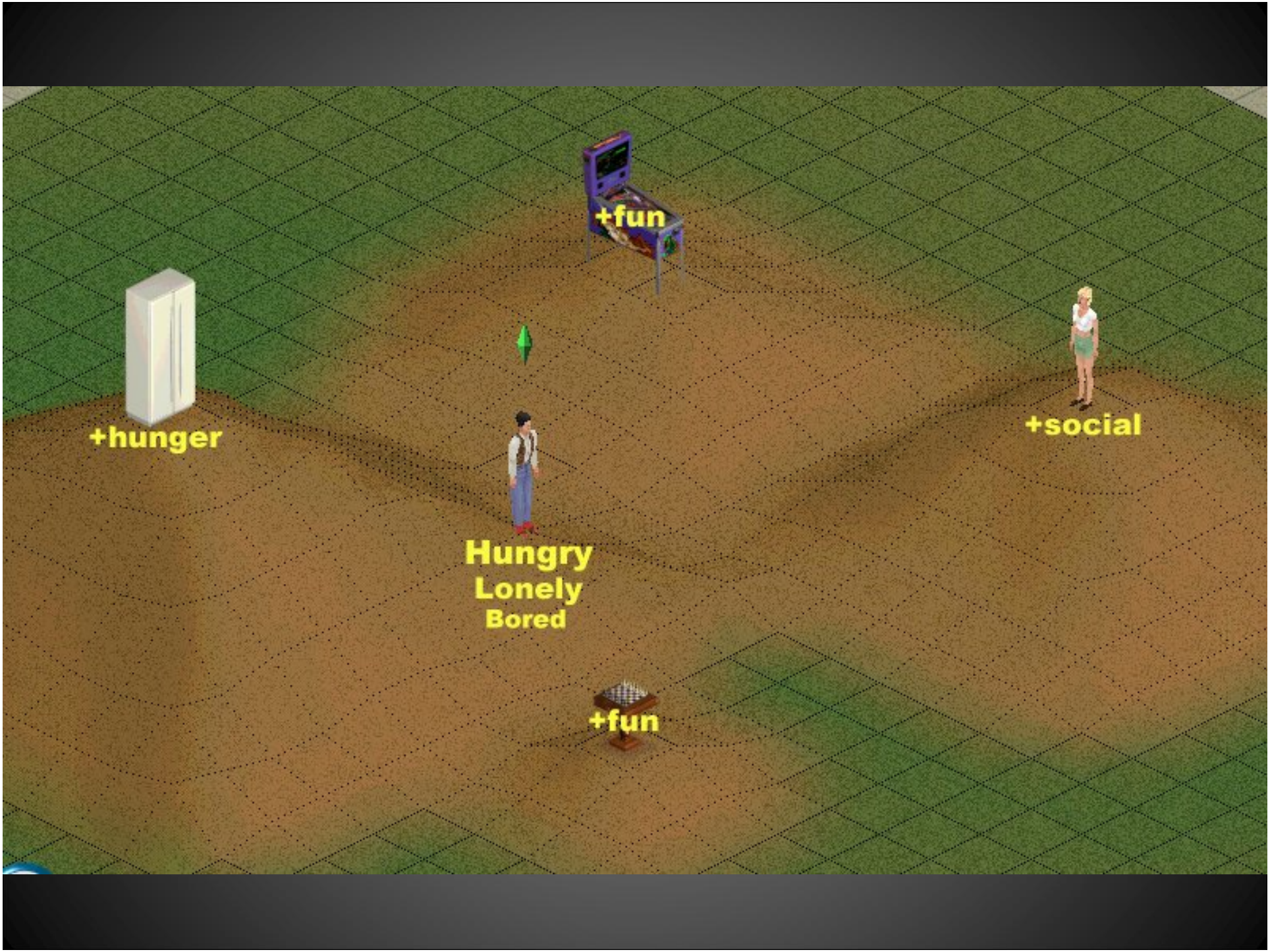
Game AI can echo the *Habitus*:

Draw no distinction between scripted and unscripted.

Focus on the problem of coordinating multiple NPCs.

Allow for layering of multiple scripts simultaneously.

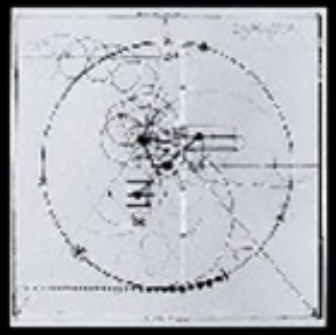
Make sure your content is declarative, not imperative.


+fun
+hunger
+social
Hungry
Lonely
Bored
+fun

Building a Better Battle

The Halo 3 AI Objectives System





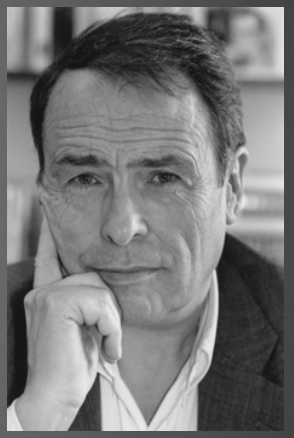
Damián Isla
Bungie Studios

Help

BUNGIE

Thanks!





