Errâncias de um Software Brasileiro

Roberto Ierusalimschy

Pontifícia Universidade Católica
do Rio de Janeiro

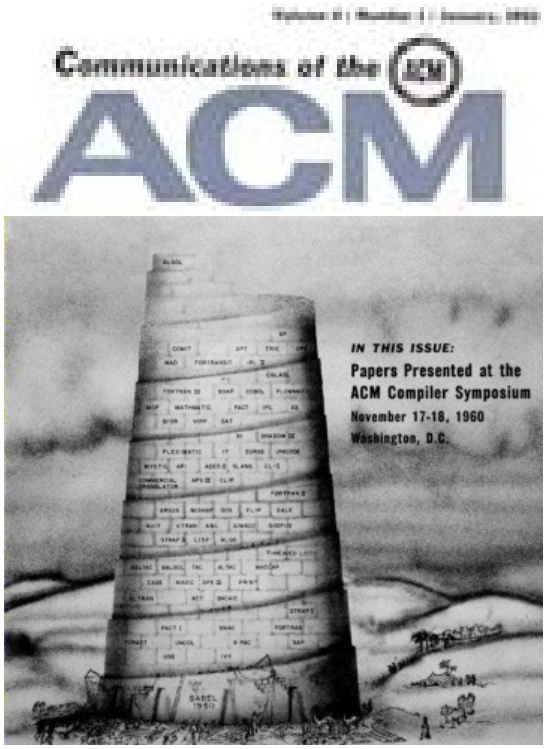
Linguagens de Programação

A mais ubíqua das ferramentas usadas na produção de software

A most important, but also most elusive, aspect of any tool is its influence on the habits of those who train themselves in its use. If the tool is a programming language this influence is, whether we like it or not, an influence on our thinking habits.

— *Edsger Dijkstra*

- Existem milhares de linguagens de programação no mundo
- Umas poucas dezenas tem uso mais difundido
  - C, C++, C#, Java, Python, Perl, Ruby, Javascript, PHP, Objective-C, Lua, Basic, Pascal, Fortran, Lisp, Scheme, Erlang, Prolog, Tcl, Ada, Haskell, R, …
- Lista continua crescendo



Capa da CACM, jan 1961

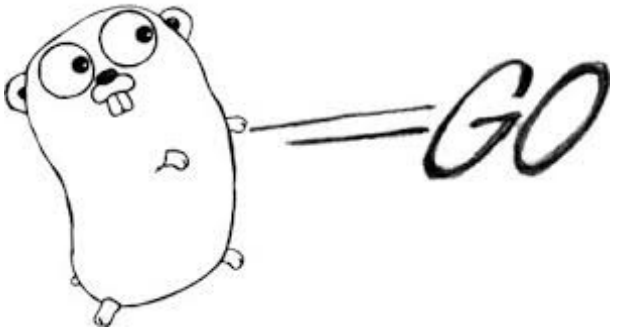
(Mozilla, 2010)

(MIT, 2012)

(Facebook, 2014)

(Apple, 2014)

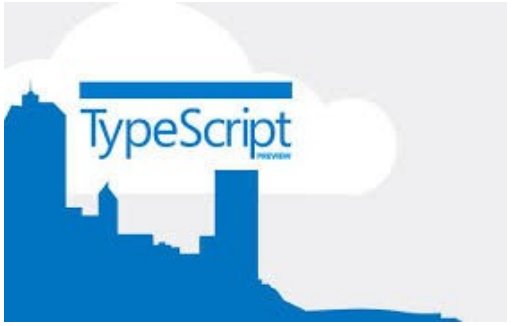
(Todos, 2015)



(Google, 2007)

(JetBrains, 2011)

(Google, 2011)



(Microsoft 2012)

A escolha de uma linguagem pode ser um enorme investimento para uma organização. Empresas como Adobe e Huawey têm mais de um milhão de linhas de código escritas em Lua.

A escolha de uma linguagem pode ser um enorme investimento para uma organização. Empresas como Adobe e Huawey têm mais de um milhão de linhas de código escritas em Lua.

Como essas e outras empresas vieram a investir tanto em uma linguagem desenvolvida na América do Sul?

O Começo de Lua: 1993

- Tecgraf - uma parceria entre a PUC-Rio e a Petrobras
  - forte cultura de desenvolvimento de ferramentas (herança da reserva de mercado)
- Dois programas com problemas de configuração
  - cada um com sua própria mini-linguagem
  - ambas as linguagens com grandes limitações

O Começo de Lua: 1993

- Idéia: uma linguagem de configuração *genérica*
  - Roberto Ierusalimschy, Waldemar Celes e Luiz Henrique de FIgueiredo
- Requisitos:
  - uma linguagem “completa”
  - facilmente acoplável
  - portátil
  - sintaxe não intimidante para usuários
    - engenheiros, geólogos, etc.

O que Havia?

- XML: não existia em 1993; não é uma LP.
- Perl: sintaxe muito complexa; não facilmente acoplável.
- Python: versões pré 1.0 em 1993; não facilmente acoplável.
- Tcl: única realmente feita para acoplar, mas “sintaxe” muito complexa para não programadores.

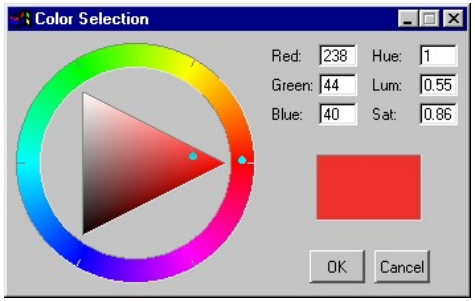
Lua 1.0 (1993)

