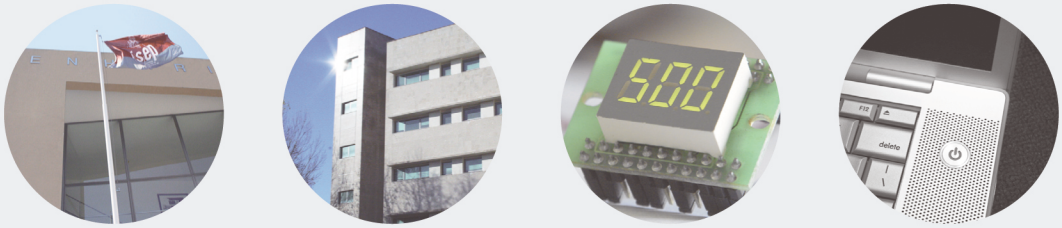
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



OWNSERVER: SISTEMA DOMÉSTICO DE PRESTAÇÃO DE SERVIÇOS CLOUD

JOÃO PEDRO BAPTISTA MARTINS BARROS MONTEIRO
Novembro de 2015

POLITÉCNICO
DO PORTO

OWNSERVER: SISTEMA DOMÉSTICO DE PRESTAÇÃO DE SERVIÇOS CLOUD

João Pedro Baptista Martins Barros Monteiro

Departamento de Engenharia Electrotécnica

Mestrado em Engenharia Electrotécnica e de Computadores

Área de Especialização em Telecomunicações

2015

Relatório elaborado para satisfação parcial dos requisitos da Unidade Curricular de Tese/Dissertação do Mestrado em Engenharia Electrotécnica e de Computadores

Candidato: João Pedro Baptista Martins Barros Monteiro, Nº 1060394, 1060394@isep.ipp.pt

Orientação científica: Prof. Doutor Jorge Botelho da Costa Mamede, jbm@isep.ipp.pt

Departamento de Engenharia Eletrotécnica

Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização em Telecomunicações

2015

Agradecimentos

Este trabalho representa não só o final deste mestrado, mas também um fruto dos valores que aprendi durante a minha campanha no ensino superior. Este fez-me aprender muito e graças a esses conhecimentos posso agora continuar o meu caminho, confiante de que conseguirei vencer no mundo do trabalho. Esta tese não seria possível sem o apoio de algumas pessoas às quais gostaria de agradecer. Agradeço desde já à minha família, à minha mãe Maria de Fátima, ao meu pai Francisco, ao meu irmão também Francisco e à minha avó Rosa, por todo o apoio que me têm dado neste período árduo e exigente. Agradeço a todos os meus amigos que não me têm visto tantas vezes como de costume e a todos os colegas que me ajudaram a chegar até aqui. Não posso também de deixar de agradecer ao meu orientador Jorge Mamede, que me ajudou a navegar nos mares revoltos das teses de mestrado e me direcionou no sentido certo quando me desviava pelos caminho errado. Sinto que um ciclo importante da minha vida está a terminar e com esta tese pretendo fecha-lo da melhor maneira.

Resumo

Neste trabalho foi considerada a possibilidade de incorporar serviços remotos, normalmente associados a serviços *web* e *cloud computing*, numa solução local que centralizasse os vários serviços num único sistema e permitisse aos seus utilizadores consumir e configurar os mesmos, quer a partir da rede local, quer remotamente a partir da Internet. Desta forma seria possível conciliar o acesso a partir de qualquer local com internet, característico nas *clouds*, com a simplicidade de concentrar num só sistema vários serviços que são por norma oferecidos por entidades distintas e ainda permitir aos seus utilizadores o controlo e configuração sobre os mesmos.

De forma a validar que este conceito é viável, prático e funcional, foram implementadas duas componentes. Um cliente que corre nos dispositivos dos utilizadores e que proporciona a interface para consumir os serviços disponíveis e um servidor que irá conter e prestar esses serviços aos clientes. Estes serviços incluem lista de contactos, mensagens instantâneas, salas de conversação, transferência de ficheiros, chamadas e conferências de voz e vídeo, pastas remotas, pastas sincronizadas, *backups*, pastas partilhadas, VoD (Video-on Demand) e AoD (Audio-on Demand). Para o desenvolvimento do cliente e do servidor foi utilizada a *framework* Qt que recorre à linguagem de programação C++ e ao conjunto de bibliotecas que possui, para o desenvolvimento de aplicações multiplataforma. Para as comunicações entre clientes e servidor, foi utilizado o protocolo XMPP (*Extensible Messaging and Presence Protocol*), pela forma da biblioteca qxmpp e do servidor XMPP *ejabberd*. Pelo facto de conter um conjunto de centenas de extensões atualmente ativas que auferem funcionalidades como salas de conversação, transferências de ficheiros e até estabelecer sessões multimédia, graças à sua flexibilidade permitiu ainda a criação de extensões personalizada necessárias para algumas funcionalidades que se pretendeu implementar. Foi ainda utilizado no servidor a *framework* ffmpeg para suportar algumas funcionalidades multimédia.

Após a implementação do cliente para Windows e Linux, e de implementar o servidor em Linux foi realizado um conjunto de testes funcionais para perceber se as funcionalidades e

seus mecanismos funcionam corretamente. No caso onde a análise da performance e do consumo de recursos era importante, foram realizados testes de performance e testes de carga.

Palavras-Chave

Sistema doméstico, serviços, XMPP, Qt, cliente, servidor, multiplataforma, acesso remoto, comunidades, comunicação, multimédia, Cloud.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

XMPP	–	Extenssible Message and Presence Protocol
JID	–	Jabber Identifier
IP	–	Internet Protocol
XML	–	eXtensible Markup Language
VoD	–	Video-on Demand
AoD	–	Audio-on Demand
P2P	–	Peer-to-Peer
C3P	–	Consumer-centric cloud portal
FIB	–	Forwarding Information Base
SVC	–	Scalable video coding
FEC	–	Raptor forward error correction
UMB	–	Universal Media Bridge
UPnP	–	Universal Plug and Play
VoIP	–	Voice over IP
IPTV	–	Internet Protocol Television
LTE	–	Long-Term Evolution
WiMax	–	Worldwide Interoperability for Microwave Access

UWB – Ultra-Wide Band

SIP – Session Initiation Protocol

QoS – Quality of Service

QoE – Quality of Experience

NAS – Network Attached Storage

CCN – Content centric network

M2M – Machine-to-Machine

s2s – Server-to-server

API – Application Programming Interface

FTTH – Fiber to the Home

iDTV – Integrated digital television

SMS – Short Message System

M2M – Machine-to-Machine

FTP – File Transfer Protocol

SCP – Secure Copy Protocol

SMB – Server Message Block

NFS – Network File System

MUC – Multi User Chatroom

NAT – Network Address Translation

IoT – Internet of Things

STUN – Session Traversal Utilities for NAT

TURN – Traversal Using Relays around NAT

URL Uniform Resource Locator

1. INTRODUÇÃO

1.1. CONTEXTUALIZAÇÃO

A Internet é um meio que tem vindo a preencher cada vez mais a vida das pessoas. Graças à sua flexibilidade em possuir todo o tipo de serviços, ao acesso cada vez mais alargado, maiores velocidades de ligação e à sua capacidade de inovação, permite novos níveis de interação e novos tipos de serviços. Conceitos como web 2.0, redes P2P (*peer-to-peer)* e *cloud computing* são alguns exemplos de novas tendências tecnológicas que providenciam um novo grau de interação com serviços web. Destes conceitos, uma das tendências mais evidentes nos dias de hoje é o *cloud computing*.

A computação em nuvem é um modelo computacional que considera a passagem do processamento de aplicações que convencionalmente é efetuado localmente para um servidor remoto. Este conceito conta com a vantagem dos serviços serem processados remotamente, reduzindo substancialmente os recursos exigidos da aplicação por parte dos clientes. Herda também a vantagem dos serviços web que possibilita o acesso a estes remotamente a partir de qualquer local, desde que tenha acesso à internet. Desde serviços como o Google Drive e Dropbox até ao Office365 e Adobe Creative Cloud, são exemplos de serviços que podem ser acedidos remotamente e onde os servidores remotos executam a maior parte do processamento.

Como os exemplos anteriores auferem, existe todo o tipo de serviços que podem ser efetuados remotamente ou que ao serem suportados por uma entidade permanentemente

presente na internet, possibilitando o seu consumo e a sua diversidade no que toca a aplicações possíveis. Infelizmente estas aplicações estão associadas a entidades distintas e mesmo que algumas desta possuam vários serviços, os utilizadores acabam por ter de se gerir vários registos para todos os serviços que consomem.

A pensar na vantagem das *clouds* e serviços web e na dispersão destes serviços por diversas entidades, leva a que surgi a ideia de criar não só um sistema que pudesse ser acedido a partir da internet e que ao mesmo tempo concentrasse numa só aplicação, um conjunto alargado de serviços. Para além disso este sistema poderia ser local, possibilitando aos seus utilizadores a sua configuração e a retensão dos dados localmente sob controlo dos utilizadores, em vez destes ficarem alojados remotamente, nos servidores das variadas entidades que prestam os serviços.

1.2. OBJETIVOS

O objetivo deste projeto é o desenvolvimento e implementação de uma aplicação cliente multiplataforma e uma aplicação servidora que em conjunto consigam conciliar o acesso remoto a partir da internet, a concentração e consumo de diversos serviços a partir de uma única interface funcional, intuitiva e prática, bem como permitir aos seus utilizadores a configuração do sistema e o alojamento dos seus dados no servidor local. A ideia é criar uma base funcional que comprove a viabilidade de implementar e concentrar um conjunto de serviços num servidor local e disponibilizar os mesmos numa única interface.

1.3. ORGANIZAÇÃO DO RELATÓRIO

No Capítulo 2, é feito um estudo aprofundado da área em redor deste projeto para que se possa perceber as tendências, os desenvolvimentos e o interesse que existe em redor das tecnologias da área. No capítulo 3, utiliza-se as conclusões do capítulo anterior de forma a perceber que serviços implementar no projeto e que ferramentas utilizar para a implementação do mesmo. No capítulo 4, é apresentada a arquitetura do projeto, incluindo a descrição do cliente e do servidor, bem como a nomenclatura de rede. No capítulo 5, é feita a documentação do desenvolvimento das aplicações e a descrição dos vários processos que suportam os serviços implementados e das interfaces que acompanham o cliente. Já no capítulo 6, é documentado o conjunto de testes efetuado de forma a verificar que os serviços funcionam corretamente, e que a sua performance é aceitável. Por fim no capítulo 7, são

tiradas as conclusões sobres os testes efetuados, é feita uma apreciação sobre a viabilidade e a potencialidade do projeto no mundo real e são ainda apresentados desenvolvimentos futuros e melhorias que possam ser aplicados na solução desenvolvida.

2. ESTADO DA ARTE

O projeto insere-se no contexto de desenvolvimento de servidores domésticos e sistemas locais, sendo assim importante perceber as implicações e as tendências tecnológicas deste ramo. Entender como tecnologias de rede e conceitos como a eficiência energética, influenciam o desenvolvimento nesta área, ajudam a estabelecer uma base de princípios que podem ser utilizados na implementação deste projeto. Existem também novos paradigmas ligados à computação que tiram proveito da Internet para permitir a prestação de serviços que anteriormente não existiam ou não eram viáveis. Casos como o *Peer-to-Peer* (P2P) e o *cloud computing* são cada vez mais aplicados na criação de serviços que suportem novas tendências na era atual de informação. Nesta secção serão explicadas as tendências e produtos que fazer parte do contexto deste trabalho, dando assim a base para o seu desenvolvimento.

2.1. TECNOLOGIAS

No domínio dos paradigmas de computação, graças ao avanço tecnológico dos computadores pessoais, nas tecnologias de redes locais e nos dispositivos domésticos, como *routers*, sistemas NAS e servidores pessoais, a computação local é um domínio computacional proeminente na atualidade, pois permite aos utilizadores a adição de funcionalidades locais na residência e oferece flexibilidade nas configurações, bem como o

controlo dos equipamentos e dos dados que na rede transitam. Embora este modelo tenha sido o padrão nas tendências tecnológicas, surgiram nos últimos anos novos paradigmas de computação e de redes, que modificaram o panorama no domínio informático. Tecnologias como o *cloud computing* têm sofrido uma evolução estonteante, devido não só à natureza prática dos serviços, mas também aos custos associados, principalmente para empresas, quando comparados com uma infraestrutura local. Outras arquiteturas de rede como o *Peer-to-Peer* também tem vindo cada vez mais a ser utilizada, devido à sua natureza descentralizada e comunitária, para a execução de serviços e as redes CCN (*Content Centric Network*), que fazem parte de um novo conceito de rede que se foca na distribuição de conteúdo, devido a uma percentagem considerável dos dados transmitidos na Internet conteúdo distribuído, como filmes, músicas e transferências de ficheiros [1] . Nas próximas secções são explicadas as tendências e pesquisas na área de cada uma destas tecnologias.

2.1.1. PEER-TO-PEER

Em Peer-to-Peer a grande tendência é o desenvolvimento de soluções que tirem proveito da escalabilidade do P2P para distribuição de conteúdo. Grande parte dos esforços centram-se na utilização de rede hibridas de CDN (Content Delivery Network) e P2P, para tirar partido de um sistema de distribuição estruturado, como no caso do CDN, com a escalabilidade e flexibilidade das redes P2P. Para além da utilização de redes hibridas, existem três grandes tendências de investigação. Distribuição de conteúdo, distribuição de *video on demand* (VoD) e distribuição de *live stream*, utilizando redes de servidores em conjunto com utilizadores. Em [2] é feita uma pesquisa na utilização da combinação entre redes CDN e P2P, para a distribuição de vídeo e áudio.

A pesquisa na utilização de redes P2P para a distribuição de conteúdo multimédia é o principal foco no desenvolvimento do P2P. Pesquisas efetuadas a esta nomenclatura de rede, confirmam a capacidade do P2P ser utilizado para distribuir *video-on-demand*, onde o *stream* de múltiplas fontes de vídeo são asseguradas pela escalabilidade da rede [3]. Outros métodos podem ser utilizados para melhorar a distribuição de *video-on-demand*, utilizando pré-carregamento, filas de espera e *caching* entre utilizadores [4]. A localização e colocação do conteúdo a partilhar nas redes P2P para que haja uma boa distribuição de carga e gestão de *uplink* dos utilizadores também é um fator importante, que obriga a uma análise [5]. Outros métodos podem ainda envolver hierarquias de utilizadores, onde os nós mais eficientes na

distribuição de conteúdo estão no topo da hierarquia, permitindo uma distribuição mais estruturada de conteúdo [6].

2.1.2. *CLOUD COMPUTING*

O *cloud computing* é um princípio que embora já exista à algum tempo, em termos práticos conceito e tal como é o caso de outras tecnologias existentes no mercado [7], só nos últimos anos é que têm surgido serviços que aproveitam ao máximo este conceito. Existe um esforço para perceber o conceito do *cloud computing*, arquiteturas a adotar e *standards* que permitam uma convergência das tecnologias relacionadas com a *cloud*. Em [8] são apresentados os conceitos do *cloud computing*, a sua arquitetura, bem como os desafios esperados nesta área. Neste momento existem apenas algumas arquiteturas de referência, sendo necessário para a redução de custo tentar convergir para uma única arquitetura de referência [9]. Para além da arquitetura para a estrutura da *cloud*, também a criação de arquiteturas para aplicações de *cloud computing* é importante, permitindo assim o desenvolvimento mais eficiente e compatível de aplicações, de forma sistemática [10]. É também necessário analisar de modo quantitativo, a capacidade da *cloud* suportar serviços de *streaming* de multimédia, já que estes dependem da capacidade dos servidores, da largura de banda alocada a cada utilizador, das condições da rede e da capacidade de processamento do dispositivo do utilizador [11]. Embora o *cloud computing* seja ainda um conceito em expansão, já existem soluções comerciais disponíveis, como a Amazon EC2, IBM Smart Cloud, Google App Engine e Windows Azure. Para além de soluções comerciais, existem também algumas soluções *open source*, que permitem o desenvolvimento de soluções para *cloud*s privadas, como Hadoop, Eucalyptus, openNebula e Nimbus [12].

A grande área de investigação do *cloud computing* é a segurança e a privacidade. Em primeiro lugar, sendo uma tecnologia nova, não existem *standards* ou regulamentos de gestão de dados e mecanismos de segurança creditados, permitindo aos provedores de serviços de *cloud computing*, a utilização de medidas de proteção próprias, possibilitando a criação problemas de segurança [13]. Outra questão inerente à segurança e privacidade está no facto da informação dos utilizadores estar hospedada remotamente. Se não existir a confiança por parte dos utilizadores que os dados serão tratados com confidencialidade, não utilizarão o serviço [14]. Aliás, estima-se que a desconfiança de utilizadores e empresas na *cloud* é o grande entrave para a adoção definitiva do *cloud computing* [15]. As questões de segurança relativas ao *cloud computing* passam por múltiplo arrendamento, elasticidade,

disponibilidade da informação, gestão segura de informação, integridade e privacidade da informação e uma federação de *cloud* segura [16]. Vulnerabilidades tais como ameaças internas, ataques esternos, interrupções de serviços, problemas ligados ao múltiplo arrendamento e perda de controlo, são uma questão séria que os prestadores de serviços de *cloud computing* têm de ter em consideração [17]. De acordo com [18] as partes mais interessadas no que diz respeito às ameaças de segurança são os prestadores de serviços *cloud* e os seus clientes, pois a perda de privacidade, informação e controlo sobre a plataforma contratada, faz os clientes não voltarem a contratar o serviço [19]. De modo a aumentar a confiança na *cloud*, é necessário criar mecanismos de segurança que consigam colmatar as ameaças à segurança do *cloud computing* [20]. Não chega utilizar um único mecanismo de segurança, e grande parte dos mecanismos utilizados em outros sistemas, não são aplicáveis à *cloud*. Uma estratégia possível pode passar pela combinação de técnicas de encriptação, autenticação de segurança e uma política de controlo de acesso pode ser uma solução viável para combater os problemas da *cloud* [21]. Uma técnica de encriptação possível é a utilização de assinaturas digitais, baseadas em algoritmos de encriptação RSA [22]. Outras soluções podem passar pelo controlo de acesso aos serviços, utilizar contramedidas, como particionamento de dados e migração, bem como análise e alocação dos recursos [23].

Outro tópico atualmente investigado nesta área é o mobile *cloud computing*. Este conceito mistura as vantagens do *cloud computing*, com a mobilidade e flexibilidade da computação móvel. As tendências nesta área consistem na pesquisa de métodos eficientes que resolvam os problemas da capacidade limitada, conectividade e qualidade de serviço da computação móvel [24]. Outra questão essencial é o efeito que a evolução desta área tem no desenvolvimento de serviços de *cloud*, *middlewares*, aplicações móveis e sistemas operativos móveis [25]. Estão a ser estudadas *frameworks* que utilizam a capacidade da computação móvel em nuvem, para permitir um funcionamento eficiente desta arquitetura e uma distribuição da computação na execução de aplicações. No caso de [26], é sugerida uma *framework* que permite dividir o processamento de aplicações entre o dispositivo móvel e a plataforma de *cloud computing*, utilizando um modelo de duas etapas, onde na primeira é feita a alocação de recursos local que serão utilizados pelo dispositivo, e numa segunda fase, é feita a alocação de recursos no servidor de *cloud*, permitindo assim uma melhor operação de processamento. Já em [27] é apresentada uma *framework* que permite a melhoria no processamento de aplicações de *stream* de dados, utilizando a *cloud* não só para ajudar no

processamento, mas também possibilitar a partilha de recursos para ajudar outros dispositivos que utilizam o mesmo serviço, ajudando assim na melhor gestão dos recursos da *cloud*,

2.1.3. PERSONAL CLOUD COMPUTING

A computação em nuvem pessoal é o conceito de misturar o mobile *cloud computing* com *clouds* virtuais de dispositivos, que se traduz numa transição de computação baseada em dispositivos para computação ligada à informação, onde são fornecido serviços adequados à situação e aos dispositivos utilizado, de forma dinâmica. É um modelo orientado ao utilizador, onde o conteúdo de um indivíduo é disponibilizado em qualquer momento, em qualquer local e a partir de qualquer dispositivos, seja a partir de um navegador ou de uma aplicação instalada.

A grande tendência na computação em nuvem pessoal é a possibilidade de utilizar qualquer dispositivo para aceder aos serviços da *cloud*, possibilitando um ambiente ubíquo onde em vez da experiência do utilizador ser considerada apenas para um dispositivo, esta é considerada de forma consistente para um conjunto alargado de dispositivos. Em [28] é apresentado o consumer-centric cloud portal (C3P), que permite o acesso e a alocação de serviços e aplicações de *cloud* por parte de um alargado conjunto de dispositivos, que passam por *smartphones*, *tablets*, *Smart* TVs e leitores de música. Já em [29] é feita uma análise da implementação de aplicações no C3P, possibilitando a implementação de aplicações multiplataforma a partir de uma interface baseada em serviços web RESTful. No desenvolvimento das aplicações em si, é importante que estas sejam desenvolvidas para funcionarem em múltiplas plataformas e dispositivos. Uma das soluções possíveis é a utilização de aplicações hibridas que misturam as funcionalidades de uma aplicação nativa com uma interface web. As soluções híbridas podem ser desenvolvidas com o auxílio de ferramentas de desenvolvimento para soluções multiplataforma, como PhoneGap, Rhomobile, Appcelerator ou Qt, entre outros, acelerando o processo de desenvolvimento deste tipo de soluções.

Exemplos de aplicações no Personal Cloud Computing podem passar por sistemas de armazenamento de ficheiros distribuído, não só permitindo o acesso a ficheiro num único sistema (servidor), mas também possibilitando a partilha de ficheiros entre os vários dispositivos do utilizador como se os ficheiros estivessem disponíveis localmente [30]. O

facto de os utilizadores atuais possuírem múltiplos dispositivos, possibilita o desenvolvimento de soluções que tiram partido dessa capacidade distribuída, como no exemplo de [31], onde foi apresentado um mecanismo que utiliza essa capacidade distribuída em conjunto com a utilização discos virtuais, para implementar um mecanismo de *backup* que permite proteger os dados de eventuais ocorrências, sem utilizar serviços de armazenamento online, de modo a não criar possíveis questões de privacidade de dados. Para além de aplicações utilizarem múltiplos dispositivos para melhorarem o rendimento ou a performance, em certos caso é considerada a partilha de recursos físicos entre vários dispositivos. Outro tipo de aplicações são os assistentes pessoais da *cloud*, soluções autónomas que assistem o utilizador na execução de serviços, reduzindo a complexidade exigida ao dispositivo. Estas soluções também permitem uma ligação não persistente pelo facto de o assistente processar os pedidos e serviços em plano de fundo, permitindo assim ao utilizador, receber o resultado ou o serviço apenas quando pode ou necessita.

2.1.4. CONTENT CENTRIC NETWORKS

As redes CCN (Content Centric Networks) constituem uma arquitetura de rede considerada para o futuro da Internet, onde a distribuição de conteúdo é muito maior do que sessões de comunicação. Daí estas redes estarem a ser investigadas, pelo facto de tornarem a partilha de conteúdo entre comunidades, mais fácil e intuitiva. As grandes áreas de pesquisa nesta área são os mecanismos de gestão, *routing* e *forwarding* de informação, mecanismos de cache e a utilização conjunta de redes CCN com dispositivos móveis.

O *routing* é muito importante em redes CCN. Por esse motivo estão a ser pesquisado métodos que melhorem a eficiência destes procedimentos. Um exemplo é o caso de [32], onde é sugerido um mecanismo de *re-routing* de um andar, que cria um canal de redundância no caso da ligação principal falhar. Já no capítulo do *forwarding*, também existem alguns desenvolvimentos nessa área. Em [33] é apresentado um mecanismo que acelera a resolução de tabelas de nome FIB (*Forwarding Information Base*) utilizando paralelismo, de forma a reduzir o tempo necessário para a seleção do servidor de cache que irá enviar os dados. Em outros casos, é apresentado um motor de *forwarding*, implementado especialmente tendo em consideração o volume e métodos de distribuição das redes CCN [34]. Outra questão também pesquisada é o facto de as rede CCN funcionarem de diferente maneira do TCP, sendo necessário o estudo de casos em que ambas as tecnologias sejam utilizadas em simultâneo e interajam entre si [35]. As redes CCN dependem bastante de sistemas de cache

para a distribuição de conteúdo, onde é necessária uma alocação bem conseguida de recursos, para que a distribuição de conteúdo seja eficiente. Estudos e pesquisas estão a ser feitas de modo a criar uma estrutura de dados que seja eficiente na gestão de cache em redes CCN [36]. Já em [37] é apresentado um esquema de gestão de cache progressivo, melhora a gestão de cache, dependendo da fonte da informação. Outro esquema de gestão de cache utiliza mecanismos de pesquisa de segmentos com a hierarquia das redes CCN, para reduzir o tempo de *download* e aumentando a quantidade de conteúdo permitido na rede CCN [38]. Todas estas melhorias na eficiência de gestão de cache resulta não só de um aumento de eficiência de entrega de informação, mas também o aumento da eficiência energética, reduzindo assim os custos associados à operação destas redes [39]. A robustez do sistema de cache também é importante, sendo a criação de métodos que possam proteger o sistema de cache contra ataques de poluição de cache, importante [40]. Devido à sua natureza e arquitetura, outra grande questão de segurança prende-se com a vulnerabilidade das redes CCN em relação a ataques DoS, que a partir da utilização de ataques de inundação de interesse, podem causar perturbações na qualidade do serviço prestado [41].

Outro ponto de investigação é a interação de redes CCN com dispositivos móveis. O desenvolvimento de esquemas de gestão móveis permitem que dispositivos móveis consigam não só consumir recursos dessas redes, mas também criar e distribuir conteúdo [42]. Para além de esquemas de gestão, soluções baseadas em servidores *proxy* [43]. Este tipo de soluções utilizam servidores *proxy* CCN para compensar as ligações instáveis e movimentos que os dispositivos móveis sofrem com o seu uso .

Embora este conceito de rede seja bastante recente, já existem algumas aplicações que utilizam redes CCN para criar comunidades de utilizadores. As aplicações mais interessantes são as chamadas comunidades virtuais privadas ou *cloud*s privadas virtuais. No caso das comunidades virtuais, cada utilizador tem uma comunidade de dispositivos. Este grupo de dispositivos está associado a esse utilizador e quando algum recurso é partilhado, este estará associado ao utilizador e não ao dispositivo em si [44].

2.2. CASAS INTELIGENTES

Com o avanço tecnológico, o surgimento de tecnologias como o IPTV, VoIP e *cloud computing*, bem como a Internet das coisas e as comunicações M2M, criam o ambiente necessário para que as habitações atuais se tornem inteligentes e interativas.

A aposta em plataformas de serviços web para casas inteligentes é evidente com as vantagens de acesso em qualquer local remoto, em qualquer dispositivo com um navegador web. Para que isso aconteça, é necessário o desenvolvimento de uma arquitetura orientada a serviços, permitindo assim, controlar todos os equipamentos domésticos, a partir de um único dispositivo [45]. Existem também *frameworks* que utilizam a *cloud* como apoio, para automação doméstica [46]. Este método permite ao utilizadores a partir de terminais e dispositivos móveis, como *smartphones* e *tablets*, quer localmente, quer a partir da internet, controlar os dispositivos da sua residência, sem a necessidade da instalação e configuração de um computador ou servidor local. Para além disso permite ainda adicionar facilmente novos serviços à residência, pois estes são prestados pela *cloud* e não por um dispositivo local. A segurança destes serviços também é importante, sendo necessária a utilização de mecanismos seguros, como os exemplificados em [47], onde é utilizado XMPP em conjunto com serviços web, para fornecer serviços de automação doméstica num ambiente *smart grid*, de forma eficaz e protegida. Com a quantidade de dispositivos numa casa inteligente, surgem questões de segurança que têm de ser levantadas e analisadas, para que seja possível proteger os dados dos seus utilizadores, por meio de *frameworks* que podem ser utilizadas para ajudar a evitar questões de vulnerabilidade [48]. Não só os dispositivos têm de ser protegidos, mas também as ligações e serviços que as casas inteligentes providenciam, criando arquiteturas de comunicação seguras e fiáveis, que permitam por exemplo a redes de sensores, comunicarem em segurança [49].

2.3. *GATEWAY*

Os *gateways* são peças fundamentais não só no ambiente tecnológico onde se inserem as residências atualmente, mas também por ser um dispositivo que permite a agregação de tráfego e a conversão tecnológica, sendo importante na construção de um ambiente heterogéneo, convergente e conectado. Os *Home Gateways*, ou *gateways* domésticos, são essenciais para as redes domésticas, já que estes estabelecem a interface entre as redes locais e a Internet. Sendo a Internet fulcral para o entretenimento e funcionalidade de uma rede doméstica, é importante que se analise este setor, bem como as suas tendências.

Como foi retratado anteriormente, existe uma tendência em geral no esforço para reduzir e controlar o consumo energético de uma habitação e seus equipamentos. Sendo que os *gateways* domésticos têm acesso quer às redes locais, quer à Internet, podem desempenhar um papel fundamental na eficiência energética de uma residência. Em [50] e [51] é implementado um esquema de alimentação no *gateway* doméstico, que permite controlar os diversos módulos, de forma independente, reduzindo assim o consumo energético do mesmo. A eficiência não se fica pela rede local. Em [52] por exemplo, é explorada a possibilidade de utilizar tecnologias de virtualização por parte dos operadores de serviços, para transferir serviços prestados pelo *gateway*, para esses sistemas virtuais, de modo a reduzir o consumo destes.

Para além da tendência de reduzir o consumo do *gateway*, este pode também ajudar na monitorização do consumo doméstico, como em [53], onde é desenvolvido um *gateway* que monitoriza e disponibiliza informação relativa ao consumo dos equipamentos domésticos na televisão, permitindo aos residentes gerir melhor os consumos energéticos. Alguns artigos vão mais além, efetuando uma pesquisa e uma análise que discute o estado atual, no panorama atual da eficiência energética nas residências, dando ainda uma solução para melhorar a eficiência do *gateway* doméstico [54].

Outra tendência tecnológica nestes dispositivos é a capacidade de serem capazes de prestar serviços multimédia como *streaming* de conteúdo multimédia, IPTV, VoIP entre outros, já que os utilizadores domésticos cada vez mais consomem este tipo de recursos [55] [56]. Em [57] é desenvolvido um *gateway* doméstico compatível com (*Fiber to the Home*) FTTH, que permite a prestação de serviços IPTV, VoIP e Internet. Em [58] é implementado um gestor de canais multimédia que permite gerir e controlar conteúdo multimédia que transite na rede. Já em [59] é considerado um sistema que permite a gestão e consumo de conteúdos

protegidos por direitos de autor numa rede doméstica. Não só os utilizadores consomem cada vez mais conteúdo multimédia, mas também o consomem num número alargado de dispositivos [60] [61]. Em [62] é proposto um sistema baseado em *scalable video coding* (SVC) e *Raptor forward error correction* (FEC), subsequentemente desenvolvido em [63], que permite a difusão de conteúdo em múltiplos ecrãs espalhados pela habitação, podendo esses ecrãs serem televisões, monitores ou *smartphones* e *tablets*. Seguindo a tendência, também o IPTV tem recebido atenção na área de *gateways* domésticos. Em [64] é considerada uma arquitetura que permite utilizar redes IMS para os residentes interagirem com serviços da rede de IPTV, com um Home Gateway para interface entre a rede de IPTV e a rede local. Já em [65] é feita a implementação de um *gateway* residencial que permite a prestação de serviços de IPTV, não só dos operadores, mas também de terceiros. Num terceiro caso, é implementada uma API num *gateway*, que permite a integração de iDTV com serviços domésticos, de modo a oferecer interatividade com a televisão digital [66]. Para possibilitar que esta quantidade de informação seja consumida na rede, será necessário que o QoS (Quality of Service) e QoE (*Quality of Experience*) das ligações e serviços seja satisfatório. Em [67], é implementada uma *framework* DLNA num *smartphone* Android, que permite controlar e melhorar QoS das ligações entre esse *smartphone* e outros dispositivos DLNA da rede, principalmente com o *gateway* com suporte para DLNA. Em [68] é implementado num *gateway* doméstico, um medidor e uma política de QoS que permite monitorizar o tráfego da rede, e mediante a análise do tráfego, aplicar políticas de controlo nos fluxos de dados da rede.

Os *gateways* domésticos têm também importância nas casas inteligentes, já que sendo inteligentes, estas irão estar ligadas à Internet, terão capacidade de prestar serviços de automação doméstica e integrar serviços embebidos. Os *gateways* domésticos facilitam estas possibilidades, como no caso de [69], onde é implementado um *gateway* adaptativo que consegue suportar um grande número de protocolos e tecnologias, e assim criar interoperabilidade entre dispositivos domésticos. Outras soluções podem passar pela apresentada em [70], solução que permite associar dispositivos domésticos com diferentes protocolos de comunicação com serviços web, possibilitando o seu acesso a partir de uma API, ou [71], onde um *gateway* doméstico de automação, utiliza 6lowPAN e UPnP (Universal Plug and Play), em conjunto com uma plataforma *cloud*, para controlar dispositivos domésticos remotamente, permitindo ainda a resolução entre IPv4 e IPv6.

2.4. TENDÊNCIAS

Anteriormente foi feita uma análise aos desenvolvimentos e investigações de diversas áreas de interesse. Após esta análise, é também importante procurar e definir as tendências dos consumidores na atualidade. Perceber o que o mercado doméstico e os utilizadores preferem ou utilizam permitirá entender funcionalidades de interesse para o mercado atual.

2.4.1. CLOUD VS LOCAL

A questão em torno da computação em nuvem é o facto de ser uma solução que funciona com base no "aluguer" de uma aplicação ou plataforma computacional remota. Isto permite não só aceder às aplicações e plataformas remotamente a partir de qualquer lugar, mas também em certos casos, principalmente para empresas, reduz consideravelmente as despesas com equipamento informático, manutenção e pessoal especializado [72]. Já no caso de fornecer serviços de *cloud* para consumidores, estes surgem normalmente na forma de aplicações remotas, como *instant messaging*, hospedagem de ficheiros e *streaming* de multimédia, geridas por empresas com infraestruturas próprias, ou que conseguem alugar grandes infraestruturas de modo a reduzir consideravelmente o custo desse aluguer. Assim conseguem suportar os custos e oferecer aos seus utilizadores aplicações de *cloud* com preços competitivos ou até de forma gratuitamente.

A posse destes serviços por parte de entidades terceiras é uma das preocupações que leva os consumidores a evitarem estes serviços para gerirem conteúdo pessoal ou privado. Por norma, estes serviços de *cloud* obrigam os seus utilizadores a confiar todos os seus dados e atividades à entidade que controla a aplicação. Este facto leva à insatisfação por parte dos utilizadores, que em conjunto com as notícias que tomaram conta de governos e agências de informação a acederem aos dados de milhões de utilizadores em todo o mundo, dados estes que são hospedados pelas grandes empresas da internet, que causa um desconforto geral na utilização destes serviços para fins privados e pessoais. Este descontentamento leva ao aumento na utilização de soluções descentralizadas e locais como alternativa a serviços de *cloud*. Um exemplo que comprova esta tendência é a aplicação BitTorrent Sync, um serviço de sincronização semelhante a soluções como Dropbox, Cloud Me e Own Cloud, que permite a sincronização de ficheiros entre dispositivos. A grande diferença entre a primeira solução e as restantes é o facto de esta funcionar de forma descentralizada entre dispositivos, localmente ou remotamente, sem hospedar os ficheiros num servidor central controlado por uma entidade que pode por em causa a privacidade e a proteção dos dados nos seus

servidores. O serviço Dropbox, um dos serviços de sincronização mais populares da atualidade [73] [74], tem sofrido ao longo dos anos, múltiplos ataques informáticos bem-sucedidos e erros técnicos que colocaram em causa a segurança e a privacidade dos ficheiros hospedados pelos seus utilizadores [75] [76]. Graças ao receio deste tipo de situações, o BitTorrent Sync conseguiu atingir a marca de dois milhões de utilizadores em menos de um ano de existência [77]. Este tipo de casos mostra que os utilizadores estão atentos às questões de privacidade que assolam as aplicações e serviços de *cloud*, utilizando alternativas descentralizadas e privadas.

A implementação de um servidor privado com acesso remoto a todas as suas funcionalidades num ambiente doméstico, pessoal e privado, seria de facto uma solução que implementaria uma das características mais vantajosas da *cloud*, acesso centralizado e remoto, sem perder o controlo sobre os dados e os recursos hospedados, nem depender de serviços de terceiros.

2.4.2. SERVIDORES DOMÉSTICOS

Um servidor tem o intuito de oferecer um conjunto de serviços à rede onde está instalado. Com a evolução e a tendência na convergência de tecnologias para redes IP, como voz sobre a forma de VoIP (*voice over internet protocol*) e televisão a partir de IPTV (*internet protocol television*), torna possível a um sistema informático local, oferecer uma diversidade de serviços que vão desde hospedagem e partilha de ficheiros, passando por serviços de conversação e comunicações, até serviços multimédia e utilitários. Exemplos de servidores que oferecem conjuntos de serviços aos seus utilizadores são os Eee Box PCs com Windows Home Server e o Mac Mini com OS X Server. Estes sistemas para além de terem uma dimensão reduzida e serem silenciosos, têm a vantagem de serem compatíveis com sistemas operativos já preparados para suportarem funcionalidade de servidor. Embora sejam servidores, estes têm interface gráfica, permitindo uma interação e configuração mais intuitivas. Indo mais para o domínio do conteúdo multimédia, existem soluções como o ASUS TS Mini [78], que permite fazer *streaming* de vídeo e música e até gerir redes de câmaras de vigilância.

2.4.3. NETWORK ATTACHED STORAGE (NAS)

O conceito da hospedagem e sincronização de ficheiros permite aos seus utilizadores, carregar documentos e pastas para o serviço, sendo estes ficheiros disponibilizados na internet para o seu acesso remoto. Para além disso, também permite sincronizar ficheiros modificados e alguns destes serviços até possibilitam a criação de *backups*, de modo a proteger os dados de possíveis perdas. Uma das soluções locais que permitem este tipo de serviços engloba os sistemas NAS (*Network Attached Storage*). Gamas de dispositivos como a MyCloud da Western Digital [79], ShareCenter da D-Link [80] ou a DiskStation da Synology, constituem conjuntos de produtos orientados para o mercado doméstico, onde em grande parte dos casos são implementadas as funcionalidades anteriores, podem ser acedidos remotamente a partir de um navegador web, têm a capacidade de armazenamento adequada a uma utilização residencial e algumas soluções até possibilitam a utilização de servidores multimédia baseados em DLNA/UPnP. Estes NAS têm também a possibilidade de utilizar dispositivos de armazenamento externo a partir das portas USB disponíveis, para permitir o acesso a estes a partir de qualquer dispositivos da rede e para efetuar *backups* de dispositivos externos para o armazenamento interno, ou a operação inversa.

