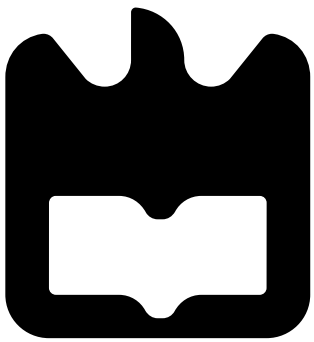
Universidade de Aveiro
Departamento de Eletrónica, Telecomunicações e Informática
2014

Nuno Aido

Integração de facilidades de voz em sistemas CRM



Nuno Aido

Integração de facilidades de voz em sistemas CRM

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e de Telecomunicações, realizada sob a orientação científica do Professor Doutor Rui Luís Andrade Aguiar, Professor Associado com Agregação do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president	**Professor Doutor José Rodrigues Ferreira da Rocha** Professor Catedrático do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro
vogais / examiners committee	**Professora Doutora Marília Pascoal Curado** Professora Auxiliar da Faculdade de Ciências e Tecnologia da Universidade de Coimbra
vogais / examiners committee	**Professor Doutor Rui Luís Andrade Aguiar** Professor Associado com Agregação do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

agradecimentos /
acknowledgements

Um profundo agradecimento ao Professor Doutor Rui Luís Andrade Aguiar, por ter aceite orientar-me neste projeto, e pela disponibilidade demonstrada. Mesmo com a agenda sempre muito preenchida, conseguiu ter o tempo necessário para me orientar, e indicar o melhor caminho a seguir para levar este projeto a bom porto.

Também um sincero agradecimento à empresa VLM Consultores, por me ter permitido inserir esta dissertação em âmbito de estágio profissional. Em especial, agradeço ao Pedro Ribeiro, orientador do estágio, pelos ensinamentos facultados nestas matérias.

Por fim, mas não menos importante, um agradecimento muito especial à Daniela, minha companheira, por todo o suporte emocional, pela força transmitida, motivação e apoio em todos os momentos, e também à minha família, essencialmente aos meus pais, pelo esforço e espírito de sacrifício que me permitiu iniciar este percurso académico.

Muito Obrigado!

Resumo

Num contexto de forte competitividade a nível empresarial, em que cada vez mais é necessário procurar relacionamentos comerciais distintos, e personalizados, por forma a aumentar a taxa de fidelização de clientes, nota-se também uma grande evolução das tecnologias de informação e comunicação, disponibilizando à sociedade diversos meios tecnológicos para auxiliar na gestão da informação, quer financeira, quer comercial. É de grande interesse para as empresas, tirarem o máximo proveito deste fator tecnológico, por forma a tornarem-se mais organizadas e eficientes, possibilitando um atendimento mais personalizado e profissional aos seus clientes, também no sentido de melhorar a capacidade de resposta às solicitações.

Nesta perspetiva de orientação para o cliente, e de aumento do seu nível de satisfação, surgem os sistemas de CRM (*Client Relationship Management*), que são plataformas informáticas de gestão de relacionamento comercial, onde são agregadas todas as informações relativas à atividade da empresa, não só a nível de clientes, mas também a nível de fornecedores, parceiros de negócio, colaboradores, potenciais negócios, evolução do estado negocial, etc. Esta agregação permite uma informação consistente, sempre atualizada e disponível da mesma forma para todos os utilizadores, permitindo uma melhoria organizacional significativa, e também uma melhor resposta ao cliente.

Por outro lado, aliado a este conceito de orientação para o cliente, também é extremamente importante que a comunicação seja fluída em ambos os sentidos, quer da empresa para o cliente (e vice-versa), quer a nível de comunicação interna entre os colaboradores da empresa. Neste sentido, implementar uma rede de comunicações é também fulcral para esta fluidez de informação. Com o aumento da largura de banda disponível, tem vindo a aumentar a utilização das tecnologias sobre IP, e a telefonia acompanhou esta tendência, sendo muito fácil e habitual digitalizar a voz para transmitir na rede IP, estabelecendo chamadas de voz (e vídeo) sobre a rede, o que leva a que a largura de banda seja rentabilizada.

À partida, este tipo de conceitos pode ser interpretado como sendo muito caro e de difícil implementação, estando neste caso fora do alcance das PME's, essencialmente. No entanto, este facto não tem de ser obrigatoriamente verdade... Este trabalho pretende demonstrar o potencial de ferramentas open-source neste tipo de sistemas, que podem ser de grande utilidade, uma vez que são soluções customizáveis à medida das necessidades, e apresentam custos de implementação muito baixos, por não implicarem o pagamento de licenças. Em concreto, será abordada a plataforma de relacionamento comercial SugarCRM, e também o sistema de telefonia Asterisk. Integrando estas duas tecnologias, é possível criar uma aplicação de Click-2-Call no SugarCRM, aplicável num ambiente empresarial de pequena/média dimensão.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Num estado social de extrema competitividade no âmbito empresarial, é fundamental para as empresas guiarem-se por um modelo de gestão eficaz, não só a nível financeiro, mas também a nível de recursos, quer humanos, quer materiais. Para além disso, é também necessário que o relacionamento com os clientes seja distinto e personalizado, por forma a marcar a diferença num mercado tão exigente e dinâmico como se vive atualmente, em que o preço do artigo predomina normalmente sobre todos os outros aspetos, no momento da decisão de compra. Para contrariar esta tendência, é fundamental criar laços de fidelização com os clientes, e este facto só é possível de ocorrer quando se dá uma resposta rápida e eficaz, a toda e qualquer solicitação do cliente. É neste contexto que os sistemas de CRM se revelam de extrema importância e utilidade, por permitirem o registo de toda a informação relativa à atividade comercial da empresa, desde dados de clientes, fornecedores, e parceiros de negócio, propostas em curso, oportunidades de negócio, etc. Esta agregação de todos os dados na mesma plataforma, que permite manter a informação sempre atualizada e acessível por todos os colaboradores da empresa, é um fator muito importante no conhecimento do cliente e na rapidez de resposta.

Simultaneamente, com a evolução das tecnologias disponíveis, torna-se quase obrigatório que qualquer pessoa esteja disponível para contacto em qualquer momento, quer seja por telemóvel, e-mail, telefone, redes sociais, ou outro meio. Em ambiente empresarial, esta necessidade de disponibilidade total para receber contactos, pode ser fulcral para ganhar ou perder uma oportunidade de negócio. Este fator corroba o descrito no parágrafo

anterior, sobre um acompanhamento distinto, personalizado, e com boa capacidade de resposta ao cliente. Neste sentido, manter sempre um canal de comunicação aberto, e promover também o contacto internamente, é fundamental para o sucesso. Atualmente, devido a este facto, a maioria das empresas tem nas suas instalações uma rede de telefones, com extensões internas que permitem a comunicação instantânea entre os colaboradores, e também de e para a rede externa. A acompanhar esta tendência, há ainda o facto de que a largura de banda disponível para ligações IP tem vindo a aumentar consideravelmente. Como resultado, começa a ser visível o efeito da digitalização de dados, para facilitar a sua transmissão, receção, e processamento, e o serviço de telefonia não foge à regra. É neste sentido que surgem sistemas de telefonia por IP, tornando possível realizar uma chamada telefónica (com ou sem vídeo) a partir do computador (ou outro dispositivo IP). É aqui que surgem os telefones IP, que permitem uma evolução tecnológica, mas mantêm o interface de hardware (são ou podem ser fisicamente muito semelhantes aos telefones analógicos). Para tirar máximo partido desta tecnologia, existem sistemas VoIP (Voice over Internet Protocol), que fazem a gestão de uma rede de telefonia IP, por exemplo. A grande vantagem é que o custo das chamadas através da ligação de dados é significativamente inferior, principalmente a nível de chamadas internacionais, permitindo às empresas uma grande poupança.

Este trabalho surge como um estudo destas duas tecnologias (CRM e Telefonia IP), e uma possível integração das duas, tendo como grande objetivo a criação de uma aplicação de click-2-call que irá permitir a integração de voz no sistema CRM, facilitando o trabalho de todos os colaboradores que diariamente têm a necessidade de interagir com a plataforma CRM, e simultaneamente receber e efetuar chamadas. O princípio é o seguinte: Se um utilizador está ligado na plataforma e precisa de contactar um cliente, porquê ter de visualizar o número no CRM, e a seguir digitá-lo no telefone para fazer a chamada? Esta situação pode, ocasionalmente, gerar chamadas para números não pretendidos, o que para além de ser desperdício de dinheiro, é desperdício de tempo, e ambos estes recursos são preciosos. Se o utilizador está a ver o número para onde quer ligar, porque não clicar nele, e estabelecer-se automaticamente a chamada?

Irei então, ao longo deste trabalho, apresentar estes temas (CRM e telefonia digital), analisando algumas das soluções existentes no mercado, e

explicar as dificuldades com que me deparei, apesar dos testes realizados serem bastante positivos, e da aplicação criada ter concretizado o objetivo pretendido, que era o estabelecimento da chamada telefónica a partir do click. Um grande motivo de satisfação é o facto de esta aplicação ser futuramente implementada num caso prático.

1.1 Enquadramento

Este trabalho é realizado no âmbito no âmbito de dissertação de Mestrado em Engenharia Eletrónica e de Telecomunicações, e inserido num estágio profissional na empresa VLM Consultores. Tem como propósito apresentar um caso prático de aproveitamento de tecnologias open-source nas PME's, que atravessam um período conturbado nesta época de crise económica, pelo que tentam todos os meios para rentabilizar ao máximo os seus recursos humanos e materiais. É também com esta visão de crescimento sustentado e de sucesso das organizações, que a VLM presta os seus serviços.

A empresa VLM Consultores é uma empresa criada em 1995, e sediada em Aveiro, que foca a sua atividade em quatro grandes setores:

- Economia & Finanças

- Eficiência Organizacional

- Tecnologias de Informação

- Capital Humano

O estágio que permitiu a realização desta dissertação está, naturalmente, inserido nas Tecnologias de Informação, sendo que, neste setor, a VLM presta apoio na implementação de sistemas de gestão aos seus clientes, com recurso a uma plataforma de CRM customizada com vista ao objetivo principal, implementando todas as customizações à medida de cada cliente, e prestando suporte informático à mesma.

Na junção de duas tecnologias já adoptadas pela empresa VLM Consultores (SugarCRM e Asterisk), foi-me lançado o desafio de criar uma aplicação click-to-call, por forma a despoletar uma chamada telefónica para um contacto, a partir da plataforma SugarCRM.

É deste desafio que nasce o conceito desta dissertação.

1.2 Estrutura da Dissertação

Ao longo desta Dissertação, o leitor será levado a uma introdução histórica e técnica, sobre cada uma das plataformas utilizadas, e de seguida, à integração de ambas, dando origem a uma solução integrada de CRM com telefonia digital, aplicável a um ambiente empresarial de pequena/média dimensão.

Faz-se aqui a descrição da estrutura deste trabalho, e um breve resumo do conteúdo de cada um dos capítulos.

Capítulo 1 - Capítulo que contém esta mesma descrição, onde é feita um introdução aos temas que serão abordados, e um enquadramento do trabalho, no que respeita à motivação, âmbito em que foi criado, objetivo, e metas atingidas.

Capítulo 2 - Introdução de alguns conceitos em termos das tecnologias que serão alvo de estudo neste trabalho, nomeadamente, sistemas CRM, sistemas VoIP, protocolo SIP, e servidores.

Capítulo 3 - Introdução e aspetos técnicos mais relevantes das plataformas de referência, neste caso, SugarCRM e Asterisk, que foram os sistemas escolhidos para a concretização deste trabalho. As razões destas escolhas são também descritas neste capítulo.

Capítulo 4 - Explicação da integração de voz no SugarCRM, com recurso à possibilidade de customização em ambas as plataformas. É aqui também descrito o procedimento para a criação de um módulo instalável, que ativa de imediato a funcionalidade construída.

Capítulo 5 - Apresentação, análise e discussão dos resultados obtidos, comprovando o sucesso da integração de voz no SugarCRM através do Asterisk. Informação sobre os passos seguintes para dar continuidade a este projeto.

Capítulo 6 - São apresentadas as conclusões que podem ser retiradas da execução deste trabalho, e da utilização das tecnologias de CRM e Telefonia Digital.

Capítulo 2

Aspetos tecnológicos

Para podermos compreender do que trata esta integração de CRM com Telefonia Digital, é fundamental saber em que consiste o conceito de CRM, e de Telefonia Digital.

Muito resumidamente, o conceito de CRM vem do termo inglês *Client Relationship Management* e é de certa forma, auto-explicativo. Trata-se de Gestão de Relaçionamento com o Cliente, referindo-se a todos os dados comerciais relativos à atividade da empresa, desde os dados de clientes, colaboradores e fornecedores, até oportunidades e potenciais negócios que a empresa esteja a levar a cabo, bem como a evolução do seu estado negocial.

Por outro lado, o conceito de Telefonia Digital, ou telefonia IP, está associado ao conceito de VoIP (*Voice over Internet Protocol,*) no sentido de utilizar a rede de Internet para estabeler chamadas telefónicas internas e externas.

2.1 CRM

2.1.1 Definição

É um conceito que define toda uma classe de ferramentas que automatizam o contacto com o cliente. Essas ferramentas implicam normalmente (mas não obrigatoriamente) sistemas informatizados e fundamentalmente, implicam uma mudança de atitude empresarial, tendo por objetivo ajudar as empresas a criar e manter um bom relacionamento com os clientes, armazenando e relacionando de forma perspicaz, informações sobre as atividades e interações com a empresa.

O CRM é uma abordagem que coloca o cliente no centro dos processos do negócio, sendo idealizado para perceber e antecipar as necessidades dos clientes, por forma a procurar superá-las. Trata-se, sem dúvida, de uma estratégia de negócio, que posteriormente se adapta a uma solução tecnológica. [2]É um sistema integrado de gestão com foco no cliente, constituído por um conjunto de procedimentos/processos organizados e integrados num modelo de gestão de negócios. Os softwares que auxiliam e apoiam esta gestão são normalmente designados de sistemas de CRM, ou plataformas de CRM.

2.1.2 Visão

Os processos de gestão que assentam em CRM's estão, sem dúvida, na linha da frente em termos estratégicos, não apenas em termos de marketing, mas também, a médio prazo, a nível económico. Efetivamente, as empresas que conhecem detalhadamente os seus clientes (as suas necessidades, perfil de consumidor em que se enquadra, etc), conseguem criar respostas personalizadas, antecipando o mercado, e respondendo de forma precisa às reais necessidades de cada um.

A tecnologia responderá apenas à estratégia da empresa a este nível, auxiliando no registo de dados acerca do cliente e na consolidação de uma plataforma central, de modo a tornar a estratégia global de CRM mais inteligente.[2] Adicionalmente, poderá integrar o marketing e as tecnologias de informação já existentes, por forma a dotar a empresa de meios eficazes e integrados para o atendimento ao cliente, tornando possível dar resposta ao cliente em tempo real. As aplicações de CRM transformam os dados recolhidos em informação que, quando analisada com algum detalhe, permite a identificação do cliente e a compreensão do seu perfil.

2.1.3 Vantagens de um CRM[2]

As vantagens de implementar um sistema de CRM são, acima de tudo, económicas, permitindo:

- Aumentar os lucros da empresa (margem em cada cliente);

- Aumentar a taxa de fidelização dos clientes (que custa 5 vezes menos do que conquistar novos, segundo estudos efetuados);

- Economizar tempo graças à automatização de certas tarefas, aumentando a produtividade;

- Otimizar a colaboração entre os diversos serviços da empresa (comercial, marketing, serviço pós-venda)

- Melhorar a reação face a um problema específico (ex : diminuição dos volumes de venda).

2.1.4 Oferta de mercado em sistemas de CRM

Tal como em qualquer outro tipo de mercado, também a nível de sistemas CRM existe uma grande quantidade de ofertas. Serão aqui apresentadas algumas das soluções disponíveis no mercado, que variam entre si em diversos aspetos, nomeadamente a nível de funcionalidades e preço, pelo que é necessário considerar qual delas se adequa melhor ao pretendido, antes de se proceder à implementação.

Refira-se que habitualmente, os serviços de CRM são fornecidos em forma de acesso na cloud, e são taxados mensalmente, em função do número de utilizadores. Estes pagamentos mensais (por vezes são cobrados anualmente), incluem normalmente serviços de suporte técnico e informático, para além do alojamento da plataforma, por forma a estar disponível a partir de qualquer local, e em todos os dispositivos, uma vez que são sistemas web-based. No entanto, há também no mercado opções de CRM interno, ou seja, é vendida a licença e o software é instalado no servidor próprio da empresa cliente. Alternativamente, há ainda sistemas CRM open-source, que permitem aceder ao código, alterá-lo, e redistribuí-lo. Não incluem, naturalmente, as funcionalidades mais avançadas, mas considerando que não existe investimento nem restrições, deverá sempre ser uma hipótese a considerar, dependendo do objetivo que se pretenda. Em cada uma destas variantes, destaco os principais produtos disponíveis no mercado:

- **CRM na cloud**

    i) SalesForceCRM[3]- Um CRM direccionado para os comerciais, que enfatiza a mobilidade, por estar acessível a partir de qualquer local com acesso à web. Permite ao utilizador gerir os potenciais negócios, e o seu estado atual, informações sobre os clientes, e

também monitorizar, por exemplo, se os comerciais estão a cumprir os objetivos definidas pela empresa.

ii) ZohoCRM[4]- Destaca-se como principal funcionalidade deste CRM, o facto de ser possível integrar o sistema com o site da empresa. Assim, os dados introduzidos no site pelos utilizadores, podem ficar automaticamente registados no ZohoCRM, deixando sujeitos para aprovação todos os potenciais negócios que surjam por este meio, e atribuíndo de imediato a tarefa de iniciar o processo negocial, a um determinado utilizador pré-definido.

iii) SugarCRM[5]- CRM direccionado para o bom conhecimento do cliente, e maturação do relacionamento comercial, com a possibilidade de análise do histórico comercial.

iv) WebCRM[6]- É um sistema CRM flexível e simples online, com todas as ferramentas mais importantes para empresas que querem melhorar as vendas através de uma maior visão geral dos processos e de uma melhor perceção das operações.

- **CRM interno** Nesta variante de sistemas CRM, há ainda duas sub variantes, sendo elas os sistemas proprietários, e os sistemas open-source. Na variante CRM na cloud, só estão disponíveis, como seria expectável, sistemas proprietários.

  (I) **Sistemas Proprietários:**

  i) Microsoft Dynamics CRM[7]- É o software CRM proprietário da Micrsosoft. O Microsoft Dynamics CRM ajuda a reduzir os custos e a aumentar a rentabilidade da organização mediante a automatização de processos de negócio que fomentam a satisfação e fidelização dos clientes. Oferece um conhecimento real de cada cliente para que os colaboradores, em contacto direto com os mesmos, tomem decisões rápidas e informadas sobre estratégias de venda, marketing e suporte ao cliente. Não foi possível obter informação de preço deste produto, pois a informação que a Microsoft disponibiliza é que "As licenças do Microsoft Dynamics CRM 4.0 não incluem licenças para outros produtos que podem ser necessários em sua instalação; as licenças destes produtos adicionais

devem ser adquiridas separadamente", nomeadamente as licenças de base de dados, que terá de ser obrigatoriamente MicrosoftSQL, e de sistema operativo, Microsoft Windows, etc. Apesar de estar enquadrada na categoria de CRM interno, a Microsoft disponibiliza ainda a opção de colocar o CRM na cloud, apesar de ser alojada por um parceiro da Microsoft.

ii) SiebelCRM[8]- Sistema de CRM da Oracle, destinado a ajudar as empresas a diferenciarem os seus negócios para maximizar de lucro. Poderá ter soluções integradas, tambem da Oracle, naturalmente, em termos de cotações e encomendas, cruzando informações por forma a sugerir automaticamente um determinado produto ao cliente, com base no seu perfil de consumo.

Também para este sistema não existe informação disponível sobre o preço praticado.

(II) **Sistemas open-source:** A grande vantagem deste tipo de sistemas serem disponibilizados em open-source, é o facto de ser possível alterar o código, ajustando o sistema às reais necessidades, sem contratos nem licenciamentos com nenhuma entidade. São exemplos nesta categoria, os seguintes produtos:

i) SugarCRM[5]- Apesar de ter também versões comerciais, é disponilizada a versão CE (Community Edition), em open-source, versão que serve de base às restantes, e que recebe contribuições da comunidade de *developers* voluntários espalhados pelo globo.

ii) SuiteCRM[9]- O SuiteCRM é um sistema muito recente, que surgiu como uma derivação do SugarCRM, devido ao descontentamento sobre a postura da SugarCRM (empresa) para com a comunidade, que passou a focar-se mais em desenvolver as versões comerciais, sem disponibilizar de regresso para a comunidade. Por este facto, o SuiteCRM torna-se uma grande ameaça à popularidade, não só do SugarCRM, como de outros sistemas proprietários, pois os seus fundadores tentam equiparar, em termos de funcionalidades, o SuiteCRM com as versões comerciais do Sugar. Também por isso, as

três versões do SuiteCRM são gratuitas e open-source.

iii) ZurmoCRM[10]- Assumem claramente, que o seu principal objetivo é facilitar na implementação de CRM nas empresas, por forma a que qualquer pessoa seja capaz de concretizar a implementação, em qualquer tipo de contexto. Têm também uma versão comercial, mas focam-se essencialmente no open-source.

Realço nesta fase, que para este trabalho, foi utilizada a versão CE (Community Edition) do SugarCRM, por ser a plataforma que serve de base à atividade da VLM há já vários anos, sendo que era para esta plataforma que se pretendia criar a aplicação de click-2-call. No entanto, o mesmo procedimento poderia ser aplicado no suiteCRM, por este ser precisamente uma derivação do Sugar, e portanto, totalmente compatível. Para outras plataformas, a solução criada seria também aplicável, embora fosse necessário proceder a algumas alterações, nomeadamente em termos de estrutura.

