Uma Arquitetura Aberta e Orientada a Serviços para Softwares Assistentes Pessoais

Saulo Popov Zambiasi[1]

Ricardo J. Rabelo [2]

Resumo: Vários esforços vêm sendo feitos na direção de softwares assistentes pessoais (SAP) com o objetivo de ajudar as pessoas em suas atividades diárias, em casa ou no trabalho. A despeito da intrínseca complexidade que SAPs podem ter, a maior parte dos trabalhos se preocupa em desenvolver soluções apenas para tarefas bastante específicas, sem preocupações de integração e interoperação com outros sistemas (incluindo outros SAPs), além de não se conectarem com os sistemas empresariais e respectivos processos de negócios. Visando atender a esses requisitos, este trabalho apresenta uma arquitetura de referência aberta para SAPs, permitindo que instâncias interoperáveis possam ser derivadas, personalizadas, implementadas e implantadas consoantes às características das pessoas e processos das organizações. Uma instância foi gerada e seus resultados analisados.

Abstract: *Several efforts have been made towards creating personal assistant software (PAS) to help people in their daily life activities, at home or at work. In spite of the complexity PAS can have, most of the works focuses on developing solutions only for very specific tasks, without supporting integration and interoperation with other systems (including other PASs) and with enterprise systems and their respective business processes. Aiming at coping with these requirements, this work presents an open reference architecture for PASs, allowing that interoperable instances can be derived, personalized, implemented and deployed according to users' profiles and to enterprises' processes. An instance of a PAS has been derived and its results are discussed.*

1 Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina. popov@das.ufsc.br

2 Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina. rabelo@das.ufsc.br

1 Introdução

O conceito de softwares que fornecem assistência às pessoas não é novo e tem influenciado a imaginação de muitos escritores de ficção científica e cientistas da computação (Markoff, 2008). Essa ideia vem sendo representada na forma de softwares baseados em conhecimento e que funcionam como um "secretário", auxiliando nas mais diversas tarefas, como pagamento de contas, organização de viagens, gerenciamento de agendas, localização de informações em bibliotecas virtuais, etc. (Michaell et al., 1994).

A definição de Softwares Assistentes Pessoais (SAP / Personal Assistant Software) é um tanto variável na literatura, normalmente muito influenciada pelo tipo de problema que se deseja resolver, pelos seus requisitos e pela abordagem arquitetural ou de implementação. Em termos mais gerais, para Bocionek (1994) um SAP é um programa autônomo que oferece auxílio no gerenciamento das atividades dos seus usuários, inclusive no caso de atividades conflitantes. Devem saber negociar, aprender, ter portabilidade e mobilidade, principalmente no cenário da computação móvel. Para Markoff (2008), um SAP deve ter por objetivo auxiliar usuários nos trabalhos com sistemas computacionais e automatizar tarefas rotineiras, de forma a liberar as pessoas para as tarefas mais importantes. Já para Hoyle (1997), são softwares que visam auxiliar os usuários em atividades rotineiras, repetitivas, desinteressantes e que consomem muito tempo, aos quais ele chama de softbots, um conceito correlato, que muitas vezes é usado como sinônimo de SAP.

Na verdade, a linha que separa um SAP de um softbot não é muito clara na literatura. As definições a seguir são tomadas como referência para este artigo, e é nesta perspectiva que se o classifica como sendo um trabalho sobre SAP. Para Hoyle (1997), um softbot (ou "robô de software") é um software embarcado em algum dispositivo ou que é executado em um ambiente computacional, que visa automatizar uma tarefa específica para um usuário, normalmente de forma autônoma, sem ou com muito pouca intervenção deste. Para Huhns (1998), um SAP tem um escopo maior, não podendo ser visto apenas como um programa de computador personalizado. Deve ser baseado em rede, ser interativo, adaptativo, de propósito geral, de execução autônoma e deve poder interagir com outros assistentes ou sistemas.

Na literatura é comum associar o conceito de SAP com o de agentes de software. Nesta perspectiva, Weiss (1999) afirma que um SAP-agente pode substituir parcial ou totalmente uma pessoa em determinadas tarefas e situações, podendo trocar informações automaticamente por meio de negociações eletrônicas, assumindo parcial ou totalmente o papel do usuário. Russel e Norvig (2004) definem um agente como algo que pode se comunicar com outros para poder alcançar seus objetivos, perceber seu ambiente através de recursos e responder com ações e assistência ao usuário, e esse conceito também pode ser observado nos SAPs.

Apesar de ser uma área relativamente recente, diversas pesquisas sobre SAP têm sido efetuadas no sentido de contribuírem para os problemas relacionados com o seu projeto e implementação. Contudo, com base em revisão bibliográfica efetuada, observa-se que as propostas existentes focam pontos isolados do problema ou foram projetadas para atender a

objetivos (i.e. tipo de tarefas) bem específicos, sem uma visão integrada e interoperável sobre as várias fontes de informação e de atividades que um usuário normalmente está envolto. Como exemplo, pode-se citar o tipo de assistente presente em sistemas como o Microsoft Office® e equivalentes, em que o software basicamente reage ao que o usuário digita e traz um conjunto de respostas que julga fazer sentido. Posteriormente, compete ao usuário analisar o resultado e atuar em outra aplicação e ambiente para dar prosseguimento às suas atividades. Isto ocorre, entre outras razões, pelo fato deles serem fechados, dedicados a ações específicas, sem a possibilidade de adaptação - inclusive em tempo de execução - sobre o que fazer.

Já em termos do uso de SAPs em ambientes de empresa, verificou-se que SAPs não têm sido desenvolvidos para serem integrados aos processos de negócios empresariais. Isso cria pouca agilidade em processos de decisão e potencializa diversos tipos de erros de interpretação, digitação, não adequada ou útil execução de ações, etc. Na prática, faz com que o usuário final tenha que migrar de ambientes ao longo do seu trabalho e, ele, usuário, tenha que realizar as ações gerais de interoperabilidade, dentro de um dado processo e contexto de negócio. Por processos de negócios empresariais quer-se dizer ações e transações usuais em empresas, como compras, vendas, negociações, consulta de preços, resposta a cotações, distribuição, gestão da produção, etc. Apesar de ainda a maioria das empresas utilizarem suas próprias representações de processos, observa-se o uso crescente de padrões de processos negócios empresariais, tais como UBL (UBL, 2011), ebXML (ebXML, 2011) e RosettaNet (RosettaNet, 2011).

A solução para todos os problemas relacionados com os vários aspectos de um SAP adaptativo, flexível, interoperável, interagente e integrado aos ambientes empresariais é das mais complexas e, sob certas perspectivas, ainda não plenamente resolvível com o estado das tecnologias atuais e abordagens conceituais. Neste sentido, este artigo visa trazer uma contribuição conceitual e de modelo de implementação que vai na direção daqueles requisitos. Trata-se de uma pesquisa aplicada, parcialmente exploratória e qualitativa, que visa investigar a possibilidade e importância de existir uma arquitetura de referência para SAPs que possa servir de base para instanciações particulares, abertas, baseada em padrões, flexíveis e passíveis de serem integradas a ambientes e processos de negócios de empresas. A premissa básica associada a esta pesquisa é de que as empresas precisam cada vez mais tomar decisões confiáveis e de forma ágil, mas seus colaboradores estão cada vez mais imersos em vários processos de negócios, simultâneos, quer repetitivos quer complexos, e cada vez mais sem o devido tempo para geri-los. Portanto, SAPs têm o potencial de atuar como um importante mecanismo / ferramenta de trabalho para auxiliar seus usuários a executar suas atividades melhor ou mesmo automaticamente, substituindo-os em certas ações.

Concretamente, este artigo visa apresentar uma proposta de um modelo e uma arquitetura de referência para SAPs, baseados no paradigma da arquitetura orientada a serviços, e uma instância de SAP derivada destes como resultado, ou seja, um SAP que atende em boa medida àqueles requisitos supramencionados.

