Mestrado em Engenharia Informática
Estágio
Relatório Final

Combinação de Mundos Reais e Mundos Virtuais – caso de estudo SAPO.Labs

David João Costa Silva
djsilva@student.dei.uc.pt

Orientador:
Jorge Sá Silva
Data: 30 de Junho de 2013

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Resumo

Este trabalho insere-se num projecto de cooperação entre o DEI/UC e o SAPO.Labs e tem como objectivo desenhar, projectar e implementar uma plataforma inovadora - HappyHour - que permitirá aos estabelecimentos comerciais, de forma simples e imediata, divulgar os seus produtos, campanhas e eventos a potenciais clientes que se encontrem nas imediações do seu estabelecimento. Para isso pretende-se criar micro-comunidades cruzando informação de geo-localização de utilizadores e os seus perfis de redes sociais.

A lógica de negócio do HappyHour passa por ir mais longe que aplicações que cruzam dados entre comunidades das redes sociais reunindo numa base de dados centralizada a geo-localização de utilizadores que instalem a aplicação HappyHour nos seus dispositivos móveis. Esta base de dados irá permitir cruzar a informação geo-espacial com os gostos partilhados por estes utilizadores, reconhecendo potenciais clientes nas imediações aos quais os estabelecimentos comerciais poderão divulgar campanhas promocionais e eventos através de notificações em tempo real. Por outro lado o utilizador pode saber que tipo de música está a passar num determinado espaço nocturno, vai ouvir num evento ou o nome da música que ouviu num determinado momento, tendo neste caso a hipótese de a adquirir através da aplicação SAPO.MusicBox.

Palavras-Chave

"Redes sociais", "Geo-localização", "Aplicações móveis", "Vida nocturna", "Marketing directo".

Índice

Anexos

Lista de Figuras

Glossário

Acrónimo	Significado
API	Application Programming Interface
DAO	Data access object
DEI	Departamento de Engenharia Informática
DJ	Disk Jockey
FCTUC	Faculdade de Ciências e Tecnologia da Universidade de Coimbra
GCM	Google Cloud Messaging
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
JSONP	JavaScript Object Notation with padding
LCT	Laboratório de Comunicações e Telemática da Universidade de Coimbra
PHP	Hypertext Preprocessor
PNG	Portable Network Graphics
POI	Point of Interest
REST	Representational state transfer
SMS	Short Message Service
SVN	Apache Subversion
WSN	Wireless Sensor Networks

Capítulo 1
Introdução

1.1. Contextualização

Este estágio é parte integrante da cadeira de Dissertação/Estágio pertencente ao Mestrado em Engenharia Informática e foi realizado no Departamento de Engenharia Informática da FCTUC no âmbito do projecto HappyHour com a equipa do LCT-WSN em colaboração com o SAPO.Labs. O projecto HappyHour tem como ponto de partida um protótipo elaborado previamente chamado Youth5G [1] desenvolvido no âmbito de outro projecto pelo aluno Duarte Raposo e vai ser desenvolvido com o apoio do aluno de doutoramento David Nunes.

1.2. Objectivos

Este projecto pretende partir do protótipo Youth5G para proporcionar aos utilizadores uma forma de se manterem informados sobre o que os estabelecimentos de diversão nocturna e os seus eventos têm para lhes oferecer. Ao terem a aplicação HappyHour instalada nos seus dispositivos móveis, os utilizadores recebem informações em tempo real sobre quais as promoções, ofertas ou eventos que irão ocorrer nos estabelecimentos ao seu redor.

A aplicação HappyHour procura também, através do paradigma Internet of Things [2], oferecer ao utilizador a possibilidade de saber em tempo real qual o grau de afluência de cada espaço de diversão. Paralelamente permitirá dar a conhecer ao utilizador o nível de agitação do local, como por exemplo se o local é um bar muito agitado onde as pessoas se encontram a dançar ou, se pelo contrário, é um bar mais calmo onde é possível conversar.

