Um Arcabouço para Assistentes de Software Virtuais em Ambientes Colaborativos

Saulo Popov Zambiasi, Ricardo José Rabelo

Depto de Automação e Sistemas Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 88.040-900 Florianópolis SC Brasil
popov@gsigma.ufsc.br, rabelo@das.ufsc.br

Abstract. *Due to the market competitiveness, enterprises need continuously to adapt to new technologies. Humans inserted in this scenario also need resources to fulfill their tasks and process in time, as well as to be agile and efficient. This paper presents a model of virtual software assistants to assist humans in some tasks, on collaborative environment scenario and ubiquitous computing environment, which can then make some activities automatically, without or with a low degree of human intervention or supervision.*

Resumo. *As empresas necessitam continuamente de novas tecnologias para poderem se adaptar a um mercado cada vez mais competitivo. As pessoas inseridas neste cenário também precisam de recursos para que possam completar suas tarefas e processos no tempo necessário, serem mais ágeis e eficientes. Este artigo apresenta um modelo de assistentes de software virtuais para auxiliar as pessoas em algumas tarefas, em um cenário de Ambientes Colaborativos e redes ubíquas, que podem executar tarefas automaticamente, sem ou com pouca intervenção ou supervisão humana.*

1. Introdução

As empresas inseridas em um ambiente colaborativo necessitam de mais agilidade e eficiência na forma como trabalham. A prática tem mostrado que essas mudanças, se por um lado têm automatizado muitos processos e integrado os sistemas das empresas, por outro as pessoas inseridas neste cenário têm tido que ser mais produtivas, acarretando um aumento de suas tarefas e responsabilidades. Elas têm estado cada vez mais ocupadas e mergulhadas em diversos problemas, muitas vezes não conseguindo cumprir as tarefas em tempo hábil e com a qualidade esperada.

Ao observar as rotinas diárias das pessoas nas empresas, nota-se que muitas das tarefas são burocráticas, repetitivas e enfadonhas. Porém, as pessoas são o mais nobre recurso para as empresas, e sua participação deveria ser direcionada para a realização de atividades tão essenciais hoje em dia quanto a operação dos processos em si, ou seja, para atividades que agregam maior valor: a realização de negócios, o exercício da criatividade para inovação, e a aprendizagem/aperfeiçoamento.

O enfoque principal da proposta deste trabalho é criar uma solução para automatizar algumas atividades, sem ou com um baixo nível de intervenção humana, deixando as pessoas com mais tempo livre para atividades que agregam mais valor.

Dessa forma, este artigo apresenta, de forma preliminar, um modelo de um Assistente de Software Virtual (ASV) para auxiliar as pessoas em determinadas tarefas. Em geral, considerando o cenário de ambientes colaborativos, seis classes de tarefas podem ser potencialmente identificadas para o início do gerenciamento de um ASV: 1) substituição do usuário em determinadas tarefas (ex. leitura e resposta de alguns e-mails ou mensagens importantes; preenchimento de alguns formulários); 2) gerenciamento da agenda e atividades do usuário; 3) procura de informações em repositórios; 4) auxílio ao usuário a compreender que tarefas precisam ser executadas e como; 5) execução de algumas tarefas do usuário (ex. parte de uma negociação eletrônica, monitoramento dos processos de negócio); e 6) filtro e gerenciamento das informações provindas de redes ubíquas (ex. filtro de conteúdo, adaptação da linguagem e formatação das informações no dispositivo móvel do usuário).

A arquitetura do ASV concebido como um agente de software e a implementação de um protótipo para o caso de "leitura/resposta de mensagens de e-mail" são apresentadas de forma preliminar com a utilização de um conjunto de ferramentas abertas relacionadas à Arquitetura Orientada a Serviços.