Este artigo é organizado da seguinte forma. A seção 1 introduz e contextualiza o problema e o trabalho de pesquisa. A seção 2 apresenta uma definição geral de Softwares Assistentes Pessoais, requisitos básicos e uma revisão da literatura sobre estes. Na seção 3 é apresentada a proposta de modelo e arquitetura de referência para SAPs. Na seção 4 há a verificação da proposta por meio de testes em uma instância implementada com base no modelo e arquitetura propostos. Na seção 5 são apresentadas as considerações finais do trabalho.

2 Softwares Assistentes

Uma vez definido o objetivo pretendido, esta seção visa buscar uma melhor definição do que é um SAP, fazer uma breve revisão da literatura sobre alguns dos esforços considerados mais relevantes no contexto deste trabalho e identificar alguns requisitos funcionais comuns às várias definições de SAP. Com isso, busca-se delinear os requisitos gerais de arquitetura de referência propriamente dita.

2.1 Definição de SAP

Como mencionado na seção anterior, existe uma grande quantidade de definições para SAP. Considerando que é muito difícil buscar uma definição que contemple as várias visões e ao mesmo tempo seja útil de ser utilizada na prática, preferiu-se descrever a definição de um SAP em termos de requisitos funcionais desejáveis, identificados por vários autores. São eles os mais comuns:

- Atuar com certa autonomia em suas tarefas, de forma que estes possam verificar a situação dos seus usuários (como por exemplo novos e-mails em sua caixa postal) e que possam responder apropriadamente a cada situação que aparece (Bocionek, 1994), (Hoyle, 1997), (Hunhs, 1998), (Schiaffino, 2006);
- Ser flexível em termos de poder atuar diante de novas situações e cenários de negócios (Hunhs, 1998), (Angehrn, 2001);
- Ser adaptável ao usuário conforme informações sobre ele, suas preferências, necessidades, e tomando em conta a evolução dessas informações em relação ao tempo (Hunhs, 1998), (Angehrn, 2001), (Schiaffino, 2006);
- Interagir com seu usuário de forma a tornar possível a assistência a ele (Hunhs, 1998), (Bush, 2006), (Schiaffino, 2006);
- Ser baseado em rede, ou seja, ter a capacidade de buscar informações via redes de computadores, tal como a Internet, e interagir com outros sistemas e assistentes pessoais de outros usuários de forma a alcançar os objetivos do seu usuário (Bocionek, 1994), (Hoyle, 1997), (Huhns, 2002);

- Ser de propósito geral, ou seja, não específico à apenas uma atividade ou um grupo de usuários (Huhns, 2002);
- Ter adaptabilidade de contexto, percebendo o ambiente e atuando conforme o contexto (Bush, 2006), (Hunhs, 1998);
- Ser integrável e interoperável aos processos de negócios da empresa (Hunhs, 1998), (Zambiasi e Rabelo, 2012).

2.2 Requisitos Gerais

Vários são os aspectos que devem ser considerados no projeto conceitual e de implementação de um SAP, que o fazem ser extremamente complexo e que devem ser atacados em sua plenitude. Pode-se analisá-los sob quatro planos ou dimensões.

Num primeiro plano, há o gerenciamento do seu ciclo de vida, basicamente composto por: i) seu projeto / configuração e lançamento no sistema; ii) sua operação (i.e. a gestão das suas ações e informações durante a execução das suas tarefas, incluindo questões de adaptação e interoperabilidade); iii) sua modificação uma vez posto em operação (devido a reconfigurações gerais que alteram seus parâmetros de ação e variáveis de estado que fazem com que passe a funcionar de outra forma); e iv) sua retirada do sistema (por decisão própria ou por desejo do usuário). Dentro de cada uma dessas fases deve-se prever um grande conjunto de funcionalidades de suporte (Zambiasi e Rabelo, 2012).

Num segundo plano, funcional, há de se considerar (Etzioni et al., 1992): i) como especificar as tarefas do e para o SAP (metas e métricas); ii) como planejar as ações de cada tarefa, como supervisionar suas execuções, e como atuar no caso de desvios; iii) como representar as informações e os conhecimentos prévios, adquiridos do SAP ao longo do seu ciclo de vida, assim como manter esta base de conhecimento coerente; iv) como aprender e se adaptar em função disso; v) como monitorar sua própria existência no sistema (por exemplo, quando do travamento de execução ou sua retirada indevida ou não autorizada); vi) como interpretar o desejo do usuário e convertê-lo em ações executáveis computacionalmente; vii) como fazer o SAP se comunicar e interoperar com os vários atores do ambiente assim como garantir uma adequada coordenação das suas ações; viii) como gerenciar sua mobilidade e clonagem.

Num terceiro plano, relacionado ao contexto de ação de um SAP (Bush, 2006): i) contexto pessoal, incluindo aspectos como preferências do usuário, históricos, e características do dispositivo; ii) contexto físico (ou de rede), considerando a representação da rede e das características de conectividade (tal como protocolos de comunicação, mecanismos de comunicação e QoS); iii) contexto de serviço, que avalia a disponibilidade de aplicações e serviços remotos; iv) contexto social, que define os diferentes tipos de papéis, regras e tarefas a serem seguidas / executadas pelo usuário em conformidade com cada contato social em cada meio e escopo de processo de negócios; v) contexto de ambiente, que gerencia as informações locais, como temperatura, tempo, previsão do tempo, localização, qualidade da rede e disponibilidade ou presença do usuário no local no momento; vi)

contexto da aplicação, que avalia a compatibilidade das aplicações a serem executadas pelo assistente pessoal; vii) contexto do dispositivo, que avalia as capacidades gerais de processamento e armazenamento do dispositivo computacional onde o SAP está no momento lançado; contexto semântico, que enquadra e define a terminologia, os seus significados e correlação de conceitos e sinônimos conforme os atores com os quais o SAP se comunica.

Finalmente, um quarto plano, que envolve os aspectos de projeto de implementação e seleção das tecnologias de informação e comunicação que devem ser usadas na implementação do SAP: i) software; ii) hardware; iii) middleware e; iv) mecanismos de segurança.

2.3 Revisão da Literatura

Na parte da implementação, tanto iniciativas privadas como iniciativas de sistemas open source têm tido influência e destaque nos vários trabalhos sobre SAPs. A seguir são citados e brevemente analisados os principais trabalhos correlatos à presente proposta.

O SAP da Shelltoys, por exemplo, é um sistema para auxílio em tarefas do usuário. Contudo, ele é proprietário e fechado, possui quase nenhuma autonomia, tem pouca flexibilidade, não está preparado para se integrar aos processos de negócio empresariais e se limita a gerenciar as tarefas do usuário via uma agenda de compromissos. Ele possui uma lista de tarefas e gera lembretes de compromissos quanto a aniversários, atividades, reuniões e outras pequenas ações (SHELLTOYS, 2011).

No mesmo sentido, da parte de gerenciamento de agenda do usuário, o projeto Sandy (Mann, 2011) foi desenvolvido para auxiliar o usuário a lembrar de assuntos (como compromissos, listas, pendências, contatos, projetos) e trabalha com troca de mensagens via SMS, Twitter e até mensagens de e-mail. Para o usuário utilizar o Sandy, basta enviar mensagens do tipo "Lembrar de pegar meu carro em 15 minutos", "O telefone do meu pai é 9999-9999", "lembrar de comprar mantimentos ovos", etc. O Sandy é também um projeto proprietário, fechado e foi descontinuado em 2008, conforme informado no próprio site da empresa. A grande vantagem do Sandy, e utilizada como inspiração para esta proposta, é a dinamicidade que o Sandy usa dos meios de comunicação utilizados para a interação entre o SAP e o usuário, por meio de SMS, Twitter, e-mail e outros, não engessando o usuário a apenas receber avisos por meios próprios da empresa ou por apenas um tipo de meio.