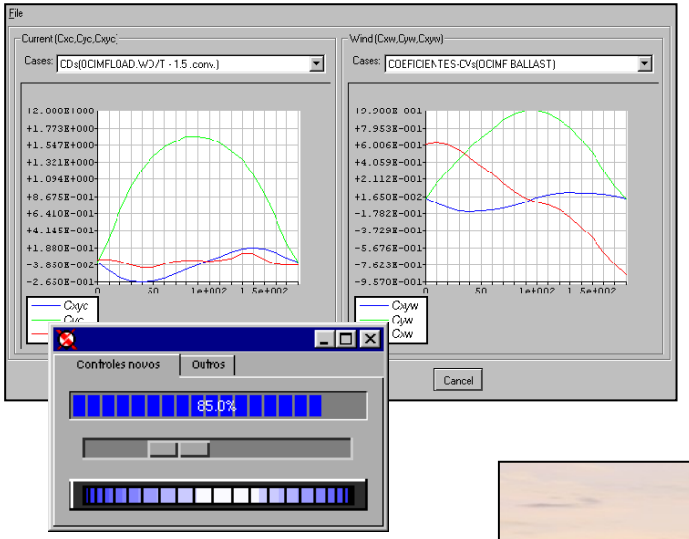
- Chamada “1.0” a posteriori
- “The simplest thing that could possibly work”
  - tabelas implementadas via listas!
- Implementação convencional
  - pré-compilador com lex/yacc
  - *opcodes* para uma máquina virtual de pilha
- Menos de 6.000 linhas de código C

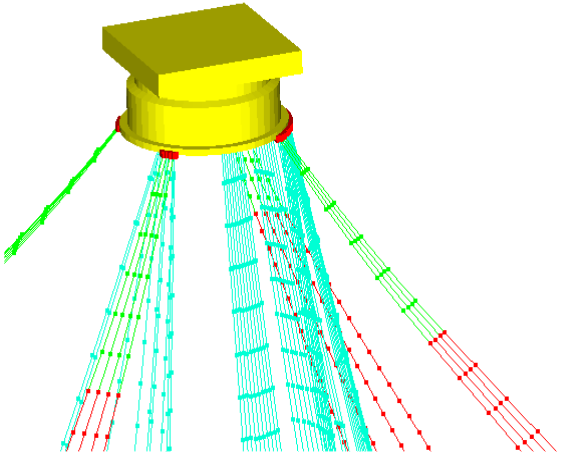
Lua 1.0

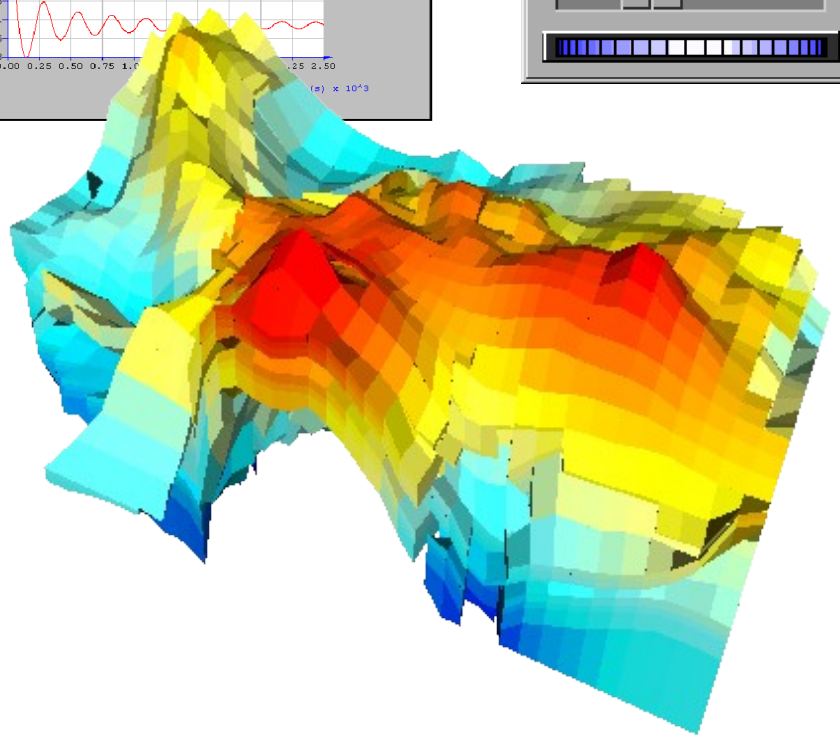
- Expectativas: solucionar nossos problemas com nossas aplicações.
  - poderia ser usada em outros projetos do Tecgraf
- Satisfez nossas espectativas!
  - usada com sucesso nas duas aplicações, por muitos e muitos anos
- Foi um grande sucesso no Tecgraf.

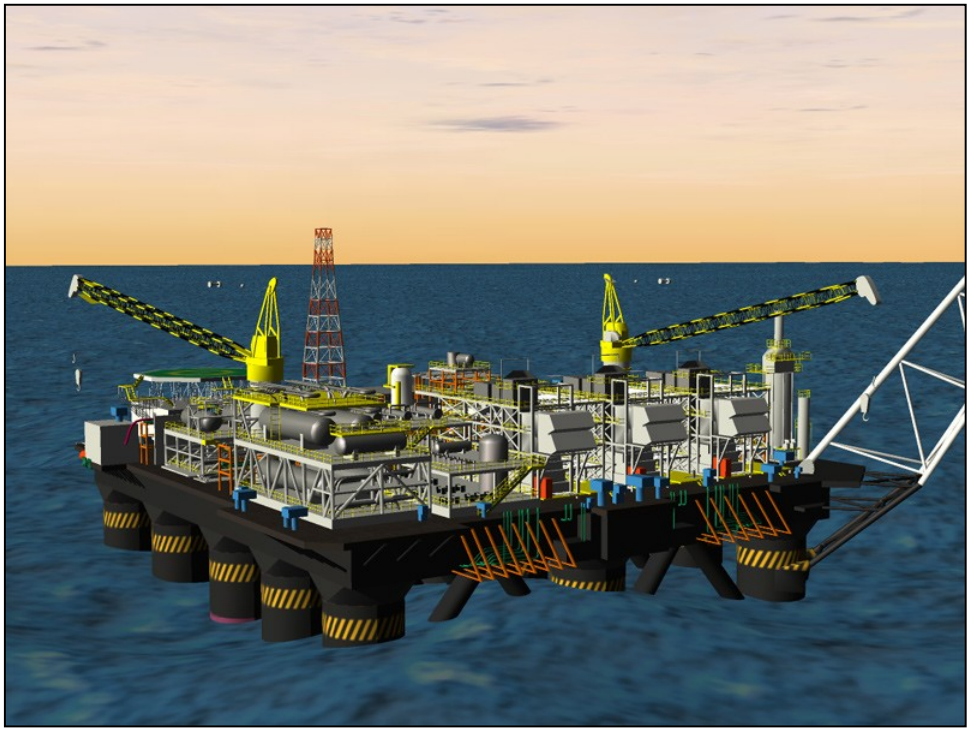
Logo, vários projetos no Tecgraf estavam usando Lua...