Caso o utilizador pretenda pode também saber antecipadamente qual o tipo de música que o espera num determinado evento ou espaço, consultando a *playlist* associada. Através desta funcionalidade podem também ser consultadas as músicas que foram ouvidas numa determinada data e hora. É comum cada utilizador associar momentos bem passados e com valor emotivo às músicas que escutou num determinado momento ou local e para as quais, em muitas situações, desconhece o autor e o nome das mesmas. Assim, poderá tirar proveito dos registos do HappyHour para encontrar a música que procura e poder adquiri-la através da integração com a aplicação SAPO.MusicBox.

Todas estas informações disponibilizadas possibilitarão ao utilizador escolher o local de acordo com a sua disposição e evitar escolhas que não se coadunem com os seus gostos ou não se adequem ao momento.

Esta plataforma pretende também disponibilizar aos estabelecimentos um modelo de negócio inovador e inexistente no mercado, permitindo divulgar as suas promoções e actividades em tempo real. Estas informações são enviadas de uma forma simples e direccionada através de "*push-notifications*" a todos os potenciais clientes que se encontrem nas imediações do local. Esta abordagem permite aos estabelecimentos comerciais implementar políticas inovadoras de marketing com o convite à participação activa dos utilizadores nas actividades do local, e que pode ser incentivada com prémios, brindes ou descontos.

Embora o modelo de aplicação inicial tenha sido desenvolvido para estabelecimentos de diversão nocturna, este poderá ser alargado a outras áreas de negócio como a restauração ou espaços comerciais.

1.3. Método de Trabalho

Depois de tomar conhecimento do projecto foi elaborado, em conjunto com o colega Eng.º David Nunes, um plano de trabalho que detalha as tarefas a serem realizadas até ao final do projecto bem como a sua duração, e que foi posteriormente aprovado pelo orientador, o professor Jorge Sá Silva.

Neste projecto foi adoptada uma política de reuniões semanais da equipa com o orientador. Todas as segundas-feiras é feito um ponto de situação do trabalho realizado na semana anterior onde são apresentadas todas as dificuldades e obstáculos que foram encontrados e é também discutido o planeamento dos objectivos para a semana seguinte. Estas pequenas reuniões têm uma duração de 15 a 20 minutos e são uma ferramenta importante na avaliação do progresso das tarefas e do projecto em geral de forma a perceber se é necessário acelerar o ritmo de trabalho e definir novas estratégias. Para além destas reuniões fomos acompanhados pela equipa do SAPO.Labs tendo tido duas reuniões para verificação do progresso, uma no dia 11 de Dezembro de 2012 e a segunda e última no dia 30 de Maio de 2013.

A nível do desenvolvimento foi utilizada uma metodologia que produza documentação sempre que haja alterações ao "*software*" de forma a assegurar que, quem futuramente necessitar de interpretar o código fonte, consiga compreender facilmente o funcionamento da aplicação e entender as motivações por trás das decisões tomadas. Assim sendo, antes de qualquer alteração, é criado um documento de requisitos do novo módulo a ser implementado e são promovidas as necessárias alterações aos diagramas de classes e modelos físicos dos módulos já implementados.

No final da implementação de cada alteração à aplicação são executados testes às funcionalidades previamente existentes de forma a garantir que estas não foram afectadas.

Ao longo do projecto novos elementos entraram para a equipa para se focarem no desenvolvimento da aplicação Android. Para a integração destes novos elementos foi necessário elaborar um novo plano de trabalho em conjunto com o Eng.º David Nunes e, ficámos ambos como responsáveis pelo acompanhamento do trabalho desenvolvido por eles. O aluno de Licenciatura João Pratas ficou encarregue do sistema de notificação enquanto os alunos de Mestrado Bruno Antunes e Elói Almeida ficaram responsáveis por tratar os dados de posição e agitação. Seguindo a mesma metodologia do resto do projecto foi adoptado um sistema de reuniões semanais todas as segundas-feiras para seguimento do progresso das tarefas definidas.