O presente artigo é organizado da seguinte forma: Na seção 2 é apresentado um resumo do nível atual da pesquisa. A seção 3 apresenta uma visão geral do cenário onde o agente deve atuar, assim como a um resumo das tecnologias da informação e comunicação envolvidas. A seção 4 descreve o modelo conceitual da proposta, apresentando os componentes relevantes da arquitetura do ASV, juntamente com uma análise preliminar das potenciais tecnologias para a implementação deste. Na seção 5 é apresentada a primeira versão de um protótipo baseado no modelo conceitual apresentado na seção 4. Na seção 6 são apresentadas as conclusões.

2. Breve Revisão das Abordagens

Atualmente existe uma quantidade considerável de soluções para resolver um ou mais dos problemas envolvidos no contexto deste trabalho. Porém, alguns desses problemas, estudados por anos, têm se mostrado bastante desafiadores, além do fato de que a complexidade para se representar um ser humano em variadas situações tem se mostrado uma tarefa não muito simples.

O enfoque da proposta de solução deste trabalho tem o intuito de criar um ambiente baseado em serviços web, com as seguintes características:

- O ASV deve poder ser acessado por demanda;
- Todas as classes e tarefas do ASV devem ter suas funcionalidades integradas em um ambiente intuitivo ao usuário de fácil configuração;
- As tarefas do ASV devem ser baseadas em padrões abertos para permitir agregar novas funcionalidades;
- O ASV deve ser fortemente baseado em padrões como forma de resolver problemas de interoperabilidade;
- Cada comportamento do ASV deve ser associado a uma determinada funcionalidade de forma dinâmica, de acordo com as preferências, configurações, negócios e contextos geográficos dos usuários;

- Não importando o escopo do ambiente, os ASVs devem ser vistos como serviços que podem ser acessados via Internet.

Do ponto de vista da implementação, não é intenção desenvolver todo o sistema, mas sim utilizar soluções padrões já existentes e abertas. Quando necessário, podem ser feitas algumas adaptações e, se for o caso, implementar algumas partes do sistema.

De fato, existem esforços que atacam alguns pontos do cenário deste trabalho, tal como a figura dos Assistentes. Porém, estes dão apenas uma ajuda limitada, com relação à problemática deste trabalho, uma vez que apenas prestam algum auxílio para tirar dúvidas sobre como uma dada ação deve ser feita. Alguns tipos de assistentes são programados para buscar certas informações em certos sites da Internet ou bancos de dados. Contudo, ainda não resolvem o problema, senão apenas prestam alguma ajuda para se ter certas informações disponíveis e se executar a tarefa com mais eficácia. Todavia, é a pessoa quem tem que executar a tarefa. Já as ferramentas de Trabalho em Grupo (*Groupware*) oferecem apenas um ambiente mais adequado para que as pessoas colaborem entre si.

Numa direção mais avançada e relacionada com a problemática de base tratada neste artigo, surge a abordagem dos sistemas multiagentes [Weiss, 1999]. Utilizando esta abordagem, por exemplo, agentes de negociação eletrônica podem realizar parte das trocas de informação com outros agentes automaticamente, sem intervenção humana, como apresentado em Weiss (1999). Ainda assim, neste caso, os agentes tratam apenas do processo de negociação e em cenários de negócios rigorosamente definidos.

Em Franco (2007) é apresentada uma arquitetura baseada em serviços web e em um cenário de dispositivos móveis, com o conceito de Entidades Ativas. Este conceito trabalha com processos de negócios colaborativos em uma rede de dispositivos móveis para suportar a colaboração entre atores em um cenário de eventos extremos, tais como desastres naturais como terremotos, furações e enchentes, ou mesmo desastres acidentais ou intencionais como incêndio e ataques terroristas. Porém, este trabalho apenas enfoca a colaboração entre pessoas envolvidas nestes cenários, e não as substitui em certas situações.