Por esse motivo os dispositivos NAS ou dispositivos que partilhem as suas funcionalidades têm interesse por parte dos consumidores e por parte dos fabricantes.

2.4.4. MENSAGENS INSTANTÂNEAS

Os serviços de mensagens instantâneas estão a ganhar popularidade, substituindo cada vez mais as mensagens curtas (SMS) *Short Message System* [81]. Estima-se que até 2019, estejam registadas 3,8 mil milhões de contas referentes a serviços de mensagens instantâneas [82]. Aplicações como o GTalk, WeChat, WhatsApp, Line e Viber permitem a comunicação entre dois ou mais utilizadores a partir de mensagens escritas gratuitamente e em alguns dos casos até chamadas de voz e vídeo, utilizando a internet como meio de comunicação.

Se as funcionalidades de mensagens instantâneas destas aplicações fossem implementadas num sistema pessoal, este poderia prestar serviços de mensagens privadas em tempo real e sem controlo de terceiros. Assim não existe o risco das mensagens serem intercetadas por entidades ou serem adquiridas por grandes empresas, mantendo a possibilidade de comunicar com familiares e amigos.

2.4.5. CHAMADAS E CONFERÊNCIAS VOIP

Com a popularidade das aplicações VoIP, como Skype, MobileVOIP, Viber, Google Hangouts e Rebtel, permitem efetuar de forma gratuita e chamadas a custos reduzidos para números fixos. Assim, disponibilizando as funcionalidades das aplicações VoIP e permitindo efetuar chamadas e conferências de voz e de vídeo, utilizando um servidor pessoal para a gestão do serviço, possibilitaria aos seus utilizadores terem o seu próprio serviço de chamadas e conferências doméstico, que poderiam utilizar para contactar com familiares e amigos de forma privada.

2.4.6. MULTIMÉDIA

O consumo de recursos multimédia é um dos grandes fatores a serem considerados, principalmente no domínio doméstico. Como foi notado anteriormente, existem grandes esforços para criar soluções ao nível das casas inteligentes, *gateways* e redes de dados, para possibilitar o consumo deste tipo de recursos que têm a propriedade de serem complexos e necessitarem de infraestruturas e equipamentos que consigam suportar a exigência no processamento dos conteúdos, bem como a largura de banda necessária. Para acompanhar esta tendência, seria essencial a implementação no servidor da capacidade de partilha e *streaming* de ficheiros de vídeo, áudio e fotografias armazenado no servidor por parte dos utilizadores. Permitir aos utilizadores armazenarem conteúdo multimédia no servidor e conseguirem posteriormente a sua execução a partir de qualquer ligar, acrescentaria valor ao projeto e conseguiria prestar um serviço que é popular entre os utilizadores.

2.4.7. COMUNIDADES VIRTUAIS

Para além de todos os conceitos anteriormente analisados, surge um conceito que conjuga grande parte do que foi considerado, a um nível que permite uma interatividade entre utilizadores, dispositivos e até domínios fisicamente distantes. As comunidades virtuais centram-se na ideia de permitir a conjugação de dispositivos em grupos organizados. Estes grupos podem ser locais, ou fazer parte de diferentes domínios físicos. Este conceito poderiam ser implementado de várias formas. Numa primeira abordagem, dado que hoje em dia cada utilizador pode possuir vários dispositivos pessoais, estes dispositivos poderiam ser conjugados num grupo que representaria esse utilizador. Assim na rede não iriam surgir os dispositivos ou serviços associados a eles, mas os serviços associados ao utilizador. Cada casa, representada por um domínio, pode conter os seus utilizadores, permitindo a estes

comunicarem entre si de forma transparente, independentemente dos dispositivos de cada um. Indo ainda mais além, vários domínios podem conectar-se remotamente entre si. Locais fisicamente separados podem conter um servidor que poderá efetuar ligações a outros servidores remotos, ou receber ligações destes. Isto permite que os utilizadores e serviços de cada um destes domínios sejam visível como se pertencessem à mesma rede local.

3. ESPECIFICAÇÃO DO SISTEMA

3.1. ANÁLISE DE REQUISITOS

Para que o projeto seja bem-sucedido é necessário estabelecer os requisitos fundamentais, de modo a ter uma base de princípios que ajudem na definição dos conceitos essenciais a respeitar durante o desenvolvimento. A partir do estudo feito anteriormente, é possível identificar não só as tendências que influenciam este trabalho, mas também justificar algumas das propriedades que este deve conter. As propriedades consideradas permitem selecionar e integrar funcionalidades que estejam de acordo com as propensões da atualidade, de forma a dar mais valor científico e económico ao projeto.

Um dos fatores importantes a definir são as aplicações ou funcionalidades que são de interesse implementar num sistema local, de modo a que este possa ser útil aos seus utilizadores e que vá de encontro às tendências da atualidade.

Da secção *Network Attached Storage* (2.4.3) é evidente a sua implementação em dispositivos locais, que suportam funcionalidade de hospedagem e acesso remoto a ficheiros, *backups* e sincronismo de pastas remotas. Assim estas funcionalidades devem ser integradas no sistema

de forma a aproveitar a sua utilidade e popularidade, oferecendo serviços como DropBox e Google Drive, mas controlados pelo utilizador.

Já da secção de mensagens instantâneas (2.4.4), é notório que aplicações deste tipo têm sofrido uma adoção cada vez maior por parte da população. A implementação de funcionalidades de presença e comunicação por mensagens escritas pode trazer a um sistema local, um método de conversação apelativa entre familiares e amigos. Considerando ainda outras formas de interação entre utilizadores, implementar partilha de conteúdo e transferência de ficheiros, permite acrescentar uma segunda camada de interação. Aproveitando a popularidade das aplicações VoIP, evidenciada na secção de chamadas e conferências VoIP (2.6.5), a incorporação de mecanismos de comunicação multimédia, utilizando voz e vídeo permitirá aos seus utilizadores, estabelecerem chamadas e conferências multimédia, oferecendo serviços como Skype, Viber e Google Hangouts no sistema local.

Na secção *gateway* (2.3) e multimédia (2.4.6), o consumo de conteúdo multimédia é bastante prezado atualmente. Por um lado os consumidores desejam visualizar vídeos, filmes e ouvir música nos seus dispositivos. Por outro lado, a indústria cada vez mais considera a inclusão de soluções nos dispositivos domésticos, que incluam este tipo de funcionalidades. Assim é imperativo a adição da capacidade de permitir o consumo de conteúdo multimédia no sistema local, pela forma de VoD (*Video-on Demand*) para vídeos e filmes, bem como AoD (*Audio-on Demand*) para música. Desta forma, os utilizadores podem ter o seu próprio Youtube, Netflix ou Spotify.

Resumindo, os requisitos do sistema em termos de serviços a disponibilizar aos seus utilizadores são os seguintes:

- Hospedagem remota de ficheiros
- Sincronização de ficheiros entre dispositivos
- *Backups* remotos
- Partilha de pastas e ficheiros
- Presença e deteção de utilizadores
- Mensagens instantâneas entre dois ou mais utilizadores
- Transferências de ficheiros entre utilizadores
- Chamadas/Conferências de Voz
- *VoD*

- *AoD*

3.1.1. SISTEMA

Com todas as funcionalidades que o servidor pretende oferecer, exige que os métodos de interação tenham de ser desenvolvidos para permitir que utilizadores casuais interajam de forma fácil e natural com o servidor.

Cliente:

A criação de um cliente que complementasse o servidor, de modo a que um utilizador pudesse aceder a todos os recursos, de forma centralizada e em cada um dos seus dispositivos, seria uma mais valia. Como foi visto na secção casas inteligentes (2.2), o número de dispositivos nas residências tem crescido, pelo que é importante que todos estes dispositivos possam interagir com o servidor para o acesso e a partilha de recursos. É necessário selecionar as ferramentas de desenvolvimento necessárias para garantir o desenvolvimento do cliente para as plataformas pretendidas, mas que também facilite o desenvolvimento posterior para plataformas móveis. Para além do lado da programação, também é importante considerar que a interface deve ser normalizada entre os vários dispositivos. É essencial tornar a interface do cliente semelhante entre as diversas plataformas, para dar ao utilizador uma experiência unificada. Também se torna fundamental permitir aos utilizadores, o consumo e partilha dos recursos e serviços, numa só aplicação cliente. Para isso é importante que esta consiga conciliar todos os serviços, sem prejudicar a experiência de utilizador. Por fim, o utilizador deverá ter acesso de forma intuitiva às informações referentes a todos os serviços a partir do cliente, com um sistema de notificações que centralize ou concentre todas as atividades do sistema.

Centralizado:

As aplicações e funcionalidades prestadas pelo servidor terão de ser centralizadas e orientadas ao utilizador. Dando um exemplo, quando um utilizador envia uma mensagem ou pedido de transferência outro, este envio é feito ao utilizador e não apenas a um dispositivo. Assim todos os dispositivos do recetor devem detetar estas ações.

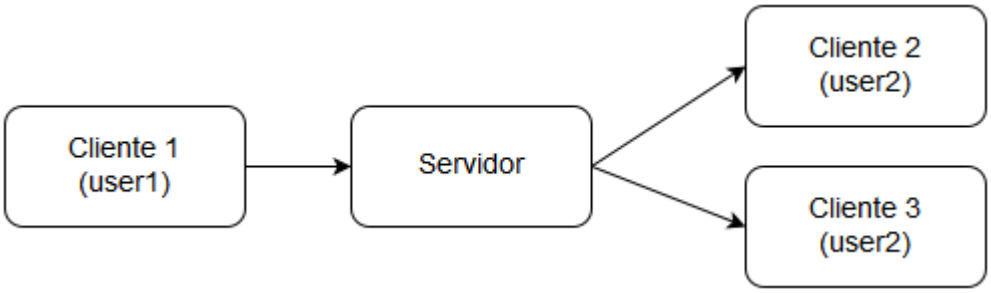


Figura 1 Ilustração de um pedido feito ao utilizador

Mais ainda, mesmo que um dispositivo se autentique posteriormente aos eventos, o sistema deve ser capaz de notificar o utilizador a partir do novo dispositivo, em relação aos acontecimentos ocorridos anteriormente. O recetor pode assim tomar conhecimento, aceitar, responder ou rejeitar às ocorrências em qualquer dispositivo no qual já esteja autenticado ou se autentique.

Administrador:

Os mecanismos de administração de um servidor devem permitir a configuração dos serviços do sistema, sem exigir ou apenas necessitando de poucos conhecimentos técnicos por parte do utilizador. Assim é exigido que o servidor possua um sistema que permita configurar todas as funcionalidades presentes, e que ao mesmo tempo essa configuração seja intuitiva e prática.

Multiutilizador:

O servidor terá a capacidade de aceitar múltiplos utilizadores em simultâneo e até vários dispositivos associados ao mesmo utilizador. Torna-se necessário garantir que o servidor consiga suportar múltiplas sessões simultâneas de utilizadores diferentes e de vários dispositivos do mesmo utilizador. Para além disso, os vários utilizadores ou até um utilizador a partir de múltiplos dispositivos pode aceder ao mesmo serviço em simultâneo. Se dois utilizadores estiverem a utilizar o serviço de mensagens instantâneas em simultâneo, mas não entre si, é necessário que o servidor consiga criar múltiplas salas e grupos de conversação distintos.

3.1.2. FUNCIONALIDADES CLOUD E COMUNIDADES VIRTUAIS

Aproveitando a popularidade dos serviços *cloud* em conjunto com a privacidade dos sistemas residenciais, o sistema terá a capacidade de ser acedido quer localmente a partir da

rede LAN (*Local Area Network*), quer remotamente, a partir da internet. Para além disso, o servidor será capaz de suportar opcionalmente, um serviço externo de DNS dinâmico, para que o sistema possa ser acedido do exterior com recurso a um domínio de internet fixo, em vez de necessitar do endereço IP público da residência, que por norma tende a variar.

Com todas as funcionalidades anteriormente definidas, a inclusão de um sistema de comunidades virtuais, que permita a conexão e partilha de recursos entre servidores, é uma mais-valia para esta solução. Assim os utilizadores devem poder interagir com outros correspondentes, registados em servidores distintos, utilizando qualquer método de comunicação disponível, como mensagens, chamadas ou transferências. Será ainda possível efetuar a partilha de pastas e ficheiros alojados num servidor, a utilizadores de outro. Desta forma todos os utilizadores presentes em vários servidores distintos fazem parte da mesma comunidade.

3.2. SELEÇÃO DE COMPONENTES

Depois de definidos os requisitos do projeto, foi altura de investigar qual as melhores tecnologias, *standards* e programas que melhor se aplicassem aos requisitos propostos.

3.2.1. FUNCIONALIDADES E SERVIÇOS

Considerando as funcionalidades pretendidas, procedeu-se à procura de soluções que ajudassem a prestar estes serviços. Depois de alguma pesquisa e reflexão, surgiram vários padrões que permitiram dividir as funcionalidades base em três categorias ou grupos distintos. Estas divisões são possíveis porque os elementos que constituem os grupos, executam funções que correspondem a uma área específica, sendo posteriormente uma vantagem na escolha da solução que vai prestar os serviços do grupo e ao mesmo tempo, restringindo as áreas de investigação nesta fase.

Tabela 1 Grupos Funcionais

Operação de Ficheiros	Comunicação entre utilizadores	Serviços Multimédia
Hospedagem remota Sincronização de ficheiros entre dispositivos Backups remotos Partilha de Ficheiros	Presença Notificações Mensagens Instantâneas Chamadas e Conferências Transferências	*Streaming de Música* VoD AoD

O grupo dos ficheiros engloba a hospedagem, sincronização, *backups* remotos e partilha de conteúdo, pelo facto de todas estas funções se focarem na gestão de ficheiros entre o utilizador e o servidor. Já o grupo de comunicação engloba os serviços a que os utilizadores do servidor podem recorrer, de forma a comunicarem entre si. Estes serviços são: presença, notificações, mensagens instantâneas, chamadas e conferências de voz e vídeo, bem como transferências de ficheiros. Por fim as funções de *Audio-on Demand* e Video-on Demand constituem o grupo de multimédia.

De seguida para cada grupo, foi realizada uma análise das possíveis soluções que conseguissem executar os serviços respetivos: Todas as soluções consideradas são aplicações *open-source* ou protocolos abertos, são atualizados e atualmente desenvolvidos, e suportam de algum modo, funcionalidades do grupo no qual estão inseridas.

- Ficheiros:
    - FTP

O FTP (*File Transfer Protocol*) é um protocolo de rede que permite a hospedagem e transferência de ficheiros entre um cliente e um servidor. Tem a desvantagem de ser um protocolo antigo e embora ainda seja utilizado, não é encriptado e obriga a medidas adicionais de segurança como a utilização de SSL/TLS ou SSH [83].

    - rSync

Ferramenta que permite a transferência e sincronismo de ficheiros entre sistemas remotos. Utiliza codificação delta para reduzir o tráfego de rede, analisando e transmitindo apenas as diferenças entre os ficheiros. Pode utilizar compressão para reduzir o tamanho dos ficheiros e utilizar SSH para encriptação [84].

- SCP

O SCP (Secure Copy) é um protocolo de rede que permite a transferência de ficheiros entre dois dispositivos remotos. É baseado em SSH, logo já possui encriptação por omissão [85].

- SMB/CIFS

SMB (Server Message Block)/CIFS (Common Internet File System) é um protocolo de rede da Microsoft que permite a partilha de ficheiros, impressoras e portos, entre sistemas Windows [86].

- Samba

Reimplementação dos protocolos SMB e CIFS, oferecendo partilha de ficheiros e impressoras, entre clientes locais, suportando plataformas Windows, e a maioria das plataformas baseadas em Unix, entre as quais Linux, Mac OS, Solaris, AIX e FreeBDS [87].

- NFS

Desenvolvido pela Sun Microsystems, o NFS (*Network File System*) permite a um dispositivo local, o acesso a ficheiros localizados na rede local, como se estivessem presentes no dispositivo.

- Comunicação:
  - *Internet Relay Chat*

O *Internet Relay Chat*, mais conhecido por IRC, é um protocolo que permite facilitar a troca de mensagens de texto, utilizando um modelo cliente/servidor. Por norma é utilizado para comunicações em grupo, utilizando servidores de *chat* para hospedar salas de conversação e retransmitir as mensagens entre os clientes. Embora seja popular por esta rasão, também suporta comunicações entre dois utilizadores e transferências de ficheiros [88].

- Mumble

Software VoIP *open-source* muito utilizado por jogadores de vídeo jogos que pretendam comunicar entre si por meio de voz, suportando *overlays* gráficos sobre os jogos e até som

direcional. Os clientes podem conectar-se a um servidor criar e aceder a salas comuns intituladas canais, e comunicar por meio de voz com todos os utilizadores conectados ao canal. Possui boa qualidade de som, baixa latência e encriptação [89].

- SIP

O SIP (*Session Initiation Protocol*) é um protocolo de comunicação que é utilizado para sinalização e controlo de sessões multimédia numa rede IP. Geralmente empregado em telefones e *software* VoIP, suporta chamadas de voz e vídeo, bem como mensagens instantâneas [90].

- XMPP/Jabber

O XMPP (*Extensible Messaging and Presence Protocol*) também conhecido por Jabber é um protocolo de comunicações orientado a serviços de mensagens instantâneas e presença. Baseado em XML (*Extensible Markup Language*) foi desenhado para ser extensível, suportando um grande número de aplicações tais como salas de conversação, sistemas de publicação e subscrição, transferências de ficheiros, sinalização de chamada VoIP, vídeo jogos, IoT (*Internet of Things*), *smart grid* e serviços de redes sociais, entre outras. Os servidores XMPP têm a capacidade de funcionarem em conjunto. Estes podem comunicar entre si por meio de comunicações s2s (*server-to-server*), possibilitando a utilizadores de um, comunicarem com utilizadores de outros servidores, sem a necessidade de se registarem nestes [91].

- Multimédia:
  - Logitech Media Server

Servidor de *streaming* de música desenvolvido e suportado pela Logitech, com o intuito de suportar a sua gama de recetores de áudio SqueeseBox. Permite aos seus utilizadores acederem à sua biblioteca de músicas, desde que tenham acesso à rede do servidor ou à Internet [92].

- Kodi (XBMC)

Kodi, anteriormente conhecido por XMBC (Xbox Media Center), é um *software* multiplataforma que permite aceder e executar conteúdo multimédia do dispositivo local, da rede LAN e possui um sistema de plug-ins que permitem aceder a conteúdo na internet,

como YouTube e Spotify. Também pode funcionar como servidor multimédia UPnP AV [93].

- PS3 Media Server

Inicialmente lançado para suportar *streaming* de vídeo e áudio para a consola de vídeo jogos PlayStation 3, este servidor de multimédia compatível com dispositivos DLNA, permite transmitir e transcodificar conteúdo multimédia [94].

- VideoLAN

VideoLAN ou VLC é um poderoso leitor e *framework* de multimédia que suporta um alargado conjunto de formatos e *codecs* de vídeo e áudio [95]. Suporta variadas fontes de vídeo e áudio, como ficheiros, fluxos de rede, placas de captura e leitores de DVD, bem como funcionalidades de *transcoding* e servidor multimédia [96].

- FFmpeg

É uma *framework* multimédia que possui um conjunto de ferramentas que permitem codificar, descodificar, transcodificar, executar, filtrar, misturar, separar, analisar e transmitir conteúdo multimédia, suportando praticamente todos os formatos existentes. É utilizado num conjunto alargado de aplicações e outras *frameworks* multimédias, sendo alguns dos exemplos, VLC, Plex, YouTube, Google Chrome, DirectShow e GStreamer [97].

Depois de pesquisados os *standards* e ferramentas de interesse, estes foram comparados, no que toca a fornecer as funcionalidades pretendidas.

Tabela 2 Tabela comparativa de capacidade funcional entre as soluções consideradas

	Ficheiros				Comunicações									Multimédia					
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
FTP	●																		1
rSync		●	●																2
SSH/SCP	●																		1
SMB/CIFS	●			●					●										3
Samba	●			●					●										3
NFS	●			●					●										3
IRC					●			●	●										3
Muble					●					●	●								3
SIP					●		●	●	●	●	●	●	●						8
XMPP					●	●	●	●	●	●	●	●	●					a	10
Logitech Media S.														●		●			2
Kodi														b	b	c		d	4
PS3 Media Server														●	●	●			3
VideoLAN														●	●	●			3
FFmpeg														●	●	●	●	e	5

Soluções Escolhidas Funcionalidades asseguradas

Tabela 3 Tabela com os significados dos parâmetros da tabela comparativa

1	Hospedagem remota	11	Conferências de voz
2	Sincronização de ficheiros entre dispositivos	12	Chamadas de vídeo
3	Backups remotos	13	Conferências de vídeo
4	Partilha de Ficheiros	14	Video-on Demand
5	Presença	15	Audio-on Demand
6	Notificações	16	*Transcoding*
7	Mensagens	17	Linha de comandos
8	Salas de chat	18	Expansível
9	Transferências	19	Pontuação Total
10	Chamadas de voz		

Comparando as opções consideradas, ficou decidido que no que toca a serviços de comunicação entre utilizadores, o XMPP cobre todas as funções essenciais. Pelo facto do XMPP conter o sistema de extensões (XEPs), permite não só a expansão de funcionalidades

e mecanismos de comunicação, mas também que o sistema possa futuramente adaptar-se a novos conceitos de comunicação, como a internet das coisas [91].

Para os serviços de operação de ficheiros, ficou patente que nenhuma tecnologia cobre todos os requisitos pretendidos. Assim seria necessário implementar duas destas tecnologias de modo a cobrir todos os serviços. Após alguma ponderação, e com o facto de o XMPP também possibilitar o desenvolvimento de extensões personalizadas, seria possível implementar os serviços de operação de ficheiros com este protocolo, substituindo a necessidade de utilizar soluções dedicada para estes serviços e assim concentrando a maior parte dos serviços no XMPP.

Para a prestação dos serviços multimédia, a escolha recaiu sobre o FFmpeg. Não só suporta *streaming* de vídeo e áudio e *transcoding*, como também suporta um conjunto invejável de codecs, formatos e protocolos de multimédia. Permite utilizar filtros e misturadores, bem como variadas fontes de multimédia como câmaras, leitores de DVDs e *streams* da rede. Mais do que isso, sendo o FFmpeg uma ferramenta que permite a sua interação por linha de comandos ou API (*application programming interface*) permitiria ao XMPP interagir com a *framework* FFmpeg no servidor, e prestar os serviços multimédia aos utilizadores, manipulando este conteúdo conforme o necessário. Assim conclui-se que o protocolo XMPP será o mecanismo principal para efetuar os serviços de operação de ficheiros, comunicação entre utilizadores e interação com o FFmpeg, concedendo os serviços multimédia à *framework* FFmpeg.

De seguida foi necessário escolher o servidor XMPP que não só suportasse os requisitos apresentados e as funcionalidades base, mas que também fosse robusto, escalável e que tivesse suporte para um alargado número de XEPs. Foram considerados para esta seleção soluções *open-source* e que sejam projetos atualmente suportados e desenvolvidos.

- Servidores XMPP:
    - ejabberd:

O ejabberd é um servidor de mensagens instantâneas baseado no protocolo XMPP. Sólido, modular, escalável e extensível, oferece ainda suporte a uma grande quantidade de extensões XMPP, desde permitir um serviço de salas de conversação até suporte para *proxies* de SOCKS5 para transferências de ficheiros e redireccionamento de *sockets* [98]. Pode ser configurado a partir de uma interface web e ainda suporta IPv6 bem como autenticação por LDAP ou PAM.

- MongooseIM:

Baseado no ejabberd, este servidor XMPP tem como foco a performance e escalabilidade, incluindo suporte para bases de dados configuráveis e WebSockets para aplicações web modernas [99].

- Jabberd2:

Inspirado pelo seu antecessor jabberd, o primeiro servidor XMPP open-source, jabberd2 foi feito do início para ser escalável, seguro e para suportar novas extensões XMPP [100].

- Prosody:

Servidor Jabber/XMPP fácil de utilizar, baixo em consumo de recursos e oferece um sistema de desenvolvimento flexível, que permite desenvolver novas funcionalidades e até novos protocolos [101].

- Metronome:

Baseado no servidor Prosody e inicialmente uma variante deste, foi desenhado para suportar principalmente funcionalidades ligadas a redes sociais, como salas de *chat* e *microblogging*, prevenção de spam e melhorias na interação entre servidores [102].

Depois de apresentadas as opções consideradas, é altura de comprara e escolher a que melhor se adapta a um ambiente doméstico e privado. Entre estes servidores o que mais se destaca é o ejabberd. Uma das facetas mais importantes é ser robusto e escalável, e por isso ter sofrido a adoção por parte de algumas soluções e serviços comerciais como Jabber.org [103], WhatsApp e Facebook [104]. Aliado ao facto de suportar federações de servidores e comunicações s2s (server-to-server), permite implementar o sistema de comunidades pretendido no projeto. Para além disso, também é flexível e modular, possibilitando não só a ativação/desativação de serviços XMPP já existentes, como também oferece a capacidade de desenvolver módulos personalizados, aumentando a ainda mais as possibilidades do protocolo XMPP. Por fim, sendo que este suporta um alargado conjunto de extensões XEPs, oferece uma base sólida de funcionalidades para o sistema pretendido. Ficou então decidido que o ejabberd seria o servidor XMPP implementado no servidor do projeto.

3.2.2. AMBIENTE DE DESENVOLVIMENTO

De seguida é feita a escolha da plataforma e da *framework* de desenvolvimento, e é efetuada a configuração e o teste de todos os módulos necessários para o desenvolvimento do projeto. Dado que o cliente será multiplataforma e o projeto assenta em redes, multimédia e interfaces gráficas, foram analisadas as soluções que auxiliassem ao desenvolvimento do projeto. Assim surgem as seguintes alternativas.

1º: Aplicação Web -> A vantagem de desenvolver uma aplicação web, prende-se com a capacidade desta correr em qualquer plataforma que possua um navegador web, sendo a solução mais abrangente em termos de número de plataformas compatíveis. Existem duas desvantagens nesta abordagem. A primeira engloba o acesso bastante restrito das aplicações web aos recursos e periféricos do sistema. Assim sendo, grande parte dos navegadores não suportam o acesso ao microfone e câmara do sistema, impossibilitando funcionalidades que utilizem esses recurso, como chamadas e conferências.

2º: Aplicação -> Em [105] é feito um trabalho extenso a identificar um grande conjunto de soluções para desenvolvimento de aplicações multiplataforma, comparando estas em termos de funcionalidades, tipo de aplicações, plataformas suportadas, facilidade de utilização e acesso aos recursos do sistema. Das aplicações que foram analisadas, a que mais chamou a atenção foi a *framework* Eqela. Esta framework utiliza uma linguagem própria e permite posteriormente, converter e compilar a aplicação para as linguagens correspondentes a diferentes plataformas (c, java, javascript, etc...). Isto permite criar aplicações nativas para todas as plataformas pretendidas, possibilitando a estas aceder aos recursos do sistema. Mas não só é possível converter para aplicações nativas, mas também permite converter para HTML/javascript, possibilitando assim a sua utilização, por exemplo em smart TVs que suportam apenas estas linguagens. O problema do Eqela é o facto de não ter bibliotecas necessárias para os protocolos implementados no servidor e das funcionalidades de rede, interface e multimédia, essenciais ao projeto, obrigando assim ao desenvolvimento das mesma para a sua implementação no cliente. Daí ter surgido a terceira alternativa.

3º: Qt-> O Qt é uma *framework* de desenvolvimento de aplicações C++ multiplataforma que suporta todas as plataformas pretendidas, bem como dispositivos móveis [106]. Não só resolveria o problema do acesso aos recursos do sistema e da performance, pois as aplicações são nativas e desenvolvidas em C++, bem como colmata em grande parte o problema que o Eqela tem. Possui bibliotecas para interfaces gráficas, multimédia e rede, permitindo assim

implementar o desenvolvimento do projeto. Sendo o Qt *open source*, possibilita a utilização de bibliotecas terceiras, desenvolvidas pela comunidade e que aumentam ainda mais as funcionalidades do Qt. Com o facto de possuir todas as bibliotecas necessárias para o desenvolver a solução pretendida, permite estabelecer a base de programação para este projeto, e ao mesmo tempo, acelerar e simplificar todo o processo.

3.2.3. RESUMO

Considerando o estudo efetuado anteriormente e selecionadas as melhores soluções para cada função, resume-se de seguida a lista com as aplicações e tecnologias escolhidas:

- ejabberd: Servidor XMPP que suportará o mecanismo de interação entre os clientes e o servidor e prestará os serviços de operação de ficheiros e comunicação entre os utilizadores do sistema.

- FFmpeg: *Framework* multimédia que será responsável por prestar os serviços multimédia à rede.

- Qt: O ambiente de desenvolvimento *Integrated Development Environment* (IDE) utilizado para fornecer a este projeto as bibliotecas de programação necessárias para a sua implementação, permitindo a integração de uma aplicação cliente *standalone* com interface gráfica e acesso aos recursos do sistema, e uma aplicação servidora que fornecesse os serviços à rede. Para a implementação da interface foi utilizado Qt Quick, uma das tecnologias que o Qt disponibiliza para o desenvolvimento de GUIs, que permite desenhar interfaces dinâmicas e com suporte para conteúdo multimédia.

3.3. FERRAMENTAS

Para o desenvolvimento deste projeto foi utilizado um conjunto complementar de ferramentas para além das já mencionadas, que suportaram os componentes principais e permitiram a implementação das funcionalidades desejadas.

- QXmpp:

O Qt por si só não possui uma biblioteca que suporte o protocolo XMPP. Daí ser necessária a utilização de uma biblioteca terceira que adicione ao Qt a funcionalidade de desenvolver aplicações baseadas neste *standard*. O QXmpp é uma biblioteca de XMPP para Qt, que é instalada em separado e implementa funções e mecanismo de gestão que facilitam a implementação de funcionalidades baseadas em XMPP. Mais informações, descrição e documentação podem ser encontradas na página oficial do QXmpp [107]. Esta biblioteca é assim utilizada para implementar no projeto, o protocolo XMPP bem como as extensões (XEPs) necessárias.

- Jitsi e Pidgin:

De forma a testar o cliente desenvolvido no projeto durante o seu desenvolvimento, é considerada a utilização de clientes XMPP já existentes para dar apoio ao projeto. O Jitsi e o Pidgin são clientes XMPP *open-source* e multiplataforma, permitindo a sua instalação e utilização em diversas plataformas durante o desenvolvimento do cliente.

3.4. CONSIDERAÇÕES

Após mencionados os componentes, *standards* e tecnologias que são utilizadas no projeto, é importante esclarecer e explicar alguns conceitos relacionados com estes elementos, para que durante a leitura deste documento, o leitor consiga compreender algumas referentes ao Qt, XMPP e QXmpp.

3.4.1. QT

O Qt possui um sistema de sinais e *slots* [108]. Este mecanismo permite que quando um determinado evento ocorre num módulo de uma aplicação, este possa emitir um sinal. Os sinais podem ser conectados a uma ou várias *slots*. Uma *slot* corresponde a uma função que será executada quando o sinal ao qual se encontra conectado, for emitido. Assim quando um

determinado evento ocorre, o sinal correspondente é emitido e todas as *slots* que estiverem conectadas a esse sinal serão executadas.

3.4.2. XMPP

Foi descrito anteriormente a flexibilidade e o conjunto de aplicações possíveis do protocolo XMPP. Toda essa flexibilidade é oferecida por extensões XMPP, chamadas XEPs (XMPP Extension Protocols), que definem funcionalidades complementares, de modo a estender o protocolo XMPP. Não só é possível utilizar extensões *standard*, mas também é possível desenvolver extensões personalizadas, possibilitando a adaptação do XMPP para executar serviços que não suporta oficialmente. Durante o desenvolvimento do projeto serão mencionadas as extensões utilizadas em cada caso, e nas situações onde são desenvolvidas extensões personalizadas, o seu funcionamento será explicado.

O XMPP utiliza mensagens XML (*e**X**tensible **M**arkup **L**anguage*), na forma de *stanzas* como método de comunicação e interação entre entidades XMPP. Existem três tipos de *stanzas*.

- Mensagem: Um *stanza* que é enviado e no qual o emissor não espera resposta. É utilizado em casos onde não é necessário uma confirmação de que a mensagem foi recebida ou o recetor não necessita de responder com detalhes ou conteúdo.

- Presença: Este é utilizado para identificar o estado de presença de um utilizador. Quando um utilizador se autentica ou muda o seu estado, são utilizados *stanzas* de presença para notificar os restantes elementos XMPP, dessas atualizações.

- Iq: Este *stanza* é utilizado em casos onde é esperada uma resposta. Casos como o desejo de confirmar que o *stanza* foi recebido ou quando um cliente requisita a lista de salas de conversação existentes, é sempre esperada uma resposta do lado do recetor. Estes três *stanzas* são a base do protocolo XMPP e de todas as extensões XEPs existentes.

Outro facto importante sobre o XMPP prende-se com os três elementos de rede utilizados. O cliente, que representa um utilizador, o recurso que representa um dispositivo de um utilizador e o servidor que fornece o serviço XMPP à rede. Cada utilizador XMPP tem um

identificador chamado jid, que é representado na forma username@192.168.1.100. Este endereço contém o nome do utilizador, e o IP ou domínio do servidor onde está registado. Quando um utilizador se autentica, o jid da sessão fica com o formato username@192.168.1.100/device. Designado *full* jid, este endereço inclui no final o nome do recurso que é atribuído ao dispositivo atual com que o utilizador está a autenticar. Isto permite a um servidor XMPP detetar e gerir os seus utilizadores e os seus dispositivos na rede. Se o utilizador user1 estiver autenticado com um computador pessoal, o full jid que representa aquele dispositivo poderá ser user1@192.168.1.100/pc. Já no caso de se autenticar com o portátil o full jid deste, pode ser user1@192.168.1.100/laptop. Algumas funções XMPP, como o envio de mensagens instantâneas ou o envio de um convite para uma sala de conversação, pode ser feito para um recurso específico ou para o utilizador. Neste último caso, todos os dispositivos do recetor irão receber a mensagem ou o convite. Já em certos casos, como quando se pretender enviar um ficheiro ou efetuar uma chamada, a criação do *socket* para efetuar a transferência em si ou a criação dos canais multimédia para estabelecer a chamada, tem de ser feito entre dois dispositivos específicos, obrigando assim a que o destinatário destes serviços tenha de ser um full jid.

3.4.3. QXMPP

O facto de o qxmpp ser uma biblioteca do Qt, inclui todas as propriedades associadas a este, nomeadamente a capacidade de ser implementado em múltiplas plataformas e utilizar o mecanismo de sinais e *slots*, já descrito anteriormente. Outra propriedade do Qt importante prende-se com os gestores. O qxmpp embora tenha funções de baixo nível é uma biblioteca de alto nível. O que os gestores permitem fazer é criar uma camada de abstração que permita utilizar serviços e extensões XMPP, sem ter de criar as mensagens por conta própria. Se o cliente pretender criar uma sala de conversação e juntar-se a ela, este teria de criar e enviar o *stanza* XMPP necessário para cada uma das operações. Com os gestores este processo não é necessário. Para criar e juntar-se a uma sala, basta pedir ao gestor QXmppMucManager para criar a sala e de seguida pedir ao mesmo para juntar o cliente à sala criada. Os *stanzas* e todos os restantes procedimentos para as ações pretendidas é efetuado pelo gestor. Para além do QXmppMucManager, o qxmpp contém ainda QXmppTransferManager para gerir transferências de ficheiros e QXmppCallManager para gerir transferências, entre outros. Durante o desenvolvimento quando um gestor for utilizado, será referido o motivo da utilização e a sua função.

4. ARQUITETURA

Neste capítulo é apresentada a arquitetura do sistema desenvolvido. Será apresentada a estrutura de rede que representará os requisitos e propriedades atribuídas ao projeto, bem como a descrição dos elementos da rede e definição dos módulos que estes constituem. Por fim serão dados alguns cenários práticos, exemplificando o funcionamento das funcionalidades e da arquitetura proposta.

Nesta secção será apresentada e explicada a arquitetura de rede na qual o projeto é assente, bem como a estrutura e componentes dos elementos essenciais. A figura 2 representa a arquitetura de rede, na qual inclui todos os conceitos, requisitos e componentes do sistema anteriormente apresentados.

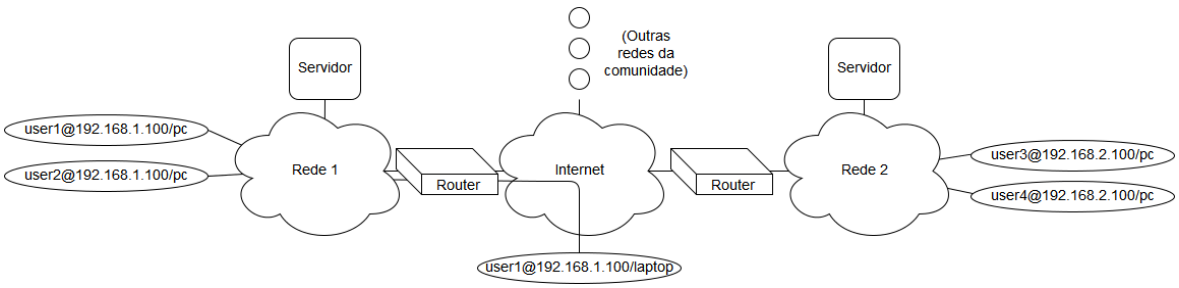


Figura 2 Nomenclatura de Rede.

Estão presentes quatro arquiteturas locais que representam domínios domésticos, exemplificando residências. Estes domínios estão conectados à Internet por via de *gateways*. A Internet será utilizada não só para conectar clientes remotos, mas também para comunicações entre servidores, permitindo comunidades virtuais entre domínios domésticos. Segue-se a descrição de cada um dos elementos desta arquitetura.

4.1. DOMÍNIO DOMÉSTICO

O domínio doméstico apresentado na nomenclatura de rede representa a estrutura habitual de uma casa, em conjunto com a aplicação do sistema proposto. Um domínio contém sempre um servidor, que representa o servidor desenvolvido neste projeto e um *router* doméstico comum em residências que providencia o acesso à Internet. Podem existir ainda clientes conectados ao servidor, a partir da rede local ou remotamente a partir da Internet.

Cada cliente representa um dispositivo, com o qual um utilizador se pode autenticar no servidor. O sistema permite que um único utilizador se possa autenticar com vários dispositivos. Isto é possível graças ao facto do protocolo XMPP incorporar os conceitos de utilizadores e recursos. Assim é possível identificar todos os dispositivos clientes como recursos XMPP e associa-los aos utilizadores que se autentiquem a partir de cada um.