Lua 1.1 (1994)

- Já tinhamos dezenas de “usuários reais”.
- Novos usuários trazem novas demandas
  - desempenho
  - manual de referência
  - API Lua-C bem definida e bem documentada
- Nenhum uso fora do Tecgraf
  - “O que, uma linguagem brasileira?”
  - “Desenvolvida em uma universidade brasileira??”

Exposição Internacional

Newsgroups:
comp.compilers,comp.lang.misc,comp.programming,comp.lang.c
From: lhf@csg.uwaterloo.ca (Luiz H de Figueiredo)
Organization: Computer Systems Group, University of Waterloo
Keywords: tools, available
Date: Fri, 8 Jul 1994 11:51:45 GMT

This is the first public release of Lua.

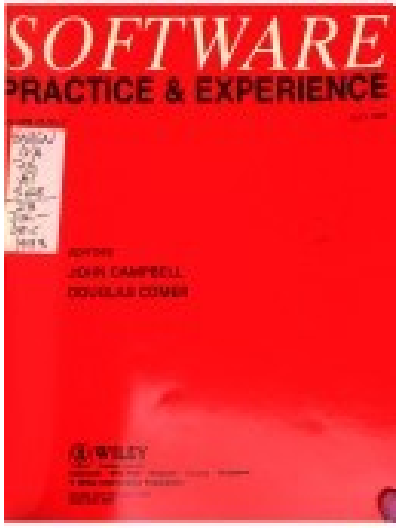
* What is Lua?
Lua is a simple, yet powerful, language for extending applications. Lua has been developed by TeCGraf, the Computer Graphics Technology Group of PUC-Rio, the Catholic University of Rio de Janeiro, Brazil. Dozens of industrial products developed by TeCGraf use Lua.
[…]

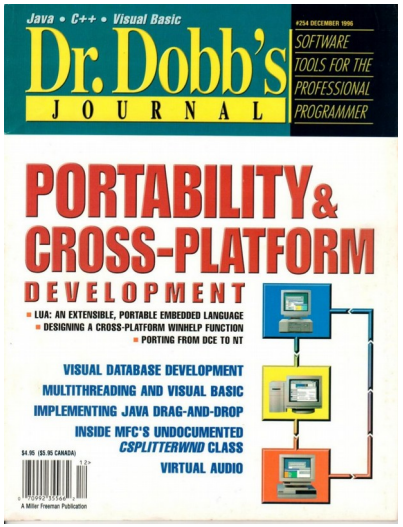
Lua 2

- Lua 2.1 (02/1995) até 2.5 (11/1996)
- Suporte para OO
  - delegação
- Biblioteca para casamento de padrões
- API para depuração

Primeiras Publicações Internacionais

- R. Ierusalimschy, L. H. de Figueiredo, W. Celes. Lua --- an extensible extension language. *Software: Practice & Experience,* 26(6):635-652, 1996.
- L. H. de Figueiredo, R. Ierusalimschy, W. Celes. Lua: an extensible embedded language. *Dr. Dobb's Journal*, 21(12):26-33, 1996.





Lua 3

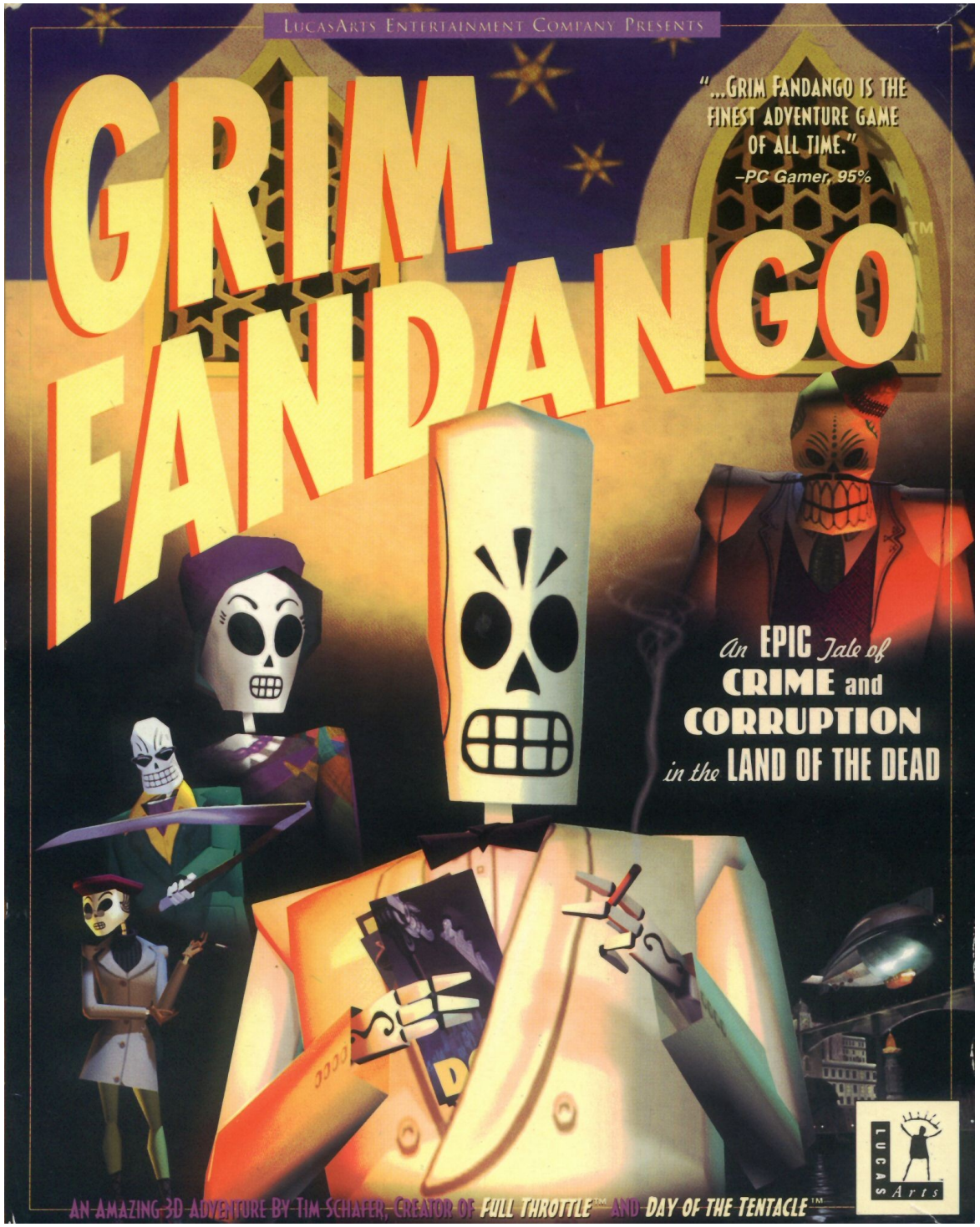
- Lua 3.0 (07/1997) até Lua 3.2 (07/1999)
- Melhor suporte a programação funcional
  - funções anônimas, *upvalues*
- Grande reorganização interna

