Um Modelo de Negociação Automatizada para Comércio Eletrônico

Jó Ueyama Edmundo R. M. Madeira
Instituto de Computação - Universidade Estadual de Campinas
Caixa Postal 6176, 13083-970, Campinas - SP - Brasil
e-mail: {joueyama, edmundo}@ic.unicamp.br

Resumo

Existem poucos modelos de negociação bilateral implementados para aplicações de comércio eletrônico que provêem a negociação entre o comprador e o vendedor. Este artigo propõe um protocolo de negociação entre dois participantes e um mecanismo para medir as similaridades entre dois produtos, utilizado para obter o produto mais similar ao que foi requisitado pelo consumidor quando um exato não for encontrado. O protocolo proposto segue o modelo bilateral do *OMG*. A negociação é tratada pelos agentes móveis (*Grasshopper*) de compra e de venda, representando respectivamente o comprador e o vendedor. Por sua vez, a negociação do preço é baseada no modelo *Kasbah*. Os catálogos no modelo foram implementados em *XML* para prover a interoperabilidade entre diferentes arquiteturas. Um protótipo foi desenvolvido para validar o modelo proposto.
Palavras-chave: E-Commerce, Negociação Automatizada, Agentes Móveis.

Abstract

Electronic commerce applications are lacking a bilateral negotiation model which provides the bargaining between two participants (supplier and consumer) in order to buy and sell goods. This paper proposes a negotiation protocol between two participants, as well as the *similarity measures* which were implemented to find a similar product, when a specific one could not be found. The proposed protocol follows the bilateral model approved by the *OMG*. The negotiation model is composed of selling and buying *Grasshopper* mobile agents which negotiate between them in order to get the best deal. The negotiation of the price is based on the *Kasbah* model. The catalogs in the model are implemented in *XML* to provide the interoperability among different systems. A simple prototype has been implemented to verify the viability of this concept.
Keywords: E-Commerce, Automated Negotiation, Mobile Agents

1 Introdução

O comércio eletrônico tem se mostrado através do crescimento comprovado pelas estatísticas, como a nova e promissora forma de comércio, dentro do contexto da economia global. Porém apesar destas estatísticas otimistas, as aplicações de comércio eletrônico necessitam ainda de

muitas melhorias, para atender de forma adequada o usuário que utiliza desse mecanismo de comércio. A grande dificuldade para os usuários da Internet, como também do comércio eletrônico, é a falta de uma estrutura padronizada que permita procurar e encontrar as informações de forma simples e otimizada.

A falta de uma padronização nos sistemas de comércio eletrônico dificulta a implementação da interoperabilidade nos mesmos. Tendo em vista que esses sistemas utilizam a Internet para operacionalizar as suas funcionalidades, fica claro que a interoperabilidade nos sistemas de comércio eletrônico é uma questão vital para que o e-commerce possa sair do seu estado embrionário, onde se encontra atualmente. Outra questão em aberto é a implementação de mecanismos eficazes de negociação nesses sistemas. Como o comércio eletrônico envolve a troca de bens, é natural que haja uma negociação entre os participantes, para que se chegue a um acordo entre as partes envolvidas. Apesar da presença maciça de leilões na Internet, mecanismos eficazes de negociação entre consumidor e vendedor ainda não são largamente utilizados.

O propósito deste trabalho é apresentar um modelo de negociação automatizada entre consumidor e vendedor. Os catálogos nesse modelo são projetados em XML [20] com o intuito de prover a interoperabilidade entre arquiteturas diferentes. Exemplos de catálogos em XML para comércio eletrônico podem ser encontrados em [11, 19]. Um protocolo de negociação bilateral (dois participantes) é proposto para que consumidor e fornecedor possam negociar entre eles. Este modelo foi implementado em *Java* utilizando o sistema de agentes móveis *Grasshopper*.

As funcionalidades básicas de negociação utilizadas neste trabalho são baseadas na Facilidade de Negociação do OMG (*Object Management Group*). O nosso protocolo utiliza também algumas funcionalidades do modelo de *Kasbah* [2] e do *Contract Net Protocol* [18]. Assim como no *Kasbah*, o nosso protocolo é baseado nos agentes móveis de venda e de compra que representam o comportamento do vendedor e do comprador nesses agentes. Da mesma forma, o nosso protocolo implementa a chamada de propostas existente no protocolo *Contract Net*.

Este artigo é organizado da seguinte forma. A seção 2 introduz alguns conceitos de comércio eletrônico e aborda a facilidade de negociação especificada pelo OMG. O protocolo de negociação e o mecanismo utilizado para definir as similaridades entre os produtos são apresentados na Seção 3. A Seção 4 será destinada para discutir aspectos relacionados com a implementação. Os trabalhos relacionados são descritos na Seção 5. A Seção 6 conclui o artigo.

2 Conceitos

2.1 Negociação em Comércio Eletrônico

O comércio eletrônico é uma nova forma de comércio, onde o produto é conhecido, demonstrado e vendido por meios eletrônicos. Atualmente o meio mais popular é a Internet. O comércio eletrônico ou e-commerce pode ser utilizado para serviços econômicos, propaganda, catálogos, compra e pagamento de contas. A localização geográfica para que esses serviços sejam operacionalizados é irrelevante, posto que a Internet tem presença mundial, de forma a contribuir sensivelmente para a globalização do comércio mundial.

As atividades de comércio na Internet têm-se limitado apenas na venda baseada em catálogos. Porém este cenário irá rapidamente se alterar no decorrer do tempo com a inclusão dos mecanismos de negociação para determinar os preços dos produtos e outros atributos que sejam negociáveis [10]. A negociação no comércio eletrônico pode resultar em grandes volumes

de negócio entre clientes em potencial em um curto período de tempo com um custo inferior aos outros mecanismos de negociação convencional utilizados hoje em dia. Além disso, a negociação no comércio eletrônico provê uma maior flexibilidade aos consumidores e fornecedores, em virtude dos atributos não serem fixos.

Na negociação automatizada as ofertas e as contra-ofertas não são fornecidas pelos respectivos compradores e fornecedores, porém as mesmas são propostas pelo sistema em que o modelo está baseado. O mecanismo automatizado poderá prover um processo mais discreto aos participantes, uma vez que não existe o contato direto com a outra parte envolvida.