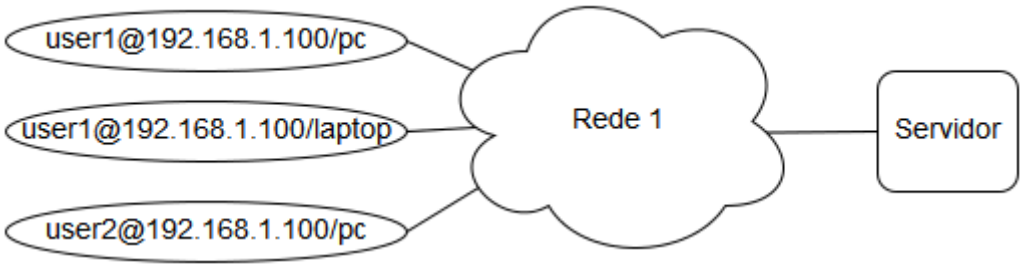


Figura 3 Ilustração do conceito de utilizador e recurso XMPP.

Cada dispositivo que se conecte fará a autenticação por meio da aplicação cliente desenvolvida neste projeto.

O servidor é o responsável por prestar os serviços pretendidos à residência e aos seus utilizadores. Estes podem estabelecer conexões com clientes ou com outros servidores

remotos, possibilitando a interação de utilizadores entre domínios. As ligações quer entre os clientes e o servidor quer entre servidores, são estabelecidas a partir de túneis encriptados, de forma a proteger as comunicações e a reencaminhar todas as conexões por um único canal de dados. Numa situação real estará conectado diretamente ao *router* doméstico a partir de uma interface de rede Ethernet.

4.2. CLIENTE E SERVIDOR

Neste projeto foram desenvolvidos dois elementos que representam as entidades da rede, que desempenhando os papeis essenciais para que o sistema funcione, e que ao mesmo tempo respeite a arquitetura proposta anteriormente. Estes são cliente e servidor. De seguida serão apresentadas as arquiteturas estruturais destes elementos, mostrando a estrutura de camadas de cada um, e dentro das quais, os componentes e blocos que executam as funções devidas, bem como a sua interação.

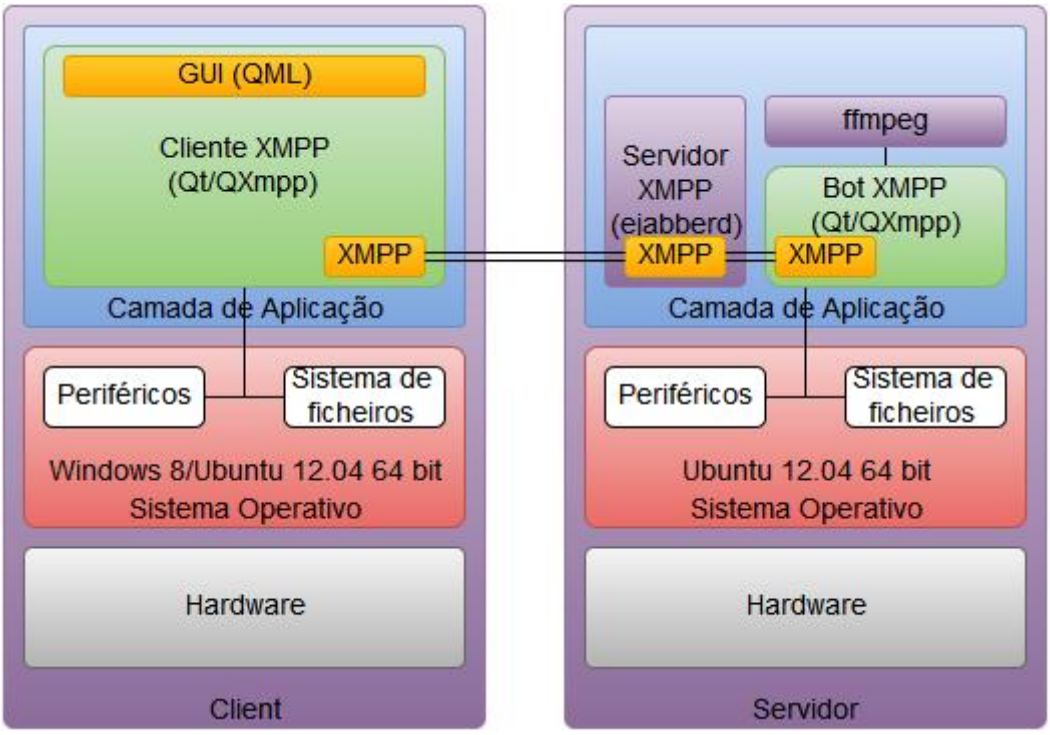


Figura 4 Arquitetura do cliente e do servidor.

- Cliente:

O cliente é o elemento que representa os dispositivos da rede com os quais os utilizadores do sistema vão recorrer para aceder aos serviços e recursos disponíveis. Este é caracterizado por três camadas, a camada física, o sistema operativo e o cliente desenvolvido.

Começando pela camada física, esta representa o dispositivo do utilizador, bem como os componentes e periféricos associados a este. Nos clientes esta camada é por norma associada a um equipamento físico. No caso do desenvolvimento esta camada foi implementada sobre a forma de uma máquina virtual, utilizando o *software* de virtualização VirtualBox. Para comprovar que o cliente tem a possibilidade de funcionar em múltiplas plataformas, o desenvolvimento foi feito em Windows 8 64-bit e posteriormente foi feita a adaptação para Ubuntu 12.04 Desktop 64-bit.

Chegando à camada de aplicação, esta contém o *software* cliente, bem como as APIs, recursos e interfaces locais. O programa cliente, desenvolvido neste projeto contém as funcionalidades XMPP que são utilizadas para a conexão entre o cliente e o servidor. Para além disso, graças às bibliotecas do Qt, tem também acesso ao sistema local de ficheiros, aos microfones, câmaras, colunas e auscultadores e à interface gráfica. A interface gráfica é oferecida pelo mecanismo Qt Quick, que a partir de QML, permite criar uma camada de interação visual e dinâmica, ao qual o utilizador vai recorrer para interagir com a aplicação [109].

- Servidor:

Representa o sistema central que fará o apoio aos clientes, oferecendo a estes serviços centralizados e serviços de comunicação entre utilizadores. Similar à arquitetura do cliente, contém três camadas. A camada física que representa o dispositivo físico onde o servidor fica alojado. Idealmente poderia ser caracterizada por um sistema de pequenas dimensões, consumo reduzido e baixo ruído, que seria conectado ao *router* da rede doméstica. Neste projeto esta camada é representada por uma máquina virtual configurada no VirtualBox. Na camada do sistema operativo do servidor será instalada uma distribuição de Linux Ubuntu 12.04 Desktop 64-bit.

Por fim, na camada de aplicação estão representadas todas as ferramentas, recursos locais e aplicações, nas quais se inclui um cliente XMPP autónomo. Este cliente apelidado bot XMPP conecta-se ao servidor XMPP como se um cliente habitual se tratasse, tirando o facto de estar alojado no servidor. Estando alojado no servidor, tem acesso ao sistema de ficheiros

local do servidor, às suas interfaces USB e de rede, e interage ainda com o conjunto de ferramentas multimédia FFmpeg e outros módulos que possam ser implementados posteriormente. Será ainda responsável por prestar os serviços de operação de ficheiros e multimédia aos utilizadores do servidor, utilizando o protocolo XMPP como base para a prestação dos serviços. O servidor XMPP é representado pelo ejabberd, servidor que suporta o protocolo XMPP na rede e que vai receber as ligações XMPP dos clientes.

4.3. COMUNIDADES

Quando se considera um contexto doméstico, o servidor é o elemento central que integra utilizadores, dispositivos e serviços. Embora seja possível registar contas para familiares e amigos, possibilitando a partilha de serviços e conteúdos com estes, a utilização de comunidades virtuais, permite a conexão, interação e partilha entre dois ou mais domínios distintos. Na arquitetura apresentada inicialmente está presente a comunidade entre os dois servidores. Significa que todos os utilizadores dos servidores pertencentes à comunidade podem interagir entre si e partilhar conteúdos, como se pertencessem todos a um único servidor.

5. DESENVOLVIMENTO E DESCRIÇÃO FUNCIONAL

Tendo em conta os conceitos e objetivos definidos anteriormente, procedeu-se ao desenvolvimento de um projeto compreendido por uma aplicação cliente que irá correr em dispositivos de utilizadores e uma aplicação servidora que será executada num sistema centralizado, e que fará a gestão dos utilizadores, dos serviços e dos recursos do sistema. Para auxiliar no desenvolvimento do projeto, foi utilizado o conjunto de ferramentas anteriormente descrito e que inclui as *frameworks* Qt e ffmpeg, o servidor XMPP ejabberd, as bibliotecas QXmpp e libvpx, os clientes XMPP pidgin e jitsi, bem como o *software* de virtualização VirtualBox.

5.1. CONFIGURAÇÃO DO EJABBERD E DAS CONTAS DE UTILIZADOR

De seguida procedeu-se à instalação do servidor XMPP ejabberd no servidor e à configuração das contas de utilizador XMPP, utilizadas durante o desenvolvimento. A partir da interface web disponibilizada pelo ejabberd e de uma conta de administrador criada previamente, procedeu-se à criação de três contas de utilizador. Estas contas, user1, user2 e user3, foram registadas no domínio 192.168.1.100, significando que o seu endereço identificativo será respetivamente, user1@192.168.1.100, user2@192.168.1.100 e

user3@192.168.1.100. Posteriormente utilizou-se o jitsi e o pidgin para adicionar cada uma das contas criadas, à lista de contactos das restantes. Este passo foi necessário pois no início, o cliente em desenvolvimento não teria capacidade de adicionar contactos ao roster da conta que lhe seria associado. Assim, foi feita a autenticação do user1 no jitsi e de user2 no pidgin. De seguida, a partir do jitsi (user1) adicionou-se o pidgin (user2) à sua lista de contactos. Ao executar esta operação o pidgin (user2) aceitava o pedido e ao mesmo tempo adicionava o user1 à sua lista de contactos. Repetiu-se o procedimento anterior entre user2 - user3 e user3 – user1. Por fim associou-se a cada cliente uma dessas contas, atribuindo ao cliente do projeto a conta de utilizador user1, ao jitsi a conta user2 e ao pidgin user3.

Assim cada utilizador criado tem na lista de contactos, os restantes utilizadores, independentemente do cliente que seja usado. Pelo facto de anteriormente se ter atribuído à lista de contactos de user1 as restantes contas, permitiu ao cliente em desenvolvimento ter essas contas no seu roster, por estas estarem associadas à conta user1.

Os clientes XMPP mencionados foram utilizados para testar as funcionalidades do cliente durante o desenvolvimento, pois pelo facto da interface gráfica só ter sido desenvolvida posteriormente, exigiu a utilização destes clientes como ferramentas auxiliares para testar com sucesso cada uma das implementações. A utilização de dois clientes distintos, como foi anteriormente explicada, deveu-se ao facto de cada um por vezes utilizar certos mecanismos que são inconsistentes com a biblioteca QXmpp, ou até não conter a funcionalidade que se deseja testar. Dando um exemplo, o jitsi tem um comportamento estranho com o QXmpp em salas de conversação. Quando envia um convite ao cliente desenvolvido, este deteta o convite, mas o contrário não acontece. Já o pidgin não tem problemas em processar os convites que o cliente desenvolvido envia. Por outro lado, este não suporta chamadas, ao contrário do jitsi, que não só suporta, como até permite configurar definições avançadas como codecs de áudio e vídeo utilizados, algo importante para testar chamadas de voz e vídeo no cliente.

5.2. FUNCIONALIDADES DE PRESENÇA, CONTACTOS E COMUNICAÇÃO ENTRE UTILIZADORES

O primeiro passo a dar será no desenvolvimento do sistema pretendido, passa por desenvolver a aplicação cliente e as funcionalidades de presença e comunicação do mesmo, que serão prestadas pelo protocolo XMPP. A *framework* Qt, em conjunto com a biblioteca QXmpp e com o auxílio dos clientes XMPP pidgin e jitsi, constituem a base para o desenvolvimento e análise desta fase. As funcionalidades implementadas são o ***login***, a gestão de **contactos, presença de utilizadores**, o envio e receção de **mensagens instantâneas, criação e interação com salas de *chat,* transferência de ficheiros** entre utilizadores, **chamadas**/**conferências** e **grupos** de utilizadores. A decisão de implementar estas funcionalidades do cliente na fase inicial, prende-se com pelo facto destas serem essenciais para todo o projeto, pelo facto dos serviços posteriores empregarem algumas destas funcionalidades e utilizarem XMPP entre o cliente e o servidor.

Primeiro, implementa-se no Qt a partir da biblioteca QXMPP a conexão e autenticação do cliente ao servidor. O processo de autenticação é inicialmente configurado manualmente com as credenciais da conta user1, pois não existindo inicialmente interface gráfica, é de interesse no início do projeto efetuar o login automaticamente, sendo que um sistema de login será posteriormente implementado. Para testar se o cliente se conecta corretamente, é iniciada a máquina virtual onde o servidor foi instalado, e com o servidor *ejabberd* a correr, executa-se o cliente. A partir da interface web de administração, é possível visualizar as contas de utilizador registadas e quais estão atualmente autenticadas. Verifica-se que este deteta a autenticação. Num segundo teste, é modificada a *password* no cliente, para testar se a autenticação falha. Não só o servidor não detetou qualquer ligação neste caso, como a consola do Qt confirma que a autenticação foi rejeitada.

Após verificar que o cliente se autentica corretamente, é necessário estabelecer uma estrutura lógica que a aplicação cliente deve suportar. Assim são criadas duas classes. A primeira, classe Core, será o módulo central da aplicação cliente. Aqui estarão as funcionalidades lógicas do cliente que incluem, estabelecer e gerir a ligação entre o cliente e o servidor, processar as comunicações entre o cliente e os restantes utilizadores, requisitar e responder aos serviços prestados pelo servidor e aceder aos recursos locais do dispositivo. A segunda classe é responsável pela gestão da interface gráfica e faz de interface entre esta e a parte lógica (Core).

Após a criação da classe Core, o processo de autenticação ao servidor XMPP criado inicialmente é implementado em Core. Todas as funcionalidades implementadas nesta secção serão incluída em Core, já que todas elas executam operações lógicas do cliente.

5.2.1. FUNCIONALIDADES DE PRESENÇA E CONTACTOS

Umas das funções base que o protocolo XMPP permite executar, é a possibilidade de detetar a presença de utilizadores e recursos XMPP, bem como definir o estado destes. De forma semelhante aos clientes de *instant messaging* tradicionais, é possível detetar se um utilizador está presente, o estado de presença, como disponível, ocupado ou ausente e ainda uma mensagem de estado editável pelo do utilizador. Utilizando então esta faceta do XMPP, é altura de integrar no cliente a capacidade de detetar a presença dos restantes utilizadores, bem como os seus recursos, estados e mensagens de estado. Na figura 5 encontra-se o processo de inicialização e o funcionamento do mecanismo de presença e contactos.

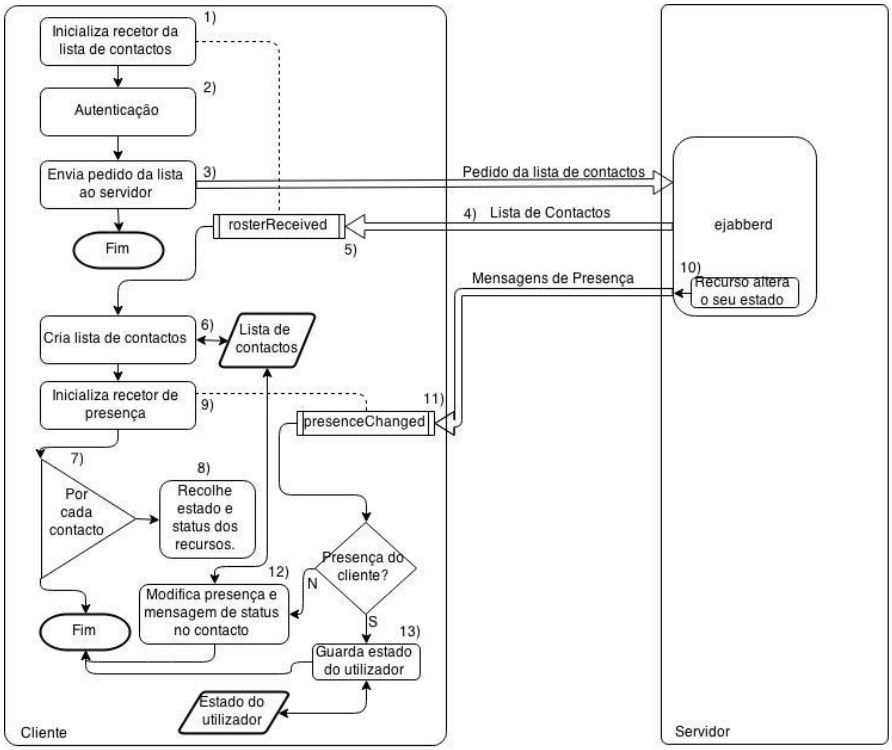


Figura 5 Sistema de Presença e Contactos.

O processo começa por 1) inicializar o recetor do *roster* para detetar quando o cliente recebe a lista de contactos. De seguida, 2) autentica-se no servidor e 3) envia um pedido a requisitar a lista de contactos do utilizador ao servidor XMPP. A resposta surge na forma de uma 4) lista de itens que representam os contactos do utilizador. Após 5) receber esta, 6) cria uma

lista onde os dados do contactos serão armazenados e 7) para cada contacto, 8) requere os seus recursos XMPP ativos e informação de presença destes, incluindo o estado e a mensagem de *status*. Ao mesmo tempo é também 9) iniciado o elemento que recebe atualizações sobre presença de recursos. Quando uma 10) mensagem de presença é emitida pelo servidor, devido a uma modificação no estado de um contacto ou do utilizador, 11) o recetor deteta esta mensagem. Se esta não for referente ao estado do utilizador, 12) atualiza a lista local com o novo estado e mensagem de *status* do contacto em questão. Se a atualização de estado for referente ao próprio, então 13) guarda esses dados localmente, oferecendo ao cliente acesso à sua própria informação de presença.

Deste modo o sistema de presenças do cliente consegue recolher a lista de contactos e monitorizar a presença e os recursos de cada um. Com este processo consegue ainda monitorizar a sua própria informação de presença e os recursos que o utilizador atual possui ativos.

5.2.2. MENSAGENS INSTANTÂNEAS

O sistema de mensagens instantâneas é implementado de forma a ser iniciado após o sistema de presença, anteriormente descrito. O diagrama da figura 6 descreve com pormenor os elementos do sistema de mensagens, a sua função e a interação entre si e com o servidor.

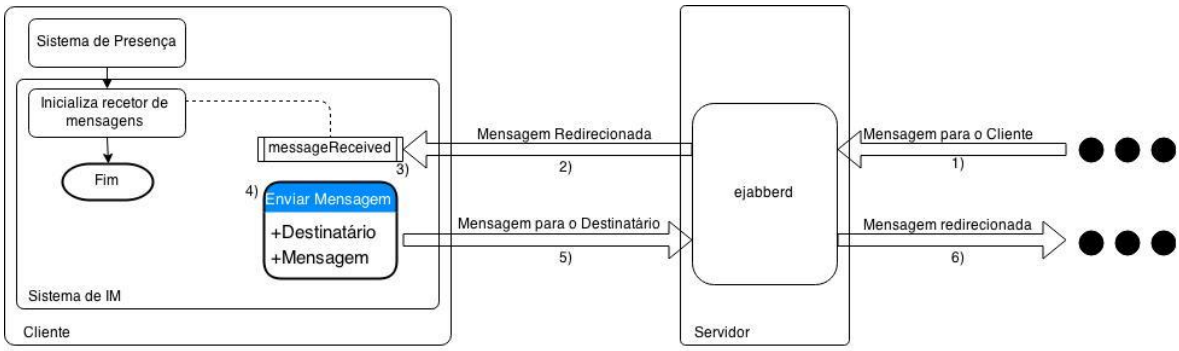


Figura 6 Sistema de Mensagens Instantâneas.

O processo de inicialização do sistema *Instant Messaging*, inicializa o módulo que deteta a receção de mensagens XMPP recebidas pelo cliente. É implementada também a função encarregue de enviar as mensagens do cliente para os seus contactos. O mecanismo de mensagens funciona da seguinte forma: 1) Quando uma entidade envia uma mensagem

instantânea ao cliente, esta é enviada ao servidor XMPP (*ejabberd*) 2) que a irá redirecionar para o cliente. Quando este receber a mensagem, 3) o recetor deteta o evento. Já o 4) módulo que envia mensagens aceita o *barejid* do destinatário e o conteúdo da mensagem como parâmetros de entrada. A 5) mensagem será então enviada ao servidor, que 6) tratará de redirecionar para o destinatário.

5.2.3. SALAS DE CHAT

A integração do sistema de salas de conversação é feita com base na XEP-0045 Multi-User Chat. Esta extensão permite que utilizadores criem e se juntem a salas de conversação, onde podem comunicar simultaneamente com os todos os participantes, suportando tópicos de conversa e convites, e dispõe de um sistema de controlo, onde é possível nomear moderadores e administradores, expulsar ou banir utilizadores e até configurar salas restritas com *password* ou com acesso reservado a membros.

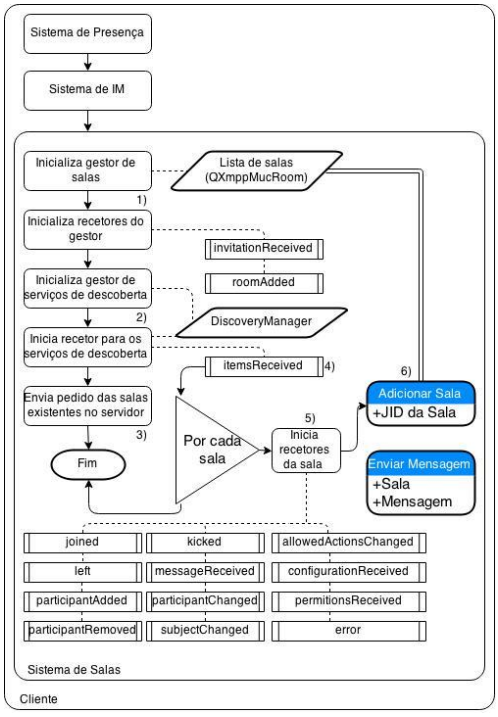


Figura 7 Sistema de Salas de Conversação.

O sistema de salas começa por iniciar o gestor de salas de conversação. Este gestor, *QXmppMucManager* pertencente à biblioteca *QXmpp* é a estrutura que gere de forma

transparente, as salas de *chat XMPP* às quais o cliente tem acesso. Permitirá adicionar de novas salas e disponibiliza uma lista de itens que compreende todas as salas geridas.

Após a inicialização do 1) gestor de salas e dos seus recetores, é feito um pedido ao servidor sobre as salas de conversação atualmente ativas no serviço MUC, neste caso chamado *conference*. Este passo é efetuado pois o gestor por si só, não é capaz de detetar as salas existentes no servidor. Assim, de forma a detetar salas públicas já existentes, é necessário um mecanismo alternativo. Aqui entra o DiscoveryManager [110]. A partir deste gestor do QXmpp, é possível efetuar pedidos ao servidor XMPP, sobre os serviços que este disponibiliza. 2) O gestor de descoberta de serviços é então inicializado bem como os seus recetores, e baseado na XEP-0030: Service Discovery é 3) requisitado ao servidor a lista de salas públicas associadas ao serviço MUC *(Multi-User Chat*), na forma de itens [111]. O servidor vai responder com uma lista de itens que contém o nome e o jid da sala.

Pedido:

```
<iq from='user1@192.168.1.100/pc'
   id='u3de5r3t'
   to='conference.192.168.1.100'
   type='get'>
  <query xmlns='http://jabber.org/protocol/disco#items'/>
</iq>
```

Resposta:

```
<iq from=' conference.192.168.1.100'
   id=' u3de5r3t'
   to=''user1@192.168.1.100/pc '
   type='result'>
 <query xmlns='http://jabber.org/protocol/disco#items'>
    <item jid='sala1@ conference.192.168.1.100' name='Sala1'/>
    <item jid='family@ conference.192.168.1.100' name='Sala da Família'/>
    <item jid='vip@ conference.192.168.1.100' name='Sala Reservada'/>
 </query>
</iq>
```

Figura 8 A) Pedido da lista de salas de *chat* ao serviço *conference* do servidor. B) Resposta ao pedido.

Após 4) receber a lista de salas do servidor a partir do recetor *itemsReceived*, para cada uma 5) conecta os recetores que vão receber os eventos de todas as salas e 6) adiciona esta ao gestor de salas. Cada um dos módulos recebe um evento específico de todas as salas, pois este é capaz de detetar qual das sala geridas emitiu o sinal. Desta forma é possível concentrar os eventos de todas as salas apenas nos módulos apresentados.

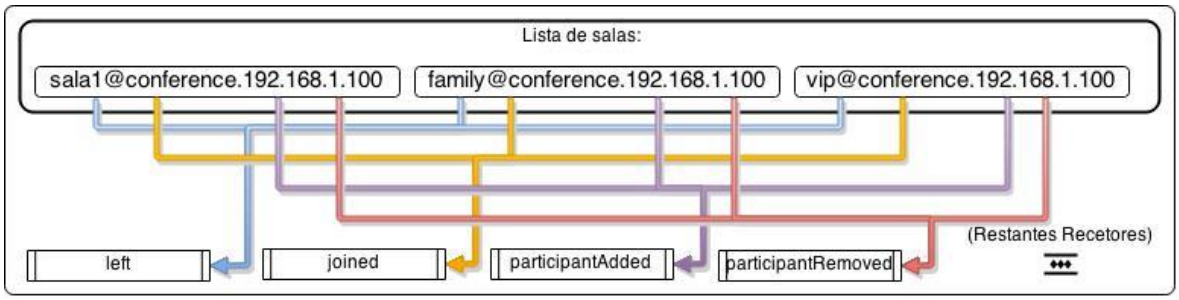


Figura 9 Ilustração da concentração dos eventos nos recetores de eventos das salas.

Por fim é implementados o módulo que envia mensagens para as salas.

5.2.4. TRANSFERÊNCIAS DE FICHEIROS

De seguida é implementada a transferência de ficheiros, utilizando para tal o gestor *QXmppTransferManager*. Este gestor da biblioteca *QXmpp* vai permitir controlar transferências de ficheiros, utilizando mensagens XMPP como mecanismo de sinalização. Este *manager* utiliza as extensões XEP-0095 Stream Initiation e XEP-0096 SI File Transfer para a sinalização e controlo da transferência [112] [113]. Para a transferência em si foi utilizada a extensão XEP-0065 SOCKS5 Bytestreams [114], que utiliza um *proxy* SOCKS5 para reencaminhar a transferência entre os clientes possibilitando uma maior performance na transferência de ficheiros.

Após iniciar o sistema de presença, IM e salas, o cliente inicializa o sistema de transferências. No processo de inicialização 1) é iniciado o gestor de transferências do QXmpp onde são armazenadas as referências das transferências de ficheiros relativas ao cliente, bem como o 2) recetor que deteta pedidos de transferência recebidos. Termina assim o processo de inicialização. É ainda implementada a função que permite 3) enviar ficheiros

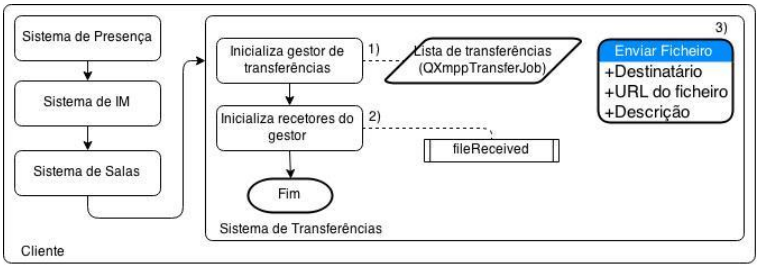


Figura 10 Sistema de transferências de ficheiros.

O processo que ocorre numa transferência está ilustrado na figura 11.

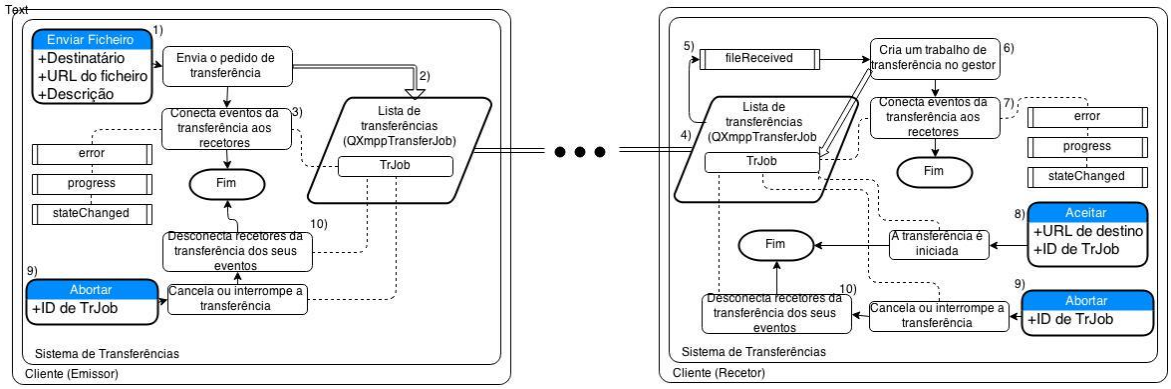


Figura 11 Processo de uma transferência.

Quando um cliente pretende enviar um ficheiro, 1) efetua um pedido de transferência 2) a partir do gestor de transferências sistema e 3) os eventos da transferência são conectados aos recetores. 4) Ao receber um pedido de transferência de ficheiros, o gestor 5) emite o sinal correspondente, 6) é gerado um trabalho de transferência e 7) os emissores de eventos da transferência são conectados aos módulos recetores de eventos. Nesse momento o recetor pode 8) aceitar a transferência e ambos os intervenientes podem 9) abortar ou interromper a mesma, durante o seu desenrolar. Quando a transferência é interrompida ou termina, 10) os emissores referentes a esta são desconectados em ambas as partes, para que quando for posteriormente eliminada da lista, as suas conexões não permaneçam ativas.

O mecanismo de transferências de ficheiros utilizado tem como destinatário um fullJid como por exemplo user1@192.168.1.100/pc. Significa isto que um pedido de transferência é feito a um recurso específico e não ao utilizador. Neste caso é implementado um sistema que em vez de emitir o pedido apenas para um recurso, este é emitido para todos os recursos que ativos do utilizador. Assim todos os dispositivos ativos do recetor detetam a transferência. Quando qualquer um dos recursos aceita ou rejeita a transferência, o cliente rejeita as transferências dos restantes recursos.

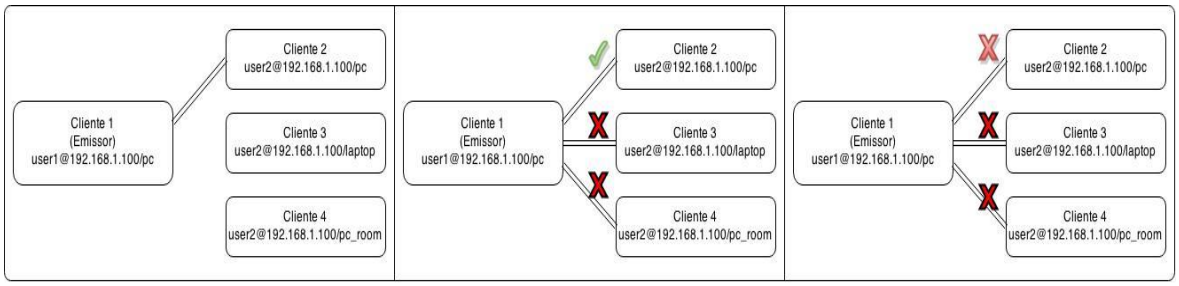


Figura 12 A) Pedido de transferência convencional. B) Pedido de transferência com aceitação. C) Pedido de transferência com rejeição.

Desta forma todos os dispositivos do recetor da transferência recebem o pedido de transferência, e o utilizador pode aceitar ou rejeitar a mesma em qualquer um deles.

Termina assim a implementação do sistema de transferência de ficheiros, permitindo o envio e receção de ficheiros entre clientes. Esta funcionalidade será também importante para serviços mais complexos que necessitem de transferir ficheiros entre entidades.

5.2.5. CHAMADAS/CONFERÊNCIAS

As chamadas estabelecidas a partir da extensão Jingle, utilizam por norma uma nomenclatura p2p entre clientes. Sendo os mecanismos p2p descentralizados por natureza, não é possível criar um canal comum onde quando um utilizador conectado fale, todos os restantes utilizadores ouçam a sua voz, como acontece com as mensagens de uma sala de *chat*. Assim de forma a possibilitar chamadas e conferências multimédia, cada utilizador estabelece múltiplas chamadas em simultâneo com os restantes participantes da conferência. Como se pode observar na figura 13, os vários intervenientes estão conectados ao servidor XMPP, utilizando o protocolo para sinalização das chamadas. As sessões multimédia em si são estabelecidas diretamente entre os clientes.

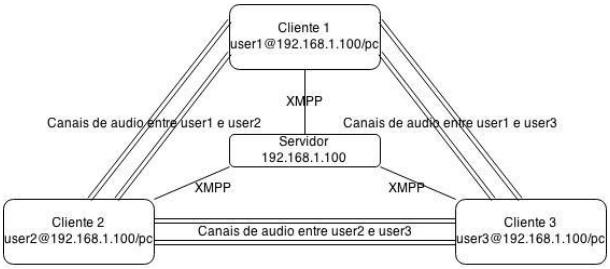


Figura 13 Nomenclatura de uma conferência de voz.

Quando um utilizador pretende criar uma conferência, é criada uma sala de conversação no servidor. A sala pode ser pública ou privada, livre ou protegida e de acesso público ou apenas para utilizadores subscritos. Após entrar na sala, convida os utilizadores com quem pretende comunicar para a mesma. Quando um utilizador se conecta à sala, este estabelece automaticamente uma chamada com os restantes participantes da mesma. Se um utilizador pretende sair da chamada, basta sair da sala e automaticamente, todas as chamadas com os restantes ocupantes são terminadas. Para um utilizador se juntar a uma conferência, basta ser convidado e aceitar, ou ele próprio conectar-se à sala.

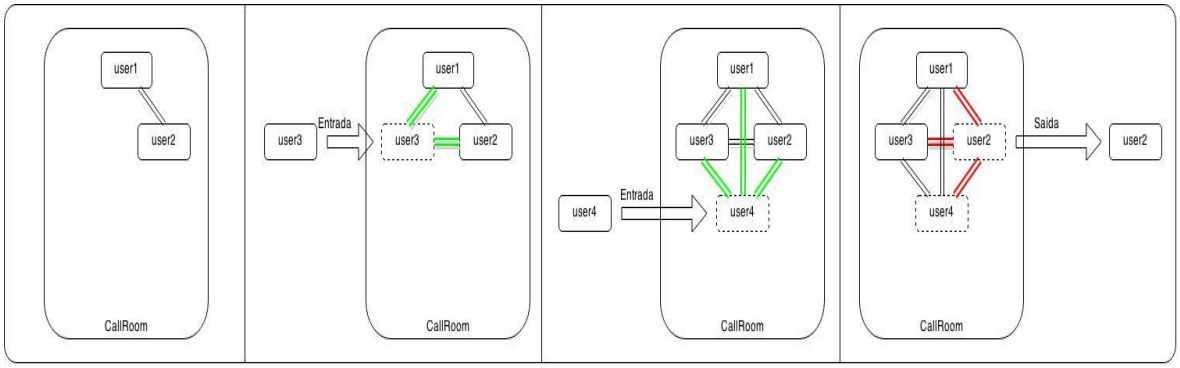


Figura 14 Ilustração de uma sala de conferências.

Ao associar as conferências a salas de conversação, permite criar um mecanismo simples e centralizado. Simplifica o estabelecimento e término das múltiplas sessões multimédia entre os participantes da conferência, bem como o controlo das chamadas.

Sinalização e canais multimédia:

O gestor de chamadas *QXmppCallManager* é o responsável por efetuar a troca de mensagens de sinalização e o estabelecimento de sessões multimédia, utilizando como base as extensões *XEP-0166 Jingle* [115]e *XEP-0167 Jingle RTP Sessions* [116]para a negociar e controlar sessões multimédia entre entidades XMPP e a extensão XEP-0176 Jingle ICE-UDP Transport [117] para estabelecer os canais UDP, onde o conteúdo multimédia transita sobre a forma do protocolo RTP (*Real-time Transport Protocol*).

O canal XMPP entre os clientes e o servidor será utilizado para os dois clientes trocarem entre si, mensagens de sinalização que permitem controlar as chamadas. Se a chamada for aceite, são então estabelecidos entre os clientes XMPP, dois canais RTP pelo método ICE-

UDP para transmissão do áudio da chamada de uma forma *full-duplex*, diretamente entre si numa nomenclatura *peer-to-peer (p2p)*.

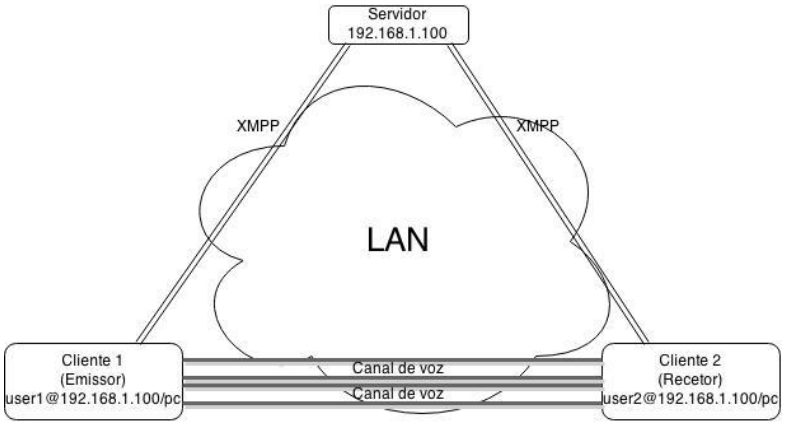


Figura 15 Chamada Convencional (P2P)

No caso de uma chamada ser estabelecida localmente, onde ambos os intervenientes estão na mesma rede local, as ligações são estabelecidas sem bloqueios. O mesmo não acontece caso um ou ambos os intervenientes estejam na internet. Neste último caso as NATs (*Network Address Translations*), existentes nos *routers* domésticos e *gateways* de rede, podem bloquear as sessões multimédia, impedindo o estabelecimento de chamadas entre as duas partes. Para resolver este problema existem técnicas que permitem ultrapassar as NATs.

O NAT transversal é a arte de a partir de um processo de troca de mensagens de rede entre as duas entidades que pretendem comunicar, esta consigam estabelecer ligações entre si, com o objetivo de ultrapassar NATs e *firewalls* que possam interferir com as ligações.

