AgentSpace versus Aglets:

Infraestruturas de Agentes para as Futuras Aplicações da Internet

Alberto Silva, José Delgado
{Alberto.Silva, Jose.Delgado}@inesc.pt
INESC & IST - Universidade Técnica de Lisboa
Rua Alves Redol, n.º 9, 1000 LISBOA, PORTUGAL

Resumo

Argumenta-se nesta comunicação que o paradigma de agentes de software deverá ser adoptado com vantagem nas futuras aplicações para a Internet. Esta tese é discutida e defendida através de uma aplicação de exemplo designada por CELIA – Comércio Electrónico de Livros na Internet de âmbito Aberto. Introduz-se a infraestrutura de agentes AgentSpace, com base na qual, e na infraestrutura Aglets, são desenvolvidos protótipos da aplicação CELIA. Por fim, comparam-se e discutem-se os resultados obtidos quer em termos de desempenho, quer ao nível das facilidades de programação providenciados por ambos os sistemas.

1 Introdução

A Internet é um vasto espaço de informação, de utilizadores, de comunidades e de interesses. Apesar dos rápidos avanços tecnológicos e da existência de alguns serviços Web não existem ainda modelos e tecnologias que suportem o aparecimento de aplicações para a Internet mais versáteis, dinâmicas e interdependentes.

Nesta comunicação defende-se a tese de que o modelo de agentes de software [GK94, WJ95, BTV96] deverá ser aplicado com vantagem na construção de um número significativo das futuras aplicações para a Internet. Estas aplicações serão caracterizadas principalmente por serem dinâmicas, distribuídas e suportarem um número elevado (da ordem dos milhares a centenas de milhares) de utilizadores. Esta classe de aplicações – designada de ora em diante por "aplicações baseadas em agentes" (ABA) – é inovadora, pois não corresponde à noção tradicional de aplicação que é gerida por determinada pessoa ou organização. Em vez dessa visão, esta classe de aplicações é melhor entendida como

- Uma teia de agentes, em que tipicamente cada agente corresponde a uma mini-aplicação.
- Possuída e gerida por um número dinâmico de entidades com diferentes objectivos e atitudes (possivelmente em conflito).
- Encontrando-se distribuída por distintos locais computacionais (tais como PC, computadores de grande porte, ou telefones móveis).

De forma a validar a tese principal desta comunicação, é apresentado um caso de aplicação, designado por CELIA – Comércio Electrónico de Livros na Internet de âmbito Aberto. São confrontadas e discutidas as aproximações baseadas na tecnologia actual da Web [SMD97a] versus a aproximação baseada em agentes.

Introduz-se a arquitectura AgentSpace [SMD97b, SMD98] desenvolvida pelos autores, que consiste numa infraestrutura para suporte, desenvolvimento e gestão de ABA. Foram desenvolvidos protótipos da aplicação CELIA para o AgentSpace do INESC e o sistema Aglets Workbench [IBM97] da IBM. Com base nestes protótipos iniciais apresenta-se nesta comunicação uma primeira comparação entre ambos os sistemas tendo em conta (1) desempenho de execução e (2) as funcionalidades providenciadas.

Esta comunicação encontra-se organizada em cinco secções. A Secção 2 introduz a visão das futuras aplicações para a Internet e o caso de estudo CELIA. A Secção 3 apresenta a visão geral da arquitectura e do modelo de objectos do AgentSpace. Na Secção 4 descreve-se a implementação da aplicação CELIA e as diferentes simulações realizadas para os sistemas AgentSpace e Aglets. Adicionalmente apresentam-se os resultados obtidos e faz-se uma análise critica dos mesmos. Por fim, no Capítulo 5 resume-se as principais conclusões e contributos da comunicação.

2 Motivação: As Futuras Aplicações da Internet

2.1 Visão da Internet para o Futuro

A visão do futuro da Internet aqui esboçada é antes de mais uma atitude de evolução relativamente à situação actual conhecida. O contributo proposto é apenas uma visão parcelar do problema. Não são referidos, por exemplo, assuntos relacionados com protocolos de baixo nível, tecnologia de redes de banda larga, realidade virtual, etc.

A concepção do que será a Internet num futuro não muito longínquo está intimamente associada às teses e discussões que envolvem os temas das **Sociedades de Informação** e das **Infraestruturas de Informação Nacionais** [Ban94, NRC94, Mis97]. Basicamente as sociedades de informação do futuro preconizam a existência de infraestruturas físicas, lógicas e organizacionais que as suportem. O objectivo último desta visão é permitir ao cidadão comum o acesso às principais fontes de informação e de conhecimento disponíveis electronicamente. E isto de forma fácil, económica e (tanto quanto possível) "democrática" (i.e., acessível à generalidade dos cidadãos).

A computação centrada no computador pessoal, a comunicação centrada no telefone/fax individual, e a difusão (passiva) de informação centrada na televisão/rádio evoluirão conjuntamente de forma que a computação, comunicação, pesquisa e acesso à informação serão todos suportados pela "rede".

O utilizador final, em vez de utilizar ferramentas básicas e primitivas, tais como clientes FTP, News, ou Archie, utilizará uma ou mais aplicações especializadas – **agentes** – que actuarão em seu nome de modo a realizar as tarefas requeridas. Deste modo a Internet tornar-se-á num espaço aberto de agentes, onde distintos agentes possam "viver", negociar, vender e comprar, serem intermediários no estabelecimento de relações entre outros agentes, procurarem parceiros, resolverem colaborativamente tarefas complexas, etc.