2.2 Negociação Bilateral do OMG

O modelo de negociação bilateral do OMG [15] diz respeito à interação entre dois participantes, um consumidor e um vendedor, que barganham até que um deles decida aceitar ou rejeitar por definitivo a oferta proposta. A sessão da negociação neste modelo poderá também ser finalizada através da transição *timeout* que é chamada quando o tempo de inatividade ultrapassar o tempo máximo preestabelecido.

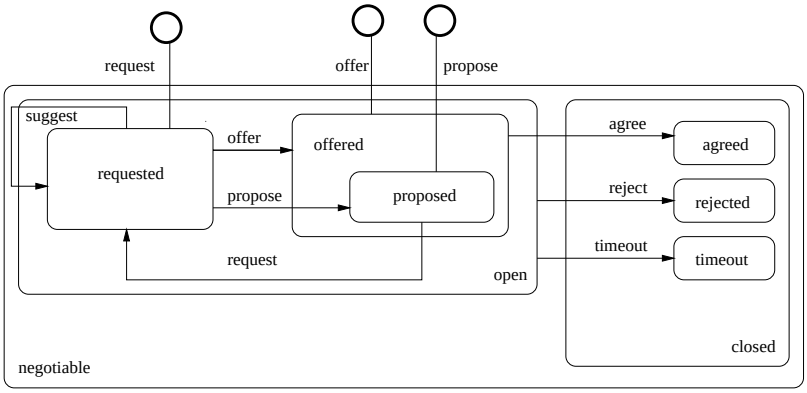


Figura 1: Transição de estados do modelo Bilateral

Conforme pode ser visualizado no diagrama de estados da Figura 1, o referido modelo possui três subestados (*requested, proposed* e *offered*) dentro do superestado *open*, que posteriormente levam a um dos subsestados de *closed* denominados de *agreed*, *rejected* e *timeout*. A negociação pode ser inicializada através das transições *request, offer* e *propose* que levam aos respectivos estados conforme indicados na Figura 1.

Os subestados de *open* representam diferentes níveis de compromissos estabelecidos entre os participantes da negociação. Um compromisso maior é definido em *offered*, onde uma oferta pode ser aceita ou rejeitada definitivamente. *Proposed* estende semanticamente o estado *offered* permitindo que a oferta possa ser aceita ou que a mesma seja modificada através da transição *request*. O estado *requested* não requer nenhum compromisso do participante que invocou a transição. A transição *suggest* é utilizada para chamar por novas ofertas de forma que se possa migrar para um estado com um maior compromisso.

O superestado *closed* possui três subestados: *agreed, rejected* e *timeout*. *Agreed* indica que ambos fecharam o acordo quanto à oferta proposta, enquanto que *rejected* implica falha na negociação em virtude da oferta ser rejeitada por uma das partes. *Timeout* é um estado que

encerra a negociação após um determinado tempo de inatividade preestabelecido. As transições *agree, reject* e *timeout* levam aos respectivos estados, conforme indicados na Figura 1.

3 Modelo de Negociação

3.1 Visão Geral

Esta Seção apresenta o modelo de negociação proposto, detalhando as funcionalidades, assim como a sua estrutura. A nossa abordagem de similaridades para determinar as semelhanças entre os produtos ofertados também é discutida nesta Seção.

No nosso modelo, o comprador e o vendedor são representados através dos agentes móveis para permitir a mobilidade dos mesmos. O agente de compra migra para a máquina do agente de venda, minimizando o uso da rede na comunicação entre eles. Na negociação, a mobilidade é bastante interessante, posto que na grande maioria das vezes, este processo envolve uma exaustiva permuta de ofertas e contra-ofertas.

O protocolo de negociação é composto de cinco fases: *chamada, seleção, negociação, apresentação* e *conclusão*. A negociação tem início quando o agente de compra faz a *chamada* por propostas, enviando os dados do produto requisitado pelo consumidor para o agente de venda. A *seleção* diz respeito à seleção dos agentes de venda que retornaram propostas potencialmente aceitáveis ao agente de compra. A *negociação* compreende a barganha entre o agente de compra e os de venda que foram previamente selecionados. Após esta fase, os resultados gerados através da negociação são *apresentados* ao consumidor que por sua vez rejeita ou aceita a oferta proposta, *concluindo* a sessão de negociação. Vale salientar que a sessão de negociação pode se repetir o número de vezes que o consumidor desejar.

3.2 Diagrama de Transição de Estado do Modelo

O diagrama de estado do nosso modelo é bastante semelhante com o diagrama ilustrado na Figura 1. A principal diferença com modelo original é a inclusão da transição *Call for Proposals* no lugar da transição *request*, presente no modelo original para a inicialização da negociação.

Call for Proposals é uma mensagem multicast do agente de compra para todos os agentes de venda que se encontram disponíveis para negociar em um determinado *marketplace* (local onde os agentes de venda possuem ofertas de produtos). A finalidade do *Call for Proposals* é a chamada por propostas e a mesma não implica compromisso algum por parte do agente chamador, porém esta transição faz com que os agentes de venda retornem com *propose, offer* ou até mesmo com *reject*. O novo diagrama de estado é ilustrado na Figura 2.

3.3 Catálogos em XML

XML (*Extensible Markup Language*) é um padrão proposto pela W3C (*World Wide Web Consortium*) [21] com a finalidade de estruturar as informações que são transmitidas pela Internet. XML teve origem do SGML (*Standard Generalized Markup Language*), portanto XML e HTML possuem a mesma origem [9], sendo ambos baseados na tecnologia de marcadores. A principal diferença entre as duas linguagens é a flexibilidade quanto à criação dos marcadores

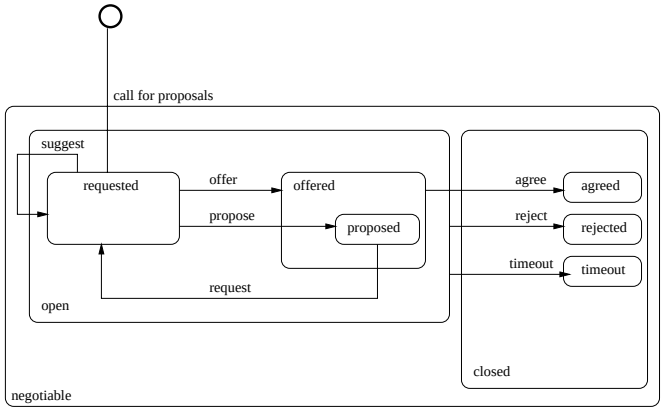


Figura 2: Diagrama de Transição de Estado do Modelo

de acordo com as aplicações do usuário. Diferente do HTML, XML não possui um conjunto fixo de marcadores pré-definidos.