O ICE (*Interactive Connectivity Establishment*) é utilizado para estabelece uma ligação UDP entre dois clientes de forma P2P, utilizando mensagens XMPP para a negociação da ligação. Suporta *NAT Traversal* a partir do mecanismo STUN (*Session Traversal Utilities for NAT*), possibilitando a clientes XMPP que estejam atrás de uma NAT, poderem estabelecer entre si uma conexão peer-to-peer UDP. Se o STUN não funcionar devido às configurações das NATs, é então utilizada a técnica TURN (*Traversal Using Relays around NAT*), que utiliza o servidor como *relay* entre os clientes.

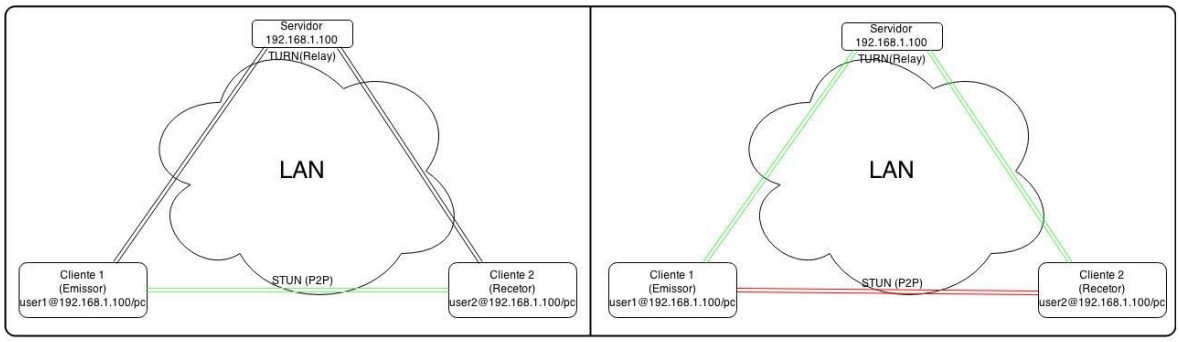


Figura 16 A) Estabelecimento de ligações por STUN (P2P). B) Estabelecimento de ligações por TURN (*relay*).

Embora o TURN seja menos eficiente em relação ao STUN, pois o servidor tem de processar o tráfego entre os clientes, como estes estão ligados ao mesmo por XMPP, será um mecanismo robusto que acrescenta uma camada de redundância ao sistema, no caso das ligações diretas falharem.

A figura 17 descreve as várias etapas que ocorrem durante uma chamada XMPP.

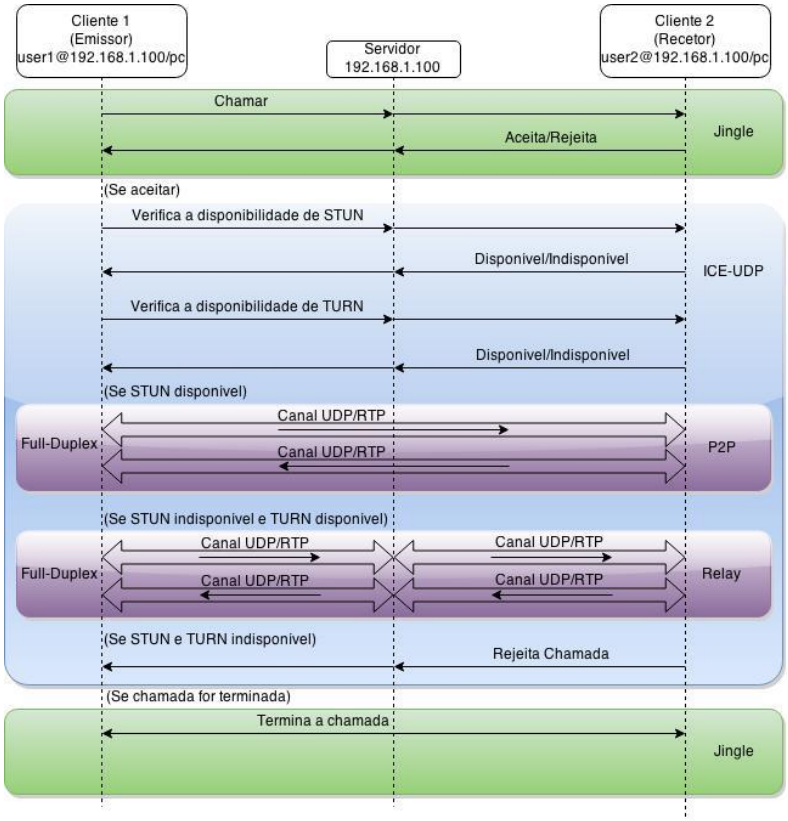


Figura 17 Etapas de uma chamada XMPP

Inicialmente é utilizada a extensão Jingle do XMPP para notificar o recetor da chamada e controlar a mesma. Será encarregue de detetar a aceitação por parte do recetor ou a anulação por parte de ambas as partes, controla o estado da chamada durante o seu decorrer e é responsável por terminar a mesma. Se a chamada for aceite é então utilizado o protocolo ICE-UDP que tentará estabelecer dois canais UDP diretos entre os participantes da chamada, com apoio do mecanismo STUN. Caso não seja possível, é então utilizado o mecanismo TURN para estabelecer os dois canais entre os participantes, utilizando o servidor como *relay*. Após estabelecer os canais UDP, dentro destes são estabelecidos canais RTP para a transmissão de conteúdo multimédia.

5.2.6. RESUMO

Está concluída a implementação das funcionalidades de presença, contactos e comunicação entre utilizadores no cliente.

Nesta fase o cliente já é capaz de se autenticar, detetar a sua presença, presença dos recursos do utilizador atual e presença dos contactos da sua lista, envio e receção de mensagens instantâneas, a criação e interação com salas de conversação, o envio e a receção de ficheiros e chamadas e conferências de voz. Os grupos de utilizadores e o sistema de notificações serão implementadas posteriormente, pois a sua integração será facilitada com a presença da interface.

5.3. INTERFACES GRÁFICAS

A interface gráfica do cliente deve ser capaz de concentrar os serviços já oferecidos num sistema fácil de utilizar, onde seja possível selecionar os contactos com os quais se deseja utilizar um determinado serviço. Também deve considerar os serviços prestados pelo servidor que serão implementados futuramente. Com estas considerações em mente, surge a seguinte estrutura de interfaces.

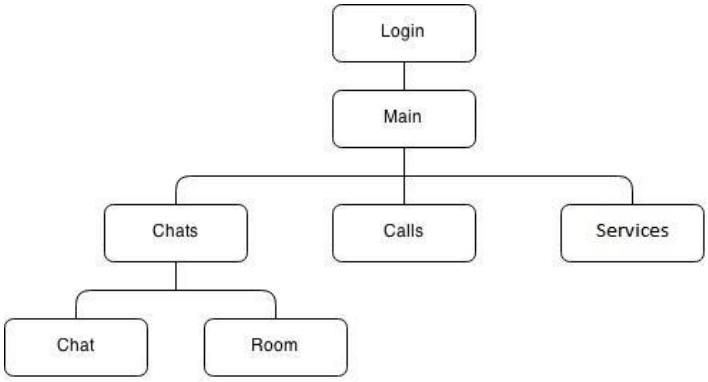


Figura 18 Estrutura de interfaces.

Quando a aplicação cliente é iniciada, surge a interface de autenticação (Login). Caso o login seja efetuado com sucesso, a interface inicial dá lugar à interface principal (Main). Esta irá conter os dados do utilizador, lista de contactos e recursos, opções e os serviços de comunicação com utilizadores. Terá ainda um sistema visual que permite monitorizar, controlar e receber notificações de todos os eventos do sistema, como mensagens e chamadas, recebidas e enviadas.

São ainda consideradas três interfaces. (*Chats)* acomoda as conversações individuais e salas de conversação nas quais o utilizador está conectado. Também é incluída a interface referente às chamadas (*Calls*). Por fim a interface (Services) permite aceder aos serviços de gestão de pasta e ficheiros e ao serviço de VoD e AoD. Estas interfaces estão inicialmente scondidas e surgem quando são necessário.

5.3.1. LOGIN

Login é a primeira interface gráfica a surgir na aplicação cliente. Quando esta é iniciada a interface é apresentada ao utilizador de forma a proceder à autenticação e acesso ao servidor.

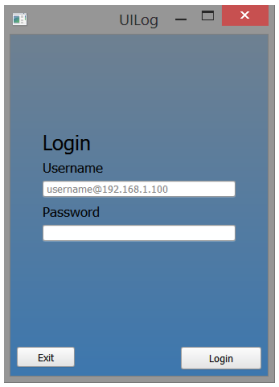


Figura 19 Interface de *Login*.

Na interface de autenticação estão presentes dois campos, três opções e o botão "Autenticar".

- *Username*: Campo onde é inserida a conta de utilizador, que corresponde efetivamente ao *jid* do utilizador (ex.: user1@192.168.1.100).
- *Password*: Campo onde o utilizador insere a senha da conta.
- Botão "Autenticar": Botão que permite iniciar o processo de autenticação com as credenciais inseridas nos campos respetivos.
- Botão "Exit": Permite sair da aplicação a partir desta interface.

5.3.2. MAIN

A interface principal (Main) tem o layout apresentado na figura 20.

Figura 20 Interface principal.

Esta pode ser separada em seis secções, cada uma com uma função.

- Utilizador: Área que possui o nome do utilizador, o nome do recurso, estado, mensagem de *status* e fotografia.
- Serviços centralizados: Botão à qual se acede à interface de serviços prestados pelo servidor. Este irá lançar uma nova interface para o efeito.
- Separadores de listas: Três separadores, que permitem selecionar a lista de elementos pretendida. As opções são contactos, grupos e salas.
- Lista atual: Dependendo da opção selecionada, apresenta a lista de elementos correspondentes. No caso dos contactos, é apresentada a lista de contactos, com os nomes e estados de cada um destes. Para as restantes opções podem ser visualizadas as listas de recursos do utilizador, grupos de contactos e salas às quais o utilizador tem acesso.
- Serviços: Lista de ícones à direita, referente aos serviços de comunicação do sistema.
- Centro de controlo: Lista situada na parte esquerda da interface, que oferece o mecanismo de notificações e controlo do sistema. Os dois valores em cada ícone representa as ocorrências de envios e receções.

No caso de o utilizador clicar em qualquer elemento do centro de controlo, faz surgir/esconder o painel de notificações. Este elemento ainda não existe e irá posteriormente conter todas as notificações relativas ao sistema, permitindo ao utilizador e filtrar os resultados, clicando no elemento em específico.

5.3.3. CHATS

Esta interface concentra numa só janela, todas as conversações individuais e em salas de *chat*.

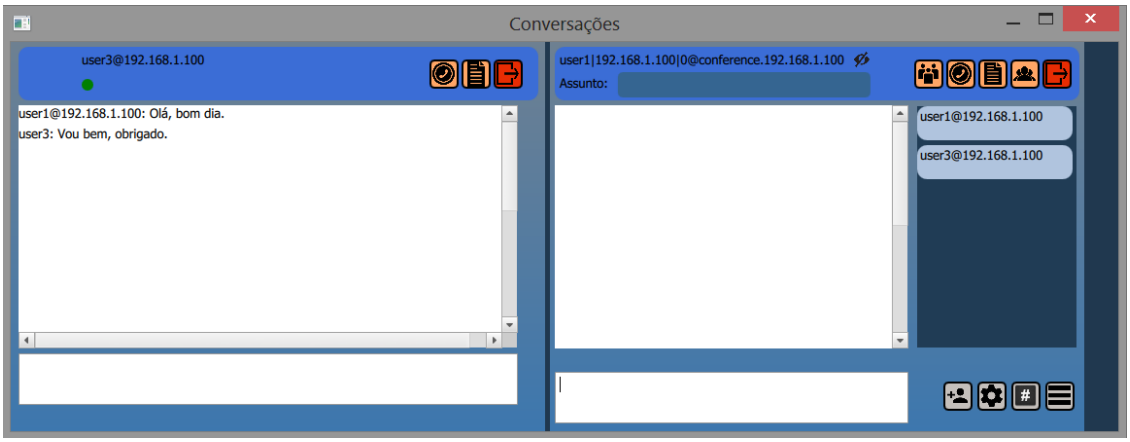


Figura 21 Interface "Chats" com uma conversação individual e uma sala de conversação.

Contém oito componentes importantes.

- Dados da seção: Contém o nome e estado no caso de uma conversação individual. No caso de uma sala contém o nome, os ícones do tipo de sala e o assunto.

- Lista de serviços: Lista de ícones que permitem automaticamente aplicar um serviço de comunicação ao(s) participante(s) da conversação, como enviar ficheiros ou iniciar uma chamada.
- Janela de mensagens: Janela que mostra as mensagens trocadas na conversação ou sala.
- Notificação de estado: Campo de texto normalmente vazio, mas que identifica o ou os utilizadores que estão de momento a compor uma mensagem.
- Campo de mensagem: Campo onde o utilizador digita a mensagem pretendida.
- Ocupantes: Campo que contém o nome dos ocupantes da sala atual.
- Opções da sala: Conjunto de botões que permitem administrar a sala atualmente ativa. Só está visível se o utilizador for o administrador ou dono da sala. A partir dos quatro ícones de gestão da sala, o administrador pode convidar, configurar a sala, configurar os privilégios dos utilizadores e visualizar os mesmos.

Cada vez que o utilizador enviar ou receber uma mensagem de um contacto com o qual já não está comunicar, é criada uma nova conversação e um novo separador. O mesmo processo ocorre com salas, quando um utilizador se junta a uma nova sala, é criado um separador que representa esta, na lista de separadores por baixo da janela de mensagens.

5.3.4. CALLS

De forma a poder usufruir da funcionalidade de chamadas é implementado no cliente uma interface que contenha os elementos necessários para o efeito.

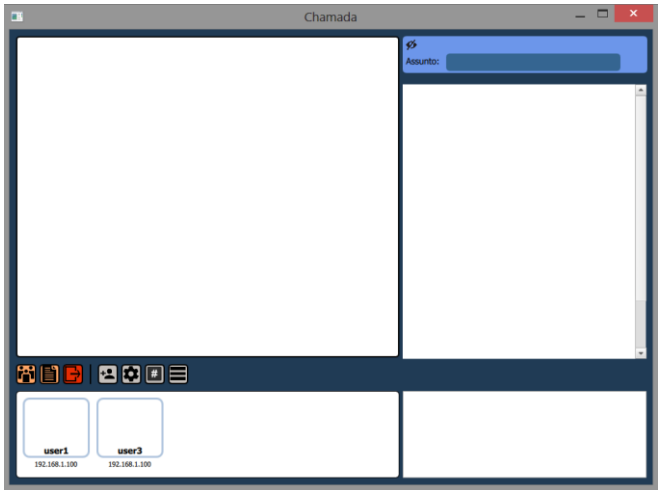


Figura 22 Interface de Chamada.

Os elementos constituintes de *Calls* são os seguintes:

- Dados da chamada: Campo onde se encontram os dados da chamadas, como assunto e tipo de chamada (pública, protegida, reservada, etc…).

- Lista de serviços: Lista de ícones que permite automaticamente aplicar um serviço de comunicação ao(s) participante(s) da chamada como enviar ficheiros ou iniciar uma chamada.
- Campo de mensagem: Campo onde o utilizador pode digitar mensagem escritas.
- Notificação de estado: Campo de texto normalmente vazio, mas que identifica o ou os utilizadores que estão de momento a compor uma mensagem escrita.
- Janela de mensagens: Janela que contém todas as mensagens de texto da conversação.
- Opções da chamada: Conjunto de botões que permitem configurar as preferências da chamada e administrar a mesma no caso de o utilizador ser o administrador da chamada. A partir do quatro ícones de gestão da chamada, o administrador pode convidar, configurar a chamada, configurar os privilégios dos utilizadores e visualizar os mesmos.

5.3.5. RESUMO

Está terminado o dimensionamento e a implementação da interface de autenticação, interface principal, janela de conversações, bem como a janela de chamadas. Desta fase resulta a base do sistema de interface, ao qual o utilizador vai recorrer para interagir com o sistema. Os elementos aplicados são os componentes primários das interfaces, sendo que a parte lógica e elementos secundários da interface, serão implementados nas etapas seguintes. A interface de serviços também será implementada posteriormente.

5.4. INTEGRAÇÃO DA INTERFACE COM OS SERVIÇOS

Após a implementação da vertente gráfica das interfaces do cliente, é altura de integrar as mesmas com a parte lógica do cliente. Nesta secção é apresentada a integração das funcionalidades implementadas na secção 5.2 (***Funcionalidades de Presença, Contactos e Comunicação Entre Utilizadores***), com as interfaces desenvolvidas em 5.3 (***Interfaces Gráficas***).

5.4.1. INTEGRAÇÃO DO LOGIN

Utilizando como base a interface Login, o mecanismo de autenticação é implementado da seguinte forma.

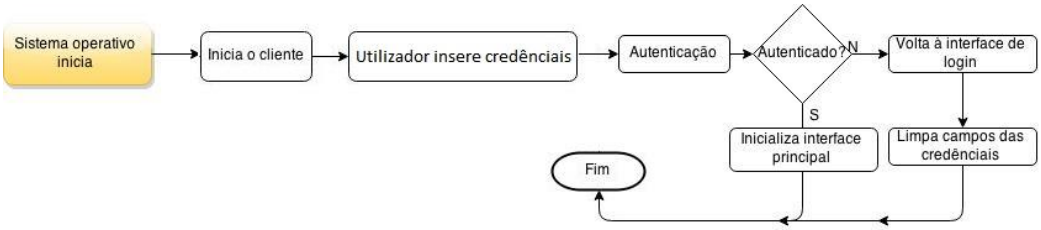


Figura 23 Sistema de autenticação.

Quando o cliente inicia, é apresentado ao utilizador a interface Login, na qual irá digitar as suas credenciais e clicar no botão "Autenticar". Se esta for bem-sucedida, a interface de autenticação dá lugar à interface principal. No caso de falhar, surge uma mensagem com o problema que ocorreu. Aqui podem ocorrer duas situações. O cliente não consegue detetar o servidor ou as credenciais de utilizador são inseridas de forma errada.

5.4.2. INTERFACE PRINCIPAL

A interface principal (*Main*) será responsável pelos dados do utilizador, visualização de contactos, recursos, grupos e salas, interação com estes por meio dos serviços de comunicação e deverá ainda conter o sistema de notificação que permite a agregação de eventos relativos ao sistema. É necessário assim a integração da interface e dos seus elementos gráficos com as funcionalidades já introduzidas.

A primeira adaptação da interface Main corresponde ao sistema de presença implementado em 5.2.1. Quando o cliente se autentica corretamente, o nome do utilizador e o nome do

recurso do dispositivo são enviados para a interface que irá apresentar os respetivos elementos. De seguida são requisitados ao servidor, o estado, mensagem de *status* e fotografia do utilizador, que são enviados de seguida para a interface. O utilizador pode modificar o estado, clicando sobre o elemento e modificando-o com as opções disponíveis.

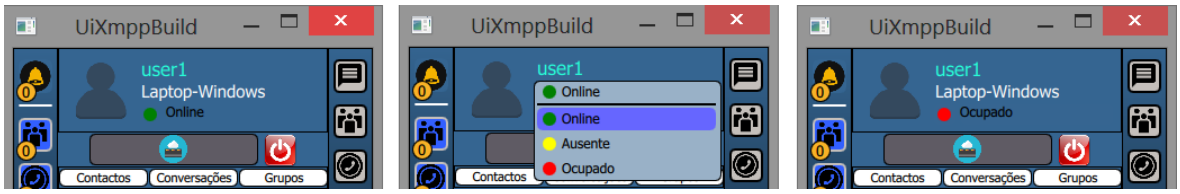


Figura 24 Ilustração do processo de alteração de estado.

A mensagem de estado também pode ser modificada, sendo que para isso se clica sobre a mesma, neste caso "Online" ou "Ocupado", e digitar a mensagem.

Segue-se o desenvolvimento do sistema de seleção que permite selecionar contactos, recurso, grupos e salas, para posteriormente aplicar um serviço aos elementos selecionados. Primeiro implementa-se o sistema de listas de contactos, salas e grupos. Estas estruturas agrupam componentes que representam cada tipo de lista.

Figura 25 A) Lista de utilizadores. B) Lista de salas. C) Lista de grupos.

Contacto: Contém a fotografia, o nome, o estado e a mensagem de *status* de um contacto do utilizador.

Sala: Contém o nome da sala, os ícones que representam o tipo de sala e as opções para entrar e eliminar a sala, no caso de ser administrador. Os ícones representam o tipo de conversação. Telefone para chamada, cadeado para conversação reservada e chave para conversação com senha.

Grupos: Contém o nome do grupo, número de contactos totais, bem como o número de contactos ativos. No caso de passar o rato sobre o elemento de um grupo é possível visualizar o nome dos utilizadores do grupo e a sua presença.

Após dimensionados os componentes para cada caso, é altura de implementar um sistema que agrupe estes componentes em listas verticais dinâmicas, e adaptar estas estruturas para reagirem aos eventos do sistema. Para o efeito foram utilizados os elementos QML *ListView* e *ListModel*. Estes permitem incorporar uma lista dinâmica, com componentes que podem ser adicionados, removidos, alterados e movimentados entre si. Assim é possível atualizar a lista conforme os eventos e o estado das entidades que esta representa. Mais informações sobre este componentes QML, segue-se a documentação oficial [118] [119].

Na figura 26 ilustra o funcionamento do sistema de listas, quer do ponto de vista de seleção, quer do ponto de vista de atualização, conforme os eventos que o cliente recebe do servidor.

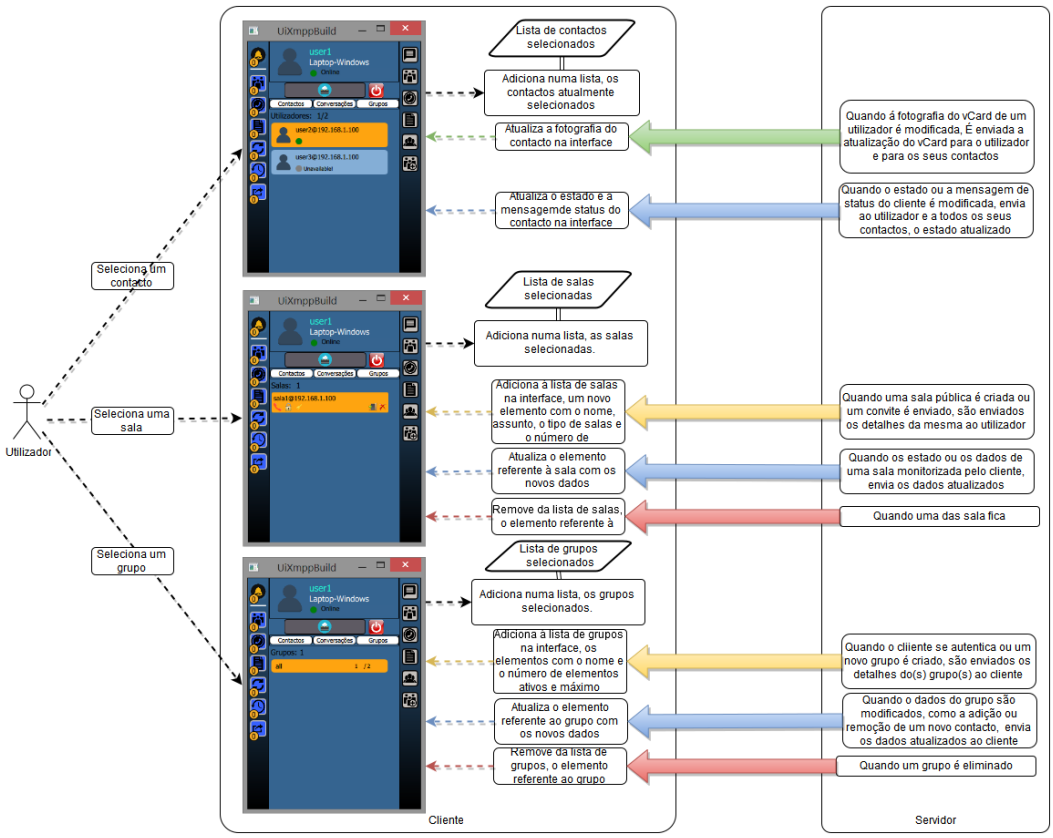


Figura 26 Sistema de Listas e Seleção.

Observando o sistema de listas e seleção existe um fator comuns a todas as listas. Todos os elementos das listas podem ser selecionados. Cada contacto selecionado é adicionado à lista de contactos selecionados. Quando uma sala é selecionada, esta é adicionada à lista de salas selecionadas, sendo o mesmo procedimento efetuado no caso dos grupos. Após implementado o sistema de listas é altura de adaptar este para suportar a seleção de serviços.

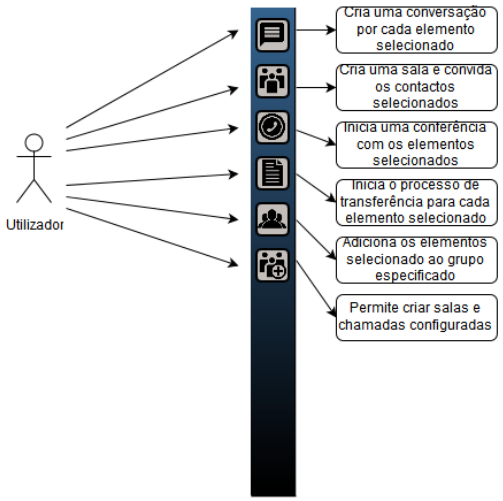


Figura 27 Sistema de seleção de serviços.

Cada ícone presente na coluna lateral direita da interface principal representa ações e serviços que o utilizador pode aplicar aos seus contactos. No momento em que escolhe uma destas opções, o cliente recolhe todos os utilizadores selecionados, todos os utilizadores das salas selecionadas e todos os utilizadores dos grupos selecionados. Elimina duplicados do mesmo utilizador e cria uma lista que representa todos os utilizadores aos quais será aplicada a opção escolhida.

Quando seleciona a opção *chat* ou sala, se já não estiver visível, surge a interface de chat, e posteriormente, é criada uma nova conversação individual ou uma nova conversação de sala. Se a conversação for individual, caso desta já existir uma para o contacto selecionado, não é criada uma nova conversa. As chamadas funcionam da mesma forma que as salas. Quando o utilizador seleciona esta opção, é estabelecida uma conferência entre este e os contactos selecionados, com base no mecanismo apresentado em 5.2.5 (**Chamadas**/***Conferências***), onde é utilizada uma sala de conversação para suportar a funcionalidade. Quando o utilizador se conecta a uma sala de chamada, surge a interface *(Call)*.

Para enviar ficheiros aos seus contactos, após a seleção dos mesmos, pode ser selecionado o ícone correspondente. De seguida é apresentado ao utilizador a interface para selecionar o ficheiro pretendido.

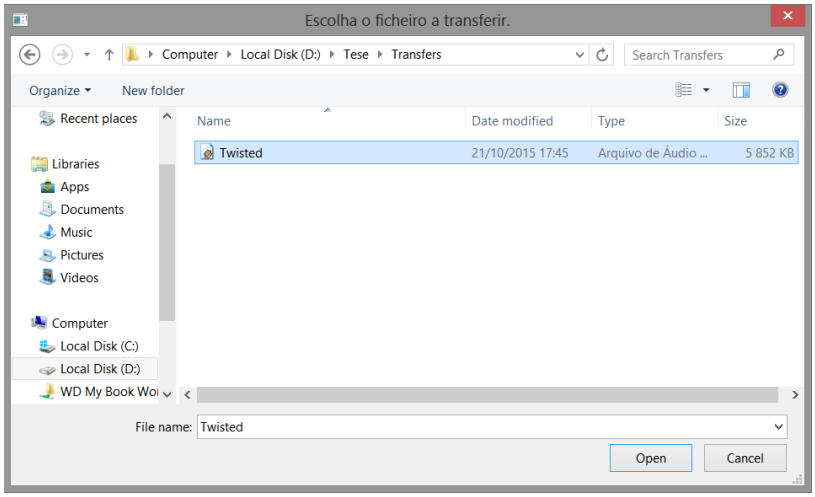


Figura 28 Interface de seleção de ficheiros.

Após confirmar o ficheiro a transferência é efetuada, com base o mecanismo descrito em 5.2.4 (**Transferências de Ficheiros**).

Na seleção da opção Grupo o utilizador é presenteado com o painel de grupos: Este elemento permite adicionar os contactos selecionados um grupo e criar novos grupos. Existe um grupo chamado "*all*" que contém todos os contactos do utilizador e que é gerado automaticamente quando uma conta de utilizador é criada. No caso de pretender criar uma sala ou chamada de conversação personalizada, o utilizador pode recorrer à opção "Criar conversação", que faz surgir o painel de criação de conversações.

Figura 29 A) Painel de grupos. B) Painel de configuração de sala. C) Painel de configuração de sala com sala protegida selecionada.

Após selecionar opções da sala desejadas e confirmar as mesmas, será criada a sala no servidor, esta será configurada com as opções definidas e por fim o utilizador conecta-se automaticamente à sala.

5.4.3. RESUMO

Fica assim concluída a fase de integração da interface com os serviços. Neste momento a área de utilizador, bem como as listas de contactos, grupos e salas, reagem aos eventos recebidos do servidor, oferecendo ao utilizador a informação de forma gráfica e intuitiva. Estes elementos são interativos e podem ser selecionados, para que possam ser posteriormente aplicados aos mesmos, os serviços de comunicação disponíveis. A interface de *chat* já mostra mensagens, deteta novas conversações e permite a partir dos botões

disponíveis, executar serviços de comunicação e gerir as opções da sala selecionada. Por fim a integração da interface de chamadas com o mecanismo de chamadas e conferências apresentado em 5.2.5 (**Chamadas**/***Conferências***), já possibilita ao cliente efetuar e controlar chamadas e conferências.

5.5. SISTEMA DE NOTIFICAÇÕES E CENTRO DE CONTROLO

Dado que o número de serviços é considerável e que ainda serão implementados serviços centralizados, é importante oferecer ao utilizador, um mecanismo que centralize todos os eventos e notificações do sistema. Utilizando como base a interface do centro de controlo presente na interface principal à esquerda, foi integrado o sistema de notificações. Este é responsável por notificar os utilizadores de eventos relevantes. O mecanismo utilizado para difundir as notificações é baseado na extensão XMPP XEP 0060 *Publish-Subscribe*. Esta extensão também conhecida por pubsub permite a criação do nós, aos quais os utilizadores se podem registar. Após o registo, é possível a leitura e edição (*publisher*), apenas a leitura (*member*) ou apenas a edição (*publish-only*) de tópicos do nó. Para mais informações sobre a XEP-0060 dirigir-se a [120].

Cada utilizador vai ter um nó no serviço pubsub com o nome do próprio utilizador e este será publisher do mesmo. Significa que user1@192.168.1.100 terá no seu servidor um nó chamado user1 e os dispositivos (recursos) em que o utilizador se autenticar, vão conseguir ler, inserir, editar e eliminar notificações desse nó. Desta forma todos os recursos do utilizador têm a possibilidade de enviar notificações para o nó do utilizador e ler as notificações do mesmo. Para além dos utilizadores, o *bot* também subscreve os nós dos utilizadores do servidor como *publish-only*, permitindo assim o envio de notificações de serviços prestados pelo mesmo, diretamente aos utilizadores. Na figura 30 é possível observar o processo que ocorre quando o utilizador recebe um convite para uma sala de conversação.

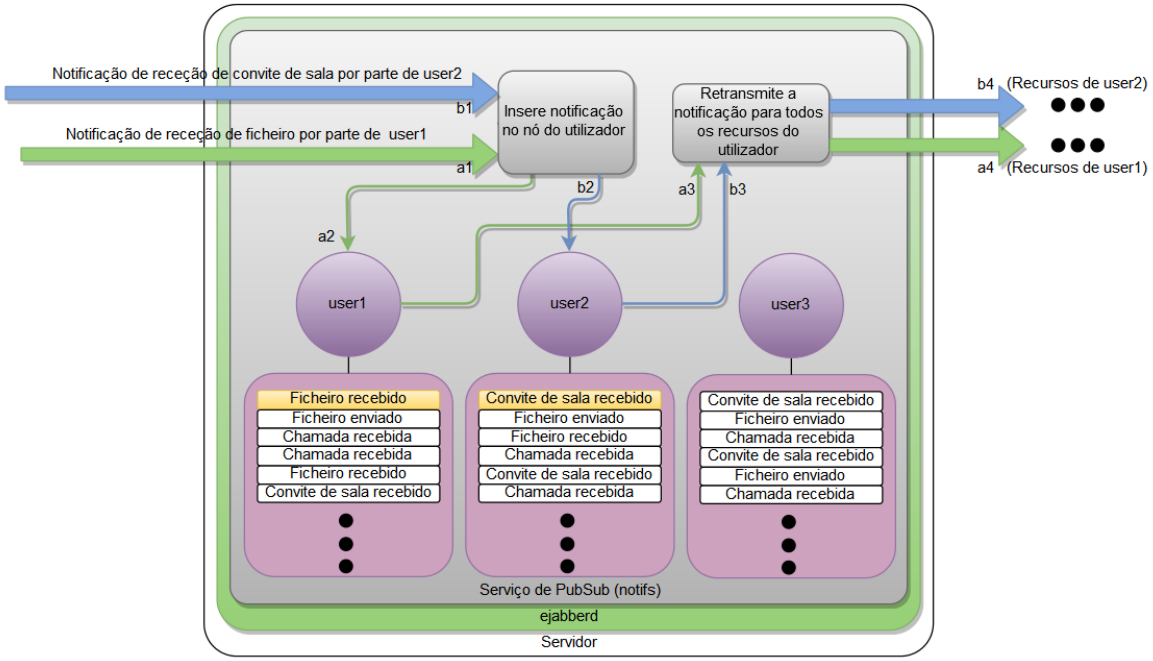


Figura 30 Publish-Subscribe.

Como exemplo, user1 recebe um pedido de transferência de um segundo utilizador. Neste caso a1) será emitida uma notificação para o nó user1, a2) sendo a mesma armazenada. De seguida a3) a4) a notificação é retransmitida para todos os recursos de user1. Assim todos os dispositivos de user1 recebem a notificação de transferência. Num outro exemplo, quando user2 recebe um convite para se juntar a uma sala de conversação, b1) é enviada uma notificação para o nó user2 do serviço de *Publish-Subcribe* do servidor XMPP, a indicar a receção do convite. O serviço ao recebe a notificação, b2) armazena a mesma no nó user2. Após o armazenamento ser efetuado, o serviço de *pubsub* b3) b4) retransmite a notificação a todos os recursos registados no nó (recursos de user2).

Este mecanismo é integrado para que na interface de notificações, surjam as mensagens relativas a cada evento ocorrido, com os detalhes necessários. Desta forma é criado um sistema de notificações que concentra todos os eventos numa única estrutura, e que ao mesmo tempo permite filtrar os resultados por tipo de serviços.

Neste momento o cliente no projeto suporta autenticação, presença do utilizador, gestão de contactos, recursos, salas e grupos, utilização dos serviços de comunicação disponíveis e um sistema centralizado de notificações. Assim termina a fase de desenvolvimento.

5.6. BOT XMPP

Com todas as interfaces e funcionalidades de comunicação já implementadas no cliente, a próxima fase passa por integrar os serviços de operação de ficheiros e multimédia. Para isso é necessária a implementação de uma aplicação que esteja alojada no servidor e seja responsável por gerir e prestar estes serviços, pois esta terá acesso aos recursos locais, sistema de ficheiros bem como aplicações do servidor. Com estes fatores em conta são consideradas duas hipóteses.

- Módulo ejabberd:

Nesta hipótese é considerado o desenvolvimento de um módulo que seria executado no servidor ejabberd, e prestaria suporte aos serviços pretendidos. Utiliza a possibilidade oferecida pelo ejabberd, de permitir o desenvolvimento de componentes personalizados em Erlang ou Elixir [121].

- Cliente autónomo:

Um cliente XMPP autónomo alojado no servidor e que a partir do protocolo XMPP permitisse comunicar com o servidor ejabberd e com os utilizadores e recursos do sistema, tendo ainda acesso aos recursos locais do servidor.

Feita a análise das duas alternativas, a escolha recai sobre o cliente autónomo. O problema de desenvolver módulos no servidor ejabberd, é o facto de não só ser necessário a utilização de uma nova linguagem (*Erlang* ou *Elixir*), mas também o projeto ficar dependente do *ejabberd*. O cliente autónomo poderia ser desenvolvido em *C++* e utilizar *Qt* como base para o seu desenvolvimento. Pode até ser utilizado o cliente já desenvolvido até ao momento, adaptar este para Linux e modificar o mesmo para executar as operações necessárias.

Fica então decidido que o módulo alojado no servidor será um *bot XMPP*, cliente autónomo que utiliza o servidor XMPP como mecanismo de comunicação para interagir com os recursos XMPP ligados ao servidor e prestar os serviços disponíveis.

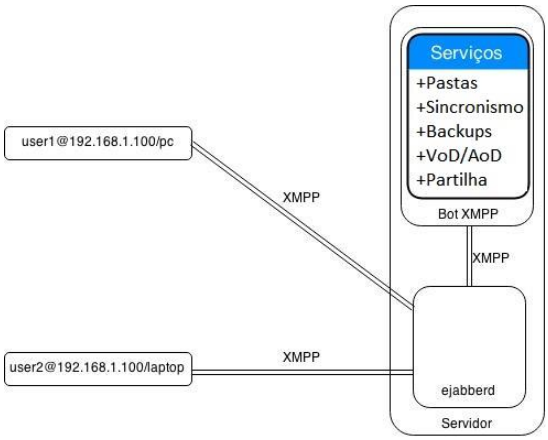


Figura 31 Interação entre o Bot e os restantes elementos do sistema.

É então efetuada a adaptação do cliente já desenvolvido em Windows para Linux. Importante notar que as funcionalidades do cliente (presença, listas, mensagens, transferências, interação com salas, chamadas e notificações) oferecem a base para os serviços pretendidos, como a utilização de transferências de ficheiros na funcionalidade de sincronização de pastas, e podem até ser uteis para funcionalidade futuras.

5.7. SERVIÇOS CENTRALIZADOS

Os serviços de operação de ficheiros e multimédia considerados são:

- Pastas Remotas: Possibilitar ao utilizador criar pastas remotas e transferir ficheiros para estas de forma a poder aceder aos ficheiros remotamente.

- Pastas Sincronizadas: Criar ou configurar pastas já existentes, para que estas sejam automaticamente atualizadas entre os dispositivos do utilizador e o servidor.

- *Backups*: Serviço que efetua para uma pasta remota, cópias de uma segunda pasta, que pode ser local ou remota. Estas cópias podem ser configuradas para ocorrerem com um período que pode ser configurado.