Por este facto, a plataforma SugarCRM será abordada com mais detalhe no próximo capítulo (pág. 37)

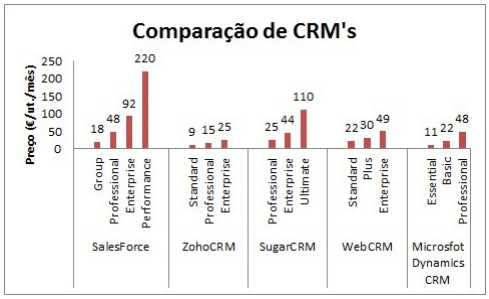


Figura 2.1: Edições e preços dos principais fornecedores de sistemas CRM[1]

Um estudo realizado em 2013 com 400 empresas nos diversos sectores de atividade, nos Estados Unidos, revelou os dados apresentados na figura 2.2, relativamente ao uso de sistemas CRM:

2.1.5 Implementação de um CRM

As plataformas de CRM alicerçam-se em processos centrados no cliente, disseminados por toda a organização. Verifica-se uma utilização exaustiva de informação relacionada com o cliente, integrando as áreas de marketing, vendas e serviços, verificando-se a criação de valor para o cliente. Antes

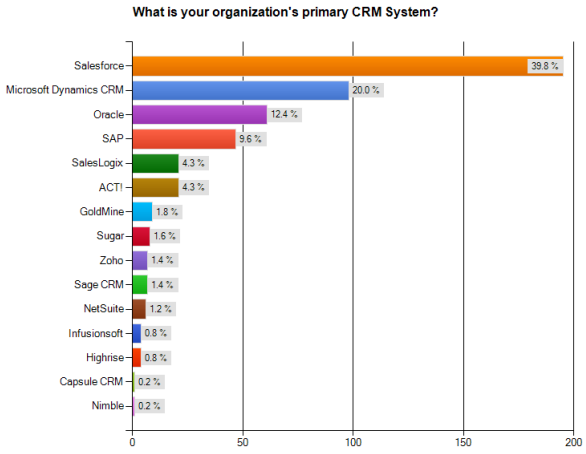


Figura 2.2: Utilização de sistemas CRM nos E.U.A.[1]

de implementar, importa perceber qual o modelo de relacionamento com o cliente que a empresa pretende adotar, sendo necessário, várias vezes, redesenhar os processos de atendimento. Aqui importa perceber dimensões como:

- Como será feita a abordagem ao cliente?
- Que procedimentos ou eventos devem ser gerados?
- Qual o plano de comunicação a adotar?

É necessário ter resposta a este tipo de questões, para se poder iniciar uma implementação de um sistema de CRM, pois é precisamente nestes pontos que o sistema se tornará de grande relevância, e terá de ser otimizado nesse sentido.[2] Para responder a estes desafios, procede-se ao levantamento rigoroso dos processos existentes, podendo ser necessário redesenhar completamente, ou apenas reajustar os mesmos e eventualmente, adicionar mais-valias, pelo fato de passar a existir suporte de tecnologia de informação orientada para o cliente.

A partir daqui é selecionada a solução, e procede-se à implementação. A seleção é baseada nas fases anteriores, sendo validadas as características das soluções disponíveis, determinada pelo modelo de relacionamento a seguir no futuro.

Ao nível da implementação do sistema, os passos passam por configurar o modelo de relacionamento na tecnologia adquirida e implementar a estratégia

de relacionamento com o cliente, incluindo um conjunto de ferramentas de apoio, como marketing, redes sociais, canais de comunicação.

Importa nesta altura, ter em conta os aspectos que contribuem para o sucesso desta fase:

- Foco nos processos, não na tecnologia. As TIC são apenas um meio para alcançar os objetivos;
- Envolvimento, dedicação e acompanhamento da gestão de topo;
- Seleção da tecnologia de acordo com o negócio;
- A tecnologia deve preencher necessidades específicas do negócio;
- Perceber como as funcionalidades são realizadas e compreender a arquitetura global do sistema;
- Prestar apoio e formação aos utilizadores;
- Competência multidisciplinar da equipa de projeto;
- Estabelecimento de etapas e prioridades;
- Consistência da integração do sistema.

Em termos organizacionais, pretende-se então que a implementação de um sistema CRM se torne uma mais-valia nos seguintes aspetos:

Vendas Permitindo prever e antecipar o mercado, e também pelo modo off-line, que permite a sincronização com o sistema central a posteriori;

Serviço ao cliente Pelo registo e acompanhamento de questões, problemas, reclamações, sugestões, e pedidos de informação. Estes factos permitem que o serviço ao cliente se torne mais organizado e eficiente;

Marketing Pela capacidade de registo e cruzamento de dados em base de dados, passa a ser mais fácil fazer análises de mercado, permitindo descobrir relações não antecipadas e padrões de consumo e comportamento;

Coerência e interoperacionalidade Base de dados de clientes e eventos de negócio, que todas as aplicações da plataforma utilizam e mantêm atualizada.

Melhor gestão dos recursos existentes Permite uma poupança de tempo significativa em termos de recursos humanos e de máquinas que até então estariam a ser utilizados para processamento de dados manual. E ainda na transmissão de informação, que deixa de ser necessária, visto estar constantemente disponível e atualizada.

Através destes sistemas são passíveis de observação os seguintes processos:

i) Planeamento - Verifica-se um plano de atividades rigoroso, com identificação de todos os pontos críticos de relacionamento e a estrutura do fluxo de trabalho; há um levantamento de todos os pontos de automação

ii) Marketing relacional - Identificar, segmentar, interagir, configurar através de programas de fidelização, com identificação clara e construção de perfis do consumidor.

iii) Produtos e serviços de extrema qualidade que a concorrência a nível global obriga. De facto, a tolerância ao erro, à falta de qualidade e à ineficiência é cada vez menor, verificando-se uma ênfase na gestão do conhecimento.

iv) Acrescento de valor ao relacionamento - o conhecimento obtido deve orientar o relacionamento. Assim quem compra deve receber exatamente o que deseja comprar e a informação que realmente lhe interessa.

v) Integração de outros canais de comunicação - telefone, fax, contato pessoal, carta, etc. Os dados recolhidos por este meio devem ser inseridos numa base de dados global.

vi) Detecção de Oportunidades de Negócio - através da análise dos dados, levando assim ao reconhecimento de padrões de comportamento econômico e de relacionamento, o que por sua vez poderá sugerir novas formas de criar negócios.

O CRM abrange, portanto, três grandes áreas:

I) Automatização da gestão de marketing;

II) Automatização da gestão comercial, dos canais e da força de venda;

III) Gestão dos serviços ao cliente;

Os processos e sistemas de gestão de relacionamento com o cliente permitem que se tenha controlo e conhecimento das informações sobre os clientes de maneira integrada, principalmente através do acompanhamento e registo de todas as interações com o cliente, que podem ser consultadas e comunicadas a diversas partes da empresa que necessitem desta informação para guiar as tomadas de decisões.

Uma das atividades da Gestão do Relacionamento com o cliente implica registar os contatos realizados, de forma centralizada. Os registos não dependem do canal de comunicação utilizado (voz, fax, e-mail, chat, SMS, MMS etc) e servem para que se tenham informações úteis e catalogáveis sobre os clientes. Qualquer informação relevante para as tomadas de decisões pode ser registada e analisada periodicamente, de forma a produzir relatórios de gestão.

2.1.6 Variantes de CRM

Consoante o objetivo principal da implementação de um sistema de CRM, este pode ser categorizado em quatro tipos[2]:

CRM Operacional : é a aplicação da tecnologia de informação para melhorar a eficiência do relacionamento entre os clientes e a empresa. Prevê a integração de todos os produtos de tecnologia para proporcionar um melhor atendimento ao cliente;

CRM Colaborativo : é a aplicação da tecnologia de informação que permite a automação e a integração entre todos os pontos de contacto do cliente com a empresa. Estes pontos de contacto devem estar preparados para interagir com o cliente e disseminar as informações levantadas para os sistemas do CRM operacional;

CRM Analítico : componente do CRM que permite identificar e acompanhar diferentes tipos de clientes dentro da carteira de uma empresa e na posse destas informações, determinar qual a estratégia a seguir para atender às diferentes necessidades dos clientes identificados. Normalmente utiliza recursos de cruzamento de dados para localizar padrões de diferenciação entre os clientes.

CRM Social : é a forma de interagir com o cliente por meio das redes sociais, e ainda de enriquecer os dados e informações sobre o cliente

com base nas informações encontradas nos respetivos perfis.

2.1.7 CRM e o Ensino

O setor do Ensino não é exceção à utilização de CRM, por ser um canal de relacionamento entre as instituições e ao mesmo tempo com a sua comunidade estudantil e docente.

Cada vez mais, existem formas de divulgação através de Web, das Instituições de Ensino Superior, para a captação de clientes (alunos), através do uso de ferramentas direcionadas apresentadas nos diversos cursos existentes.

Cada vez mais o CRM é usado como forma de ligação com o cliente em grande parte das empresas, e também no Ensino Superior. Os processos e sistemas de gestão de relacionamentos para com o cliente, exigem que se tenha um controle e informação precisa das novidades sobre os clientes de maneira integrada, que podem ser consultadas e participadas a distintas partes da empresa que precisem desta informação para acompanhar as tomadas de decisões.

Logo no Ensino Superior, através dos registos efetuados pelos alunos nas matriculas, é possível ter acesso à comunidade universitária, e divulgar através, por exemplo, de e-mail, quais as novidades existentes em cursos, estágios, oportunidades de carreira, etc, de forma a potenciar, não só novos candidatos, como também, novas inscrições de alunos já matriculados.

2.1.8 CRM e a Informática

Por vezes o CRM é entendido única e exclusivamente como o sistema de computador desenvolvido para a gestão de clientes, ou mesmo como um sistema de venda ainda mais simplificado[2].

Na realidade o CRM é apenas o conceito conforme descrito anteriormente, e os sistemas de informática são as ferramentas que auxiliam na gestão do relacionamento com clientes, são os chamados sistemas de CRM. Seria possível implementar o conceito de CRM numa empresa, sem o recurso à informática, mas no entanto, seria muito mais exigente e mais difícil a sua manutenção. Tal como já foi referido anteriormente, as Tecnologias, são apenas um meio para alcançar os objetivos propostos pelo CRM. É sem dúvida um enorme suporte auxiliar, e uma preciosa ajuda para manter a empresa bem organizada e focada no cliente.

Origem dos sistemas CRM

A origem dos sistemas de CRM remonta ao início dos sistemas de informação voltados exclusivamente a vendas, com os primeiros sistemas de vendas ou gestão de vendas, e mais tarde sendo aperfeiçoados para gestão de clientes e do seu relacionamento com a empresa. [2] Com a melhoria da capacidade de processamento e armazenamento dos equipamentos, somada com as necessidades criadas pela concorrência cada vez maior no mercado, os sistemas foram sendo aperfeiçoados, tornando-se efetivamente sistemas com suporte a aplicações de CRM e gestão de relacionamentos comerciais.

2.2 Telefonia IP

2.2.1 Definição

O conceito de telefonia IP está relacionado com a Voz sobre IP, ou VoIP (Voice over Internet Protocol). É o estabalecimento de chamadas telefónicas usando a Internet ou qualquer outra rede de computadores baseada no Protocolo de Internet, tornando a transmissão de voz mais um dos serviços suportados pela rede de dados.

A telefonia IP utiliza uma rede de dados IP para fornecer comunicações de voz a toda a empresa, o que significa que não é necessário um PBX e rede de voz separados. Contudo, a telefonia IP permite a uma organização migrar da sua rede PBX existente para a telefonia IP sem interrupções para os utilizadores. A telefonia IP é um componente chave da Arquitectura para Vídeo, Voz e Integração de Dados. Esta convergência de serviços de comunicação de dados, voz e vídeo numa única rede traz consigo as vantagens de custos mais baixos, procedimentos de suporte ou configuração simplificados e maior integração de locais remotos e escritórios nas instalações de rede da empresa.

Uma das principais vantagens da convergência assenta nas aplicações que passam a estar disponíveis para os utilizadores. Por exemplo, a disponibilização de serviços de lista telefónica e web directamente num telefone; mensagem unificada (unified messaging) que significa que se pode aceder aos e-mails através de um telefone ou às mensagens de voz através do e-mail; a implementação de um centro de contactos completo em que os clientes podem contactar uma organização através da web, do telefone, e-mail, fax,

formulários da web, etc. O cliente beneficia com a possibilidade de escolher um método de comunicação com maiores níveis de serviço enquanto que o centro de contactos beneficia de custos mais baixos, e melhor performance dos agentes.[11]

Esta tecnologia permite uma grande redução de custos devido ao uso de uma única rede para carregar dados e voz, especialmente quando os utilizadores já possuem uma rede com capacidade subutilizada, que pode transportar dados VoIP sem custo adicional. Chamadas de VoIP para VoIP são geralmente gratuitas, enquanto chamadas VoIP para redes públicas (PSTN) podem ter custo para o utilizador VoIP, embora que mais reduzidos, comparados com o custo da rede pública.

Para integar um sistema de telefonia IP com a rede telefónica pública, é necessário o fornecimento de serviços VoIP por parte de empresas de telecomunicações, sendo que as chamadas são realizadas pelo método chamado *SIP Trunk[12]*, caracterizado pelas seguintes grandes vantagens[13]

- Os serviços de voz SIP Trunk têm numeração telefónica nacional associada, o que permite ao cliente receber chamadas de outros operadores de telecomunicações, quer sejam nacionais ou internacionais;

- Pode manter a numeração telefónica e as funcionalidades do serviço de voz;

- O serviço com numeração nómada pode ser utilizado em qualquer parte do Mundo, desde que o cliente tenha conectividade de dados, permitido ao mesmo estar contactável com custos equivalentes a como se estivesse em Portugal;

- O serviço SIP Trunk permite usufruir de tarifários bastante competitivos sem necessidade de alterar o contrato com o actual operador de telecomunicações;

- O serviço é bastante flexível em termos de contratação de número de canais externos em simultâneo (depende somente dos equipamentos utilizados e da largura de banda da ligação de dados) e de numeração telefónica.

2.2.2 Finalidade

O VoIP pode facilitar tarefas difíceis em redes tradicionais. Chamadas recebidas podem ser automaticamente encaminhadas para o telefone VoIP, independentemente da localização na rede. Por exemplo, é possível levar um telefone VoIP para uma viagem, e onde este tiver ligação à Internet pode receber ligações, assumindo que a ligação seja rápida e estável o suficiente. O facto da tecnologia estar diretamente ligada à Internet também traz a vantagem de poder integrar telefones VoIP a outros serviços como vídeo-chamadas, chat, partilha de ficheiros. Estar ligado à Internet também significa que o custo da chamada é independente da localização geográfica e do horário de utilização, ambos os parâmetros normalmente usados para taxação fixa e móvel, e cujos valores variam consoante a operadora.

Vários pacotes de serviço VoIP incluem funcionalidades que em redes tradicionais seriam cobradas à parte, como conferência a três, reencaminhamento de chamadas, e identificador de chamadas.[11]

No fundo, o VoIP uniformiza as chamadas telefónicas e os restantes dados transmitidos na rede IP, digitalizando a voz em pacotes de dados que são transmitidos e recebidos.

2.2.3 Funcionamento

O princípio base de funcionamento da tecnologia VoIP consiste em digitalizar a voz em pacotes de dados para que navegue pela rede IP e converter novamente estes pacotes em voz no destino. Explica-se de seguida o processo para o estabelecimento de uma chamada: O utilizador levanta o telefone, e nesse momento é emitido um sinal para a aplicação, a sinalizar "telefone levantado". A parte de aplicação emite um sinal de digitação. O utilizador digita o número de destino, cujos dígitos são acumulados e armazenados pela aplicação. Os gateways comparam os dígitos acumulados com os números programados; quando há uma coincidência , mapeia o endereço digitado com o IP do gateway de destino. A aplicação inicia o protocolo de sessão sobre IP (SIP), para estabelecer um canal de transmissão e recepção em ambos os sentidos através da rede IP. [11]Se a ligação for concretizada por um PBX, o gateway troca a sinalização digital com o PBX, informando o estado da ligação. Se o número de destino atender a ligação, é estabelecido um fluxo

RTP[1] sobre UDP entre o gateway de origem e destino, iniciando-se então a conversação. Quando qualquer um dos terminais desligar a chamada, a sessão é encerrada.

2.2.4 Protocolos utilizados[11]

a) Transporte

UDP - *User Datagram Protocol*

É o principal protocolo utilizado para transporte dos datagramas, sendo um protocolo simples da camada de transporte que permite que a aplicação escreva um datagrama encapsulado num pacote IPv4 ou IPv6, e seja de seguida enviado ao destino. É também conhecido como sendo um serviço sem conexão, por não haver a necessidade de manter um relacionamento longo entre cliente e o servidor. No entanto, não há qualquer tipo de garantia de que o pacote chegue ou não ao destino, uma vez que este protocolo não utiliza sinais de acknowlegments. Caso sejam necessárias garantias de confirmação de receção, é preciso implementar uma série de estruturas de controle, tais como timeouts, retransmissões, acknowlegments, controle de fluxo, etc. Cada datagrama UDP tem um tamanho fixo e pode ser considerado indivisível, contrariamente ao TCP, que é um protocolo orientado a fluxos de bytes sem início e sem fim. A grande vantagem do UDP é que este Protocolo também fornece os serviços de broadcast e multicast, permitindo que um único cliente envie pacotes para vários outros na rede, sendo por todos estes fatores, o protocolo ideal para envio de dados em tempo real.

RTP - *Real-Time Transport Protocol*

Os pacotes RTP e RTCP (Real-Time Control Protocol) utilizam o UDP como protocolo de transporte, e definem como deve ser fragmentado o áudio, indicando em cada fragmento, informação de sequência e de tempo de entrega.

Serão os pacotes RTP trocados entre os clientes SIP, que irão permitir o bom funcionamento de uma chamada telefónica, por transportar os dados entre eles em tempo real.

b) Sinalização

[1] Real-time Transport Protocol

Alguns dos principais protocolos utilizados para sinalização de chamadas são: SIP, H.323, e IAX-2. Dado que o SIP é o mais comum, mais à frente neste trabalho será discutido com algum detalhe.

2.2.5 Uso Empresarial[11]

Apesar de poucos ambientes de escritório e residências utilizarem uma infra-estrutura puramente de telefonia IP, os fornecedores de serviços de telecomunicações usam esta tecnologia recorrentemente, geralmente numa rede IP dedicada para ligar estações e converter sinais de voz em pacotes IP e vice e versa. O resultado é uma rede digital genérica (tráfego de voz e dados) com escalabilidade. O consumidor empresarial usa a telefonia IP para obter as vantagens da abstração da informação na rede. Com o VoIP é apenas necessário fornecer mais largura de banda, não sendo necessário distribuir uma rede específica para a telefonia no ambiente de trabalho. Empresas maiores também fazem uso de gateways para as redes tradicionais, reduzindo custos de mão de obra externa ao serviço. Esta redução de custos é ainda mais visível quando uma empresa efetua regularmente, chamadas internacionais. Outro tipo de aplicação empresarial deste sistema é a video-conferência com custos reduzidos (em alguns casos, até sem qualquer custo), em que os sistemas envolvidos, sejam eles software cliente ou hardware específico para a aplicação, disponibilizam formas simples para vários utilizadores (colaboradores das empresas) comunicarem entre si sem ser necessário recorrer a grandes centrais telefónicas e/ou sequências complexas de números e símbolos no telefone para darem início a uma sessão. Nas situações de uso do sistema através de software proprietário do fornecedor de serviço VoIP este poderá disponibilizar outro tipo de ferramentas como transferência de ficheiros, partilha de pastas e em alguns casos a partilha do próprio computador.

2.2.6 O Futuro do VoIP[11]

Pelos projetos atuais das empresas que hoje trabalham com VoIP, segundo analistas de mercado e alguns pontos de opinião, uma das próximas etapas na evolução do VoIP é a extinção por completo do modelo atual de ligações de longa distância (DDD/DDI) pela rede PSTN e, mais adiante, talvez a erradicação dos sistemas convencionais de telefonia.

Parte desta evolução estará condicionada à medida que os telefones IP

chegarem aos lares e os acessos em banda larga forem mais abrangentes em termos de população. Neste sentido, vários segmentos trabalham no intuito de criarem redes convergentes, seja utilizando os meios de transmissão telefônica atual, já compartilhado por serviços ADSL, seja partilhando meios de transmissão de serviços de televisão por cabo, entre outros.

O futuro da tecnologia de Voice over Internet Protocol (VoIP) são as comunicações unificadas (UCoIP) (Unified Communication over IP).

2.2.7 Software e Hardware para VoIP

Servidores[14]- também conhecidos por *Sistemas VoIP*, podem ser compreendidos como sendo o *Sistema Operativo* que vai colocar o servidor VoIP a atuar como tal. É nele que serão configurados todos os parâmetros necessários para colocar a rede VoIP em funcionamento. O Asterisk, naturalmente, não é o único sistema VoIP disponível. Analisemos algumas alternativas ao Asterisk, e suas características, na tabela seguinte.

Programa	SO compatíveis	Licença	Protocolos	Encriptação
Asterisk	Linux/BSD Mac OS X Solaris	GPL	SIP H.323 IAX	TLS SRTP
3CX	Windows	Fechado	SIP	TLS SRTP
AS5300	Linux, Win Server '03	Fechado	SIP UNIStim MLPP	SSL TLS SRTP SDESC
Cisco Unified Comm. Mannager	Linux	Fechado	SIP SCCP MGCP H.323	SSL TLS SRTP
Elastix[2]	Linux	GPL	SIP IAX H.323	-

[2]Aplicação baseada em Asterisk

			XMPP	
FreeSwitch	Linux/BSD MAC OS X Solaris Windows	GPL	SIP STUN XMPP IAX H.323 RSS Skype	TLS SRTP ZRTP
GNU Gatekeeper	Linux/BSD MAC OS X Solaris Windows	GPL	H.323	H.235
MediaCore Softswitch	Linux	Fechado	H.323 SIP	SSL TLS HTTPS
Kamailio	Linux/BSD Solaris	GPL	SIP XMPP	-
SIP Express Router	Linux/BSD Solaris	GPL	SIP	-

Tabela 2.1: Sistemas VoIP

Normalmente, os fabricantes deste tipo de soluções apostam no seu próprio software, e combinam a central telefónica com os telefones (normalmente telefones IP), comercializando o pacote completo, sendo que o software PBX é compatível apenas com os seus próprios telefones. São exemplo deste caso, Cisco, Alcatel, NEC, Siemens, etc.

Para este trabalho, foi escolhido o sistema Asterisk porque a solução de telefonia VoIP que a VLM utiliza é uma adaptação do sistema Asterisk. Concretamente, é uma central telefónica EdgeBox que utiliza este software, pelo que não seria de esperar que fosse utilizado outro sistema, até por uma questão de aproveitamento de todas as configurações do sistema, que não se pretendia que fossem alteradas.