Lua em Jogos (o começo)

From: Bret Mogilefsky <mogul@lucasarts.com>
To: '"lua@icad.puc-rio.br"' <lua@icad.puc-rio.br>
Subject: LUA rocks! Question, too.
Date: Thu, 9 Jan 1997 13:21:41 -0800

Hi there...
After reading the Dr. Dobbs article on Lua I was very eager to check it out, and so far it has exceeded my expectations in every way! It's elegance and simplicity astound me. Congratulations on developing such a well-thought out language.

Some background: I am working on an adventure game for the LucasArts Entertainment Co., and I want to try replacing our older adventure game scripting language, SCUMM, with Lua.



Lucas Arts, 1998

"Grim Fandango was the first game that shows Lua could not only be used to make a good game, but that it could be used to make some of the best games ever."

Diehard GameFAN: Hall of Fame Nomination

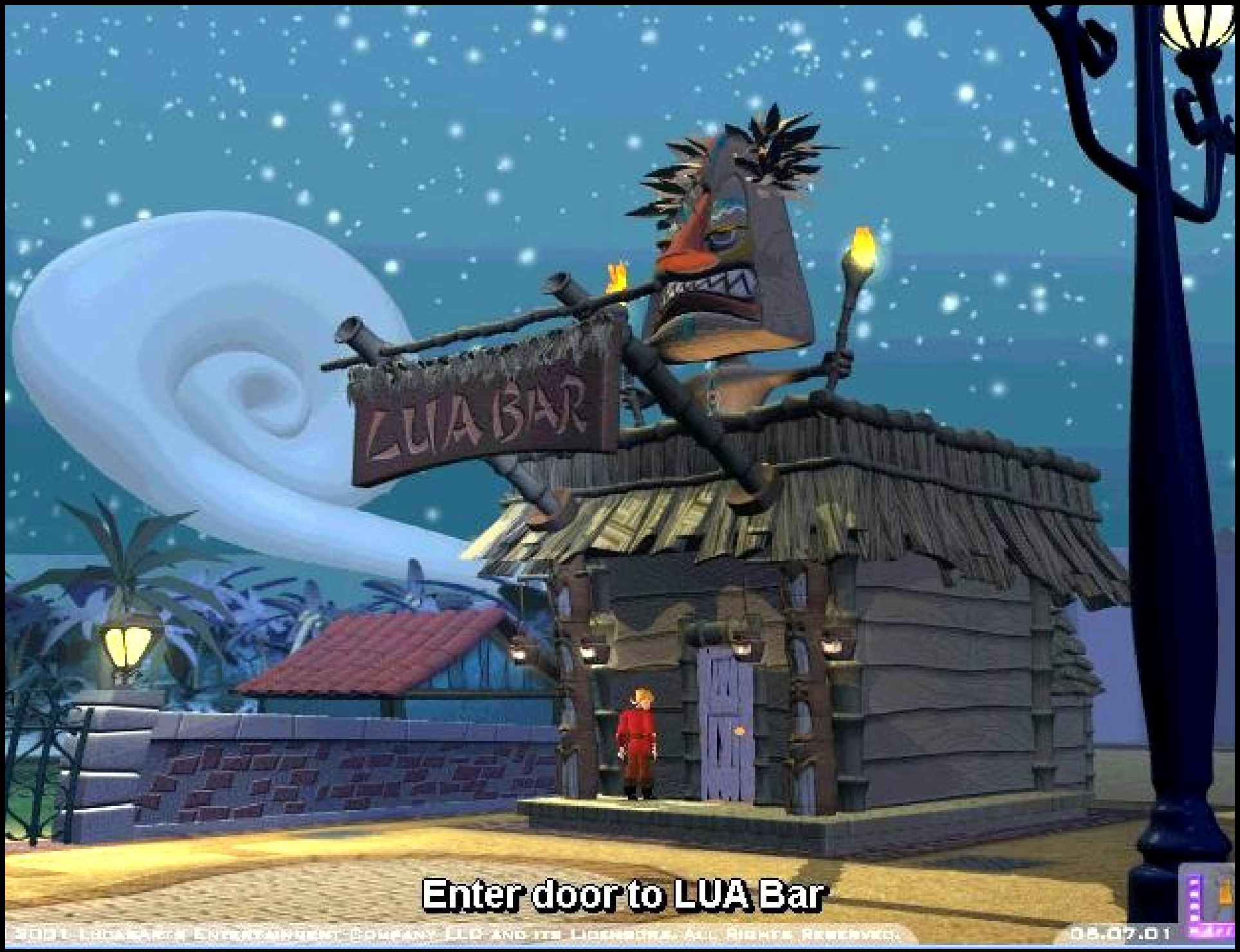
What happened next

- Game of the year...almost
  - Half-Life relegated us the Adventure Game of the Year
- GDC 1999 (2000?)
  - Panel discussion of scripting languages
    - Rob Huebner on embedding Java
    - Kevin Bruner on interpreted C++
    - Seamus McNally on not using a scripting language
  - 200 miserable people
  - “Or you could just use Lua...”
  - Furious scribbling

Bret Mogilefsky, Lua workshop 2005

Logo, vários outros jogos estavam usando Lua...