O projeto Narval (Network Assistant Reasoning with a Validating Agent Language, sob licença GNU LGPL) pode ser executado no próprio computador pessoal ou em um servidor remoto. Este se comunica por meios normais, como e-mail, web, telnet, celular, GUI específica, etc. O sistema executa sequências de ações descritas pelo usuário para uma ampla gama de tarefas, como filtrar notícias para o jornal da manhã, ajudar a navegar na web por meio de filtro de anúncios e lixo eletrônico, cuidar de tarefas repetitivas (como responder e-mails), negociar data e hora de reuniões, etc. A sua arquitetura é bastante complexa, fazendo uso de Inteligência Artificial, Agentes, Redes de Petri, sistemas baseados em regras, programação de contratos, planejamento e aprendizagem automática. Podem ser

desenvolvidos plugins em Python (Chauvat, 2000), (Thenault, 2011). Tal projeto é interessante por possuir a possibilidade de se trabalhar com comportamentos configuráveis, na forma de plugins. Contudo, seu processamento é local, sem distribuição do comportamento. Além disso, a programação dos plugins é feita no próprio ambiente em que o sistema é executado, o projeto não tem tido atualizações recentes, indicando uma provável descontinuação. A proposta apresentada neste artigo amplia o conceito dos plugins do Narval para o contexto de SOA. Dessa forma, os plugins podem ser vistos como serviços web conectados dinamicamente ao SAP. Sua execução se torna distribuída, liberando o processamento do SAP para o gerenciamento, coreografia e orquestração das ações em si.

O Siri (2011) é um projeto bastante recente. Ele se utiliza de informações de preferências pessoais dos indivíduos e de um histórico de interação para ajudá-lo a resolver tarefas específicas. É baseado na web e links de resultados de buscas. O usuário diz para seu assistente o que quer fazer e este pesquisa em fontes de informações para poder auxiliar o usuário. O software evolui com a experiência de interação. Ele se utiliza também do contexto pessoal (lugar, horário, histórico) do usuário. O Siri foi adquirido pela Apple e é um sistema fechado e proprietário. Entretanto, o mesmo serve de inspiração para o modelo proposto na medida em que o SAP age para buscar na Internet informações que podem servir de auxílio ao usuário, conforme um contexto específico. Outra característica é a forma de comunicação com o usuário. O usuário se comunica e interage com seu SAP como se estivesse conversando com outra pessoa.

Um esforço bastante relevante é o projeto PAL (2011), financiado pela agência americana DARPA. Seu nome é um acrônimo para Personalized Assistant that Learns. Este tem por objetivo auxiliar os usuários nos trabalhos com sistemas computacionais e automatizar tarefas rotineiras para liberar as pessoas para tarefas mais importantes (Markoff, 2008). A intenção é criar um sistema cognitivo que possa raciocinar, aprender e lidar com situações imprevistas como forma de fornecer assistência em situações militares, especificamente. Além disso, o sistema deve poder se adaptar a mudanças em seu ambiente, aos objetivos dos seus usuários e as suas tarefas, sem a necessidade de alterar sua programação ou de intervenção técnica. Por ser um projeto com enfoque militar, há muito pouca informação aprofundada sobre o projeto. Este projeto é conduzido por pesquisadores em inteligência artificial, percepção, aprendizado de máquina, processamento de linguagem natural, representação de conhecimento, diálogo multimodal, ciberconsciência, interação homem-máquina e planejamento flexível (PAL 2011). Este projeto supre quase todos os requisitos funcionais desejáveis para SAPs, a não ser a integração com processos de negócio da empresa. Contudo, a proposta apresentada neste artigo busca também a padronização na criação de assistentes pessoais por meio de uma arquitetura de referência, e não especificamente criar comportamentos em si. Os comportamentos são baseados na distribuição desses na forma de plugins, serviços web, distribuídos na Internet. Os esforços apresentados no projeto PAL podem e devem servir de inspiração para a criação de comportamentos de planejamento, aprendizado, inteligentes, etc., que podem ser criados por terceiros e utilizados pelo SAP apresentado aqui.

Neste ponto foram apresentados esforços no sentido de se desenvolver SAPs para auxiliar os usuários em certas tarefas. Contudo, não foi encontrado qualquer modelo ou arquitetura de referência que visasse a criação de SAPs abertos ou com integração aos processos de negócios das empresas. Em qualquer projeto encontrado, os comportamentos ou são específicos já criados pela própria empresa, ou se em formato plugável (como no projeto Narval), estão em um formato próprio do projeto. Caso se deseje adicionar uma nova funcionalidade, os novos comportamentos devem ser desenvolvidos especificamente para tal projeto e no formato definido por eles, não havendo um padrão realmente aberto que permita uma escalabilidade em larga escala das funcionalidades.

3 Proposta de Modelo e Arquitetura de Referência de SAP

A construção de um SAP, além de envolver aquelas quatro dimensões, todas complexas e inter-relacionadas, requer que o projetista defina o fim para o qual o SAP deve ser usado, ou seja, os seus objetivos e funcionalidades. Neste sentido, é importante que o projetista possa ser, de alguma forma, amparado no processo de "derivação", sobre todos aqueles requisitos, para o SAP em particular desejado.

A concepção de uma arquitetura de referência é vista como uma forma de apresentar um padrão genérico para um projeto e deve abordar os requisitos para o desenvolvimento de soluções, guiado pelo modelo de referência e por um estilo arquitetural, de forma a atender as necessidades do projeto (MacKenzie et al., 2009). Um modelo de referência, por sua vez, é uma divisão de funcionalidades, juntamente com o fluxo de dados entre as partes. Esse deve possuir características de domínios maduros, decorrente da experiência sobre este domínio. Ele se caracteriza como um padrão de decomposição do problema. Já o estilo arquitetural ou modelo de arquitetura descreve os tipos de elementos e suas relações, juntamente com um conjunto de restrições sobre como eles podem ser utilizados, além de padrões de interação entre os elementos. Tais restrições, sobre a arquitetura e sobre o sistema em si, são vistas na forma de uma imagem da utilização do sistema como um todo (Bass, 2003).

Para o modelo e arquitetura propostos, a abordagens de SOA (service oriented architecture - SOA) é considerada essencial e norteia as suas concepções.

Os conceitos mais recentes de softwares para assistência pessoal interseccionam-se em diversas premissas com a tecnologia de agentes de software, sendo que a abordagem de agentes pode ser usada como metáfora para a explicação dos fenômenos e aspectos de comportamento que um SAP pode desempenhar dentro de um ambiente computacional e, com isso, ser usada como fundamentação teórica para a modelagem de SAPs. Este é o caso deste trabalho, que embora não possa ser considerado um SAP baseado em agentes, se apropria desses conceitos para sua modelagem: um agente é um sistema computacional, posicionado em algum ambiente, que é capaz de agir com autonomia flexível, visando atingir os objetivos para o qual foi projetado (Jennings, 1994). Posicionamento em termos de que o agente recebe sinais de entrada dos seus sensores vindos do ambiente no qual está localizado,

e de que pode executar ações contextualizadas que modifiquem de alguma forma este ambiente. Autonomia em termos de que o agente deve ter a possibilidade de agir sem a intervenção direta de usuários ou de outros agentes, e de que deve poder controlar totalmente suas ações e seu estado interno. Flexível em termos de como o agente age para atingir seus objetivos, envolvendo as capacidades de responsabilidade, pró-atividade e sociabilidade.

SOA representa uma nova forma de pensar quanto ao projeto da arquitetura de um sistema, quanto à sua filosofia de desenvolvimento e quanto à sua posterior integração a outros sistemas. O princípio que rege SOA é de que o desenvolvimento de uma aplicação grande, monolítica, toda nova e complexa deve ser evitada e substituída por um conjunto de aplicações pequenas, já existentes (quando possível) e mais simples (Singh, 2005). Ou seja, uma aplicação passa a ser fisicamente composta (em tempo de projeto ou de execução) por vários e pequenos módulos de software especializados, distribuídos, acessados remotamente, interoperáveis e reutilizáveis, que são "inteligentemente" unidos conforme o processo, seguindo padrões, podendo a aplicação ser fácil e rapidamente (re)composta para um (novo) processo desejado dado ao baixo acoplamento da aplicação / serviços. A tais módulos dá-se o nome de serviço de software, que pode ser implementado em várias tecnologias, sendo o padrão de facto o de web services. Para a visão de SAP deste trabalho, este então pode ser visto como um projeto SOA, em que seus comportamentos (funcionalidades) são modelados como serviços, invocados de repositórios locais (do ambiente do SAP), da empresa ou de provedores externos de serviços de software (como software-houses). Esta visão dá bases para o SAP para não apenas se adaptar a novos processos como também aos diferentes ambientes computacionais onde ele vier a ser executado. A seleção de serviços mais adequados para o processo pode ser feita dinamicamente (portanto, não definida a priori, rigidamente, e considerando inclusive aspectos de QoS (Perin e Rabelo, 2011). Graças ao baixo acoplamento dos serviços (do SAP), e conforme o ambiente da empresa e de disponibilização de serviços de software, modelos arquiteturais, de disponibilização de serviços (sob demanda) e até mesmo de pagamento podem ser adotados, como o SaaS (Software-as-a-Service) (IBM, 2008).