Tendo em conta todos os interesses e actividades emergentes, é de prever que a Internet evolua de um espaço estático com utilização/navegação principalmente directa (por acção do ser humano), para um espaço complexo e dinâmico com um maior número de utilização indirecta (por intermédio da acção dos agentes de software) e um número crescente de interacções entre agentes. Esta visão pode ser perspectivada segundo diferentes pontos de vista

- **Utilizador doméstico**: Este possuirá um ou mais agentes (alojados em distintos servidores de agentes), cada qual especializado numa gama de possíveis actividades, tais como: pesquisa e obtenção de informação, notificação de ocorrência de eventos especiais, gestão "inteligente" de correio electrónico, gestão de contas bancárias, compra de produtos, tratamento de questões burocráticas com a administração pública, etc.
- **Organizações de telecomunicações**: Estas organizações, para além das suas funções existentes actualmente, providenciarão "espaço computacional" para instalação e gestão de agentes. Poderão providenciar serviços de valor acrescentado, tais como serviços de contabilização, de monitorização, acesso e gestão de bases de dados, conjuntos de (tipos de) agentes prontos a serem instanciados, etc.
- **Organizações** (cuja área de negócio não são as telecomunicações): Estas providenciarão um leque de agentes especializados com objectivos e comportamentos bem definidos. Esses agentes serão responsáveis por inúmeras tarefas, tais como: venda e compra de produtos ou serviços, estatísticas e *marketing* da organização, promoção e distribuição de outros agentes (de nível utilizador final), etc.

Esta visão requer, todavia, novos esforços e desafios designadamente no que se refere entre outras às seguintes tecnologias: mecanismos de contabilização e pagamentos; linguagens de comunicação e interoperação entre agentes; linguagens de representação de conhecimento/conteúdo das mensagens e dos seus formatos; mecanismos de segurança e de robustez.

2.2 Caso de Estudo: CELIA

Considere-se uma aplicação de comércio electrónico de livros baseada na Internet/Web e de âmbito aberto (CELIA). Considere-se a existência de pelo menos três tipos distintos de entidades: clientes/leitores, vendedores/livreiros e mediadores/intermediários.

A aplicação CELIA deverá suportar o aparecimento e desaparecimento de quaisquer das entidades referidas de forma dinâmica e fácil. Clientes e livreiros interactuam de forma a averiguarem a existência de determinado livro e a negociarem as melhores condições de transação. Os mediadores providenciam serviços de páginas amarelas (de livreiros e eventualmente de livros) e de notificação (e.g., a possibilidade de um cliente ser notificado do aparecimento de um novo livreiro com determinadas características, ou de um novo livro).

Suponham-se, para efeito de concepção e construção da aplicação CELIA, as seguintes possíveis abordagens:

- Baseada na tecnologia conhecida da Web
- Baseada no paradigma de computação baseado em agentes

2.2.1 Limitações da Tecnologia Actual da Web

Segundo a tecnologia corrente da Web, há basicamente duas possibilidade de concepção da aplicação CELIA: baseada numa solução centralizada ou numa solução descentralizada.

A **solução centralizada** exigiria que uma organização promotora da aplicação CELIA, desenvolvesse e mantivesse o sistema numa máquina gerida por si (vid. Figura 1). A aplicação providenciaria capacidades de gestão de clientes com as suas características relevantes (perfis de leitura, interesses pessoais, temas e autores preferidos, etc.), assim como as funcionalidades para gestão de informação de livreiros.

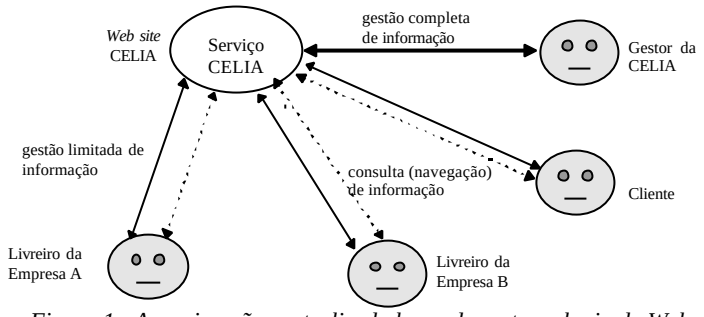

Figura 1: Aproximação centralizada baseada na tecnologia da Web.

Esta abordagem exigiria o devido tratamento de controlo de acessos e de segurança, e apresentaria pouca autonomia das entidades pertencentes ao CELIA e limitações ao nível do dinamismo do sistema. Por exemplo, não seria fácil a introdução de novas capacidades e funcionalidades no sistema, já que tal exigiria, à partida, o acordo concertado entre, pelo menos, os principais livreiros associados.

Nesta abordagem os intervenientes não estão satisfeitos porque não podem controlar os seus próprios serviços (e.g., através de páginas HTML ou applets Java) quando e como desejam, de forma a distinguirem-se entre si. Esta abordagem traduz uma situação de compromisso, implicando negociações por vezes difíceis e tornando-se, por conseguinte, pouco adequada em aplicações com muitos intervenientes e pressupostamente dinâmicas. No entanto, esta abordagem apresenta um desenvolvimento fácil das aplicações e principalmente uma gestão e manutenção simples de realizar.

Por outro lado, a **solução descentralizada** seria baseada em um servidor de suporte ao CELIA mas apenas com os serviços básicos necessários, nomeadamente o serviço de "páginas amarelas". O objectivo deste serviço seria o registo de todos os livreiros com as suas principais informações, tais como endereço de correio electrónico e de servidor Web. Deste modo providenciar-se-ia uma maior autonomia e flexibilidade, já que cada livreiro poderia criar e manter o seu próprio serviço. Todavia, não seria providenciado qualquer suporte a tarefas mais complexas como seja a automatização do processo de procura de livro, ou mesmo da sua compra. Da perspectiva dos livreiros, estes teriam de manter os seus próprios serviços, com funcionalidades semelhantes tais como catálogos de livros pesquisáveis por título, autor, tema, etc., e suporte a transações comerciais.