No nosso modelo, os catálogos foram projetados em XML com a finalidade de prover a interoperabilidade entre arquiteturas diferentes. XML da mesma forma que HTML, pode ser manuseado em qualquer plataforma, porém é importante ressaltar que a interoperabilidade nos catálogos só poderá ser estabelecida mediante a padronização dos marcadores. Vários esforços vem sendo realizados com o intuito de prover a interoperabilidade nos catálogos em XML, dentre os diversos é importante citar o projeto *RosettaNet* [4] e o padrão *eCo Framework* [3]. Atualmente, os desenvolvedores de XML possuem o mesmo problema dos desenvolvedores de componentes: padronização do vocabulário utilizado para descrever as informações transmitidas entre aplicações diferentes [16].

Uma outra vantagem é em relação a sua simplicidade. As informações em XML não são armazenadas em formato binário, portanto o seu conteúdo pode ser lido a partir de qualquer editor de texto, sem a necessidade de se converter do seu formato original.

3.4 Medidas de Similaridades

Esta Seção apresenta o mecanismo para determinar o produto mais similar ao produto requisitado pelo consumidor, entre os diversos que foram ofertados. O referido mecanismo é utilizado quando o produto requisitado pelo consumidor não for encontrado no catálogo.

A nossa abordagem é baseada nos pesos que são atribuídos a cada informação. Conforme ilustrado na Seção 4, o formulário de entrada de dados contém vários campos de informação que por sua vez são ligados a um objeto *Slider* onde os pesos (de 1 a 10) são informados.

A medida de similaridade é calculada pela soma de todos os pesos elevados ao quadrado. Porém, o peso só é adicionado nos atributos cujas informações são as mesmas do valor do produto encontrado no catálogo. Se as informações não forem as mesmas, nenhum valor é adicionado. O processo descrito pode ser formalizado através da Equação 1.

$$W_T = \sum_{i=0}^{n} w_i^2 x_i \quad (1)$$

onde W_T é o valor da similaridade final; w_i é o valor peso de cada atributo; x_i corresponde ao

valor 0 ou 1, dependendo se o valor encontrado for igual ao valor requisitado; e n é o número de atributos considerados na pesquisa. Quanto maior o valor de W_T, mais similar é o produto com o que foi requisitado.

Os pesos são elevados ao quadrado com o intuito de enfatizar os maiores valores, levando a similaridade para a característica mais relevante do consumidor. Existem muitas discussões sobre o fato de elevar-se ao cubo ou mesmo para valores mais altos. No entanto, diante das nossas simulações encontramos produtos mais condizentes com o que foi requisitado pelo consumidor ao elevar-se o peso ao quadrado.

O peso com valor 10 é um caso especial: o mesmo é informado com a finalidade de determinar que aquela propriedade não é negociável e deve ser igual ao valor requisitado. Por exemplo, se o consumidor deseja adquirir um carro e informa 10 como peso e o valor do atributo como *Honda Civic*, então o único valor aceitável para este atributo será *Honda Civic*. Um exemplo do uso da medida de similaridade será apresentada a seguir.

3.4.1 Exemplo

Suponha que o consumidor esteja procurando por um carro novo modelo *Honda Civic*, cor *azul*, com *ar condicionado, CD player, alarme* e *air bag*. Os pesos informados para cada atributo estão descritos na Tabela 1.

Atributo	*Valor*	*Peso*
Modelo	Honda Civic	9
Cor	Azul	7
Novo	Sim	6
Usado	Sim	4
Ar Condicionado	Sim	8
CD Player	Sim	5
Alarme	Sim	6
Air Bag	Sim	5

Tabela 1: Atributos e valores informados pelo consumidor

Atributo	*Valor*	w_i^2	
Modelo	Honda Civic	9^2	81
Cor	Amarelo	0	0
Novo	Sim	6^2	36
Usado	Não	0	0
Ar Condicionado	Não	0	0
CD Player	Não	0	0
Alarme	Sim	6^2	36
Air Bag	Não	0	0

Atributo	*Valor*	w_i^2	
Modelo	Nissan Sentra	0	0
Cor	Preto	0	0
Novo	Sim	6^2	36
Usado	Não	0	0
Ar Condicionado	Não	0	0
CD Player	Sim	5^2	25
Alarme	Sim	6^2	36
Air Bag	Sim	5^2	25

Tabela 2: Veículos similares encontrados com (a) W_T = 153, e (b) W_T = 122

As Tabelas 2.a e 2.b apresentam veículos similares ao requisitado com os seus respectivos pesos elevados ao quadrado. A Tabela 2.a descreve um veículo mais similar do que o veículo na Tabela 2.b, de acordo com o maior valor de W_T. A primeira ocorrência possui um veículo mais similar, uma vez que ela representa um veículo contendo um número maior de atributos relevantes para o consumidor. Neste exemplo, os atributos mais relevantes são: modelo (*Honda Civic*) e *ar condicionado*. Ambos são mais importantes para o consumidor, tendo em vista que os mesmo foram informados com um peso maior. Note que nenhum destes valores estão presentes na Tabela 2.b. Portanto, nesse exemplo, a proposta selecionada seria a primeira em função do maior valor de W_T.

É importante observar que a soma dos pesos (w_i) na Tabela 2.a (21) é menor que a soma na Tabela 2.b (22), porém a soma dos quadrados é maior na Tabela 2.a (153) do que na Tabela 2.b (122). Fica claro a partir deste exemplo que elevando-se o peso ao quadrado é possível enfatizar cada característica atribuída pelo consumidor.

3.5 Protocolo de Negociação

O nosso modelo de negociação é composto pelos agentes móveis de compra e de venda. O referido modelo possui duas modalidades de procura: *fixa* e *negociável*. Além disso, o protocolo é dividido em cinco fases (*chamada, seleção, negociação, apresentação* e *conclusão*), cada uma delas contendo funcionalidades diferentes [6].

