

O Mundo dos Atores: uma perspectiva de introdução à programação orientada a objetos

Antonio Carlos Mariani
Departamento de Informática e de Estatística
Universidade Federal de Santa Catarina
mariani@inf.ufsc.br
<http://www.inf.ufsc.br/~mariani>

Resumo

Este artigo apresenta o *Mundo dos Atores*, uma ferramenta que propicia ao aprendiz um método gradativo para a exploração/apropriação dos conceitos da programação orientada a objetos (POO). Ele é particularmente indicado como ferramenta prática para uma disciplina de introdução à programação em cursos de computação/informática. Mas por ser um ambiente aberto e extensível, ele possibilita a exploração de várias outras matérias (como paralelismo, protocolos, sistemas distribuídos) que transcendem o conteúdo clássico de uma disciplina introdutória de programação.

Abstract

This article presents the World of the Actors, a tool that propitiates the apprentice a step-by-step method for exploration/appropriation of the concepts of object oriented programming (OOP). It's particularly indicated as an easy-to-use tool for an introductory discipline of computer programming in computer science courses. As an open and expandable environment, it facilitates the exploration of several other themes (such as parallelism, protocols, and distributed systems) that transcend the classic contents of an introductory course of programming.

Introdução

Diferentes abordagens têm sido utilizadas ao longo dos anos em disciplinas de introdução à programação em cursos de computação/informática. Estas abordagens, em geral, implicam a escolha de um paradigma de programação (a exemplo de: programação funcional, programação em lógica, programação estruturada e programação orientada a objetos), assim como de uma linguagem e de um ambiente computacional que suporte este paradigma. A opção discorrida neste texto é a de utilizar a programação orientada a objetos (POO).

A POO foi concebida nos final dos anos 70, mas somente nos anos 90 passou a ser amplamente utilizada como metodologia para o desenvolvimento de software. São bem conhecidas as vantagens da POO sobre a clássica programação estruturada. Contudo, o processo de familiarização com a POO não é simples, especialmente para os estudantes que estão ingressando num curso de computação/informática. São vários conceitos que precisam ser compreendidos e praticados de forma concomitante, num quadro de aparente complexidade. O aprendizado da POO demanda um tempo razoável, pois o aprendiz deve familiarizar-se com:

- a) uma perspectiva de desenvolvimento de software,
- b) um ambiente de programação;
- c) uma linguagem de programação;
- d) uma hierarquia de classes composta de dezenas ou centenas de classes.

É interessante notar que na POO os quatro itens citados acima são extremamente relacionados e interdependentes; nas palavras de Tadao Takahashi, é como se o aprendiz tivesse que conhecer tudo antes de conhecer qualquer coisa. A escolha, portanto, de uma boa linguagem/ambiente é crucial para favorecer o processo de aprendizagem. Também é igualmente relevante dispor-se de uma método que favoreça este processo, na expectativa de que o aprendiz possa explorar o ambiente/linguagem e se apropriar dos conceitos da POO de forma gradativa e concomitante.

Com esta perspectiva em mente, foi iniciada, em 1995, a definição e implementação do *Mundo dos Atores*, cuja proposição inicial era a de servir como ferramenta prática para uma disciplina de introdução à POO em cursos de computação/informática. Sua potencialidade, contudo, superou as expectativas, passando a oferecer um ambiente riquíssimo para a implementação de modelos e simulações que apresentam um grau razoável de complexidade. Por ser um ambiente aberto e extensível, o *Mundo dos Atores* possibilita também a exploração e o aprendizado de outros conceitos (como paralelismo, protocolos, sistemas distribuídos) que transcendem o conteúdo clássico de uma disciplina introdutória de programação.

Os tópicos que se seguem apresentam o *Mundo dos Atores* e descrevem suas características e potencialidades. Os dois últimos tópicos preocupam-se em apresentar algumas linhas de trabalho que dão andamento a este projeto, assim como as conclusões que já podem ser formuladas.