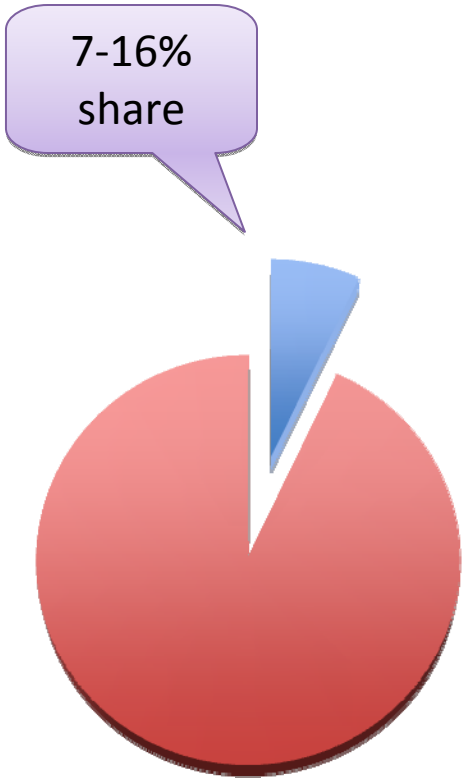
Adam Russell

a.russell@derby.ac.uk

Trials and Tribulations of AI Middleware

John Funge

(Most) Developers are not Rich



- Developers get a (relatively) small cut of game revenue
- And they make few games
- Publishers get a cut of many games
- Platform owners get a cut of every game!
  - Best middleware customers

Protecting IP is Controversial

- Developers want source code
- Developers want to know how it works
- Developers may implement techniques themselves after free evaluations
- Patents are controversial
  - Locally optimal, globally suboptimal
  - Don’t patent behind the scenes features
- No-one will want to credit or even acknowledge you

Indemnification is Pointless

- Everyone wants it
  - Delays and complicates contracts
- If there's a successful law suit, it's not meaningful
  - Middleware company has to declare bankruptcy
  - Someone still has to pay the damages
- In good conscience, it's impossible to provide
  - Submarine patents, number of patents
  - People give it anyway (little to loose)

Business Models

- Improve productivity
  - Middleware for streamlining existing techniques
  - Path planning is an example
- Provide new techniques
  - Ideally hard to re-create
  - Speech recognition, motion recognition, etc.
- No-one will want to give you royalties

Evaluations

- Developers often responsible for evaluations
  - Developers often wary, see middleware as a threat
  - Focus solely on (important) low-level details
  - (Understandable) skepticism about “hype”
  - Good for productivity based middleware
- Potential for new creativity rarely evaluated
  - What kind of new games could be made?

Quit Being Afraid of the Flippin' Floats

Kevin Dill

Lockheed Martin

AI Summit GDC 2010

AI GAME PROGRAMMERS GUILD

Basic Premise

- Things We Hope To See*
  - Compelling Characters
    - Characters you care about, that feel alive
  - Cunning Opponents
    - Outmaneuver & ambush you, cut lines of supply, etc.
    - No cheating on hard mode, “dumbed down” on easy
  - Intelligent Allies
    - Can operate independently, not micro-managed
    - See “Compelling Characters,” above
- We need to embrace float-based logic to achieve these things!

* From AI Game Programming Wisdom 4

Why not Boolean

- Lack of subtlety & nuance
  - If I see him, shoot at him
  - Unless my health is low, then run away
  - Unless his health is low, then stick it out
  - Unless…
- Predictability
  - Bad: always do the same thing
  - Better: fixed probability (run away 40% of the time)
  - Best: probability depends on situation

But… But… “Bucket of Floats”

- Common complaints (a.k.a. things that are not true)
  - Hard to think that way
  - Interdependencies make it hard to balance
- My response
  - Get over it
  - Beat your head on it until it “clicks”
  - Balancing is a skill – learn to be good at it
  - Yes there can be interdependencies – but that’s true of any approach

Isn’t It Slow?!

- Why would it be?
  - O(n)
  - Computers are good at math.
  - No branching!
- Minimal rethink on events
- Embarrassingly parallelizable