A procura *fixa* citada anteriormente é utilizada quando o consumidor deseja receber propostas de produtos com os mesmos atributos requisitados pelo consumidor. Portanto, essa pesquisa é mais utilizada quando o comprador está à procura de um produto específico, como é o caso da compra de um determinado livro. Os mecanismos utilizados para determinar as similaridades não são utilizadas, uma vez que o consumidor não está interessado em produtos similares. Caso o mesmo não seja encontrado, uma mensagem é exibida informando o consumidor.

A outra modalidade de pesquisa, *negociável*, é uma extensão da primeira. Nesta modalidade, o consumidor informa o produto a ser adquirido que por sua vez é procurado em vários catálogos do *marketplace*. Porém, se o mesmo não for encontrado, um produto similar será pesquisado utilizando a métrica da similaridade, de forma a satisfazer o consumidor.

Na fase da *conclusão*, o usuário pode iniciar uma outra sessão de negociação, caso o produto ofertado por cada agente de venda não satisfaça o consumidor.

3.5.1 Fase da Chamada

A primeira fase diz respeito ao envio de uma mensagem *multicast* do agente de compra para todos os agentes de venda com a finalidade de invocar por propostas. O agente de compra pode consultar o serviço de *Trader* [13] ou algum outro similar para encontrar agentes de venda apropriados. A presente fase é formalizada através da transição *call for proposals* ilustrada na Figura 2.

Após a chamada por propostas, o agente de venda pode responder com: *propose*, *offer* ou *reject*. *Propose* e *offer* são utilizadas para enviar propostas para o consumidor. Por outro lado, *reject* pode ser utilizada para rejeitar o pedido do cliente em virtude do mesmo estar negociando com um número excessivo de agentes de compra ou até mesmo em função do agente não vislumbrar um bom negócio nessa negociação. Nesta fase, o agente de compra pode

chamar o *Timeout* caso o agente de venda demore em responder à chamada por propostas em virtude do mesmo se encontrar bastante ocupado com outras negociações.

A Figura 3 ilustra o cenário proposto para a primeira fase onde um agente de compra envia uma chamada por propostas para quatro agentes de venda. Nesse exemplo, apenas o *SellAgent1* rejeita o pedido do cliente, negando a sua participação na negociação, enquanto que *SellAgent2*, *SellAgent3* e *SellAgent4* aceitam negociar, respondendo para isso com *offer* ou *propose*. A diferença entre as duas transições, conforme foi explicado anteriormente, é de que o primeiro indica uma proposta final enquanto que a última é uma oferta que é negociável. É válido salientar que se o consumidor opta pela procura *fixa*, a próxima fase a ser executada é a de *apresentação*, onde todas as ofertas são expostas ao consumidor.

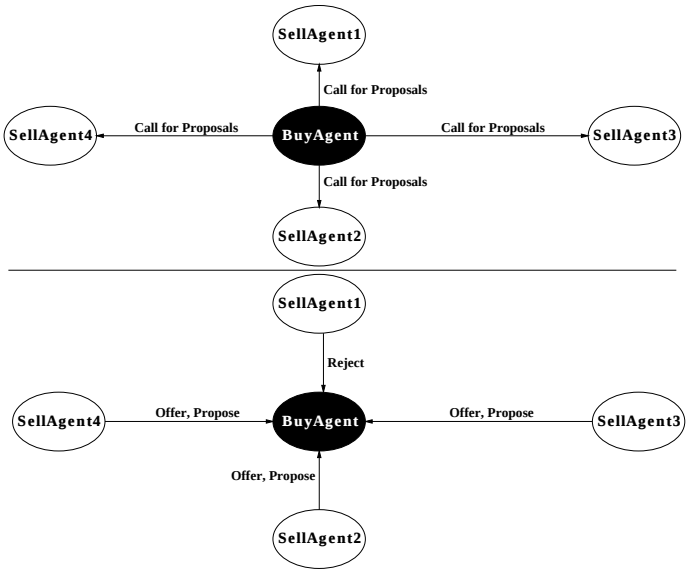


Figura 3: Cenário proposto para a fase da Chamada

3.5.2 Fase da Seleção

Após o recebimento das propostas na fase anterior, os agentes de compra estão agora aptos a selecionar os agentes de venda com quem irão interagir. O critério utilizado para selecionar é a proposta enviada. Se o agente de compra vislumbrar um mau negócio, o mesmo invoca o *reject* de forma que possa negar a negociação com esse agente. A previsão de uma negociação com um ganho desfavorável pode vir de dois fatos:

- preço desejado muito distante do preço ofertado.
- o número de atributos iguais do produto ofertado e do requisitado está aquém do que foi preestabelecido.

A seleção dos agentes de venda é formalizada através da transição *request* que na realidade é uma contraproposta da chamada lançada na primeira fase (*Chamada*). Se o agente de venda responder a chamada através da transição *offer* e caso o agente de compra vislumbre um ganho

favorável nessa proposta, então a mesma será armazenada com a finalidade de ser apresentada ao consumidor juntamente com outras propostas na fase da *Apresentação*.

A seleção dos produtos é realizada através do cálculo da similaridade abordada anteriormente. No nosso modelo, cada agente de venda calcula o W_T para cada produto e manda ao agente de compra como proposta o que prover o maior valor do W_T. O preço do produto é negociado na próxima fase (*negociação*).

Na Figura 4, apenas o *SellAgent3* e o *SellAgent4* foram selecionados para negociar com o agente de compra, posto que o mesmo não vislumbrou um ganho muito bom a partir da negociação com o *SellAgent2*.

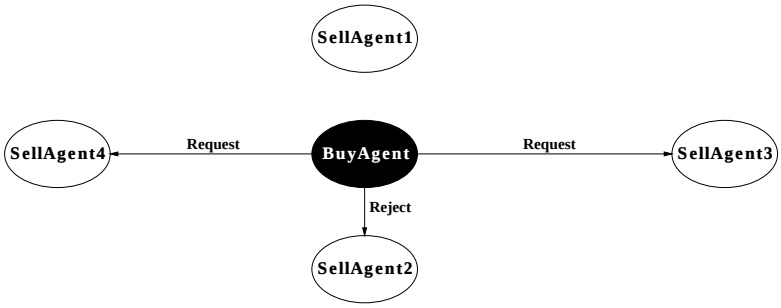


Figura 4: Cenário proposto para a fase da Seleção

3.5.3 Fase da Negociação

Após a seleção dos agentes, o agente de compra está agora apto a negociar com os mesmos de forma a alcançar o melhor preço. Os mecanismos de barganha de preço nesse modelo são baseados no *Kasbah*. Vale salientar que a negociação neste presente trabalho se limita apenas à barganha de preço, porém a nossa pretensão é estender estes mecanismos para outros atributos.