Um cenário de assistentes pessoais é apresentado em Huhns (1998), para representar indivíduos na Internet, ajudando-os nas atividades diárias como e-mails e chamadas de telefone celular. Estes assistentes pessoais podem coordenar e negociar com diversos tipos de autonomia, baseados em informações definidas pelos usuários. Este cenário é visto sobre um ambiente de computação ubíqua e verifica o contexto de cada situação ou local. Finalmente, este responde e interage com seus usuários por meio de um dispositivo móvel. Este cenário possui diversas idéias correlatas com o presente trabalho, todavia é apresentado apenas no nível conceitual, e não nenhum modelo, propriamente dito, é apresentado.

Em outro sentido, os dispositivos móveis possuem capacidades limitadas de informações, e comunicação, necessárias para filtrar, analisar e trocar informações entre o ambiente. Para resolver este problema, Carrilho-Ramos (2007) define um arcabouço baseado em sistemas multiagentes que provê adaptações das informações dos usuários conforme suas preferências e histórico no sistema, de acordo com as capacidades limitadas dos seus dispositivos móveis. O trabalho de Carrilho-Ramos descreve o

gerenciamento do conhecimento e a troca de informações e a adaptação do seu arcabouço para suportar capacidades de adaptação [Carrilho-Ramos, 2007].

No presente trabalho, o que se deseja é ir além das soluções ora propostas, não só no sentido de se inserir situações empresariais concretas e práticas, como também de se criar um arcabouço aberto e genérico, onde novas situações possam ser adicionadas ao longo do tempo. É importante ressaltar novamente que não se deseja substituir o ser humano. Deseja-se essencialmente diminuir o seu excesso de trabalho, potencializando que se dedique majoritariamente a tarefas mais nobres caso o usuário assim o deseje.

3. Assistentes de Software Virtuais

No cenário deste trabalho, o Assistente de Software Virtual pode ser definido como um processo virtual (módulo de software) que representa uma pessoa em um processo, executando suas atividades essenciais via um conjunto de serviços, com alguma segurança, restrições de ações e adaptação de forma efetiva às condições do ambiente. Mais especificamente, o intuito deste trabalho é desenvolver um ASV que trabalhe em seis categorias de ações: 1) Substituir o usuário em algumas tarefas; 2) Gerenciar as atividades e agenda do usuário; 3) Pesquisar informações em repositórios; 4) Auxiliar o usuário no entendimento do que e como precisa ser feito; 5) Executar algumas tarefas pelo usuário e; 6) Gerenciar as informações provindas de redes ubíquas.

Estas ações podem ser feitas em três níveis: autonomia total (o ASV decide e age sozinho), autonomia parcial (o ASV decide e age com uma possível intervenção ou aprovação do usuário, ou algum tipo de sistema) ou sem autonomia (ele decide e age de acordo com intervenção/aprovação).

O ASV precisa ser capaz de se comunicar com outros agentes de através de uma plataforma interoperável, enviando e requisitando informações, processos, atividades e serviços. Dessa forma, os usuários do ASV podem colaborar com outros ASVs ou mesmo os ASVs podem colaborar entre si na execução de alguns tipos de tarefas. Isto significa que os ASVs precisam saber o que se deve fazer, como, quando, com quem deve se comunicar e interagir, como se adaptar, que recursos usar e o quanto usar.

A comunicação entre os ASVs e os sistemas empresariais pode acontecer por meio de serviços, em que um cliente faz uma requisição e o serviço envia uma resposta. Estes serviços devem ser padronizados para manter a interoperabilidade do ambiente.

De forma a permitir que um usuário de um ASV possa ser informado de todas as tarefas em execução, dos resultados já encontrados, e outras informações, é necessário que essas possam ser acessadas de qualquer lugar e a qualquer hora. Para tornar esse cenário possível é necessário que os agentes possam funcionar em um ambiente integrado com trocas de informações sem a necessidade de cabos, assim como o usuário pode trafegar pelo ambiente recebendo e enviando informações para seu ASV de qualquer lugar e a qualquer momento.

