Taxonomia Tecnologia Java

Resumo

As altas taxas de adoção da Internet e Web pela sociedade tem sido acompanhadas por um vigoroso crescimento da complexidade e abrangência de tecnologia de código móvel, em especial da tecnologia Java, considerada por muitos a pedra fundamental da construção de aplicações na Internet e Web. O crescimento do universo de tecnologias Java se dá através da criação de uma grande quantidade de modelos, ferramentas, bibliotecas e frameworks de desenvolvimento, aplicáveis aos mais diversos domínios da informática e telecomunicações, como hipermídia e multimídia, computação distribuída, gerenciamento de redes, telefonia convencional e móvel, comércio eletrônico, arquitetura de computadores, entre outros. Este artigo propõe uma Taxonomia do Ciberespaço, aplicável à classificação do universo Java, e potencialmente extensível a outros componentes tecnológicos. A relevância da Taxonomia é mostrada através da discussão sobre os eixos e graus que criam um espaço de classificação, e do posicionamento de vários elementos que compõem a tecnologia Java dentro deste espaço, discutindo-se também algumas relações de dependência e evolução entre estes elementos. O objetivo da taxonomia é fornecer um panorama global e sintético da tecnologia Java, potencialmente aplicável em outros contextos do ciberespaço, e que sirva como instrumento de orientação aos que planejam conhecer ou antever os rumos deste universo em expansão.

1 - Introdução

A tecnologia Java[Kramer, 1996] foi lançada na Internet há três anos, e após um breve período de frenesi em torno de seus possíveis impactos no desenvolvimento da computação, esta vem sendo refinada e adotada em um processo gradual e estável, cujos números reais estão descritos em estudos recentes[Byte, 1998]. O crescimento da complexidade e abrangência das diversas facetas da tecnologia Java criaram o que se pode chamar de Universo Java, cujos elementos incluem uma linguagem de programação orientada a objetos, bibliotecas e frameworks de desenvolvimento básicos e avançados, protótipos e especificações de aplicações e modelos computacionais, listas de discussão na Internet, repositórios de documentos, applets e aplicações na Web, livros, etc. O contato não estruturado com esta miríade de elementos dificulta a compreensão da extensão e das possíveis aplicações da tecnologia, principalmente por parte do noviço na área. Este artigo propõe um esquema de classificação, chamado Taxonomia do Ciberespaço, capaz de categorizar estes diversos elementos e que sirva como guia para os que desejam utilizar ou antever os rumos da tecnologia.

O layout deste artigo é como se segue. Na seção 2 é discutida a origem, estrutura e relevância no uso de taxonomias naturais e artificiais, e são propostos e descritos três eixos taxonômicos e seus graus, capazes de enquadrar os diversos elementos do universo Java e do ciberespaço. Na Seção 3 alguns elementos da tecnologia Java são descritos sucintamente e mapeados no espaço taxonômico proposto. Na Seção 4 são apresentadas algumas discussões que demonstram a relevância da taxonomia. Na Seção 5 são apresentadas algumas conclusões, e ao final do artigo estão as referências bibliográficas.

2 - Taxonomias

Taxonomias Biológicas - Taxonomias (esquemas de identificação, nomenclatura, classificação de seres vivos[Barsa, 1997]) tem sido historicamente afins às ciências biológicas[Pereira e Agarez, 1980], [Carrera, 1980]. Aristóteles foi o primeiro a classificar a diversidade de seres vivos que existem na natureza. A taxonomia criada por Lineu, em 1758, e refinada posteriormente por Lamarck e Darwin, mantém ainda hoje sua estrutura geral, tendo sofrido alguma reorganização em função da filogenética (genética aplicada à teoria da evolução). Taxonomias biológicas constituem uma hierarquia de grupamentos, taxon ou taxa(no plural), entre os quais são estabelecidas relações de ordem, como Reino > Filo > Classe > Ordem > Família > Gênero > Espécie. As mais de 1,25 milhões de espécies animais podem ser enquadradas segundo o esquema mostrado na Figura 1.