Sob a ótica dessas duas abordagens de base, quatro aspectos principais balizaram o desdobramento dos requisitos gerais de SAP no modelo e arquitetura de referência:

- SAPs devem ser concebidos para permitir a integração de SAPs a processos de negócios das empresas. Ou seja, um SAP não deve ser algo completamente independente e desconexo dos sistemas da empresa e dos processos de negócios associados. Com isso, um SAP deve deixar de ser um sistema puramente reativo à interação do usuário, mas deve também ser proativo, executando (pelo menos algumas e com variados graus de autonomia, configuráveis ou não) tarefas de processos de negócios por ele.
- Em grande medida como consequência do aspecto anterior, SAPs devem ser concebidos para serem integrados, como mais uma "peça" de um grande ambiente de operações de uma empresa. SAPs costumam ser softwares fechados, projetados para realizarem atividades sem uma preocupação de

que devem transacionar com outras aplicações (de uma mesma ou até de outras empresas / processos de negócios interorganizacionais), num ambiente distribuído e heterogêneo.

- SAPs devem ser flexíveis, adaptativos e escaláveis. No escopo do ciclo de vida da empresa e das suas transações empresariais, novos processos de negócios são inseridos, uns modificados, outros retirados. Portanto, um SAP deve ser tanto flexível para se adequar aos processos em questão, como sua arquitetura funcional escalável para poder incorporar novas funcionalidades e processos de negócio. Cada processo de negócio tem inúmeras particularidades, tanto de contexto como de ambientes computacionais envolvidos. Portanto, há necessidade de que se adaptem ao ambiente geral de execução. Isso é potencializado pela noção de serviços de software, em que SAPs devem poder ter novas funcionalidades ou mesmo ter a possibilidade de "escolherem" de onde deverão acessar a funcionalidade requerida ou desejada.
- SAPs devem ser concebidos para interoperar. Isso significa que devem usar tanto quanto possível especificações ou padrões abertos de projeto, comunicação (em seus vários níveis, incluindo o semântico) e integração, criando-se um cenário aberto, que minimize os problemas de interoperabilidade entre o SAP e os sistemas envolvidos.

3.1 Modelo de Referência

Entre os elementos principais do modelo de referência para assistentes pessoais proposto nesse trabalho (Figura 1) está a Ação, que se refere aos comportamentos do assistente, a Interação para com o usuário ou outros assistentes ou softwares, e o Gerenciamento, para gerenciar a execução do assistente, organizar as informações e o fluxo dessas entre os demais elementos.

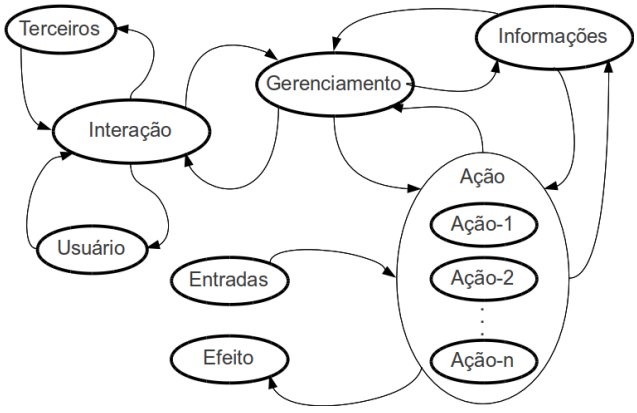

Figura 1. Modelo de Referência para Softwares Assistentes Pessoais.

A Ação é a forma como o assistente age, que é composta por um conjunto de n ações que podem ser utilizadas pelo SAP. As Informações são os elementos úteis para a execução de uma determinada ação ou de um conjunto de ações. Por Entradas, tal qual um agente sente seu ambiente por meio de sensores, referem-se ao conjunto de informações que chegam de elementos externos ao assistente e que servem para iniciar uma determinada ação. Essa ação reage com um Efeito, que pode ser tanto interno do próprio assistente, como externo, com parceiros, outros assistentes, outros softwares, ou mesmo pela interação com o usuário.

O Gerenciamento é responsável pela organização das informações do usuário, pela execução das ações e pelo fluxo dessas informações entre os elementos do modelo. Um conjunto de informações acerca do usuário do SAP e das ações é necessário para que o assistente gerencie sua execução. Elas devem evoluir com o tempo em concordância com a evolução do usuário, tarefas, preferências, etc. Por meio delas, o Gerenciamento deve decidir quando e como iniciar uma ação, ou seja, tornar uma ação operacional quando certas condições forem satisfeitas.

A Interação é a forma como se dão as comunicações e negociações entre o usuário e módulo de Gerenciamento do assistente, ou entre o módulo de Gerenciamento e outros elementos.

3.2 Arquitetura de Referência

A existência de uma arquitetura de referência para assistentes pessoais é relevante, mas não suficiente para fazer frente àqueles requisitos anteriormente expostos. É importante que o assistente possa ter a característica de comportamentos "plugáveis", se manter em um padrão de comunicação e ainda trabalhar com a questão de serviços de software, propiciando assim as bases para a flexibilidade e escalabilidade desejadas. Assim, os comportamentos do assistente pessoal são habilitados através da invocação de serviços web, distribuídos na Internet, na forma de uma "federação" de serviços web (Zambiasi e Rabelo, 2012), também podendo ser vista como uma "nuvem", dentro da qual estão inseridos os vários repositórios de serviços, fornecidos por diferentes empresas, organizações ou desenvolvedores (Figura 2).

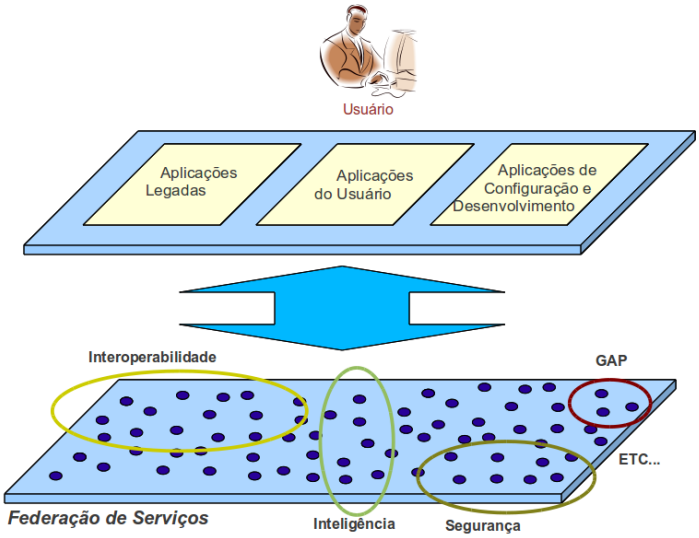

Figura 2. Visão geral da proposta.

Conforme apresentado no modelo de referência, para que o assistente possa auxiliar o usuário em suas tarefas diárias, é necessário que haja a "interação" entre eles. Aliando o modelo de referência com o estilo arquitetural SOA, pode-se modelar uma visão geral (Figura 2). Nessa visão, há o usuário, interagindo com o seu assistente pessoal por meio de três elementos: (i) Aplicações Legadas (softwares da empresa, ou outros softwares, que podem ser utilizados pelo SAP para auxílio do usuário. Deve haver uma interface para interoperabilidade); (ii) Aplicações do Usuário (aplicações que o usuário lida em seu dia a dia e que devem ser utilizadas para interagir com seu assistente pessoal como e-mails, twitter, instant messaging, SMS, etc.); e (iii) Aplicações de Configuração e Desenvolvimento (podem ser específicas para configurar um assistente pessoal ou serviços que esse assistente pessoal possa precisar), também chamado de "Ferramentas".