1.4. Visão Geral do Relatório

De forma a facilitar a leitura do presente documento foi feita uma divisão em cinco capítulos cujo conteúdo é aqui descrito.

No primeiro capítulo - Introdução, é feita uma contextualização do projecto e da equipa, são enumerados os objectivos que se pretendem atingir, é definido o método de trabalho em que este é desenvolvido e são referidas todas as tecnologias que foram utilizadas.

O estado da arte está desenvolvido no capítulo 2 em que são analisadas e comparadas diversas aplicações concorrentes a este projecto bem como serviços que já deixaram de estar activos mas que podem fornecer ideias interessantes. É também apresentado um comparativo entre APIs de pontos de interesse, bem como uma análise às tecnologias escolhidas para o desenvolvimento.

No terceiro capítulo está descrita a arquitectura geral e específica de cada módulo da aplicação HappyHour.

O quarto capítulo engloba o estado do protótipo Youth5G no momento em que me juntei ao projecto bem como todo o trabalho desenvolvido ao longo deste ano. No final deste capítulo encontra-se uma referência aos desafios encontrados até agora.

O planeamento de todo o projecto é apresentado no quinto capítulo com um comentário no final com a justificação e o impacto que os desvios tiveram no projecto.

Por fim, no último capítulo são apresentadas as conclusões tiradas após este ano de estágio bem como planos futuros para a aplicação.

Anexados ao documento podem ser encontrados o relatório de requisitos da aplicação HappyHour Management (anexo A) e o modelo físico da base de dados (anexo B).

Capítulo 2
Estado da Arte

Neste capítulo são enumeradas aplicações com algum tipo de semelhança ao serviço HappyHour. Algumas destas aplicações podem ser vistas como concorrentes, mas outras podem até levar a possíveis parcerias ou pontos de integração. Neste capítulo é feita uma pequena descrição sobre cada aplicação e quais os seus objectivos.

2.1. Análise Concorrencial

2.1.1. Concorrentes Activos

2.1.1.1. Foursquare

O Foursquare [3] é uma rede social com foco em dispositivos móveis que incentiva os seus os utilizadores a partilhar a sua localização com terceiros. A rede social é incrivelmente popular, contando, desde Setembro de 2012, com cerca de 20 milhões de utilizadores em todo o mundo [4]. Estes resultados são obtidos devido a uma forte presença em dispositivos móveis tendo aplicações disponíveis para iOS, Android, Windows Mobile, Blackberry e Symbian. O Foursquare suporta várias funcionalidades que estimulam os utilizadores a partilhar os seus dados. Os utilizadores podem fazer "*check-in*" em locais de interesse ou estabelecimentos a partir da própria aplicação que detecta a sua posição através de GPS ou através de localização baseada em redes Wi-Fi ou móveis. Com cada "*check-in*", o utilizador é recompensado com "pontos" e por vezes "crachás" que ficam associados ao seu perfil e que incentivam a contínua utilização da aplicação. De acordo com os dados disponibilizados pela empresa, em Setembro de 2012 o Foursquare contava com mais de 3 mil milhões check-ins [4]. Os "crachás" podem ser ganhos por realizar certas acções num local, pela frequência de "check-ins" de um local ou por efectuar "check-ins" em determinadas alturas. A aplicação suporta também o envio de notificações "push-notifications" entre utilizadores (chamados de "*Pings*"), bem como a criação de "*Tips*", sugestões do que ver, fazer ou comer naquele local em particular. Outra funcionalidade interessante denomina-se de "*Mayorship*": quando um utilizador faz "*check-in*" num determinado local mais do que qualquer outra pessoa nos últimos 60 dias, torna-se no "Mayor" desse local. Isto incentiva os utilizadores a "fidelizarem-se" a locais que apreciem. O Foursquare permite também a pesquisa de contactos que se encontrem perto do utilizador, bem como saber a localização dos utilizadores que estejam na sua lista de amigos.