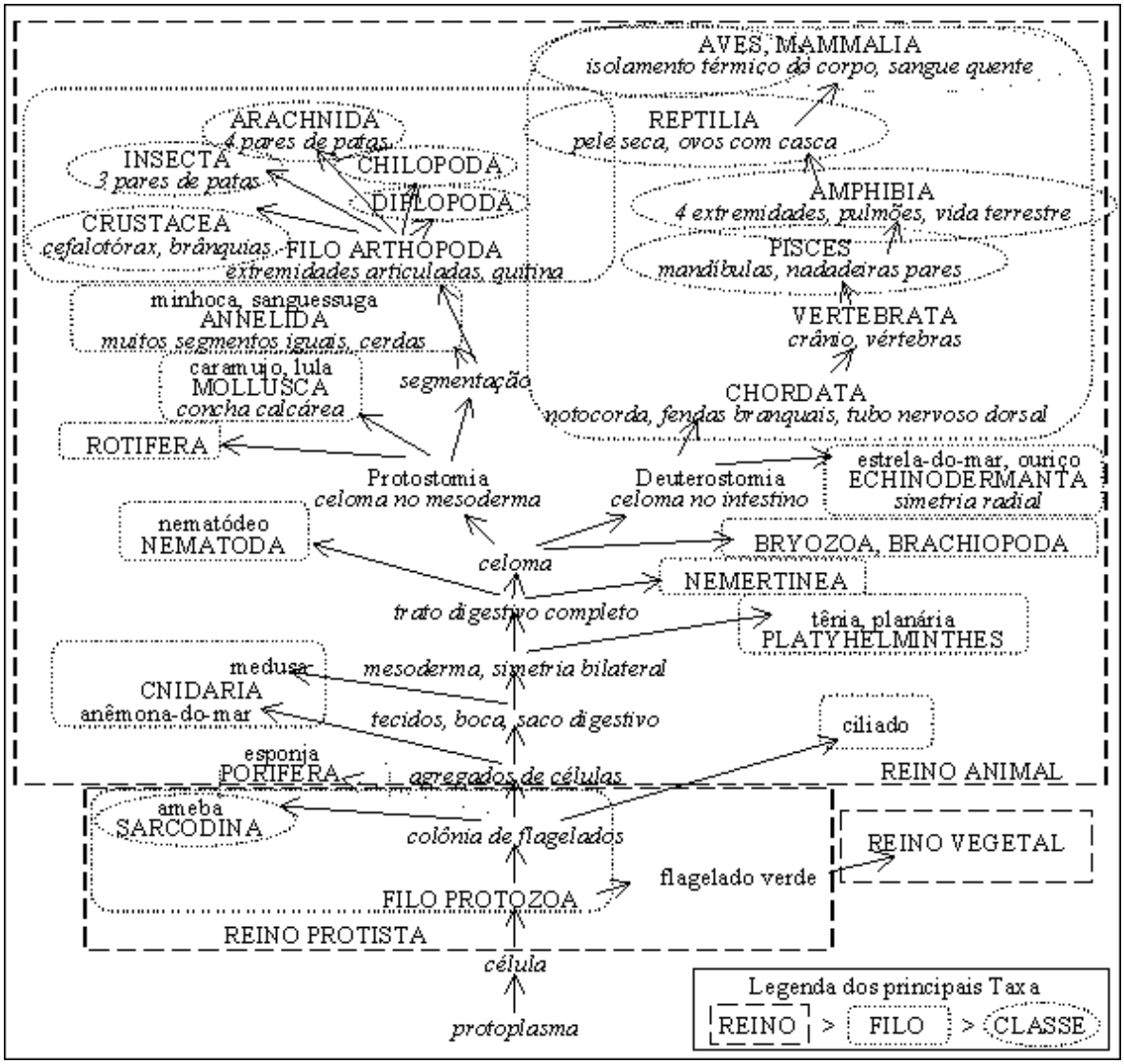

Figura 1 – Esquema Taxonômico do Reino Animal (adaptado de [Storer et alli, 1984], p. 269).

Taxonomias Tecnológicas - Relações de ordem entre os taxa não se adequam plenamente à classificação de componentes tecnológicos, pois a diversidade de seres vivos reflete as "limitações" da natureza no que se refere à criação de complexidade no seres vivos, cuja gênese é resultado dos processos de mutação e reprodução sexuada entre organismos da mesma espécie, guiados por pressões

ambientais e pelo isolamento geográfico[Dobzhansky, 1973]. A gênese de organismos tecnológicos (produtos e processos) também é guiada por pressões ambientais (economia) e por isolamento geográfico (economias fechadas)[Rothschild, 1990], mas sua essência inovadora é fruto do trabalho humano, e deste modo a evolução é direcionada. Enquanto que na natureza não é comum encontramos seres vivos que pertençam simultaneamente a dois taxa disjuntos, como animal e vegetal, peixe e crustáceo, inseto e mamífero, é comum a ocorrência de artefatos tecnológicos que incorporam características de grupos distintos como veículo+casa (trailer), vídeo-cassete+televisão, ou pneu+escova de dentes (uma interessante possibilidade de mercado ou um desastre de engenharia?). O resultado da produção intelectual humana é a criação de universos tecnológicos que seguem linhas evolutivas bastante complexas, e suas taxonomias são consequentemente mais complicadas. Em [Hein et. Alli, 1998] é mostrada uma taxonomia aplicada à descrição de 26 classes de modelos que podem ser construídos utilizando-se a notação VHDL – VHSIC Hardware Description Language, e que se baseia na construção de nove eixos taxonômicos, que expressam resolução temporal, de dados, funcional, estrutural e o nível de programação.

Uma Taxonomia para o Ciberespaço.

Os critérios utilizados para composição dos eixos taxonômicos do ciberespaço foram a expressão de características estruturais, funcionais, temporais e espaciais de componentes tecnológicos, que se traduzem na capacidade de indicar: abstração empregada durante o uso do componente (estrutural-funcional), impacto do componente sobre os "ambientes" concretos onde "vivem"(espacial), e impacto do componente sobre o ciclo de vida de produtos e processos (temporal). São definidos três eixos com os nomes: **Abstração**, **Computação** e **Ciclo de Vida**, que criam o espaço de classificação tridimensional sintetizado na Figura 2.

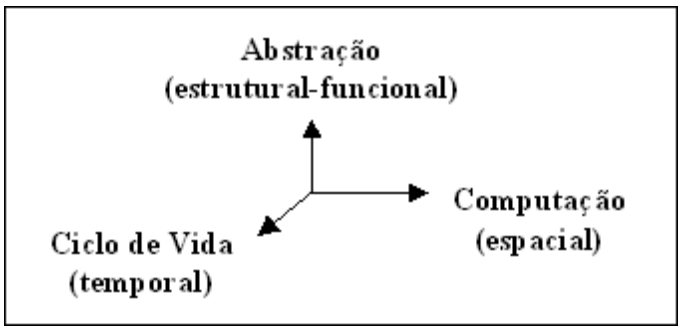

Figura 2 – Espaço gerado pela Taxonomia do Ciberespaço. Cada eixo é descrito a seguir.

Eixo 1 - Nível de Abstração O nível de abstração enfatiza as características estruturais e funcionais de um componente. Os graus de sua escala indicam qual a abstração computacional que é percebida pelo usuário imediato do componente tecnológico, partindo de uma visão estrutural do componente (hardware onde é executado), para uma visão mais funcional (ecologia de componentes tecnológicos). **Hardware** - Elemento computacional digital que tem representação e funcionalidade físicas, como um *chip*, *smart card*, placa de sistema. Necessita de acoplamento a um software para funcionamento pleno. **Sistema Operacional** - Elemento computacional abstrato, gerenciador de recursos de hardware, como memória, dispositivos de armazenamento permanente e periféricos, e normalmente implementado através de software de baixo nível, cuja interface com os níveis acima é uma linguagem de programação de baixo nível e cujo objetivo é representar um sistema computacional de modo abstrato e independente, com vistas à independência do hardware (portabilidade). **Linguagem (+ferramenta e ambiente) de**