Escape from Monkey Island (2000)



Planos para Lua 4.1

- *multithreading?*
  - múltiplos personagens em jogos

Planos para Lua 4.1

- *multithreading?*
  - múltiplos personagens em jogos
- Co-rotinas!
  - implementação portátil
  - semântica determinística
  - co-rotinas + escalonador = *multithreading* não preemptivo

Planos para Lua 4.1

- Co-rotinas
- Novo algoritmo para tabelas
  - armazena arrays como arrays
- Funções de primeira classe com visibilidade léxica (“*closures*”)
- Máquina virtual baseada em registradores

Planos para Lua 4.1

- Co-rotinas
- Novo algoritmo para tabelas
  - armazena arrays como arrays
- Funções de primeira classe com visibilidade léxica
- Máquina virtual baseada em registradores

Muita coisa para 4.1...

Lua 5.0 (04/2003)

- “Maturidade” da linguagem
  - livro texto
- Co-rotinas
- Funções de primeira classe com visibilidade léxica
- Sistema de módulos
- Tabelas fracas, *proper tail calls*, etc.

2007: III ACM History of Programming Languages

Por quê Lua?

- Mais uma linguagem de script
- Mas com objetivos bem específicos
  - Ênfase em *scripting*
  - Portátil
  - Pequena
  - Simples
  - Eficiente

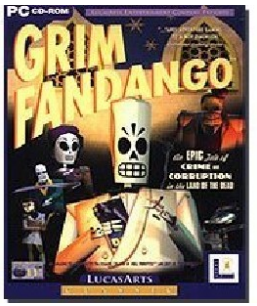
Scripting

- Linguagem de script x linguagem dinâmica
  - script enfatiza comunicação inter-linguagens
- Programa escrito em duas linguagens:
  - uma linguagem de script e uma de *sistema*
- Linguagem de sistema implementa as partes *hard* do programa:
  - algoritmos, estruturas de dados
  - poucas mudanças
- Linguagem de script conecta essas partes:
  - flexível, fácil de modificar

Lua e Scripting

- Lua é implementada como uma biblioteca
- Lua foi projetada para scripting
- Boa tanto para *estender* quanto para *embutir*
- Embutida em C/C++, Java, Fortan, C#, Perl, Ruby, Python, etc.

Scripting em Grim Fandango



“[The engine] doesn't know anything about adventure games, or talking, or puzzles, or anything else that makes Grim Fandango the game it is. It just knows how to render a set from data that it's loaded and draw characters in that set. […]

“The real heroes in the development of Grim Fandango were the scripters. They wrote everything from how to respond to the controls to dialogs to camera scripts to door scripts to the in-game menus and options screens. […]

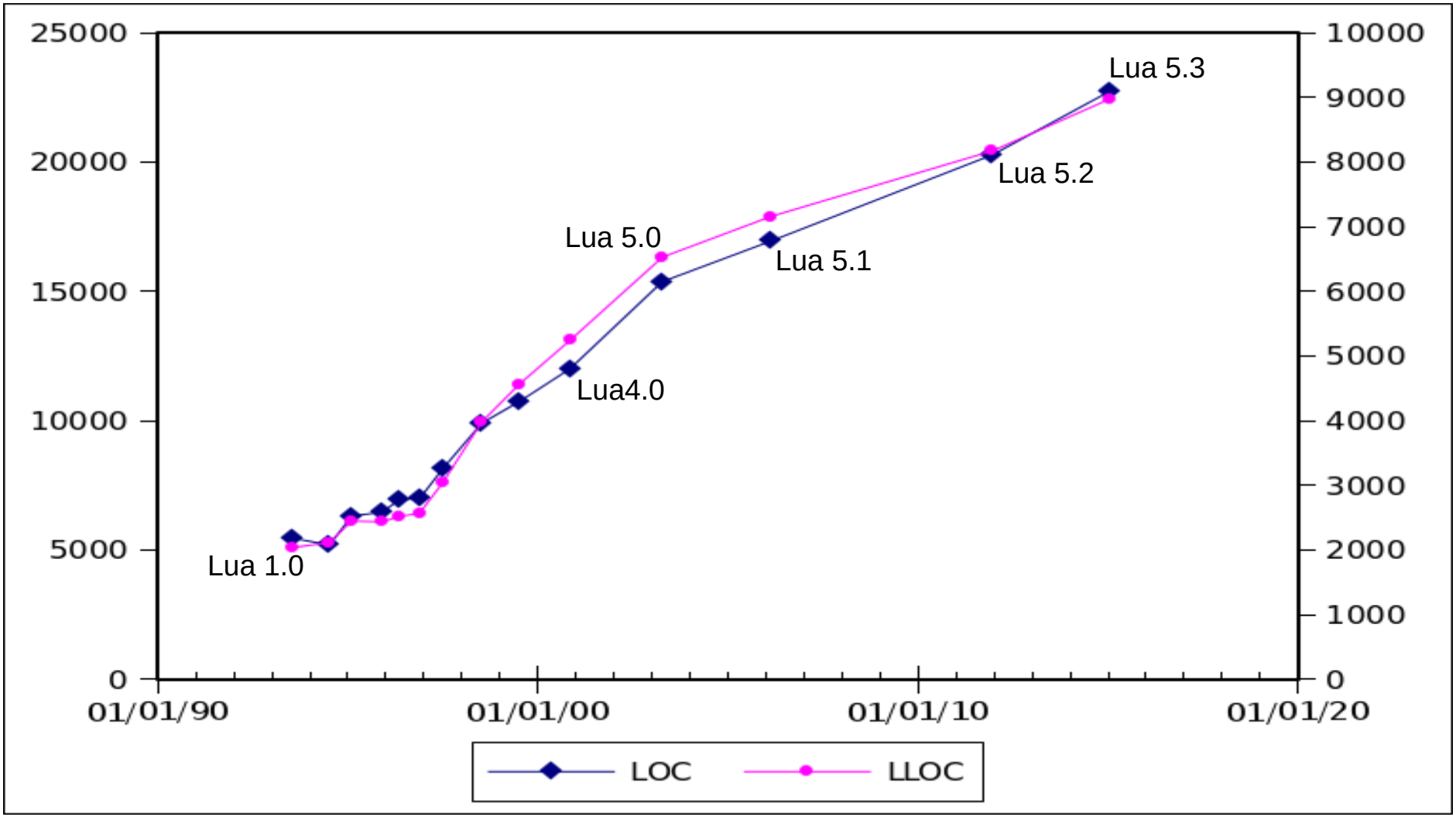
“A TREMENDOUS amount of this game is written in Lua. The engine, including the Lua interpreter, is really just a small part of the finished product.”