O Mundo dos Atores

O aprendiz inicia o seu trabalho com o *Mundo dos Atores* num ambiente similar ao oferecido pela linguagem LOGO (Seymour Papert). Através de um conjunto de comandos como “*caneta anda: 50*” ou “*caneta gira: 30*”, o aprendiz pode controlar uma caneta (representada pela seta na figura 1), fazendo-a traçar diferentes desenhos no “palco” (área gráfica) onde ela está inserida.

As ações *#anda:* e *#gira:* mencionadas acima fazem parte do repertório de ações básicas conhecidas pelo ator caneta. Contudo, novas ações podem facilmente ser adicionadas a este repertório, a exemplo das ações apresentadas nas figuras 2, 3 e 4, que fazem parte do trabalho de desenhar o palhaço da figura 1.

A tarefa de construção de uma nova ação é em muito facilitada pela existência de um editor que oculta uma série de detalhes não relevantes para o iniciante.

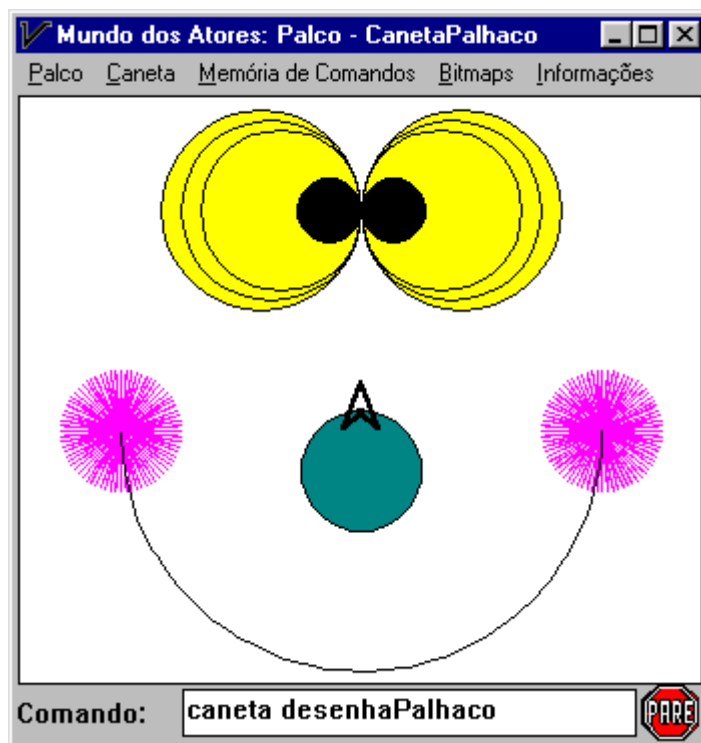


Figura 1

```
desenhaPalhaco
  "Desenha o rosto de um palhaço"

  caneta desenhaOlhos;
  desenhaNariz;
  desenhaBoca.
```

Figura 2

```
circunferência: raio
  "Desenha uma circunferência conforme
  o raio passado como parâmetro"

  llado!
  lado := (2 * Float pi * raio) / 36.
  caneta gira: 5.
  36 vezesRepita: [
    caneta anda: lado;
    gira: 10
  ].
  caneta gira: -5.
```

Figura 3

```
desenhaOlhos
  "Desenha os olhos do palhaço"

  caneta semRastros;
  anda: 100;
  fixaCorDosRastros: Preto.
  2 vezesRepita: [
    caneta desenhaUmOlho;
    gira: 180.
  ].
  caneta semRastros;
  anda: -100;
  comRastros.
```

Figura 4

Ao interagir com o ator *caneta*, o aprendiz estará exercitando todo um conjunto de noções básicas de programação de computadores, a exemplo das noções de algoritmo, variáveis, parâmetros, estruturas de controle (sequência, seleção, repetição, recursão), modularização/refinamento e reusabilidade. Mas por estarem inseridas num ambiente de POO (Smalltalk, neste caso), todas estas noções estão completamente ajustadas a esta perspectiva. Mesmo construções sintáticas simples como “*caneta anda: 50*” são relevantes, pois introduzem o mecanismo básico de comunicação por troca de mensagens inerente à POO. Neste exemplo, dir-se-ia que o objeto *caneta* recebe a mensagem *anda:* tendo o valor 50 como parâmetro.