Programação - Notação formal para construção de programas, bem como as ferramentas de suporte básico à edição, compilação e depuração. **Biblioteca** - Conjunto de módulos de software reutilizável, com funções bem definidas, utilizados no suporte à construção de programas. Bibliotecas são agrupadas segundo um domínio de programação específico como gráficos, comunicação em rede, manipulação de estruturas de dados e algoritmos. **Framework** - Biblioteca orientada a objetos, formada por um ou mais conjunto de classes e interfaces, cuja organização entre as classes cria uma relação arquitetural entre os componentes, e cujo uso envolve normalmente a especialização de uma ou mais classes que originalmente compõem o framework. Um framework é um esqueleto de aplicação. **Aplicação** - Módulo de software completamente funcional e interface bem definida. Pode ser agregado a outras aplicações na composição de ambientes de software, normalmente distribuídos. **Domínio** - Conjunto difuso e dinâmico de aplicações distribuídas, integradas ou semi-integradas, e que que tem uma funcionalidade não completamente definida. Atendem a um conjunto de usuários na realização de atividades abrangentes como trabalho cooperativo, educação à distância, sistemas de informação corporativos, pesquisa, etc. **Ecologia** - Sistema "econômico" [Lindgren and Nordah, 1995] [Rothschild, 1990] de recursos tecnológicos como aplicações, frameworks, modelos computacionais, especificações, etc, materializadas através de organizações, empresas e indivíduos que estabelecem relações de troca, e cujos recursos (tecnológicos) são criados, evoluem, co-evoluem ou desaparecem na luta pela permanência neste sistema econômico (mercado). A relevância estrutural-funcional do **Nível de Abstração** é sintetizada na Figura 3. Componentes de níveis mais altos na escala apresentam características cada vez mais funcionais, o que os torna independentes dos níveis abaixo. Caso a relevância estrutural esteja presente em um componente (desvio à esquerda) se diz que ele apresenta baixa portabilidade. O nível de ecologia é uma possibilidade explorada nesta proposta taxonômica, que a torna capaz de enquadrar um maior número de artefatos tecnológicos. A possível relevância de fatores físicos (estruturais) na emergência de fenômenos ecológicos[Steels, 1995] não é discutida neste artigo.

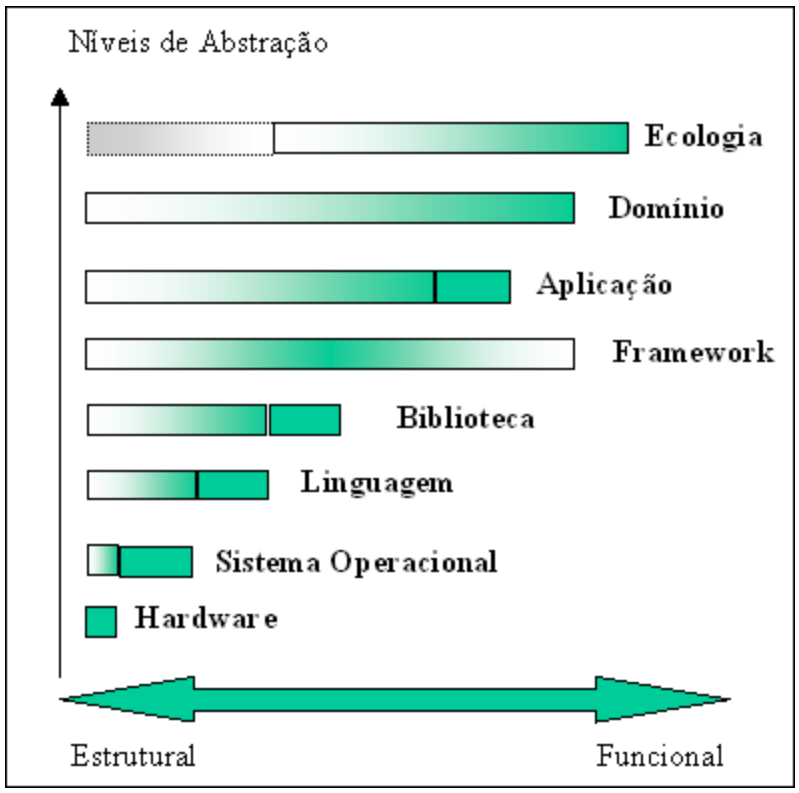

Figura 3 – Relevância Estrutural-Funcional do Eixo Abstração.

Eixo 2 – Ciclo de Vida

O desenvolvimento e uso de componentes tecnológicos (com ênfase em sistemas computacionais distribuídos) passa por diversas etapas, que vão da concepção de teorias e modelos (computacionais) até o projeto, programação e execução (software, hardware, pessoas, procedimentos, etc). Este eixo taxonômico indica qual o impacto do componente tecnológico sobre uma ou mais destas diversas etapas do ciclo de vida dos componentes. Os graus mais baixos da escala tem efeito temporal imediato (Execução), enquanto que os outros possivelmente ocorrem em um tempo remoto e tem uma duração mais longa. As fases de empacotamento e transferência representam a necessidade de suportar mobilidade de código, inerente às tecnologias de construção do ciberespaço, como é o caso de Java. **Execução** – É a etapa imediata do ciclo de vida de um componente, realização da tarefa para a qual ele foi concebido ou da qual ele toma parte. Em outras palavras, é o uso do componente. Um componente tecnológico tem impacto sobre esta etapa se está presente durante a execução da tarefa, por exemplo: *chips*, objetos e processos. **Ligação**- Esta etapa compreende a ligação que é feita entre o código e dados, possivelmente transferidos através da rede, com o contexto de execução local, este último responsável pelo gerenciamento dos recursos e por conduzir a execução do sistema. Carregadores de classes (*ClassLoader*) e verificadores de código tem impacto direto sobre esta etapa. **Transferência** – Ocorre durante a atualização parcial ou total de código e dados que compõem um ambiente de execução, no caso de código móvel, bem como durante a migração de processos, no caso de agentes móveis. Envolve os protocolos de transferência, o processo de certificação, autenticação e criptografia empregado para transferir adequadamente um pedaço de código, um agente ou um objeto computacional entre um duas entidades computacionais independentes, como cliente e servidor. **Empacotamento**.- Envolve atividades como configuração, arquivamento, nomeação e contabilização, necessárias para que um componente seja encontrado e transferidos. Esquemas de nomeação e diretórios tomam parte ativa nesta etapa. **Codificação (programação)** – É vista em amplo espectro e envolve a seleção de algoritmos, estruturas de dados e *patterns* de programação adequados, seleção de elementos de linguagem que atendem às características funcionais e de execução do sistema, escrita de código, compilação, depuração, integração e teste. Compiladores, editores de texto, depuradores, repositórios de código, dados, bibliotecas, frameworks, documentos e especificações tem impacto direto sobre esta etapa. **Projeto** - Implica na seleção e mapeamento dos modelos computacionais e arquitetura do sistema no conjunto de dados, bibliotecas, frameworks, domínios e aplicações específicas que serão usadas na construção do sistema. **Modelo** – Compreende a escolha de uma ou mais estruturas de interligação, conexão ou comunicação entre os grandes módulos distribuídos que compõem o sistema, como modelo computacional, software, hardware, pessoas, procedimentos, etc. **Evolução** – Síntese e/ou seleção de estratégias artificiais e naturais de adaptação e evolução como redes de comunicação, *groupware*, comunidades do ciberespaço[Resnick, 1996], mecanismos de controle[Schmauch, 1994], cooperação e co-evolução[Lindgreen and Nordahl, 1995],[Dobzansky, 1973], empregadas durante o ciclo de vida de componentes. A Figura 4 mostra a relevância temporal dos graus do eixo de Ciclo de Vida. À esquerda do gráfico da Figura 4 estão indicadas algumas propriedades interessantes que surgem quando determinadas etapas se dispersam temporalmente para a esquerda, e passam a ocorrer simultaneamente ao uso (execução) de um componente, ou possivelmente em decorrência deste uso. Quando a evolução, modelagem, projeto ou codificação/programação de componentes é resultado direto ou indireto de seu uso surgem esquemas auto-sustentáveis.