Clientes

I) **SoftPhones[14]** Os Softphones são programas de computador para clientes receberem chamadas de voz e vídeo sobre a rede IP com a funcionalidade básica dos telefones originais, que geralmente permite integração com Telefone IP e Telefone usb em vez de utilizar o microfone e colunas do PC. A maioria de softfones correm no protocolo aberto Protocolo de Iniciação de Sessão (SIP) e suportam vários codecs. O sistema Skype, por exemplo, funciona numa rede fechada proprietária, apesar da rede (não o software de cliente oficial) também suportar clientes SIP. Atualmente, este tipo de programas de "Chat"online, também incorporam comunicações de voz e video. São exemplos de softphones, os seguintes:

Programa	SO compatíveis	Licença	Protocolos	Encriptação
Ekiga	Linux	GPL	SIP H.323 H.263 H.264/MPEG-4	-
Jitsi	Windows Linux Mac OS	LGPL	SIP XMPP	TLS, SRTP
Linphone	Linux Windows Android Iphone Blackberry	GPL	SIP	-
SFL Phone	Linux	GPL3	SIP RTP IAX2 SRV	TLS, SRTP
Zoiper	Linux Windows Mac OS Android	Freeware Poprietário	SIP	TLS SRTP

	Iphone			
Skype	Linux Mac OS Windows Android IPhone	Freeware Proprietário	SIP Proprietário	AES-256 Encr.Skype
3CX	Windows Android Iphone	GPL	SIP	TLS SRTP

Tabela 2.2: Clientes Voip - Softphones

II) **HardPhones** Os Hardware Phones, ou seja, os telefones que estamos habituados a ver em cima da secretária numa empresa. Ligam-se diretamente à rede, e são configuráveis e endereçáveis por IP. Também são conhecidos como telefones VoIP, telefones IP ou telefones SIP, devido a este ser o protocolo mais comum (SIP) neste tipo de soluções. Também por este motivo dedico uma secção deste trabalho exclusivamente à análise deste protocolo. Apresenta-se de seguida a listagem e uma breve descrição dos hardphones mais comuns:

   i) Cisco[15]- Marca de referência em telefones IP. São idealmente concebidos para trabalharem conjuntamente com a central telefónica Cisco, sendo, no entanto, também compatíveis com outros servidores VoIP, pela universalidade do Protocolo SIP. São reconhecidos pela sua ótima qualidade e durabilidade, sendo no entanto, bastante caros (os mais baratos rondam os 100€, chegando os mais caros e com mais funcionalidades a ultrapassar os 400€)

   ii) Alcatel[15]- Telefones concebidos para uma utilização intensiva, com características semelhantes entre si. É uma gama inferior, em relação à Cisco, sendo reconhecidos pela boa relação qualidade/preço. Preços variam entre os 60 e os 130€, consoante o modelo.

iii) YEALINK[16]- Quando o aspeto visual assume menos importância, e se procuram produtos mais económicos, entram em análise os telefones Yealink, conhecidos como a marca branca na telefonia IP. Muito orientados para a funcionalidade, e menos para a componente estética. Preços médios na ordem dos 50€, tendo também disponíveis alguns modelos de gama superior, que chegam a custar 300€.

iv) OUTROS[16]- Existem ainda mais algumas marcas bem colocadas no mercado de telefonia IP, sendo as anteriormente descritas, as mais relevantes, e com maior abrangência. No entanto, neste âmbito, destaco ainda as marcas Siemens, Polycom, NEC, Linksys.

Realço que todas estas marcas possuem modelos de telefones IP sem fios, conhecidos como DECT *(Digital Enhanced Cordless Telecommunications)*, o que se apresenta como um enorme ponto a favor da adoção de telefonia IP, por permitir a mobilidade dos colaboradores dentro da empresa, mantendo a possibilidade de serem contactados. Tal não acontece na telefonia analógica, que implicava que, se por exemplo, um colaborador se ausentasse temporariamente do gabinete, ficaria incontactável durante esse período.

2.3 Protocolo SIP

O Protocolo de Iniciação de Sessão (Session Initiation Protocol - SIP) é um protocolo de aplicação, que utiliza o modelo "pedido-resposta", similar ao HTTP, para iniciar sessões de comunicação interativa entre utilizadores.[17][18]

SIP é um protocolo de sinal para estabelecer chamadas e conferências através de redes via Protocolo IP, utilizado em larga escala em sistemas VoIP. O estabelecimento, mudança ou término da sessão é independente do tipo de dados ou aplicação que será usada na chamada; uma chamada pode utilizar diferentes tipos de dados, incluindo áudio e vídeo.

Este protocolo teve origem em meados da década de 1990, quando o H.323 estava a começar de ser adotado como protocolo padrão, para que fosse possível adicionar ou remover participantes dinamicamente numa sessão multicast. O desenvolvimento do SIP concentrou-se em ter um impacto

tão significativo quanto o protocolo HTTP, a tecnologia por detrás das páginas da web que permite que uma página com links clicáveis conecte com textos, áudio, vídeo e outras páginas da web. Enquanto o HTTP efetua essa integração através de uma página web, o SIP integra diversos conteúdos a sessões de administração. O SIP rapidamente foi adotado como protocolo padrão para comunicações integradas e aplicações que usam presença, no sentido da aplicação estar consciente da sua localização e disponibilidade.

SIP foi moldado e inspirado em outros protocolos de Internet baseados em texto como o SMTP (email) e o HTTP (páginas da web) e foi desenvolvido para estabelecer, mudar e terminar chamadas num ou mais utilizadores numa rede IP de uma maneira totalmente independente do conteúdo de dados da chamada. Como o HTTP, o SIP leva o controlo da aplicação para o terminal, eliminando a necessidade de uma central de comutação.

O protocolo SIP possui, portanto, as seguintes características:

- Simplicidade e possui apenas seis métodos[3].

- Independência do protocolo de transporte.

- Baseado em texto.

2.3.1 Arquitetura do SIP[19]

Os principais componentes da arquitetura do SIP são:

- Agente do Utilizador

  O Agente do Utilizador é o terminal SIP ou o software de estação final. Funciona como um cliente no pedido de inicialização de sessão e também age como um servidor quando responde a um pedido de sessão. Dessa forma, a arquitectura básica é cliente/servidor.

  O Agente do Utilizador tem a capacidade de armazenar e gerir todas as situações de chamada. O Agente do Utilizador faz chamadas com um endereço parecido com o de e-mail ou número de telefone (E.164), como por exemplo *SIP:myphone@ip.serv.er*

  Este facto faz com que URL's SIP sejam fáceis de associar com o endereço do uutilizador. O Agente do Utilizador pode aceitar e receber

[3]Consultar tabela 2.3 na página 29

chamadas de outro Agente do Utilizador sem requerer nenhum componente adicional do SIP. Os componentes restantes fornecem funcionalidades adicionais.

- Servidor Proxy

  Servidor Proxy SIP

  Um tipo de servidor intermediário do SIP é um Servidor Proxy SIP. O Servidor Proxy SIP encaminha pedidos antes do Agente do utilizador para o próximo servidor SIP retendo informações para mapeamentos futuros. Além disso, o servidor proxy SIP pode operar com comunicação stateful (por exemplo, como um circuito, TCP) ou stateless (por exemplo como um UDP). O servidor SIP stateful pode "dividir" chamadas por ordem de chegada, ou seja, há várias extensões que tocam ao mesmo tempo e a primeira a atender ficará com a chamada. Essa capacidade significa que se pode especificar que um telefone de desktop SIP, um telefone celular SIP e aplicações de videoconferência de casa SIP possam sinalizar simultanemente quando estiver a receber uma chamada. Ao atender um dos dispositivos e iniciada a conversação, os restantes param de sinalizar. O servidor proxy SIP pode utilizar múltiplos métodos para tentar resolver o pedido de endereço de host, incluindo busca de DNS, busca em base de dados ou retransmitir o pedido para o "próximo" servidor proxy.

  Servidor de Reencaminhamento SIP

  Um outro tipo de servidor intermediário do SIP é o Servidor de Reencaminhamento SIP. A função do servidor de reencaminhamento SIP é fornecer a resolução de nome e localização do usuário. O servidor de redirecionamento SIP responde ao pedido do Agente do Usuário fornecendo informações sobre o endereço do servidor para que o cliente possa contactar o endereço directamente.

- *Register*

  O SIP *register* fornece um serviço de informação de localizações; recebe informações de localização do Agente do Utilizador e armazena essa informação de registo.

  A arquitectura do SIP utiliza o SDP (Session Description Protocol,) que é uma ferramenta de conferência multicast via IP desenvolvida

para descrever sessões de áudio, vídeo e multimédia. Na realidade, qualquer tipo de MIME (Multipurpose Internet Mail Extension) pode ser caracterizada pela sua capacidade de suportar todos os tipos de anexos em mensagens, tal como o e-mail. A descrição da sessão pode ser usada para negociar uma aceitação de um conjunto de tipos de *media* compatíveis.

Como resultado dessa arquitectura, o endereço do utilizador SIP remoto é sempre o mesmo (por exemplo sip:myphone@ip.serv.er), mas em de estar ligado a um endereço estático, comporta-se como um endereço dinâmico que reflete a localização actual do destinatário. A combinação de Proxy e Servidor Redirecionador dá ao SIP grande flexibilidade de arquitectura; o utilizador pode aplicar vários esquemas simultaneamente para utilizadores localizados e é o que faz a arquitetura do SIP ser bem adaptada para suportar mobilidades. Mesmo quando o utilizador remoto é móvel, o Proxy e o redireccionador podem ser usados para encaminhar o pedido de ligação para o utilizador da locação actual. As sessões podem envolver múltiplos participantes, de forma similar a uma chamada multiponto H.323. Comunicações dentro de uma sessão em grupo podem ser via multicast ou via uma rede de chamadas unicast, ou até mesmo uma combinação dos dois. Um outro resultado da arquitectura do SIP é a sua adequação natural como um ambiente de colaboração devido às suas capacidade de apresentar múltiplos tipos de dados, aplicações, multimédia, etc. com uma ou mais pessoas.

2.3.2 Pacotes SIP[20]

Os pacotes SIP (Session Initiation Protocol) são os pacotes trocados pelas entidades envolvidas numa chamada, onde se encontra a informação sobre a comunicação. São trocados numa lógica pedido-resposta, de forma muito semelhante ao protocolo HTTP.

Tipo de Pedidos:

Tipo de pedido	Descrição
INVITE (convidar)	Estabelece uma sessão
ACK (confirmar)	Confirma o comando CONVIDAR

BYE (terminar)	Finaliza uma sessão
CANCEL (cancelar)	Cancela a sessão ainda não respondida
REGISTER (registo)	Informa a localização do usuário (nome do usuário, IP)
OPTIONS (opções)	Informa a capacidade e disponibilidade dos telefones de chamada e recebimento SIP

Tabela 2.3: Tipos de pedidos no Protocolo SIP

Tipos de resposta:

Código	Categoria	Descrição
1xx	Provisionamento da chamada	Respostas informativas, indicam que o servidor contactado está a realizar alguma tarefa intermédia, e não tem ainda uma resposta definitiva. Este tipo de mensagens é enviado quando o servidor estima que durará mais de 200ms até ter uma resposta definitiva. Nestas mensagens não existe confirmação de receção por parte do cliente, pelo que se não for entregue, o servidor não chega a conhecer este facto.
2xx	Sucesso	O pedido foi processado com sucesso
3xx	Redirecionamento	Fornece informação sobre nova localização do utilizador a ser contactado, ou sobre serviços alternativos que possam estar disponíveis para concretizar a chamada.
4xx	Falha do pedido	Indicam que o pedido falhou definitivamente no servidor. Não deve ser feita nova tentativa do mesmo pedido sem alterar nenhum parâmetro, por exemplo, adicionar as permissões adequadas. No entanto, o mesmo pedido pode ser aceite em outro servidor.
5xx	Falha do servidor	O próprio servidor falhou na execução do pedido.

6xx	Falhas globais	Este tipo de resposta é enviada quando o servidor tem informação relativa a um utilizador em particular.

Tabela 2.4: Tipos de resposta no Protocolo SIP

Os Códigos de resposta mais comuns, e seu significado, são apresentados na tabela seguinte:

Código	Descrição
100 Trying	Outra ação está a ser executada, por exemplo, a consulta a uma base de dados.
180 Ringing	O terminal que recebeu o convite esta a ser alertado da tentativa de contacto.
181 Call Is Being Forwarded	Para indicar que a chamada foi reencaminhada.
182 Queued	Se o terminal chamado estiver ocupado ou temporariamente indisponível, o servidor pode colocar a chamada em espera em vez de a rejeitar. Este tipo de respostas pode ser enviado por diversas vezes, para manter o cliente informado da situação da espera.
183 Session Progress	Se ocorrer alguma situação que não se enquadra em mais nenhum dos restantes códigos, mas que ainda não levou ao interrompimento da mesma.
300 Multiple Choices	Se o endereço no pedido apresenta diversas opções,cada qual numa localização diferente. O utilizador pode escolher o terminal de comunicações preferido e redireccionar o pedido para essa localização.

301 Moved Permanently	O utilizador já não se encontra no endereço indicado no pedido. O cliente que fez o pedido deverá tentar o contacto para o novo endereço fornecido pelo campo contacto, no cabeçalho da mensagem, e atualizar os diretorios e listas de contacto com o novo valor, por forma a que futuros pedidos para este endereço sejam automaticamente reencaminhados para o novo endereço.
302 Moved Temporarily	O utilizador não se encontra no endereço indicado no pedido, embora que temporariamente. O cliente que fez o pedido deverá tentar o contacto para o novo endereço fornecido pelo campo contacto, no cabeçalho da mensagem, mas nao guarda o novo endereço.
305 Use Proxy	O recurso pedido deve ser acedido via PROXY, indicado no campo contacto.
380 Alternative Service	Se não foi possível realizar a chamada, mas existem serviços alternativos, que são descritos no corpo da mensagem.
400 Bad Request	O pedido não foi entendido, por exemplo, por erro de sintaxe. É normalmente dada a informação detalhada do erro.
401 Unauthorized	O pedido requer validação de utilizador.
403 Forbidden	Se o pedido foi entendido, mas por algum motivo, o servidor recusa-se a executá-lo.
404 Not Found	O servidor tem informação de que o utilizador não existe no endereço especificado no pedido.
500 Server Internal Error	O servidor deparou-se com alguma condição inesperada, que o impede de continuar a processar o pedido. O cliente deve tentar novamente o pedido.
501 Not Implemented	O servidor não suporta a funcionalidade solicitada, ou seja se não a reconhece, pois quando reconhece, responde com o códgo 406 se não a puder executar.
502 Bad Gateway	O servidor, atuando como gateway ou proxy, recebeu uma resposta inválida do servidor com que comunicou para executar o pedido.
503 Service Unavailable	Se o servidor se encontra temporariamente indisponível, quer por exemplo, para manutençao, ou devido a sobrecarga do sistema.

504 Server Time-out	Se o servidor não recebeu resposta em tempo útil, de um outro servidor externo a que tenha acedido.
505 Version Not Supported	O servidor não suporta a versão do protocolo SIP usada no pedido.
513 Message Too Large	O tamanho da mensagem excede a capacidade do servidor, impossibilitando o processamento do pedido.
600 Busy Everywhere	Se o terminal pretendido foi contactado com sucesso, mas não pretende receber a chamada naquele momento,podendo conter a informação de qual será o melhor momento para fazer nova tentativa.
603 Decline	O terminal foi contactado com sucesso, mas o utilizador não pretende de forma alguma receber a chamada, dando a indicação clara de rejeição de chamada.
604 Does Not Exist Anywhere	O servidor tem a informação clara de que o utilizador solicitado não existe no sistema.
606 Not Acceptable	O utilizador foi contactado com sucesso, mas existem aspetos da sessão que não são aceites, tais como largura de banda.

Tabela 2.5: Códigos mais comuns no Protocolo SIP

2.4 Servidor

Para que se possa ter este tipo de soluções a operar em rede local, é essencial e obrigatório que o sistema esteja instalado em algum tipo de servidor, por forma a que os terminais de PC possam, por exemplo ligar vários utilizadores na plataforma Sugar em simultâneo, em diferentes pontos na rede. Este servidor vai precisamente fornecer serviços aos seus clientes, que serão os restantes PC's da rede. O serviço fornecido neste caso, é a utilização da plataforma em si.

2.4.1 Definição

Um servidor é um sistema de computação centralizada que fornece serviços a uma rede de computadores. Esses serviços podem ser de natureza diversa, como por exemplo, arquivos e correio eletrónico. Os computadores

que acedem a algum serviço instalado num servidor são chamados clientes. As redes que utilizam servidores são do tipo cliente-servidor, utilizadas em redes de média/grande dimensão (com muitas máquinas) e em redes onde a questão da segurança desempenha um papel de grande importância. O termo servidor é largamente aplicado a computadores completos, embora um servidor possa equivaler a um software ou a partes de um sistema computacional, ou até mesmo a uma máquina que não seja necessariamente um computador.

A história dos servidores tem, obviamente, a ver com as redes de computadores. Redes permitiam a comunicação entre diversos computadores, e com o crescimento destas, surgiu a idéia de dedicar alguns computadores para prestar algum serviço à rede, enquanto outros se utilizariam destes serviços. Os servidores ficariam responsáveis pela primeira função.

Com o aumento do tamanho das redes, foi crescendo também a necessidade das redes terem servidores, aumento este também impulsionado pelo crescimento das empresas de redes e o crescimento do uso da Internet entre profissionais e utilizadores regulares. Tornou-se assim necessário desenvolver e aperfeiçoar as tecnologias para servidores.

Existem diversos tipos de servidores, com as mais variadas funcionalidades, mas para o objetivo deste trabalho, os servidores que interessam são os servidores web (para fornecer aos terminais cliente a utilização da plataforma Sugar CRM), e os servidores VoIP (para fornecer serviços de telefonia IP). Apresento então de seguida estes dois tipos de servidor.

2.4.2 Servidor web

Um servidor web é um programa de computador responsável por aceitar pedidos HTTP de clientes, geralmente os browsers, e servi-los com respostas HTTP, incluindo opcionalmente dados, que geralmente são páginas web, tais como documentos HTML com objetos embutidos (imagens, etc.), ou pode ser um computador que executa um programa que fornece a funcionalidade descrita anteriormente. O servidor web mais comum, e mais utilizado, é o servidor Apache (software livre). Os pedidos HTTP que se referem habitualmente a páginas HTML são normalmente feitos através de browsers. O processo inicia-se com a conexão entre o computador onde está instalado o servidor web e o computador do cliente (como na web não é possível prever a que horas um determinado serviço será acedido, os servidores web devem

estar em funcionamento contínuo).

A partir daí é processado o pedido do cliente, e conforme as restrições de segurança e a existência da informação solicitada, o servidor devolve os dados. Geralmente tudo o que se enquadre no conceito de ficheiro pode ser enviado como resultado de um pedido HTTP. Finalmente, os servidores web também podem executar programas e scripts, interagindo mais com o utilizador.

Para este trabalho, foi utilizado o servidor web Apache, não só pela sua popularidade e segurança, mas também por ser aquele que é recomendado para a execução do SugarCRM, para o qual a plataforma está otimizada.

Servidor Apache[21][22]

O servidor Apache é o mais popular servidor web. É um software livre, criado em 1995 por Rob McCool, então funcionário do NCSA (National Center for Supercomputing Applications). Numa pesquisa realizada em dezembro de 2007, foi constatado que a utilização do Apache representa cerca de 47.20% dos servidores ativos no mundo. Em maio de 2010, o Apache serviu aproximadamente 54,68% de todos os sites e mais de 66% dos milhões de sites mais movimentados. É a principal tecnologia da Apache Software Foundation, responsável por mais de uma dezena de projetos envolvendo tecnologias de transmissão via web, processamento de dados e execução de aplicações distribuídas.

O servidor é compatível com o protocolo HTTP versão 1.13. As suas funcionalidades são mantidas através de uma estrutura de módulos, permitindo inclusive que o utilizador construa os seus próprios módulos — utilizando a API do software.

SEGURANÇA Para garantir segurança nas transações HTTP, o servidor dispõe de um módulo chamado mod_ssl, que adiciona a capacidade do servidor para atender pedidos utilizando o protocolo HTTPS. Este protocolo utiliza uma camada SSL para encriptar todos os dados transferidos entre o cliente e o servidor, fornecendo um maior grau de segurança, confidencialidade e confiabilidade dos dados. A camada SSL é compatível com certificados X.509, que são os certificados digitais fornecidos e assinados por grandes entidades certificadoras no mundo.

CONFIGURAÇÃO Os arquivos de configuração, por padrão, em ambientes Unix-like, residem no diretório etcapache. O servidor é configurado por

um arquivo *core* nomeado httpd.conf e opcionalmente pode haver configurações para cada diretório utilizando arquivos com o nome .htaccess, onde é possível utilizar autenticação de usuário pelo próprio protocolo HTTP utilizando uma combinação de arquivo .htaccess com um arquivo .htpasswd, que guardará os utilizadores e senhas (criptografadas).

2.4.3 Servidor VoIP

A análise de servidores VoIP já foi realizada na secção 2.2.7 na página 21 deste trabalho.

2.4.4 Hardware

Servidores dedicados, que possuem uma alta requisição de dados por partes dos clientes e que atuam em aplicações críticas utilizam hardware específico para servidores. Já servidores que não possuam essas atuações podem utilizar hardware de um computador comum.

Como não é objeto de estudo deste trabalho, a análise profunda do funcionamento de servidores, fica apenas a nota de que neste trabalho, foi usado um servidor do tipo *servidor web*, sem a existência de nenhum hardware específico, apenas um computador normal.

Capítulo 3

Plataformas de referência

3.1 SugarCRM

3.1.1 Introdução

Esta foi a plataforma de CRM utilizada para este trabalho, por ser aquela com que a VLM tem vindo a trabalhar ao longo dos útlimos anos, e era concretamente para esta plataforma que se pretendia criar a aplicação. No entanto, como será analisado noutra secção deste trabalho, sendo entendido o conceito, a customização feita para acrescentar a funcionalidade de click-2-call seria facilmente aplicável a qualquer outro projeto desta natureza, desde que fosse possível aceder e alterar o código fonte.

Portanto, analisando o SugarCRM de uma forma global, pode dizer-se que este assume duas vertentes[23]:

- Vertente projeto open-source

- Vertente empresa

O projeto open-source nasce em 2004, iniciado por três indivíduos com um total de 50 anos de experiência em CRM proprietário, no entanto, descontentes e frustrados pelas falhas no software e pela falta de inovação. Desta conjugação de fatores, nasce a hipótese para a construção de um novo CRM, mas disponibilizando o código gratuitamente (sourceforge.net), obtendo assim o feedback dos utilizadores e também a integração de novos programadores para o projeto. Nascia então a versão CE (Community Edition) do SugarCRM. O número de Downloads em poucos meses e a popularidade da

aplicação (foi traduzida pela comunidade em dez idiomas diferentes) permitiu no ano seguinte, e com a injeção de capital por parte de empresas locais, criar e expandir uma nova empresa, a empresa SugarCRM.