A continuidade da exploração dos recursos do *Mundo dos Atores* conduz à apresentação e ao exercício de vários outros conceitos da POO, como herança e polimorfismo. A separação das ações escritas para um determinado fim conduz à noção de especialização e, junto com ela, às noções de herança e polimorfismo. Na figura 5, é apresentada uma situação onde foram definidas três novas canetas (*CanetaFlor*, *CanetaPalhaco* e *CanetaRelogio*), que são especializações da *Caneta*. Na *CanetaPalhaco*, por exemplo, são descritas apenas as ações específicas para o desenho do palhaço (figura 6). Contudo, a *CanetaPalhaco* continua fazendo tudo o que a *Caneta* faz, ou seja, herda todas as características da *Caneta*.

Figura 5

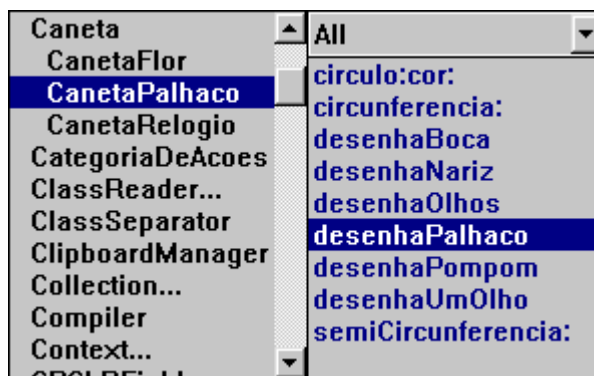


Figura 6

Numa analogia com um teatro, o lugar onde os atores atuam é o *Palco* (área gráfica da figura 1). De forma similar a um palco real, o palco do *Mundo dos Atores* pode ser modificado para se adequar a uma nova situação. Esta modificação é efetivada pela definição de um palco mais específico, a exemplo do *PalcoDosAtores* (figura 7), no qual podem ser inseridos vários outros atores além da *Caneta*.

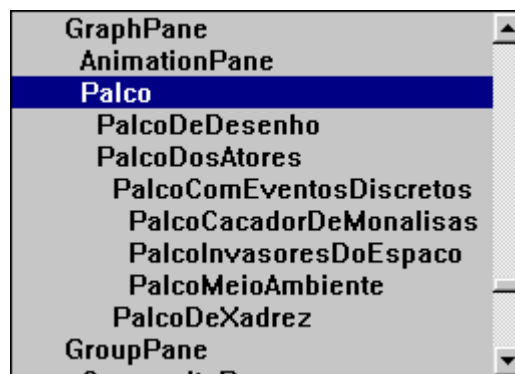


Figura 7

Assim como o palco, os atores também podem ser especializados (figura 8). No caso do *AtorComEventosDiscretos*, por exemplo, cada ator pode estar associado a uma espécie de roteiro que indica seu papel no palco. A definição de papéis possibilita o estabelecimento de comportamentos diferenciados para os atores, em particular no que se refere à movimentação no palco e à interação com outros atores. Iniciado o processo de animação, cada ator passa a “desempenhar” seu papel de forma independente. O comportamento geral do sistema emerge, então, dos comportamentos individuais de cada ator.

O *PalcoComEventosDiscretos* é que implementa o processo de animação descrito acima, conhecido por *modelo de eventos discretos*. Segundo este modelo, a cada unidade de tempo discreto, os atores executam atômicamente seus papéis, gerando uma ilusão de paralelismo.