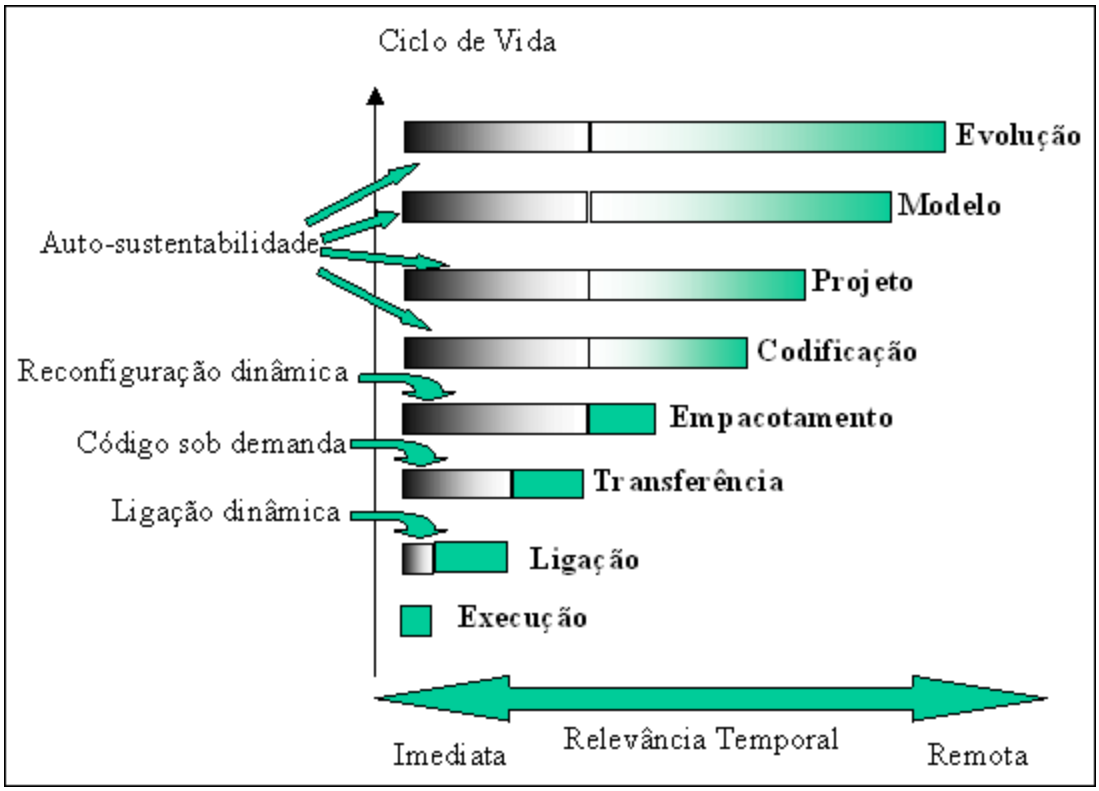

Figura 4 – Relevância Temporal do Eixo Ciclo de Vida.

Eixo 3 - Computação. Os graus deste eixo indicam a relevância do espaço (físico) para o uso do componente. O grau mais primitivo é o uso do componente localizado em um ponto determinado do espaço, como um programa *stand-alone*. Os esquemas cliente-servidor, de código móvel e agentes são baseados no esquema proposto em [Carzaniga et alli, 1997]. O esquema ecológico é baseado em [Huberman, 1986]. **Cliente/Servidor** - Envolve pelo duas entidades computacionais independentes, chamadas abstratamente de *sites* A e B. O *site* A hospeda um cliente que deseja obter um resultado mas não dispõe de recursos como dados, dispositivos físicos ou capacidade computacional, necessários à computação do resultado. O *site* B hospeda um servidor que detém recursos necessários à computação dos resultados ou sabe onde encontrá-los. O cliente recorre então aos serviços do *site* B, no qual é realizada a computação e o envio dos resultados ao cliente. **Código Móvel –** Ocorre sob duas formas complementares, chamadas de **Código sob Demanda** e **Avaliação Remota.**