Atualmente, são comercializadas três versões pagas do SugarCRM (Professional, Enterprise, e Ultimate)[5], com funcionalidades acrescidas em relação à versão CE (Community Edition), que serve de base para as restantes, e que é em grande parte, desenvolvida pela comunidade.

Sugar Professional	Sugar Enterprise	Sugar Ultimate
	Sugar PRO +	Sugar ENT +
-Automação de vendas, marketing e suporte ao cliente	-Gestão de oportunidades	-Suporte 24x7
-Relatórios e histórico de atividades	-Antecipação ao mercado	Gestor técnico atribuído
-Função mobile, com sincronização a posteriori	-Portal auto-atendimento ao cliente	-Cloud privada
-Instalação na cloud ou interno	-Streaming de atividade personalizado	-250 GB para alojamento
-15GB de espaço para alojamento	-Relatórios em SQL nativo	
-Formação em E-learning ilimitada	-Suporte técnico por telefone	
-Portal de suporte online	-Possibilidade de cloud privada	
-Suporte de ocorrências ilimitado	-60GB para alojamento	
25€/utilizador/mês	45€€/utilizador/mês	110€€/utilizador/mês

Tabela 3.1: Edições, características e preço do SugarCRM

Relativamente à versão CE, é distribuída livremente sob a licença GNU AGPL, que permite aceder, modificar e distribuír livremente o código.

Recentemente, a empresa SugarCRM adoptou uma postura muito comercial, dando pouco ênfase e desenvolvimentos à versão CE, focando-se mais nas versões pagas, facto que provocou algum descontentamento na comunidade de *developers*. Como tal, estão a emergir algumas soluções alternativas em termos de open-source, como são exemplo o SuiteCRM, que surgiu como uma derivação do Sugar, e atuando desde então de forma independente, continuando com o projeto pelos seus próprios meios.

3.1.2 Conceitos Chave[24]

O SugarCRM possui alguns conceitos que são fundamentais para a compreensão do modo de funcionamento da plataforma. Passo a apresentar aqueles que, na minha perspetiva, são os mais importantes.

Módulo - Estrutura pré-definida da conjugação de vários campos, com funcionalidades base que podem ser editadas.

Campo - Cada uma das propriedades dos módulos, por exemplo, o módulo Contactos tem campos tais como *primeiro nome, último nome, contacto telefónico, morada, etc*

Relacionamento - Quando definido, permite a visualização de informação cruzada entre vários módulos, por exemplo, na visualização de uma entidade, aparece um sub painel com os contactos associados a essa mesma entidade. Para tal, é definido um relacionamento do tipo *one-to-many.*

Pesquisa - Painel existente em todos os módulos, que permite filtrar os resultados de acordo com os parâmetros pretendidos.

Pesquisa Básica - Campos fundamentais de pesquisa que aparecem de imediato, em qualquer uma das vistas.

Pesquisa Avançada - Campos adicionais de pesquisa de resultados no módulo.

Controlador - Direcciona todos os pedidos de páginas para o sítio correto;

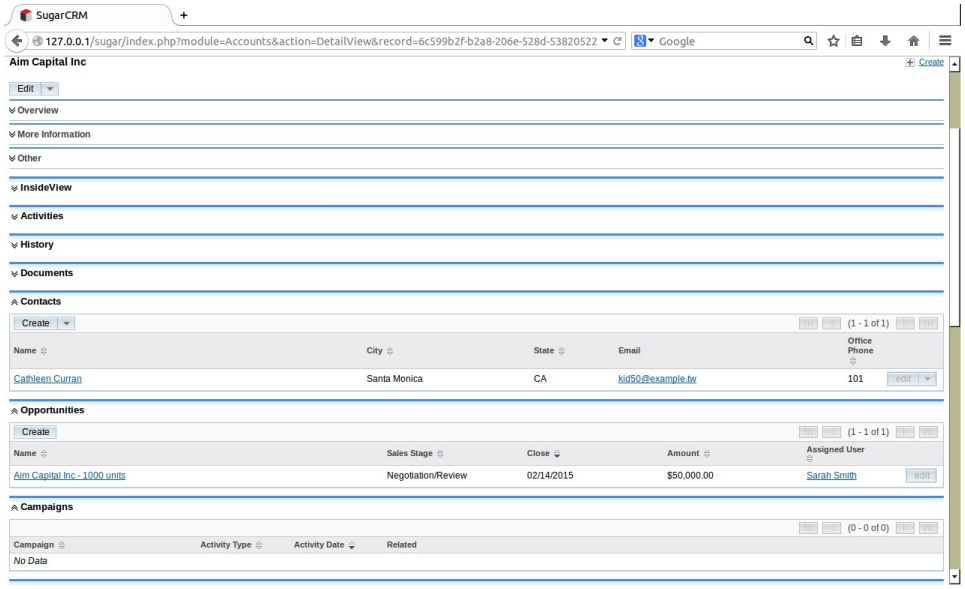


Figura 3.1: Exemplo de relacionamento definido

Vistas

Vista de Lista - Listagem de todos os registos do módulo;

Vista de Detalhe - Visualização dos dados, em modo de leitura, de um registo em particular;

Vista de Edição - Visualização, em modo de edição, de um registo. Em qualquer uma das vistas, é possível definir quais os campos do módulo que se pretende que sejam visualizados.

Display Strings - Para facilitar a tradução para os vários idiomas, o sistema refere-se sempre às variáveis pelo mesmo nome, sendo que o que é apresentado no ecrã varia com o idioma seleccionado, desde que os respetivos módulos de linguagem tenham sido previamente instalados. Existem strings da aplicação, e strings dos módulos, definidas separadamente. Por exemplo, o array $GLOBALS['app_strings'] contém todas as *display strings* referentes à aplicação.

Drop-Down-Lists - São representadas como um array de pares no formato nome=>valor, e servem para o utilizador seleccionar um valor, entre vários, como p.e, o país de origem. O nome corresponde ao que fica guardado em base-de-dados, enquanto que o valor é o que será

apresentado no ecrã. Estes valores estão definidos no array $GLOBALS['app_list_strings'].

SugarBean.php - Ficheiro localizado na raíz da aplicação. Contém a classe *SugarBean base,* que é extendida pelos módulos para ler, escrever ou apresentar dados.

$dictionary - Um array da aplicação que contem todas as definições de variáveis (vardefs), bem como os relacionamentos entre as tabelas na base de dados. Este array é construído dinamicamente, com base nas definições existentes em vardefs.php

Entry Point - O ponto de entrada da plataforma é o ficheiro index.php, que garante o login do utilizador. Requere, pelo menos, os parâmetros *module e action* ou seja, o módulo que se pretende aceder,e a acção a realizar. Portanto, o index.php irá gerar um URL do tipo http://sugar-addr/index.php?module=Contacts&action=DetailView

3.1.3 Tarefas de administração da plataforma SugarCRM

As tarefas que a seguir se indicam,apesar de não terem intervenção direta no aspeto técnico e executável da plataforma, são de grande importância em termos de administração do sistema.

Gestão de Utilizadores - É uma característica fundamental em qualquer aplicação que serve diversos utilizadores. Fornece um método de autenticação segura no sistema. Guarda em base de dados a informação de todos os utilizadores, e permite que cada utilizador possua preferências e propriedades distintas. No painel de administração, é possível criar, editar, ativar e desativar utilizadores. Cada um dos utilizadores pode assumir papéis diferentes, inclusivé ser administrador do sistema, ou não. Caso não seja, o administrador pode restringir o acesso a determinadas ações e/ou módulos, se pretender.

ACL sigla para Access Control List, é através deste recurso que é possível restringir o acesso a módulos, bem como dados e ações disponíveis para determinados utilizadores, por exemplo, salvar e apagar. São definidas no painel administrativo da plataforma, e podem até, no

caso das versões PRO e ENT, restringir o acesso a campos específicos dentro dos módulos.

Gestão de Passwords Esta funcionalidade habilite o administrador de sistema a criar e gerir as passwords geradas pelo próprio sistema, e também regras de password para os utilizadores. É despoletada quando é criado um novo utilizador, e o sistema envia automaticamente a password de acesso. É possível enviar uma nova password de utilizador em qualquer instante, definir uma data de expiração da validade, definir tamanho máximo e mínimo, o conteúdo obrigatório ou permitido (Maiúsculas/minúsculas,algarismo,símbolos)

Acompanhar a ação dos utilizadores Originalmente, esta característica apenas permitia ao administrador acompanhar as ações de visualização e edição de registos por parte dos utilizadores. Desde a versão 5.1, esta funcionalidade foi desenvolvida, para fornecer também informação sobre performance do sistema e de ligações à base de dados. Este acrescento é de grande utilidade ao administrador, pois assim pode otimizar a plataforma baseado nas necessidades reais dos utilizadores. É possível a criação de relatórios de atividade recorrendo a estas informações.

Gestão de grupos Esta funcionalidade, apesar de só estar disponível nas versões pagas PRO e ENT, é muito útil em casos de um grande número de utilizadores, em que estes podem ser divididos em categorias, atribuíndo permissões e/ou restrições, tal como na gestão de utilizadores, mas neste caso, em grupo.

3.1.4 A ferramenta Studio

O studio é uma ferramenta que permite desenvolver e customizar a plataforma, sem que seja necessário recorrer diretamente ao código. É possível com esta ferramenta, através de uma interface gráfica, criar módulos, criar campos, customizar os tipos de vista para qualquer módulo, etc, facilitando em muito, os utilizadores e/ou administradores do sistema que não teêm conhecimentos de programação. Todas as customizações efetuadas com este recurso, são *uprage-safe,* visto que ficam guardadas na pasta *custom/* da plataforma, não sendo alterada em caso de atualização da versão Sugar.

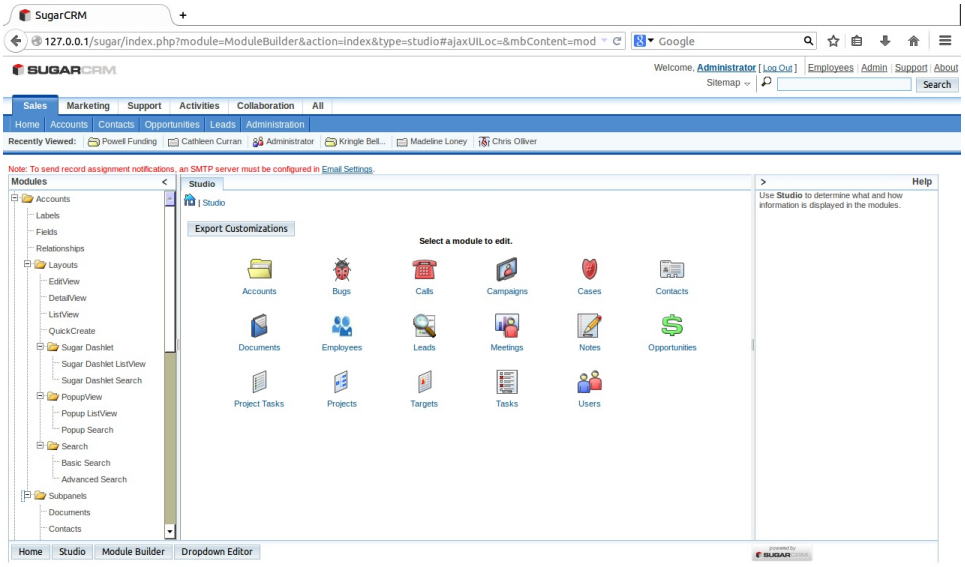


Figura 3.2: Ferramenta Studio

3.1.5 Aspetos Técnicos[23][25]

A plataforma SugarCRM foi inicialmente concebida para a chamada *LAMP Stack (Linux,Apache,MySQL,PHP)*, mas entretanto a comunidade esforçou-se para criar compatibilidade com outros sistemas, e atualmente já é possível instalar o SugarCRM em todos os SO. No entanto, recomenda-se a instalação em sistemas Unix.

É uma plataforma web-based, isto é, acedida por um web-browser (Firefox ou Chrome preferencialmente), e é estruturada com recurso a módulos, que representam um aspeto funcional muito característico dos sistemas CRM, tais como Entidades(clientes,fornecedores,parceiros, etc), Contactos, Atividades (reuniões, contactos telefónicos, etc), potencias negócios, e oportunidades de negócio.

Por exemplo, o módulo Entidades permite a criação e gestão de contas de clientes (e outro tipo de entidades), o módulo Atividades permite criar e gerir atividades relacionadas com entidades, oportunidades, tarefas a realizar, etc.

Os módulos do Sugar estão desenvolvidos por forma a apoiar o relacionamento comercial em cada etapa do ciclo de vida de um negócio, desde a identificação do potencial negócio, passando pela fase de venda, até ao suporte comercial no período pós-venda. Por forma a facilitar o cruzamento de informação, cada módulo permite visualizar toda a informação relacionada.

O SugarCRM permite customizar e acrescentar funcionalidades, utilizando os módulos base, ou construíndo novos módulos derivados destes.

Sugar *Framework*

O código da aplicação Sugar é baseado numa *framework* modular com ponto de entrada obrigatório (index.php), por forma a garantir que o utilizador tem login válido, e reencaminhando daqui os pedidos para os pontos solicitados.

Estrutura de diretórios

Os diretórios chave da plataforma SugarCRM são os diretórios *cache, custom, data, include, metadata,modules* pois é nestes que está contido o principal código.

Realço pessoalmente o diretório *custom,* pois é aqui que fica guardada toda a informação sobre as customizações efetuadas na plataforma, e para onde são copiadas novas configurações, e os novos campos que são adicionados quando se instalam novos módulos. É portanto, necessário garantir que não há *override* em nenhum ficheiro, quando procedemos a alterações, sob pena de serem perdidas configurações anteriores. É também o diretório conhecido como *upgrade-safe,* isto é, quando é feito um upgrade na versão da plataforma, este diretório não é alterado.

3.1.6 Arquitectura MVC

MVC significa *Model View Controller*,e é uma arquitectura muito utilizada para aplicações web ou outras em que seja necessário a interacção com o utilizador. O objetivo desta arquitetura é separar a gestão de interface, da gestão da aplicação, criando uma fase de processamento intermédia que facilite a comunicação entre as duas. Cada um dos componentes MVC tem determinadas tarefas a cumprir.

Model

Este componente representa a camada da aplicação. Tem como tarefa gerir as comunicações com os recursos externos, tais como base de dados e web services. Serve ainda para calcular os valores de alguns campos, embora no caso concreto do sugar, essa tarefa não se aplique. Um bom modelo

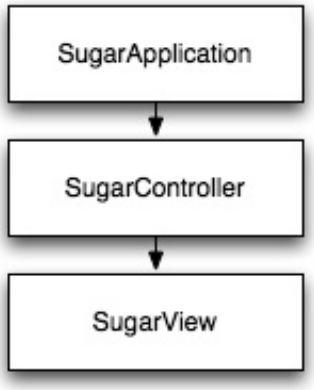


Figura 3.3: Modelo MVC

---

fornece uma interface limpa à raíz da aplicação, por forma a entregar todos os dados, prontos a apresentar, sem necessitarem de nenhum tipo de operação ou interpretação.

View

O componente *View* representa a interface do utilizador. Aqui é gerida toda a lógica por detrás da apresentação de formulários e de dados, por exemplo.

Controller

Este componente funciona como que uma cola entre as camadas *Model* e *View*.Um bom controlador aceita pedidos do utilizador, solicita o modelo específico para a informação que necessita, e de seguida, solicita a vista para apresentar essa informação ao utilizador. Deverá ser o mais simples possível, e não deve comunicar com a BD diretamente nem considerar a forma como os dados serão apresentados.

3.1.7 MVC na perspetiva do Sugar

O Sugar utiliza o modelo MVC para gerir os pedidos do utilizador. Cada solicitação ao ponto de entrada (index.php) especifica variáveis no pedido HTTP , indicando o módulo (que corresponde ao *Controller,*) e a ação,(que

corresponde à componente *View.*)

Sugar - Aplicação

É na camada da aplicação (*Model*) que ocorre o primeiro tratamento de pedidos. Nesta camada são geridos todos os pré-requisitos da plataforma , incluíndo validação de sessão, e de login de utilizador, o tema para apresentação de dados, e a ligação à base de dados. Inclui-se também nesta camada, o tratamento de dados respeitante a definições do sistema, tais como fuso horário do utilizador (escolhido pelo utilizador no primeiro login), e informação da expiração ou não, da password, entre outro tipo de informações. Portanto, esta camada da aplicação carrega o controlador, mas antes de o executar, concretiza as seguintes tarefas:

i) Validação de utilizador: se este já estiver ligado no sistema, o pedido continua, caso contrário, o pedido é direccionado para a página de login;

ii) Verificação das definições de utilizador, nomeadamente se este tem permissão para aceder a todos os módulos;

iii) Verificação se é o primeiro login. Em caso afirmativo, levar o utilizador a escolher algumas definições,

iv) Verificação se todas as strings de idioma foram carregadas com sucesso.

Assim que todos este pré-requisitos estiverem tratados, avança-se para a execução do controlador, que está concebido por forma a receber o pedido, e executá-lo.

Sugar - Controlador

Esta camada é responsável por gerir o fluxo principal do pedido, e controlar todos os pedidos para o módulo especificado. A implementação do Controlador do Sugar está em perfeita sintonia com a interpretação do que um controlador deve fazer segundo o modelo MVC, e contém muitas das ações de *hook*[1] subjacentes às ações mais comuns num módulo normal, tais como implementação dos tipos de vista, salvar e apagar registos num módulo.

[1] associado ao conceito de logic hook no Sugar. Esta propriedade será analisada em detalhe noutra secção

O controlador do Sugar fornece a representação para os três tipos de vista, no módulo que estiver a controlar. Existem duas formas de concretizar esta tarefa:

I) A primeira é ter um método de ação definido na classe de controlador, que representa a ação e transfere o controlo ao longo da vista. Especifica, pelo menos, a vista a usar nesta ação, mas pode também, no entanto,realizar alguma operação lógica que não esteja diretamente relacionada com a camada de vista, tais como verificar se o valor de um campo está ou não definido, e caso não esteja, direccionar o pedido para outro local, alterando o fluxo.

II) A segunda forma é recorrer aos ***Logic Hooks.***

Um Logic Hook é uma customização à plataforma, mas a grande diferença entre um logic hook e uma outra customização comum, é que, usando logic hook, não se está a substituír nem a alterar código, mas sim a acrescentar código que está concebido para ser executado mediante determinadas circunstâncias ao longo da aplicação. Um logic hooks pode ser entendido como uma operação lógica que é despoletada em circunstâncias definidas, por exemplo *"pre_ action"* ou *"post_ action"*, sendo que action pode representar as ações de *save, delete, retrieve, restore, login, logout* entre outras. Este tipo de operação é util quando se pretende, por exemplo, modificar algum valor antes deste ser guardado na base de dados, ou atualizar uma outra base de dados, com base em ações realizadas no Sugar, e até para o envio condicional de e-mails. Um ação que também tira proveito do uso de *logic hooks*, é colocar o valor atual da datahora aquando da criação ou alteração de algum registo. É uma funcionalidade que não está visível pelo utilizador, mas é automaticamente despoletada no momento *pre_ save*, ou seja, quando o utilizador clica em salvar um novo registo, após ter preenchido os campos, o sistema acciona o logic hook, por forma a guardar a hora corrente num campo especificado, e só depois salva efetivamente e guarda na base de dados esse registo, já com a informação da hora da ação. Esta funcionalidade não está definida de base no sistema, mas pode ser criada para qualquer módulo.

Sugar - Vista

Após o controlador ter sido executado, e caso a vista especificada seja válida, a respetiva classe derivada do SugarView é carregada. Esta classe gere a informação necessária para o tratamento da vista, tal como carregar o template de visualização.

3.1.8 Camada de Metadados

O princípio base de funcionamento de um módulo segue normalmente uma lógica conhecida como CRUD (Create, Retrieve, Update, Delete), que significa Criar, retornar, atualizar e apagar. São as ações possíveis nos registos de todos os módulos. Ao mesmo tempo, cada um dos módulos utiliza o mesmo formulário base para executar a mesma ação em módulos diferentes, então a camada de metadados surge como a forma de evitar que o mesmo código seja copiado para diferentes partes do sistema, para executar por exemplo, uma listview no módulo entidades e uma listview no módulo contactos, alterando os nomes dos campos.

O que a camada de metadados faz é precisamente gerir os formulários para cada um dos três tipos de vista, ou seja, criar um ficheiro de metadados que define o layout de cada uma delas, por forma a facilitar a construção e manutenção dos formulários, sendo que o principal exemplo desta metodologia são as editview e detailview.

Vistas de Detalhe e de Edição

A principal forma de introduzir dados no SugarCRM é através da vista de edição, onde são apresentados os campos do módulo para que o utilizador introduza o respetivo valor,que será guardado em base de dados. Este propósito é o mesmo quer para criar novos registos, quer para atualizar registos. Assim que o registo é gravado com sucesso, o utilizador é encaminhado para a vista de detalhe, que é a versão só de leitura do registo. Ambos estes formulários (editview e detaiview) lidam entre si, por forma a fornecer o principal meio de interagir com um registo de um módulo. No modelo MVC usado pelo Sugar, o editview e detailview estão predefinidos para usar os metadados para construir os formulários apresentados ao utilizador, recorrendo à classe editview (definida em include/EditView/EditView2.php), para a editview, e de seguida, extendendo esta classe na sub-classe detail-

view (definida em include/DetailView/DetailView2.php). Ambas as classes funcionam da mesma forma. O primeiro passo é construír a vista carregando os metadados. Está estruturado como um array associativo pertencente ao array global $viewdefs. O esquema está subdivido em duas partes,

- A secção templateMeta, que apresenta os dados gerais ao formulário.
- A secção form, que contém as definições dos campos.

A primeira secção apresenta dados como por exemplo, os botões de salvar, editar ou apagar o registo, cabeçalhos e rodapés do formulário, numero de colunas em que será fornecida a visualizaçao, e outras partes que sejam constantes e comuns a todos os formulários.

A segunda secção contém as definições dos campos, que estão também organizadas em arrays associativos, mas agrupados por secção, e depois, por linhas. O agrupamento por secções permite fazer um formulário multi-sessão, agrupando campos em comum. Para além disso, agrupar por linhas permite definir a localização exata dos elementos no formulário. Por exemplo, se o formulário foi definido para ser apresentado em duas colunas, e dividido em duas secções, o array $viewdefs deverá ter o formato indicado na figura 3.4, que resulta na vista apresentada na figura 3.5, na página seguinte.