Com base no modelo de referência anteriormente exposto, a Figura 3 apresenta uma proposta de arquitetura de referência para SAP. Ela corresponde a uma derivação do modelo abstrato, e é independente de modelo e de tecnologia de implementações.

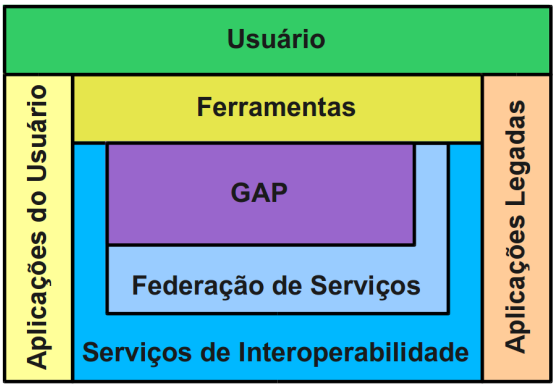

Figura 3. Arquitetura de Referência para Softwares Assistentes pessoais.

A área mais acima da Figura 3 não é exatamente um elemento da arquitetura, mas sim a visão que o usuário possui do SAP. O interfaceamento entre o usuário e seu assistente pessoal é feito via três elementos da arquitetura, que são:

- Aplicações Legadas: Softwares de terceiros, ou desenvolvidos pela empresa, e que o usuário necessita que o SAP se integre a ele.
- Aplicações do Usuário: Aplicações que podem servir para interação entre o usuário e o SAP. Pode ser, conforme apresentado no sistema Sandy (Mann, 2011), serviços de e-mails, mensagens via Twitter, sistemas de Instant Messaging, ou mesmo SMS via telefonia móvel.
- Ferramentas: Aplicações de configuração de SAPs, de comportamentos ou ambientes de desenvolvimento.

Com relação as pessoas que podem trabalhar com os SAPs dessa proposta, elas podem possuir vários níveis de conhecimento acerca da estrutura envolvida, tais como:

- Usuário Simples: Pode cadastrar um assistente pessoal, criar e configurar comportamentos e as informações dos comportamentos do seu assistente.
- Usuário Avançado: Pode compor comportamentos mais complexos no assistente por meio da composição de condicionais, laços e orquestração de serviços web via uma interface de algoritmos ou fluxogramas, textual ou preferencialmente gráfica.
- Desenvolvedor de serviços web: Desenvolve serviços web que podem ser utilizados para compor comportamentos.
- Desenvolvedor de serviços de interoperabilidade: Desenvolve interfaces de serviços web para sistemas desenvolvidos sem essa tecnologia (e padrões associados).

- Desenvolvedor de Assistentes Pessoais: Desenvolvedor avançado que pode criar núcleos de execução de assistentes pessoais, além de interfaces de criação e configuração desses assistentes.

A Federação de Serviços representa o conjunto de serviços web distribuídos na Internet e que podem ser utilizados nos comportamento do SAP isoladamente, ou que podem servir para compor outros serviços web (orquestração e composição de serviços).

Os Serviços de Interoperabilidade, por sua vez, são criados para servirem de interface com outros sistemas que não estão no padrão SOA de serviços web. Neste caso, adota-se uma abordagem de integração onde os aspectos de interoperabilidade são tratados de forma desacoplada dos aspectos intrínsecos funcionais e não funcionais de um serviço, como proposto no projeto COIN (COIN, 2009).

O Gerenciador de Assistentes Pessoais (GAP) é responsável pela coordenação da execução dos comportamentos e suas informações. Os comportamentos que podem ser configurados no GAP podem ser classificados em três níveis básicos, apresentados aqui de forma simplificada:

- Comportamento Simples: Composto de informações para a chamada do serviço web. Este é executado toda vez que o comportamento é chamado, enviando os parâmetros de entrada, executando a operação requisitada e retornando um resultado. A informação de retorno pode ser usada em outros comportamentos.
- Comportamento Condicional: Além das características do Comportamento Simples, esta forma de comportamento é composta também por especificações condicionais para a execução do comportamento.
- Comportamento Complexo: Necessita de um conhecimento mais avançado do usuário. O usuário pode modelar o comportamento na forma de um fluxograma, ou algoritmo, com laços, condicionais e outros elementos.

Os comportamentos são modelados em uma interface de configuração do SAP, sendo que a forma como isso é feito depende de cada desenvolvedor. Porém, partindo-se de uma arquitetura de referência, garante-se que instâncias de SAPs devem ser coerentes com o modelo maior, mesmo que tecnologias mudem ou que funcionalidades num dado momento não presentes no SAP venham a ser introduzidas no futuro.

4 Verificação

Uma das formas de verificação da Arquitetura de Referência é a factibilidade de criação de uma instância implementada a partir dela. Para tal, foi gerado um protótipo baseado em um exemplo específico, ou seja, uma instância que implementa comportamentos de um SAP para um conjunto de atividades para um usuário.

No exemplo estudado, é considerado o cenário em que um funcionário de uma empresa tem a função de atuar em cima do processo de negócio gerenciamento do estoque de produtos. De acordo com a lógica (configurada) desse processo, quando o estoque de um produto diminui e passa de um limite mínimo, fica a cargo desse funcionário efetuar nova compra, de modo a manter o estoque sempre em dia e não permitir que a produção fique parada por falta de matéria-prima, por exemplo. No final do dia, o funcionário faz um relatório das atividades realizadas durante todo o dia, desde o início do processo de uma compra, até as modificações em uma ordem de compra e fechamento desta. O funcionário possui acesso a um sistema legado de controle de estoque, por meio de um login de usuário e uma senha. Neste caso, o SAP deve ser utilizado para auxiliar o funcionário a gerenciar o sistema de controle de estoque e para a criação do relatório das atividades diárias. Ainda, é considerado que o usuário deseja se comunicar com o seu SAP por meio de aplicações que não precisam ser instaladas em seu computador e que podem ser acessadas de qualquer lugar, inclusive do seu dispositivo móvel. No caso instanciado implementado, o usuário pretende se comunicar com seu SAP como se fosse uma outra pessoa, por meio de mensagens via Twitter, Gtalk, e-mail ou mensagens de texto no celular.

4.1 Protótipo

Sob o cenário apresentado, alguns elementos de implementação para compor o SAP foram derivados a partir da Arquitetura de Referência proposta (Figura 4). Em termos gerais implementou-se uma instância de SAP. Considerando a visão SOA do modelo, a arquitetura de referência pode ser instanciada com a utilização de diferentes abordagens, com diferentes serviços (funcionalidades), ferramentas de implementação e mecanismos de comunicação. Mais do que isso, o SAP passa a ser não um software monolítico, mas sim um conjunto de sistemas distribuídos, abertos, que se comunicam via rede, envoltos num mesmo arcabouço lógico.

As implementações feitas são aqui apresentadas conforme sua classificação na Arquitetura de Referência proposta. Todos os sistemas servidores foram desenvolvidos em linguagem de programação Java, os serviços web em linguagem PHP – mantendo o padrão SOA da OASIS (2006) – e as interfaces em PHP e HTML via web.

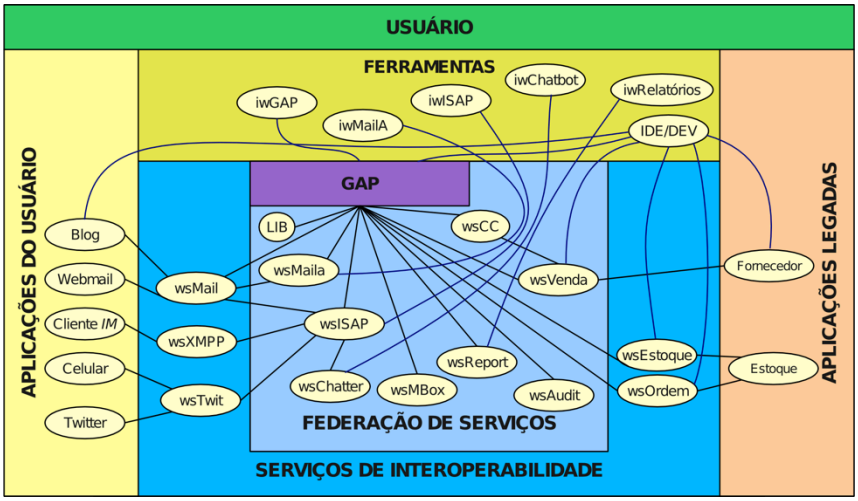

Figura 4. Arquitetura derivada para o Caso particular.