Código sob Demanda – O cliente hospedado no *site* A dispõe parcialmente de recursos para computar resultados. O servidor localizado no *site* B contém o restante dos recursos necessários à computação (código, e possivelmente dados) e os envia ao cliente localizado no *site* A. O cliente utiliza os recursos provenientes dos dois *sites*, A e B, para computar resultados. **Avaliação Remota** - O cliente no *site* A tem o código necessário à realização da computação, mas não dispõe de dados ou outros recursos necessários à execução. O cliente então envia o código ao servidor localizado no *site* B, que computa resultados utilizando parcialmente o código obtido através do cliente, e por fim envia os resultados ao cliente. A principal diferença entre este esquema e o esquema cliente/servidor é a capacidade do servidor de oferecer uma ampla gama de serviços que são programados através de uma linguagem computacionalmente completa. **Agentes Móveis** - O cliente hospedado no *site* A contém parte dos recursos necessários à computação. O *site* B oferece a possibilidade de hospedar temporariamente o cliente, que migra para o *site* B todo o seu estado de execução e outros recursos como dados e código. Após a migração o cliente se hospeda efetivamente no *site* B, perdendo o vínculo com o site original. Os resultados da computação passam a integrar o conjunto de recursos do cliente, que pode migrar de volta

ao *site* A, ou migrar para outro *site* após a computação dos resultados. **Ecologia[Huberman, 1986]** – Tendem a desaparecer os limites entre clientes e servidores, e a localização espacial tende a tornar-se irrelevante. A computação é em grande parte assíncrona, ocorre em ambientes com conhecimento imperfeito e dados inconsistentes. São empregadas estratégias evolucionárias e ecológicas para alocação de recursos, migração de processos e agentes. A alocação de processos e dados a processadores tende a ser uma propriedade que emerge a partir da dinâmica do sistema e se tornar a imprevisível.

A Figura 5 mostra a relevância espacial do eixo de Computação, cujos graus apresentam relação de ordem. Ecologias dependem de Agentes, que dependem de Código Móvel, que depende de Clientes/Servidores, que ocorrem, em última instância, em algum local do espaço.

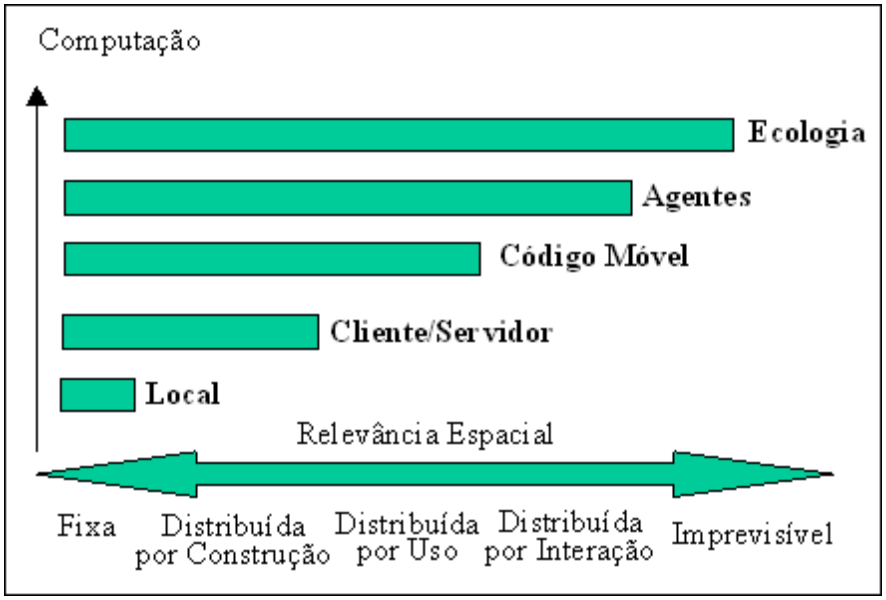

Figura 5 – Relevância Espacial do Eixo Computação.

A próxima seção trata de mostrar os espaços taxonômicos ocupados por alguns elementos que compõem o Universo Java.

3 - Elementos do Universo Java

A classificação de 57 elementos que compõem parte significativa do Universo Java está descrita em um relatório técnico disponível através da Internet[Taxonomia, 1998]. Alguns elementos representativos deste universo são descritos abaixo, agrupados pela abstração à qual mais se adequam. Após a breve descrição de cada elemento são indicadas suas projeções sobre os eixos Abstração, Ciclo de Vida e Computação. São indicados ainda outros componentes dos quais estes dependem ou agregam, bem como outros componentes que são seus possíveis ancestrais. A Figura 6 mostra a distribuição de 9 elementos no esquema Taxonômico do Ciberespaço.

picoJava^{TM} I microprocessor core [Sun, 1998a]- Arquitetura de núcleo de processador projetada especificamente para executar bytecode Java como definido pela **Java Virtual Machine**. O núcleo trata de aspectos peculiares à linguagem Java como sincronização de *threads*, variadas formas de coleta de lixo, invocação de métodos e carga de variáveis locais. A arquitetura baseia-se em um *pipeline* de quatro estágios, com unidades de execução inteira e de ponto flutuante. Contém ainda cache de instruções e dados e uma interface com o barramento do *chip*. As várias implementações do núcleo podem ser

otimizadas com relação a potência, tamanho ou velocidade.

Abstração: Hardware, Sistema Operacional

Ciclo de Vida: Execução

Computação: Local

Depende/Agrega: **Java Virtual Machine**

Ancestrais: **RISC Machines**

MicroJava 701[Sun, 1998b] - Microprocessador de alta performance e uso geral, otimizado para execução de aplicações Java baseado no **picoJava I microprocessor core**. Contém controlador integrado de barramento PCI, controlador de memória, *timers*, etc.

Abstração: Hardware

Ciclo de Vida: Execução

Computação: Local

Depende/Agrega: **picoJava I**

Ancestrais: **SPARC**

Agregados:

JavaOS for Business[Sun, 1998c] - Especialização do **JavaOS**[Madany, 1998], desenvolvida em conjunto pela Sun e IBM, com foco na construção de plataformas Java para aplicações empresarais gerenciadas centralmente por servidores. O objetivo da plataforma é reduzir os custos de manutenção e gerenciamento de sistemas em rede.

Abstração: Sistema Operacional

Ciclo de Vida: Projeto, Modelo, Evolução

Computação: Cliente/Servidor, Código Móvel

Ancestrais: **JavaOS**

Java Virtual Machine[Lindholm, 1996] - Parte do ambiente de runtime de Java que interpreta **bytecodes**[Taxonomia, 1998].

Abstração: Sistema Operacional

Ciclo de Vida: Execução, Ligação

Computação: Local

Java Programming Language[Gosling et alli, 1996] - Linguagem de programação de uso geral com sintaxe similar à da linguagem C++. As principais características da linguagem Java são: orientada a objetos, fortemente tipada, suporte a classes, interfaces e herança, threads e sincronização, tratamento de exceções, suporte à interoperabilidade com código dependente de plataforma, sem aritmética de ponteiros e com gerenciamento automático de memória.