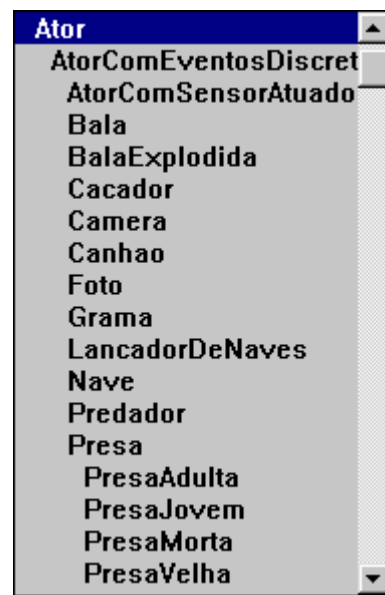






Figura 8

Para ilustrar o potencial pedagógico desta proposta, considere o ecossistema chamado de *PalcoMeioAmbiente* (figura 10), habitado por :

-  predador
-  presa adulta
-  presa jovem
-  grama

O papel do ator *grama* (figura 9), por exemplo, consiste em reproduzir-se a cada 300 unidades de tempo e morrer quando atingir 750 unidades de tempo de vida.

DefinePapel

"Define o papel da grama"
|gramal

```
self cada: 300 faca: [
    grama := Grama new.
    self palco adiciona: grama.
    grama
        pulaPara: self posicao;
        apontaPara:
            (self aleatorio: 360);
        anda: self tamanho x.
].
```

```
self em: 750 faca: [self destroi].
```

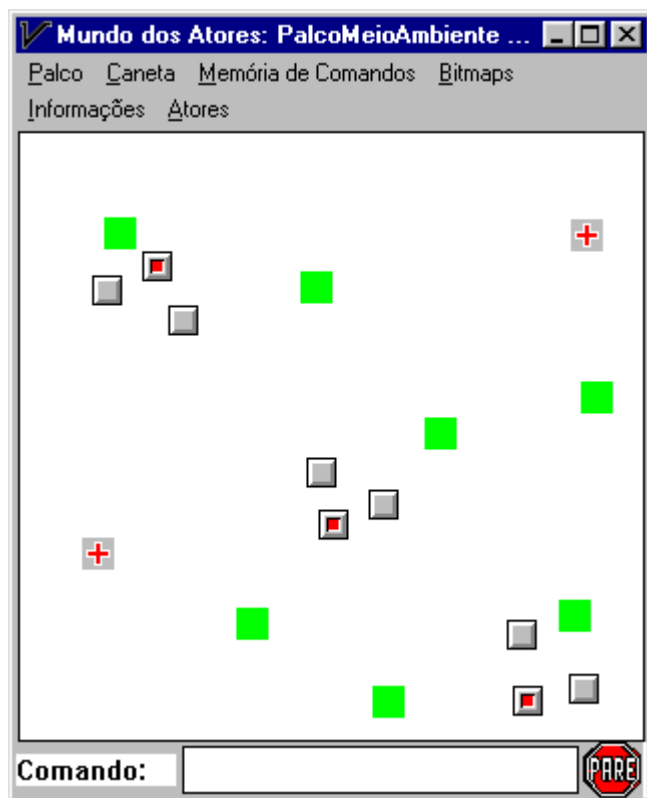


Figura 9

Figura 10

Os papéis dos demais atores são ligeiramente mais complexos, a exemplo do papel da *PresaJovem* (figura 11) e do *Predador* (figura 12).

```
definePapel
  "Define o papel de um presa jovem "
  | distancial
  self sempreQue: [mae isNil or: [mae palco isNil]] faca: [mae := self adultoMaisProximo].
  self sempreFaca: [
    mae isNil ifTrue: [
      self gira: (self aleatorio: 90) - 45; anda: 5.
    ] ifFalse: [
      (distancia := self distanciaPara: mae) > 15 ifTrue: [
        self apontaPara: mae; anda: (distancia - 15 min: 13)
      ].
    ].
  ].
  self em: 1000 faca: [ self tornase: PresaAdulta new].
```

Figura 11

O comportamento de uma *PresaJovem* (figura 11) é o de adotar uma *PresaAdulta* como mãe e segui-la aonde ela for. Caso não tenha mãe (verificada pela condição “mae isNil”), ela anda aleatoriamente (sem rumo) pelo ambiente. Após 1000 unidades de tempo, ela passa a ser uma *PresaAdulta*.