Adicionalmente os livreiros teriam de registar (manualmente) os seus serviços em servidores conhecidos de páginas amarelas. Por outro lado, os clientes para comprarem um determinado livro, teriam de consultar um ou mais serviços de páginas amarelas, e a partir daí aceder aos serviços de um conjunto alargado de livreiros. Mais tarde, com base na pesquisa anterior, efectuarem o pedido de compra ao livreiro com melhores condições.

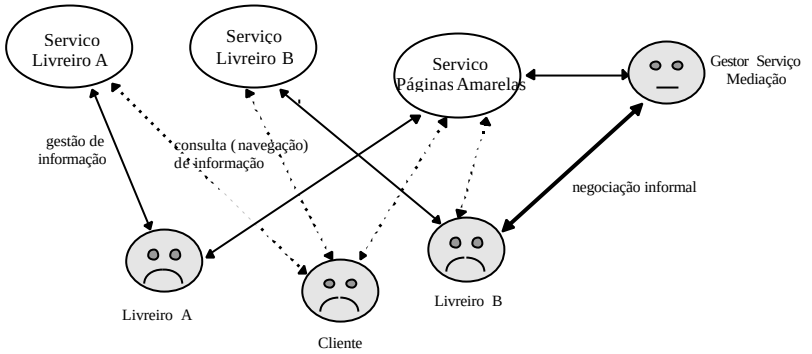

Figura 2: Aproximação descentralizada baseada na tecnologia da Web.

Por estas razões, estes utilizadores não têm motivos para ficarem contentes, conforme apresentado no esquema simplificado da Figura 2.

2.2.2 Abordagem Baseada em Agentes

Uma aproximação baseada em agentes, adaptada às características da CELIA, exigiria a existência de uma ou mais organizações com funções particulares de **mediação**, funcionando quer como serviços de páginas amarelas, quer como serviços de controlo e acompanhamento. O número deste tipo de serviços seria tipicamente reduzido. O tipo de acordos de interacções entre mediadores poderia ser bilateral ou concertado entre todos eles. Por outro lado, os **livreiros** registar-se-iam em um ou mais mediadores de modo a tornarem-se membros da CELIA e só depois a tornarem-se conhecidos, serem solicitados, ou poderem solicitar informação e serviços. Por fim, os **clientes** também se poderiam registar eventualmente nesses serviços de forma a serem notificados de eventos relevantes, ou de forma a terem a garantia de utilização de transacções seguras, com promoções, etc.

Numa abordagem baseada em agentes cada utilizador interactua com o seu agente especializado definindo-lhe e delegando-lhe tarefas mais ou menos complexas. Os agentes interactuam entre si de forma a realizarem as tarefas que lhe foram delegadas. A Figura 3 ilustra uma visão esquemática e de alto nível desta aproximação. (Para simplificar o modelo, assumiu-se que na CELIA existiria apenas um único agente com características de mediação, designado por agente mediador ou agente de páginas amarelas, que seria responsável pela gestão, manutenção, controlo de acessos e atribuição de identificações.)

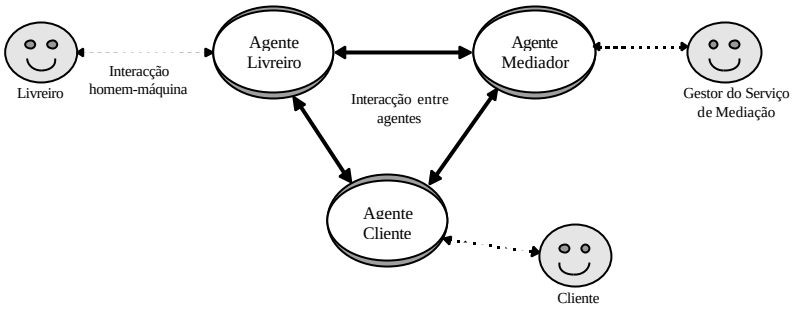

Figura 3: Aproximação baseada em agentes.

Cada interveniente desenvolveria de forma independente os seus próprios agentes. Esta aproximação não exige que cada interveniente desenvolva necessariamente os seus próprios agentes. Pode-se supor, por exemplo, que numa fase inicial seria divulgado e disponibilizado aos membros da CELIA o código comum de dois tipos de agentes: o agente para livreiros, e o agente para clientes. Estes intervenientes instalariam, criariam (por instanciação do tipo/classe providenciado) e poderiam gerir os seus próprios agentes, que apresentariam por conseguinte um comportamento semelhante. Em fases subsequentes poderiam surgir efectivamente novas versões de agentes – quer para livreiros, quer para clientes –, desenvolvidos especificamente para esta ou aquela entidade, com funcionalidades adicionais, mantendo contudo compatibilidade de interacção com os agentes dos restantes intervenientes.

O agente mediador, apresentaria funcionalidades e características particulares. Seria ele o responsável pela criação e gestão da comunidade CELIA. Pelo facto da comunidade CELIA ser de acesso aberto, o agente mediador poderia ser também o responsável pela autenticação dos agentes das diferentes entidades.

No entanto, de forma que todos os agentes pudessem comunicar entre si, seria exigido que as diferentes entidades chegassem a acordo sobre um protocolo/interface específico, comum a todas as aplicações envolvidas. Esta tarefa, não sendo por si fácil de implementar, assume que, na prática, um número reduzido de entidades (e.g., os livreiros) os definissem e que todas os restantes intervenientes os aceitassem e adoptassem. Esta questão é relativamente semelhante à descrita na abordagem centralizada, anteriormente referida. Contudo, enquanto na abordagem anterior a autonomia é limitada ao nível de configuração, gestão e manutenção dos serviços, na abordagem baseada em agentes a autonomia é apenas limitada ao nível da especificação do protocolo/interface de interacção entre agentes.