Abstração: Linguagem, Biblioteca

Ciclo de Vida: Projeto, Programação, Empacotamento, Transferência, Instalação, Execução.

Computação: Local, Código Móvel

Depende/Agrega: **Java Virtual Machine**

Java Telephony API[Sun, 1998d] - Esforço combinado da Sun Microsystems, Lucent, Nortel, Novell, Intel, IBM, Siemens, Dialogic e Enterprise Computer Telephony Forum, na definição de um conjunto de APIs e arquiteturas de sistema para desenvolvimento de aplicações Java para automação de centrais telefônicas, na realização de tarefas como atendimento e solicitação de chamadas telefonicas, para programação de secretárias eletrônicas, controle de central, etc.

Abstração: Framework, Domínio, Ecologia

Ciclo de Vida: Programação, Projeto, Modelo

Computação: Cliente/Servidor, Código Móvel

Depende/Agrega: **Applet**, **Java Development Kit[Sun, 1998e]**

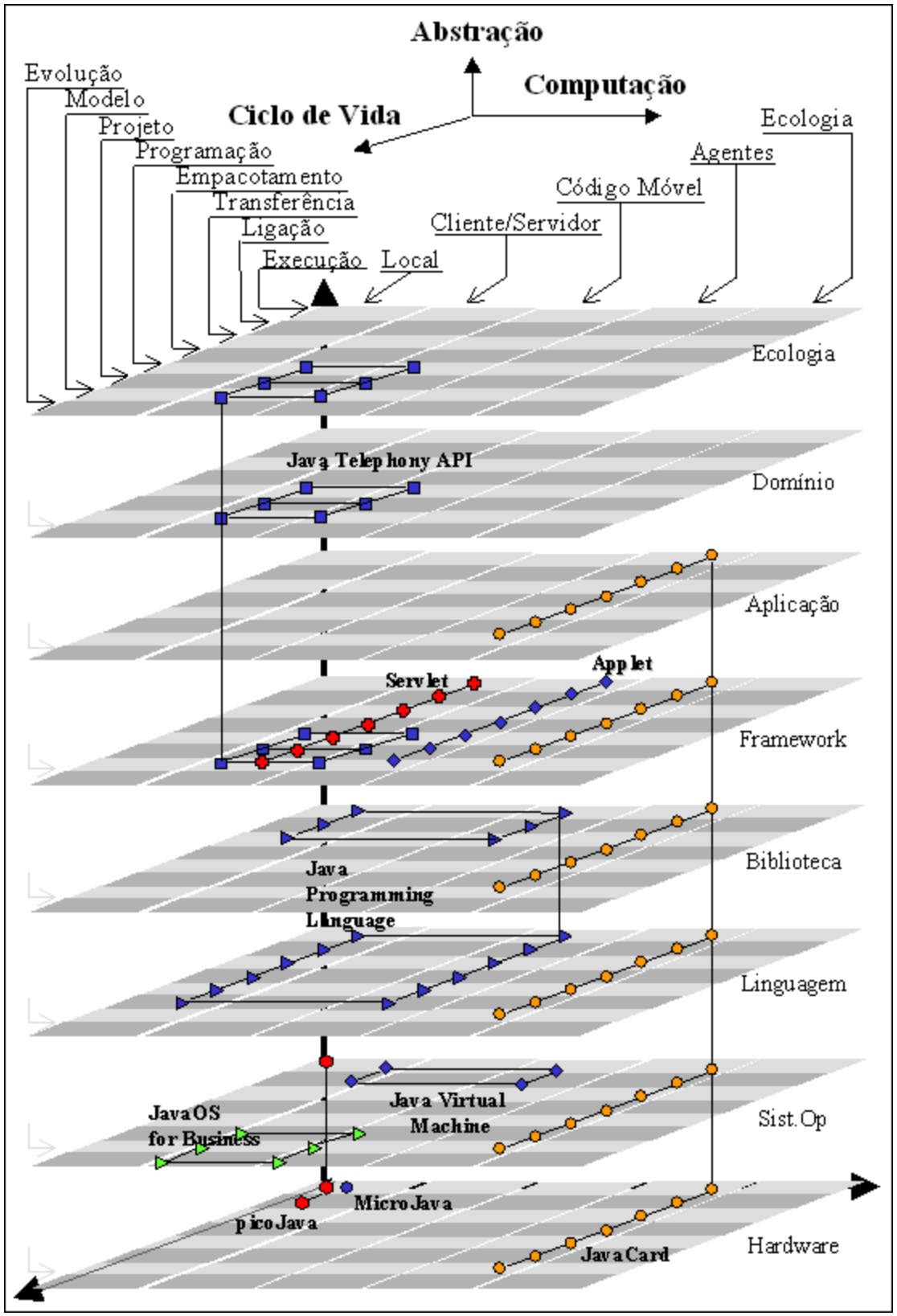

Figura 6 – Alguns elementos do Universo Java na Taxonomia do Ciberespaço.

Java Applet[Campione and Walrath, 1998] – Framework para desenvolvimento de pequenas aplicações trasmitidas através da Web. Suporta o desenvolvimento de aplicações multimídia na Web (animação de

páginas , menus, mapas e gráficos atualizados em tempo real, calculadoras, simuladores, instrumentos de educação a distância, etc). O servidor Web armazena um programa Java (applet) que normalmente apresenta uma interface gráfica multimídia e é transmitido e executado dentro de páginas da Web, à medida em que estas páginas nas quais se hospedam são visitadas por usuários Web.

Abstração: Framework

Ciclo de Vida: Modelo, Projeto, Programação, Empacotamento, Transferência, Instalação, Execução

Computação: Código Móvel

Depende/Agrega: **Web**, **Java Development Kit**

Java Card[Zhiqun, 1998] - Plataforma para construção de (micro) applets que executam no interior de *smart cards*. Composto pela Java Card 2.0 API[Sun, 1997], pelo Java Card Workstation Development Environment[Sun, 1998f] (JCWDE), Java Card Checker, para verificação de conformidade de applets à Java Card 2.0 API. Smart cards[Sun, 1998g] são pequenas unidades computacionais de natureza descartável, que armazenam informações pessoais, como preferências de alimentação, números de contas bancárias, créditos bancários e virtuais, etc. O applet carregado no *smart card* Java altera os dados contidos na pequena memória do cartão, o que corresponde à execução de uma micro-transação comercial. Usa uma limitada biblioteca de applet e tipos primitivos da linguagem Java. A transferência de código de dá através da interface de pinos que há no cartão, usando o protocolo APDU. A **Java Virtual Machine** do smart card só executa enquanto o cartão estiver conectado a uma unidade leitora.