Bret Mogilefsky

Portabilidade

- Roda virtualmente em qualquer plataforma
  - Posix (Linux, BSD, etc.), OS X, Windows, Android, iOS, Arduino, Raspberry Pi, Symbian, Nintendo DS, PSP, PS3, IBM z/OS, etc.
  - escrita em ANSI C
- Núcleo é uma aplicação *free-standing*
  - Pode rodar dentro do núcleo do SO
  - Pode rodar direto no hardware, sem um SO

Tamanho

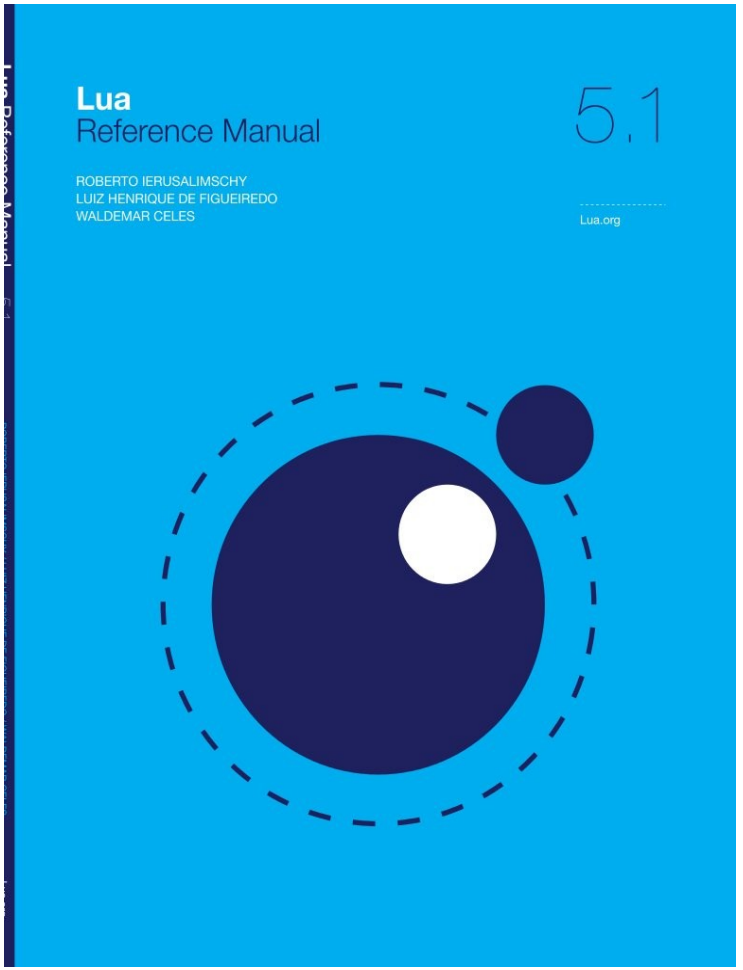

25000
20000
15000
10000
5000
0
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
01/01/90
01/01/00
01/01/10
01/01/20
Lua 1.0
Lua4.0
Lua 5.0
Lua 5.1
Lua 5.2
Lua 5.3
LOC
LLOC

Simplicidade

Manual de referência com 100 páginas

Documenta a linguagem, a API com C e as bibliotecas padrão

(lombada)



Simplicidade

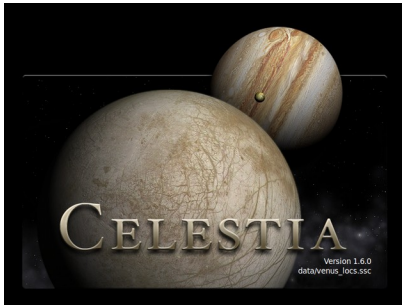
- Poucos mas poderosos mecanismos bem explorados.
- Arrays associativos
  - implementa todas as estruturas de dados (*maps*, arrays, estruturas, objetos, módulos, etc.)
- “Closures”
  - implementação eficiente e sem restrições.