Esta solução, comparativamente com as abordagens descritas na secção anterior, apresenta as seguintes vantagens principais:

- **Descentralizada** e **escalável**: O único ponto de congestão é o agente mediador, o qual pode ser desagravado pela introdução incremental de novos agentes semelhantes.

- **Dinamismo**: Qualquer agente pode (com maior ou menor facilidade) entrar ou sair da comunidade. A funcionalidade e complexidade dos diferentes agentes pode evoluir de forma independente e transparente de todos os restantes.
- **Autonomia** e **flexibilidade**: Cada entidade interveniente é responsável pela construção e manutenção das suas próprias aplicações e agentes em particular.
- **Interacção indirecta**: O facto dos utilizadores apenas terem de interactuar com os seus agentes, elimina os problemas da descontinuidade ao nível de interface com o utilizador, e reduz o número de interacções, nomeadamente as interacções humanas, por vezes demoradas e difíceis. O utilizador interactua com o seu agente, dando-lhe informação e tarefas precisas.

2.3 Desafios em Abordagens Baseadas em Agentes

Aparentemente a abordagem baseada em agentes parece oferecer um paradigma adequado para concepção de aplicações do tipo da CELIA. Contudo, existem alguns problemas/desafios que são referidos de seguida:

- **Designação ambígua**: O facto de um agente suscitar inúmeras interpretações, apresentar um conjunto de características alargado, ser influenciado por diferentes comunidades científicas e existir em diferentes áreas de aplicação, conduz a uma ambiguidade e confusão generalizada do que é, de facto, um agente de software. Podendo conduzir a discussões pouco produtivas e a um consequente descrédito geral. Por conseguinte, quando se falar numa aplicação baseada em agentes é fundamental definir-se o contexto respectivo.
- **Desconfiança do utilizador**: O utilizador não está habituado, em geral, a interactuar de forma indirecta com o computador. Não está habituado a utilizar aplicações que lhe executam tarefas de forma autónoma. Entre outros, surge o "síndroma da competência e da confiança" referido por Maes [Mae94]. Por um lado, o utilizador tem de ter confiança nos agentes que utiliza – nomeadamente, confiança nos fabricantes dos agentes em causa –, de forma a delegar-lhe tarefas relativamente complexas e/ou criticas. De forma a facilitar essa confiança, o utilizador deverá poder monitorizar o estado corrente dos seus agentes e eventualmente obter um registo de actividades realizadas, com maior ou menor detalhe. Por outro lado, os agentes têm de mostrar que são competentes/capazes de realizar adequadamente e com flexibilidade as tarefas para os quais foram desenvolvidos e configurados.
- **Novos modelos de negócio**: Por vezes os problemas da adopção/utilização de agentes não tem a ver com questões tecnológicas, mas sim com a adequação e integração das novas aplicações aos modelos de negócio existentes ou emergentes. O caso do BargainFinder [And96], em que os serviços Web de venda de CD impediram o acesso de seus pedidos, é um exemplo paradigmático desta questão. Ou seja, embora de um ponto de vista tecnológico seja possível, por exemplo conceber agentes de pesquisa do melhor preço de um produto, com eventual compra correspondente, ou agentes de jornais electrónicos personalizados, é fundamental que as soluções tecnológicas sejam acompanhadas com correspondentes soluções legais e de carácter ético. Por exemplo, embora sendo possível tecnicamente, é legal/ético ou não, a criação de um jornal

electrónico personalizado a partir da informação providenciada periodicamente nos principais jornais electrónicos Portugueses, sem o consentimento e a referência expressa dos produtores da fonte de informação?

Desta forma, o modelo de agentes de software, nomeadamente no contexto da Internet, vem suscitar novos modelos e oportunidades de negócio, entre outros: a desintermediação e a redução de cadeias de valor em mercados electrónicos; a produção de informação especializada e adaptada aos interesses/necessidades do utilizador; ou a banca virtual personalizada.

- **Inexperiência de desenvolvimento**: A área de concepção e implementação de aplicações baseadas em agentes é relativamente recente, nomeadamente segundo a perspectiva da Engenharia de Software. Não existem propostas maduras de métodos de desenvolvimento de software para este conjunto de aplicações, nem tão pouco ambientes e ferramentas de suporte e de desenvolvimento. Este último aspecto consiste no principal desafio abordado nesta tese: propõe-se uma infraestrutura de suporte, desenvolvimento e de gestão de aplicações dinâmicas e distribuídas baseadas em agentes, de forma que seja mais fácil e eficiente o desenvolvimento de aplicações, mas também, que seja mais fácil a utilização/gestão das mesmas.
- **(in)Segurança**: No âmbito de aplicações baseadas em sistemas de agentes móveis, são expectáveis os benefícios referidos na secção anterior, mas levantam-se, por outro lado, questões importantes do ponto de vista da segurança. O facto de um agente poder mover-se para uma máquina remota, e aí executar-se, exige mecanismos de segurança a diferentes níveis e segundo diferentes perspectivas, nomeadamente segurança (1) do local remoto relativamente ao agente móvel: o sistema destino não pode aceitar agentes "desconhecidos" (e.g., agente cuja classe/código é desconhecida, ou não é certificada); (2) do agente relativamente ao local remoto: o agente não pode migrar para um local remoto, se não tiver a garantia de que é um local seguro ou certificado, de forma que o seu comportamento e os seus dados não sejam corrompidos ou modificados; (3) do canal de comunicação: durante a operação de navegação, o agente deverá mover-se preferencialmente num canal seguro, de forma que o seu conteúdo não seja roubado, corrompido ou alterado.

3 Visão Geral do AgentSpace

Os objectivos principais do AgentSpace são o suporte, desenvolvimento e gestão de aplicações dinâmicas e distribuídas baseadas em agentes. Estes objectivos são concretizados através de três componentes integradas conforme ilustrado na Figura 4.