- Serviço de Multimédia: Possibilitar ao utilizador efetuar *AoD (Audio on Demand)* de conteúdo de áudio e VoD (Video on Demand) de vídeo do servidor para o cliente.

- Partilha de Pastas: Permite ao utilizador partilhar pastas com contactos e aceder a partilhas dos mesmos.

De forma a concentrar todos os serviços, é considerada a utilização de uma janela comum, onde estarão separadores para os respetivos serviços. A partir do botão de serviços na interface principal, é criado o mecanismo que mostra a janela de serviços. Nas próximas secções são descritos os serviços apresentados anteriormente e explicada a sua implementação.

5.7.1. SISTEMA DE PASTAS

De forma a facilitar o acesso aos diferentes tipos de pastas, é tomada a decisão de se concentrar numa única interface, todos os serviços de pastas. Significa o separador de pasta serão apresentadas todas as pastas base de cada tipo de pasta do sistema.

Figura 32 Interface de Pastas.

Esta interface separa as pastas de cada tipo de serviço em quatro divisões distintas. Cada uma corresponde a um tipo de pasta, sendo estes pasta remota, pasta sincronizada, pasta de *backup* e pasta de partilha. Para criar uma pasta esta pode ser do tipo que o utilizador pretender, selecionado a opção pretendida na barra de opções. Ao criar uma pasta nesta interface, esta será alojada na lista respetiva, associada ao tipo de pasta criada.

A estrutura de pastas do servidor está dividida nos vários tipos de pastas do sistema, sendo estes "*Pastas*", "*Sincs*", "*Backups*" e "*Shares*". Dentro de cada uma destas pastas, existe uma pasta por cada utilizador, como por exemplo "user1" e "user2", onde estão localizadas as pastas daquele tipo referente ao utilizador.

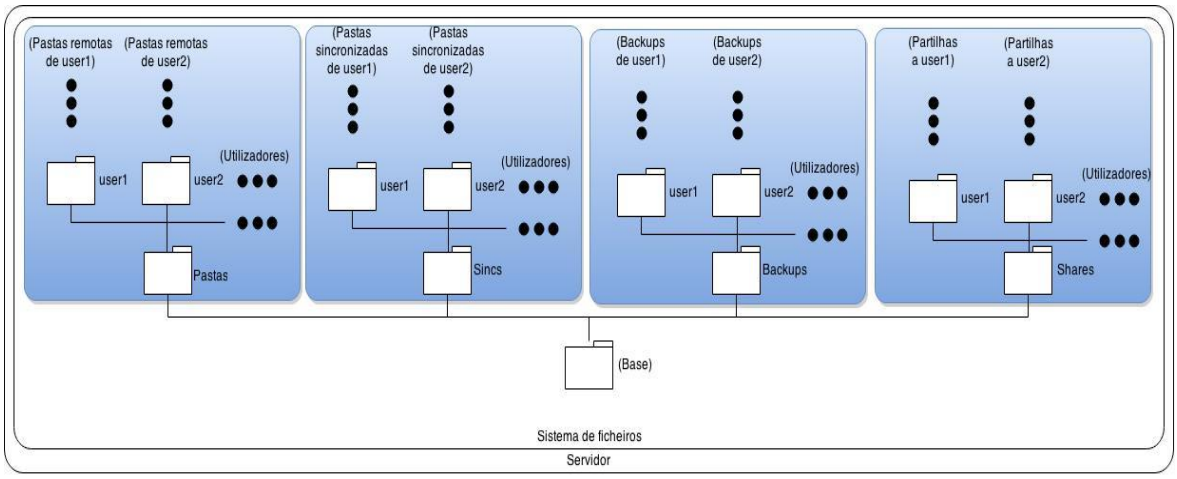


Figura 33 Sistema de pastas no servidor.

Como exemplo, a pasta "user1" localizada dentro da pasta "Sincs", contém todas as pastas sincronizadas do utilizador user1. Já a pasta "user2" em "Shares", contém todas as pastas e ficheiros partilhados por user2.

Quando o cliente está na raiz da interface de pasta, onde estão visíveis os quatro tipos de pastas, pede ao servidor o conteúdo de todas as pastas raiz referentes ao utilizador. Esse conteúdo é mostrado na interface devidamente ordenado.

5.7.2. PASTAS REMOTAS

Na figura 34, encontra-se o processo que retorna o conteúdo de uma pasta no servidor.

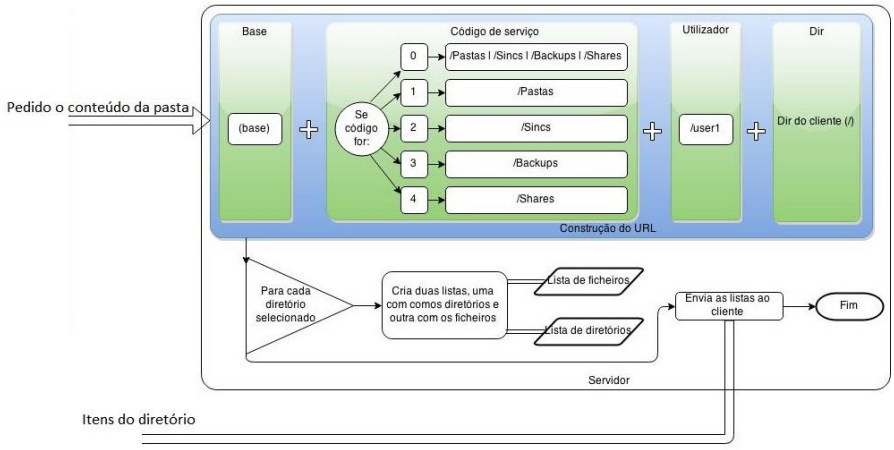


Figura 34 Ilustração do pedido do conteúdo de uma pasta.

Ao clicar numa pasta, será enviado um pedido ao *bot* por meio de um IQ personalizado com o tipo de serviço e a pasta alvo, a requerer o conteúdo da pasta selecionada. O tipo de serviço é especificado a partir de um código. Este é um número que representa o tipo de pasta que está a ser requisitada. Quando o *bot* deteta o pedido, vai a partir dos dados inseridos no IQ, definir o URL da pasta que o utilizador pretende. O URL é construído da seguinte forma. Começa pela base, que é o diretório do servidor, onde estão as pastas "Pastas", "Sincs", "Backups" e "Shares". De seguida verifica o código de serviço recebido. Se for 1 significa que a pasta requisitada referente-se à pasta "*Pastas*". Se for 2, 3 e 4 considera a pasta "*Sincs*", "*Backups*" ou "*Shares*" respetivamente. No caso de o código ser zero, significa que o utilizador se encontra na base do serviço de pastas, requisitando assim todas as subpastas. De seguida é selecionada a pasta relativa ao utilizador que enviou o pedido. Se o utilizador estiver na base, onde vê todas as pastas dos serviços, o diretório atual é a raiz (/). No caso de clicar numa subpasta remota (Ex.: Pasta1), o diretório passa a ser o dessa pasta (Ex.: /Pasta1). Após determinar todos estes componentes, o *bot* adiciona todos num único elemento, dando origem ao URL da pasta requisitada, sendo neste caso (/(base)/Pastas/user1/Pasta1/). De seguida recolhe as pastas e os ficheiros da mesma para duas listas e envia as mesmas num IQ personalizado, como resposta ao pedido do cliente.

Dentro das pastas remotas, pastas sincronizadas e pastas partilhadas, podem ser criadas novas pastas e serem carregados ficheiros. Estes últimos pode ser posteriormente descarregados clicando nos próprios ou eliminados, clicando na opção disponível. Quer o carregamento (*upload)* quer o descarregamento (*download)* destes ficheiros, utilizam a funcionalidade de transferência de ficheiros descrita na secção 5.1.4 (***Transferência de Ficheiros***) como base para os processos. Finalizando, todas as pastas são pastas remotas, oferecendo assim aos restantes tipos de pastas, todas as funcionalidades e características de uma pasta remota.

5.7.3. PASTAS SINCRONIZADAS

Para sincronizar pastas é necessário dois fatores. Primeiro, tem sempre de existir uma pasta remota e pelo menos uma pasta local num recurso do utilizador. Segundo, o mecanismo de sincronismo tem de saber as diferenças entre as duas pastas, e caso existam, qual o conteúdo de cada pasta é atualizado.

Quando um utilizador cria uma pasta sincronizada e a conecta a uma pasta local, estas ficam sincronizadas entre si. Sempre que o cliente se autentica, deteta que uma das pastas locais sincronizadas foi modificada ou recebe um pedido de sincronismo, é executado o procedimento de sincronismo. O cliente começa por recolher o conteúdo da pasta a sincronizar e efetuar um mapeamento desse conteúdo. Este mapeamento cria uma estrutura com todos os diretórios, subdiretórios e todos os ficheiros da pasta. Concluído esse processo, é enviada ao *bot* um pedido de sincronismo, a partir da extensão personalizada Sync, que envia um IQ no qual é incluída a estrutura da pasta e o URL da pasta remota. Quando o *bot* recebe o pedido de sincronismo, extrai o seu conteúdo e cria um registo do mesmo.

Enquanto o mapeamento é utilizado para estruturar o conteúdo da pasta no IQ, o registo é utilizado para guardar o conteúdo para posterior comparação e armazenamento.

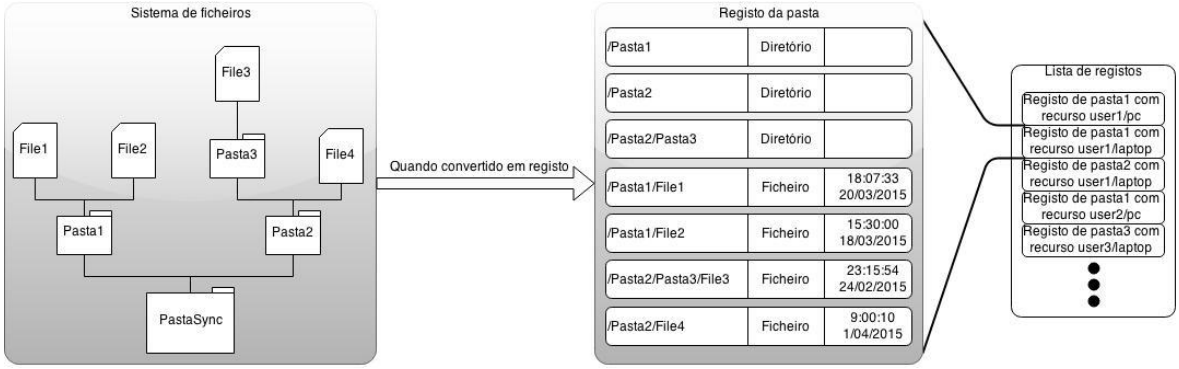


Figura 35 Conversão do conteúdo de uma pasta para um registo.

Cada registo é referente a um recurso de um utilizador. Significa que para um utilizador com o recurso "pc" e o recurso "laptop", o servidor vai gerar um registo para cada recurso. Estes registos também são gerados para cada uma das pastas do servidor, sempre que estas forem sincronizadas.

Se for a primeira vez que a pasta é sincronizada, são iniciadas as transferências dos ficheiros da pasta sincronizada para a nova pasta e armazenado o registo da pasta. Também é criado um segundo registo para a pasta no servidor. Assim o servidor tem os registos quer da pasta do cliente da última atualização, quer o registo da pasta no servidor, da última atualização. Se a pasta já foi sincronizada anteriormente, é feita uma comparação entre os registos atualizados das duas pastas envolvidas e os registos armazenados no servidor, referentes à última sincronização efetuada das pastas. Este bloco permite a partir do estado atual (registos atuais) e do estado desde o último sincronismo (registos armazenado), detetar a eliminação de ficheiros e diretórios e a criação ou edição de ficheiros nas duas pastas. Da comparação resultam três listas de componentes. Estas contêm os diretórios a eliminar na pasta do cliente e do servidor, a lista de *uploads* que representa os ficheiros que o cliente vai enviar ao servidor e a lista de *downloads* que representa os ficheiros que o servidor vai enviar ao cliente. O bot envia estas listas no IQ de resposta ao cliente, em simultâneo com os pedidos de transferência ao cliente, relativos aos *downloads*. Quando o cliente recebe o IQ, vai recolher os diretórios a eliminar e remove os mesmos da pasta. De seguida recolhe os *uploads* e envia os pedidos de transferência dos ficheiros dessa lista para o *bot e aceita as transferências relativas aos downloads*, enviadas pelo *bot* ao mesmo tempo que o IQ. A interação entre o cliente e o *bot* é feita inicialmente por mensagens XMPP. Esta troca de mensagens permite a identificação de diferenças entre as pastas. Para cada caso são avaliados quais os ficheiros de cada pasta são atualizados. As transferências são aceites por ambas as partes automaticamente para os URLs de destino dos vários ficheiros. No final, após as transferência serem concluídas, ambas as pastas terão o mesmo conteúdo.

5.7.4. *BACKUPS*

O sistema de *backups* tem como objetivo efetuar cópias de uma pasta local ou remota, com intervalos periódicos entre as cópias, e armazenar estas numa única pasta.

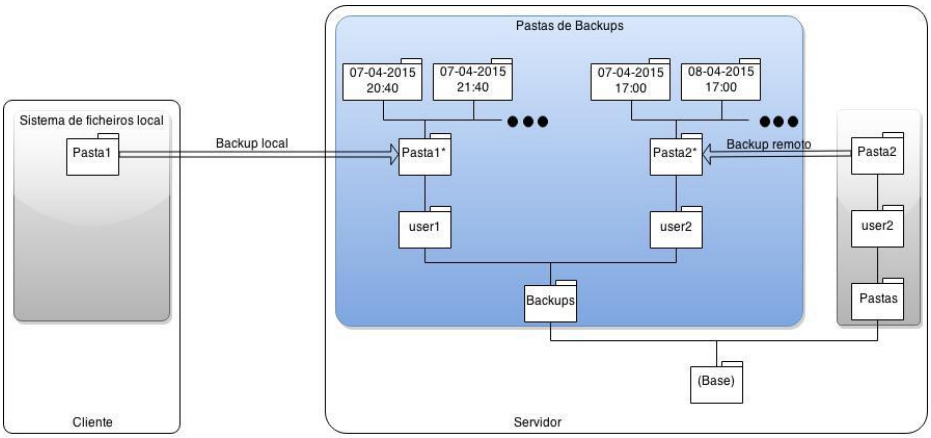


Figura 36 Sistema de *Backups*.

Quando é efetuada a criação de uma pasta do tipo *backup*, esta é emparelhada com uma pasta local ou outra pasta remota e configurado o intervalo de tempo. Após feita a criação e emparelhamento da pasta *backup* o processo que efetua as cópias é automático. Na figura 37 é apresentado o processo que ocorre no momento em que o temporizado de uma pasta de *backup* chega ao fim do intervalo de tempo.

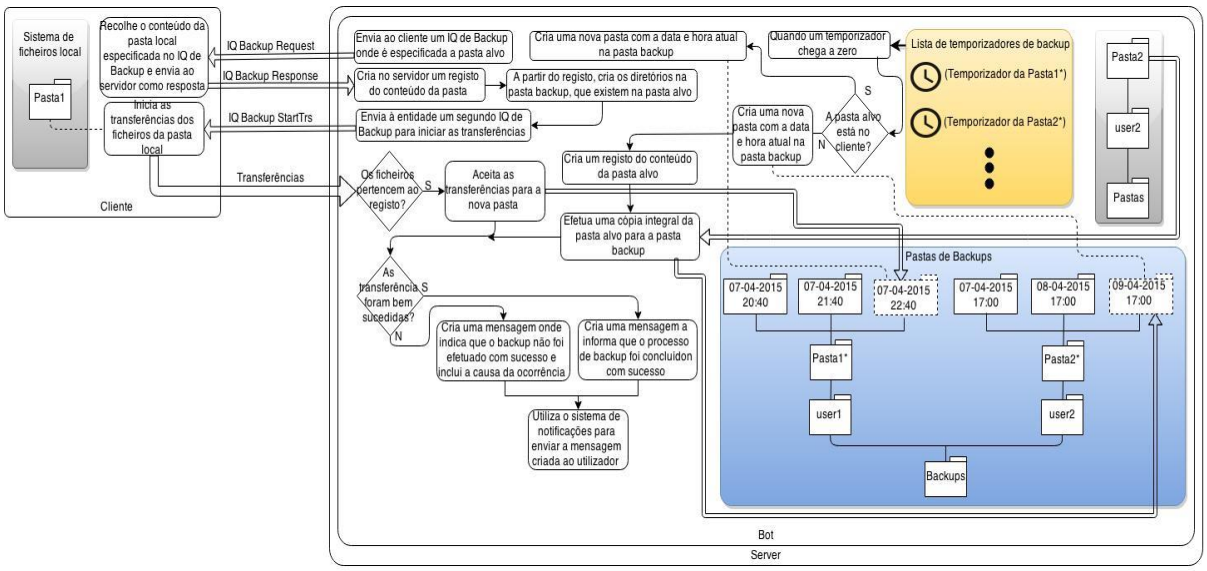


Figura 37 Processo de um *Backup*.

Quando o timer de uma pasta de *backup* atinge o intervalo configurado, é iniciado o processo de backup. No caso da pasta original se localizar no servidor, cria uma nova pasta com a data e hora atualizada e copia o conteúdo da pasta alvo, para a nova pasta. Se a pasta alvo estiver num recurso, cria uma nova pasta com a data e hora, na pasta de backup e é enviado ao recurso um IQ tipo *Backup* onde é requisitado o conteúdo da pasta alvo. O cliente envia então os pedidos de transferência para o *bot*, que as vai aceitar automaticamente. Se as transferências forem bem sucedidas o cliente responde com um IQ a confirmar as transferências. O nome das cópias nas pastas de *backup* contêm a data e a hora, sendo automaticamente ordenadas da mais recente para a mais antiga.

5.7.5. SERVIÇO MULTIMÉDIA

O serviço de multimédia pressupõe a possibilidade do servidor emitir um ficheiro multimédia aos clientes, permitindo a estes a sua execução imediata, antes de a transferência ser concluída. Com isto em mente foram consideradas duas vertentes.

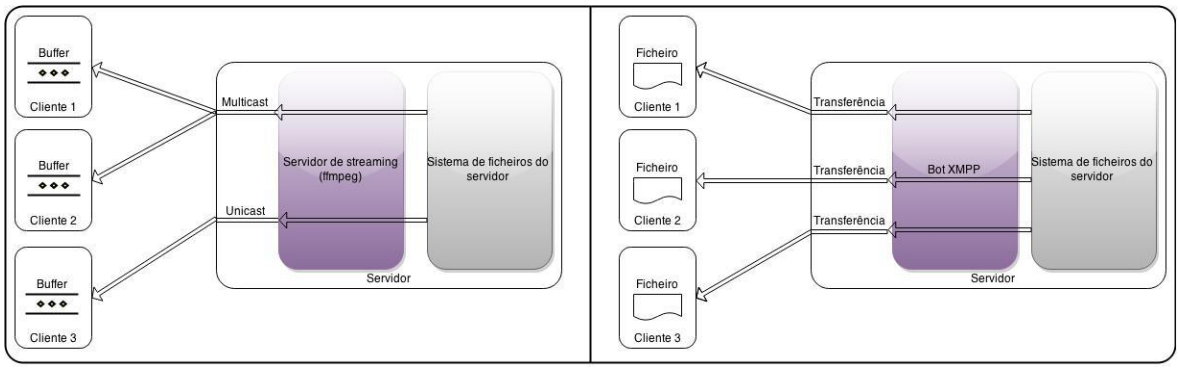


Figura 38 A) *Streaming*. B) *Download* Progressivo.

A primeira abordagem utiliza um servidor de *streaming* para transmitir e controlar uma sessão multimédia, por meio de protocolos para o efeito como RTP, RTSP ou MMS. Utilizando este método o servidor envia um fluxo de multimédia ao cliente. Este tem um *buffer*, estrutura que armazenar temporariamente na memória, o conteúdo que vai sendo emitido, ao mesmo tempo que vai sendo executado. Tem a vantagem de suportar quer *unicast* (emissão para um cliente), ou *multicast* (o mesmo fluxo enviado para vários clientes).

Já o *download* progressivo utiliza uma abordagem distinta da anterior. Na prática quando o cliente pretende consumir um ficheiro multimédia remoto, é efetuado o *download* do ficheiro pretendido, do servidor para o cliente e o ficheiro começa a ser executado no cliente durante a transferência.

O primeiro é mais indicado para *live-streaming* e *streaming* sincronizado, onde vários utilizadores podem ver o mesmo fluxo em simultâneo, pois como possibilita *multicasting*, só é necessário processar um fluxo para todos os utilizadores. O segundo é mais indicado para fluxos individuais, onde cada utilizador visualiza o seu conteúdo de forma paralela e independente dos restantes. Para além destes fatores, sempre que se utiliza um protocolo de transporte como RTP, é necessário configura *payloads* e *bitrates* para cada formato utilizado. Já no caso de um *download*, qualquer ficheiro pode ser descarregado, desde que suporte a sua leitura de forma progressiva e não seja ilegível caso não o ficheiro não esteja completo. Assim para o caso deste sistema, é aplicado o método de *download* progressivo para o serviço de VoD e AoD do servidor.

Quando um utilizador clica na opção adicionar à playlist de um ficheiro, é executado o processo descrito na figura 39.

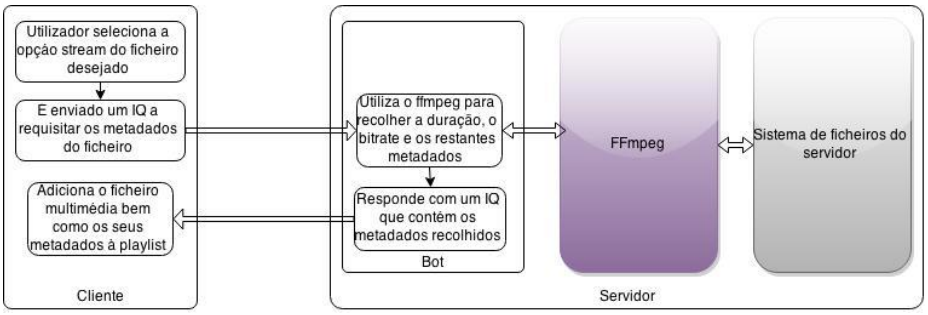


Figura 39 Processo de streaming.

O utilizador pode consumir qualquer ficheiro multimédia ao qual tem acesso e esteja localizado no sistema de pastas, selecionando a opção respetiva no ficheiro desejado. O processo começa por requisitar ao servidor os meta dados, como o *bitrate* e a duração do ficheiro, a partir do FFmpeg. A título opcional pode recolher títulos, álbuns e artistas do vídeo ou música. Após o cliente receber estes detalhes, o ficheiro é adicionado à *playlist*

local. O utilizador pode continuar a selecionar os ficheiros que pretende e posteriormente dirigir-se à interface de multimédia onde pode interagir com a *playlist*.

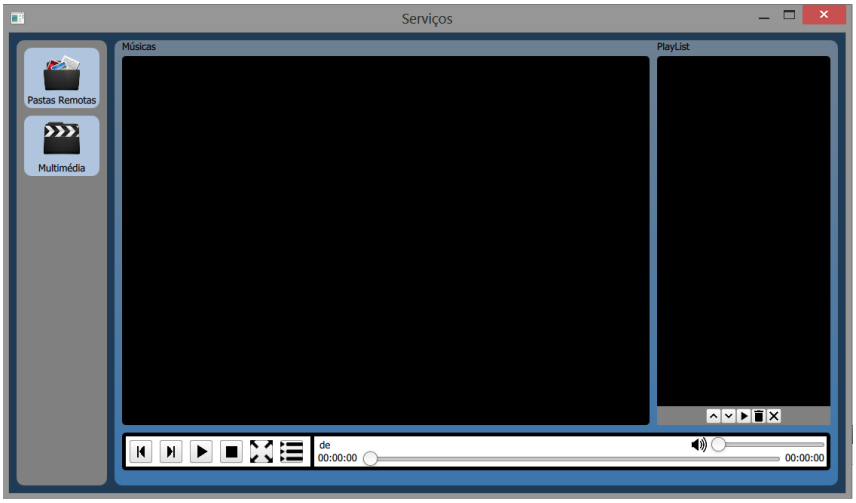


Figura 40 Interface de multimédia.

A interface é constituída pelas seguintes partes:

- Janela de vídeo: Janela na qual será mostrado o conteúdo visual do ficheiro, no caso de ser um vídeo ou filme. Tem a opção de expandir para *fullscreen* no botão existente e voltar atrás a partir da tecla 'Esc'.

- Playlist: Contém a lista de ficheiros que foram selecionados anteriormente. Estes podem ser executados, eliminados, ou movimentados na lista. Pode ainda ser limpa toda a lista na opção respetiva.

- Painel de controlo: Apresenta o nome e a duração do ficheiro em execução, permitindo controlar o ficheiro atual a partir das opções pausar, parar, continuar e passar ao ficheiro seguinte ou anterior. Permite ainda visualizar a posição atual a partir da barra de estado e utilizar esta para se movimentar na posição de execução do ficheiro.

A partir desta interface é então possível controlar a playlist, iniciar a execução de conteúdo multimédia e controlar o mesmo. Quando a opção de executar um ficheiro é clicada, o ficheiro selecionado na *playlist* passa a ser o conteúdo ativo. Na figura 41 é ilustrado o processo que ocorre ao selecionar essa opção.

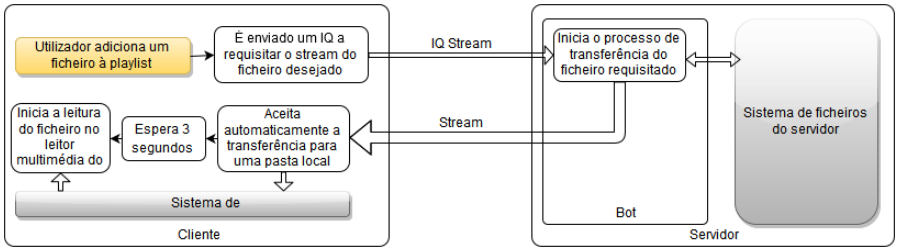


Figura 41 Execução de um ficheiro multimédia.

Começa por enviar um IQ ao servidor, a requisitar a transferência do ficheiro multimédia. O servidor ao receber o IQ inicia a transferência do ficheiro pretendido. A transferência após ser iniciada continua até ser completamente transferido, o ficheiro ser eliminado da playlist ou outro ficheiro ser executado. Após receber o IQ de resposta a confirmar que a transferência está a decorrer, o cliente inicia a leitura do ficheiro utilizando a biblioteca *QMediaPlayer* do Qt. Se durante a sessão atual, o utilizador colocar a posição de execução numa determinada posição, a transferência do ficheiro pode ainda não ter descarregado essa parte. Neste caso é efetuado um novo pedido do mesmo ficheiro, mas a começar nessa posição.

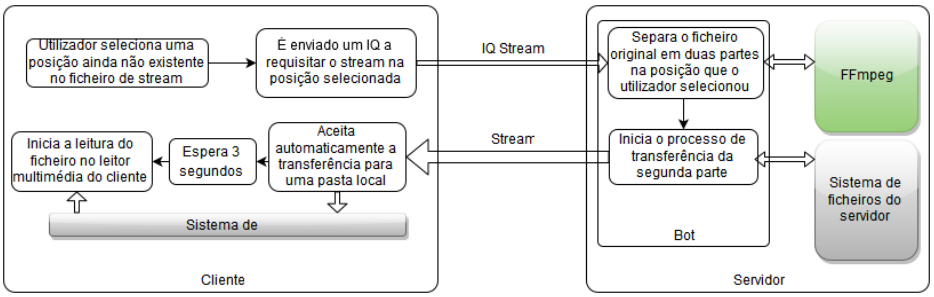


Figura 42 Ilustração da retransmissão de um ficheiro multimédia numa posição não carregada.

Neste caso o pedido é feito com a posição desejada. Quando o *bot* recebe o pedido, requere ao FFmpeg para partir o ficheiro de multimédia nessa posição. De seguida, a segunda parte resultante da separação é transferida, e o cliente executa o mesmo, mantendo as características do ficheiro inicial (tamanho, meta dados e *bitrate*), mas a começar no ponto que o cliente requisitou.

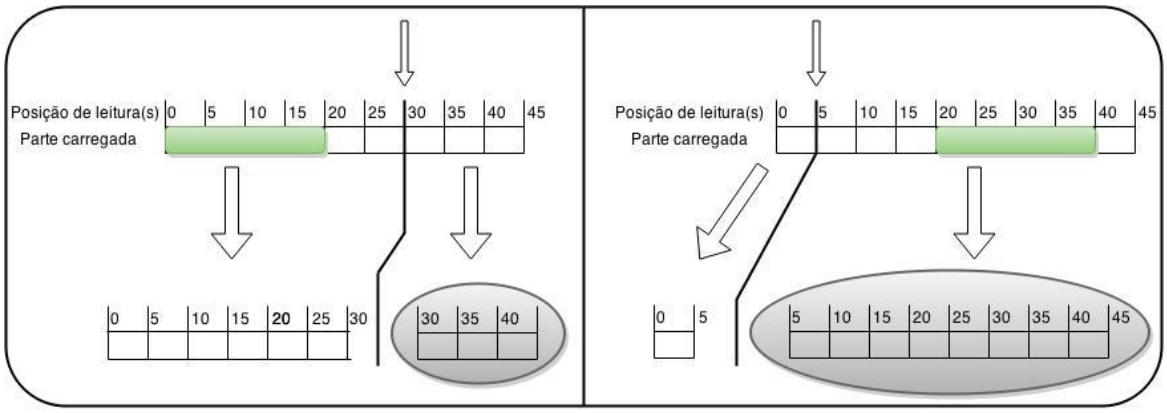


Figura 43 Ilustração da separação de um ficheiro multimédia em duas partes. A) Pede posição de 30s mas o ficheiro só chegou aos 20s. b) Deseja posição de 5s mas o ficheiro tem a partir de 20s.

Este processo acontece quando o ficheiro ainda não chegou à posição selecionada ou no caso do ficheiro atual começar posteriormente à posição desejada.

5.7.6. SISTEMA DE PARTILHA DE PASTAS

Por fim é altura de criar o mecanismo que permite a utilizadores, partilhar pastas com os seus contactos. Quando a opção de partilha de uma pasta é clicada, é iniciado o processo descrito na figura 44.

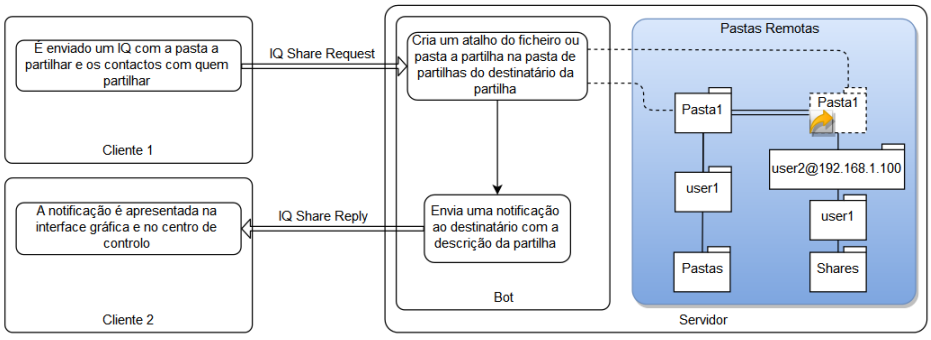


Figura 44 Processo de partilha.

Quando um utilizador (Ex.:user1) partilha uma pasta ou um ficheiro com um segundo utilizador (Ex.:user2), se já não existir, é criada uma subpasta em user1. Esta terá o jid do recetor da partilha (Ex.:/user1/user2@192.168.1.100). Dentro desta subpasta é criado um atalho para a pasta original.

Quando um utilizador se encontra na raiz e o cliente requere as pastas partilhadas, ocorre o processo descrito na figura 45.

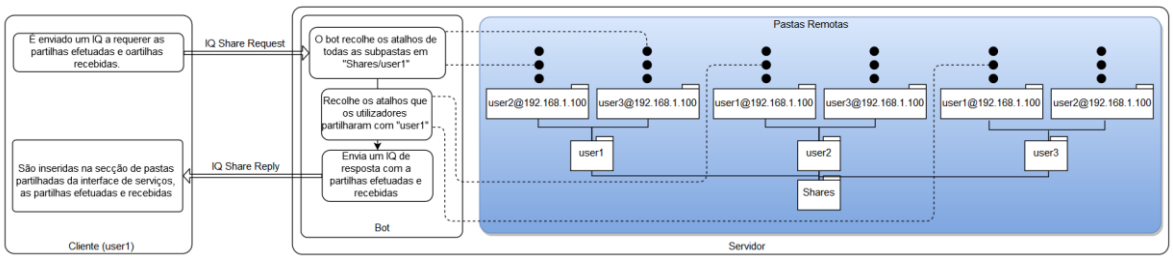


Figura 45 Recolha de partilhas para a interface de serviços.

O cliente efetua um pedido ao *bot*, sobre as pastas que o utilizador (Ex.:user1) partilha e pastas partilhadas com o mesmo. O *bot* acede à pasta do utilizador (Ex.:/Shares/user1/) e recolhe os atalhos de todas as subpastas, referentes aos seus contactos. Estes atalhos referem-se às partilhas que o utilizador efetuou com outros utilizadores. De seguida o *bot* vai às pastas de todos os utilizadores, e recolhe os atalhos de todas as pastas referentes ao utilizador original (Ex.: /Shares/user2/user1@192.168.1.100 e /Shares/user3/user1@192.168.1.100). Estes atalhos representam as partilhas feitas por outros utilizadores. Após efetuados os processos anteriores é enviado ao cliente as suas partilhas feitas pelo próprio e feitas pelos restantes utilizadores.

5.8. SISTEMA DE ADMINISTRAÇÃO

Com as funcionalidades já desenvolvidas, é necessário ainda implementar um sistema de administração que permita a um utilizador configurar os vários aspetos do servidor. Daí surge o conceito de administrador. Neste sistema existirão dois tipos de intervenientes. Os utilizadores que são os consumidores que se conectam e consomem os serviços prestados pelo servidor e o administrador que constitui um utilizador com acesso privilegiado às configurações do servidor. Para aceder às configurações, é necessário proceder à autenticação com o *username* e a *password* de administrador na interface de autenticação.

O bot XMPP será o responsável por receber os comandos do administrador e interagir com o servidor XMPP ejabberd e com os recursos locais do servidor, para suportar o sistema de administração na configuração do sistema. A interface de administração no dispositivo cliente deteta as opções inseridas ou escolhidas pelo administrador.

5.8.1. REDE

Nesta área é possível configurar as definições de rede do servidor, escolhendo se esta configuração é feita automaticamente por DHCP (*Dynamic Host Control Protocol*), ou se o administrador configura manualmente a partir de IP estático. Neste caso os campos de IP, máscara de rede e *gateway* são editáveis.

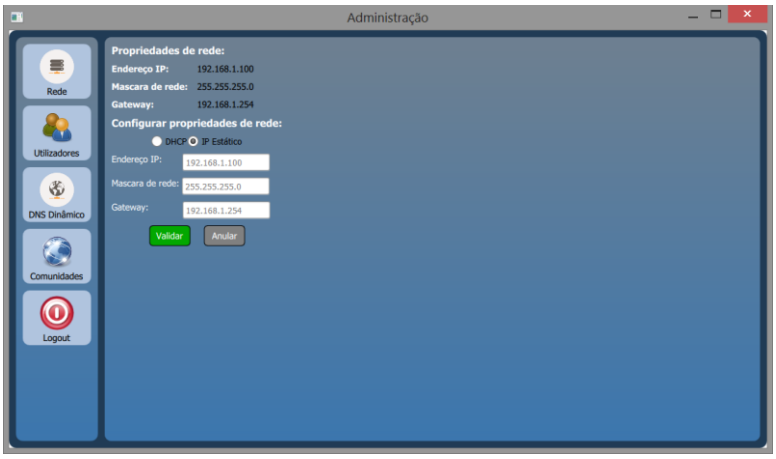


Figura 46 Painel de administração de definições de rede.

Se o utilizador não pretender a configuração que inseriu ou deseja abortar a operação, clica no botão "Anular", que reverte as alterações efetuadas. Se desejar confirmar a configuração inserida, clica no botão "Confirmar". Nesse momento o é enviado um IQ ao *bot* com o tipo de configuração (DHCP ou estático), IP, máscara de rede e *gateway*. O *bot* recolhe os mesmos e edita o ficheiro de configuração do *Ubuntu* que contém a configuração da interface de rede com as novas definições de rede e reinicia a interface de rede.

5.8.2. UTILIZADORES

O separador utilizadores oferece a possibilidade de criar novas contas de utilizador, editar ou remover contas já existentes e visualizar a lista de utilizadores e seus recursos ativos.

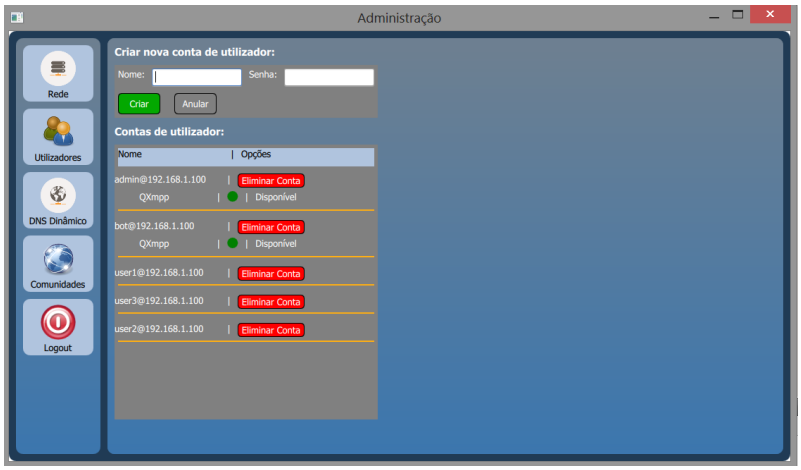


Figura 47 Interface de administração de utilizadores.

Quando um administrador cria uma nova conta, insere o nome de utilizador e a senha da nova conta. Quando estes dados são validados é enviado um IQ com os dois parâmetros que foram inseridos. O *bot* ao receber os dados vai recolher os mesmos e vai adicionar o novo utilizador ao sistema. Neste processo a nova conta de utilizador é criada no servidor XMPP, é adicionado ao seu *roster* de contactos os utilizadores do servidor e o novo utilizador é adicionado nos *rosters* dos restantes contactos. Por fim é criado o nó do serviço *de Publish-Subscribe* referente ao utilizador, este regista-se no nó como *editor* e o *bot* regista-se no nó como *publish-only*.