The End

The Pain and Suffering Caused by Custom Scripting Languages

Various Custom Scripting Languages

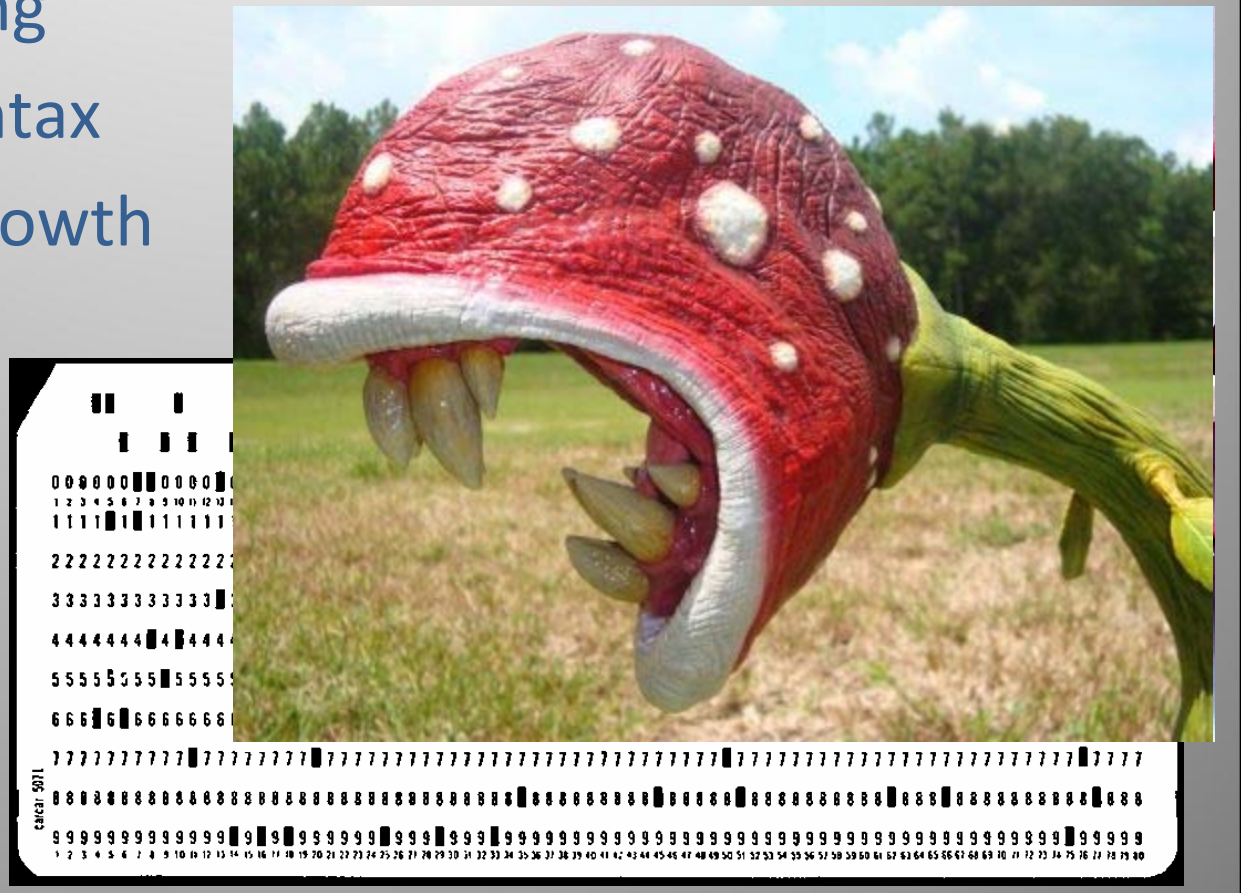
- ID Software: QuakeC
  - Quake series, Doom 3, games that use Quake Engine
- Epic Games: UnrealScript
  - Games that use Unreal Engine
- BioWare: NWscript
  - Neverwinter Nights series, The Witcher, Knights of the Old Republic series
- Lucas Arts: SCUMM (Script Creation Utility for Maniac Mansion)
  - 50+ games
- Maxis: Simantics
  - The Sims, The Sims 2
- Gas Powered Games: Skrit
  - Dungeon Siege, Dungeon Siege 2, Supreme Commander
- Naughty Dog: GOAL (LISP-like language)
  - Crash Bandicoot series, Jak and Daxter series
- WizBang Studio Productions: Adlib
  - Activision's HyperBlade, Microsoft Baseball series

The Promise

- Easier to write AI and better abstraction
  - Focused
  - Safer
  - Programming-lite
- Faster iteration
  - No compiling
  - Change script and reload

The REALITY

- Harder to author logic
  - Error reporting
  - Contrived syntax
  - Haphazard growth
  - No niceties



The REALITY

- Not stable
  - Never finished
  - Bugs not fixed
  - Not well tested

The REALITY

- Impossible to debug
  - No breakpoints, conditional breakpoints
  - No stepping
  - Reload is risky

```
printf("printf debugging for the win!");
```