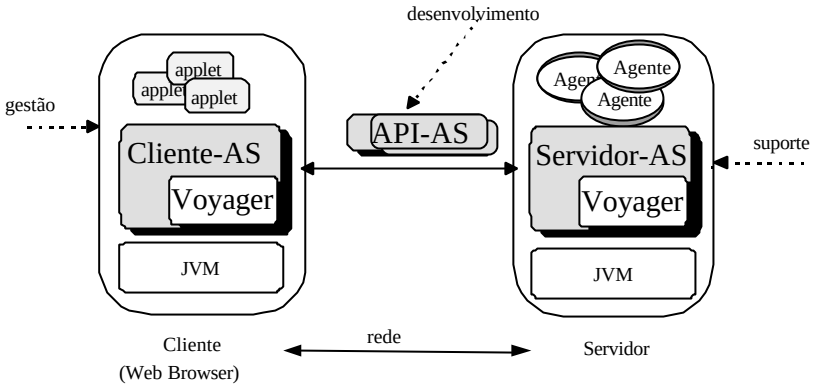

Figura 4: Visão geral da infraestrutura AgentSpace.

- **Servidor AgentSpace (Servidor-AS)**: Consiste em um processo Java, de múltiplas actividades, no qual os agentes são executados. O Servidor-AS providencia vários serviços, designadamente e entre outros: criação de agentes/locais de execução, execução de agentes, persistência, controlo de acessos, suporte à mobilidade e comunicação de agentes, geração de identificadores únicos e globais, e opcionalmente uma interface (*shell*) simples de gestão/monitorização.
- **API AgentSpace (API-AS)**: Consiste em uma biblioteca (*package* na terminologia Java) de classes e interfaces, que definem as regras de como se desenvolve (classes de) agentes, em particular, e aplicações baseadas em agentes, em geral. A API-AS suporta nomeadamente o programador na construção de classes de agentes que são criadas e armazenadas no Servidor-AS para posterior utilização; e classes de applets específicos de forma a providenciar-se uma interface de utilizador para agentes remotos.
- **Cliente AgentSpace (Cliente-AS)**: Consiste em um conjunto de applets Java, armazenados na máquina do Servidor-AS, que permite a gestão e monitorização de agentes e de outros recursos existentes, de forma integrada com a Web/Internet. Oferece a qualquer utilizador, salvaguardando questões de controlo de acessos e de segurança, a possibilidade de gerir facilmente os seus próprios recursos mantidos em um ou mais Servidores-AS.

As componentes cliente e servidor AgentSpace são executadas sobre a máquina virtual Java e utilizam funcionalidades providenciadas pelo Voyager. Ambas as componentes podem ser executadas na mesma, ou em máquinas distintas.

Os agentes são executados sempre no contexto de um servidor. Por outro lado, estes interagem com os utilizadores através de applets (específicos ou genéricos) executados no contexto de clientes Web.

A Figura 5 apresenta um diagrama de classes UML [Rat97] em que são evidenciadas as principais relações de dependência entre as diferentes componentes identificadas. Os subsistemas `as.server`, `as.agentspace`, `as.client` e

`COM.objectspace.voyager` correspondem respectivamente às componentes Servidor-AS, API-AS, Cliente-AS, e ORB Voyager. Não se apresentou propositadamente os subsistemas correspondentes às bibliotecas standard Java de forma a simplificar a leitura da figura.

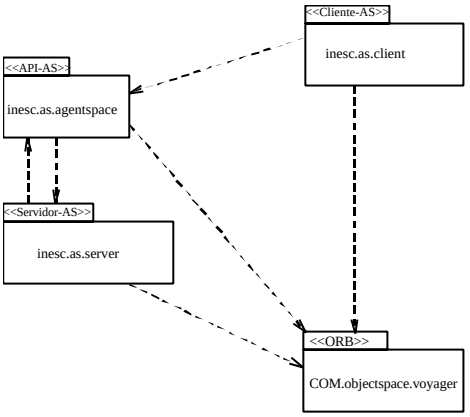

Figura 5: Interdependências entre as diferentes componentes do AgentSpace.

Para mais informações sobre a infraestrutura AgentSpace, designadamente o seu modelo de objectos e a sua interface de programação consulte-se [SMD98] ou informação geral no endereço electrónico `http://berlin.inesc.pt/~agentspc/`. Para informações sobre detalhes de arquitectura relativamente ao agente AgentSpace consulte-se [SD98].

4 Protótipos CELIA em AgentSpace e Aglets

O caso de estudo CELIA anteriormente introduzido (vid. Secção 2.2) foi implementado sobre os sistemas AgentSpace (versão alpha 1) e Aglets WorkBench (versão alpha 5c), ambos sistemas de suporte de agentes móveis Java. O objectivo da sua implementação foi, por um lado, a validação da tese original, e por outro lado, permitir uma comparação entre ambos os sistemas.

Foram apenas desenvolvidas classes de agentes com limitadas capacidades de interacção homem-máquina. Numa situação real ter-se-ia de desenvolver classes específicas de interface (e.g., applets Java no caso da arquitectura AgentSpace) de modo que os utilizadores pudessem facilmente gerir os respectivos agentes.

As listagens do código Java de ambas as versões podem ser obtidas na Web, em `http://berlin.inesc.pt/~agentspc/celia/`.

4.1 Estrutura da Aplicação

São definidas quatro classes de agentes principais para representar os três principais intervenientes no sistema: o cliente, o livreiro, e o mediador. Existem duas classes de clientes de forma a serem comparadas as interacções locais versus remotas.

Adicionalmente, existe a classe de um agente particular que funciona como o simulador da execução da aplicação. Na prática, correspondendo ao agente que cria e controla os restantes agentes, bem como contabiliza os seus tempos de execução.