Antes da negociação propriamente dita, o agente de compra é copiado para os *hosts* onde cada agente de venda está localizado. Cada cópia do agente de compra pode negociar de forma autônoma, retornando a melhor oferta ao agente de compra original. Neste protocolo não existe comunicação entre as cópias dos agentes de compra.

A cópia e a mobilidade para as respectivas máquinas são vantajosas principalmente em virtude da negociação envolver uma quantidade de mensagens muito grande, ocasionada pelas propostas e contrapropostas entre os agentes. A cópia pode evitar a comunicação na rede, reduzindo o tráfego na rede.

A fase de negociação não se faz presente caso agente de venda oferte um produto que contenha os mesmos atributos requisitados pelo consumidor, além do seu preço ser inferior ou igual ao desejado pelo comprador.

A Figura 5 ilustra as funcionalidades e as transições presentes durante a fase de *negociação*. Os agentes de venda e de compra ofertam propostas e contrapropostas através das transições *offer* e *propose*. Por outro lado, as transições *request* e *suggest* são utilizadas para invocar por novas propostas.

A sessão da negociação é fechada diante de três eventos:

- o agente de venda oferta uma proposta final através da transição *offer*;

- chamada do *timeout* em virtude da ausência de proposta durante um tempo pré-estabelecido.
- o agente de venda oferta o preço desejado pelo consumidor.

A estratégica do agente de venda para obter um maior ganho será iniciar a negociação com um preço acima do desejado para venda e diminuir até o seu preço mínimo estabelecido pelo vendedor. De forma similar, o agente de compra poderá iniciar ofertando propostas com preços abaixo do desejado pelo consumidor e aumentar o mesmo durante o processo de negociação até que se alcance o preço máximo estabelecido.

Atualmente estamos empenhados para desenvolver um modelo que possa barganhar outros atributos além do próprio preço, que utilize também nesta fase a métrica de similaridade.

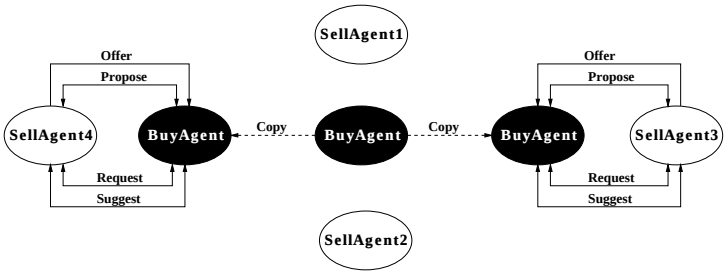


Figura 5: Cenário proposto para a fase de Negociação

3.5.4 Fase da Apresentação

Após a conclusão da negociação de preço executada por cada agente de venda e de compra, cada cópia do agente comprador apresenta os resultados alcançados na sessão de negociação. Em outras palavras, na fase da *apresentação* (Figura 6) cada cópia do agente de compra envia uma proposta resultante da negociação para a sua cópia original. Estas propostas são apresentadas ao consumidor que poderá aceitar ou negar cada proposta. Após a entrega das propostas, cada cópia do agente de compra pode ser removida do *host* para onde foi copiado.

3.5.5 Fase da Conclusão

A fase da *conclusão* diz respeito ao fechamento da sessão de negociação. A negociação é concluída através da formalização de uma das respostas: *agree* ou *reject*.

É válido ressaltar que o resultado final é informado pelo consumidor. Neste modelo o consumidor pode também abrir uma outra sessão de negociação, iniciando na fase da *Chamada*. Antes de iniciar a nova sessão, o usuário pode alterar os valores e os pesos assinalados a cada atributo.

A Figura 7 ilustra a última fase do protocolo de negociação. Nesta Figura a proposta do *SellAgent3* é rejeitada, enquanto que o do *SellAgent4* é aceita pelo consumidor.

4 Implementação

O modelo foi implementado em Java utilizando o JDK versão 1.2.2 (*Java Development Kit*) e o sistema de agentes utilizado foi o *Grasshopper*. Esta plataforma de agentes foi adotada, em

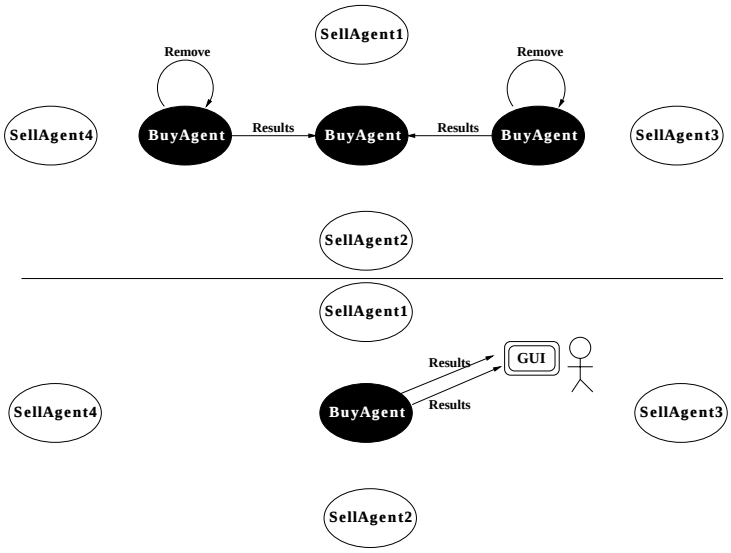


Figura 6: Cenário proposto para a fase da Apresentação

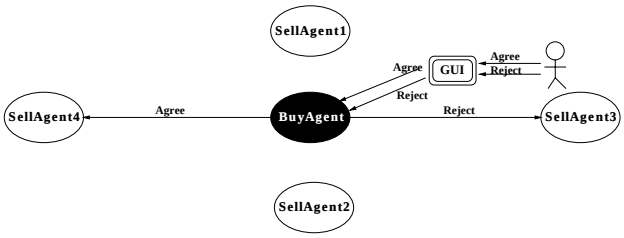


Figura 7: Cenário proposto para a fase da Conclusão

virtude da mesma possuir a interface MASIF (*Mobile Agent System Interoperability Facility*) [14]. Esta interface é importante para o modelo, uma vez que ela provê a interoperabilidade entre sistemas de agentes diferentes. Como o nosso modelo é voltado para aplicações de comércio eletrônico, a interoperabilidade torna-se um requisito indispensável. Os catálogos foram escritos em XML e a interface gráfica utilizada pelos usuários foi codificada em Java *Swing* que é parte do JFC (*Java Foundation Class*), podendo ser utilizada junto com o *JDK 1.2.2*.