Neste contexto, a tecnologia da computação ubíqua se adéqua a essa necessidades do projeto. Nesta, os dispositivos computacionais distribuídos em uma organização podem trocar informações entre eles. Da mesma forma, sistemas integrados podem enviar informações contextuais de diversas fontes e com diferentes tipos de estruturas [Abowd 1997]. Dessa forma, os agentes podem apresentar informações aos usuários caso essas sejam consideradas relevantes. Isto permite que os usuários

executem tarefas, auxiliados por seus ASVs, sem necessariamente estarem presentes na organização ou empresa. Não obstante é necessário que o ASV possa se adequar, em tempo de execução, a diversos formatos e protocolos que os usuários encontram em seus ambientes computacionais. Este deve considerar o contexto (consciência do contexto) para poder filtrar informações não essenciais, restrições de acesso de segurança e tipos de usuários. [Henricksen 2006].

De fato, a arquitetura do ASV e o ambiente envolvido (computação ubíqua, consciência do contexto e dispositivos móveis) propostos neste trabalho são um arcabouço integrado, composto por um ou mais agentes que auxiliam as pessoas em suas atividades.

4. Proposta do Modelo do Assistente de Software Virtual

Algumas características são necessárias para que o ASV possa ser executado com certa autonomia e independência. Essas compõem o modelo do ASV, composto por sistemas e módulos que o permite completar seu ciclo de vida, como mostrados na Figura 1.

O ambiente onde o software do ASV é executado precisa estar constantemente disponível para que o mesmo possa executar suas operações, mesmo se o usuário não estiver conectado ou não estiver disponível. No mesmo sentido, o ASV precisa enviar informações ou requisições ao usuário quando for necessário, de forma a permitir a execução dos seus processos.

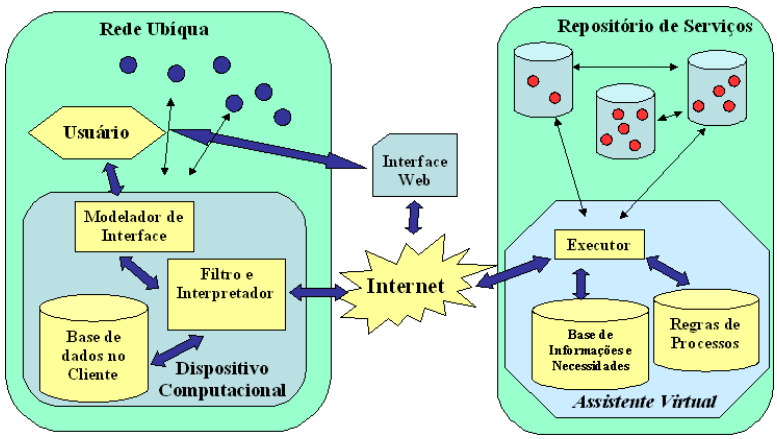

Figura 1 Arquitetura Prévia do Modelo de Representação do ASV.

A seguir, são apresentadas as descrições dos módulos básicos para a execução do Assistente Virtual:

- **Base de Informações e Necessidades:** Possui informações relevantes, tais como as preferências do usuário e informações necessárias para a execução dos processos. Estas informações precisam estar armazenadas em um Banco de Dados e disponíveis quando o ASV necessitar.
- **Regras de Processos:** Regras para a execução de cada situação específica (comportamentos). Devem ser modulares e configuradas nos ASVs na forma de *plug-ins*, podendo ser adquiridas ou desenvolvidas pelo próprio usuário do ASV. Estes comportamentos podem ser organizados e conectados, criando-se

comportamentos mais complexos (orquestração) a partir dos comportamentos mais simples.