- `TimeTest`: Classe do agente simulador. Cria, controla e monitoriza os tempos de execução dos restantes agentes.
- `Broker`: Classe do agente mediador. Mantém listas de livreiros e respectivos livros. Tipicamente recebe e trata pedidos quer de agentes livreiros quer de agentes clientes.
- `BookStore`: Classe de agente livreiro. Mantém listas de livros com respectivos preços. Regista-se/desregista-se em um ou mais agentes mediador e responde a pedidos de clientes.
- `ClientMov`: Classe de agente cliente. Este agente procura o livreiro que oferece o melhor preço de um determinado livro. Comunica inicialmente com o agente mediador de forma a obter a lista dos livreiros que supostamente têm o livro especificado. Seguidamente interactua com todos os livreiros obtidos anteriormente de modo a obter o melhor livreiro. (Numa aplicação real, este agente deveria posteriormente iniciar a transação comercial de compra com o livreiro escolhido – esta fase não foi todavia desenvolvida). A política de interacção entre este tipo de agentes e os livreiros é local – implicando que os agentes clientes se movam até ao local de execução dos agentes livreiros, com base em operações de navegação (`moveTo` no AgentSpace e `dispatch` no Aglets).
- `ClientMsg`: Classe de agente cliente. Situação semelhante à descrita relativamente à classe `ClientMov`, só que nesta classe, a política de interacção é remota – implicando que os agentes clientes e livreiros interagem remotamente, a partir dos locais de execução em que foram instalados.

A Figura 6 ilustra através de um diagrama de sequências UML o cenário típico de interacções entre os objectos/agentes envolvidos. A classe `Client` é uma generalização das subclasses `ClientMov` e `ClientMsg`.

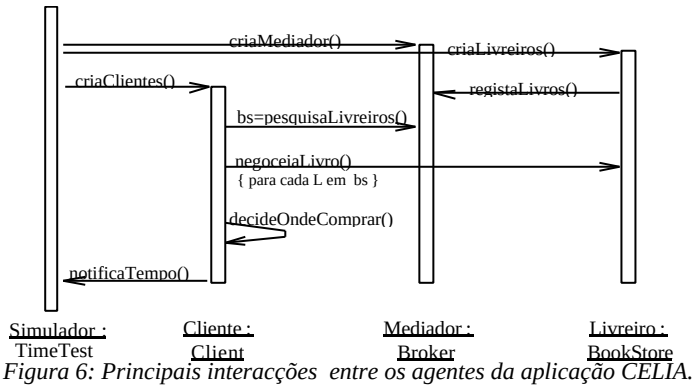

Figura 6: Principais interacções entre os agentes da aplicação CELIA.

4.2 Análise de Desempenho

A análise de desempenho, conforme ilustrado na Figura 6, baseia-se nos tempos de execução dos agentes clientes. Estes agentes quando terminam as suas tarefas (método `beforeDie` no AgentSpace ou `onDisposing` no Aglet) notificam o agente simulador dos tempos de execução das tarefas. Os testes foram realizados no contexto de uma rede local de computadores (LAN), em máquinas Pentium com 16 e 32 Mb de RAM, sobre o JDK 1.1 para ambiente Windows 95. Os resultados apresentados nas Tabelas 1 e 2 estão em milisegundos e correspondem ao tempo médio de execução de 5 clientes.

Foram definidas duas situações de teste, mas em ambas mantiveram-se os seguintes invariantes:

- 1 agente mediador da classe `Broker`
- 5 agentes clientes da classe `ClientMov` e outros 5 da classe `ClientMsg`

E em ambas variou-se o número de agentes livreiros – entre 5 a 50 agentes.

4.2.1 Situação 1: Simulação na mesma Máquina

Na primeira situação, todos os agentes foram criados na mesma máquina.

No entanto, os agentes da classe `ClientMov` continuam a realizar operações de navegação (embora, como é evidente, de forma pouco eficiente). A Tabela 1 apresenta os resultados de testes efectuados nesta situação.

	ClientMov		**ClientMsg**	
Numero de livrarias	**AgentSpace**	**Aglets**	**AgentSpace**	**Aglets**
5	1800	3812	988	110
10	1998	10172	1044	232
25	5282	32052	3558	492
50	13084	61604	8956	1025

Tabela 1: Tempos médios de execução de agentes clientes (Situação 1)

4.2.2 Situação 2: Simulação em várias máquinas

Na segunda situação os agentes foram todos criados na máquina `foca` e depois enviados pelo agente simulador para distintas máquinas conforme ilustrado na Figura 7. O agente mediador manteve-se na máquina `foca`, os clientes foram lançados para a máquina `atenas`, e as livrarias para as máquinas `salmão` e `minie`.

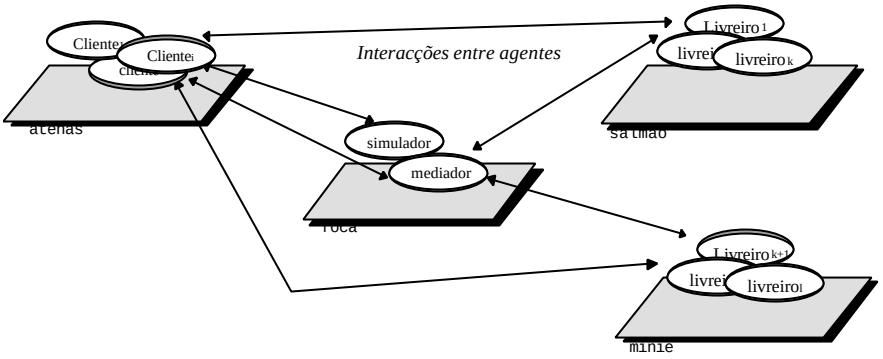

Figura 7: Distribuição de agentes por máquinas de suporte.

A Tabela 2 apresenta os resultados de testes efectuados na situação 2.