Abstração: Hardware, Sistema Operacional, Linguagem Biblioteca, Framework, Aplicação

Ciclo de Vida: Execução, Ligação, Transferência, Empacotamento, Programação, Projeto, Modelo.

Computação: Agentes

Ancestrais: **Java Virtual Machine**, **Smart Card**

Java Servlet[Sun, 1998h] - Framework para construção de programas Java que executam no interior de servidores Web. Um aprimoramento da tecnologia CGI (Common Gateway Interface).

Abstração: Framework

Ciclo de Vida: Execução, Ligação, Transferência, Empacotamento, Programação, Projeto, Modelo

Computação: Cliente/Servidor

Depende/Agrega: **Web Server**, **Java Virtual Machine**

Ancestrais: **CGI**

4 – Discussão

A Taxonomia do Ciberespaço surgiu a partir de várias perspectivas: 1 - a necessidade de um esquema de classificação capaz de enquadrar o universo de tecnologias que se criou em torno da Internet, especialmente em torno da tecnologia Java, buscando entender quais os princípios aplicados à sua estrutura e evolução, 2 – o estudo de teorias e modelos naturais e artificiais que são capazes de gerar complexidade, evolutibilidade e auto-sustenibilidade [Kaneko e Tsuda, 1994], [Lindgren and Nordahl, 1995], [Kelly, 1994], [Rothschild, 1994], [Fogel, 1997] como comunicação[Wegner, 1995], [Stryer, 1995], distribuição[Berners-Lee, 1998], [Orfali et alli, 1996], controle[Schmauch, 1994] e *out-of-controlness*[Kelly, 1994], bem como 3 – o estudo dos princípios que guiam a construção de taxonomias naturais [Storer, 1984] e artificiais[Hein et alli, 1998], instrumentos fundamentais para compreensão de universos complexos.

Começamos recentemente a explorar o espaço gerado pela Taxonomia do Ciberespaço, e resultados de nossos primeiros ensaios indicam que, em contraste com organismos vivos naturais, componentes tecnológicos não ocupam porções individuais e/ou contínuas do espaço taxonômico. O elemento JavaCard, por exemplo, se estende sobre seis níveis de abstração e tem impacto sobre várias etapas do ciclo de vida, enquanto que o framework Java Applet se restringe a um único nível de abstração. Estamos particularmente interessados em explorar possíveis ligações de causa e efeito entre a dispersão dos componentes no espaço, suas relações evolutivas e de dependência com outros componentes e o sucesso atual e futuro obtido por este elemento junto aos usuários da tecnologia. É possível que existam caminhos previsíveis ou possíveis para obter sucesso tecnológico, resultado de uma composição bem estruturada entre elementos de vários níveis de abstração, ou que tenham impacto sobre etapas diferentes do Ciclo de Vida ou do eixo de Computação.

Achamos também que este mesmo esquema taxonômico pode ser empregado na classificação de outros modelos, tecnologias e aplicações criadas na Internet, e em termos mais amplos na descrição de esquemas interativos naturais e artificiais, como indica a Figura 6.

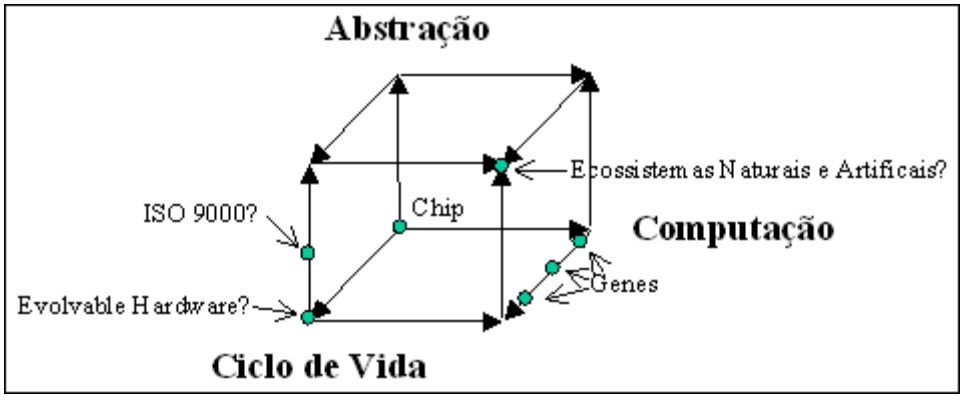

Figura 6 – Possível classificação de componentes diversos aplicados à Taxonomia do Ciberespaço.

5 – Conclusões

Este artigo mostra alguns princípios aplicados à construção de um esquema taxonômico, empregado na descrição e classificação de "organismos" do ciberespaço, em particular

relacionados dentro do universo de tecnologias Java. É mostrada a relevância de esquemas taxonômicos na compreensão de universos complexos como é o caso dos seres vivos e no caso de nove elementos que compõem a tecnologia Java.

Nosso trabalho de classificação está em suas etapas iniciais, e como qualquer esquema taxonômico, reflete o ponto de vista de seus autores, e consequentemente o universo de ciência e tecnologia com o qual tem contato. Sendo assim, a ordem e relevância dos graus contidos em cada eixo pode não ser a mais adequada em todas as situações, bem como é possível que o uso de outros eixos seja bastante relevante em outros contextos, como um que represente Domínios de Aplicação.

Referências

[Barsa, 1997] Taxionomia, Em: *Enciclopédia Barsa*, Vol X. Encyclopaedia Britannica do Brasil Publicações Ltda, 1997.

[Berners-Lee, 1998] Berners-Lee, T. *The World Wide Web Consortium*. W3 Consortium. MIT/LCS USA. 1998. Available from <http://www.w3c.org>

[Campione and Walrath, 1998] Campione, Mary and Walrath, Kathy. *The Java™ Tutorial Second Edition: Object-Oriented Programming for the Internet*. Addison-Wesley. March, 1998. Available from <http://java.sun.com/docs/books/tutorial>.