Com finalidade de validar o modelo proposto, um protótipo foi desenvolvido para aplicações de compra e venda de automóveis. As Figuras 8 e 9 ilustram o formulário para pesquisa *fixa* e pesquisa *negociável*, respectivamente.

4.1 Agentes de Compra e Venda

A plataforma *Grasshopper* é composta de *regiões*, *agências* e *lugares*. De acordo com [8], uma *agência* é o local onde os agentes estacionários e móveis são executados. *Região* é um conceito utilizado no *Grasshopper* que possui a finalidade de facilitar o gerenciamento dos componentes distribuídos (*agências*, *lugares* e *agentes*). *Lugar* permite um agrupamento lógico entre os agentes pertencentes a uma mesma *agência*. As *agências* juntamente com os *lugares* podem ser associados a uma determinada *região* permitindo um nível de agrupamento maior.

A comunicação entre o agente de venda e de compra é baseado no *serviço de comunicação*

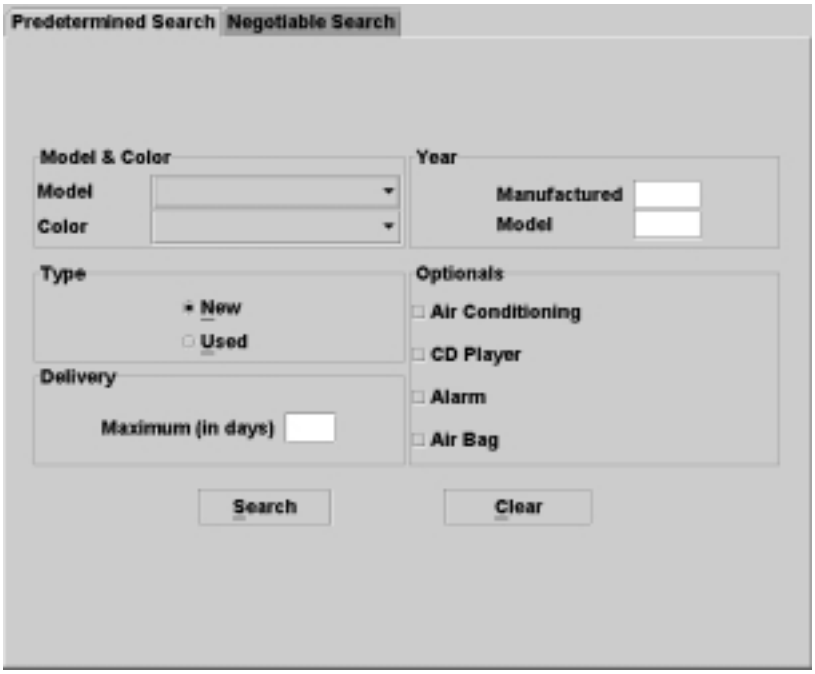


Figura 8: Formulário utilizado para pesquisa *fixa*

disponível no *Grasshopper*. No nosso protótipo, o agente de venda é executado como se fosse o servidor que está a espera de requisições dos clientes. Os clientes seriam os agentes de compra que chamam métodos do agente de venda (servidor) para realizar a chamada por propostas, assim como para ofertar e requisitar propostas.

Existem vários tipos de comunicação implementados no *Grasshopper*. O mecanismo assíncrono foi utilizado para evitar que o agente de compra ficasse bloqueado durante o período em que o servidor estivesse processando a requisição do cliente.

No protótipo construído, a *chamada por propostas* é enviado a todos os agentes de venda que fazem parte de uma mesma *região*. Portanto, os agentes de venda podem estar em *agências* separadas, desde que façam parte de uma mesma *região* para que os mesmos recebam a chamada. A partir desta conclusão, é possível afirmar que o *marketplace* do nosso protótipo é composto por todos os agentes de venda que pertencem a uma determinada *região*.

Cada transição presente no modelo é chamada invocando-se um método da interface *SellAgent* (Figura 10).

O método *CallForProposals* é utilizado para chamar pelas propostas e possui como parâmetro o vetor que armazena os atributos do veículo a ser encontrado, retornando outro vetor que possui as propriedades do veículo disponível no catálogo que mais se assemelha ao veículo requisitado.

Os métodos *request* e *suggest* são utilizados para requerer por novas sugestões. Ambos os métodos possuem como parâmetro a classe *Identifier*, implementada no sistema *Grasshopper* para identificar cada agente. *Suggest* retorna um *String* usado para certificar de que a transição foi bem sucedida. No protótipo implementado, *request* é utilizado também para formalizar a seleção dos agentes de venda que serão interagidos. *Request* retorna os endereços onde os agentes de venda estão localizados, uma vez que esses endereços são necessários para informar os *hosts* para onde as cópias dos agentes de compra serão migrados.

Propose e *offer* são utilizados na fase da *negociação* para ofertar propostas e contrapropos-