- **Executor:** Organiza os processos e comportamentos, assim como a prioridade de cada, seleciona os comportamentos para cada situação que surge, faz a interface com os dispositivos computacionais do usuário e com os outros sistemas ou ASVs. Este módulo é o núcleo operacional do Assistente de Software Virtual.
- **Base de Informações no Cliente:** Informações necessárias para filtrar a grande quantidade de informações provenientes de dispositivos de redes ubíquas em conformidade com os interesses, necessidades, escolhas e objetivos do usuário. Essa base de informações precisa ser armazenada no dispositivo móvel do usuário e devem ser sincronizas com a base de dados do ASV quando conectado na Internet.
- **Filtro e Interpretador:** Interpretador que, do lado do cliente, recebe as informações providas do ASV, redes ubíquas e entradas do usuário. Este interpretador compila essas informações para modelar os processos do ASV.
- **Modelador de Interface:** Ajusta a interface do usuário (no dispositivo móvel) conforme cada diferente tipo de processo. As informações para tal são providas pelo ASV e baseado no conjunto das informações do processo e preferências do usuário.
- **Interface Web:** Interface para criação e configuração dos ASVs, e seus processos, e preferências dos usuários de forma segura e ágil, ainda com a possibilidade do usuário poder acessá-la de qualquer lugar e a qualquer momento, por meio de uma interface web. Isso provê o acesso a opções e operações que nem sempre são possíveis com o uso de dispositivos móveis, devido a suas limitações visuais.

4.1. Tecnologias da Informação Envolvidas na Implementação

Para o presente projeto foi realizado um estudo prévio de tecnologias da informação e comunicação necessárias para a criação e execução do ASV na web, conforme citadas a seguir.

A tecnologia dos serviços web é amplamente utilizada nos dias de hoje, já com um bom nível de maturidade e padronização. De fato, soluções distribuídas dessa tecnologia podem prover a padronização necessária para manter a compatibilidade entre serviços em uma arquitetura orientada a serviços. Os serviços web são compostos de pequenos módulos de softwares, acessados por outras aplicações por meio de um conjunto de protocolos web, por uma URL e baseado em um conjunto de padrões já consolidados. Através de um estudo prévio, podem-se citar alguns desses padrões tais como a linguagem XML (*Extended Markup Language*), uma interface para descrever os serviços - WSDL (*Web Services Description Language*), o protocolo SOAP (*Simple Object Access Protocol*) e a UDDI (*Universal Description, Discovery and Integration*). Através do uso desses padrões, é possível desenvolver aplicações modulares, abertas, baseadas na Internet e com interfaces padronizadas [Alonso 2004].

Os processos executados pelo ASV, e a comunicação destes com outros serviços, necessitam de uma padronização. A Linguagem de Execução de Processos de

Negócios (*Business Process Execution Language BPEL*) pode prover uma especificação formal para a execução e interação do ASV, fornecendo uma fácil interação entre organizações, assim como cenários *business-to-business* (B2B) [Akenhurst 2004]. Mais especificamente no nível de implementação, WS-BPEL é uma especificação atual para a interação entre serviços web e estruturada com documentos XML [IBM 2003].

A especificação BPEL define variáveis, interações como laços, condições e paralelismos. Isto permite aos desenvolvedores criarem processos de negócios que interagem com serviços web de organizações, assim como serviços web externos, de outras organizações. As primitivas do BPEL, e suas exceções são interpretadas e executadas por um mecanismo chamado orquestração [Michael 2004]. Por meio da orquestração, o comportamento de um ASV pode ser baseado na composição de diversos serviços web disponível na própria organização, em outras organizações ou na Internet, como um todo. Estes serviços web podem compor um comportamento de forma algorítmica, com estruturas de laços, serviços que executam em paralelo e condicionais.

A tecnologia de Agentes surge como uma das tecnologias mais apropriadas para dar suporte a objetos inteligente , como ASV. Primeiramente, a tecnologia de agentes foi desenvolvida em ambientes próprios, linguagens próprias e protocolos de comunicação específicos. Porém, a tecnologia dos agentes tem migrado para o uso de tecnologias padronizadas como o XML e serviços web. Dessa forma, o comportamento dos agentes pode ser modelado como serviços, interagindo com serviços web e redes *peer-to-peer*. Ainda nesse sentido, já existem muitas plataformas de sistemas multiagentes das quais muitas delas já com um bom nível de maturidade e robustez [Weiss, 1999].