A palavra *self* que aparece nos textos é um elemento sintático da linguagem Smalltalk, que representa uma auto-referência para o próprio ator (objeto).

```
DefinePapel
  "Define o papel de um predador "
  | joveml
  “Inicialmente o predador tem um certo estoque de energia (nivelDeAlimentacao). A cada unidade de tempo discreto, este nível é reduzido em uma unidade”
  self sempreFaca: [ nivelDeAlimentacao := nivelDeAlimentacao - 1].

  “O nivelDeAlimentacao chegou a zero; ele morre”
  self sempreQue: [nivelDeAlimentacao = 0] faca: [self destroi].

  “O nivelDeAlimentacao é menor que 700; ele procura caça. Cada presa que ele caça aumenta o seu nível de alimentação”
  self sempreQue: [nivelDeAlimentacao < 700] faca: [self caca].

  “Após 3000 unidades de tempo, ele morre”
  self em: 3000 faca: [self destroi ].

  “De tempos em tempos, se reproduz, somente se estiver alimentado”
  self cada: 1050 faca: [
    nivelDeAlimentacao > 400 ifTrue: [
      jovem := Predador new.
      self palco adiciona: jovem.
      jovem pulaPara: self posicao.
    ].
  ].
```

Figura 12

Características do Mundo dos Atores

Há uma série de características no *Mundo dos Atores* que o tornam um ambiente muito rico para exploração e exercício dos conceitos de POO. Estas características são:

a) Metáfora de Palco: no *Mundo dos Atores*, o aprendiz trabalha com elementos virtuais como a caneta, os atores e o palco, os quais têm correspondência direta com elementos concretos equivalentes. Desta forma, desde o início, o aprendiz estará manipulando e falando sobre *objetos*, mesmo que esta noção não lhe tenha sido formalmente apresentada. Como consequência, é possível começar atividades práticas para exploração/exercício de conceitos da POO em laboratório tão cedo quanto se deseje.

b) Aberto e expansível: o palco mais simples do *Mundo dos Atores* (figura 1) oferece funcionalidade similar à disponível em implementações da linguagem LOGO (Seymour Papert), ou seja, uma área gráfica (o palco) e uma tartaruga (a caneta). Mas, divergindo de LOGO, o *Mundo dos Atores* está estruturado de forma a permitir que o aprendiz possa definir outros atores e colocá-los em interação, como também criar outros palcos particulares, a exemplo do *PalcoMeioAmbiente* (figura 10). Esta possibilidade de construção progressiva de modelos cada vez mais complexos é o ponto básico para a exploração sistemática dos conceitos da POO.

c) Integração com um ambiente de POO: a versão atual do *Mundo dos Atores* está implementada na linguagem/ambiente Smalltalk-V/Win. Seguindo a filosofia de completa integração de aplicações/ferramentas oferecida por este ambiente, o *Mundo dos Atores* pode ser facilmente integrado com qualquer outra aplicação disponível. Isto significa que o aprendiz não está restrito aos recursos oferecidos pelo *Mundo dos Atores*, e sim livre para utilizar todos os recursos oferecidos por Smalltalk. Ele é, na realidade, uma aplicação que faz uso dos recursos do ambiente Smalltalk, inclusive de sua linguagem de programação, que é utilizada para descrever o comportamento dos atores e definir novos palcos. Esta característica é deveras desejável, pois permite uma passagem gradativa e progressiva para o desenvolvimento de sistemas computacionais completamente independentes do *Mundo dos Atores*, a exemplo de aplicações comerciais.

Potencialidades Atuais

As características descritas acima abrem um conjunto razoável de possibilidades para a utilização do *Mundo dos Atores*, que vão muito além do seu objetivo inicial de servir como ferramenta de introdução à POO. Para identificar suas potencialidades, considere o ecossistema definido pelo *PalcoMeioAmbiente* (figura 10).