Figura 3.4: Código do ficheiro detailviewdefs.php do módulo Entidades

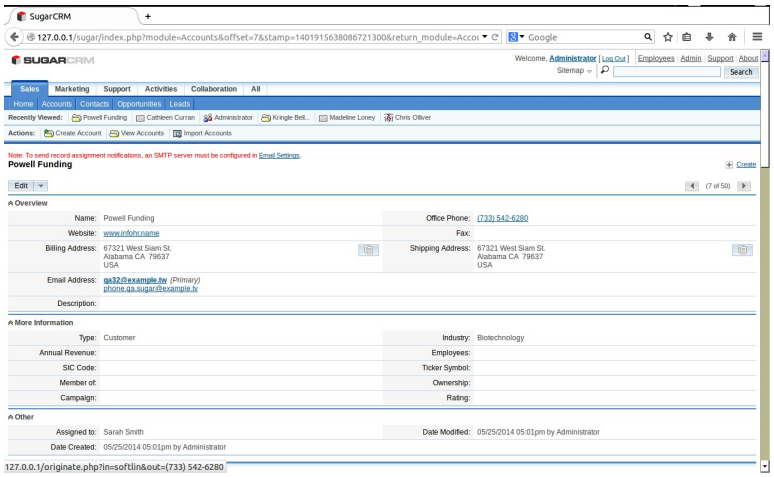


Figura 3.5: Visualização em detalhe de um registo do módulo Entidades

3.2 Asterisk

3.2.1 Introdução[26]

O Asterisk é um sistema VoIP, uma framework de código aberto para construção de aplicações de comunicação capaz de transformar um computador num servidor de comunicações. É um sistema implementado para tirar o máximo partido de IP PBX's[2] e tecnologias VoIP. É atualmente utilizado em call-centers, pequenas e grandes empresas, entidades governamentais, um pouco por todo o mundo. É uma marca registada da empresa Digium, que patrocina este projeto, e como open-source, recebe também contribuições (a nível de progamação) da comunidade Asterisk.

Este projeto nasceu em 1999, quando Mark Spencer[3] disponibilizou o código com licença GPL[4], e desde entao tem vindo a evoluír com a ajuda da empresa Digium, pela qual é patrocinado, e também da comunidade de programadores.

Este sistema está para aplicações de comunicação como o Apache está para as aplicações web, por exemplo. Serve para criar aplicações e soluções de comunicação em tempo-real, recorrendo a protocolos de comunicação tais como SIP, IAX-2, ZAP. O SIP é no entanto, o mais comum, por ser o protocolo padrão.

3.2.2 Aspetos técnicos[28]

O asterisk diferencia-se dos restantes sistemas VoIP, essencialmente pela abordagem em relação ao plano de chamadas, uma vez que trata todos os canais de comunicação da mesma forma, enquanto que por exemplo, em sistemas proprietários, há diferenciação entre um canal de saída e uma extensão interna. Na perspetiva do Asterisk, todos os dados que entrem ou saem do sistema, teêm de passar por algum tipo de canal, e apesar de haver vários tipos de canais, o Asterisk trata todos da mesma forma, pelo que ter um utilizador terminal a atender uma chamada na sua extensão interna, ou no seu telemóvel, na rede externa, é indiferente em termos de processos do Asterisk.

[2] do inglês Private Branch Exchange
[3] Fundador do Asterisk e Diretor Técnico da Digium
[4] Licença utilizada em projetos open-source

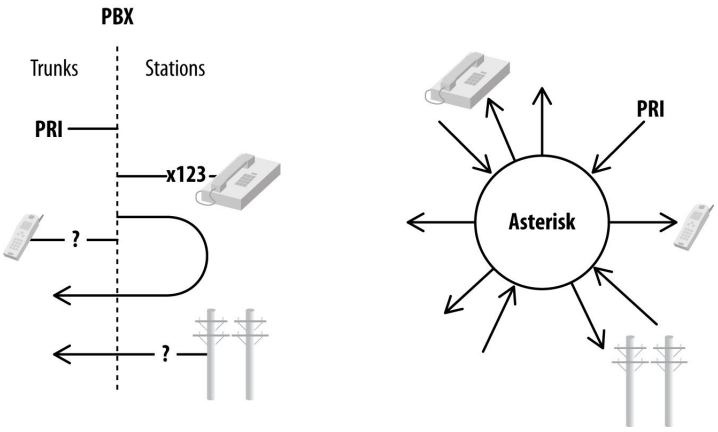


Figura 3.6: Exemplo da arquitetura Asterisk

Estrutura

O Asterisk é construído baseado em módulos, em que um módulo representa um componente passível de ser carregado para implementar uma funcionalidade em concreto, seja um controlador de canal (chan_sip.so) ou um recurso que permite ligação a tecnologias externas (func_odbc.so). Os módulos necessários são carregados com base no conteúdo do ficheiro de configuração */etc/asterisk/modules.conf* Introduzido o conceito de módulo, facilmente se percebe que se nenhum módulo for carregado, o sistema é incapaz de executar o que quer que seja, é inútil.

Os módulos estão agrupados por categorias, de acordo com os seus objetivos, sendo que durante a apresentação de cada categoria, serão expostos aqueles que são entendidos como os mais pertinentes, pois não se pretende com este trabalho analisar detalhadamente cada um dos módulos existentes, mas sim explicar o modo de funcionamento do Asterisk.

A) Aplicação Os módulos de aplicação são usados para definir as várias ações que podem ser aplicadas a uma chamada. A aplicação Dial(), por exemplo, é responsável por realizar uma chamada, função fulcral no Asterisk, e está definida no módulo app_dial.so, pelo que este módulo deve ser obrigatoriamentte carregado no ficheiro modules.conf.

B) Gravação dos detalhes de chamada Estes módulos permitem registar automaticamente, em base de dados, com as devidas configurações, os detalhes de chamada, desde origem e destinatário, duração,

terminais envolvidos, entre outros.

C) Controladores de Canal Sem estes módulos, seria impossível o Asterisk concretizar as chamadas. Cada controlador é específico do protocolo que o canal suporta. Se os canais estiverem definidos em protocolo SIP (o mais utilizado), é essencial carregar o módulo chan_sip.so.

D) Tradutores de Codecs Permite ao Asterisk converter streams de áudio entre as chamadas. Ou seja, caso o sistema receba num canal SIP, uma chamada do circuito externo (que utiliza outro codec de audio), deve ser traduzido para o formato de receção. Os codecs mais comuns no Asterisk são g.722, alaw, gsm, ulaw, e os módulos tradutores que devem ser carregados para cada um deles tem o nome codec_g722, e assim sucessivamente.

E) Funções do plano de chamadas Complementam os módulos de aplicação. Acrescentam valor no sentido de processarem strings, gerem a conectividade com a base de dados, a data e hora, etc.

F) Módulos PBX Módulos periféricos que melhoram os mecanismos de controlo e configuração. Nesta categoria, destacam-se os módulos pbx_config.so, por ser o que permite ler o plano de chamadas, e o pbx_spool.so, que suporta a funcionalidade dos *call files*. Também estão nesta categoria os módulos referentes a definições de data e hora, e também de RTP (Realtime Transport Protocol).

Estrutura de ficheiros

O Asterisk é um sistema complexo, que utiliza vários recursos do SO. É então importante conhecer a estrutura de ficheiros associada ao Asterisk, para se saber onde se procurar e acrescentar determinada informação.

I) Ficheiros de configuração Estes são os ficheiros de configuração do sistema, onde são definidas as extensões, o plano de chamadas, os canais, e tanta outra informação. Encontram-se no diretório *(/etc/asterisk/)*, que acaba por ser o diretório de trabalho de um administrador de sistema Asterisk.

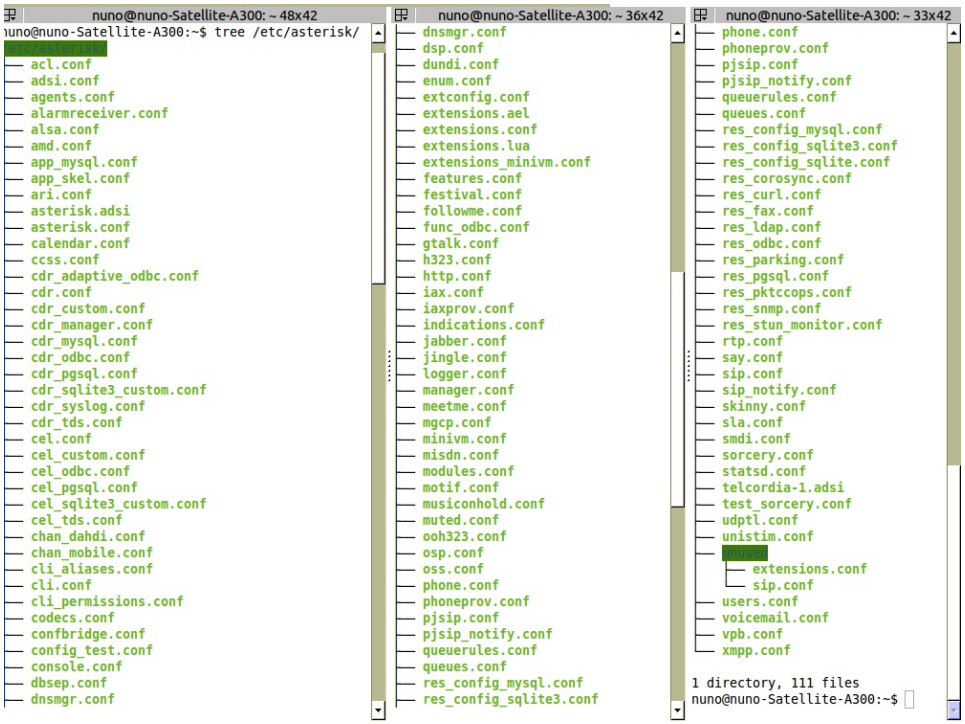


Figura 3.7: Exemplo de Ficheiros de configuração Asterisk

II) Módulos *(/usr/lib/asterisk/modules)* Normalmente não há necessidade de intervir neste diretório, mas é aqui que são instalados os módulos.

III) Libraria de recursos *(/var/lib/asterisk/)* É onde o sistema vai procurar ficheiros de personalização, por exemplo, sons do sistema, música de chamadas em espera, etc.

IV) Diretório de Spool *(/var/spool/asterisk/)* É onde são guardados items temporariamente, tais como mensagens de voz, gravação audio de chamadas, etc.

V) Diretório de Log *(/var/log/asterisk/)* Muito útil para efeitos de *debugging*, pois aqui são escritas todas as mensagens de avisos, erros, etc.

VI) Plano de chamadas *(/etc/asterisk/extensions.conf)* É o motor do Asterisk. Todos os canais que entram no sistema passam pelo

plano de chamadas, que contém as informações do fluxo para lhe dar seguimento.

Interação com o sistema Asterisk - Protocolo AMI[28]

Para se poder tirar o máximo proveito de um sistema Asterisk, é necessário que se conheça o sistema, por forma a adicionar todas as customizações para que fique mais orientada para o pretendido. Uma das características fundamentais deste tipo de plataformas, é a facilidade de interação com outros serviços e aplicações, característica esta que foi explorada para concretizar o objetivo deste trabalho.

O AMI (*Asterisk Manager Interface*), que deverá estar devidamente configurado, é a forma de se dar instruções ao Asterisk através de aplicações externas. Existem duas formas de interagir com o AMI, uma forma unilateral, em que apenas o sistema informa sobre eventos ocorridos, e uma forma bilateral, em que o cliente do sistema envia pedidos, e o sistema responde. Apresento a seguir os modelos de comunicação referidos:

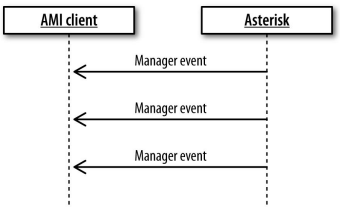


Figura 3.8: Modelo de Comunicação de eventos

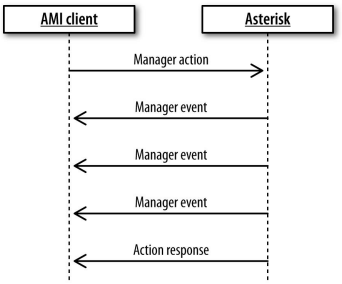


Figura 3.9: Modelo de Comunicação de Ações

Comunicação de Eventos: é uma comunicação unilateral, do Asterisk para o cliente AMI, para informar de alguma ocorrência no sistema.

Comunicação de Ações: é uma comunicação bilateral, em que o cliente envia um pedido ao sistema,e é enviada uma resposta, mesmo que seja para informar que o pedido não é válido. Pode utilizar-se um pedido para conhecer todos os pedidos que são válidos para o sistema. Este pedido é o *action:ListCommands*

Figura 3.10: Exemplos de Eventos a informar

Figura 3.11: Exemplos de Ações a executar

Para interagir com o AMI, existem dois ficheiros de configuração fundamentais[27], sendo eles */etc/asterisk/manager.conf* e */etc/asterisk/http.conf*, em que é necessário ativar o AMI e o HTTP, com as configurações pretendidas, nomeadamente em termos de conexões permitidas, o porto que se pretende utilizar para estas ligações, etc.

manager.conf (/etc/asterisk) - VIM 151x22

```
;
; AMI - The Asterisk Manager Interface
;
; Third party application call management support and PBX event supervision
;
; This configuration file is read every time someone logs in
;
; Use the "manager show commands" at the CLI to list available manager commands
; and their authorization levels.
;
; "manager show command <command>" will show a help text.
[general]
enabled = yes
webenabled = yes

port = 5038
bindaddr = 0.0.0.0
[nuno]
secret=nuno
read=all
write=all
```

http.conf + (/etc/asterisk) - VIM 151x20

```

; Asterisk Builtin mini-HTTP server
;
[general]
enabled=yes
bindaddr=127.0.0.1
;
; Port to bind to for HTTP sessions (default is 8088)
;
;bindport=8088
;
; Prefix allows you to specify a prefix for all requests
; to the server.  The default is blank.  If uncommented
; all requests must begin with /asterisk
;
;prefix=asterisk
;
; sessionlimit specifies the maximum number of httpsessions that will be
; allowed to exist at any given time. (default: 100)
```

Figura 3.12: Ativação dos protocolos TCP e HTTP

Por sua vez, o AMI pode ser acedido por diferentes meios, que passo a apresentar:

I) **AMI sobre TCP** O sistema responde aos pedidos do cliente AMI. Um exemplo desta comunicação é usando, por exemplo, o programa *telnet*. Por defeito, o Asterisk utiliza o porto 5038 para esta conexão, mas pode ser configurado manualmente, para qualquer outro porto. Esta solução é de extrema utilidade, para gerir remotamente o sistema.

II) **AMI sobre HTTP** Este método é o mais utilizado por programadores para construirem aplicações web, uma vez que tem a vantagem, em relação ao anterior, de poder apresentar vários tipos de codificação, sendo que o tipo pretendido é definido através de um parâmetro passado no URL. Neste tipo de conexão, o Asterisk utiliza por defeito, o porto 8088, que também pode ser manualmente configurado no ficheiro

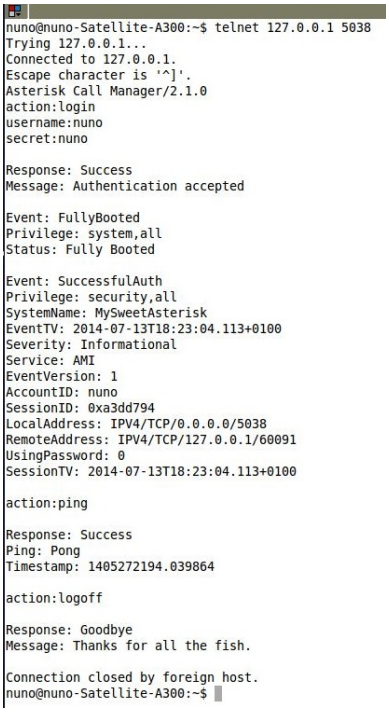


Figura 3.13: Exemplo de AMI sobre TCP usando *telnet*

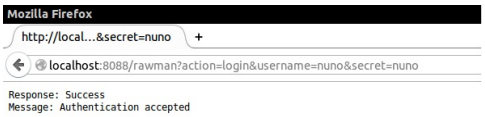


Figura 3.14: Exemplo de AMI sobre HTTP

---

http.conf. Para ativar este método, é necessário configurar os ficheiros *manager.conf* e *http.conf*. O conceito de comunicação com o sistema é igual ao anterior, mas neste caso, a comunicação é feita por http, habitualmente através de um web-browser.

Tipos de Codificação

i) /**rawman** A resposta, quando se envia um pedido nesta codificação, vem no formato igual ao que é apresentado na conexão por TCP.

ii) /**manager** A resposta é recebida num formato de formulário HTML.

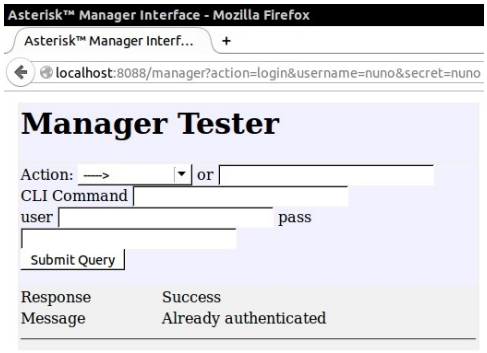


Figura 3.15: Exemplo de AMI sobre HTTP com resposta em HTML

iii) /**mxml** A resposta vem no formato xml.

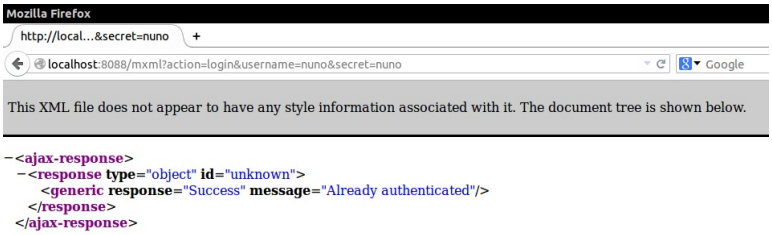


Figura 3.16: Exemplo de AMI sobre HTTP com resposta em XML

Agora estamos em condições de conhecer um exemplo de uma comunicação tipo gestão de eventos. Por exemplo, se pretendermos registar o evento de registo de um telefone ao sistema, podemos efetuar um login via HTTP, utilizando o URL *http://ip_server:8088/rawman?action=login&user=nuno&secret=nuno*, registamos o telefone no sistema, e de seguida, voltamos ao HTTP com a acção de waitEvent. A figura 3.17 mostra este procedimento.

III) **Call File** É uma funcionalidade do Asterisk exlusiva para originar chamadas, visto este ser o principal objetivo com que normalmente se interage com o Asterisk. O único pré-requisito necessário para ter acesso a este funcionalidade é ter permissão de escrita a um diretório do sistema, em concreto */var/spool/asterisk/outgoing*. Um *call file* é um ficheiro de texto bastante simples, que contém toda a informação necessária para o estabelecimento da chamada. Veremos de seguida

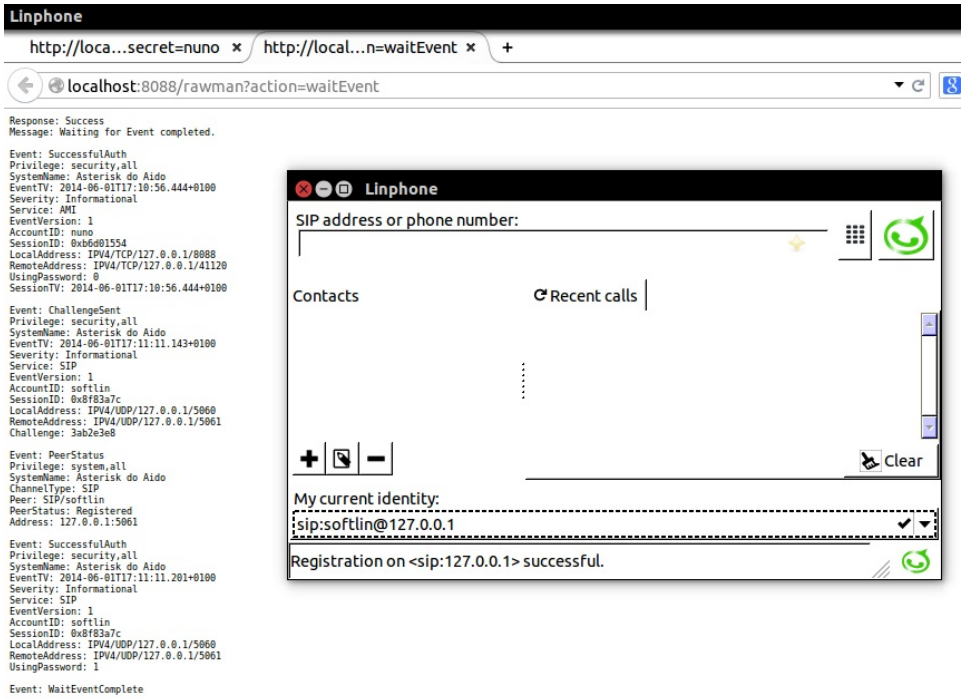


Figura 3.17: Comunicação do evento *Registo de um telefone no sistema*

um exemplo deste ficheiro. Para já, seguimos com o procedimento a seguir para concretizar esta chamada. Antes de mais, e estritamente necessário, o Asterisk necessita de ter o módulo pbx_spool carregado e ativo, pois será este módulo a despoletar a chamada. Com o ficheiro criado contendo toda a informação, é necessário mover o mesmo para o diretório do sistema */var/spool/asterisk/outgoing/*. O sistema Asterisk deteta automaticamente um ficheiro naquele local, e de imediato despoleta a chamada com os dados contidos no ficheiro. É recomendável, neste caso, que se utilize a opção de mover o ficheiro, e não de copiar, pois corre-se o risco de o Asterisk despoletar a chamada mal detecte o início do ficheiro, mesmo sem conter toda a informação necessária, pelo que poderá não se conseguir o objetivo pretendido.

i) Sintaxe do ficheiro

Para se conseguir o resultado de despoletar uma chamada no sistema, é necessário que o *call file* siga uma determinada sintaxe,

que é explicada de seguida. O nome, pode ser um qualquer, sendo aconselhado evitar caracteres de símbolo.