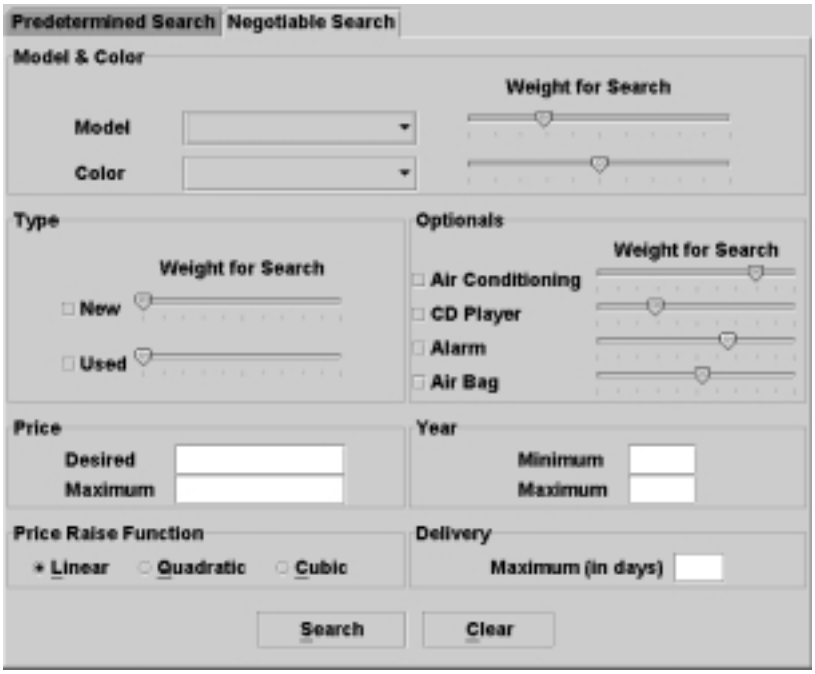


Figura 9: Formulário utilizado para pesquisa *negociável*

tas. Ambos os métodos passam como parâmetros o identificador e o preço sugerido. Por outro lado, durante a fase da negociação *propose* e *offer* retornam a contraproposta da oferta enviada.

Agree e *reject* foram implementados para aceitar ou rejeitar a proposta ofertada. Os referidos métodos possuem como parâmetro os identificadores de agente e retornam um *String* que confirma a execução da operação chamada.

No modelo implementado o vendedor pode selecionar uma das três estratégicas de geração dos preços a serem ofertados: *linear, quadrática* e *cúbica*. Da mesma forma, o comprador pode também selecionar um dos mecanismos para elevar o preço a ser ofertado. A Figura 11.a ilustra os gráficos da geração das ofertas para o vendedor. Por outro lado, o gráfico do comprador seria um semelhante ao apresentado na figura, porém com o valor do preço sendo elevado durante o decorrer do tempo.

A cópia do agente de compra para o *host* do agente de venda é realizada apenas para a barganha de preço, uma vez que a troca de mensagens é bem maior durante esta fase. A cópia é realizada através do método *copy()* existente no *Grasshopper*. Este método necessita do endereço de destino que no modelo seria o *host* do vendedor. O *Grasshopper* também provê o método *beforeCopy()* que é utilizado para codificar as instruções referentes ao armazenamento temporário dos atributos do automóvel, até que complete a barganha de preço. A exclusão das cópias é realizada através do método *remove()* disponível também no *Grasshopper*.

4.2 Catálogos

Os catálogos foram implementados em XML com a finalidade de prover a interoperabilidade necessária nos sistemas de comércio eletrônico. Para rodar o protótipo, foi construído um pequeno catálogo para armazenar os automóveis disponíveis para a venda. A Figura 11.b apresenta parte do catálogo usado no protótipo.

```
package BuySell;

import de.ikv.grasshopper.type.*;
import de.ikv.grasshopper.communication.*;

public interface SellAgent
{
      public String[] CallForProposals(String[] vetarg);
       public String suggest(Identifier agentId);
       public GrasshopperAddress request(Identifier agentId,long pricebidden);
       public String agree(Identifier agentId);
       public String reject(Identifier agentId);
       public String propose(Identifier agentIdent,long pricebidden);
       public String offer(Identifier agentIdent,long pricebidden);
}
```

Figura 10: Métodos utilizados para invocar as transições

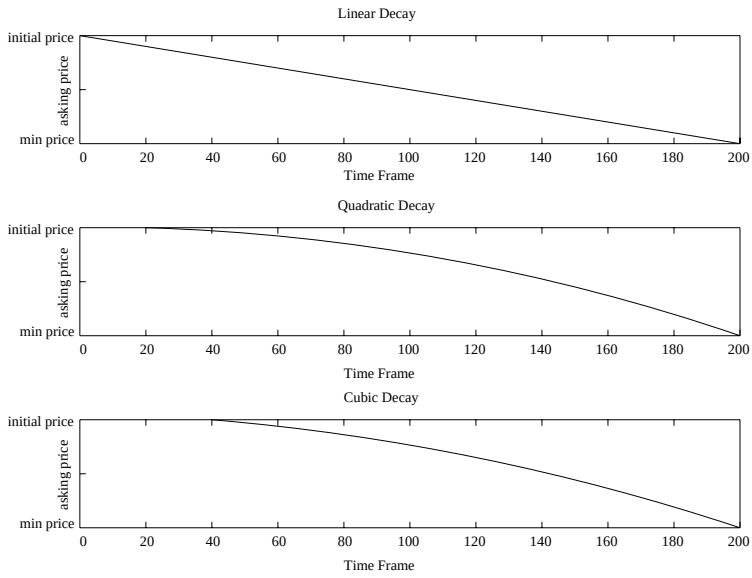
A integração entre Java e XML é implementada no nosso modelo, através do uso da API (*Application Program Interface*) DOM (*Document Object Model*) [9] que transforma um documento XML em uma árvore de objetos. A API SAX (*Simple API for XML*) [9] foi avaliada como uma API alternativa, porém para o nosso modelo DOM se mostrou mais vantajoso, uma vez que o catálogo é inteiramente lido. Além disso, SAX requer que o desenvolvedor escreva o seu próprio gerenciador de eventos que possa tratar cada parte do documento lido. Informações adicionais sobre os catálogos em XML podem ser encontradas em [7, 5].

5 Trabalhos Relacionados

Existem vários trabalhos desenvolvidos na área de negociação em comércio eletrônico, porém grande parte da pesquisa nessa área é focalizada para solucionar os mecanismos de leilão como é o caso do *OFFER* [10], que trata da negociação entre um grupo de participantes. Outro trabalho na área diz respeito ao uso da similaridade para realizar negociações [17]. Este trabalho utiliza a notação *fuzzy* para determinar as similaridades e assim poder ofertar propostas que sejam condizentes com a realidade de ambas as partes.

Outros trabalhos existentes utilizam os mecanismos de aprendizado com objetivo de prover a melhor estratégica de barganha. Estes mecanismos obtiveram resultados satisfatórios para solucionar certos problemas, porém a sua maior desvantagem é o tempo decorrido para o seu aprendizado e a complexidade gerada para desenvolver este modelo. É válido citar neste momento que o uso dos algoritmos genéticos proposto em [12], requer cerca de 20 gerações até 4000 gerações para obter uma boa estratégica de negociação, segundo um estudo no artigo [1].