The REALITY

- Game designers and everyone else on the team will HATE it

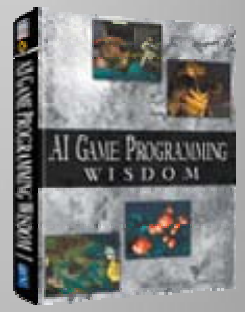
Evidence #1

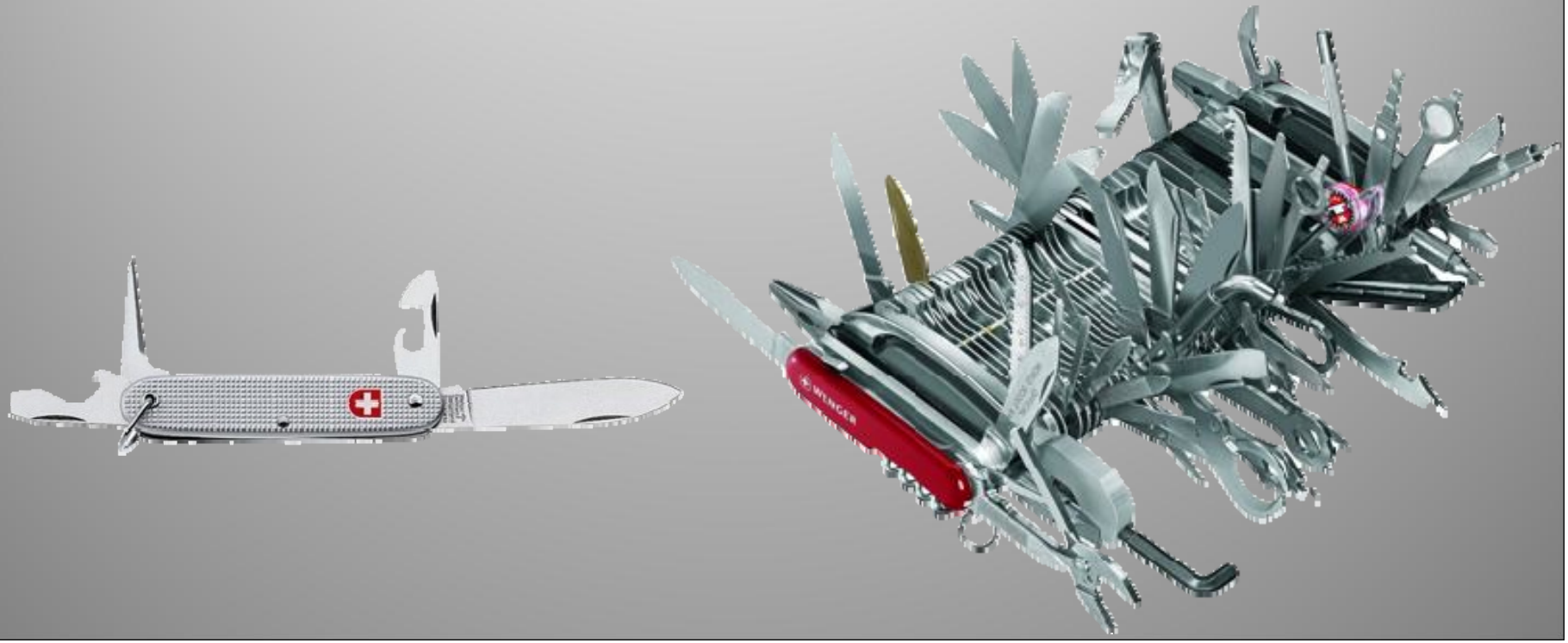
- **Stephen White, Programming Director, Naughty Dog**
  - Postmortem: Naughty Dog's *Jak and Daxter: the Precursor Legacy,* Gamasutra, 2002

- **“GOAL sucks!** While it's true that GOAL gave us many advantages, GOAL caused us a lot of grief. A single programmer (who could easily be one of the top ten Lisp programmers in the world) wrote GOAL. While he called his Lisp techniques and programming practices "revolutionary," others referred to them as "code encryption," since only he could understand them. Because of this, all of the support, bug fixes, feature enhancements, and optimizations had to come from one person, creating quite a bottleneck. Also, it took over a year to develop the compiler, during which time the other programmers had to make do with missing features, odd quirks, and numerous bugs. “

Evidence #2

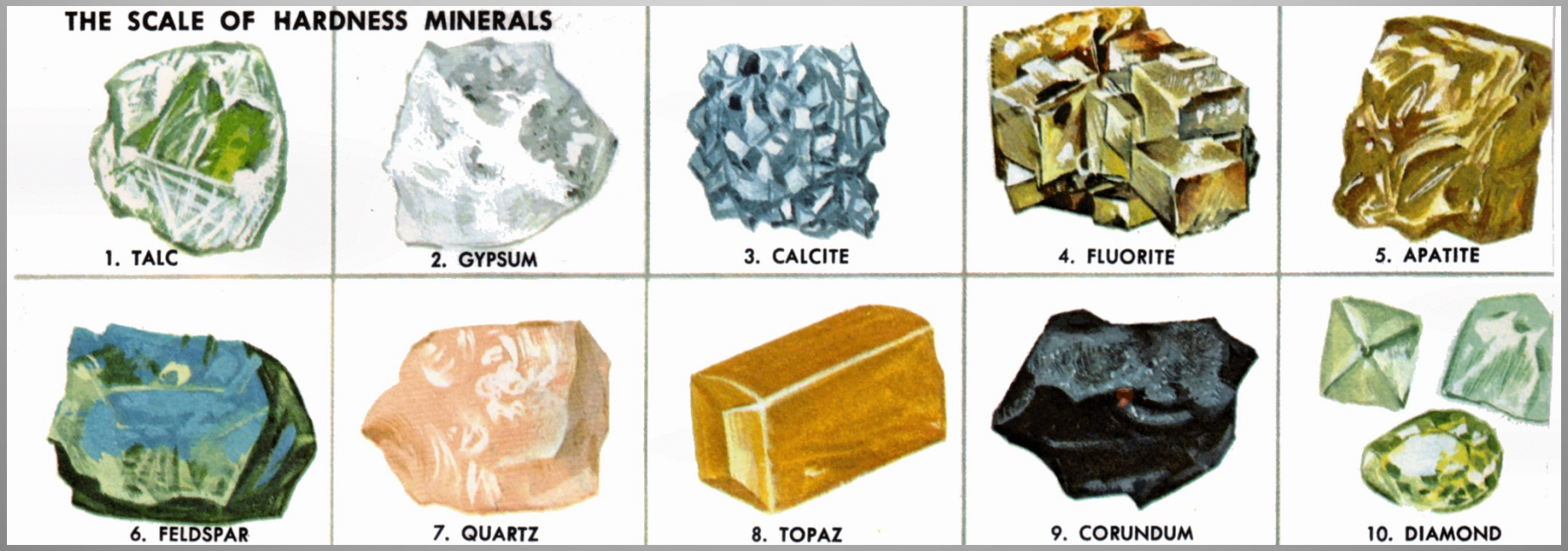
- How NOT to Implement a Basic Scripting Language
  - Mark Brockington, Mark Darrah (BioWare)