Opção	Descrição
Channel	Parâmetro fulcral para qualquer ficheiro deste tipo. É o que define o canal de comunicação configurado no sistema que deve ser ativado, e que será o ponto de origem da chamada.
Context	Contexto do sistema em que a extensão a contactar está incluída. Este parâmetro está sempre relacionado com os parâmetros Extension e Priority.
Extension	Extensão a contactar após o canal inicial ser atendido. Pode entender-se como o ponto de destino da chamada.
Priority	Prioridade da extensão onde o Asterisk deve entrar no *Dialplan*.
MaxRetries	Campo não obrigatório, que por defeito tem o valor 0. Corresponde ao número de tentativas que o sistema deve fazer, antes de desistir, caso a chamada não seja atendida no ponto de origem.
RetryTime	Tempo de espera (em segundos) entre novas tentativas. Só é aplicável caso o parâmetro anterior esteja definido, e por defeito, tem o valor 300 (5min).
WaitTime	Tempo de espera para que a chamada seja atendida na origem, antes de desligar.Opcional, por defeito assume o valor 45 (segundos).
CallerID	Define o ID de chamador, que será o nome que aparece no visor dos terminais de telefone.
AlwaysDelete	Opção que define se o ficheiro é apagado ou não, após o término da chamada. Toma, por defeito, o valor *yes*, ou seja, estes ficheiros são apagados.
Archive	Se este parâmetro for enviado com o valor *yes*, o ficheiro de chamada irá ser arquivado no diretório */var/spool/asterisk/outgoing_done/* apos o término da chamada.

Tabela 3.2: Tabela de exemplo de campos de um call file

Estas são as opções mais utilizadas nestes ficheiros, no entanto existem outras. O ficheiro terá de conter as opções e o seu valor no formato *opção:valor*. Um exemplo de contéudo de um *call file* é:

```
# Uma linha começada com # é um comentário no ficheiros
#As três opções seguintes são obrigatórias
channel:SIP/myphone
extension:101
context:LocalSets
#As opções seguintes são opcionais
callerID: Nuno Aido < 100 >
MaxRetries:2
RetryTime:120
waitTime:15
Archive:yes
```

Se colocarmos este contéudo num ficheiro de texto com o nome abc, e assumindo que temos o sistema Asterisk instalado em Linux, podemos correr o comando *mv abc /var/spool/asterisk/outgoing* na shell, a partir do diretório onde se encontra o ficheiro, e esta será suficiente para despoletar um chamada com o seguinte comportamento: O telefone que está registado no sistema como *myphone* começa a tocar, e quando for atendido, começa automaticamente a chamar para a extensão 101, que deverá estar devidamente configurada no sistema. Quando esta extensão for atendida, a chamada está estabelecida entre myphone e a extensão 101. De realçar que no Asterisk, o telefone myphone tem também uma extensão associada (100, por exemplo), mas para este efeito, deve ser passado o parâmetro como o nome que está registado no sistema, pois é este dado que contém a informação do canal de comunicação.

3.2.3 Hardware

Como o Asterisk utiliza maioritariamente o protocolo SIP, qualquer hardware phone que também conheça este protocolo será aceite pelo sistema. No

entanto, para realizar chamadas para a rede pública, é necessário hardware específico, e também ter fornecimento de serviços VoIP por parte de empresas de telecomunicações. A própria marca Digium fabrica este hardware, bem como telefones IP para integrar o sistema.

Capítulo 4

Criação da aplicação Click 2 Call no SugarCRM

Antes de mais, é necessário realçar que existem já no mercado, algumas soluções disponíveis para este efeito. No entanto, são soluções pagas e que implicam vínculos contratuais com outras empresas, com o pagamento de uma mensalidade. Uma vez que se pretende tirar partido das tecnologias em open-source, criou-se esta aplicação de raíz, o que implicou um estudo detalhado da forma de funcionamento das plataformas SugarCRM e Asterisk.

Para criar esta aplicação no SugarCRM, é então necessário idealizar o que se pretende. O pretendido é que, ao navegar na plataforma, quando se faz uma consulta a um contacto ou entidade, módulos que têm por natureza um contacto telefónico associado, seja possível despoletar uma chamada para esse mesmo número, a partir de um click, utilizando a extensão interna associada ao utilizador que esteja ligado na plataforma. Ou seja, um utilizador *"A"* está ligado na plataforma, e ao consultar o contacto telefónico do cliente *"B"*, pretende que o sistema Asterisk estabeleça uma chamada para o seu telefone de secretária que está à sua frente, e ao atender, que inicie automaticamente uma chamada para o cliente. Para que tal seja possível, é necessário que o Asterisk tenha conectividade com o SugarCRM, por forma a poder ser manipulado e instruído. O Asterisk necessita, portanto, de receber instruções do SugarCRM, e para além disso, necessita de conhecer os parâmetros da chamada pretendida, neste caso, qual a extensão interna para a qual deve estabelecer a primeira chamada, e qual o número externo para onde se pretende ligar.

É então necessário que o Asterisk aceite comunicações externas, e esta condição verifica-se, e processa-se pelo *AMI (Asterisk Manager Interface).* O AMI é uma interface de monitorização e gestão do sistema Asterisk, fornecida pelo próprio sistema. Permite monitorizar em tempo real os eventos ocorridos, bem como ativar/validar pedidos para o Asterisk concretizar alguma tarefa, como por exemplo, originar uma chamada. De seguida, irão ser introduzidos alguns conceitos no que toca ao Protocolo AMI, quais as formas de interagir com o mesmo, vantagens e desvantagens de cada uma delas, e qual a forma de comunicação adoptada para este trabalho em concreto. No final deste capítulo, iremos ainda ao lado do SugarCRM conhecer qual a forma de participar nesta comunicação.

4.1 Operação do Asterisk

Do ponto de vista da aplicação a criar, o Asterisk apenas necessita de ter uma via de comunicação aberta para receber instruções, neste caso, fornecidas pelo SugarCRM. Visto que as formas de interação com o Asterisk já foram analisadas anteriormente, importa nesta secção referir qual o método escolhido.

Fazendo então um resumo dos métodos possíveis, temos as seguintes hipóteses:

a) Criar um conta com username e password no sistema, e

   i) Aceder remotamente por TCP (utilizando o telnet, p.e.)

   ii) Aceder remotamente por HTTP (utilizando um web-browser)

   Em qualquer um destes acessos, é necessário enviar comando de login seguidos de outras ações que se pretendam realizar. No caso de acessos por HTTP, estes comandos são enviados via URL.

b) Outra forma mais simples, passa por inserir os parâmetros necessários para o estabelecimento da chamada num ficheiro de texto e movê-lo para um diretório do sistema. Neste caso, não é necessário qualquer login no sistema, o que fragiliza a solução.

Comparando as soluções disponíveis, deparamo-nos com algumas vantagens, e algumas desvantagens em todos elas. Na minha perspetiva, as interações por:

TCP são à partida excluídas para o que se pretende, pois já que a idéia é criar uma aplicação click2call, seria muito difícil, se não mesmo impossível, passar todos os parâmetros de login e de ação simultaneamente, através de um click recorrendo a programas de acesso remoto.

HTTP são aparentemente, a melhor solução pois, por ser necessário efetuar login no sistema, pode-se restringir o acesso remoto ao sistema Asterisk apenas a utilizadores credenciados. No entanto, tem a desvantagem de não permitir duas ações no mesmo URL, ou seja, tem de se concretizar um acesso para fazer login ao sistema, passando os parâmetros por um URL, e de seguida outro acesso para se informar a ação que se pretende, neste caso, originar uma chamada, e enviar os parâmetros por um segundo URL. Outra desvantagem é de o login ser feito por URL visível, sendo assim uma falsa segurança, pois qualquer pessoa consegue visualizar diretamente o username e a password no URL.

Call File é a forma mais simples de se realizar o pretendido, pois é extremamente fácil criar e manipular um ficheiro de texto, inserindo nele todos os parâmetros necessários,de forma dinâmica,pois serão vários os utilizadores do sistema, e os números externos a contactar, e de seguida, mover o ficheiro. Tem a desvantagem de não ser necessário qualquer login, mas como esta é uma funcionalidade interna, para ser usada pelos colaboradores da empresa, que à partida já têm acesso aos telefones, e consequentemente, acesso ao sistema, acaba por não ser impeditivo o seu uso, uma vez que o próprio sistema Asterisk irá ser configurado por forma a apenas aceitar pedidos vindos da sua rede local.

Expostas as vantagens e desvantagens de cada uma das abordagens para interagir com o Asterisk, foi esta última solução que se enquadrou melhor no que se pretendia, pela maior facilidade e versatilidade na manipulação dos dados a inserir no *call file* dinamicamente, pelo que foi a abordagem efetuada para a concretização deste trabalho.

4.2 Operação do Sugar

Do lado do sugar, no contexto desta aplicação click2call, é necessário que comunique com o Asterisk, por um dos métodos explicados anteriormente. Tal como foi já analisado, entendeu-se que a criação de um *call file* seria o

melhor caminho a seguir. Realce-se nesta fase que este ficheiro não é criado nem manipulado pelo Sugar, mas sim pelo servidor. Relembrando o que já foi apresentado anteriormente, cada módulo apresenta três tipos de vista, a *vista de lista, vista de detalhe e vista de edição*; Quando se entra em qualquer dos módulos do Sugar, por defeito, é de imeditao apresentada a vista de lista desse módulo, sendo que no caso concreto dos módulos Entidades e Contactos, o número de telefone é um dos campos apresentados. Nesta fase, a grande dificuldade foi a implementação na vista de lista, pois o método de criação do link na vista de detalhe não se verificou aplicável na vista de lista. Posteriormente, e após algumas tentativas falhadas, foi contornado este assunto, utilizando outra abordagem para a vista de lista, que passou por fazer um pré-processamento da vista de lista antes desta ser apresentada. Este pré-processamento incluía a criação do link para o sítio pretendido, associado aos campos de números de telefone.

A função a realizar do lado do Sugar é gerar um URL com os parâmetros necessários, por forma a informar o servidor, que vai processar os dados recebidos, e solicitar ao Asterisk o estabelecimento da chamada. Esta função de gerar o URL vai ser integrada nos campos associados a números de telefone no sítio pretendido, neste caso, vistas de lista e de detalhe dos módulos Entidades e Contactos.

4.2.1 *Por detrás do click*

Explicar o que se vai passar em *background* quando se clicar no número de telefone é o objetivo desta secção.

Então, quando ocorre esse click, o Sugar irá aceder por HTTP ao servidor, onde o próprio Sugar e o Asterisk estão instalados e a correr paralelamente. Este acesso é feito através de uma customização da propriedade contacto telefónico [1] nas vistas de lista e de detalhe dos módulos Entidades e Contactos. Nesta customização, irá ser definido que um determinado número de telefone, ao ser clicado, vai solicitar um ficheiro chamado *originate.php* que se encontra no servidor, passando-lhe os parâmetros necessários para a chamada, através de URL. Veremos mais à frente o que se passa neste ficheiro *originate.php,* mas desde já se realça que é este quem vai criar o *call file*

[1] para efeitos deste trabalho, apenas se irá considerar o contacto telefónico fixo de uma entidade ou contacto, e a extensão interna de colaboradores da mesma empresa, sendo no entanto, facilmente aplicável a qualquer outro tipo de contacto telefónico.

necessário, e movê-lo para o local a que pertence.

Para já, e continuando na perspetiva do Sugar, é necessário proceder à customização dos campos pretendidos.

Como já foi explicado em capítulos anteriores, todas as customizações a efetuar na plataforma SugarCRM são guardadas e processadas no diretório *custom/* localizado na raíz da aplicação. Este diretório possui diversos sub-diretórios, mas os diretórios que nos interessam neste caso, são:
/var/www/html/sugar/custom/modules/Accounts/, e
/var/www/html/sugar/custom/modules/Contacts/

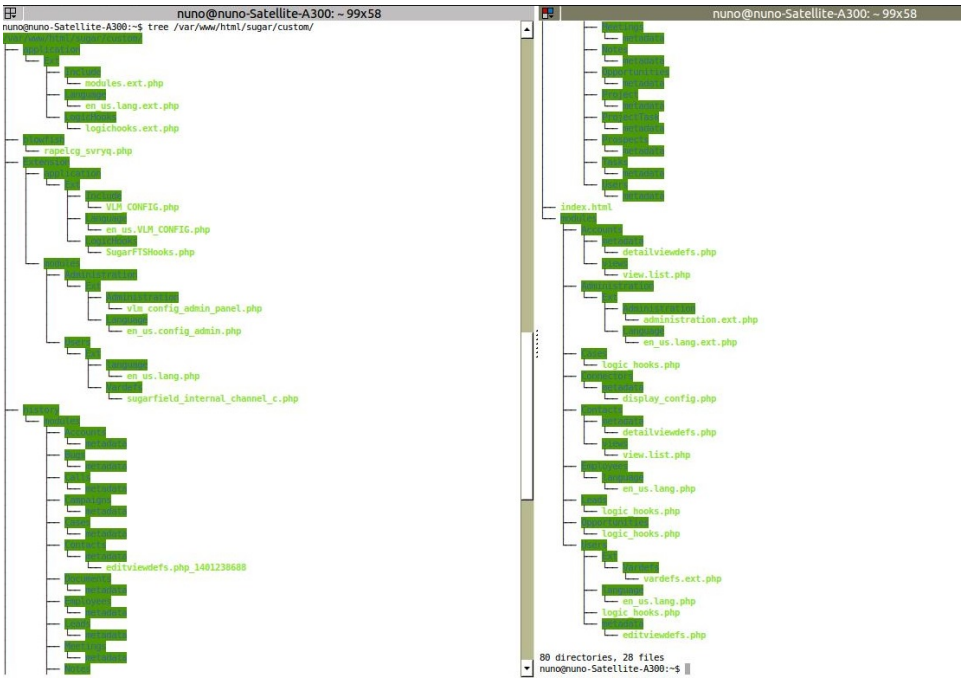


Figura 4.1: Estrutura do diretório *custom/*

No sub diretório *metadata* é onde se processam as customizações para os diversos tipos de vista nestes módulos, através dos ficheiros *listviewdefs.php, editviewdefs.php,e detailviewdefs.php*. Visto que o nosso add-on é aplicável na vista de detalhe, iremos criar precisamente o ficheiro editviewdefs.php nos sub diretórios referidos, e respeitantes aos módulos Entidades e Contactos. Salienta-se, nesta fase, que estes ficheiros não são criados originalmente pelo sugar, pois as definições standard não são carregadas a partir do diretório

custom. A localização destas definições standard encontram-se em:
/var/www/html/sugar/modules/Accounts/metadata/, e
/var/www/html/sugar/modules/Contacts/metadata/,

Procedemos então à cópia desses mesmos ficheiros, referentes aos dois módulos que nos interessam (Os ficheiros são diferentes para cada módulo, visto conterem informação diferente dos campos a apresentar nestas vistas), para os respetivos diretórios *custom*, uma vez que não é objeto de estudo neste trabalho, a criação de ficheiros de vistas. Fica no entanto, a nota, de que este ficheiro pode ser construído de raíz, configurando todos os campos que se pretende que sejam visualizados, qual o tamanho que cada um ocupa na estrutura, e em que posições. Deste modo, iremos apenas acresentar o código respetivo ao campo *phone_work* no módulo Contactos e campo *phone_office* do módulo Entidades. Seguiu-se esta lógica, devido a uma política da empresa VLM, em que apenas é permitido realizar chamadas a partir do telefone fixo, para números também fixos, sendo que as chamadas para números móveis devem ser realizadas a partir do telemóvel da empresa. No entanto, seria fácil acrescentar também a funcionalidade aos contactos de telemóvel.

Passo então a descrever o procedimento para tal customização dos campos.

Já é conhecido o facto de que a plataforma Sugar é programda em linguagem PHP, mas este tipo de linguagem é habitualmente integrada com código HTML, daí ser utilizada maioritariamente para aplicações web[2]. A customização que se pretende fazer, nada mais é, do que a criação de um URL, apontando para um ficheiro localizado no servidor, que irá processar a informação contida no URL.

Um hyperlink pode ser criado em código HTML, usando a sintaxe:
texto clicável,
sendo que a opção

- **target='_blank'** indica que a nova página irá ser aberta num novo separador;

- **href="URL_destino",** neste caso, deverá ser o URL que contém o endereço do servidor, , o nome do ficheiro *originate.php*, e os parâme-

[2]O Facebook é um exemplo de página web programada em PHP

tros para a chamada, pois o ficheiro *originate.php* foi concebido para receber estes parâmetros por URL nas variáveis *in* e *out*; Um exemplo de URL_destino válido é http://192.168.1.66/originate.php?in=linphone_pc1&out=234111222[3] assumindo que o servidor que contém os sistemas tem o endereço IP 192.168.1.66.

- **texto clicável** Neste caso em concreto, pretende-se que o texto clicável seja o contacto telefónico associado (234 111 222 no exemplo anterior), para que este continue visível para outra situações, mesmo que não se pretenda utilizá-lo como clicável.

Voltando um pouco atrás, um dos parâmetros que é necessário que este código customizado comunique ao servidor, é o parâmetro *in*, que corresponde ao telefone da extensão onde a chamadá irá ter início. Para podermos passar esta variável, é necessário conhecê-la, e para tal, foi criado um campo no módulo de Utilizadores, para inserir manualmente esta informação. Foi atribuído a este campo, o nome *internal_channel*, e só aparece na vista de edição do módulo, para que possa ser editado, se for necessário. Como não afeta a atividade normal do utilizador em nenhum aspeto, não foi acrescentado a mais nenhuma vista. Cada utilizador terá de introduzir neste campo, o nome do canal de comunicação associado ao seu telefone interno. Ressalve-se que um mesmo telefone interno pode ser partilhado por vários utilizadores num mesmo gabinete, e neste caso, poderia ser introduzido o mesmo valor para este campo associado a todos estes utilizadores, ou em alternativa, manter um nome distinto para cada utilizador, e configurar no Asterisk cada um dos nomes para o mesmo telefone. No sistema de testes, o canal de comunicação associado ao utilizador *Admin* foi configurado para ser o softlin, sendo que este foi o nome atribuído ao canal de comunicação configurado e ligado no Asterisk.

Gerar o URL

i) **Apontar ao ficheiro originate.php no servidor** Para aceder ao ficheiro que irá criar o call-file, é necessário conhecer o endereço IP do servidor do sistema, mas como este parâmetro pode ser alterável, foi criado em paralelo a esta implementação, um módulo de configurações

[3] linphone_pc1 é um softphone Linphone que foi configurado no sistema de testes, que simula um telefone IP

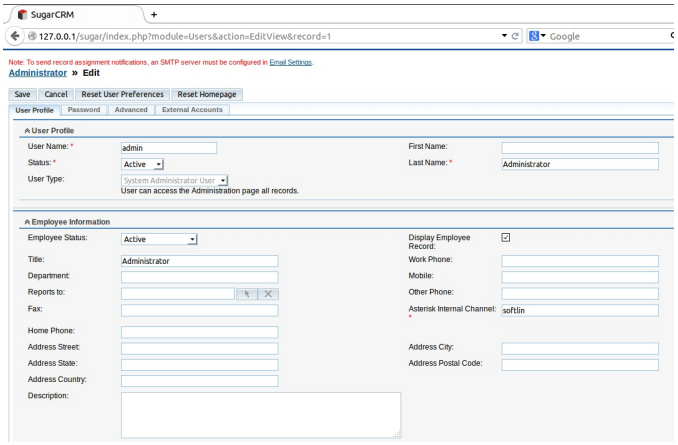


Figura 4.2: Campo internal_channel_c na edição de utilizadores

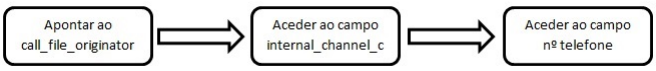


Figura 4.3: Workflow para gerar o URL

(módulo CONFIGURADOR), apenas acessível pelos administradores, para guardar esta informação, bem como outras informações necessárias. Desta forma, fica facilitado o acesso a este parâmetro, e quando for necessário alterá-lo, basta aceder ao módulo CONFIGURADOR, e alterar o valor desse registo. Da mesma forma, é criado também neste módulo um registo para guardar o nome do ficheiro originador, que também poderá ser alterado.

Por exemplo, se o IP do servidor for 192.168.1.66, cria-se um registo no módulo CONFIGURADOR, com as propriedades name='ip_server' e description='192.168.1.66/', e outro registo com name='call_file_originator' e description='originate.php'. A forma de aceder a estes valores será analisada na subsecção referente à manipulação da vista de detalhe (secção 4.2.2), e será criada uma variável local com estes valores, por forma a facilitar a sua visualização e identificação, com os nomes $ip_server e $call_file_originator, respetivamente.

ii) **Aceder ao campo internal_channel_c** O código customizado terá de conter uma referência para este valor, por forma a introduzi-

lo no URL que irá solicitar o *originate.php*. Um ponto pertinente é que este valor irá variar consoante o utilizador que estiver ligado no Sugar. No entanto, o Sugar possui uma propriedade interessante para verificar o utilizador que está ligado no sistema. É o seguinte:
Enquanto uma sessão está aberta, o sistema cria e mantém uma variável global chamada *$current_user*, que permite saber qual é o utilizador corrente, e ao mesmo tempo aceder a qualquer campo deste utilizador. Foi precisamente com este objetivo que foi criado anteriormente um campo customizado no módulo Utilizadores (internal_channel). Para aceder a este campo, é necessário informar o PHP de que vamos utilizar a variável global do sistema *$current_user*. Esta informação deve ser dada antes do início do array $viewdefs['Contacts'], da seguinte forma:
global $current_user;
Desta forma, podemos aceder ao valor de qualquer um dos campos associados ao utilizador corrente, usando a sintaxe
$current_user->first_name, se quisermos, por exemplo, aceder ao primeiro nome do utilizador para qualquer outro fim. Neste caso, como queremos aceder ao campo *internal_channel*, teremos de usar a variável como *$current_user->internal_channel_c.* Realço o facto de me referir ao nome do campo usando _c[4] no final do nome 'real' do campo. Esta situação deve-se ao facto de que, todos os campos adicionados ao Sugar, em qualquer módulo, são automaticamente acresentados desta notação no final do nome, para que possam ser facilmente distinguidos dos campos que o sistema traz por defeito. Inclusivé, na base de dados, o sistema guarda o nome dos campos com o acrescento _c.

iii) **Aceder ao campo número de telefone** Falta ainda ter acesso ao número de telefone (ou extensão externa). Da mesma forma que o Sugar cria uma variável global relativa ao utilizador corrente, existe também uma variável de sistema relativa ao último retorno da base de dados, que é a variável $focus. Na vista de detalhe, estamos perante este mesmo retorno, pois para poder ser apresentada a vista de detalhe de um contacto, foi feita pelo sistema uma consulta à base de dados, para ir buscar os valores a apresentar. Sabendo que o campo do módulo Contactos que nos interessa tem o nome de phone_work, podemos

[4] c significa custom

referir-nos a este valor da seguinte forma: *$focus->phone_work,* sendo que no módulo Entidades, o campo pretendido é o *phone_ofice*. No entanto, esta variável encontra-se vazia na vista de lista, pelo que foi efetuada outra abordagem para acesso a este valor, que será analisada na subsecção referente à manipulação da vista de lista (secção 4.2.3).