2.1.1.2. Facebook Nearby

O Facebook [5], maior rede social a nível mundial, começou em Dezembro de 2012 a incluir um serviço semelhante ao Foursquare [6] na sua plataforma. O separador "*nearby*" presente nas aplicações móveis do Facebook (iOS, Blackberry, Android, Windows Phone, Nokia e telemóveis com suporte java [7]), que antes continha apenas a funcionalidade de "*check-in*", passou a disponibilizar avaliações, recomendações e informações de estabelecimentos comerciais fornecidas pelos seus utilizadores. Os utilizadores podem agora saber onde os seus amigos estiveram e se gostaram de lá estar, bem como que locais de interesse possam existir nas imediações. Estas funcionalidades, em conjunto com a gestão de eventos já existente, tornam-se um adversário de peso para os serviços semelhantes, principalmente devido à grande base de utilizadores que já marca presença nesta rede social.

No final de 2012, dos cerca de 1000 milhões [8] de utilizadores do Facebook, 250 milhões [9] já utilizavam as suas funcionalidades de localização. Ao observarmos que 600 milhões [10] de pessoas utilizam a aplicação móvel para aceder ao Facebook, e com a integração de localizadores GPS a crescer nestes tipos de dispostitos, prevê-se um crescimento do número de utilizadores deste serviço.

2.1.1.3. Waze

O Waze [11] é uma aplicação que permite realizar uma gestão de tráfego automóvel e oferece um sistema de navegação com base em informação colaborativa, isto é, baseada no feedback dos vários utilizadores do sistema. Assim sendo, não é propriamente uma aplicação rival ao HappyHour, mas poderá ser um possível ponto de integração no futuro.

O Waze aprende através dos tempos de condução dos utilizadores com o objectivo de fornecer rotas e informações de tráfego em tempo real. Para isto, os utilizadores podem inserir informações sobre o trânsito, disponibilizar informação sobre o estado das vias, traçar rotas até ao local desejado, indicar pontos de abastecimento económicos ou mesmo editar os mapas. A aplicação também permite coordenar encontros e boleias com amigos, através da partilha de localização e de tempos estimados de viagem. Devido a estas funcionalidades, o principal incentivo que esta aplicação fornece ao utilizador é, assim, a possibilidade de traçar rotas através dos dados disponibilizados por outros utilizadores, de forma colaborativa, que ao contrário das aplicações de navegação tradicionais unicamente utilizam dados estáticos de navegação.

2.1.1.4. GetGlue

O GetGlue [12], contrariamente às aplicações anteriores, é uma rede social construída através dos gostos culturais [13]. Cada utilizador é incentivado a partilhar os seus gostos desportivos, televisivos e cinematográficos, indicando à aplicação o que está a ver e recebendo a partir desse momento informações sobre esse programa. Consoante os gostos do utilizador e dos seus amigos são sugeridas recomendações de novos conteúdos que podem interessar-lhe. Ao atingir certos objectivos o utilizador é recompensado com autocolantes virtuais ou descontos em empresas de entretenimento. Uma vez por mês pode ser pedida a impressão e o envio destes autocolantes para casa tornando a experiencia virtual numa experiência real.

O GetGlue encontra-se em crescimento tendo uma plataforma de 3 milhões de utilizadores que, com a fusão com a Viggle, se prevê que passe para os 4,5 milhões [14]. Para além da presença nos mais utilizados sistemas operativos móveis (IOS e Android) possui uma aplicação integrada nas set-top-boxes do operador de televisão DirecTV [15] permitindo aos utilizadores fazerem "*check-in*" dos programas que estão a ver. Nas aplicações móveis existe uma integração com o Foursquare [16] que permite partilhar o que se está a ver e onde se está a ver. Esta integração com o Foursquare tem como objectivo potenciar a acumulação de pontos e prémios em ambas as redes sociais e assim atrair mais utilizadores.