Language Design is HARD and takes TIME

- C++ has been around for 30 years
- Lua has been around for 17 years



Tool Design is HARD and Takes TIME

- Microsoft has been working on compilers and debuggers for 27 years
- Microsoft Visual C++ has been around for 17 years

Who the Hell Makes These Things? (There's always this one guy…)

Why do I care about this???

- 2 years of pain
- Hampered growth as a programmer early in career

- Lua is widely accepted and successful
- GTA Chinatown Wars was scripted by designers in C++

Go make new games, not new programming languages!!!

(unless you have 10+ years to get it right)

Dave Mark
Intrinsic Algorithm LLC

No Respect, I Tell Ya!

How I'm really annoyed at game reviewers.

(And how we respond to them.)

AI Summit GDC 2010

AI GAME PROGRAMMERS GUILD

If you can’t say something nice…

Google for: game AI review

If you can’t say something nice…

Google for:
bad game AI review

If you can't say something nice…

Bad

Retarded

Weak

Pathetic

Stupid

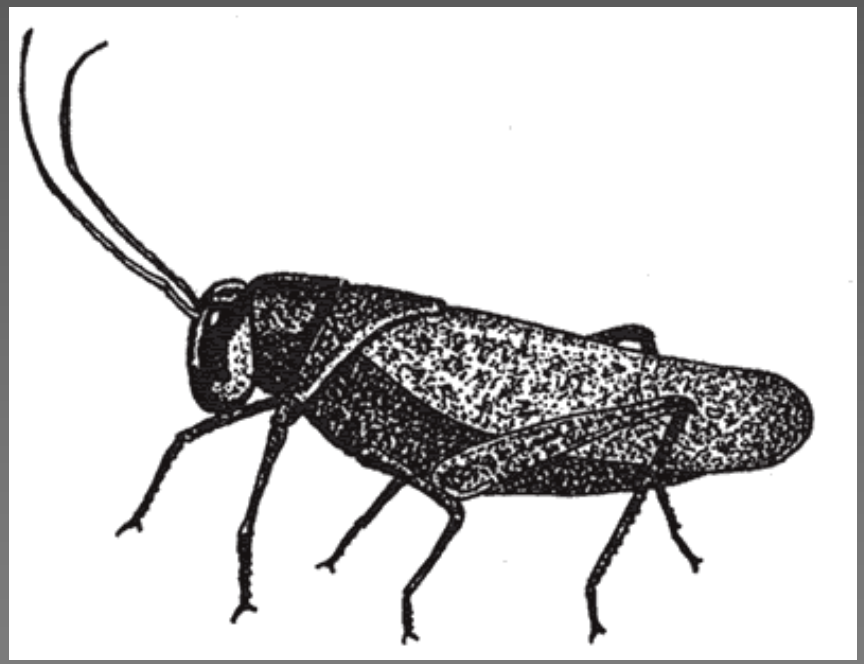
Annoying

Idiotic

If you can’t say something nice…

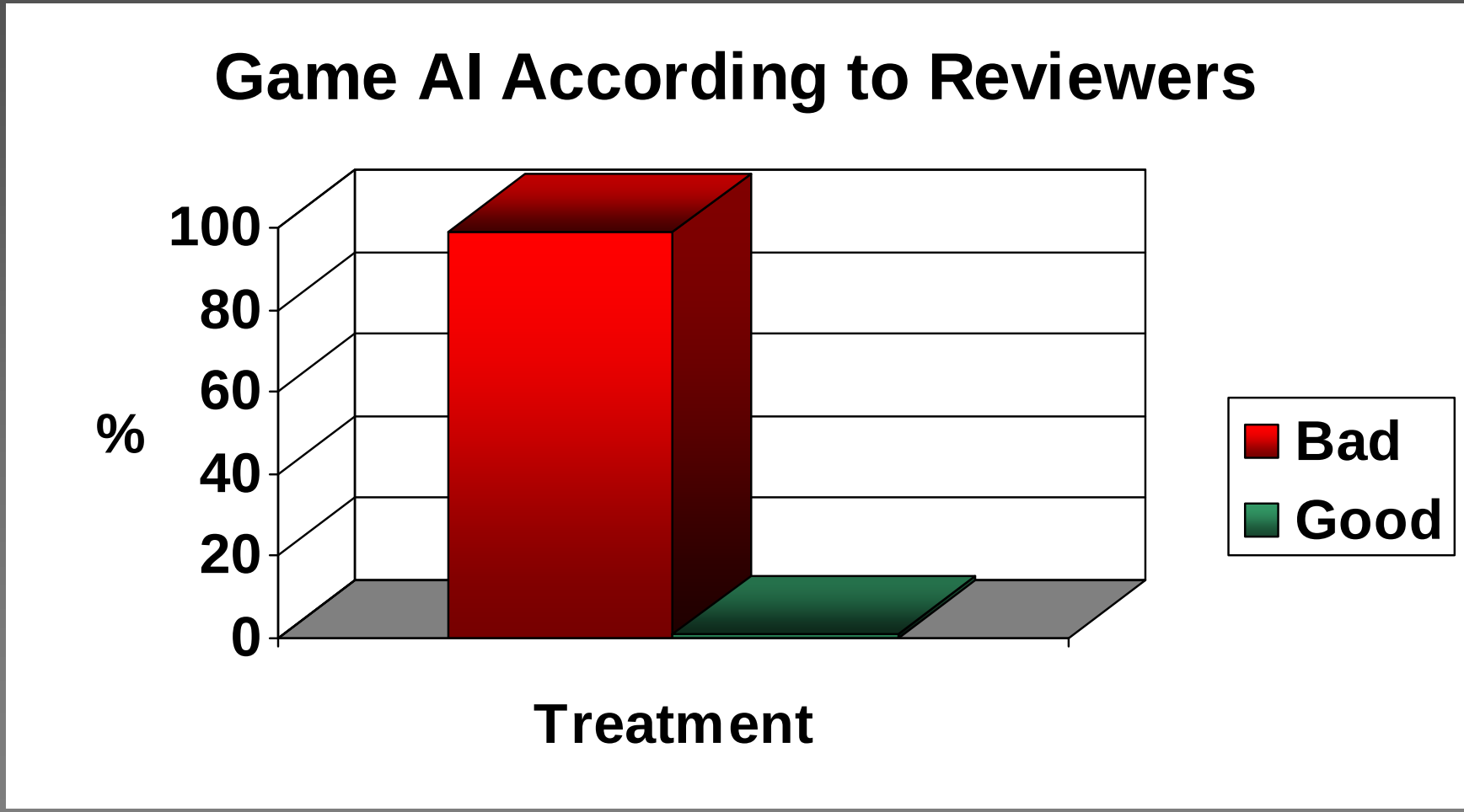
Google for:
good game AI review

If you can’t say something nice…

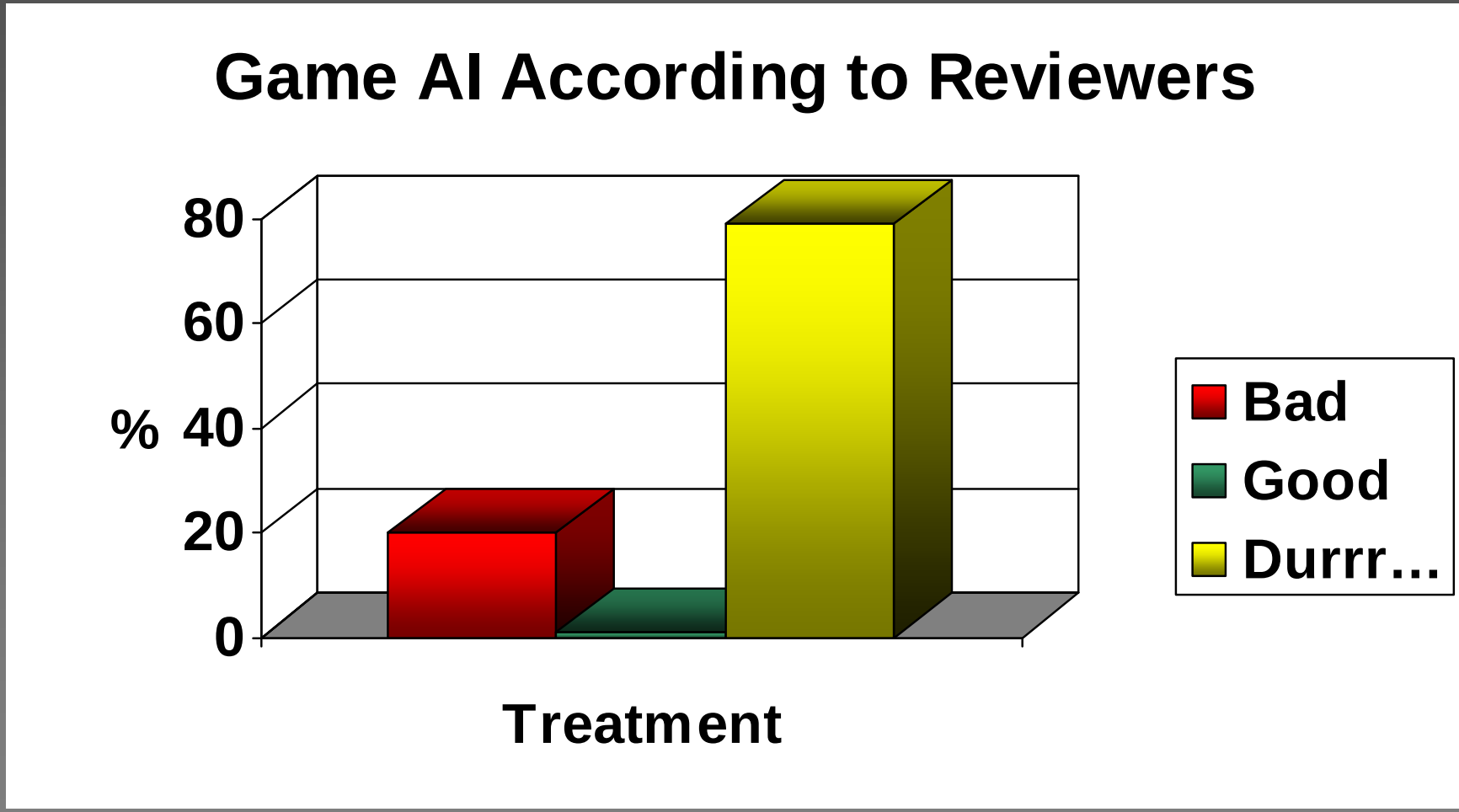


No wonder people prefer PvP

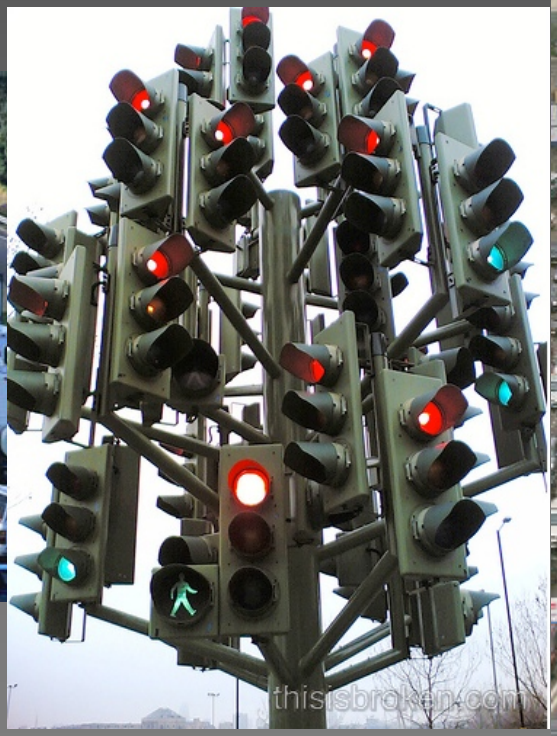
All I ever hear outta you is…



Excuse me? Helloooo…?



What is there to notice?



We don’t notice the typical stuff

We notice when things go wrong!

We notice when things are cool!

What *is* cool, though?

Game players
(and reviewers)
don't even realize
when we do things
that are cool!