Temos portanto, três componentes para gerar o URL pretendido:

i) *$ip_serv/$call_file_originator*;

ii) *$current_user->internal_channel_c*;

iii) *$focus->phone_work*;

Falta então integrá-los, por forma a obter o URL. Em linguagem PHP, é possivel aceder ao conteúdo das variáveis, usando a sintaxe ".$nome_da _variavel.", pelo que o nosso URL_destino será definido por:
http://$ip_serv/$call_file_originator?in= ".$current_user->internal _channel_c."&out=".$focus->phone_work."

Temos então, desta forma, um URL dinâmico, construído instantaneamente quando se utiliza a funcionalidade de clicar no contacto para estabelecer a chamada. De seguida irá ser apresentada a inclusão desta funcionalidade nas vistas de detalhe e de lista.

4.2.2 Manipulação da vista de Detalhe

É no ficheiro *detailviewdefs.php* que se econtram todas as informações para apresentação de dados na vista de detalhe do módulo respetivo. Existe uma versão deste ficheiro que o Sugar instala, e cuja configuração assume por defeito. Se entretanto forem efetuadas alterações a esta mesma (ou outra) vista de um determinado módulo, usando por exemplo, a ferramenta para *Developers*, o Studio, irá ser criado pelo sistema um outro ficheiro com as novas configurações, na pasta sugar_dir*/custom/modules/*nome_do_ módulo*/metadata/* sendo que qualquer outra customização que se pretenda acrescentar, será editando este ficheiro nesta pasta. O detailviewdefs.php é um ficheiro em código PHP que contém um array de arrays, sendo o array principal definido por $viewdefs['Contacts']=array(...);

Nesta fase, é necessário acrescentar ao campo phone_work, na vista de detalhe do módulo Contactos, o código customizado para fazer a hiperliga-

ção ao URL gerado. Este procedimento é feito por edição do ficheiro detailviewdefs.php que foi copiado para a pasta sugar_dir/*custom/modules/* nome_do_módulo/*metadata/*, acrescentando um par nome=>valor ao sub array que define o campo phone_work (ou phone_office, no caso do módulo Entidades), sendo que o nome é 'customCode', e o valor é o código HTML que foi apresentado anteriormente para definir o campo como link. Em concreto, temos de procurar a referência ao campo phone_work, e acrescentar o seguinte código ao respetivo array, da seguinte forma:

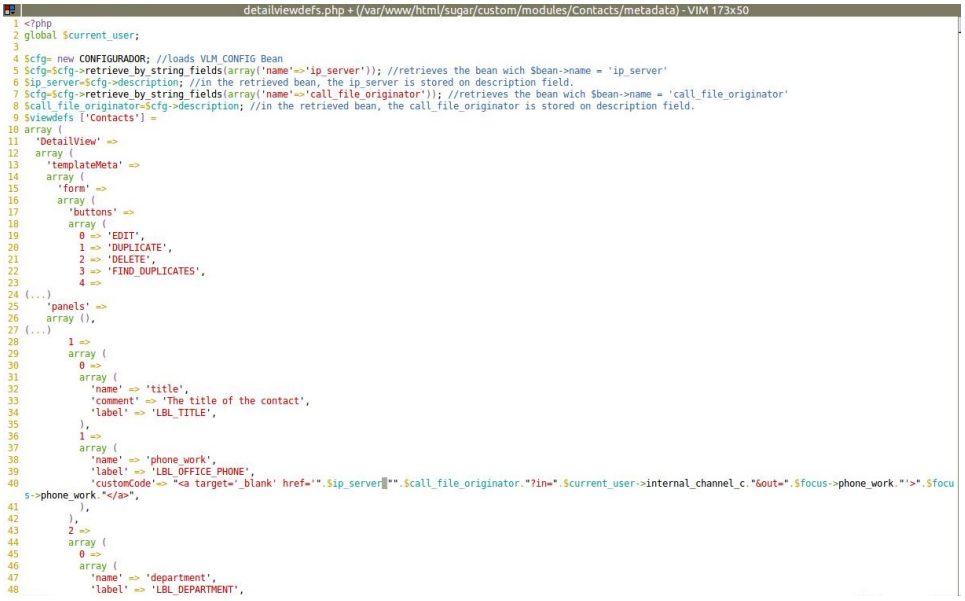


Figura 4.4: Manipulação da vista de detalhe do módulo Contactos

No código apresentado nesta imagem, as linhas 4 a 8, inclusivé, referem-se ao acesso aos valores guardados em registos do módulo CONFIGURADOR. A linha 4 carrega um objeto do tipo CONFIGURADOR, e carrega para o mesmo (linha 5) o resultado da função *retrieve_by_string_fields*, em que é feita uma pesquisa ao módulo onde "name"="ip_server", daí a importância deste registo ser guardado precisamente com este nome. A linha 6 acede ao campo onde está introduzido o valor (campo description), e guarda-o na variável local $ip_server. O mesmo sucede para o registo *call_file_originator*.

4.2.3 Manipulação da vista de Lista

A primeira abordagem realizada para definir os campos relativos a contactos telefónicos de Entidades e Contactos, foi a abordagem apresentada na subsecção anterior, que acedia ao campo phone_work através da variável *focus*, que se refere ao último retorno da base de dados. No entanto, esta abordagem não é aplicável na vista de lista. Uma vez que a vista de lista acede a vários registos simultaneamente, a variável *focus* encontra-se sem dados. Foi portanto necessário, abordar o assunto de outra forma, e a solução encontrada foi fazer um pré processamento da vista de lista antes de apresentar no ecrã. Para tal, é criado um diretório *views* em *sugar_dir/custom/modules/nome_do_módulo/*, com o ficheiro *view.list.php*. Neste ficheiro, será programada a função de pré-processamento da vista de lista.

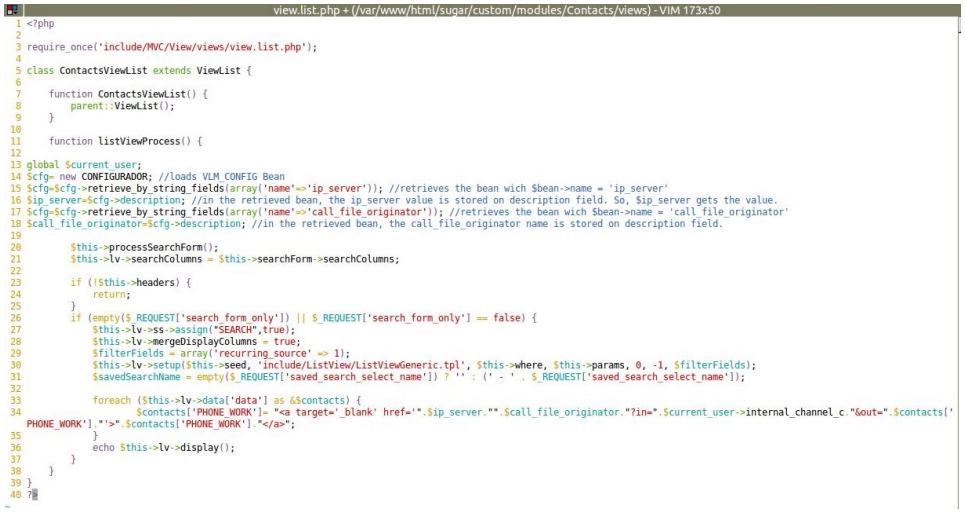


Figura 4.5: Manipulação da vista de lista do módulo Contactos

O cerne desta abordagem está no ciclo *foreach*, que vai definir que, para cada contacto apresentado, o campo phone_work (módulo Contactos) ou o campo phone_office (módulo Entidades), será um link para o URL gerado, da mesma forma que foi apresentada para a vista de detalhe. Para aceder aos registos do módulo CONFIGURADOR, foi necessário proceder da mesma forma apresentada na secção de manipulação de vista de detalhe.

4.3 Operação do servidor

Para integrarmos esta solução num rede local, é necessária a existência de um servidor, acessível por todos os PC's e telefones IP, por forma a ser possível os utilizadores ligarem-se na plataforma a partir dos seus terminiais, e fornecerem instruções ao Asterisk, embora que de forma indireta. Para o efeito, é suficiente um servidor web, instalado num PC com características normais. Escolheu-se um servidor apache, muito comum em sistemas Unix, e onde está instalado o Sugar. Também o Asterisk está instalado neste mesmo computador. O servidor será o intermediário da comunicação entre Sugar-CRM e Asterisk, visto que é aqui que se encontra o ficheiro *originate.php,* responsável pela criação do *call file* que dará origem ao estabelecimento da chamada. Este mesmo ficheiro irá receber e processar os dados vindos do Sugar, e de seguida, dar a instrução ao Asterisk para estabelecer a chamada.

4.3.1 Ficheiro *originate.php*

Este ficheiro foi programado em linguagem PHP, por uma questão de coerência com o Sugar. No entanto, seria possível ter sido programado com recurso a outra linguagem, nomeadamente Pearl e Python, que são também linguagens muito usadas neste tipo de aplicações, e em interações com o Asterisk.

A função deste ficheiro é precisamente receber e processar a informação, por forma a criar o *call file*, com base nos parâmetros recebidos.

Tarefas a desempenhar:

a) Retirar a informação do URL - O ficheiro foi concebido por forma a receber os parâmetros por URL, em concreto, recebe na variável *in* a extensão de origem da chamada, e na variável *out*, a extensão destino.

b) Criar o call file no formato exigido pelo Asterisk, inserindo os dados recebidos.

c) Mover o ficheiro originado, para o diretório */var/spool/asterisk/outgoing/*

Por facilidade de manipulação dos dados, criou-se um template do *call file* no formato pretendido, utilizando expressões no lugar onde era pretendido colocar os valores.

Neste caso, o originate.php necessita de carregar o template para uma string, substituír nessa mesma string as expressões pelo valor correto, e colocar a string alterada num novo ficheiro, por forma a que o template continue disponível e inalterado para a utilização seguinte.

Este novo ficheiro é o *call file* pretendido. Todas estas tarefas são executáveis recorrendo a funções do PHP, nomadamente as funções:

file_get_contents() - Lê o conteúdo de um ficheiro para uma string. Neste caso, utiliza-se esta função para ler o conteúdo do ficheiro *template*, na forma *file_get_contents("template");*

str_replace() Esta função substitui as ocorrências de um determinado texto numa string, por um outro texto, na mesma string. Neste caso, pretendiam-se substituir os textos {EXT_IN}, {EXT_OUT}, {CONTEXT}, e ainda um parâmetro opcional {CALLER}, pelos valores de `SIP/"."$_GET[in]" [5] ,"$_GET[out]", "Local"[6], e "VLM - "."$_GET[in]"[7], respetivamente.

tempname() Cria um ficheiro de texto com nome aleatório de 6 caracteres. É possível acrescentar um prefixo ao nome, da seguinte forma:
$call_file=tempnam(".","VLM_") sendo que o primeiro argumento da função significa que o ficheiro será criado no diretório corrente, e o segundo argumento,o prefixo.

De realçar que a função *tempname* cria um ficheiro com a permissão 0600, ou seja, apenas o *owner* do ficheiro pode ler/escrever. Dado que é necessário que o Asterisk faça a leitura do ficheiro, tem de se alterar as permissões do mesmo antes de ser movido para o diretório *spool*. Este ponto é realizado com recurso à função PHP *chmod*.

chmod() Altera as permissões do ficheiro passado no primeiro argumento, para o modo indicado no segundo argumento (base octal). Exemplo de utilização:
chmod($call_file,0604);

[5] concatenação do parâmetro in com a string SIP/, pois é esta a forma como o canal está configurado no sistema

[6] ou outbound, dependendo do tamanho da extensão out

[7] concatenação da string "VLM - " com o nome do originador da chamada, pois sem a passagem deste parâmetro, o sistema efetua chamadas anonimamente

file_put_contents() Insere um sring num ficheiro. Aqui recorreu-se a esta função para inserir a string resultante das subsitutições de valores anteriormente referidas, no ficheiro temporário criado com a função *tempnam()*. Exemplo de utilização:
file_put_contents($call_file,$str_alter);

system() Executa um comando na shell. O comando que se pretende que seja executado deve ser passado como argumento em formato de string. Neste caso, queremos que o ficheiro gerado seja movido para o diretório de spool. Utilizou-se esta função na forma *system("mv ($call_file /var/spool/asterisk/outgoing")*, dando início ao estabelecimento da chamada de imediato.

template...tml) - VIM 25x44

```
channel:{EXT_IN}
extension:{EXT_OUT}
context:{CONTEXT}
callerid:{CALLER}
```

originate.php + (/var/www/html) - VIM 71x44

```
<?php
#le o conteúdo do template para a variavel $str_orig
$str_orig=file_get_contents("template");

#substitui o valor da extensão de origem na string
$str_alter=str_replace("{EXT_IN}","SIP/"."$_GET[in]",$str_orig);

#substitui o valor da extensão destino na string
$str_alter=str_replace("{EXT_OUT}","$_GET[out]",$str_alter);

#Insere o valor do CallerID na string
$str_alter=str_replace("{CALLER}","VLM - "."$_GET[in]",$str_alter);

#Insere o contexto da chamada,em função
#do tamanho do número destino. Caso seja 3 digitos,
#significa que o número que se pretende contactar
#é uma extensão interna,caso contrário, é
#um número externo
$len_ext=strlen($_GET[out]);
if ($len_ext==3){
$str_alter=str_replace("{CONTEXT}","Local",$str_alter);
}
else {
$str_alter=str_replace("{CONTEXT}","Outbound",$str_alter);
}

#Criação do call file no diretório corrente com o prefixo "VLM_"
$call_file=tempnam(".","VLM_");
#Alteração das permissões do ficheiro,
#por forma a poder ser lido pelo sistema Asterisk
chmod($call_file,0644);

#Inserção da string alterada no call file
file_put_contents($call_file,$str_alter);

#Mover o call file para o diretório especificado
#por forma a despoletar a chamada
system("mv $call_file /var/spool/asterisk/outgoing");

?>
```

Figura 4.6: Conteúdo dos ficheiro template e originate.php

Quando este ficheiro é solicitado (por HTTP), e com os parâmetros devidamente inseridos no URL, são processados os dados por forma a criar o *call file* pretendido, ficheiro este que por sua vez, é movido para o diretório de spool do Asterisk, que vai reconhecê-lo como tal, e estabelecer a chamada pretendida.

4.4 Criação do Módulo instalável Click2Call

Como já foi referido anteriormente,na introdução ao SugarCRM, esta plataforma está organizada em módulos. Um dos pontos fortes do Sugar é precisamente fornecer a possibilidade de serem construídos módulos adicionais e personalizados. Percebendo o princípio de funcionamento dos módulos, é possível criar um ficheiro zip com uma determinada estrutura, e carregá-lo recorrendo à ferramenta *Studio,* para instalação.

Para maior facilidade na forma de colocar esta aplicação em funcionamento, foi construído um módulo instalável no sugar. Será explicado nesta secção, o processo de construção deste módulo.

Figura 4.7: Estrutura de diretórios para a criação do módulo instalável

4.4.1 O ficheiro manifest.php

Este é o ficheiro *core* para criação de um módulo instalável. Tem de estar localizado na raíz do zip, e aqui está contida toda a informação sobre os ficheiros restantes.

Começando pelo ficheiro manifest.php, este ficheiro tem obrigatoriamente de conter dois vetores, sendo que um deles é precisamente o array $manifest, este contendo outros sub-arrays. O primeiro sub-array define as versões do sugar para que o módulo foi concebido, mas pode ainda ser compatível com versões anteriores, a não ser que se defina especificamente as únicas versões que se pretende que sejam compatíveis. A versão atual do Sugar é a 6.5.16,

no entanto, seria possível definir este módulo como aceitável para todas as versões superiores a 6.0.0, indicando no array *acceptable_sugar_versions* o valor 6.X.X. Outro sub-array é o *acceptable_sugar_flavors,* onde se deve introduzir as edições com que pretende que o módulo seja instalável, CE para a versão comunidade, ENT para Enterprise e PRO para profissional.

O array $manifest contém ainda diversas variáveis predefinidas, onde devem ser definidas algumas opções do módulo, nomeadamente o ficheiro README, para ser visualizado quando se instalar o módulo (se estiver definido), o autor, a descrição, o nome, a opção se é possível de ser desinstalado, se por defeito, as bases de dados associadas devem ser removidas quando for desintalado, entre outras.

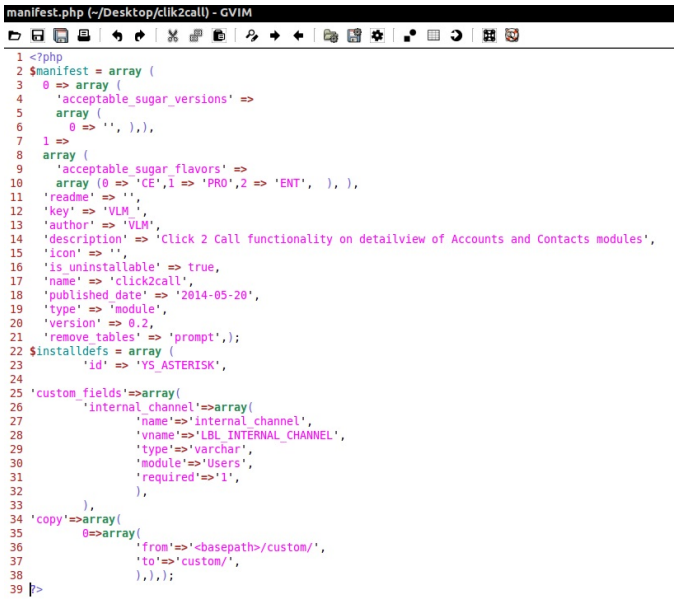


Figura 4.8: Ficheiro manifest.php

O segundo array principal e obrigatório neste ficheiro, é o array $installdefs que deve conter a informação sobre campos customizados que se pretenda adicionar a um (ou mais) módulos, e os diretórios eou ficheiros que devem ser copiados para o sistema, e para onde. Neste caso, apenas era pretendido adicionar o campo *internal_channel* ao módulo *Utilizadores*, para definir o canal de comunicação associado à extensão interna de cada utilizador. No entanto, pretendia-se ainda copiar os ficheiros detailviewdefs.php e view.list.php dos módulos *Entidades e Contactos* para os

respetivos diretórios de *custom/,* bem como o ficheiro editviewdefs.php do módulo *Utilizadores,* por forma a acrescentar a esta vista o campo customizado. Uma vez que este campo foi criado, necessita também de ser criada a sua identifiação no sistema, ou seja, o nome do campo que aparece no ecrã, e que é variável consoante o idioma escolhido pelo utilizador. Esta informação é inserida no ficheiro en_us.lang.php, que deve residir no diretório *custom/Extension/modules/Users/Ext/language/.* Para facilitar a operação de copiar ficheiros, foi criada a estrutura de diretórios pretendida, com todos os ficheiros incluídos, e no manifesto dá-se a instrução de copiar o diretório *custom* completo.

4.5 Construção do Módulo CONFIGURADOR

O módulo CONFIGURADOR foi criado com vista a guardar os valores do IP do servidor, e do nome do ficheiro originador, para serem facilmente introduzidos e alterados na plataforma, em caso de necessidades futuras. Não será alvo de análise detalhada, mas fica, no entanto, a informação que é um módulo que deriva da classe *basic* do Sugar, e foram utilizados os campos *name, e description* dos registos deste módulo, para guardar o nome e o valor destes dados. Realço o facto que estes registo devem ter, obrigatoriamente, os nomes 'ip_server' e 'call_file_originator', respetivamente, pois é por estes nomes que o módulo click2call faz a pesquisa na base de dados para aceder aos respetivos valores e gerar corretamente o URL.

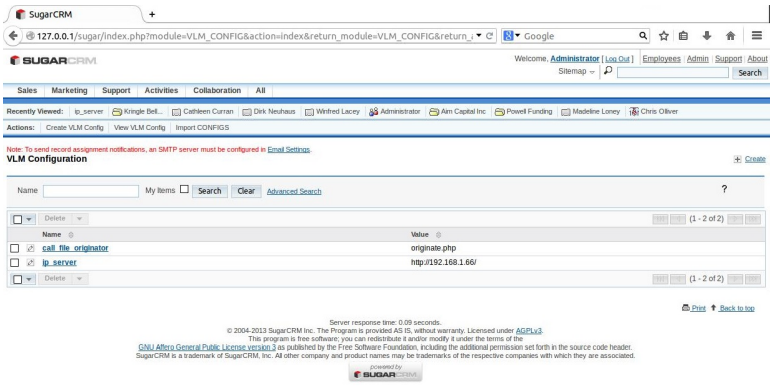


Figura 4.9: Módulo CONFIGURADOR - listview

Capítulo 5

Análise e Discussão de Resultados

Neste capítulo, iremos analisar, ao nível de sistema, os processos que estão por detrás da aplicação.

5.1 Montagem e caracterização do ambiente de testes

Foram utilizados, no sistema de testes, três PC's e um *SIP Hardware Phone* ligados em rede, sendo que um dos PC's atua como servidor, e onde está instalado o software indicado na tabela seguinte. O analisador de pacotes Wireshark foi utilizado para analisar toda a atividade da rede, com foco principal na troca de pacotes entre estações, nos momentos de pré estabelecimento de chamada, durante a chamada propriamente dita, e pós chamada.