2.1.1.5. Yelp

O Yelp é um directório de serviços que o utilizador pode pesquisar tendo acesso a detalhes de funcionamento e à descrição dos estabelecimentos numa cidade [17] das 195 disponíveis [18]. A grande vantagem em relação aos restantes directórios de serviço é apostar numa vertente de rede social para os seus utilizadores.

Os utilizadores contribuem classificando a sua experiência no local e partilhando uma breve opinião. Para além disso podem adicionar detalhes ou informações extra ao perfil do estabelecimento (informações essas que têm de ser aprovadas por moderadores). De forma a motivar a partilha de informação, as melhores opiniões são colocadas em destaque, bem como a primeira a ser escrita, criando uma competição entre os utilizadores e dando crachás a quem atingir certos objectivos [19]. Para além das contribuições e classificações, os utilizadores podem seguir os seus amigos tendo acesso às opiniões que estes partilharam sobre os locais que visitaram bem como a uma plataforma de comunicação que permite trocar mensagens. Os utilizadores recebem sugestões de estabelecimentos que pessoas com gostos semelhantes visitaram e que lhes podem interessar. Os proprietários dos estabelecimentos para além de poderem visualizar as opiniões dos visitantes do seu espaço

podem também comunicar com estes através de comentários na sua página ou mensagens directas para os utilizadores [20].

O Yelp conta com 84 milhões de visitas mensais em média [17] e com uma plataforma de utilizadores registados na ordem dos 60 milhões.

2.1.1.6. Gypsii

GyPSii é uma aplicação cujo lema é "sincroniza-te com o mundo à tua volta" [21], permitindo aos seus utilizadores ligarem-se aos seus amigos e descobrir onde eles se encontram a cada momento. Os utilizadores podem registar a sua vida através da partilha de fotografias e vídeo, associadas à sua localização [22]. Utilizando GPS, a localização do utilizador é determinada e apresentada num mapa. O utilizador recorre então ao GyPSii para procurar o quê e quem se encontra à sua volta. A aplicação permite também a integração com as redes sociais Facebook e Twitter.

2.1.1.7. Geocha

O GeoCha [23] é uma aplicação de "*chat*" que permite aos utilizadores comunicarem com outros utilizadores que se encontrem nas imediações através de salas que são criadas por localização geográfica [24]. Os utilizadores podem também criar pontos de interesse que partilham com as pessoas que os rodeiam ou apenas com amigos, permitindo que outros descubram locais que alguém achou interessante. Para além dos pontos de interesse podem ser também partilhados noticias e eventos, bem como a sua localização, que ficam disponíveis num mapa para outros poderem consultar [25]. A conta de um utilizador tem um canal associado, o qual outras pessoas podem seguir e receber notificações em caso de actualizações do mesmo [26]. O GeoCha tem disponível uma aplicação web [27] no seu "*website*", preparada tanto para computadores como dispositivos móveis.

2.1.1.8. LocalResponse

O LocalResponse [28] difere dos serviços apresentados anteriormente pois, em vez de ser uma rede social, trata-se de um serviço de colocação de anúncios. Em vez de a publicidade ser generalista o LocalResponse pesquisa os gostos dos utilizadores em redes sociais como o Twitter [29], Foursquare ou Instagram [30] e coloca anúncios a produtos ou serviços em que estes mostraram interesse [31]. Para além desta funcionalidade é de destacar o serviço de resposta directa que permite às empresas contactar os seus clientes directamente oferecendo sugestões ou descontos relativamente aos seus produtos.

2.1.2. Concorrentes Inactivos

Nesta secção serão descritos serviços que, apesar de já não se encontrarem activos, são interessantes para serem estudados devido a apresentarem características semelhantes ou que possam interessar ao serviço HappyHour.