5.8.3. SERVIÇO DE DNS

No caso do acesso ao servidor ser feito pela internet este terá de possuir um IP público e ter de configurar o *router* doméstico para redirecionar um porto do router para o servidor. O problema do IP público está no facto de ser atribuído dinamicamente pelo operador de internet, levando a que periodicamente o IP público varie. Assim é necessário implementar um mecanismo que permita resolver este problema e ao mesmo tempo facilitar a memorização do endereço público do servidor. A solução está em utilizar um serviço de *Dynamic DNS*. Estes serviços baseados no protocolo DNS *(Dynamic Name System*)

permitem adquirir um domínio público (Ex.: myserver.com) e a partir de um *software* cliente, o serviço deteta e atualiza automaticamente o IP público da rede local. Este separador permite o registo por parte do servidor, num serviço de DNS dinâmico. O *ddcliente* é o *software* cliente implementado no servidor que permite a utilização de serviços de DNS dinâmico para facilitar o acesso ao servidor, a partir da internet.

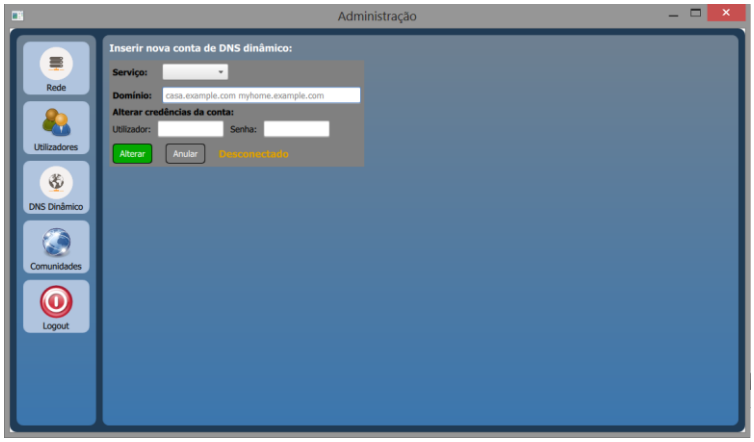


Figura 48 Interface de administração de DNS dinâmico.

Escolhendo o serviço pretendido e inserindo o *username*, a *password* e o domínio, estes dados são enviados ao *bot* que configura o ficheiro de configuração do *ddclient* e ativa o serviço.

5.8.4. COMUNIDADES VIRTUAIS

As comunidades virtuais possibilitam a que vários servidores distintos se possam conectar entre si, e permitir aos seus utilizadores comunicarem entre si. Este mecanismo é ativado, inserindo o IP do servidor com o qual se pretende estabelecer a ligação e um identificador que permite ao utilizador atribuir um nome ao servidor remoto. Se o servidor recetor desejar estabelecer a conexão, o seu administrador efetua o mesmo passo anteriormente descrito.

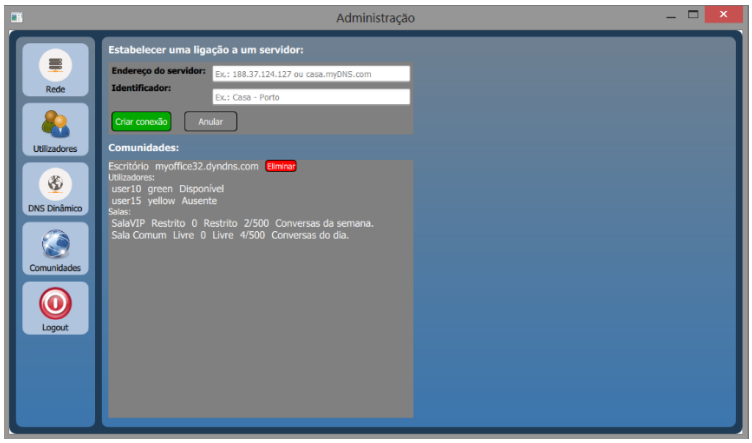


Figura 49 Interface de administração de comunidades.

Quando um servidor envia o pedido a um servidor remoto, este pedido vai ser recebido e armazenado. Se o servidor remoto enviar um pedido para o mesmo servidor, vai configurar a *whitelist* do *ejabberd* para aceitar comunicações com o servidor remoto. Quando ambos os servidores efetuam este passo, ambos ficam a poder comunicar entre si.

Após as *whitelists* serem configuradas é ainda necessário que cada um dos *bots* referentes aos servidores em questão, adicionem nas suas listas de contactos e nas listas de contactos dos seus utilizadores, os utilizadores do servidor remoto.

Assim fica concluído o processo de estabelecimento de uma ligação entre dois servidores. Por omissão o *ejabberd* está configurado para rejeitar qualquer ligação de um servidor externo. Só os servidores na *whitelist* podem comunicar com o mesmo. Com este mecanismo os utilizadores passam a ter como contactos utilizadores de outros servidores e podem interagir com estes, formar grupos e até aceder a conversações e chamadas a decorrer no servidor remoto.

6. TESTES

Após terminado o desenvolvimento e a implementação do cliente e do servidor, é necessário verificar se as propriedades e serviços implementados não só funcionam corretamente, mas também se têm uma performance aceitável nos casos onde esta é relevante. De seguida é realizado um conjunto de testes que demostram diversos casos onde é possível comprovar-se o funcionamento dos mecanismos implementados.

De forma a tentar simular uma situação realista foi utilizado um laboratório que contém os computadores necessários para criar a arquitetura de testes realista.

6.1. ARQUITETURAS DE TESTE

As arquiteturas para os testes podem ser observadas na figura 50.

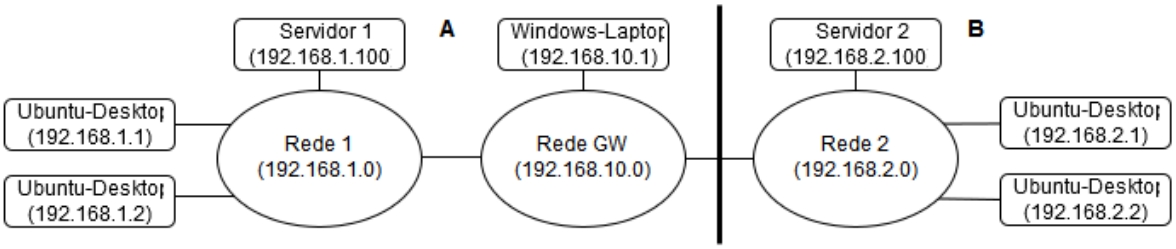


Figura 50 Sistema de Listas e Seleção.

A rede (192.168.1.0) representa um domínio doméstico, como uma habitação, escritório ou edifício. Nesta rede está presente o servidor da rede (192.168.1.100) e dois clientes (192.168.1.1) e (192.168.1.2) associados às contas de utilizador (user12) e (user13) respetivamente.

Para além desta rede, existe uma segunda rede (192.168.10.0). Esta tem como objetivo representar a internet durante os testes. Aqui está o *gateway* que redireciona os pacotes de dados entre as várias redes e permitir a estas comunicar entre si. Nesta rede será ainda conectado o portátil referido anteriormente. Sendo que esta rede simula a Internet, ao associar o portátil (192.168.10.1), vai criar-se a situação de um utilizador remoto que se pretende conectar ao servidor local a partir da Internet e consumir os seus serviços remotamente. Em termos de contas de utilizador, aos computadores da rede local são atribuídos os utilizadores (user12@192.168.1.100) e (user13@192.168.1.100). Ao portátil será atribuído o utilizador (user11@192.168.1.100).

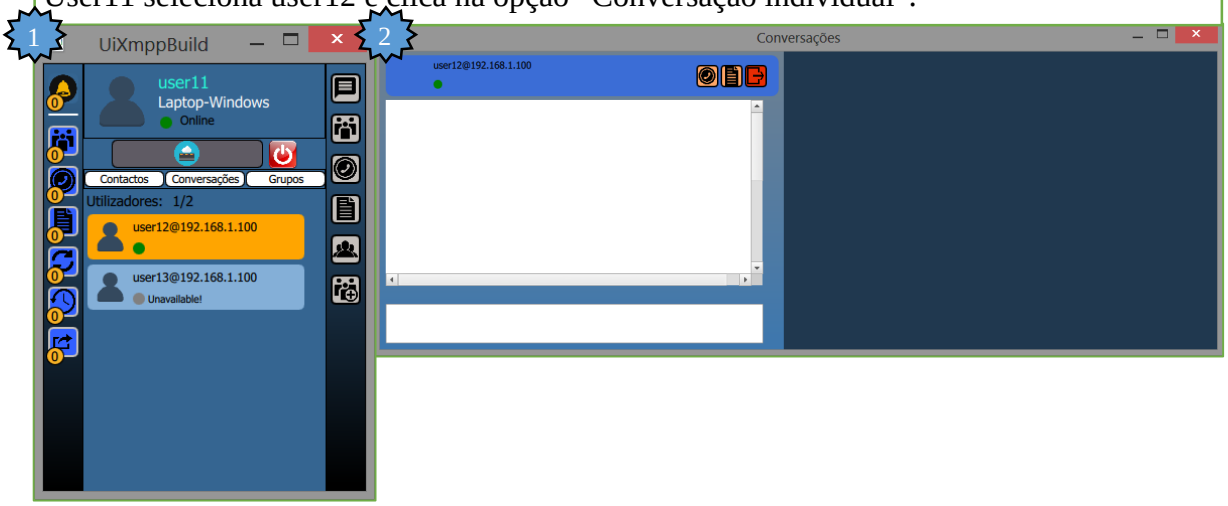
Os testes são divididos em dois tipos. Os testes 1 a 9 são efetuados considerando apenas um domínio. O teste 10 utiliza dois domínios sob a forma da segunda arquitetura apresentada anteriormente na figura 50. Esta segunda configuração serve para testar as funcionalidades numa situação onde existe uma comunidade de servidores. A rede do segundo servidor será (192.168.2.0), o servidor será (192.168.2.100) e os clientes (192.168.2.1) para user21 e (192.168.2.2) para user22.

Por fim os testes são por norma captações de ecrã tiradas à medida que se vão realizando os testes. Estas têm por cima a ação que é executada e tem ainda um número no canto superior esquerdo. Este valor corresponde à ordem pela qual ocorrem os eventos. Por vezes várias captações podem ter o mesmo número. Nesses casos, significa que os eventos ocorrem ao mesmo tempo. User11 é representado por verde, user12 por amarelo, user13 por roxo, user21 por prateado e user22 por vermelho. Esta é uma forma de associar a que utilizador pertence uma determinada captação. Estas numerações também são utilizadas para associar as explicações no final de cada teste com as imagens que as representam.

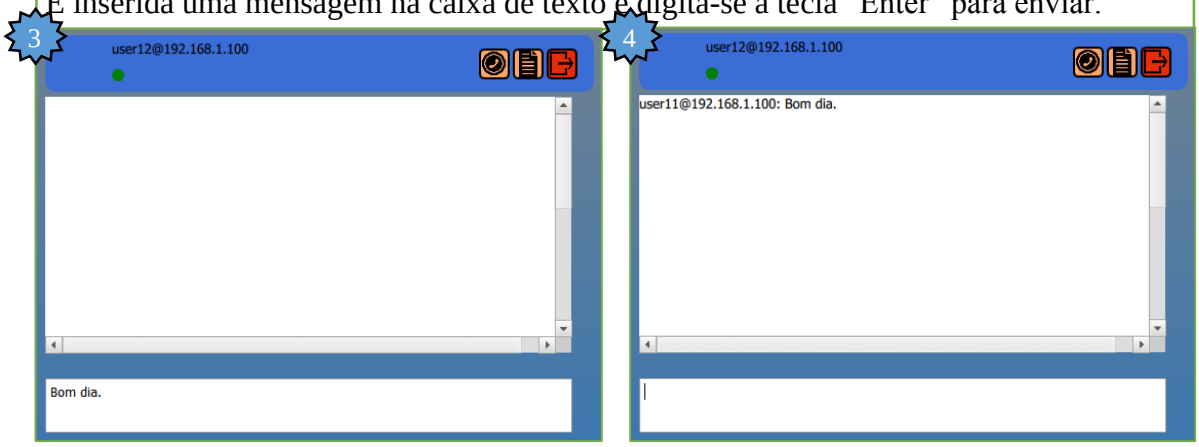
6.2. TESTE 1: MENSAGEM

Pretende-se com este teste verificar se os utilizadores de um servidor conseguem comunicar por meio de mensagens escritas, estabelecendo uma conversação individual entre si.

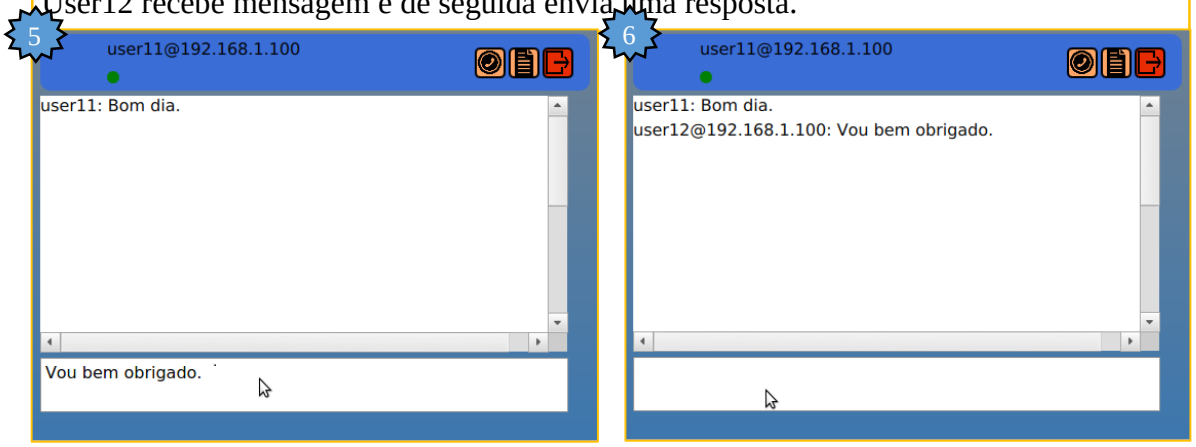
User11 seleciona user12 e clica na opção "Conversação individual".



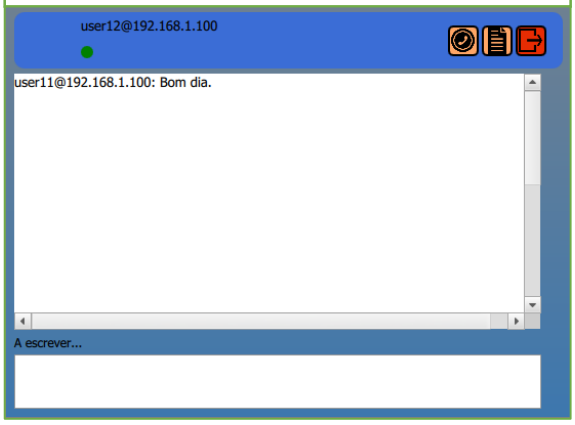
É inserida uma mensagem na caixa de texto e digita-se a tecla "Enter" para enviar.



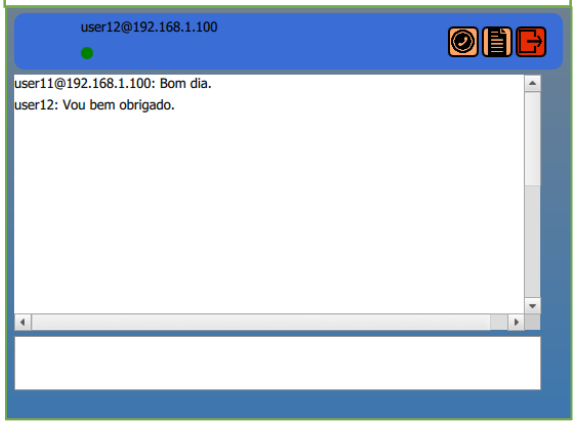
User12 recebe mensagem e de seguida envia uma resposta.



5 Enquanto user12 digita na caixa de texto.



7 Quando a mensagem é recebida por user11.

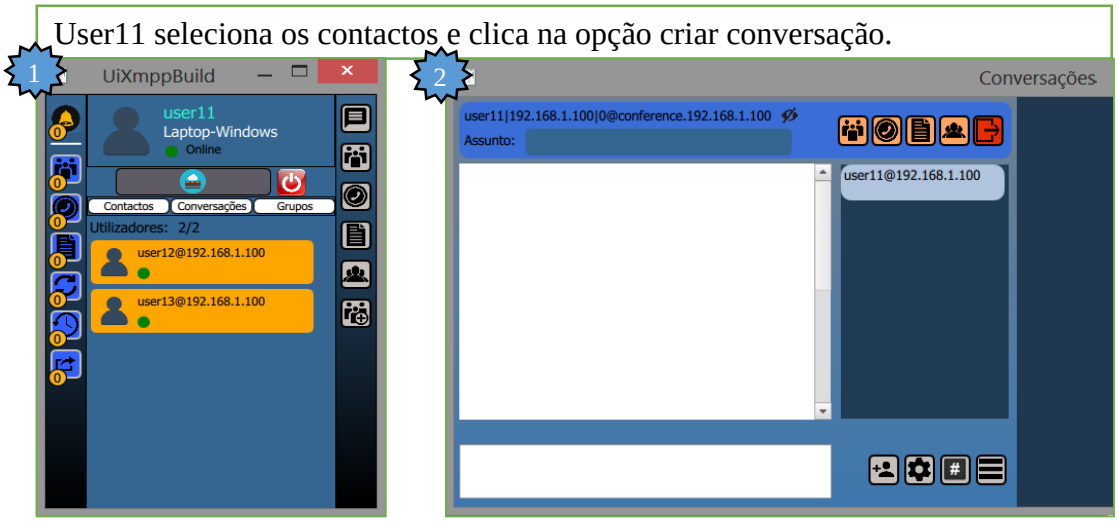


Deste teste pode-se concluir que o sistema de mensagem instantâneas entre dois utilizadores funciona corretamente. Quando inativa, a interface de conversações surge (2), as mensagens são enviada (4) (6) e recebidas (5) (7) com sucesso e quando uma mensagem está a ser escrita, surge no destinatário a indicação "A escrever" (5).

6.3. TESTE 2: INICIAR UMA CONVERSAÇÃO COM DOIS UTILIZADORES

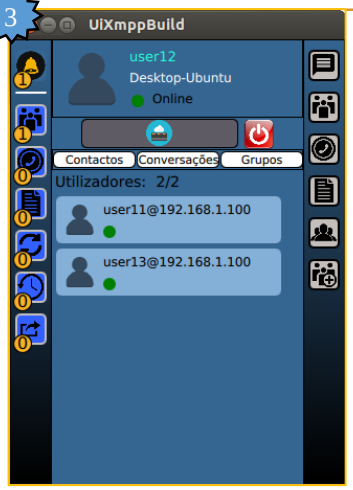
O objetivo deste teste é iniciar uma conversação entre os três utilizadores do servidor. É suposto o utilizador selecionar os contactos e ao clicar em "Criar conversação", ser criada uma sala de conversação e os contactos receberem um convite para a mesma.

User11 seleciona os contactos e clica na opção criar conversação.



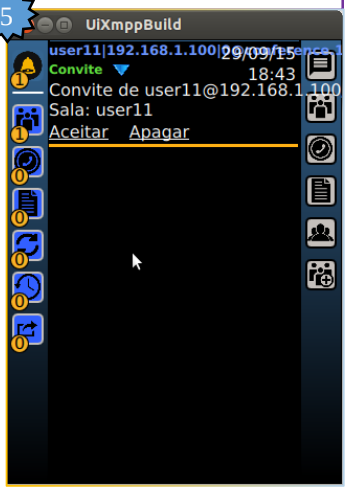
User12 e user13 recebem e abrem a notificação com o convite para a conversação.

3

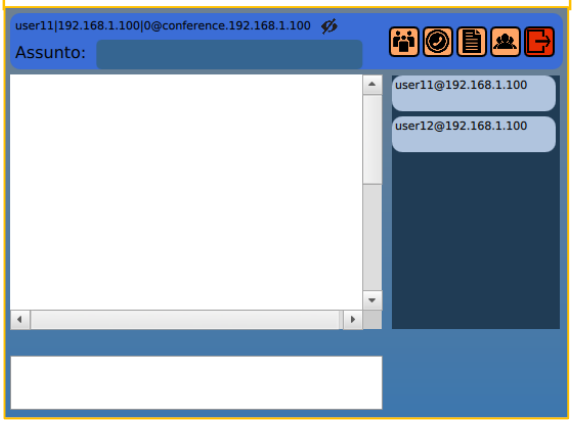


4

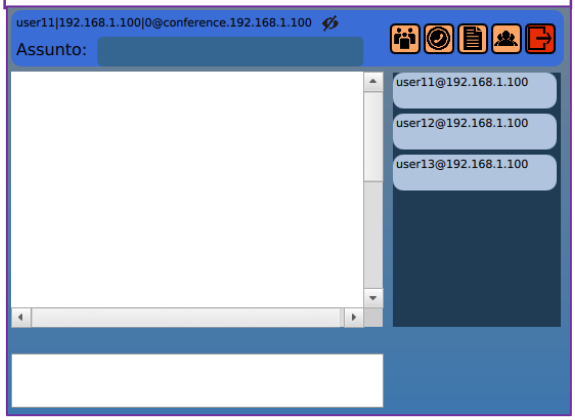
5



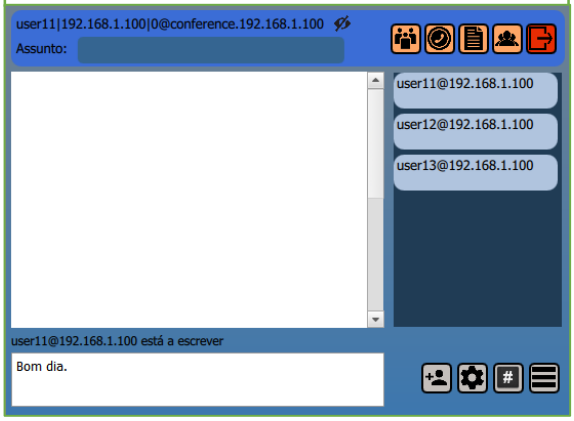
6 User12 aceita o convite.



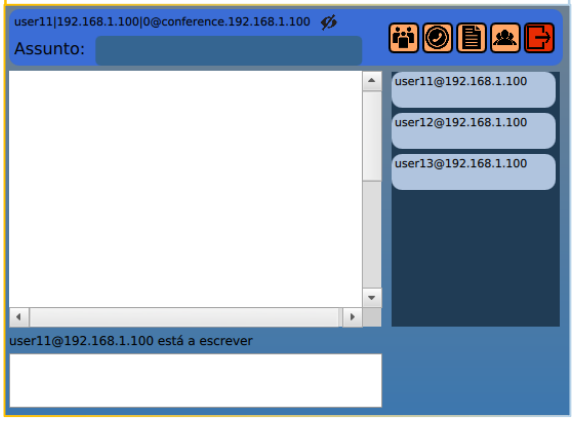
7 User13 aceita o convite.



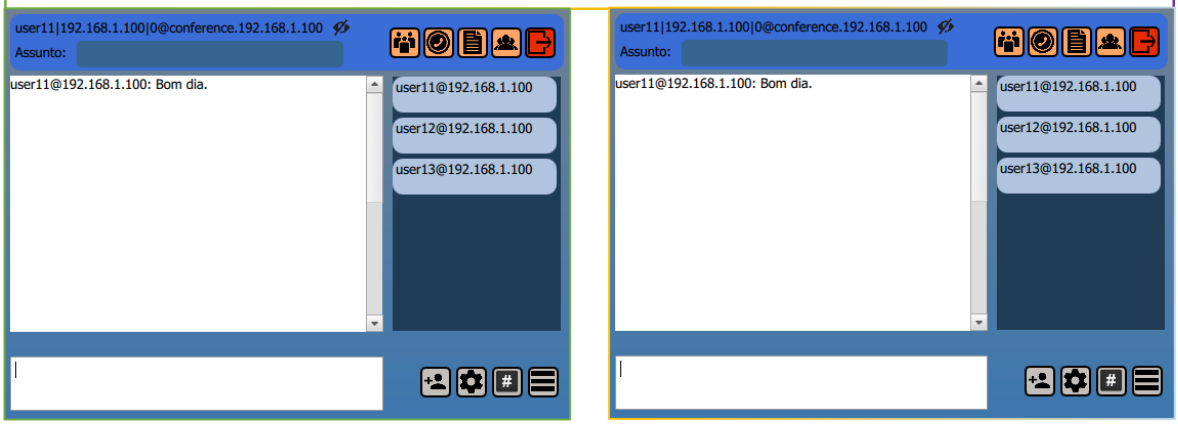
8 User11 digita e envia uma mensagem.



9 User12 e user13 detetam que user11 está a escrever.



10 User11, user12 e user13 recebe a mensagem sala.

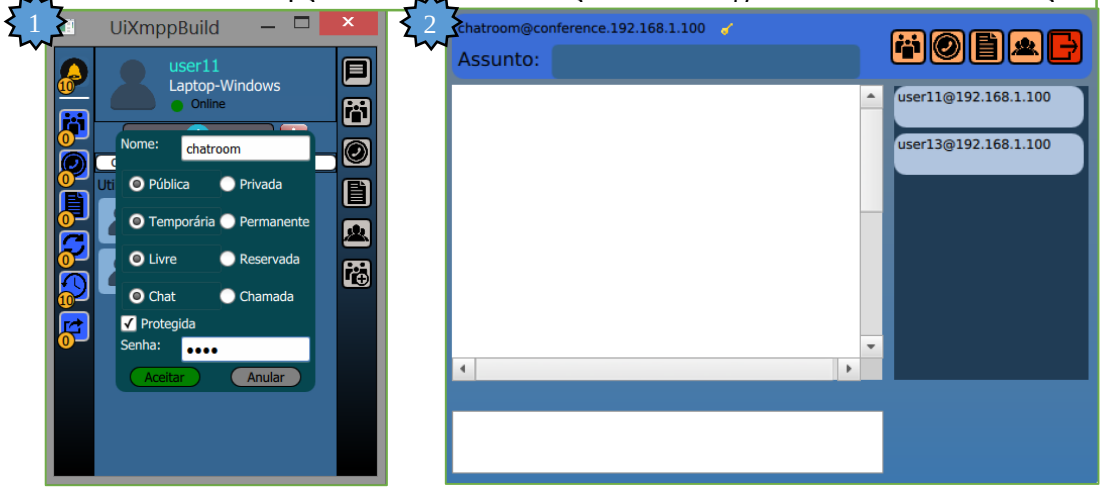


Neste teste, ao selecionar os contactos e clicar na opção considerada, o utilizador criou e entrou na sala (2) e enviou convites aos restantes utilizadores (3) (4) (5). Estes aceitaram os mesmos (6) (7) e acederam à sala de conversações. Quando user11 digitou a mensagem (8), todos os utilizadores receberam a informação que user11 estava a escrever (8) (9). Quando a mensagem foi enviada todos os utilizadores a receberam na interface (10).

6.4. TESTE 3: CRIAR UMA SALA DE CONVERSAÇÃO CONFIGURADA

Com este teste é pretendida a criação de uma sala de conversação com senha. De seguida irá convidar user13 para a sala. Por fim user12 e user13 tentarão entrar na sala. Se funcionar, irá aparecer em ambos os casos a interface de senha.

User11 clica na opção "Criar Conversação" e configura a sala de conversação

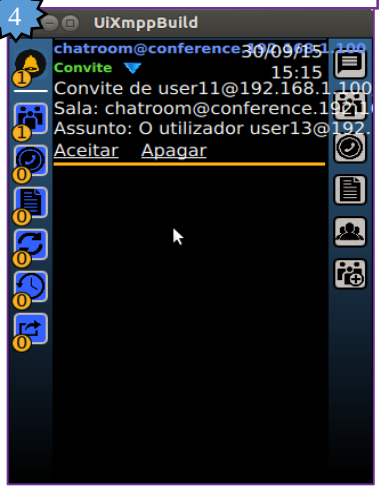


User11 seleciona user13 e convida para chatroom.

3

User13 recebe convite e abre notificações.

4

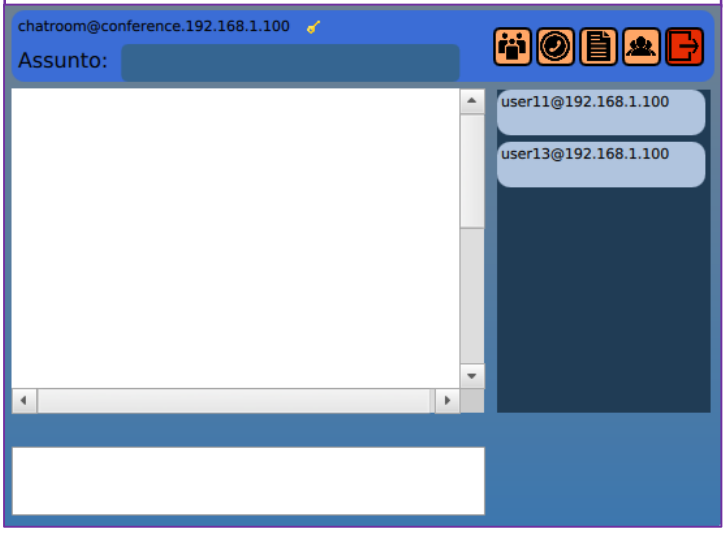


User13 clica em “Aceitar” e insere senha.

5

6

Confirma a senha, clicando no botão “Aceitar”.



7

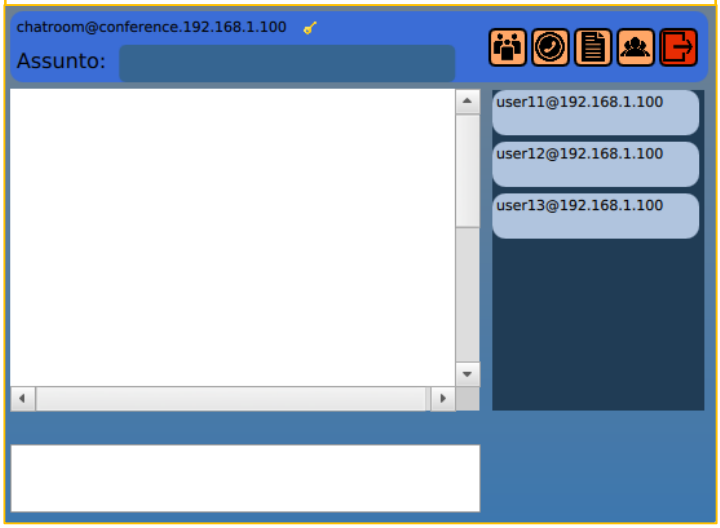
User12 seleciona a lista de salas públicas.

8

Clica no símbolo de acesso à sala e insere a senha.

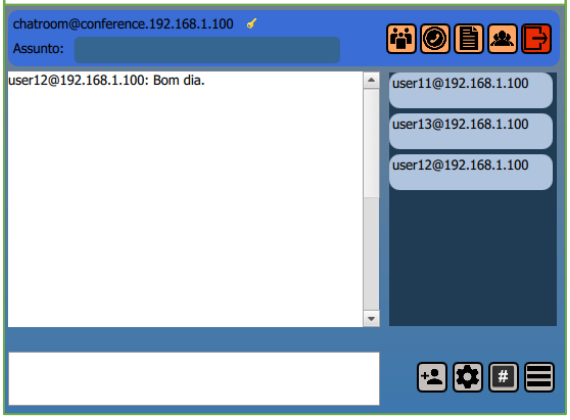
9

Confirma a senha, clicando no botão “Aceitar”.

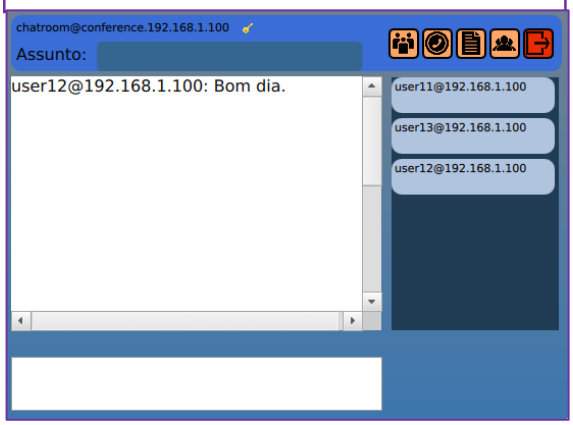


10 User12 digita e envia uma mensagem.

11 User11 recebe mensagem da sala.



11 User13 recebe mensagem da sala.



11 User12 recebem mensagem da sala.

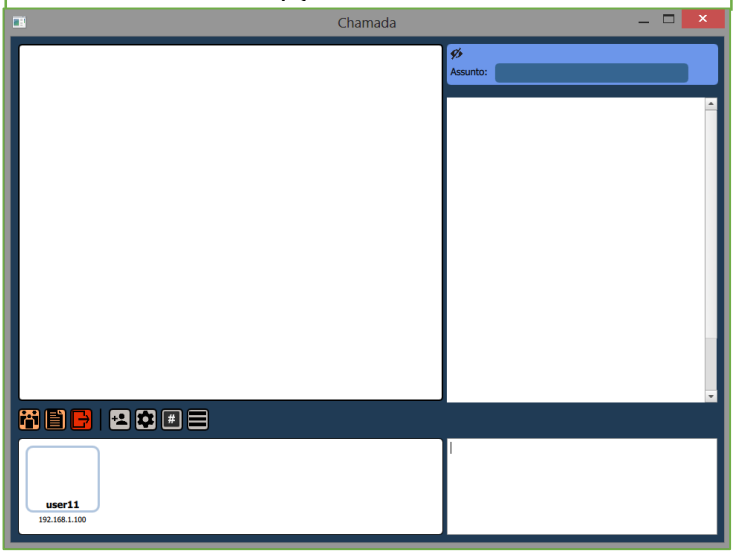
Este teste comprova o funcionamento do mecanismo que permite a partir da opção "**Criar conversação**" presente na interface principal, criar uma sala configurada com as opções pretendidas (1). A sala em questão foi configurada com uma senha e este mecanismo foi testado nas duas situações de acesso, quer a partir de um convite (5), quer recorrendo à lista de salas disponibilizado na interface principal (8). Qualquer método de acesso que utilize, o utilizador é sempre confrontado com o painel de senha. No caso de inserir a senha correta, conecta-se à sala (6) (9). Por fim foi ainda confirmado o funcionamento correto de envio e receção de mensagens entre os participantes da sala (10) (11).

6.5. TESTE 4: INICIAR UMA CHAMADA ENTRE TRÊS UTILIZADORES

Com este teste pretende-se estabelecer uma chamada de voz entre os três utilizadores do servidor. Se correr como desejado, quando na sala existirem pelo menos dois utilizadores, é estabelecida uma chamada XMPP entre o utilizador que entrou, com os participantes já presentes na sala.

1 User11 seleciona user12 e user13.

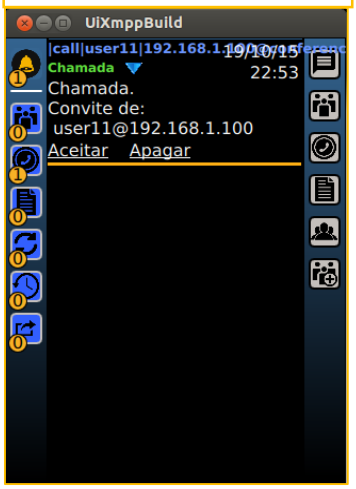
2 User11 seleciona a opção "Chamada" na interface.



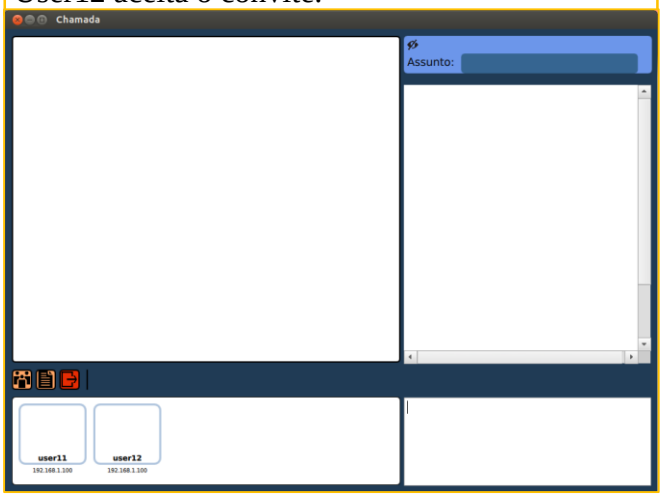
3 User12 recebe convite para a chamada.

3 User13 recebe convite para a chamada.

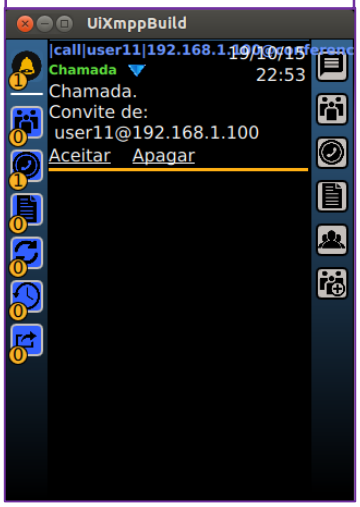
4 User12 abre notificações.



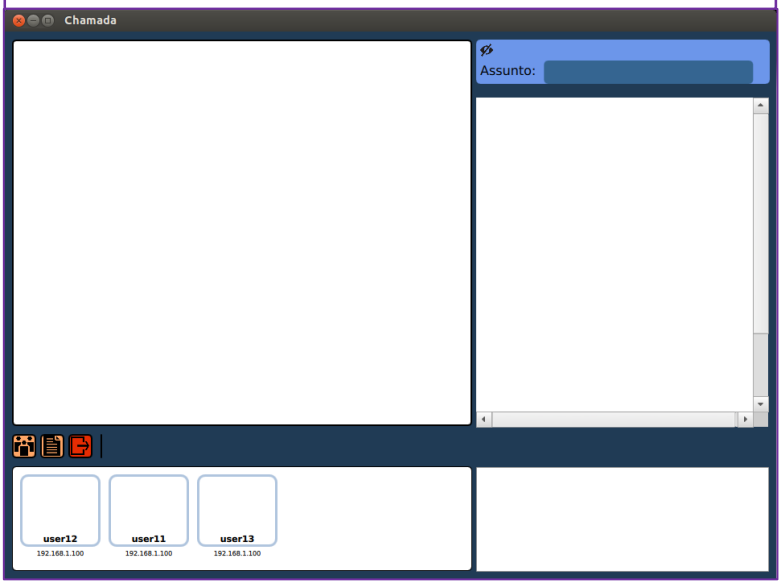
5 User12 aceita o convite.



6 User13 abre notificações.



8 User13 aceita o convite.



Neste teste user11 inicia uma chamada com user12 e user13. Ambos recebem o convite (3) e aceitam o mesmo (4) (6). Quando user12 entra na sala (5) e mais tarde também user13 se junta (8), consegue ouvir-se som nos clientes, ao falar no microfone.de user11. Comprova-se com este teste que as chamadas de voz funcionam corretamente.