Software	Versão
Ubuntu	14.04
Asterisk	12.1.1
SugarCRM	CE 6.5.16
Apache	2.4.7
MySQL	5.5.37
PHP	5.5.9
Wireshark	1.10.6

Tabela 5.1: Versões do Software instalado no PC servidor

Para simplificação de nomenclatura, daqui para a frente, o PC que atua como servidor será referido como PC1, e os PC's clientes como PC2 e PC3. O softphone no PC1 é o linphone, resgistado no Asterisk como linphone_pc1 (extensão associada < 101 >), para ser mais facilmente associado ao PC1. Nos PC's 2 e 3, foi instalado como cliente VoIP, o softphone Jitsi, que se verificou ser o que melhor funcionava em ambiente windows. Para manter a coerência e a facilidade na identificação dos clientes, o softphone foi registado como jitsi_pc2 e jitsi_pc3, com as extensões associadas de 102 e 103, para os PC's 2 e 3, respetivamente. O Hardphone foi registado como yealink, e foi atribuída a extensão 104. Estas extensões foram definidas no sistemas Asterisk, configurando os respetivos ficheiros, da forma apresentada na figura seguinte:

sip.conf (/etc/asterisk) - VIM 38x44

```
[general]

context=public
allowguest=yes
srvlookup=no
udpbindaddr=0.0.0.0
tcpenable=no

[phone](!)
type=friend
context=Local
host=dynamic
nat=force_rport,comedia
dtmfmode=auto
disallow=all
allow=g722
allow=alaw
allow=ulaw


[linphone_pc1](phone)
secret=linphonepc1
callerid=PC1 - Linphone

[jitsi_pc2](phone)
secret=jitsipc2
callerid=PC2 - Jitsi

[jitsi_pc3](phone)
secret=jitsipc3
callerid=PC3 - Jitsi

[yealink](phone)
secret=yealink
callerid=Hard yealink
```

extensions.conf + (/etc/asterisk) - VIM 46x44

```

[Local]

exten=>101,1,Dial(SIP/linphone_pc1)
exten=>102,1,Dial(SIP/jitsi_pc2)
exten=>103,1,Dial(SIP/jitsi_pc3)
exten=>104,1,Dial(SIP/yealink)
```

Figura 5.1: Definição das extensões e contexto no Asterisk

Indico de seguida as características dos PC's utilizados em ambiente de testes.

	PC1 (Servidor)	PC2	PC3
Marca	Toshiba	IBM	Acer
Modelo	Satellite A300	ThinkPad R52	TravelMate C300
Processador	Intel Core 2 Duo 2.4GHz x2	Intel Pentium 4 1.7GHz	Intel Pentium 4 1.6GHz
RAM	3GB	1.5GB	500MB
S.O.	Ubuntu 14.04	Win XP PRO SP3	Win XP PRO SP3

Tabela 5.2: PC's utilizados em ambiente de testes

Foi ainda utilizado neste âmbito, um router Thomson TG784n com Wi-Fi e 4 portas Ethernet. O PC1 e PC2 conectaram à rede por wireless, e o PC3 foi ligado à rede por cabo ethernet. Por forma a facilitar a manutenção, foram utilizados endereços IP fixos. Na tabela seguinte apresentam-se os IP's atribuídos, e qual o cliente softfone instalado em cada um deles.

Dispositivo	IP	Cliente
PC1	192.168.1.66	Softphone Linphone
PC2	192.168.1.74	Softphone Jitsi
PC3	192.168.1.75	Softphone Jitsi
Hardphone	192.168.1.73	Hardphone Yealink T22P

Tabela 5.3: Endereços IP atribuídos aos dispositivos de teste

Realça-se o facto de terem sido utilizados maioritariamente softphones para simulação, mas em ambiente real, o mais provável e mais funcional será a existência de hardware phones. No entanto, pretende-se apenas demonstrar

as fases de processamento de dados, que é equivalente em ambos os casos.

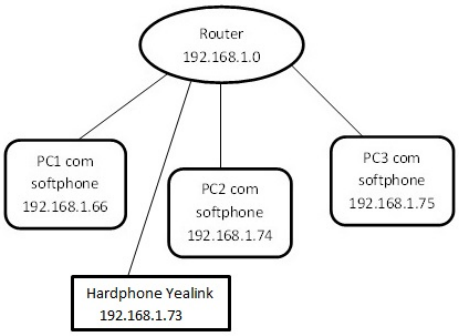


Figura 5.2: Esquema do ambiente de simulação

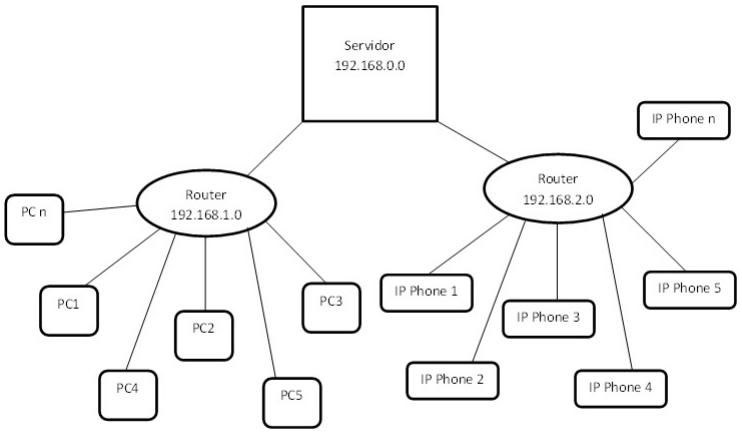


Figura 5.3: Esquema de ambiente real

Para dar início aos testes, foi necessário proceder à montagem da rede conforme já explicado anteriormente. Após a instalação do SugarCRM e de todos os componentes necessários ao seu correto funcionamento, foram também instalados na plataforma, os módulos criados anteriormente, click2call e CONFIGURADOR. Para verificar o sucesso da aplicação click-2-call, as extensões criadas no Asterisk (101,102,103) foram inseridas como sendo o número de telefone de três registos do módulo Contactos, e também de três registos do módulo Entidades. Cada um dos PC's estaria ligado na plataforma com o seu utilizador (foram usados os utilizadores teste que a plataforma trás

por defeito), e a cada um destes, foi atribuído um *internal_ channel* correspondente a linphone_pc1, jitsi_pc2, jitsi_pc3, e yealink. Neste contexto, apenas foi possível simular chamadas internas, pois para realizar chamadas para a rede exterior, seria necessário ter um fornecedor de serviços VoIP externo, situação que não se verificou viável, no entanto, é possível de entender que, estando a aplicação a funcionar em modo interno, seria muito fácil adicionar um contexto de chamadas externas no Asterisk, e encaminhar para este contexto, todos os clicks em números de comprimento igual a 9 (dígitos) por exemplo. Esta situação já está prevista no ficheiro call_file_originator.

Neste ambiente de testes, foram então verificadas as ligações possíveis, e foram analisadas as ocorrências, em termos de comunicação, durante o registo dos vários telefones ao sistema Asterisk, bem como antes, durante, e após, o estabalecimento de chamadas entre as várias extensões definidas. Para esta análise, recorreu-se ao software analisador de pacotes Wireshark, esperando obter-se um comportamento similar ao indicado na figura seguinte:

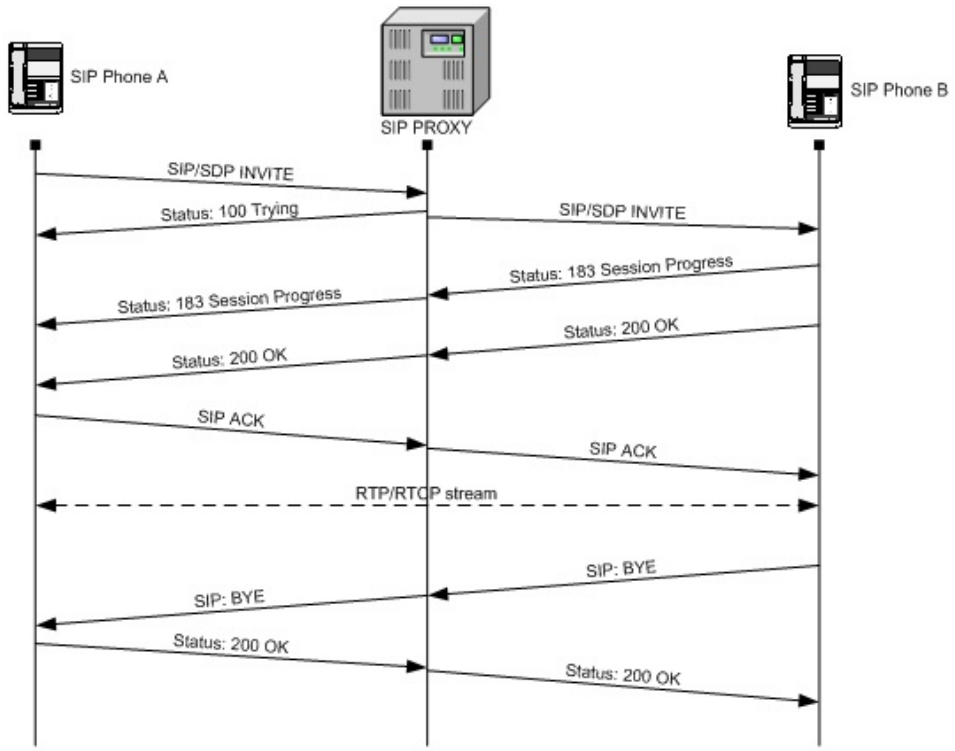


Figura 5.4: Esquema de interações SIP

Verificou-se o seguinte:

a) Registo dos telefones no sistema: É enviado um Pacote SIP do cliente para o servidor, solicitando o registo . É um pacote SIP request, que segue no porto 5060, que é o porto pré-definido do Asterisk. O sistema vai analisar a informação, e envia um resposta num novo pacote SIP, neste caso com o código 200, que significa OK. O cliente está então registado no sistema. O mesmo se passa com os outros clientes VoIP. Realça-se aqui o excelente tempo de resposta, que por análise dos pacotes captados com recurso ao wireshark, se verifica que entre o pedido de registo e a resposta, decorre cerca de 1ms.

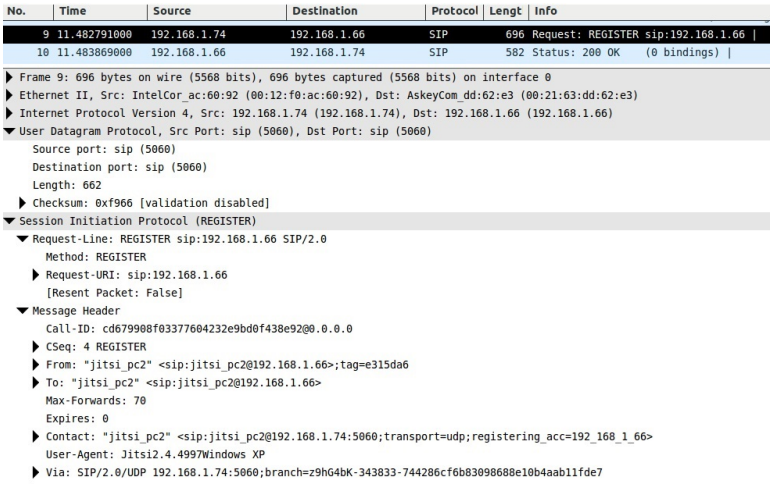


Figura 5.5: Pacote SIP Request para registo do softphone jitsi_pc2

Esta informação foi semelhante à obtida para o registo dos restantes clientes VoIP no sistema.

b) Envio de pedido e estabelecimento de chamada: Para estabelecimento de uma chamada telefónica, por exemplo, inciada na extensão 104 (yealink), para a extensão 102 telefone jitsi_pc2, o primeiro cliente envia um pacote sip invite, ao que o sistema responde com trying, visto que vai tentar contactar o segundo cliente. Imediatamente a seguir, envia um pacote SIP Invite para o cliente associado à extensão 102, que o sistema já informou ser o utilizador jitsi_pc2. Quando este recebe a informação que está a ser convidado para estabalecer uma chamada, envia um pacote SIP ringing ao servidor, que o reencaminha para o cliente que iniciou o

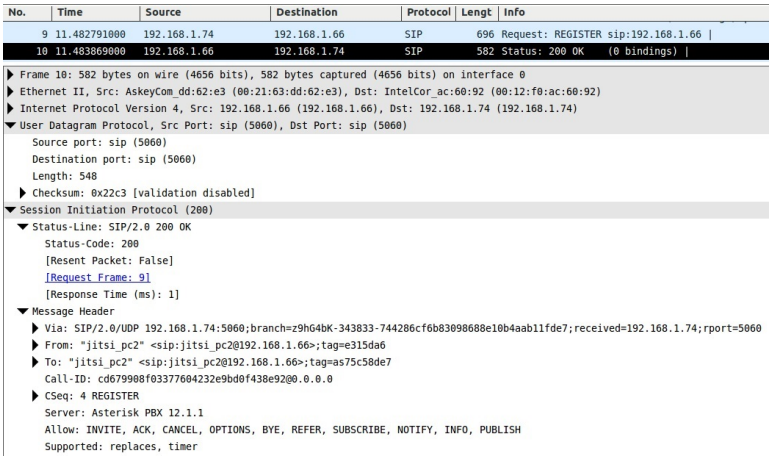


Figura 5.6: Confirmação de registo do softphone jitsi_pc2

processo de invite.Este processo vai sendo repetido até o segundo cliente atender a chamada. Por análise dos pacotes, recorrendo ao wireshark, verifica-se que desde o sip invite até sip ringing, decorrem cerca de 75ms. Quando a chamada é atendida no segundo cliente, inicia-se a troca de pacotes RTP, (RealTime Transport Protocol) entre os dois clientes do sistema, que se mantem até um deles dar indicação de que termina a chamada.

Na imagem 5.7, verifica-se o pedido de estabelecimento de chamada (invite), a informação de *100 Trying*, e a repetição de pacotes *180 Ringing*, até a chamada ser atendida no outro terminal.

c) Terminar a chamada: O cliente que procede ao término, envia um sip request BYE ao sistema, ao que o sistema responde com sip ok, informando que recebeu a informação do término da chamada, e informa de seguida o cliente no outro terminal, através de um SIP Request Bye. Este processo de término de chamada desde o pimeiro sip bye para o servidor, até à confirmação do segundo cliente, demora cerca de 78ms.

5.2 Continuidade do projeto

A próxima etapa deste projeto será realizar testes na rede de telefonia da empresa, por forma a concretizar a implementação desta funcionalidade na

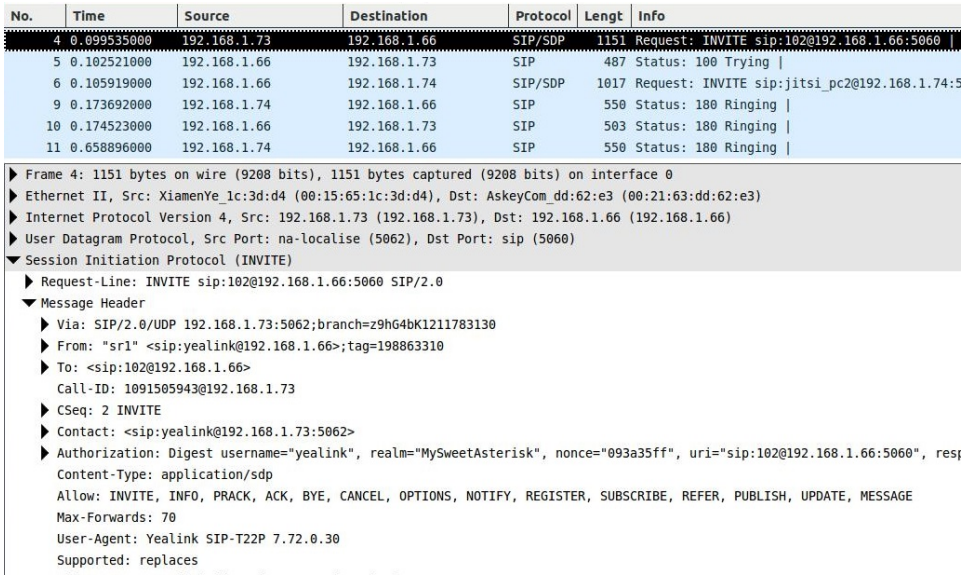


Figura 5.7: Pedido de início de chamada - Invite (extensão 104 para 102)

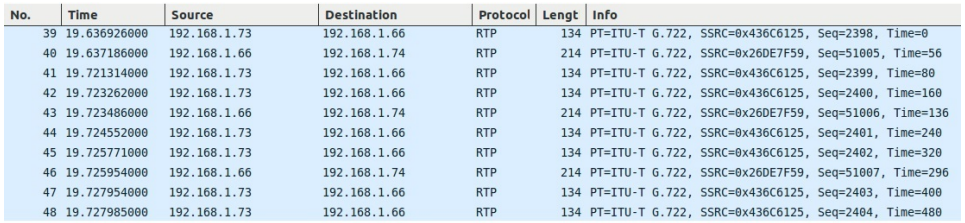


Figura 5.8: Pacotes RTP durante chamada (extensão 104 para 102)

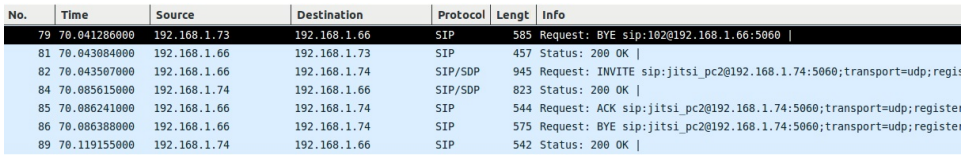


Figura 5.9: Pacotes trocados para terminar chamada

---

plataforma desenvolvida e utilizada pela VLM, onde já são também utilizados telefones VoIP. Quando se obtiver sucesso nesta implementação, faltará certamente otimizar algumas secções de código, por forma a não ter implicações no desempenho da plataforma, nem do próprio sistema de telefonia.

Pretende-se ainda acrescentar outras funcionalidades a esta aplicação, nomeadamente despoletar janelas de pop-up, quando o sistema receber chamadas de algum número existente na base de dados do Sugar, por forma ao utilizador poder incluír imediatamente após o término da chamada, um resumo da mesma, e o conteúdo da conversa, por forma a todos os utilizadores terem acesso, através do SugarCRM, a todos os contactos feitos com as entidades.

Como objetivos a longo prazo para este projeto, a idéia é que seja incluído também na distribuição SugarCRM customizada que a VLM implementa aos seus clientes. Neste caso, cada situação teria de ser analisada detalhadamente, para que se possa perceber até que ponto esta integração é implementável. Fatores condicionantes na implementação aos clientes, são, por exemplo, se o cliente já possui rede IP nas instalações, se possui telefones IP, se já tem fornecedor de serviços VoIP, etc... Também muito importante, é o facto de o cliente estar ou não disposto a ser desafiado pela mudança, no caso de não ter ainda nenhum serviço VoIP.

Capítulo 6

Conclusão

Conclui-se, com o sucesso dos testes que foram levados a cabo, que customizar uma soluçao como o SugarCRM e o Asterisk, apesar de serem estruturas bastantes complexas, é relativamente fácil, devido à organizaçao das suas estruturas. Tanto uma aplicação, como outra, são de extrema utilidade para as pequenas/médias empresas, pois com um CRM como o Sugar, gratuito, e totalmente customizável, é muito mais fácil gerir o relacionamento comercial com os clientes, facilitando um acompanhamento mais personalizado em todas as situações, o que pode dar uma vantagem competitiva muito grande. Uma vez que também aponta à melhoria, ajuda a reduzir custos supérfluos, e a manter o foco nos processos de negócio.

Simultaneamente, uma plataforma como o Asterisk, sendo também open-source, e customizável, revela-se também muito útil, pois permite manter todos os colaboradores em contacto de forma simples e rápida. Atualmente, serão raros os casos de empresas que não tenham ainda adotado uma central telefónica para as suas instalações. Escolhendo o Asterisk como solução, a redução de custos é significativa, pois não implica pagamento de licenças, nem de representantes de marcas, como é o caso dos sistemas proprietários.

A troca de pacotes de informação entre os terminais testados verificou-se de acordo com o esperado, e concluímos ainda que o sistema apresenta uma performance muito boa, tendo respostas na casa dos milisegundos.

Pelas suas características, funcionalidades, e facilidade de implementação, existem já diversas empresas no mercado que prestam serviços de consultoria nestes âmbitos, de telefonia digital, e de sistemas CRM com base nestas e outras ferramentas open-source. Desta forma, a sua existência passa

a ser cada vez mais conhecida, e a sua utilização continuará certamente em expansão durante os próximos tempos.

Bibliografia

[1] CRM Switch | URL *http://www.crmswitch.com/crm-industry/*

[2] CRM Wiki | URL *http://pt.wikipedia.org/wiki/Customer_relationship_management*

[3] Sales Force web | URL *http://www.salesforce.com/sales-cloud/overview/index*

[4] ZohoCRM web | URL *http://www.zoho.com/crm/zohocrm-pricing.html*

[5] SugarCRM web | URL *http://www.sugarcrm.com/pricing*

[6] WEBCRM online | URL *http://www.webcrm.com/pt/products*

[7] Microsoft web | URL *http://www.microsoft.com/pt-pt/dynamics/crm.aspx*

[8] Oracle web| URL *http://www.oracle.com/us/products/applications/siebel/crm-technology/overview/index.html*

[9] Suite web | URL *http://suitecrm.com/index.php?option=com_content&view=article&id=141&Itemid=1299*

[10] Zurmo web | URL *http://zurmo.com/pricing*

[11] VoIP Wiki | URL *http://pt.wikipedia.org/wiki/Voz_sobre_IP*

[12] Christina Hattingh, Darryl Sladden, ATM Zakaria Swapan: *SIP Trunking*, Cisco Press (2010)

[13] Optitelecom - Consultoria em serviços VoIP | URL *http://www.optitelecom.pt*

[14] VoIP Software Wiki | URL *http://pt.wikipedia.org/wiki/Anexo:Comparaç~ao_de_software_VoIP*

[15] Telefones SIP Proprietários | URL *http://www.onedirect.pt/telefones/telefones-ip-sip-proprietarios*

[16] Telefones SIP Livres | URL *http://www.onedirect.pt/telefones/telefones-ip-sip-livre*

[17] SIP Wiki | URL *http://pt.wikipedia.org/wiki/Protocolo_de_Inicia%C3%A7%C3%A3o_de_Sess%C3%A3o*

[18] VoIP-Info web | URL *http://www.voip-info.org/wiki/view/SIP*

[19] Alan B. Johnston: *SIP - Understanding the Session Initiation Protocol 2nd edition*, Artech House (2003)

[20] Gonzalo Camarillo: *SIP Demystified*, McGraw-Hill (2001)

[21] Apache web | *URL http://www.apache.org/*

[22] Ken Coar, Rich Bowen : *Apache Cookbook 2nd Edition*, O'Reilly (2008)

[23] John Mertic : *The Definitive Guide to SugarCRM - Better Business Applications*, Apress (2009)

[24] Community : *Sugar Developer Guide 6.5*, Sugar Documentation (2012) | URL `http://support.sugarcrm.com/02_Documentation/04_Sugar_Developer/Sugar_Developer_Guide_6.5/`

[25] John Mertic : *Building on SugarCRM*, O'Reilly (2011)

[26] Asterisk web | *URL http://www.asterisk.org*

[27] Asterisk Docs web | *URL http://www.asterisk.org/community/documentation*

[28] Russel Bryant, Leif Madsen, Jim Van Meggelen : *Asterisk - The Definitive Guide 4th edition*, O'Reilly (2013)