Por trás do *PalcoMeioAmbiente* há um modelo muito simples e intuitivo de paralelismo baseado em eventos discretos: em cada unidade de tempo, cada ator executa de forma atômica o seu papel, ou seja, as atividades previstas para cada unidade de tempo. Este modelo é suficientemente poderoso para que se construa facilmente um ecossistema que contenha presas e predadores, e mostre a dinâmica de comportamento do ecossistema em função de variações na cadeia alimentar.

O modelo de eventos discretos abre margem para rápida implementação de problemas que envolvam paralelismo, justificando sua introdução já como tema de uma disciplina introdutória de programação. Em particular, pode-se trabalhar com temas lúdicos como o clássico jogo dos ‘invasores do espaço’, no qual as naves passam no céu enquanto o usuário comanda um canhão que dispara torpedos para derrubá-las. A proposição de temas lúdicos mostra-se estimulante para o aprendiz, levando-o a explorar mais e mais os recursos do ambiente e, desta forma, favorecendo a compreensão de novos conceitos.

Alguns outros exemplos de problemas que foram (ou estão sendo) implementados segundo este modelo são:

- Um sistema de comunicação entre antenas: as antenas são atores que trocam pacotes contendo informações. Nesse caso, também se entra em contato e se constroem noções básicas de protocolos e de redes de comunicação. Alternativamente, e de forma similar, pode-se trabalhar com computadores em substituição a antenas.
- Simulação de Modelos Econômicos: as famílias e as empresas são atores atuando num palco que simula um mercado de bens, serviços e capitais.
- Modelo de tutores: apresenta um palco que pode ser povoado com tutores (atores capazes de estabelecer conversa em linguagem natural a respeito de algum tema). Além de trocar mensagens escritas com o usuário, os tutores são capazes de interagir uns com os outros e alterar seus estados, em atenção às mensagens do usuário.

É importante salientar que o modelo de eventos discretos descrito acima é apenas um dos possíveis modelos de paralelismo que se pode implementar. Por ser aberto e extensível, o *Mundo dos Atores* possibilita a implementação de diferentes modelos. Para tanto, é necessário implementar um novo tipo de palco que introduza o modelo desejado. Os atores podem também ser especializados para atender aos requisitos necessários para “atuar” neste novo palco.

Perspectivas Futuras

Há, atualmente, várias linhas de trabalho dando andamento a este projeto. Por um lado, trabalha-se na preparação de um material instrucional baseado em HTML que suporte a atividade prática que vem sendo desenvolvida com o *Mundo dos Atores*, tanto no nível da graduação como no da pós-graduação. Os primeiros resultados podem ser vistos em <http://www.inf.ufsc.br/poo/atores>.

Uma segunda linha de ação consiste em portar o *Mundo dos Atores* para VisualWorks, uma versão de Smalltalk que pode ser executada em diferentes plataformas além do MS-Windows. Esta atividade já foi iniciada, devendo estar concluída até o final de 1999. Está prevista também a possibilidade de implementar uma versão na linguagem Java. Esta tarefa é deveras mais complicada do que a anterior, pois há diferenças conceituais significativas entre Java e Smalltalk, em particular a não existência da noção de metaclasses em Java.

De forma a enriquecer o *Mundo dos Atores*, novas ferramentas que lhe serão acrescentadas já estão com sua implementação iniciada, em particular um modelo de redes neurais e de algoritmo genético. A idéia é introduzir a noção de improviso

(aprendizagem) aos atores, ou seja, permitir que eles sejam capazes de fazer algo mais do que apenas seguir um papel preestabelecido.

Também está sendo realizada a implantação dos conceitos de palco e atores em ambientes de realidade virtual (3D), pela incorporação do *Mundo dos Atores* na ferramenta de autoria de “adventures educacionais” criada por Souza (1997).