Aplicações Legadas:

- Estoque: Sistema de controle de estoque. Possui informações dos produtos, fornecedores, quantidades máxima, atual e mínima dos produtos que são utilizadas para iniciar um processo de compra.
- Fornecedor: Controle de produtos no fornecedor. Informa os produtos, quantidades, e preço. Vários fornecedores podem ser cadastrados.

Aplicações do Usuário:

- Twitter: Serviço de publicação de mensagens de até 140 caracteres. Acesso via web ou aplicativos especiais para cada sistema operacional.
- Celular: Via Twitter, o usuário pode habilitar receber/enviar mensagens via mensagem de texto no celular.
- Cliente IM: Sistema para troca de mensagens com o assistente pessoal. O usuário vê seu assistente pessoal como uma pessoa de seus contatos conectada ao serviço. É necessário uma conta no Gmail para o assistente.
- Webmail: Sistema para troca de mensagens de e-mail com o assistente.
- Blog: O assistente envia postagens a um blog para publicar relatórios que podem ser vistos por outros, ou seja, relatórios públicos.

Serviços de Interoperabilidade: Estes são serviços que fornecem uma interface de serviço web SOA da OASIS para outros sistemas que possuem tal interface. Isso pode se dar por meio de chamadas ao tipo específico de serviço de software ou na forma de acesso direto ao banco de dados do sistema legado. Neste último caso, deve haver a permissão para acesso

ao recurso do banco de dados e, em geral, pode ser feito pelos desenvolvedores da empresa que utilizam tal sistema. Para esse exemplo são definidos:

- wsEstoque: Acessa informações do sistema de controle estoque.
- wsOrdem: Gerenciamento das ordens de compra.
- wsVenda: Acessa o sistema automático de vendas, um programa servidor executando do lado do sistema de fornecedores.
- wsTwit: Acessa uma conta no serviço de microblog Twitter.
- wsXMPP: Acessa um servidor de conexão ao serviço de IM Gtalk.
- wsMail: Acessa os e-mails de um usuário em um servidor.

Federação de Serviços:

- wsCC: Simula transações com cartão de crédito.
- wsAudit: Utilizado para armazenar informações de auditoria.
- wsReport: Geração automática de relatórios. Recebe mensagens referente as atividades e quando requisitado retorna o relatório compilado.
- wsMBox: Gerencia mensagens provindas do ISAP.
- wsChatter: Gera respostas para o usuário e quem mais se comunicar com o assistente pessoal tal como um chatbot.
- wsISAP: Serviço de interface com o usuário. Se utiliza de serviços como o wsChatter, wsXMPP e wsMail para trocar mensagens com o usuário. Há um sistema servidor que faz o gerenciamento das mensagens. Quando um usuário se comunica com a Interface Social para Assistentes Pessoais (ISAP) a mensagem é avaliada e enviada ao GAP. Ela também decide a melhor forma de enviar uma mensagem do GAP para o usuário.
- LIBS: Conjunto de bibliotecas de funções que são utilizadas para criar os comportamentos no GAP (string, math, datetime, kqml, myMessageTable).
- Ferramentas: Programas de interfaces web para configuração de alguns sistemas.
- iwISAP: Interface para configuração do ISAP do assistente pessoal.
- iwChatbot: Configuração das conversas do assistente pessoal. Tem o objetivo de servir como um feedback das requisições do usuário e como uma forma mais amigável de interação entre o usuário e seu assistente pessoal.
- iwRelatorios: Configuração do sistema gerador automático de relatórios.
- iwGAP: Ambiente de configuração do assistente pessoal e seus comportamentos. Os comportamentos são aqui configurados na forma de um algoritmo. O GAP (Gerenciador de Assistentes Pessoais) é um servidor que gerencia a execução autônoma dos assistentes pessoais.

Havendo esse ambiente aberto de softwares e serviços, foi criado um SAP e alguns comportamentos para efetuar o gerenciamento do sistema de controle de estoque, de responsabilidade do usuário. Em verdade, neste protótipo existem dois caminhos distintos para a criação do comportamento automático de compra:

- Usuário avançado: pode criar comportamentos na forma de algoritmos na interface de configuração do Assistente Pessoal;
- Usuário básico: Requisita a um desenvolvedor da empresa um serviço web que efetua o processo automático de compra. O usuário utiliza este serviço web no seu comportamento de compra automática;

Aqui, neste exemplo estudado, é considerado que o usuário é do tipo avançado e configurou as informações e algoritmos que efetua a compra automática. Para tal, faz uso de uma interface especial de configuração, implementada neste trabalho (Figura 5).

Figura 5. Interface de configuração do assistente pessoal.

Neste caso, os seguintes comportamentos foram criados:

- AtualizaHora: Invoca operações do serviço web datetime para atualizar data e horário para serem utilizadas por outros comportamentos.
- ISAPalive: Envia, de quando em quando, uma mensagem ao ISAP, informando que ele deve manter a interface do assistente funcionando.

- mbox: Invoca operações do wsISAP para verificar e buscar mensagens do usuário para o GAP. As mensagens são identificadas e armazenadas no wsMBox, com informações de qual comportamento essa é destinada.
- Report: É verificado se é um horário específico. Caso afirmativo, são requisitados os relatórios das últimas atividades. Relatórios públicos são enviados ao Blog e privados são enviados ao usuário por e-mail.
- Compra-ordem: Se um produto está com estoque baixo, gera uma ordem de compra. Essa ordem é enviada ao fornecedor escolhido pelo sistema gerenciador de estoques. Cada nova ordem criada é enviada uma mensagem ao usuário e uma atividade é enviada ao wsRelatorio.
- Compra: Se há uma ordem aberta, verifica o estado dela no fornecedor. Se esta foi alterada, altera-a também no wsOrdem e envia uma mensagem ao usuário do assistente pessoal, requisitando confirmação. Se a ordem foi aceita pelo usuário, efetua o pagamento, finalizando a compra. Quando uma ordem é cancelada, o sistema seleciona outro fornecedor para o produto.
- Compra-altera: Aguarda a confirmação do usuário de uma ordem alterada. Se o usuário confirma, então a ordem volta a ficar aberta. Se for cancelada, então cancela a ordem no wsOrdem e no wsVenda.

O GAP implementado executa os comportamentos por ciclos. Cada comportamento pode conter diversas atividades e em cada ciclo uma atividade de cada comportamento é executada. Essas atividades podem ser atribuição de valor a uma informação, invocação de uma operação em um serviço web, verificação de um condicional ou de um laço, etc.. Na Figura 6 é possível visualizar o editor de comportamentos do assistente pessoal desenvolvido.

Figura 6. Editor de comportamentos do assistente pessoal.

A esquerda da interface (Figura 6) há uma área para configurar as informações básicas do comportamento, ligar / desligar sua execução, efetuar um reload do comportamento e lista de links para repositórios e provedores de serviços web. A direita há o editor de comportamentos propriamente ditos, com opções de criação de condicionais, laços, atribuição de valores a variáveis e configurações de chamadas de serviços web.

Para que o SAP possa ser visto pelo usuário como se fosse outra pessoa conectada na internet e interagindo com ele, é necessário a criação de contas nos tipos de serviços utilizados pelo ISAP, tal como conta de e-mail (no caso de ser criada uma conta no Gmail, a mesma já pode ser utilizada pelo assistente pessoal como conta no Gtalk), conta de Gtalk, conta no Twitter e um Blog. Na implementação foi criada uma assistente pessoal de nome Arisa (Acrônimo para Assistant Representative: an Instance using Services Architecture), com suas respectivas contas para serem utilizadas pelo ISAP: (i) Gmail e Gtalk: personal.assistant.arisa@gmail.com; (ii) Twitter: @arisa_ap; e (iii) Blog: http://www.projetoarisa.com.br do Blogspot. Também foram cadastradas informações de dois fornecedores de produtos e informações referentes ao sistema de estoque que o SAP deve gerenciar.