Unless we tell them we are!

What *is* cool, though?

“The way it should be…”

The more our AI starts to look “natural” the less people notice it!

“Waaa! It’s too hard!”

The better our AI performs, the more people complain that it isn’t fun.

You can't have it both ways

"The AI just stands in the middle of the room waiting for me to shoot him in the head."

"It pisses me off that the AI hides so I can't shoot him!"

You can't have it both ways

"The AI walks straight towards me and doesn't show any sign of intelligent strategy or tactics."

"Damnit! It's no fair that the AI sneaks around next to me and snipes me in the head!"

You can't have it both ways

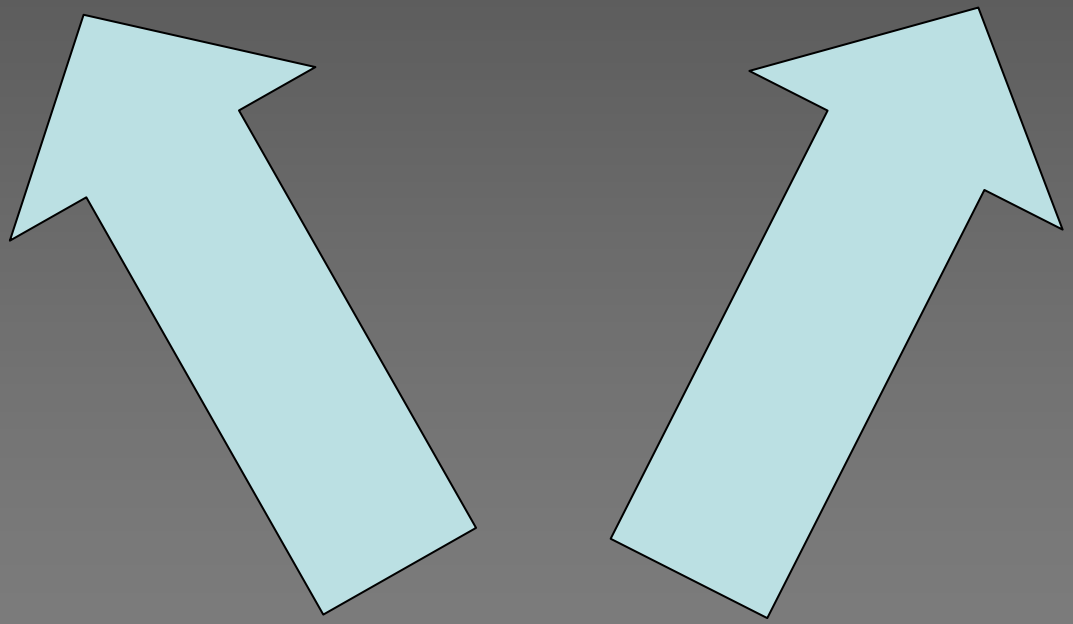
"The stupid AI doesn't seem to notice me when I enter a room and do [stuff] right in front of him."