	ClientMov		ClientMsg	
Numero de livrarias	**AgentSpace**	**Aglets**	**AgentSpace**	**Aglets**
5	7118	8412	1648	1616
10	15172	18476	3546	3248
25	48356	55484	8208	7370
50	93088	145564	17402	14652

Tabela 2: Tempos médios de execução de agentes clientes (Situação 2).

4.2.3 Discussão dos Resultados

Os resultados relativos à Situação 1 acentuam algumas das particularidades técnicas de ambos os sistemas. É nesta situação que os resultados são mais díspares. Em operações de navegação (versão `ClientMov`) é nítida a vantagem do AgentSpace: a versão Aglets é 110% a 370% mais lenta. Por outro lado, na interacção exclusiva por mensagens (versão `ClientMsg`) é esmagadora a vantagem do Aglets: a versão AgentSpace é 350% a 790% mais lenta.

Os resultados relativos à Situação 2 mantêm, embora de forma menos acentuada, a tendência anteriormente referida. Em operações de navegação (versão `ClientMov`) a versão Aglets é 20% a 60% mais lenta. Por outro lado, na interacção exclusiva por mensagens (versão `ClientMsg`) a versão AgentSpace é 2% a 19% mais lenta, o que já não é muito significativo.

Os resultados obtidos explicam-se tendo em conta as seguintes particularidades:

- As operações de navegação são mais eficientes no AgentSpace pelo facto de tais operações (ao nível da mobilidade de objectos) serem optimizadas pelo ORB Java utilizado (Voyager). Em particular, o Voyager optimiza as operações de mobilidade de objectos que se executam dentro da mesma máquina.

- As operações de troca de mensagens são mais eficientes no Aglets pelo facto de, neste sistema, ser criado uma actividade (*thread*) por cada nova mensagem recebida. De tal forma que um agente-Aglet pode tratar várias mensagens concorrentemente. Por outro lado, na versão corrente do AgentSpace tal funcionalidade não existe – ou seja, o tratamento de cada mensagem (o método `handleMessage`) é executado na actividade corrente do agente. (A próxima versão do AgentSpace já suportará tratamento de mensagens *multithread*.)

Por fim é importante referir que a versão `ClientMsg` é sempre, em ambas as situações 1 e 2, mais rápida que a versão `ClientMov`. Tal diferença, entre os melhores tempos das duas versões, é significativa, variando entre 1176% (na Situação 1, com 50 livreiros) e 556% (Situação 2, 25 livreiros). Tal facto deve-se ao desenho da própria aplicação – que de facto não justificava a existência de operações de mobilidade –, bem como às características da rede de suporte – rede local. (De toda a forma, não é objectivo, com estes testes, validar a importância da mobilidade dos agentes de software. Esta questão deverá ser tema para uma outra comunicação).

4.3 Análise do Modelo de Objectos

Apesar de algumas semelhanças entre ambos os sistemas AgentSpace e Aglets, os seus respectivos modelos de objectos e API correspondentes são distintos. Contudo, como se pode constatar da análise do respectivo código, a implementação da aplicação CELIA foi concebida de forma que o código fosse tão semelhante quanto possível – designadamente para que fossem credíveis os resultados comparativos de desempenho.

No entanto há a realçar as seguintes diferenças, que em geral correspondem a benefícios do AgentSpace relativamente ao Aglets, segundo a perspectiva do programador:

- **Manutenção de canais abertos**: O Aglets não possui esta característica, ou seja o facto de um agente para comunicar com outro agente ter de conhecer, para além da sua identificação, também a sua localização corrente. Como consequência desta particularidade, o exemplo CELIA só funciona no Aglets, ao admitir-se certos pressupostos, tal como o facto do agente mediador não se mover.

  Benefício para o programador: **Transparência de localidade, abstracção, simplicidade.**
- **Obtenção de uma referência através da identificação do agente**: No AgentSpace é possível obter uma referência (`AgentView`) para um agente a partir da identificação do agente denominada `AgentID`. Em Aglets para se obter tal referência (`AgletProxy`), além de se fornecer a identificação do agente (designada por `AgletID`), é também necessário fornecer um URL, que corresponde ao local (`AgletContext`) onde o agente se encontra. (A classe `AgentInfo`, definida para a versão Aglets, foi necessária devido a este motivo.) Adicionalmente esta referência no AgentSpace é mais poderosa que a homóloga no Aglets, já que esconde a localização do agente e providencia de facto uma política de controlo de acessos definida de forma flexível e atribuída dinamicamente.

  Benefício para o programador: **Simplicidade**, **transparência**

- **Possibilidade de especificar o método a executar após uma operação de navegação**: Contrariamente ao AgentSpace onde é possível a especificação do método que vai ser executado após a conclusão de uma operação de navegação, no Aglets é sempre executada a *callback* `onArrival`. (Devido a esta característica foi necessário introduzir na versão Aglets do `ClientMov`, um campo que mantém o estado do cliente. Assim no `onArrival` invocamos o método desejado com base no teste ao estado.)
  Benefício para o programador: **Elegância, simplicidade, manutenção**.
- **Envio de mensagens com objectos arbitrários**: No AgentSpace é possível o envio de mensagens contendo objectos de tipos arbitrários, em particular de tipos/classes inexistentes na máquina onde o receptor da mensagem se encontra (caso tais tipos não se encontrem no *ClassPath* do receptor são carregados remotamente a partir do *ClassPath* do emissor). Tal funcionalidade não existe no Aglets, pelo que os tipos de objectos passados numa mensagem têm de se encontrar previamente instalados nas máquinas envolvidas.
  Benefício para o programador: **Abstracção, manutenção**.