4.2 Teste

No sistema de estoque, unidades do produto mouse (código 001) são vendidas, ficando suas informações da seguinte forma: quantidade = 3, quantidade mínima = 10 e quantidade máxima = 20. Ou seja, quando a quantidade deste produto estiver abaixo de 10, são comprados os produtos necessários para chegar à quantidade máxima. Este produto é fornecido pelos fornecedores, de nomes fictícios, John Doe Ltda e Sora Konpyuuta ao valor de 25 reais e 28 reais a unidade, respectivamente.

No caso, o primeiro fornecedor possui apenas 10 produtos em estoque para venda, e o segundo possui 50. O fornecedor selecionado inicialmente para a compra é o primeiro que foi cadastrado no sistema como fornecedor, no caso foi selecionado o fornecedor John Doe Ltda. As trocas de mensagens entre assistente pessoal e usuário para efetuar o gerenciamento das ordens de compra são visualizadas na Figura 7.

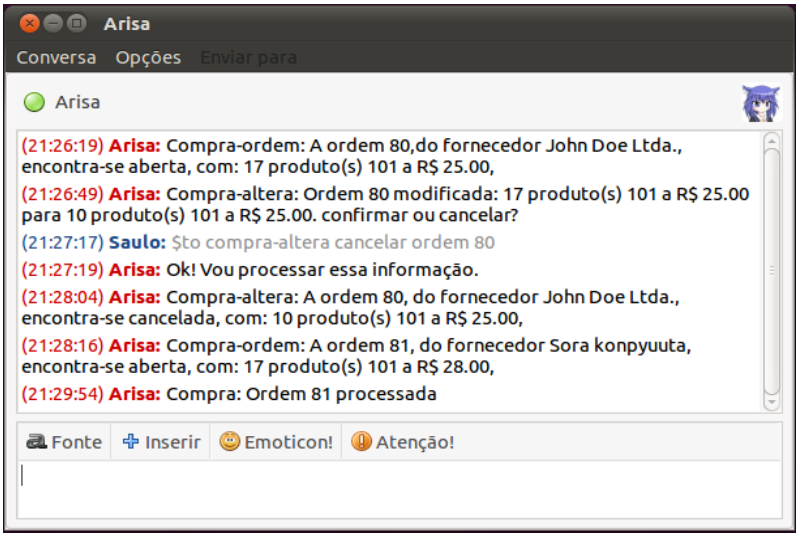

Figura 7. Troca de mensagens entre o assistente pessoal e o usuário.

Quando o assistente pessoal verifica que o estoque está baixo, é iniciado o processo de compra. Como a quantidade do produto 001, do fornecedor John Doe Ltda, não é o suficiente para o total da ordem de comprar (i.e. 10 produtos), então a ordem é modificada e o assistente requisita confirmação da alteração. Quando o usuário necessitar comprar todos os produtos para repor o estoque no máximo, a ordem deve ser cancelada e uma nova criada, para outro fornecedor. Tendo corrido tudo corretamente com a segunda ordem criada, o pagamento é efetuado e a ordem é fechada, atualizando os estoques no fornecedor e no

cliente. Por fim, um relatório completo é enviado por e-mail no horário definido pelo usuário.

Um questão a se observar é que este teste foi realizado com o usuário conectado em seu usuário no Gtalk. Caso o usuário não estivesse conectado, as mensagens seriam trocadas via mensagem de texto do dispositivo móvel – Telefone Celular (Figura 8).

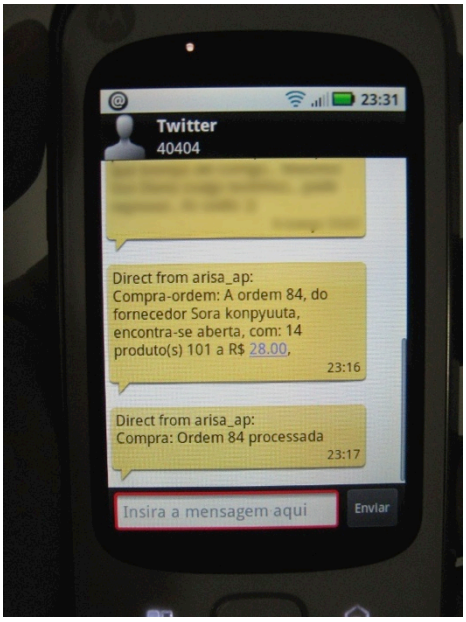

Figura 8. Comunicação via SMS em um dispositivo móvel.

Em tempo, é importante se observar que pode não ser interessante que o usuário fique recebendo informações do SAP a todo o momento, ou mesmo o usuário pode querer que esse cancelamento de ordem de compra seja automático. Contudo, neste exemplo várias mensagens são trocadas com o usuário para fins de testes e verificação do funcionamento do comportamento.

4.3 Avaliação

Uma outra forma de verificação, além a implementação e dos testes efetuados nesta, foi a apresentação de uma palestra para dois tipos de possíveis utilizadores de SAPs condizente com a proposta. As palestras tiveram uma horas de duração cada. A primeira palestra foi direcionada à potenciais usuários de SAPs e a segunda para potenciais desenvolvedores de SAPs ou comportamentos para SAPs (serviços web).

Em cada apresentação foi aplicado um questionário específico. Ao todo houveram 61 pessoas que responderam os questionários, sendo 42 pessoas na visão de usuário e 19 pessoas na visão de desenvolvedor es. Os questionários foram elaborados com respostas "baseadas" na escala Likert (1932) – Concordo Fortemente, Concordo, Indiferente / Não vejo correlação, Discordo Discordo Fortemente, Não sei opinar.

As questões para os desenvolvedores focaram na relevância da proposta, inovação, se o protótipo atende os requisitos do modelo conceitual, se a proposta pode se tornar um modelo de negócio, se pode ser utilizado em cenários reais, sobre a praticabilidade da interoperabilidade do modelo, além de um espaço para comentários. As questões para usuários focaram mais na usabilidade, praticidade da interface de configuração, se SAPs seriam de utilidade em tarefas diárias, benefícios de poder escolher comportamentos distribuídos e de terceiros, inovação, possibilidades de pagar pela utilização de SAPs e flexibilidade dos comportamentos.

O Resultado geral das respostas foi satisfatório, com a sua grande maioria selecionando a resposta "Concordo". Contudo, algumas observações foram feitas, tais como:

- "A formulação das regras (algoritmos) pode ser algo muito complexo para usuários que mal sabem utilizar a internet";
- "a personalização e design deve ser mais facilitado, retirando a parte lógica para o usuário final, já que muitos mal sabem mandar e-mail".
- "...a interface de usuário poderia ser um pouco mais elaborada, mas se tratando de um protótipo está bom".
- "gostei da possibilidade de ter um assistente pessoal que possua uma interface 'humanizada'. Isso certamente atrai mais o usuário e facilita a relação com o software".
- "Pela demonstração não foi possível identificar até que nível o assistente consegue interpretar as frases digitadas na janela de chat... Seria interessante integrar a capacidade de aprendizado".
- "pode ajudar tanto quanto pode trazer problemas, se o software parar depois do funcionário já ter 'viciado' nas facilidades que o assistente traria".

Ao que se pode verificar nas observações feitas pelos entrevistados, muito circunda em relação à falta praticidade da parte de configuração dos comportamentos por estar em formato de algoritmos. Por ser um protótipo, a interface não foi desenvolvida para ser uma amigável aos usuários. Contudo, isso pode ser modificado para ser mais amigável, sem que a implementação se se distancie da proposta apresentada. Um outro ponto atacado foi em relação à interação entre SAP e usuário. Houve boa aceitação quanto à uma interface de comunicação humanizada mas foi citado a necessidade de melhoria na conversação. Isso pois o chatbot foi implementado de forma bastante simplificada, não seguindo conceitos e implementações de Processamento de Linguagem Natural (PLN), isso pois não era esse o enfoque desse trabalho. Um serviço mais "inteligente" de chatbot pode ser substituído pelo existente de forma bastante simples, uma vez que a proposta permite isso.