5. Protótipo

Neste ponto é apresentada uma primeira versão do protótipo do ASV, que faz a gerência de uma conta de e-mails dos usuários, apresentada como uma das instâncias da primeira macro ação mencionada na seção 3. Esta é apresentada como um módulo da forma menos intrusiva possível (sem a necessidade de instalar qualquer software adicional no desktop do usuário), definindo-se como um requisito do projeto. Este processo é derivado da arquitetura genérica, apresentada na Figura 1, composta por: uma interface web, para criar e configurar os ASVs (linguagem PHP); base de dados MySQL para armazenar as informações dos ASVs; comportamentos em BPEL (Apache ODE); serviços web para o acesso aos serviços (Apache Axis) e a implementação do agente em linguagem de programação Java.

Nesta situação, o ASV verifica as informações do usuário, por exemplo, a situação no Gtalk (*Available, unavailable, away*), e verifica quando uma nova mensagem chega à conta de e-mail do usuário. Após isso, o ASV faz a gerência dessas mensagens por meio de um conjunto de regras previamente definidas. Estas interfaces são desenvolvidas com serviços web para acessar as funcionalidades do Gtalk, servidor de e-mails e agenda. Isto permite a invocação destes serviços por meio de documentos BPEL. A Figura 2 apresenta, de forma geral, a execução dos passos neste caso:

1. O usuário configura suas preferências no ASV.

2. Uma nova tarefa ou mensagem chega.
3. Quando a tarefa ou mensagem é detectada, o ASV verifica em sua base de dados os critérios para a ativação e o grau de autonomia para a execução deste comportamento.
4. O ASV elabora um planejamento de execução em conformidade com uma regra ou conjunto de negócios para a situação.
5. Finalmente, o *ASV* executa o comportamento.

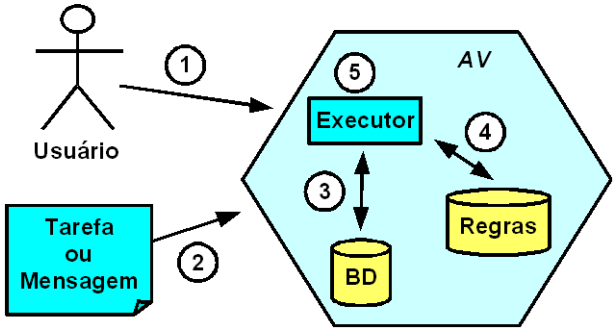

Figura 2: Gerenciamento da conta de e-mails do usuário.

Na Figura 3 é apresentado o diagrama de seqüência para a execução do gerenciamento de e-mails do usuário. Neste, quando o Executor do ASV é informado de que o usuário não está disponível no Gtalk, este lê as informações da base de dados do usuário para selecionar critérios de ativação de determinado comportamento. Após isso, é requisitada uma confirmação do usuário, ou não, conforme o nível de autonomia desta tarefa. Então é feita a verificação na conta de email do usuário conforme a chegada de novas mensagens, estas são interpretada e executada em conformidade com as preferências do usuário, armazenadas na base de dados do VA.