Por último, estuda-se a viabilidade de transformar o *Mundo dos Atores* no *Mundo dos Agentes*. Esta perspectiva implica em transformar o modelo de objetos passivos, inerente à perspectiva classicamente implementada pela programação orientada a objetos, em um modelo de objetos ativos. Neste contexto, prevê-se também o oferecimento de uma arquitetura que possibilite a implementação de modelos e processos distribuídos.

Conclusão

Várias experiências de uso do *Mundo dos Atores* já foram realizadas, tanto em disciplinas quanto individualmente, por estudantes de graduação e de pós-graduação em Computação. Estas experiências foram muito importantes, pois permitiram a identificação e a eliminação de falhas conceituais e de programação, ao mesmo tempo em que apontaram direções que conduziram à atual versão.

O trabalho em disciplinas, ou seja, com grupos de estudantes, não nos permitiu avaliar adequadamente a ferramenta, pois ocorreu paralelamente à incorporação de novas características e facilidades ao *Mundo dos Atores*. Também não havia ainda material descritivo que apresentasse a arquitetura do sistema e exemplos de aplicações. Novas experiências estão sendo realizadas para avaliar com maior profundidade seu valor pedagógico. Mesmo assim, o que se nota é que há uma certa facilidade por parte dos estudantes quanto à compreensão dos conceitos que estão por detrás da metáfora do palco. Da mesma forma, é perceptível a facilidade com que os estudantes desenvolveram novos protótipos, muito também em função da realimentação oferecida pelo ambiente gráfico e pela característica lúdica dos problemas propostos.

É importante notar que boa parte dos trabalhos desenvolvidos pelos estudantes dificilmente seria implementada em linguagens sequenciais clássicas. Considerando também que as experiências foram realizadas com estudantes de primeiras fases, muito provavelmente eles não teriam condições de trabalhar em outro ambiente de programação. As características e potencialidades do *Mundo dos Atores* se mostraram significativas e em muito facilitaram o desenvolvimento das tarefas propostas.

Bibliografia

- MEYER, Bertrand. Towards an O-O Curriculum.
<http://www.eiffel.com/doc/manuals/technology/curriculum/>
- TAKAHASHI, Tadao. Introdução à Programação Orientada a Objetos. III EBAI (Escola Brasileiro-Argentina de Informática). Curitiba, Janeiro 1998.
- KERAVNOU, E.T. Introducing computer science undergraduates to principles of programming through a functional language. Lecture Notes in Computer Science 1022. Functional Programming Languages in Education. First International Symposium, FPLE'95. Netherlands, December 1995.
- JACQUOT, J.P. & GUYARD, J. Requirements for an ideal first language. Lecture Notes in Computer Science 1022. Functional Programming Languages in Education. First International Symposium, FPLE'95. Netherlands, December 1995.
- BUDD, Timothy A. Teaching Object-Oriented Programming Across the Internet.
<http://www.cs.orst.edu/~budd/remote.html>
- MINAR, Nelson & et alii. The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations. <http://www.santafe.edu/projects/swarm/>, June, 1996.
- BRIOT, Jean-Pierre. Actalk: a Testbed for Classifying and Designing Actor Languages in the Smalltalk-80 Environment. In Proceedings of European Conference on Object-Oriented Programming (ECOOP'89), British Computer Society Workshop Series, Cambridge University Press, pages 109-129, July 1989.
- LESCAUDRON, Loïc & et alii. Prototyping Programming Environments for Object-Oriented Concurrent Languages: a Smalltalk-Based Experience. In Proceedings of the 5th Conference on the Technology of Object-Oriented Languages and Systems (Tools Usa'91), pages 449-462, Prentice-Hall, 1991.
- MARIANI, A. Carlos. O Mundo dos Atores. <http://www.inf.ufsc.br/poo/atores/>
- SOUZA, Patrícia C.; WAZLAWICK, Raul S. An Authoring System for the Creation of Educational Adventures in Virtual Reality. Revista GRAPH&TEC 1(2):39-53. Editora da UFSC. Julho, 1997. ISSN 1413-6481
- SMALLTALK/V – Tutorial and Programming Handbook. Digital Inc. Los Angeles, 1992.