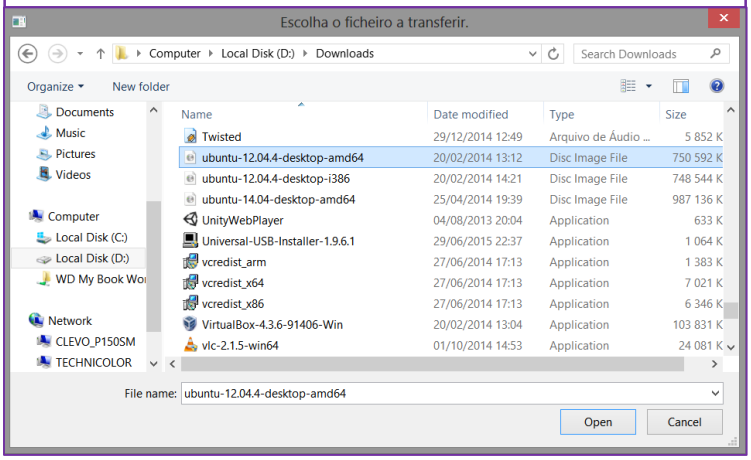
6.6. TESTE 5: ENVIAR UM FICHEIRO A UM UTILIZADOR

Neste caso é testado o mecanismo de envio ficheiros, onde se inclui a medição de performance das transferências. É ainda testada a hipótese dos utilizadores se autenticarem em dispositivos não habituais. User11 (normalmente associado ao portátil Windows) será autenticado a partir de um dos sistemas com Ubuntu. Já user13 será autenticado no portátil.

1 User13 autentica-se.

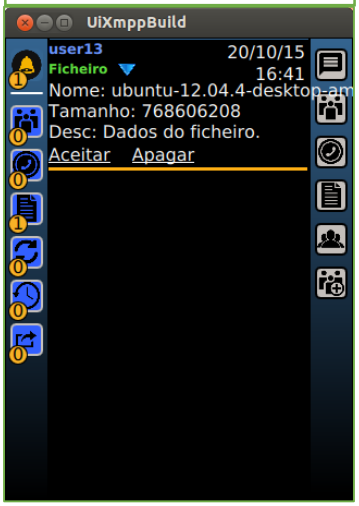
2 User13 seleciona user11.

3 User13 seleciona a opção “Enviar ficheiro”.

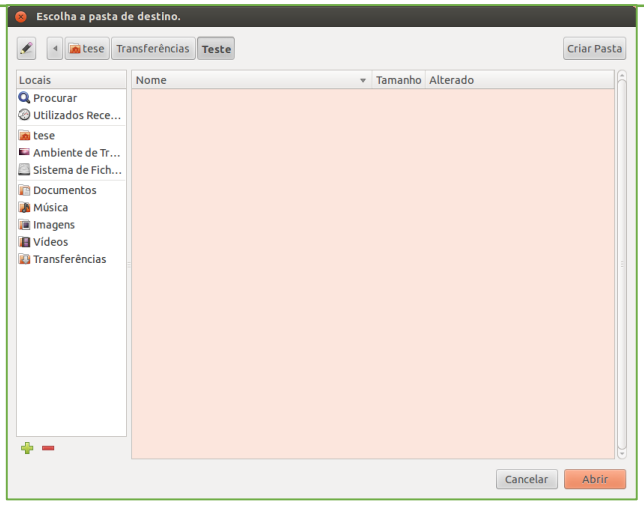


4 User11 recebe notificação.

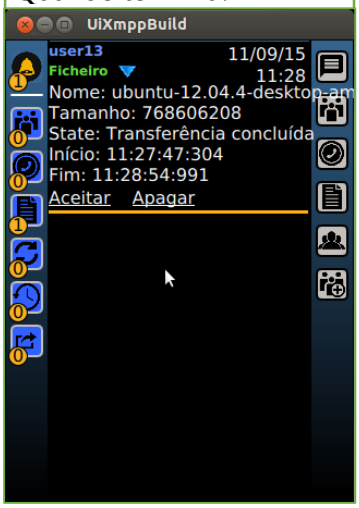
5 User11 abre notificações.



6 User11 aceita transferência e escolhe a pasta de destino.



7 Quando termina.



Após confirmar que a transferência funciona corretamente (7), é necessário avaliar a performance da mesma. Para isso efetuou-se a transferência de um ficheiro de user13 para user11, oito vezes.

Cada notificação de transferência contém dois *timestamps*, um que indica a data e hora do início da transferência e um segundo com a data e hora do fim da transferência. Com estes *timestamps* é possível recolher a duração de cada uma das transferências, subtraindo o *timestamp* Fim com o *timestamp* Início. Após obter a duração das oito transferências e sabendo o tamanho do ficheiro enviado, para cada um dos casos, se o tamanho do ficheiro em MB (MegaBytes) for dividido pela duração em segundos, é possível calcular a taxa de transferência média em MB/s (Megabytes por segundo). É necessário ainda considerar que as interfaces de redes dos computadores utilizados para este teste têm largura de banda de 100 Mb/s. Após a recolha dos *timestamps* e de todas as considerações anteriormente delineadas, é apresentado na tabela 4, os resultados derivados destes testes.

Tabela 4 Tabela de resultados do teste de transferências

#	Início	Fim	Duração	Tamanho (B)	Tamanho (MB)	Taxa (MB/s)	Taxa (Mb/s)
1	11:27:47.304	11:28:54.991	67.687	768606208	733	10,829	86,634
2	11:58:14.725	11:59:27.213	72.512	768606208	733	10,109	80,869
3	11:59:42.132	12:00:54.787	72.655	768606208	733	10.088	80,710
4	12:01:25.163	12:02:37.595	72.432	768606208	733	10,119	80,959
5	12:03:35.746	12:04:48.493	71.747	768606208	733	10,216	81,731
6	12:05:41.200	12:06:53.869	72.669	768606208	733	10,087	80,695
7	12:07:13.817	12:08:26.979	73.162	768606208	733	10,189	80,151
8	12:09:08.635	12:10:16:454	67.819	768606208	733	10,808	86,465

Resultado mais reduzido: 80,710 Mb/s

Resultado mais elevado: 86,634 Mb/s

Média: 82,277 Mb/s

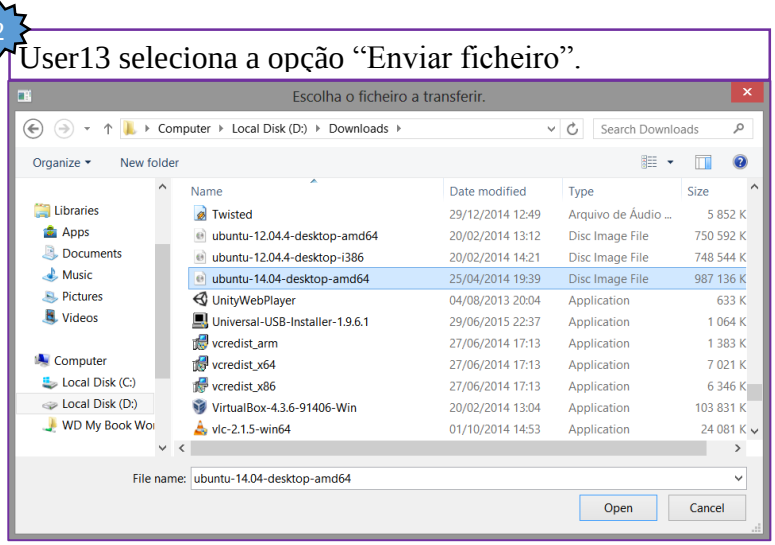
Considerando a largura de banda de 100 Mb/s e os clientes estão em redes distintas, a velocidade média de 82 Mb/s é aceitável. Também tem de se considerar que os 100Mb/s são teóricos. Na prática, uma interface não consegue chegar a esses valores [122] [123].

6.7. TESTE 6: ENVIAR UM FICHEIRO A MÚLTIPLOS UTILIZADORES

De forma a perceber como o gestor de transferência *QXmppTransferManager* lida com múltiplas transferências em simultâneo, é efetuado um teste onde user13 envia simultaneamente para user11 e user12, o mesmo ficheiro ao mesmo tempo.

1 User13 seleciona user11 e user12.

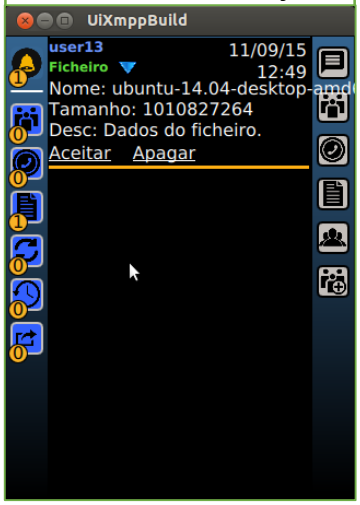
2 User13 seleciona a opção "Enviar ficheiro".



3 User11 recebe notificação.

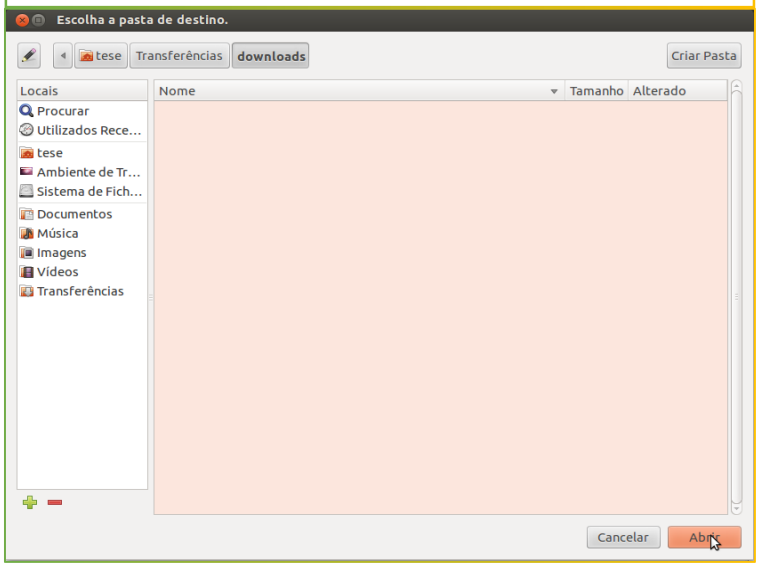
3 User12 recebe notificação.

4 User11 abre notificações.

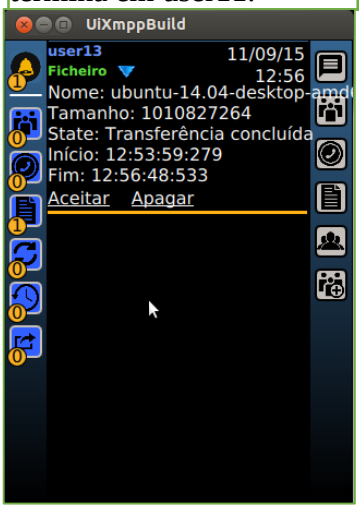


5 User12 abre notificações.

6 User11 e user12 aceitam as transferências em simultâneo.



7 Quando transferência termina em user11.



8 Quando transferência termina em user12.

No quadro seguinte podem ser observados os resultados bem como os cálculos efetuados para este teste.

Tamanho (MB)	964	964
Taxa (MB/s)	5,704	5,738
Taxa (Mb/s)	45,632	45,904

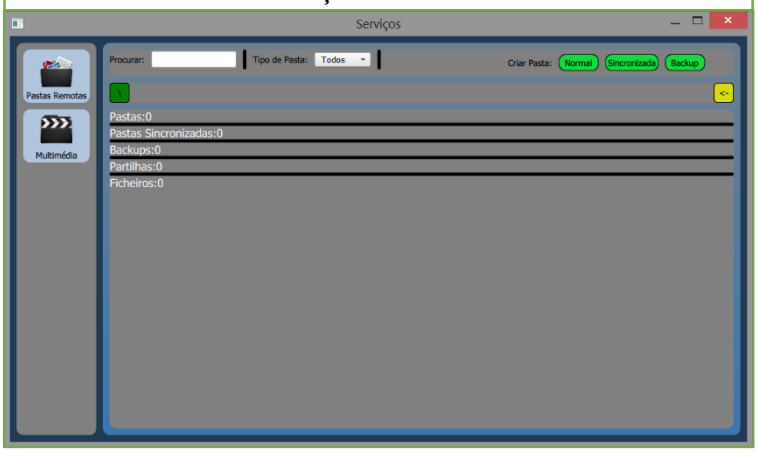
Observando os resultados obtidos e comparando os mesmos com o teste anterior, podem-se tirar duas conclusões. Primeiro, ao somar a taxa de ambas as transferências (91,536) conclui-se que a soma desta é um pouco mais elevado que as velocidades anteriormente conseguidas (82,277). O facto de as transferências terem um desfasamento de três segundos, também afeta os valores obtidos. A segunda conclusão a tirar, prende-se com o facto de as transferências ocorrerem em simultâneo. Os gestores de transferências dos cliente adaptam-se dinamicamente para que quando ocorrem múltiplas transferências em simultâneo, estas ocorram em paralelo, sendo que para isso, a largura de banda da ligação seja dividida entre as várias transferências.

6.8. TESTE 7: CRIAR PASTA REMOTA

O objetivo deste teste é criar uma pasta remota, efetuar o *upload* de um ficheiro para a mesma e efetuar o *download* desse ficheiro para uma segunda pasta. Pretende-se assim comprovar que o sistema de pastas remotas funciona e que se pode carregar e descarregar ficheiros.

1 User11 autentica-se.

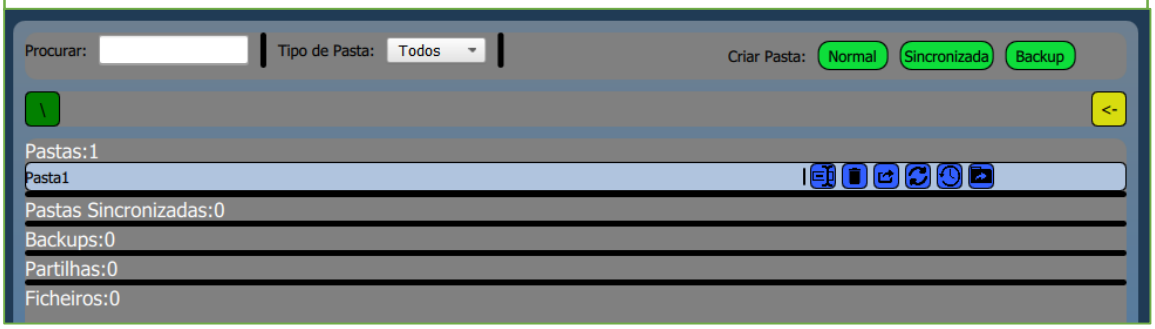
2 Clica-se no botão ”Serviços”.



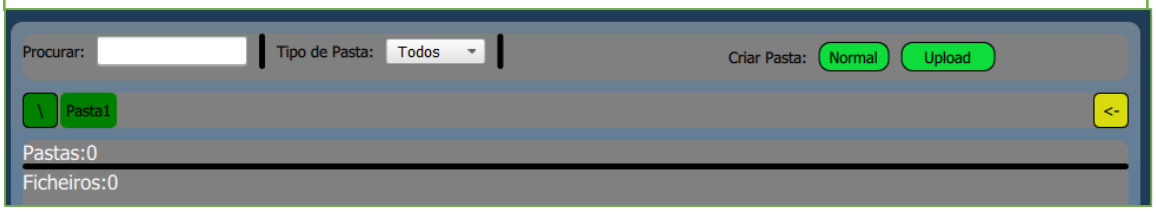
3 De seguida clica-se no botão “Normal” para criar uma pasta remota.

4 Insere-se o nome da pasta que deseja.

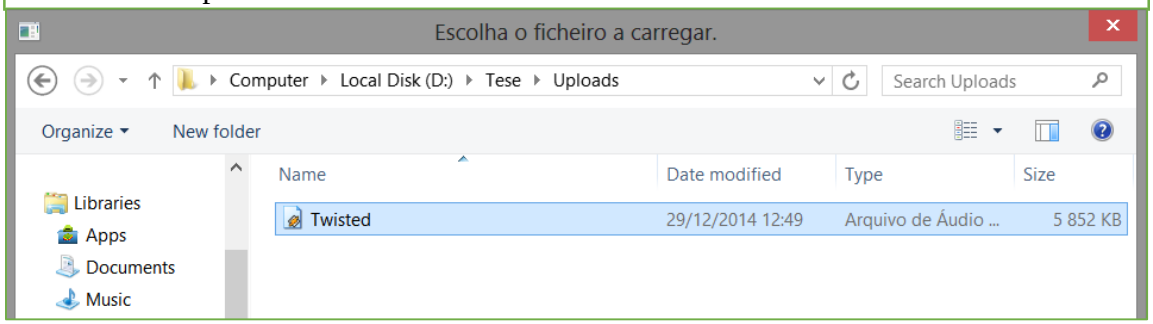
4 Clica-se no botão “Criar”.



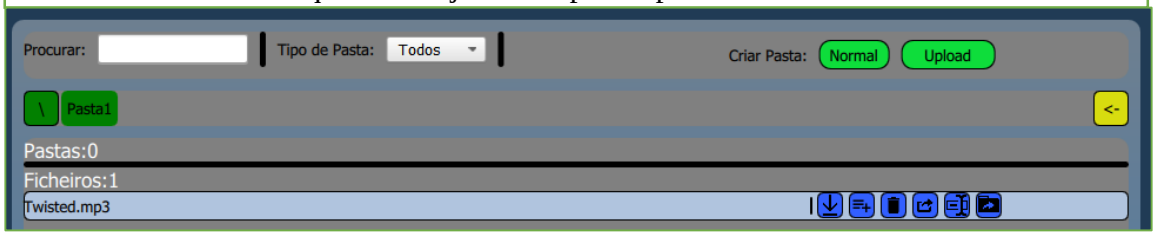
5 Clica-se na “Pasta1”.



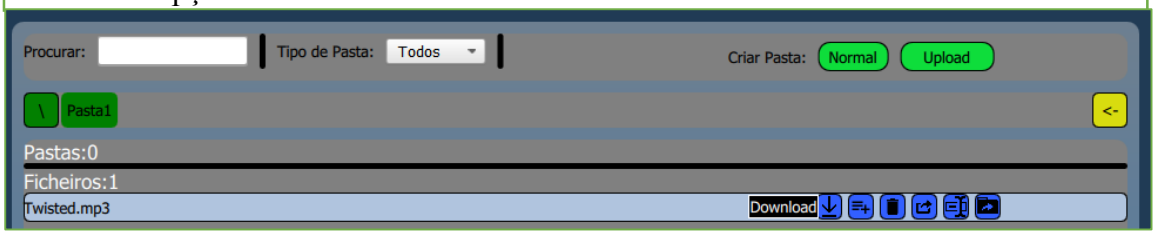
6 Clica-se em “Upload” e escolhe-se o ficheiro local a enviar.



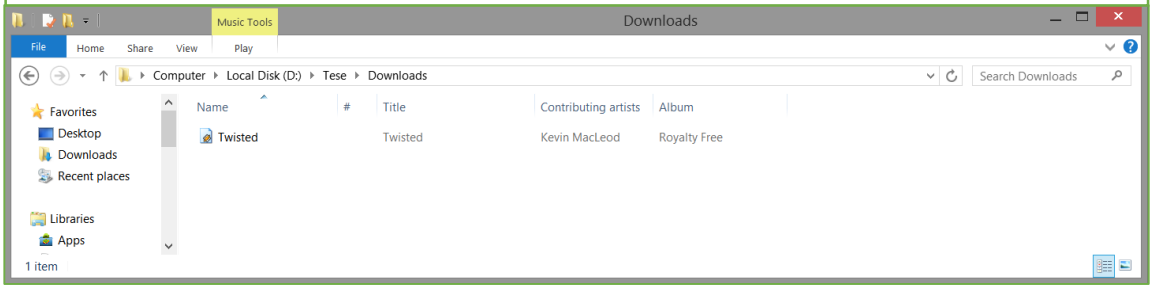
7 Confirma-se o ficheiro que se deseja enviar para a pasta remota.



8 Clica-se na opção “Download” do ficheiro.



9 Escolhe-se a pasta local de destino para o ficheiro e confirma-se a escolha.



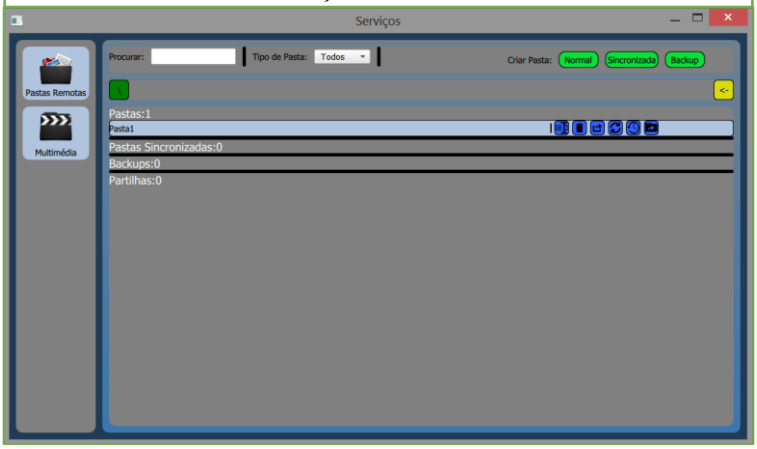
Neste teste foi criada uma pasta remota no servidor (4) e ao carregar o ficheiro do dispositivo local (6), este surgiu na interface da pasta remota (7). De seguida procedeu-se ao descarregamento do ficheiro para uma segunda pasta (8). As transferências quer do *upload* quer do *download* decorreram com sucesso.

6.9. TESTE 8: CRIAR PASTA SINCRONIZADA

Com este teste pretende-se criar uma pasta remota que esteja sincronizada com uma pasta local. Após terminado o processo coloca-se um ficheiro na pasta local, para verificar se a pasta remota atualiza com o mesmo.

1 User11 autentica-se.

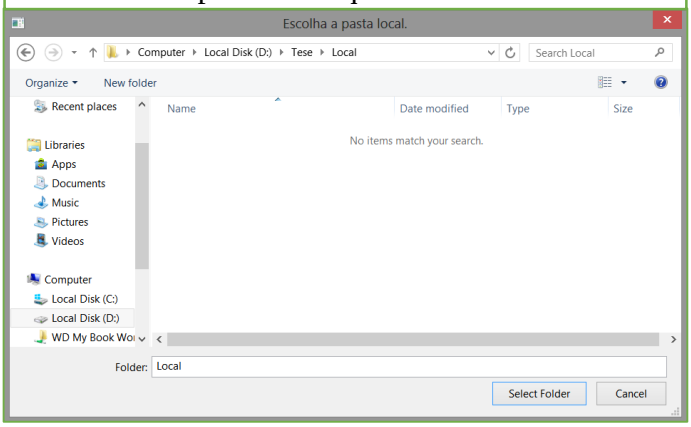
2 Clica-se no botão ”Serviços”.



3 De seguida clica-se no botão “Sincronizada”.

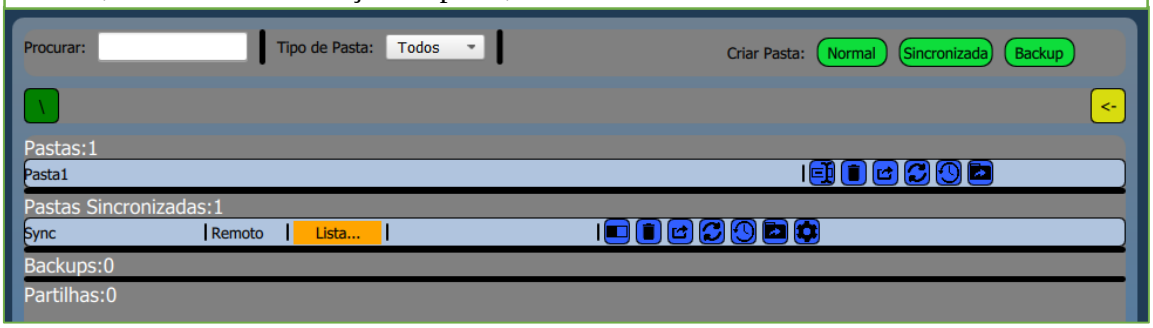
4 Insere-se o nome da nova pasta.

5 Seleciona-se a pasta local que será sincronizada.

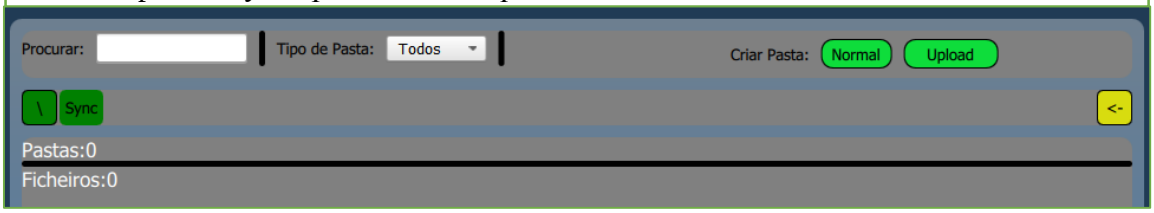


6 Seleciona-se a pasta local desejada.

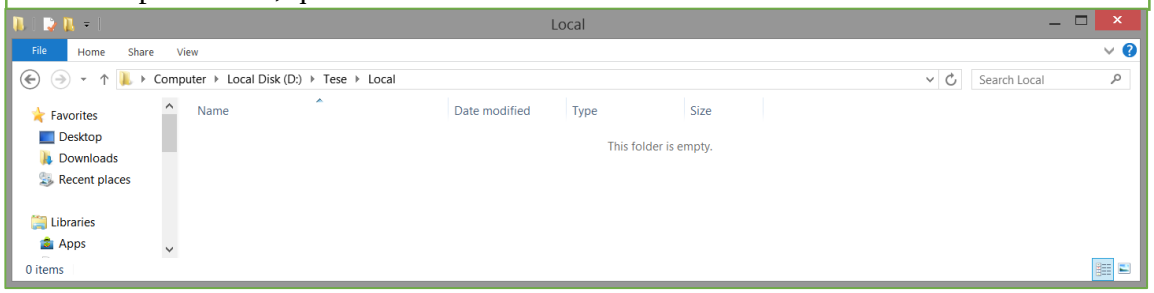
7 Por fim, confirma-se a criação da pasta, recorrendo ao botão "Criar".



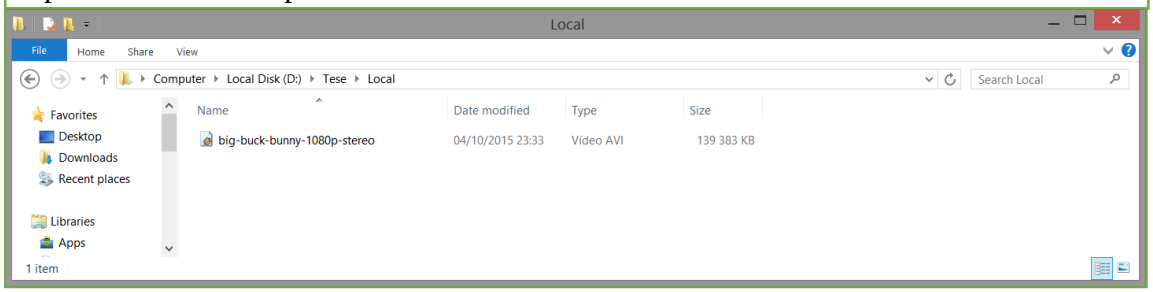
8 Abre-se a pasta "Sync" para verificar que não existe conteúdo na mesma.



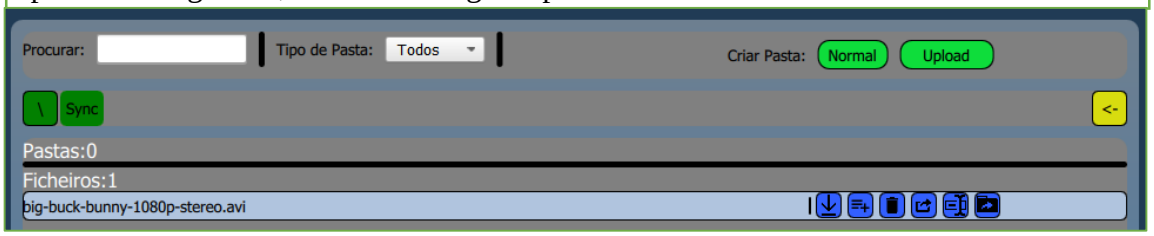
9 Abre-se a pasta local, que se encontra atualmente sincronizada.



10 Copia-se um ficheiro para a mesma.



11 Após cinco segundos, o ficheiro surge na pasta remota.



12 Surge a notificação de sincronismo.

13 Acede-se às notificações.

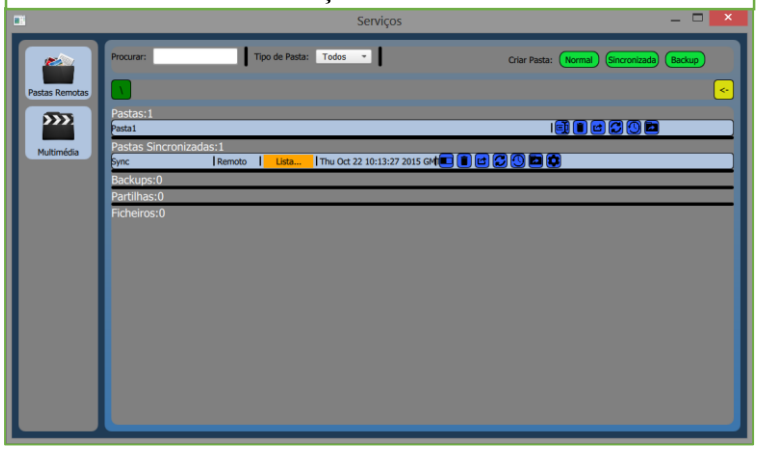
A partir deste teste é possível verificar que após a criação da pasta sincronizada (7),ao colocar o ficheiro na pasta local (10), a pasta remota atualiza automaticamente com o ficheiro (11). Também o sistema de notificações funcionou devidamente, notificando o utilizador sobre o sincronismo (12) (13).

6.10. TESTE 9: CRIAR *BACKUP* DE UMA PASTA LOCAL

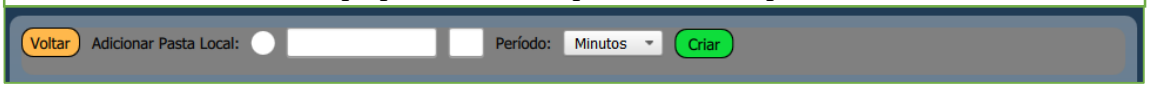
Este teste tem o objetivo de verificar o funcionamento do mecanismo de pasta tipo *backup*. Se funcionar como planeado, ao ser criada uma pasta *backup* associada a uma pasta local, ao fim do período configurado é efetuada uma cópia integral do conteúdo da pasta local.

1 User11 autentica-se.

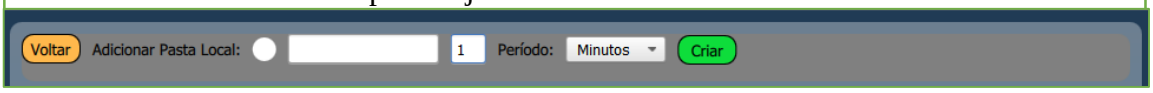
2 Clica-se no botão ”Serviços”.



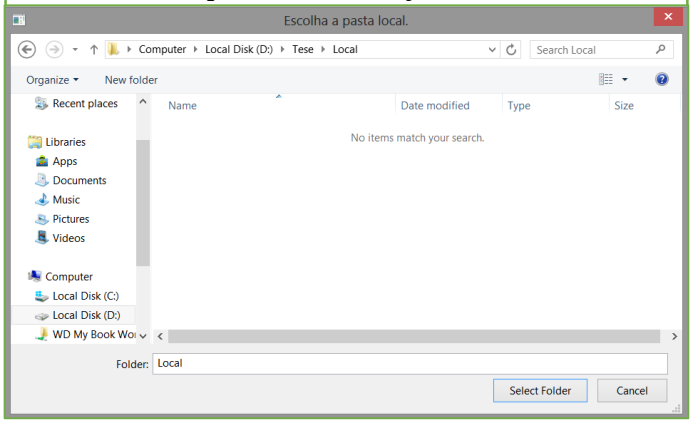
3 Clica-se no botão ”Backup” para criar uma pasta de Backup remota.



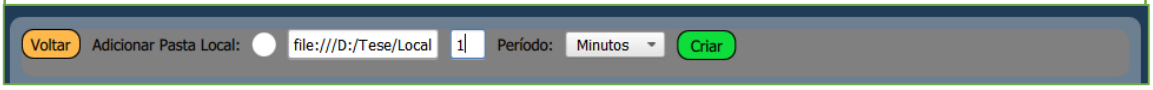
4 Insere-se o intervalo de tempo desejado.



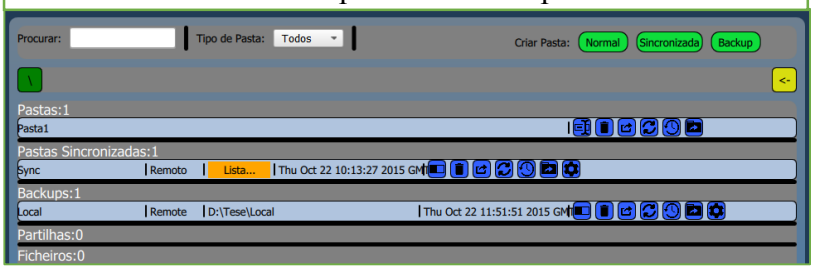
5 Seleciona-se a pasta local desejada.



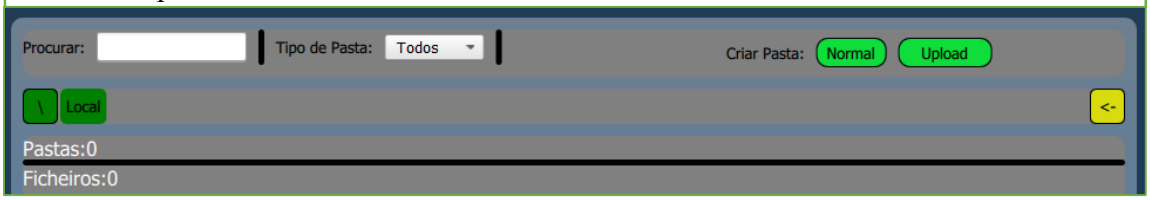
6 Confirma-se a seleção da pasta local.



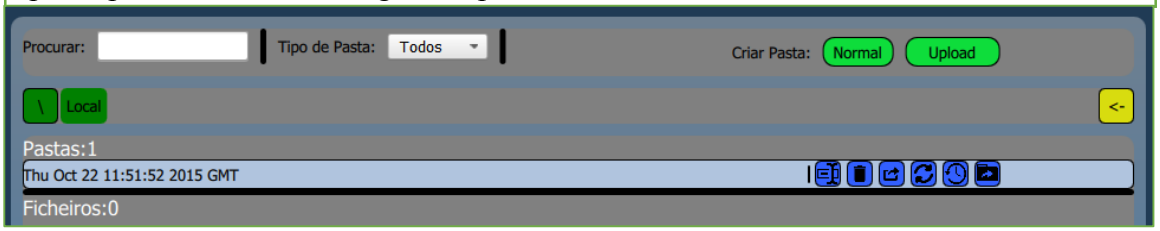
7 Clica-se no botão “Criar” para finalizar o processo.



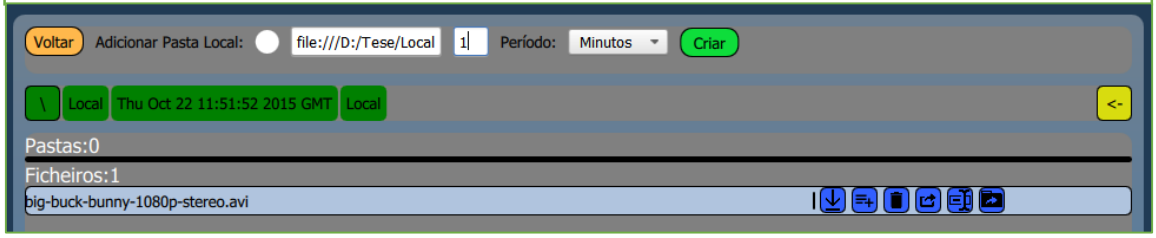
7 Clica-se na pasta “Local”.



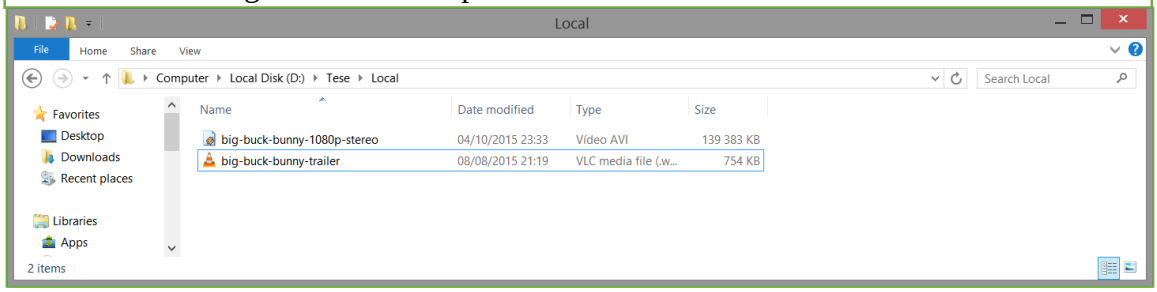
8 Após esperar um minuto, surge uma pasta nova em “Local”.



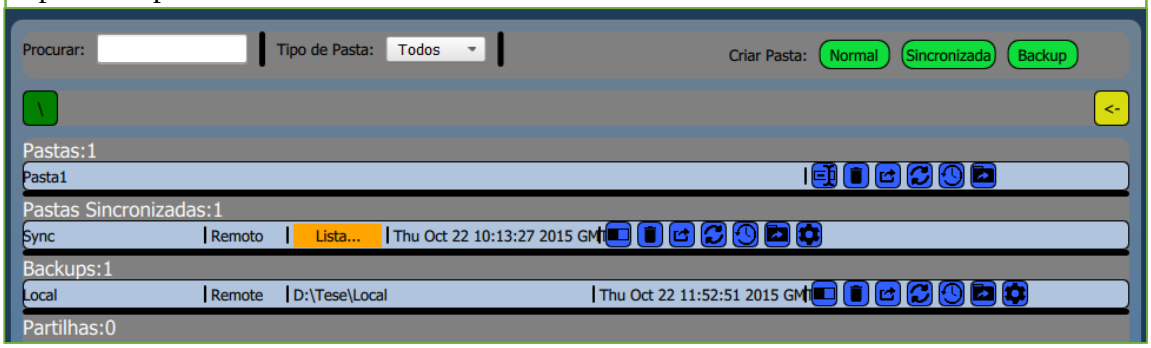
9 Clica-se na nova pasta.