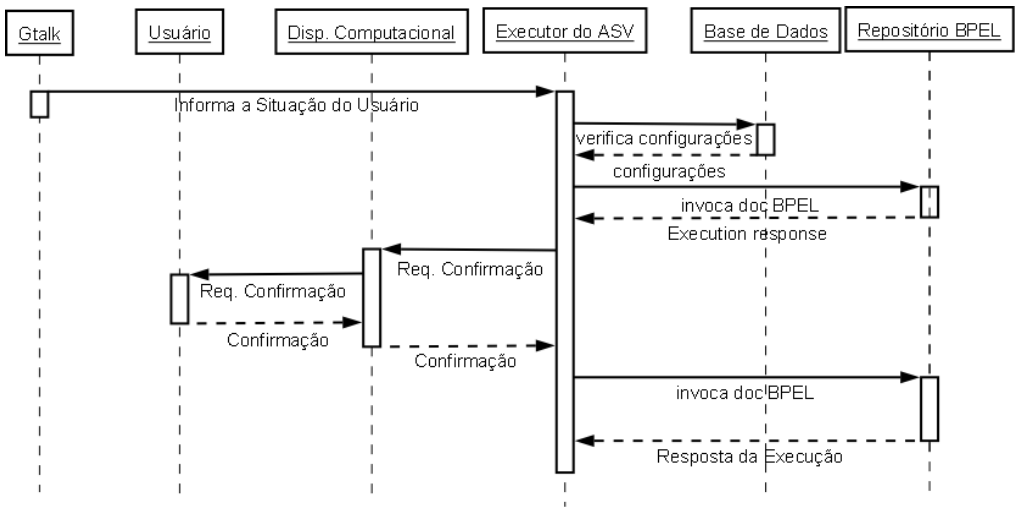

Figura 3. Diagrama de sequência para o gerenciamento de e-emails.

O ASV utiliza o Apache ODE como motor para a execução dos documentos BPEL. Estes documentos descrevem o comportamento deste agente com variáveis, estruturas,

condicionais, assim como acessam funções na forma de serviços (Figura 4). Neste primeiro exemplo, o ASV pode identificar a situação do usuário no GTalk e algumas palavras especiais no assunto das mensagens, por exemplo, URGENTE, TAREFA, TRABALHO, FAZER. Desta forma, o ASV responde aos e-mails ou executa essas atividades em conformidade com as informações registradas na base de dados do usuário. Na Figura 4, o ASV invoca a operação "*getPresence*" do Gtalk (serviço web WsJabber) e a operação "*open*" da conta de e-emails (serviço web WsMailMonitor). Caso seja verificado que o usuário está "*unavailable*" no Gtalk, o ASV inicia o procedimento para o gerenciamento de novas mensagens nesta conta de e-mails.

```
<bpws:invoke inputVariable="WsJabberRequest"
        name="connect to jabber" operation="getPresence"
        outputVariable="WsJabberResponse" partnerLink="WsJabber"
        portType="ns0:WsJabber"/>
<bpws:invoke inputVariable="WsMailMonitorRequest"
        name="connect to mail account" operation="open"
        outputVariable="WsMailMonitorResponse"
        partnerLink="WsMailMonitor" portType="ns1:WsMailMonitor"/>
<bpws:while name="Jabber user unASVailable">
        <bpws:condition>
        <![CDATA[$WsJabberResponse.openReturn = 'unASVailable']]>
        </bpws:condition>
        <bpws:sequence name="Sequence">
        <bpws:invoke inputVariable="WsMailMonitorRequest"
        name="verify emails" operation="hasMessage"
        outputVariable="WsMailMonitorResponse"
        partnerLink="WsMailMonitor"
        portType="ns1:WsMailMonitor"/>
        <bpws:if name="new email">
```

Figura 4. Parte de um documento BPEL para o gerenciamento dos e-mails.

6. Considerações e Próximos Passos

Este artigo apresentou uma proposta de um arcabouço para Assistentes de Software Virtuais em um cenário de computação ubíqua. Os ASVs servem para ajudar a minimizar o tempo gasto na execução de tarefas repetitivas e que não agregam valor, ou quando os usuários não estão disponíveis ou estão ocupados. Desta forma, as pessoas podem direcionar seu tempo para atividades mais importantes como novos negócios, melhoramento de suas tarefas, inovações e auto-aprendizado. Este trabalho também tenta garantir que os usuários completem suas tarefas no tempo determinado e com mais qualidade.