Eficiência

- Vários *benchmarks* mostram Lua entre as linguagens mais rápidas no reino das linguagens dinâmicas
  - simplicidade
  - projeto da linguagem
  - inovações técnicas

Lua Hoje

- Scripting de aplicações
- Forte nicho em sistemas embarcados
- Forte nicho em jogos



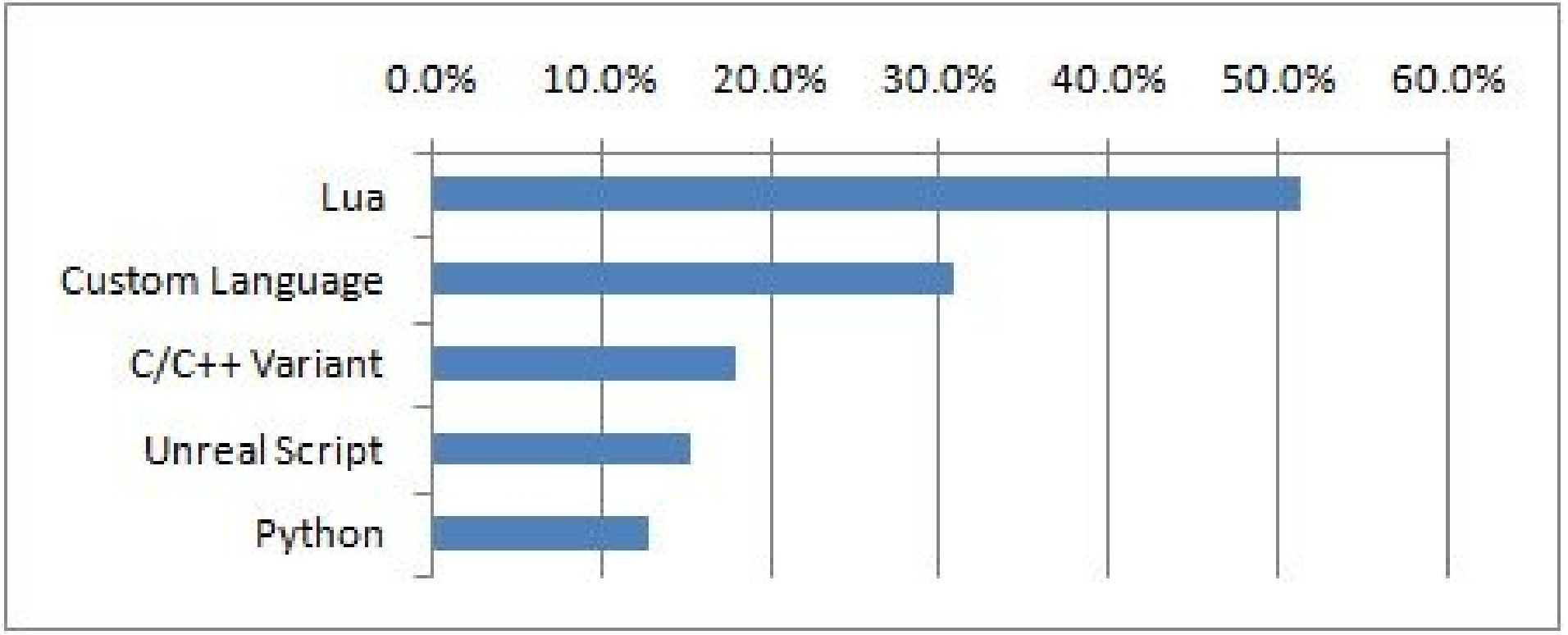

```
if not _params.STD then
  assert(loadstring(config.get("LUA.LIBS.STD")))()
  if not _params.table_ext then
    assert(loadstring(config.get("LUA.LIBS.table_ext")))()
    if not __LIB_FLAME_PROPS_LOADED__ then
      __LIB_FLAME_PROPS_LOADED__ = true
      flame_props = {}
      flame_props.FLAME_ID_CONFIG_KEY = "MANAGER.FLAME_ID"
      flame_props.FLAME_TIME_CONFIG_KEY = "TIMER.NUM_OF_SECS"
      flame_props.FLAME_LOG_PERCENTAGE = "LEAK.LOG_PERCENTAGE"
      flame_props.FLAME_VERSION_CONFIG_KEY = "MANAGER.FLAME_VERSION"
      flame_props.SUCCESSFUL_INTERNET_TIMES_CONFIG = "GATOR.INTERNET_CHE
      flame_props.INTERNET_CHECK_KEY = "CONNECTION_TIME"
      flame_props.BPS_CONFIG = "GATOR.LEAK.BANDWIDTH_CALCULATOR.BPS_QUEU
      flame_props.BPS_KEY = "BPS"
      flame_props.PROXY_SERVER_KEY = "GATOR.PROXY_DATA.PROXY_SERVER"
      flame_props.getFlameId = function()
        if config.hasKey(flame_props.FLAME_ID_CONFIG_KEY) then
          local l_1_0 = config.get
          local l_1_1 = flame_props.FLAME_ID_CONFIG_KEY
          return l_1_0(l_1_1)
        end
        return nil
        end
```

Sistemas Embarcados

Samsung (TVs), Cisco (roteadores), Technicolor (vários), Logitech (teclados), Volvo (paineis de carros), Mercedes, Olivetti (impressoras), Ginga (middleware para TV digital), Verison (*set-top boxes*), Texas Instruments (calculadoras), Huawei (celulares), Sierra Wireless (dispositivos M2M), NodeMCU (IoT), …

Lua em Jogos

- *The Engine Survey* (Mark DeLoura, 03/02/09,Gamasutra)
  - linguagens de script