10 Adiciona-se um segundo ficheiro à pasta local.



11 Espera-se aproximadamente um minuto.



12 Clica-se em "Local".

13 Clica-se na nova pasta (Thu Oct 22 11:52:51 2015 GMT).

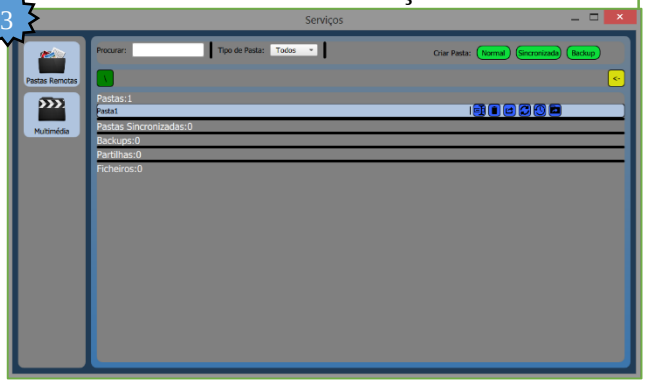
Neste teste foi confirmado o funcionamento do sistema de *backups* do servidor. Ao criar uma pasta tipo *backup* associada a uma pasta local (7), o servidor vai efetuando cópias integrais desta pasta, sempre que o intervalo pré-definido chega ao fim (8) (12). Como é possível observar ao adicionar um segundo ficheiro à pasta local (10), quando a segunda pasta é criada (12), já contém os dois ficheiros locais (13).

6.11. TESTE 10: EFETUAR UMA PARTILHA DE UM VÍDEO ENTRE UTILIZADORES DE DOMÍNIOS DISTINTOS

Este último teste tem como objetivo testar dois mecanismos do sistema. Um refere-se ao mecanismo de partilha de pastas. User11 irá efetuar uma partilha de uma pasta com user21. Não só este teste permite testar o mecanismo em si, mas também permitirá demonstrar que é possível partilha pastas com utilizadores de servidores remotos. Dado que a pasta irá conter um ficheiro multimédia, será também testado o serviço de multimédia. Se funcionar corretamente, user21 deverá conseguir executar remotamente o ficheiro multimédia que se encontra na pasta partilhada, de um utilizador de um segundo servidor.

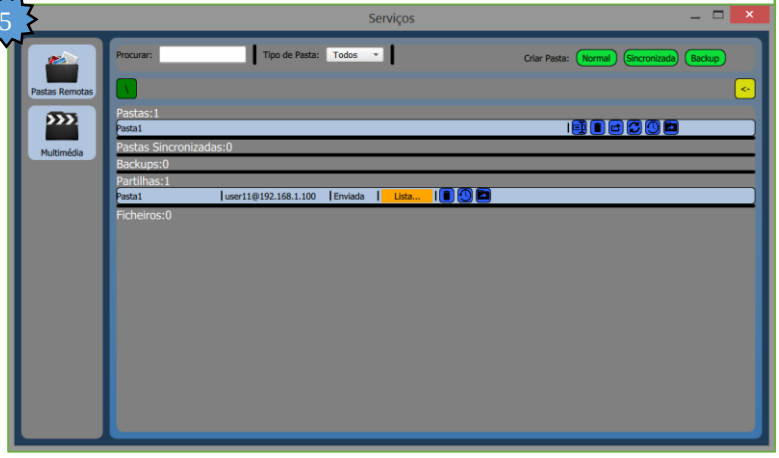
User11e user21 autenticam-se.

User11 clica no botão de serviços.

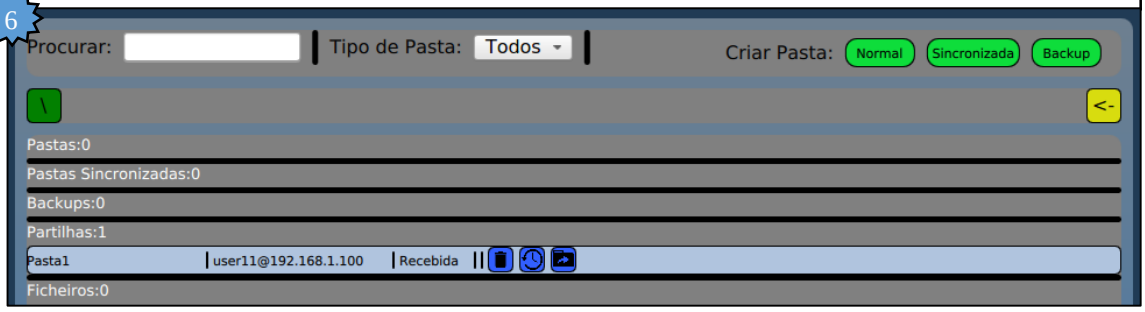


User11 seleciona user21.

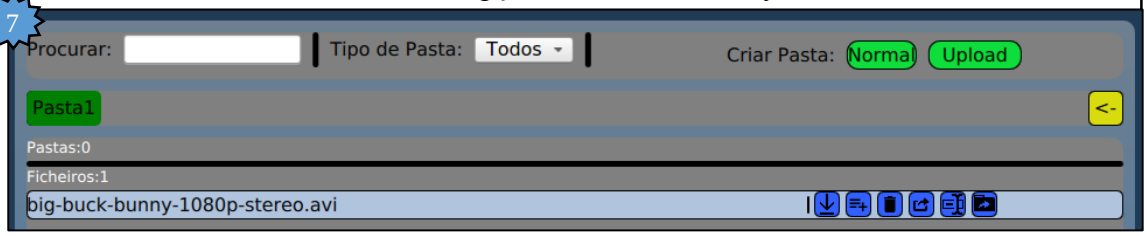
User11 clica na opção partilhar de “Pasta1”.



6 Surge nas partilhas de user21, “Pasta1”.

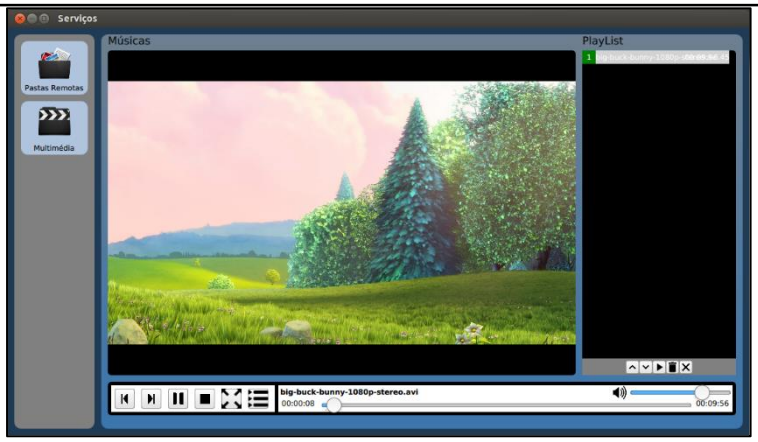


7 User21 abre “Pasta1” e seleciona a opção “Adicionar à Playlist” do ficheiro.



8 User21 clica no separador “Multimédia”.

9 Executa o ficheiro existente na Playlist.



Com este teste foi possível comprovar que não só os utilizadores de servidores diferentes podem partilhar pastas entre si (6), como também confirma a possibilidade destes

executarem conteúdo multimédia (9), remotamente e de um servidor da comunidade. Pelo facto do serviço de multimédia utilizar transferências de ficheiros como base, significa que as transferências também funcionam entre comunidades.

Após efetuado este teste, comprova-se que o mecanismo de VoD e AoD funcionam corretamente, adicionando o ficheiro pretendido à lista de execução, quando selecionada a opção correspondente, e de seguida a sua execução. Este mesmo teste foi efetuado para vários formatos de vídeo e música e para Windows e Ubuntu, de forma a verificar que formatos o sistema suporta atualmente. Na tabela 5, estão expostos os resultados destes testes.

Tabela 5 Tabela de resultados do teste de VoD e AoD

	Vídeo					Música	
Formatos	MP4	WebM	MKV	AVI	MOV	Mp3	WAV
Windows	✗	✓	✓	✗	✓	✗	✗
Ubuntu	✓	✓	✓	✓	✓	✓	✓

Como se pode verificar, alguns formatos não funcionam em Windows. Quando isto acontece, significa que a leitura do ficheiro falha antes de a transferência terminar.

6.12. DISCUSSÃO DE RESULTADOS E TESTES

Ao longo deste capítulo foram apresentados os testes efetuados para demostrar o funcionamento dos serviços e mecanismos implementados neste projeto. Nesta secção são resumidos os pontos essenciais e tiradas algumas ilações dos testes apresentados.

A primeira consideração a tirar é a de que todos os serviços funcionam corretamente, tendo sido demonstrada a sua execução a partir da captação das interfaces que surgem durante os testes. Mensagens, salas de conversação, transferências, chamadas, grupos, pastas remotas, pastas sincronizadas, *backups*, partilhas, bem como VoD e AoD, todos os serviços funcionaram devidamente. Outro ponto importante a considerar é a velocidade de transmissão das transferências. Esta não é perfeita, pois não utiliza toda a largura de banda disponível, mas tendo em conta que num ambiente real nunca se consegue atingir as velocidades teóricas [122] e que os pacotes de rede relativos às transferências passam pelo *gateway* e por redes distintas, as velocidades conseguidas são aceitáveis [123].

Os únicos casos onde existem reservas quanto à funcionalidade do serviço é no caso do *VoD*. Aqui o fator chave é o *QMediaPlayer*, biblioteca do *Qt* responsável por executar os ficheiros.

Dependendo dos formatos dos ficheiros, o cliente Windows por vezes não é capaz de executar o ficheiro antes de este ser completamente descarregado. Um dos exemplos está no formato MP3. Neste caso, o cliente Windows só consegue executar o ficheiro quando a transferência termina. Já em Ubuntu a execução ocorre sem problemas. A biblioteca do Qt QMultimedia utiliza *backends*, que são *interfaces* com as quais o Qt acede aos recursos e *codecs* do sistema operativo. Estes variam conforme o sistema operativo, oferecendo determinadas funcionalidades. A informação sobre os backends dinponíveis para o QMulimedia pode ser encontrada em [124]. Em Windows o *QMediaPlayer* utiliza o *DirectShow* para aceder às funcionalidades multimédia. No caso do *Ubuntu*, é utilizado o *GStreamer*, uma *framework* multimédia que executa as operações multimédia no *Ubuntu*. Dado que as estas são diferentes e que em *Ubuntu* não ocorre qualquer erro em nenhum dos formatos testados, conclui-se que existe um possível problema de compatibilidade ou um bug na interação entre o *QMediaPlayer* e o *DirectShow*.

7. CONCLUSÕES

Todo o processo descrito ao longo deste documento teve como motivação a possibilidade de conjugar as vantagens dos serviços de *cloud* com as vantagens de um sistema local. Assim surgiu esta ideia, de concentrar num servidor local configurável, um conjunto de serviços habitualmente dispersos na internet por diversas entidades, que fossem de encontro às necessidades e preferências dos consumidores e que pudessem ser acedidos remotamente.

Com estes pressupostos em consideração, o objetivo principal deste trabalho era centralizar um conjunto de serviços num sistema local e criar uma forma de aceder a todos esses serviços remotamente, a partir de uma interface unificada. Este objetivo foi cumprido recorrendo às componentes cliente e servidor, desenvolvidos durante o projeto. Mais especificamente os serviços que se desejava implementar incluíam um sistema de presença que permitisse detetar a presença e o estado dos utilizadores, mensagens instantâneas, salas de conversação, chamadas/conferências, transferências de ficheiros, criação de grupos de contactos, pastas remotas, pastas sincronizadas, pasta de *backup*, partilha de pastas, bem como VoD (video-on demand) e AoD (*audio-on demand*). No geral estes serviços foram implementados com sucesso, sendo que no caso do VoD e AoD alguns formatos não funcionaram corretamente. A performance das transferências e o facto de o QXmpp conseguir adaptar-se a múltiplas transferências, dinamicamente e automaticamente, demonstra que se está na presença de um sistema de transmissão de ficheiros robusto, flexível e funcional.

Com este projeto, os utilizadores conseguem usufruir de um conjunto de serviços de comunicação, gestão de pastas e de consumo de conteúdo multimédia, de forma intuitiva e com apenas um único cliente. De outra forma teriam de instalar múltiplos clientes para múltiplos serviços ou aceder a vários serviços web distintos. Com a propensão atual de convergir tecnologias e centralizar o seu consumo num dispositivo que utilize interfaces intuitivas, este sistema vem cobrir essas tendências. Outro fator que este trabalho resolve é o facto de conservar os dados e os ficheiros dos utilizadores no servidor. Mesmo no caso das comunidades, nem os utilizadores nem os *bots* de outro servidor podem aceder diretamente a pastas e recursos do servidor onde o utilizador está registado, a não ser que se encontrem partilhadas. Considerando também o mecanismo de comunidades implementado, este possibilita aos utilizadores criarem comunidades de servidores, e possam comunicar entre si. O sistema desenvolvido tem o potencial de se integrar não só em residências, como também em pequenas empresas e até hotéis. Imagine-se uma pequena empresa com dez colaboradores. Bastando um servidor na empresa, estes poderiam comunicar entre si criar salas de conversação, efetuar chamadas e enviar ficheiros. Poderiam também partilhar pastas e consumir vídeos ou músicas. Expandindo este cenário para um hotel, os seus utilizadores poderiam comunicar entre si da mesma forma e até o hotel prestar serviços próprios, como VoD e AoD aos seus utilizadores. Utilizando ainda a possibilidade de se estabelecer comunidades, a empresa poderia conectar o seu servidor a uma sucursal ou divisão pertencente à mesma empresa e assim todos os utilizadores de ambos os locais poderiam comunicar ente si utilizando os mecanismos já apresentados. Também no caso de um hotel, se este pertencer a um grupo hoteleiro, significa que existem outros hotéis que se podem conectar entre si e formar uma comunidade de hotéis, oferecendo as vantagens de uma comunidade à rede de hotéis.

É importante também analisar desenvolvimentos futuros que se poderiam implementar posteriormente de forma a melhorar os serviços já disponibilizados e incorporar novos serviços no sistema. Nas conversações individuais, salas de conversação e chamadas, poderia ser implementado na área de mensagens, um mecanismo que permitisse inserir fotografias e até vídeos na própria conversação. Sendo esta uma possibilidade já oferecida em algumas soluções [125] [126], seria uma camada extra de interatividade e dinâmica que os utilizadores poderiam usufruir quando comunicam por meio de texto. As conferências também poderão implementar vídeo. A interface de chamadas já contém a janela preparada e o qxmpp pode implementar vídeo utilizando o formato VP8. Neste momento, quando o

mecanismo de pastas sincronizadas deteta diferenças entre duas pastas sincronizadas, os ficheiros são copiados na íntegra. De forma a combater este problema, e assim diminuir o tráfego gerado pelo serviço, poderia ser implementado um sistema como o *delta coding*. Como no caso apresentado em [127], este mecanismo pode ser utilizado para detetar disparidades entre os ficheiros e transferir apenas a diferença. Também seria interessante a implementação de novos serviços que oferecessem funcionalidades adicionais ao sistema. Um desses serviços poderia passar por um mecanismo de *updates* que permitisse atualizar os vários componentes, quer do servidor quer do cliente. Neste momento este sistema não inclui um sistema de encriptação e ainda não está preparado para ser utilizado num cenário real. Assim é importante implementar futuramente um mecanismo de proteja os dados do sistema. Dado que este estará em contacto com a internet e até conecta servidores entre si, é importante implementar um mecanismo que proteja a privacidade da informação que transita no sistema. Posteriormente poderia ser implementado um sistema de *add-ons*, onde as aplicações e serviços fossem modulares e pudessem ser instalados no servidor, como extensões adicionais. Os clientes que se conectarem ao servidor iriam recolher os serviços existentes no mesmo e adaptar-se para poderem interagir com os mesmos. Um serviço interessante de se implementar poderia passar por permitir aos utilizadores criarem salas de cinema e salas de rádio. Essencialmente os utilizadores iriam conectar-se a uma sala de conversação onde poderiam visualizar o mesmo filme ou ouvir a mesma música e comentar o mesmo por meio de mensagens ou voz. Outra funcionalidade interessante que se poderia implementar considera a utilização do sistema como servidor de domótica. Utilizando um programa como *OpenRemote*, o servidor poderia interagir com *gateways* de standards de domótica, como *KNX*, *Z-Wave* ou *Insteon.* O cliente poderia proporcionar uma interface que apresentasse os dispositivos de domótica que o servidor detetasse, e permitisse visualizar o seu estado e interagir com os mesmos. Por fim, seria também interessante adaptar o cliente para outras plataformas ainda não suportadas como *Mac OS* e para plataformas móveis como *iOS* e *Android*.

Concluindo, este trabalho permitiu comprovar que este conceito de conjugar numa solução local, um conjunto de serviços normalmente oferecidos na internet e tem potencial para ser uma solução abrangente e que vai de encontro às tendências da atualidade. Conjugando a utilidade do acesso remoto das *clouds* com a flexibilidade do *Qt* e do *XMPP*, permite não só oferecer os serviços já disponíveis, remotamente aos utilizadores, mas também criar novos serviços que vão de encontro às tendências da indústria bem como da sociedade.

Referências Documentais

[1] CISCO, “Forecast and Methodology, 2014–2019,” *Cisco Visual Networking Index,* pp. 8-12.

[2] C. Liu, G. Yucheng, Y. Luo, W. Jiang, X. Hu, X. Wu, X. Meng, Y. Chen, J. Zhang, Y. Jiao, Z. Chen e P. Liu, “The Research of audio and video public network platform based on CDN and P2P.”.*2012 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science.*

[3] C. Zhao, J. Zhao, X. Lin e C. Wu, “Capacity of P2P On-Demand Streaming with Simple, Robust and Decentralized Control”.*2013 Proceedings IEEE INFOCOM.*

[4] F. Liu, B. Li, B. Li e H. Jin, “Peer-Assisted On-Demand Streaming: Characterizing Demands and Optimizing Supplies”.*IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 2, FEBRUARY 2013.*

[5] B. Tan e L. Massoulié, “Optimal Content Placement for Peer-to-Peer Video-on-Demand Systems”.*IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 21, NO. 2, APRIL 2013.*

[6] J. You, X. Li, J. Song e J. Wang, “A Feedback Control Model based Hierarchical VoD System,” Chinese Academy of Sciences, China.

[7] “CSA Cloud Security Alliance,” [Online]. Available: https://cloudsecurityalliance.org/media/news/cloud-maturity-study-reveals-top-issues. [Acedido em 20 10 2015].

[8] Y. Jadeja e K. Modi, “Cloud Computing - Concepts, Architecture and Challenges”.*2012 International Conference on Computing, Electronics and Electrical Technologies [ICCEET]*.

[9] Y. Amanatullah, C. Lim, H. Purnomo Ipung e A. Juliandri, “Toward Cloud Computing Reference Architecture: Cloud Service Management Perspective”.

[10] S. A. Z. Hassan, “STAR: A Proposed Architecture for Cloud Computing Applications”.*Proceedings of 2012 1ntemational of Cloud Computing, Technologies, Applications & Management.*

[11] F. Yu, Y.-w. Wan e R.-h. Tsaih, “Quantitative Analysis of Cloud-based Streaming Services”.*2013 IEEE 10th International Conference on Services Computing.*

[12] W. Zeng, J. Zhao, M. Liu, J. Zhao e M. Liu, “Several Public Commercial Clouds and Open Source Cloud Computing Software”.*The 7th International Conference on Computer Science & Education (ICCSE 2012) Several Public Commercial Clouds and Open Source.*

[13] A. Weiss, “eSecurity Planet,” 12 03 2013. [Online]. Available: http://www.esecurityplanet.com/network-security/cloud-security-standards-what-you-should-know.html.

[14] G. Kulkarni, J. Gambhir, T. Patil e A. Dongare, “A Security Aspects in Cloud Computing”.

[15] D. Linthicum, “InfoWorld,” 25 Junho 2013. [Online]. Available: http://www.infoworld.com/article/2611402/cloud-computing/cloud-adoption-s-tipping-point-has-arrived.html.

[16] A. Behl e K. Behl, “An Analysis of Cloud Computing Security Issues”.*2012 World Congress on Information and Communication Technologies.*

[17] A. Behl, “Emerging Security Challenges in Cloud Computing. An insight to Cloud security challenges and their mitigation”.*2011 World Congress on Information and Communication Technologies.*

[18] D. Chen e H. Zhao, “Data Security and Privacy Protection Issues in Cloud Computing”.*2012 International Conference on Computer Science and Electronics Engineering.*

[19] A. Abuhussein, H. Bedi e S. Shiva, “Evaluating Security and Privacy in Cloud Computing Services: A Stakeholder's Perspective”.*The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012).*

[20] F. B. Shaikh e S. Haider, “Security Threats in Cloud Computing”.*6th International Conference on Internet Technology and Secured Transactions, 11-14 December 2011, Abu Dhabi, United Arab Emirates.*

[21] W. Liu, “Research on Cloud Computing Security Problem and Strategy,” Department of Computer and Information Engineering, Wuhan Polytechnic University, Wuhan Hubei Province 430023, China.

[22] U. Somani, K. Lakhani e M. Mundra, “Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing”.*2010 1st International Conference on Parallel, Distributed and Grid Computing (PDGC - 2010).*

[23] F. Sabahi, “Cloud Computing Security Threats and Responses,” Faculty of Computer Engineering, Azad University, Iran.

[24] H. Qi e A. Gani, “Research on Mobile Cloud Computing: Review, Trend and Perspectives,” Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia.

[25] P. A. Kotwal e A. R. Singh, “Evolution and Effects of Mobile Cloud Computing, Middleware Services on Cloud, Future Prospects: A peek into the Mobile Cloud Operating Systems”.*2012 IEEE International Conference on Computational Intelligence and Computing Research.*

[26] Y. Wang, X. Lin e M. Pedram, “A Nested Two Stage Game-Based Optimization Framework in Mobile Cloud Computing System”.*2013 IEEE Seventh International Symposium on Service-Oriented System Engineering.*

[27] L. Yang, J. Cao, S. Tang, T. Li e A. T. S. Chan, “A Framework for Partitioning and Execution of Data Stream Applications in Mobile Cloud Computing”.*2012 IEEE Fifth International Conference on Cloud Computing.*

[28] A. Kazi, R. Kazi e R. Deters, “Supporting the Personal Cloud. From Mobile Cloud Computing (MCC) to Ubiquitous Cloud Computing (UCC)”.*2012 IEEE Asia Pacific Cloud Computing Congress (APCloudCC).*

[29] R. Kazi, X. Zhang e R. Deters, “Supporting Apps in the Personal Cloud. Using WebSockets within Hybrid Apps”.*2012 IEEE Second Symposium on Network Cloud Computing and Applications.*

[30] S. Dhumbumroong e K. Piromsopa, “Personal Cloud Filesystem: A Distributed Unification Filesystem for Personal Computer and Portable Device”.*2011 Eighth International Joint Conference on Computer Science and Software Engineering (JCSSE).*

[31] P. Casas, H. R. Fischer, S. Suette e i. Schatz, “A First Look at Quality of Experience in Personal Cloud Storage Services”.*IEEE International Conference on Communications 2013: IEEE ICC'13 - 1st International Workshop on Mobile Cloud Networking nternational Workshop on Mobile Cloud Networking.*

[32] Y. Kim, J. An e Y.-H. Lee, “CCNFRR: Fast one-hop Re-Route in CCN”.*International Workshop on the Network of the Future.*

[33] S. Ding, Z. Chen e Z. Liu, “Parallelizing FIB Lookup in Content Centric Networking”.*2012 Third International Conference on Networking and Distributed Computing.*

[34] J. Kim, M.-W. Jang, J. Park, S. Choi e B.-J. (. Lee, “Enhanced Forwarding Engine for Content-Centric Networking (CCN)”.*2013 IEEE International Conference on Consumer Electronics (ICCE).*

[35] S. Braun, M. Monti, M. Sifalakis e C. Tschudin, “An Empirical Study of Receiver-based AIMD Flow-Control Strategies for CCN,” Department of Mathematics and Computer Science, University of Basel, Switzerland.

[36] H. Wang, Z. Chen, F. Xie e F. Han, “A Data Structure for Content Cache Management in Content-Centric Networking”.*2012 Third International Conference on Networking and Distributed Computing.*

[37] J. M. Wang e B. Bensaou, “Progressive Caching in CCN”.*Globecom 2012 - Next Generation Networking and Internet Symposium.*

[38] Y. Li, T. Lin, H. Tang e P. Sun, “A Chunk Caching Location and Searching Scheme in Content Centric Networking”.*IEEE ICC 2012 - Next-Generation Networking Symposium.*

[39] N. Choi, K. Guan, D. C. Kilper e G. Atkinson, “In-Network Caching Effect on Optimal Energy Consumption in Content-Centric Networking”.*IEEE ICC 2012 - Next-Generation Networking Symposium.*

[40] M. Xie, I. Widjaja e H. Wang, “Enhancing Cache Robustness for Content-Centric Networking”.*2012 Proceedings IEEE INFOCOM.*

[41] S. Choi, K. Kim, S. Kim e B.-h. Roh, “Threat of DoS by Interest Flooding Attack in Content-Centric Networking,” Department of Computer Engineering, Graduate School of Ajou University, Suwon, Republic of Korea.

[42] J. Lee, D. Kim, M. Jang e B.-j. (. Lee, “Mobility Management for Mobile Consumer Devices in Content Centric Networking (CCN)”.*2012 IEEE International Conference on Consumer Electronics (ICCE).*

[43] J. Lee, D. Kim, M.-W. Jang e B.-J. (. Lee, “Proxy-based Mobility Management Scheme in Mobile Content Centric Networking (CCN) environments”.*2011 IEEE International Conference on Consumer Electronics (ICCE).*

[44] J. Lee e D. Kim, “Proxy-assisted Content Sharing Using Content Centric Networking (CCN) for Resource-limited Mobile Consumer Devices”.

[45] J. Wang, W. Li e S. Ma, “A Convergence Service Oriented Architecture in Smart Homes,” State Grid Information & Telecommunication Co., Ltd, Beijing, China.

[46] X. Ye e J. Huang, “A Framework for Cloud-based Smart Home”.*2011 International Conference on Computer Science and Network Technology.*

[47] A. A. Khan e H. T. Mouftah, “SECURED WEB SERVICES FOR HOME AUTOMATION IN SMART GRID ENVIRONMENT”.*2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE).*

[48] A. Arabo, I. Brown e F. El-Moussa, “Privacy in the age of Mobility and Smart Devices in Smart Homes”.*2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust.*

[49] M. Hager, S. Schellenberg, J. Seitz, S. Mann e G. Schorcht, “Secure and QoS-aware Communications for Smart Home Services”.

[50] W.-K. Park, C.-s. Choi e J. Jang, “Energy Efficient Multi-function Home Gateway in Always-On Home Environment”.

[51] W.-K. Park, C.-s. Choi, I.-w. Lee e J. Jang, “Energy Efficient Multi-Function Home Gateway in Always-On Home Environment”.*IEEE Transactions on Consumer Electronics, Vol. 56, No. 1, FEBRUARY 2010.*

[52] J.-P. Gelas, L. Lefevre, T. Assefa e M. Libsie, “Virtualizing Home Gateways for Large Scale Energy Reduction in Wireline Networks”.

[53] A. Pal, C. Bhaumik, J. Shukla e S. Kolay, “Energy Information Gateway for Home”.*2011 Second International Conference on Intelligent Systems, Modelling and Simulation.*

[54] N. Saito, “Ecological Home Network: An Overview”.*Proceedings of the IEEE | Vol. 101, No. 11, November 2013.*

[55] “Innovation Series,” 24 May 2013. [Online]. Available: http://www.innovation-series.com/2013/05/24/streaming-media-opportunities-and-challenges-in-2013.

[56] B. Wagner e C. Gonsalves, “TMCnet,” [Online]. Available: http://www.tmcnet.com/voip/0706/featurearticle-driving-IPTV-growth.htm. [Acedido em 20 10 2015].

[57] W. Ji, Y. Liu e W. Cui, “The Design of Home Gateway Which Used In FTTH,” 2010 International Conference on Networking and Digital Society.

[58] M.-J. Lee, J.-Y. Oh e S.-J. Kang, “Design of Multimedia Stream Channel Arbiter in Home Network Gateway”.*IEEE Transactions on Consumer Electronics, Vol. 57, No. 4, November 2011.*

[59] J. PARK, J. KIM, S. KIM e D. NAM, “An Intra-Gateway for Entertainment Contents in Digital Home”.*Feb. 13~16, 2011 ICACT2011.*

[60] “UNIVERSAlbite,” 4 SETEMBRO 2012. [Online]. Available: http://universalbite.com/1171/people-consume-content-multiple-devices-day.

[61] Google, “The New Multi-screen World: Understanding Cross-platform User Behavior,” 2012.

[62] E. S. Ryu e N. Jayant, “Architecture of a Home Gateway for Three-Screen TV”.*2011 IEEE International Conference on Consumer Electronics (ICCE).*

[63] E.-S. Ryu e N. Jayant, “Home Gateway for Three-Screen TV Using H.264 SVC and Raptor FEC”.*IEEE Transactions on Consumer Electronics, Vol. 57, No. 4, November 2011.*

[64] S. Ying, K.-h. Lee e J.-h. Lee, “IPTV Accessing Approach With IMS Home Gateway,” Department of Electronic Telecommunication Engineering, Inje University, Gimhae, S.Korea.

[65] P.-S. Kim e S. H. Ahn, “A Home-oriented IPTV Service Platform on Residential Gateway,” Department of Electronic Engineering, Korea Polytechnic University, Shiheung-City, Kyunggi-Do, 429-793, KOREA.

[66] V. Lucena, N. Viana, O. Maia, J. Filho e W. Silva, “Designing an Extension API for Bridging Ginga iDTV Applications and Home Services”.

[67] K. J. Patel, S. V. Anand e S. K. S.P, “A Robust QoS framework on Android for Effective media delivery to DLNA Enabled Home Gateway in Smart Home Environment”.

[68] M. Zhanikeev, “A Home Gateway Box with Meter, Probe and L2 QoS Policy Edge”.*2013 IEEE 37th Annual Computer Software and Applications Conference Workshops.*

[69] J.-h. zheng, Y. wang e W.-r. tan, “AN Adaptive gateway for smart home”.*2013 International Conference on Computational and Information Sciences.*

[70] M. Jung, G. Kienesberger, W. Granzer, M. Unger e W. Kastner, “Privacy enabled Web service access control using SAML and XACML for home automation gateways”.*6th International Conference on Internet Technology and Secured Transactions, 11-14 December 2011, Abu Dhabi, United Arab Emirates.*

[71] M. Z. Bjelica, B. Mrazovac, N. Teslic, I. Papp e D. Stefanovic, “Cloud-Enabled Home Automatio n Gateway with the Support for UPnP Over IPv4/IPv6 and 6LoWPAN”.*2012 IEEE International Conference on Consumer Electronics (ICCE).*

[72] IBM, “Look to the cloud to save money and build business,” 2010.

[73] “eBiz|MBA,” Novembro 2015. [Online]. Available: http://www.ebizmba.com/articles/file-sharing-websites. [Acedido em 20 10 2015].

[74] “statista,” [Online]. Available: http://www.statista.com/statistics/261820/number-of-registered-dropbox-users. [Acedido em 20 10 2015].

[75] F. Y. Rashid, “pcmag,” 6 Agosto 2015. [Online]. Available: http://www.pcmag.com/article2/0,2817,2489189,00.asp.

[76] C. Page, “theINQUIRER,” 6 Agosto 2015. [Online]. Available: http://www.theinquirer.net/inquirer/news/2421076/dropbox-and-onedrive-at-risk-from-malware-injecting-man-in-the-cloud-attacks.

[77] E. Klinker, “BitTorrent Blog,” [Online]. Available: http://blog.bittorrent.com/2013/12/05/bittorrent-sync-hits-2-million-user-mark/. [Acedido em 5 Dezembro 2013].

[78] “ASUS,” [Online]. Available: event.asus.com/server/tsmini. [Acedido em 20 10 2015].

[79] “WD,” [Online]. Available: http://www.wdc.com/pt/products/products.aspx?id=1140. [Acedido em 20 10 2015].

[80] “D-Link | ShareCenter,” [Online]. Available: http://sharecenter.dlink.com/. [Acedido em 20 10 2015].

[81] D. Meyer, “GIGAOM RESEARCH,” 29 Abril 2013. [Online]. Available: https://gigaom.com/2013/04/29/chat-apps-have-overtaken-sms-by-message-volume.

[82] The Radicati Group, Inc, “Instant Messaging Statistics Report, 2015-2019,” 2015.

[83] “w3,” [Online]. Available: http://www.w3.org/Protocols/rfc959. [Acedido em 20 10 2015].

[84] “rsync,” [Online]. Available: https://rsync.samba.org. [Acedido em 20 10 2015].

[85] “ComputerHope,” [Online]. Available: http://www.computerhope.com/unix/scp.htm. [Acedido em 20 10 2015].

[86] “Microsoft | Developer Network,” [Online]. Available: https://msdn.microsoft.com/en-us/library/ee442092.aspx. [Acedido em 20 10 2015].

[87] “Samba,” [Online]. Available: https://www.samba.org/samba/docs. [Acedido em 20 10 2015].

[88] “irchelp,” [Online]. Available: http://irchelp.org/irchelp/rfc/rfc.html. [Acedido em 20 10 2015].

[89] “Mumble,” [Online]. Available: http://wiki.mumble.info/wiki/Main_Page. [Acedido em 20 10 2015].

[90] “ietf,” [Online]. Available: https://www.ietf.org/rfc/rfc3261.txt. [Acedido em 20 10 2015].

[91] “Xmpp Standards Foundation,” [Online]. Available: http://xmpp.org/xmpp-protocols/xmpp-extensions. [Acedido em 20 10 2015].

[92] “Logitech Media Server,” [Online]. Available: http://www.mysqueezebox.com/download. [Acedido em 20 10 2015].

[93] “Kodi,” [Online]. Available: http://kodi.tv/. [Acedido em 20 10 2015].

[94] “Ps3mediaserver,” [Online]. Available: http://www.ps3mediaserver.org. [Acedido em 20 10 2015].

[95] “VideoLAN Organization,” [Online]. Available: http://www.videolan.org/vlc/features.html. [Acedido em 20 10 2015].

[96] “VideoLAN Organization,” [Online]. Available: http://www.videolan.org/vlc/. [Acedido em 20 10 2015].

[97] “FFmpeg,” [Online]. Available: https://www.ffmpeg.org/documentation.html. [Acedido em 20 10 2015].

[98] “Process One,” [Online]. Available: https://www.ejabberd.im/protocols. [Acedido em 20 10 2015].

[99] “Erlang Solutions,” [Online]. Available: https://www.erlang-solutions.com/products/mongooseim-massively-scalable-ejabberd-platform. [Acedido em 20 10 2015].

[100] “Jabberd,” [Online]. Available: http://jabberd2.org. [Acedido em 20 10 2015].

[101] “Prosody IM,” [Online]. Available: https://prosody.im. [Acedido em 20 10 2015].

[102] “LW.org,” [Online]. Available: http://www.lightwitch.org/metronome. [Acedido em 20 10 2015].

[103] “ejabberd,” [Online]. Available: https://www.ejabberd.im/jabber.org-migrated. [Acedido em 20 10 2015].

[104] M. Redmond, “process-one,” 23 Febreiro 2014. [Online]. Available: https://blog.process-one.net/whatsapp-facebook-erlang-and-realtime-messaging-it-all-started-with-ejabberd.

[105] Reaserch2guidance, “Cross Platform App Development Tool Benchmarking 2013,” 2013.

[106] “Qt,” Version:5.5, [Online]. Available: http://www.qt.io/.

[107] “QXmpp,” Version:0.9.1, [Online]. Available: http://doc.qxmpp.org/qxmpp-0.9.1.

[108] “Qt,” Version:5.5, [Online]. Available: http://doc.qt.io/qt-5/signalsandslots.html.

[109] “Qt,” Version:5.5, [Online]. Available: http://doc.qt.io/qt-5/qtquick-index.html.

[110] “QXmpp,” Version:0.9.1, [Online]. Available: http://doc.qxmpp.org/qxmpp-0.9.1/classQXmppDiscoveryManager.html.

[111] “XMPP Standards Foundation,” Version:1.25, [Online]. Available: http://xmpp.org/extensions/xep-0045.html#disco-rooms.

[112] “XMPP Standards Foundation,” Version:1.1, [Online]. Available: http://xmpp.org/extensions/xep-0095.html.

[113] “XMPP Standards Foundation,” Version:1.2, [Online]. Available: http://xmpp.org/extensions/xep-0096.html.

[114] “XMPP Standards Foundation,” Version:1.8.1, [Online]. Available: http://xmpp.org/extensions/xep-0065.html.

[115] “XMPP Standards Foundation,” Version:1.1, [Online]. Available: http://xmpp.org/extensions/xep-0166.html.

[116] “XMPP Standards Foundation,” Version:1.1, [Online]. Available: http://xmpp.org/extensions/xep-0167.html.

[117] “XMPP Standards Foundation,” Version:1.0, [Online]. Available: http://xmpp.org/extensions/xep-0176.html.

[118] “Qt,” Version:5.5, [Online]. Available: http://doc.qt.io/qt-5/qlistview.html.

[119] “Qt,” Version:5.5, [Online]. Available: http://doc.qt.io/qt-5/qml-qtqml-models-listmodel.html.

[120] “XMPP Standards Foundation,” Version:1.13, [Online]. Available: http://www.xmpp.org/extensions/xep-0060.html.

[121] “ejabberd Docs,” [Online]. Available: http://docs.ejabberd.im/developer/modules. [Acedido em 20 10 2015].

[122] P. McLean, “Apple Insider,” 28 Março 2008. [Online]. Available: http://appleinsider.com/articles/08/03/28/exploring_time_capsule_theoretical_speed_vs_practical_throughput.

[123] M. King, “Michael King,” 11 Junho 2013. [Online]. Available: http://www.mpking.com/2013/06/theoretical-speed-vs-practical.html.

[124] “Qt,” Version:5.5, [Online]. Available: https://wiki.qt.io/Qt_5.5.0_Multimedia_Backends.

[125] “WhatsApp,” [Online]. Available: http://www.whatsapp.com/faq/en/android/23112542. [Acedido em 20 10 2015].

[126] “Messenger,” [Online]. Available: https://www.messenger.com/features. [Acedido em 20 10 2015].

[127] T. Suel e N. Memon, “Algorithms for Delta Compression and Remote File Synchronization,” CIS Department, Polytechnic University, Brooklyn, NY 11201.