Foi apresentado um modelo representativo de um Assistente de Software Virtual na execução de alguns tipos de tarefas. A arquitetura foi mostrada em um nível conceitual com a indicação dos módulos relevantes para a execução do ciclo de vida dos processos dos ASVs. Foram apresentadas uma análise preliminar e potenciais tecnologias para um primeiro protótipo. Este protótipo se encontra atualmente parcialmente implementado para um caso de processo de negócios. Neste, o agente faz a gerência dos e-mails do usuário quando este não está disponível.

A execução do protótipo se apresentou de maneira satisfatória para o que foi proposto no caso específico do gerenciamento de emails. Quando o usuário tornava-se indisponível no seu GTalk, o agente assumia a gerência de seus emails. Mesmo que com uma interpretação bastante simples, e com poucas palavras-chave, pôde responder determinados emails e agendar algumas tarefas para o usuário. O acréscimo de novas regras e informações na base de conhecimento deste comportamento deve refinar como

ele age com uma maior quantidade de emails. Contudo, a principal motivação do trabalho é gerenciar não apenas uma tarefa, mas diversas tarefas, de forma dinâmica, detectando as informações que chegam e encontrando o comportamento que mais se adapta a cada situação. Além disso, o arcabouço deve permitir que o usuário agregue novos comportamentos dinamicamente.

Os próximos passos do trabalho incluem um estudo mais aprofundado do modelo conceitual e um maior refinamento da arquitetura genérica, apresentando de forma mais detalhada o uso dos seis tipos macro ações que precisam ser tratadas, assim como a análise das tecnologias e padrões para a implementação final.

Agradecimentos

Este trabalho foi parcialmente financiado e desenvolvido no escopo do projeto brasileiro IFM-II (http://www.ifm.org.br) e do projeto europeu IST IP ECOLEAD (http://www.ecolead.org).

Referências

Abowd, G. D. et al. (1997) Cyberguide: A Mobile Context-Aware Tour Guide . Vol. 3, pages 421-433. Wireless Network.

Akehurst, D. H. (2004) Validating BPEL Specifications Using OCL . Computing Laboratory. University of Kent. Cantebury.

Alonso, G. At al. (2004) Web Services: Concepts: Architecture and Applications . Springer Verlag NY, 1th edition.

Carrillo-Ramos, A., et all. (2007) Knowledge Management for Adapted Information Retrieval in Ubiquitous Environments , Springer-Verlag Berlin Heidelber, WEBIST 2005/2006, pages 84 96.

Franco, R.D., Neyem, A, et al. (2007) Supporting Mobile Virtual Team s Coordination With Soa-Based Active Entities: Establishing The Foundation of Collaborative Networks , IFIP TC 5 Working Group 5.5 Eighth IFIP Working Conference on Virtual Enterprises, September 10-12, 2007, Guimarães, Portugal. IFIP 243 Springer.

Henricksen, K., Indulska, J. (2006) Developing Context-Aware Pervasive Computing Applications: Model and Approach: Pervasive and Mobile Computing. pages 37-64. http://www.sciencedirect.com. (December).

Huhns, M.N., Singh, M.P. (1998) Personal Assistants . IEEE Internet Computing, Vol. 2, Issue 5, (September/October), pages 90-92.

IBM. (2007) Business Process Execution Language for Web Services . http://www-128.ibm.com/developerworks/ library/specification/ws-bpel/ (March).

Michael C. (2004) What Is BPEL And Why Is It So Important To My Business? SoftCare EC. http://www.softcare.com/whitepapers/wp_whatis_bpel.php. (December).

Russel, S., Norvig, Peter. (1995) Artificial Intelligence: A Modern Approach ; Prentice-Hall. EUA. pages 773-814.

Weiss, G. (1999) Multiagent Systems: A Modern Approach to Distributed AI . Weiss, Gerhard, MIT.