5 Considerações Finais

Este artigo apresentou uma proposta para uma Arquitetura de Referência para Softwares Assistentes Pessoais. Essa proposta se apropriou da Arquitetura Orientada a Serviços para servir de estilo arquitetural.

Com base nos resultados de testes sob a implementação final, observou-se nos testes de verificação que esta instância se comportou conforme o que foi proposto na arquitetura de referência e executou corretamente as ações associadas aos comportamentos do processo de negócio envolvido no estudo de caso. Embora apresentado apenas parcialmente neste artigo, o SAP proposto foi implementado em sua totalidade. Quanto ao desempenho, embora neste momento do trabalho não tenha sido o principal foco, apresentou boa estabilidade perante as atividades realizadas em uma arquitetura distribuída, mas em rede local.

O protótipo mostrou-se coerente com o modelo e arquitetura de referência, o que garantiu a coerência do SAP derivado. Com a implementação realizada, mostrou-se igualmente que é possível realizar aquela derivação assim como ter um SAP integrado ao ambiente de negócios da empresa, interoperando com vários subsistemas, dentro de uma arquitetura aberta e de intenso uso de padrões. Além disso, ela teve boa aceitação pelos entrevistados em palestras apresentadas sobre a implementação efetuada.

A proposta aqui apresentada, além de apresentar uma inovadora arquitetura de referencia para a criação de SAPs, fornece indicativos da criação de comportamentos baseados em SOA. Este estilo arquitetural aberto permite que funcionalidades do SAP não precisem ter sido criadas especificamente para o SAP do usuário em si, mas podendo ser utilizadas para a composição de comportamentos de forma dinâmica. Isso também permite que a própria empresa do usuário forneça funcionalidades que façam a integração do SAP do usuário com os processos de negócio da empresa. Outra vantagem é que funcionalidades podem ser desenvolvidas por terceiros, e utilizadas pelo usuário de forma gratuita ou paga. Isso pode levar a um foco específico de negócios, que é a criação de comportamentos e funcionalidades específicas para SAPs, apenas criando serviços web e disponibilizando-os aos usuários para a composição dos seus comportamentos. Como os serviços web vêm se firmando como um padrão de mercado, isso provê, consequentemente, um leque já existente de opções de funcionalidades e comunicação para quem queira desenvolver SAPs que sigam a proposta desse trabalho.

Apesar disso tudo, ainda se constata que a área de SAPs têm ainda enormes desafios a serem enfrentados, incluindo ao que se refere a limitações das TIs atuais para certos propósitos (por exemplo, o problema da interoperabilidade semântica), os impactos organizacionais, a complexidade do problema de composição de comportamentos dentro da abordagem SOA, etc.

Neste sentido, este trabalho deve ser visto como uma contribuição para a área de SAPs, indo na direção dos requisitos, em vários níveis, identificados na literatura. Há ainda vários aspectos que devem ser mais bem estudados e implementados. Os próximos passos de

curto prazo focam no aprimoramento da questão de composição de serviços e da "inteligência" quando as tomadas de decisão, ou seja, em relação à autonomia do assistente.

Agradecimentos

Este trabalho foi parcialmente financiado pela CAPES e CNPq.

Referências

Angehrn, A., Nabeth, T., Razmerita, L. et al. K-InCA: Using Artificial Agents for Helping People to Learn New Behaviours. Proc. IEEE International Conference on Advanced Learning Technologies , 225-226. 2001.

Bass, Len; Clements,Paul and Kazman, Rick. "Software architecture in practice". Addison-Wesley, 2003.

Bocionek, S. "Software secretaries: learning and negotiating personal assistants for the daily office work In Systems, Man, and Cybernetics", In: Humans, Information and Technology. 1994 IEEE International Conference on, 12 vol.1. 2-5 Oct. 1994.

Bush, J.; Irvine, J. and Dunlop, J. "Personal Assistant Agent and Content Manager for Ubiquitous Services". Wireless Communication Systems, 2006. ISWCS'06. 3rd International Symposium on , 169-173. 2006.

Chauvat, N. "Narval, the intelligent personal assistant or how the french Linux Gazette is built", In: Linux Gazette, issue 59. Nov, 2000. Disponível em <http://linuxgazette.net/issue59/chauvat.html>, acessado em 14/02/2011.

COIN. "Coin: enterprise collaboration & interoperability", Disponível em: <http://www.coin-ip.eu>. 2009. Acessado em Jun/2011.

ebXML, Electronic Business using eXtensible Markup Language, Disponível em <http://www.ebxml.org/> Acessado em Jun/2011.

Etzioni, O.; Weld, D.. "A Softbot-based interface of the internet". Comm. of ACM. July. 1994.

Hoyle, M.A. and Lueg, C. "Open Sesame!: A Look at Personal Assistants". Proceedings of the International Conference on the Practical Application of Intelligent Agents (PAAM97), London, 1997, 51 60, 1997.

Huhns, M.N.; Singh, M.P., "Personal assistants", In: IEEE Internet Computing, Vol. 2, Issue 5, pp. 90-92. Sep./Oct. 1998.

Huhns, M.N. "Agents as Web services", In: Internet Computing, IEEE 6, 93-95. 2002.

IBM. Software as a service: Build a Web-delivered SaaS framework for forms and workflow-driven applications. <http://www.ibm.com/developerworks/architecture/library/ar-saasframe/>. 2008. Acessado em Jun/2011.

Jennings, N. - Cooperation in Industrial Multi-Agent Systems, World Scientific Series in Computer Science (Vol 43), 1994.

Likert, R. "A technique for the measurement of attitudes". New York. 1932

MacKenzie, C.; Laskey, K.; McCabe, F. at all. "Reference Model for Service Oriented Architecture 1.0", In: OASIS Standard, 12 October 2006. <http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf> acessado em Fev/2009.

Mann, H. "Free Personal Assistant - Meet Sandy". Disponível em <http://honestholly.com/free-personal-assistant-meet-sandy/>, acessado 14/02/2011.

Markoff, J. "A Software Secretary That Takes Charge", In: New York Times, 13/12/2008. Disponível em: <http://www.nytimes.com/2008/12/14/business/14stream.html?_r=1&scp=7&sq=personal%20assistant&st=cse>. Acessado em: 17/03/2009.

Michaell, T.; et all. "Experience With a Learning Personal Assistant", In: Communications of the ACM, July, 1994.

PAL, PAL Project, <http://pal.sri.com/>, acessado em 14/02/2011.

Perin de Souza, A.; Rabelo, R. J. "Uma solução aberta e flexível de descoberta de serviços para maior agilidade na integração BPM&SOA". Simpósio Brasileiro de Sistemas de Informação / V Workshop de Gestão de Processos de Negócios, p. 509-516, 2011.

RosettaNet, Disponível em <http://www.rosettanet.org/>. Acessado em Junho/2011.

Russel, S. e Norvig, P., "Inteligência Artificial". 2aEd, Tradução da segunda edição. Rio de Janeiro: Elsevier, 2004.

Schiaffino, S., Amandi A.. "Polite Personal Agents". in IEEE Intelligent Systems, p12-18. Jan-Fev 2006.

SHELLTOYS. "Personal assistant - day planner and personal information manager". Disponível em: <http://www.shelltoys.com/personal_assistant/>, acessado em 14/02/2011.

Singh, M. P. and Huhns, M. N. Service-Oriented Computing: Semantics, Processes, Agents. John Wiley & Sons, New York, NY, EUA, 1a. Edição. 2005.

SIRI, disponível em <http://siri.com/>, acessado em 14/02/2011.

Thenault, Sylvain. Narval-moved. Logilab Project. Disponível em: <http://www.logilab.org/908/>, Acessado em 14/02/2011.

UBL, Disponível em <Universal Business Language, http://www.oasis-open.org/committees/ubl/> Acessado em Jun/2011.

Weiss, G. “Multiagent systems: A Modern Approach to Distributed A.I.”. MIT Press, 1999.

Zambiasi, S.P.; Rabelo, R.J. “A Proposal for Reference Architecture for Personal Assistant Software Based on SOA”. In: IEEE Latin America Transactions. vol. 10, no. 1. - pg.1227-1234 Jan, 2012.