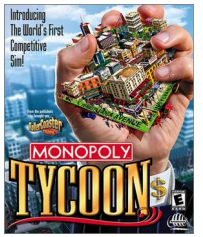

Games for Windows
COMPANY of HEROES

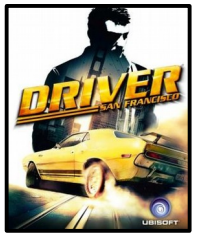

Games for Windows
UNIVERSE AT WAR
SEGA


Games for Windows
DUNGEONS


Tap Tap
Revenge


BLITZKRIEG

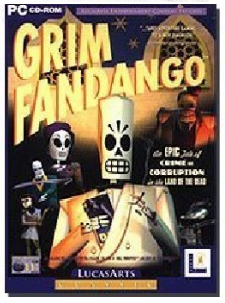

Introducing The World's First Competitive Sim!
MONOPOLY
TYCOON

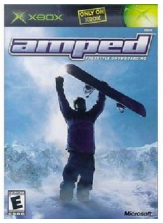

DRIVER
SAN FRANCISCO
UBISOFT


RōBLOX


Games for Windows
The Lord of the Rings Online
SHADOWS OF ANGMAR

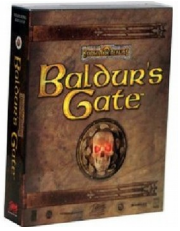

REQUIEM
MEMENTO MORI

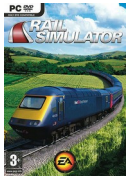

GRIM FANDANGO
LUCASARTS

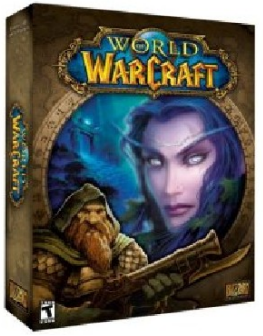

amped

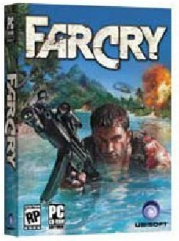

XBOX 360
HULK
SEGA


PlayStation.2
DESTROY ALL HUMANS!
ONE GIANT STEP ON MANKIND


Baldur's Gate


RAIL SIMULATOR


WORLD of WARCRAFT

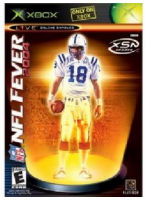

FARCRY
UBISOFT

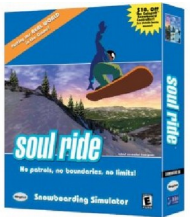

IMPOSSIBLE CREATURES


STALKER


DINER DASH 2
Restaurant Rescue
Flo is back & better than ever!


Blossom


XBOX
NFL FEVER

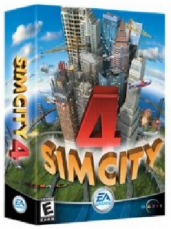

soul ride
Snowboarding Simulator


NATURAL SELECTION 2


ALADDS
ONLINE


XBOX
Links

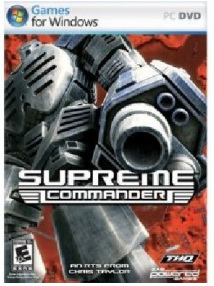

BRÜTAL LEGEND


SIMCITY 4


ANGRY BIRDS
PLAY


SpellForce


NAPOLEON
TOTAL WAR
16
SEGA


Games for Windows
SUPREME COMMANDER


BATTLE ZONE
98 REDUX


XBOX 360
TOPSPIN 2


XBOX 360
SONIC
THE HEDGEHOG


BUZZ!
Quiz TV


Games for Windows
THE WITCHER


XBOX 360
CRACKDOWN


CRIMSON
YEARN PIRATES

MDK2

Games for Windows
CRYSIS

Games for Windows
SUPREME COMMANDER
Forged Alliance

HOMEWORLD2

PlayStation.2
PSYCHONAUTS

Bubble
Ball

THE INCREDIBLES

STAR WARS
EMPIRE AT WAR

PlayStation.
Home

There

Games for Windows
DUNGEONS

PlayStation.2
BRAVE

The Guild 2

ESCAPE FROM MONKEY ISLAND

EMPIRE
TOTAL WAR
16+
SEGA

PC DVD
CIVILIZATION

plus+

XBOX 360
CRACKDOWN

MIGHT AND MAGIC
HEROES V

PC DVD
DAWN WAR
DARK CRUSADE

YOU ARE EMPTY

XBOX 360
FABLE II

E Agora?

- “Eterno” conflito: flexibilidade x desempenho
  - linguagens estáticas x linguagens dinâmicas
- Algumas soluções:
  - compiladores JIT
  - scripting
  - gradual typing

JIT (the good)

- Alguns compiladores JIT para linguagens dinâmicas conseguem resultados bastante promissores.
  - Particularmente em *benchmarks.*
- Linguagem sem modificações.

JIT (the bad)

- Vantagens são excludentes.
- Desempenho é imprevisível.
- Várias técnicas típicas de linguagens dinâmicas sem suporte (“optimization killers”).
- No fim, acabamos programando em C com uma sintaxe diferente.
- Implementação extremamente complexa.

Scripting (the good)

- Uso de duas linguagens:
  - uma linguagem de script e uma de “sistema”
- Partes de desempenho crítico escritas na linguagem de sistema.
- Une desempenho da linguagem de sistema com flexibilidade da linguagem de script.

Scripting (the bad)

- Alta impedância entre as linguages.
- Impedância estática: linguagens muito diferentes
  - tradução de código
- Impedância dinâmica: custo alto de comunicação entre as linguagens
  - tradução de dados

Tipagem Gradual (the good)

- Programa pode ser gradualmente tipado.
- Partes mais “estáticas”, com desempenho crítico, recebem tipos.
  - mudanças mínimas no código
- Continuum entre partes estáticas e dinâmicas.

Tipagem Gradual (the bad)

- Difícil de otimizar.
- Sistemas de tipos complexos, para abarcar toda a linguagem.

Proposta: uma *companion language*

Pallene (yet another language)

- Subconjunto de Lua + tipos
- Compilação convencional
- Compartilhamento do *runtime* de Lua
  - representação de dados, coletor de lixo
- *Foreign Function Interface* para C.

Pallene (the good)

- Compilação simples
- Baixa impedância
  - no código, idealmente apenas adicionar tipos
  - opera diretamente em dados Lua
- Desempenho previsível
- Compatível com distribuição Lua padrão
- Sistema de tipos simples
  - abarca apenas o que é eficiente

Pallene (the bad)

- Mais uma linguagem para aprender.
- Desempenho sub-ótimo se comparado a C.
  - representação de dados
  - tipagem na interface
- ???
  - (a ser descoberto)

Perspectiva

- Linguagens são a base de praticamente todo software.
- Linguagens evoluem.
- Existe um conflito permanente entre compatibilidade e evolução.
- Linguagens de programação são “a arte do possível”.

PONTIFÍCIA
UNIVERSIDADE
CATÓLICA
DO RIO DE JANEIRO

errância
substantivo feminino.
Atributo, estado, particularidade ou circunstância de errante.

errante
adjetivo.
Que anda sem destino; característica do que ou de quem erra; que vive a vaguear.