[Carrera, 1980] Carrera, Messias. *Entomologia para Você, 5 ed*. Nobel, São Paulo, 1980.

[Carzaniga et alli, 1997] Carzaniga, Antonio and Picco, Gian Pietro and Vigna, Giovanni. Designing Distributed Applications with Mobile Code Paradigms. In *Proceedings of the International Conference on Software Engineering (ICSE'97)*. ACM Press. 1997.

[Dobzansky, 1973] Dobzansky, T. *Genética do Processo Evolutivo,* trad. Celso Mourão, Editora Polígono, São Paulo, 1973.

[Fogel, 1997] Fogel, David. Evolutionary computation: a new transactions. IEEE Transactions on Evolutionary Computation. 1(1):1-2, Apr 1997. Available from <http://engine.ieee.org/society/nmc/pubs/tec/ec_toc.html>.

[Gosling et alli, 1996] Gosling, James and Joy, Bill and Steele, Guy, *The Java Language Specification*. Addison-Wesley. September, 1996. Available from <http://java.sun.com/docs/books/jsl>.

[Hein, 1998] Hein, Carl et alli. VHDL Modeling Terminology and Taxonomy, Revision 2.4. RASSP Taxonomy Working Group (RTWG). USA, 1998. Available from <http://rassp.scra.org>.

[Huberman 1988] Huberman, B. A. (Ed.). *The Ecology of Computation*. Elsevier, Amsterdan, 1988.

[Kaneko and Tsuda, 1994] Kaneko, Kunihiko and Tsuda, Ichiro. Constructive complexity and artificial reality: An Introduction. *Physica D – Nonlinear Phenomena*, 75:1-10, August 1994.

[Kelly, 1994] Kelly, Kevin. *Out of Control: The New Biology of Machines, Social Systems and the Economic World*. Addison-Wesley Publishing Company, Reading, MA, 1994.

[Kramer, 1996] Kramer, Douglas. The Java™ Platform: A White Paper. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. May, 1996. Available from <http://java.sun.com>.

[Langton, 1995] Langton, Christoper, editor. *Artificial Life: An Overview*. MIT Press, 1995.

[Lindgren and Nordahl, 1995] Lindgren, Kristian and Nordahl, Mats. Cooperation and community structure in artificial ecosystems. In Lagton, C, editor, *Artificial Life: An Overview*, pages 15-38. MIT Press, 1995.

[Lindholm, 1996] Lindholm, Tim and Yellin, Frank. The Java™ Virtual Machine Specification. Addison-Wesley. September, 1996. Available from <http://java.sun.com/docs/books/vmspec>.

[Madany, 1997] Madany, Peter. JavaOS™: A Standalone Java™ Environment. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. October, 1997. Available from <http://java.sun.com/products/javaos>.

[Meyer et alli, 1993] Meyer, Jean-Arcady et alli, editors. *From Animals to Animats 2 – Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, MIT Press, 1993.

[Orfali et alli, 1996] Orfali, Robert and Harkley, Dan and Edwards, Jery. *The Essential Distributed Objects Survival Guide*, John Wiley, New York, 1996.

[Pereira e Agarez, 1980] Pereira, Cezio e Agarez, Fernando. *Botânica: Taxonomia e Organografia dos Angiospermae – Chaves para Identificação de Famílias*. Editora Interamericana, Rio de Janeiro, 1980.

[Resnick, 1996] Resnick, Mitchel. Distributed construcionism. In *Proceedings of the International Conference on Learning Sciences*. Association for the Advancement of Computing in Education, 1996. Available from <http://www.media.mit.edu/mres>.

[Rothschild, 1990] Rothschild, Michael. *Bionomics: Economy as Ecosystem*, Henry Holt and Company, Inc. New York, 1990.

[Schmauch, 1994] Schmauch, Charles. *ISO 9000 for Software Developers*. ASQC Quality Press, Milwaulkee, 1994.

[Steels, 1995] Steels, Luc. The artificial life roots of artificial intelligence. In Langton, C, editor, *Artificial Life: An Overview*, pages 75-110. MIT Press, 1995.

[Storer et alli, 1984] Storer, T. et alli. *Zoologia Geral*, 6 ed., trad. Froelich, C, et alli. Companhia Editora Nacional, São Paulo, 1984.

[Stryer, 1995] Stryer, Lubert. Biochemistry. 5th Ed., W. H. Freeman and Company, New York, 1995.

[Sun, 1997] Sun Microsystems, Inc. Java Card 2.0 API. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. 1997. Available from <http://java.sun.com/products/javacard>.

[Sun, 1998ª] Sun Microsystems, Inc. Whitepaper: picojava™ I Microprocessor Core Architecture (WPR-0014-01). Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, CA 94303 USA. 1998.

Available from <http://www.sun.com/microelectronics/java>.

[Sun, 1998[b]] Sun Microsystems, Inc. microJava-701 Processor (SME1701BGA). Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, CA 94303 USA. 1998. Available from <http://www.sun.com/microelectronics/microJava-701>.

[Sun, 1998[c]] Sun Microsystems, Inc. JavaOS For Business. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. March, 1998. Available from <http://java.sun.com/products/javaos>.

[Sun, 1998[d]] Sun Microsystems, Inc. The Java Telephony API: an overview. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. October, 1997. Available from <http://java.sun.com/products/jtapi>.

[Sun, 1998[e]] Sun Microsystems, Inc. Java™ Development Kit Version 1.2 New Feature Summary. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. 1998. Available from <http://java.sun.com/products/jdk>.

[Sun, 1998[f]] Sun Microsystems, Inc. Java Card Reference Implementation User's Guide, Developers Release 2.0. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. February 20, 1998. Available from <http://java.sun.com/products/javacard>.

[Sun, 1998[g]] Sun Microsystems, Inc. What is a Smart Card?. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. March 02, 1998. Available from <http://java.sun.com/products/javacard>.

[Sun, 1998[h]] Sun Microsystems, Inc. Java Servlet API. Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, CA 94043. 1997. Available from <http://java.sun.com/products/jdk/1.2/docs>.

[Taxonomia, 1998] Taxonomia de Java. Relatório Técnico, 1998. Disponível em <http://>.

[Wegner, Peter] Wegner, Peter. Models and paradigms of interaction, *OOPSLA'95 Tutorial Notes*, October 1995.