Adicionalmente o AgentSpace apresenta um modelo de objectos mais completo, nomeadamente providencia: (1) uma melhor organização/estruturação de recursos, através da noção de locais de execução; (2) a gestão integrada de utilizadores, grupos de utilizadores e permissões; (3) mecanismos de segurança flexíveis permitindo a associação dinâmica de gestores de segurança a locais e a agentes; (4) a afectação transparente de um utilizador e de uma política de segurança no momento de criação de qualquer agente; (5) uma adequada integração nas aplicações da Internet/Web, ao permitir a interacção entre applets e agentes.

5 Conclusões

Argumentou-se nesta comunicação que o paradigma de agentes de software é adequado ao desenvolvimento das futuras aplicações para a Internet, em particular adequado para aplicações dinâmicas, distribuídas e potencialmente com um número elevado de utilizadores.

Esta tese foi defendida através de um caso de estudo simples (para ser facilmente discutido no espaço disponível) mas típico: a aplicação CELIA – Comércio Electrónico de Livros na Internet.

De modo a validar a tese foram desenvolvidos protótipos da aplicação CELIA segundo uma abordagem baseada em agentes, e suportados por duas infraestruturas de agentes Java (1) o conhecido sistema da IBM – Aglets –; e (2) o sistema desenvolvido pelos autores – AgentSpace.

Foram realizados testes de desempenho relativamente aos protótipos desenvolvidos sobre as duas infraestruturas, tendo-se concluindo que em termos médios e em situações normais – i.e., em que os agentes são executados em diferentes computadores –, os desempenhos obtidos são basicamente equivalentes.

Por outro lado, segundo a análise (mais subjectiva) da interface de programação e do modelo de objectos providenciados, conclui-se que o AgentSpace apresenta, comparativamente com o Aglets, um sistema mais completo e mais fácil de utilizar/desenvolver aplicações, conforme referido na Secção 4.3.

Uma das razões do bom desempenho do AgentSpace é necessariamente devido ao facto de se ter utilizado o sistema Voyager como sistema de suporte de comunicações, em vez do equivalente da Sun (o RMI), ou até mesmo a utilização de sockets. Adicionalmente utilizou-se as capacidades de persistência do Voyager de modo a garantir-se a persistência dos objectos manipulados pelo AgentSpace – em particular, dos agentes.

A versão corrente do AgentSpace encontra-se disponível para utilização não comercial em `http://berlin.inesc.pt/~agentspc/`.

Entre outras aplicações, o AgentSpace está a ser utilizado na implementação de uma ferramenta genérica de gestão de agentes com interface gráfica (o Cliente-AS); e no desenvolvimento de um protótipo de sistema de pagamentos electrónicos baseados no SET. No futuro próximo utilizar-se-á o AgentSpace no projecto COSMOS [COS97] que tem como objectivo a construção de uma infraestrutura para negociação de contratos na Internet.

Agradecimentos

Agradecemos ao Prof. Miguel Mira da Silva pelo seu apoio e motivação no desenvolvimento do AgentSpace. Agradecemos também aos alunos João Furtado e Adilio Soares pela implementação dos protótipos da aplicação CELIA e dos respectivos simuladores de desempenho.

Referências

[And96] Andersen Consulting. Bargain Finder. 1996.
`http://bf.cstar.ac.com/bf/`

[Ban94] M. Bangemann. *Recomendations to the European Council – Europe and the Global Information Society* (the Bangemann Report), The High-Level Group on the Information Society, Maio 1994.
`http://www.earn.net/EC/report.html`

[BTV96] J. Baumann, C. Tschudin, J. Vitek (editores). *Proceedings of the 2nd ECOOP Workshop on Mobile Object Systems. (*Linz, Austria*)* Dpunkt, 1996.

[COS97] Ponton, COGEFO/CEFRIEL, Hamburg University, INESC, Interzone Music Publishing, Oracle UK, and SIA. *COSMOS – Common Open Service Market for SMEs, ESPRIT Research Project Proposal*, 1997. (Início em Junho 1998).

[GK94] M. Genesereth, S. Ketchpel. Software Agents. Em [*Rie94*].

[IBM97] IBM Tokyo Research Laboratory. The Aglets Workbench: Programming Mobile Agents in Java, 1997.
`http://www.ibm.co.jp/trl/aglets`

[Mae94] P. Maes. Agents that Reduce Work and Information Overload. Em [*Rie94*].

[Mis97] Missão para a Sociedade de Informação. *Livro Verde para a Sociedade de Informação em Portugal*. Abril 1997.
`http://www.missao-si.mct.pt/livroverde/`

[NRC94] National Research Council. *Realizing the Information Future – The Internet and Beyond*. 1994.
`http://www.nap.edu/nap/online/rtif/`

[Obj97] ObjectSpace. *Voyager Core Package Technical Overview*. 1997.

[Rat97] Rational Software Corp. *UML – Unified Modeling Language, version 1.0*. 1997.
`http://www.rational.com/uml`

[Rie94] D. Riecken (editor). Tema especial: Intelligent Agents. *Communications of the ACM*, 37(7), Julho 1994.

[SMD97a] A. Silva, M. Mira da Silva, J. Delgado A Survey of Web Information Systems. In *Proceedings of the WebNet'97 – World Conference of the WWW, Internet & Intranet* (Toronto, Canada) 1997.

[SMD97b] A. Silva, M. Mira da Silva, J. Delgado. Improving Current Agent Support Systems: Focus on the Agent Execution System. *Mobile Agents Workshop of the OOPSLA'97,* (Atlanta, USA) 1997.

[SMD98] A. Silva, M. Mira da Silva, J. Delgado. AgentSpace: An Implementation of a Next-Generation Mobile Agent System. *Proceedings of the Mobile Agents'98*. (Stuttgart, Germany), 1998.

[SD98] A. Silva, J. Delgado The Agent Pattern: A Design Pattern for Dynamic and Distributed Applications. *Proceedings of the EuroPLoP'98*. (Irsee, Germany), 1998.

[WJ95] M. Wooldridge, N. Jennings. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2), Cambridge University Press, 1995.