"No matter what I do, the AI always sees me and opens fire… even when I'm hidden!"

Catch-22

Relegated to Obscurity

People Start Whining!



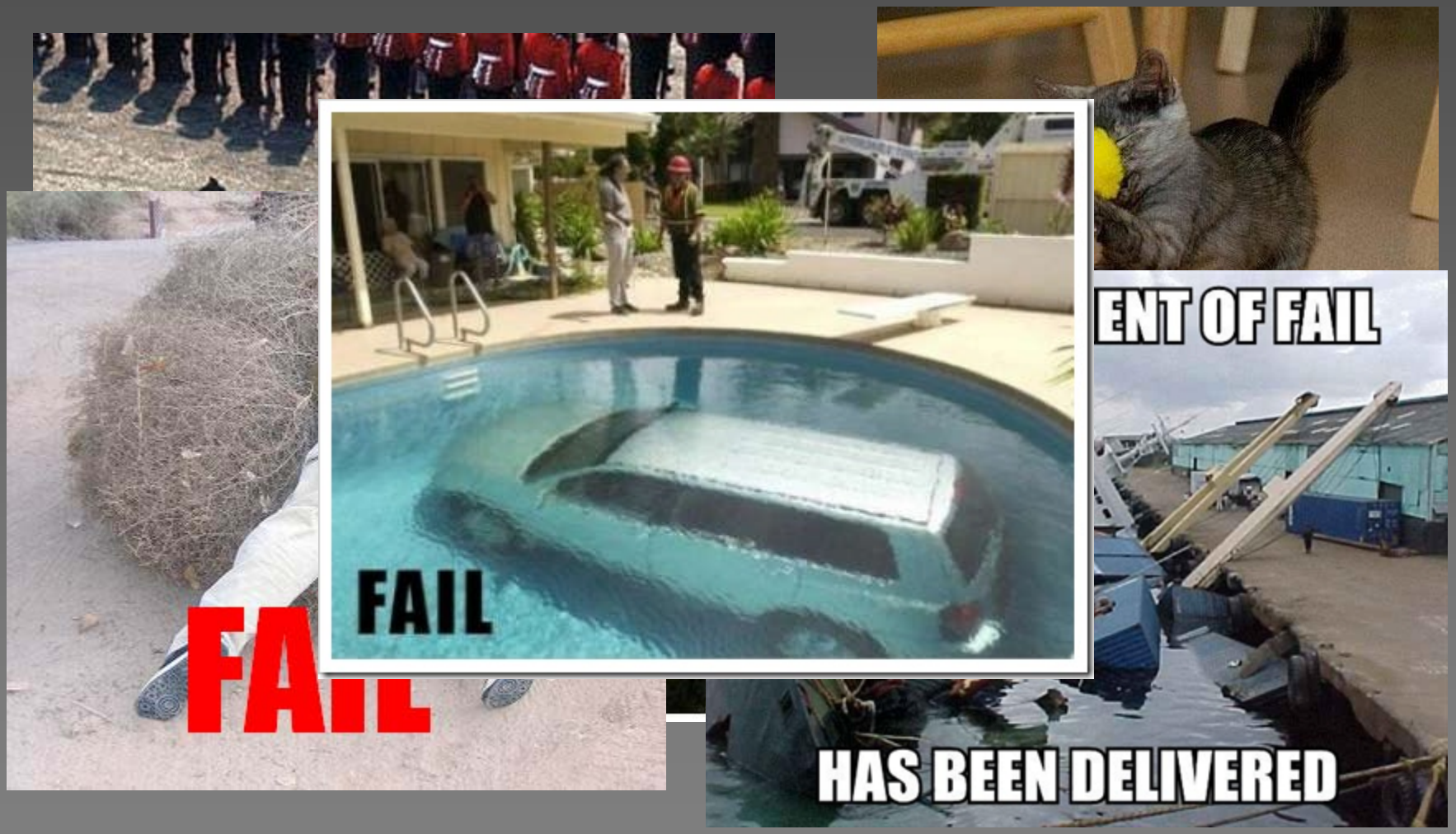
Improvement

Don’t be dumb…

The #1 rule of creating game AI is don’t let the agent do anything stupid.

Don’t be dumb…

Don’t be dumb…



Don’t be dumb…

The #1 rule of creating game AI is don’t let the agent do anything stupid.

Are we selling ourselves short?

Maybe if we started creating…

- More challenging AI
- More life-like AI
- AI that actually *did* flank the player
- AI that actually *did* run away
- AI that actually *did* try to hide and sneak

Eventually we might start to get some respect from the players, and the reviewers?

Don’t be dumb…

Perhaps since the public expects more from us…

… we should now expect more from ourselves?