O protocolo de negociação e as medidas de similaridades que foram propostos no presente artigo não estão baseados em nenhuma abordagem complexa que requeira conhecimentos específicos para que os mesmos sejam implementados. Além disso, o mecanismo para calcular a similaridade é bastante simples se comparado com as outras técnicas que utilizam o aprendizado para desempenhar as mesmas funções.



(a) Gráfico para geração do preço a ser proposto pelo vendedor

```
<?xml version = "1.0"?>
<AUTOMOBILES>
   <AUTOMOBILE>
      <MODEL>HONDA CIVIC EX</MODEL>
      <TYPE>USED</TYPE>
      <YEAR>1999</YEAR>
      <COLOR>BLUE</COLOR>
      <PRICE>9000</PRICE>
      <DELIVERY>0</DELIVERY>
      <OPTIONAL>
        <AIRCOND>YES</AIRCOND>
        <CD>NO</CD>
        <ALARM>NO</ALARM>
        <AIRBAG>YES</AIRBAG>
      </OPTIONAL>
   </AUTOMOBILE>
   <AUTOMOBILE>
      <MODEL>HONDA ACORD EX</MODEL>
      <TYPE>NEW</TYPE>
      <YEAR>2001</YEAR>
      <COLOR>WHITE</COLOR>
      <PRICE>25000</PRICE>
      <DELIVERY>0</DELIVERY>
      <OPTIONAL>
        <AIRCOND>YES</AIRCOND>
        <CD>YES</CD>
        <ALARM>YES</ALARM>
        <AIRBAG>YES</AIRBAG>
      </OPTIONAL>
   </AUTOMOBILE>
<AUTOMOBILES>
```

(b) Catálogo em XML

Figura 11: Geração de preços e parte do catálogo utilizado no protótipo

6 Conclusão

O presente trabalho contribuiu com um protocolo de negociação para comércio eletrônico. Uma das vantagens deste modelo é o paralelismo na negociação. Uma vez que cada cópia do agente de compra é copiado para o *host* do agente de venda, negociações autônomas e paralelas podem ser operacionalizados, otimizando bastante o tempo de procura do consumidor por um determinado produto. A outra contribuição deste trabalho diz respeito à medida de similaridade. O mecanismo apresentado não se baseia em bases de dados, onde são armazenados uma grande quantidade de informações que poderão ajudar na determinação das similaridades. Além disso, a mesma também não é baseada nos mecanismos de aprendizado, porém a métrica proposta é determinada de forma simples, bastando que o usuário indique o seu perfil de preferência através dos valores dos pesos requisitados.

A interoperabilidade é alcançada através do uso de *XML* na confecção dos catálogos, assim como na implementação em Java para prover um código portável entre arquiteturas diferentes.

Como trabalho futuro pretendemos estender o uso da medida da similaridade também na fase da negociação, permitindo que propostas e contrapropostas possam ser ofertadas a partir do uso dessas medidas, até que o protocolo encontre uma métrica que seja mais adequada frente à requisição do consumidor.

Uma outra idéia que estamos investigando diz respeito à abrangência dos agentes a serem interagidos. Pretendemos que o nosso protocolo tenha acesso a outros domínios de serviços gerenciados por outros sistemas de agentes. Esta idéia é possível de ser implementada, posto que o *Grasshopper* possui interface *MASIF* que provê a interoperabilidade entre diferentes plataformas de agentes. Um outro mecanismo que concluímos ser importante para o bom desempenho deste protocolo é a inserção de um mecanismo que limite o tempo máximo de negociação.

Agradecimentos. Este trabalho foi parcialmente financiado por FAPESP, CAPES, CNPq e PRONEX.

Referências

[1] Carrie Beam and Arie Segev. Electronic catalogs and negotiations, Agosto 1996. CITM Working Paper 96-WP-1016, Fisher Center for Information Technology and Management, University of California at Berkeley.

[2] Anthony Chavez and Pattie Maes. Kasbah: An agent marketplace for buying and selling goods. In *First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'96)*, pages 75–90, 1996.

[3] CommerceNet. The eCo Framework Standardization Project. `http://www.commerce.net/projects/currentprojects/eco/`.

[4] RosettaNet Consortium. Rosettanet standard. `http://www.rosettanet.org/`.

[5] Jó Ueyama e Edmundo R. M. Madeira. Aplicação de Catálogos XML para Comércio Eletrônico. *Developer's Magazine*, (46):20–28, Junho 2000.

[6] Jó Ueyama e Edmundo R. M. Madeira. An Automated Negotiation Model for Electronic Commerce. IEEE Fifth International Symposium on Autonomous Decetralized Systems - ISADS'01, Dallas, EUA, Março de 2001 (Aceito para Publicação).

[7] Jó Ueyama e Edmundo R. M. Madeira. Using XML for Electronic Catalogs. International Workshop on Information Integration on the Web - WIIW'2001, Rio de Janeiro, Brasil, Abril de 2001 (Aceito para Publicação).

[8] IKV. Grasshopper - The agent platform. Users' Guide.

[9] JavaWorld. XML for the Absolute Beginner: a Guide Tour from HTML to Processing XML with Java, Abril 1999.

[10] Carrie Beam Martin Bichler and Arie Segev. OFFER: An object framework for electronic requisitioning, December 20, 1997. CITM Working Paper 97-WP-1026, Fisher Center for Information Technology and Management, University of California at Berkeley.

[11] Martsoft. IntuiCat application suite. `http://www.martsoft.com/products/index.html#INTUICAT`.

[12] Jim Oliver. A machine learning approach to automated negotiation and prospects for electronic commerce. *Journal of Management Information Systems*, 13(3):83–112, Winter 1996-97.

[13] OMG. Trading Object Service Specification, 1997.

[14] OMG. MASIF (Mobile Agent System Interoperability Facility), 1998.

[15] OMG. Negotiation Facility - Final revised submission, Março 1999.

[16] OMG. E-Commerce and the OMG (white paper), Fevereiro 2000.

[17] C. Sierra P. Faratin and N. R. Jennings. Using similarity criteria to make negotiation trade-offs. In *Proc. 4th Int. Conf. on Multi-Agent Systems (ICMAS-2000), Boston, EUA*, pages 119–126, 2000.

[18] R.G Smith. The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver. 12:1104–1113, 1980.

[19] Requisite Technology. Technology and Services. `http://www.requisite.com`.

[20] W3C. XML specification. `http://www.w3.org/XML/`.

[21] World Wide Web Consortium (W3C). Página oficial. `http://www.w3.org/`.