2.1.2.1. Loopt

Loopt era uma rede social baseada em localização móvel que permitia aos seus utilizadores descobrir o mundo que os rodeia e encontrar amigos, sítios e eventos à sua volta, através de *smartphones*. A aplicação mostrava aos utilizadores a localização dos seus amigos e as suas actividades através de mapas nos *smartphones*. Em Março de 2012 a empresa foi adquirida pela Green Dot Corporation [32], e terminou o seu serviço para se focar no desenvolvimento de novas aplicações na área dos serviços financeiros e de pagamentos [33].

2.1.2.2. Buzzd

O buzzd [34] era um guia de cidades baseado em contribuições sociais. Os utilizadores tinham acesso a avaliações e análises a espaços perto de si, criadas por membros da rede buzzd. A grande vantagem do buzzd foi que, para além da rede própria, agregava informações de outras redes como Twitter, Foursquare, Loopt, Gowalla e BrightKite, além de conteúdos estáticos disponíveis em directórios de empresas [35]. Os utilizadores podiam aceder através da web bem como através de dispositivos móveis com Blackberry, iOS ou Android [36]. O buzzd foi abandonado pois a equipa que o criou era responsável simultaneamente pelo LocalResponse que tinha maior rentabilidade [37].

2.1.2.3. Gowalla

Gowalla era uma rede social que permitia aos utilizadores fazerem "*check-in*" num determinado local ("*spot*") ou numa série de locais ("*trip*") e partilhar essa informação com outras pessoas. Ao fazer "*check-in*" em determinados locais era oferecido um item ao utilizador que poderia ser guardado no seu "baú virtual" ou deixado nesse ou noutro local, tornando-se o utilizador num fundador do ponto de interesse onde deixasse o item [38]. Nas últimas versões passou a permitir aos utilizadores terem acesso aos "*check-in's*" que os seus amigos faziam noutras aplicações como o Facebook e o Foursquare [39]. O Gowalla estava presente em Android, iOS, Windows Phone, Palm WebOS, BlackBerry e através da

aplicação web no seu "*website*". Em Março de 2012 o Gowalla cessou o seu funcionamento no seguimento da sua aquisição por parte do Facebook no ano anterior [40].

2.1.2.4. Whrrl

O whrrl era um jogo social que tinha como objectivo proporcionar novas experiências [41]. O utilizador juntava-se a sociedades (grupos) que partilhavam os seus interesses e estilos de vida recebendo recomendações de locais que encaixassem nesses perfis. Ao experimentarem novos locais os utilizadores recebiam pontos de influência que serviam para aumentar o seu nível numa dada sociedade. Ao subir de nível numa sociedade o utilizador era recompensado com prémios de empresas com parcerias [42]. Com a aquisição da empresa mãe do whrrl – pelago – pela Groupon o serviço acabou por ser descontinuado [43].

2.1.2.5. BrightKite

Brightkite era uma rede social em que os utilizadores podiam fazer "*check-in*" num local, saber quem estava por perto, quem já lá tinha estado e trocar mensagens em grupo [44]. A actualização do estado do utilizador podia ser feita através de aplicações móveis para Android, iOS e Symbian ou através de mensagens de texto. Os utilizadores tinham um sistema de níveis que garantiam recompensas consoante o número de "*check-in's*" num local [45]. Em 2010 o Brightkite mudou de rumo e passou apenas a disponibilizar funcionalidades de mensagens em grupo [46]. Perdendo importância e público face ao seus rivais Foursquare e Gowalla acabou por deixar de fornecer os seus serviços no final de 2011 prometendo regressar com algo melhor [47], o que ainda não aconteceu.

2.1.3. Comparativo

Nesta secção é apresentado um estudo comparativo das vantagens e desvantagens do HappyHour face a outras aplicações semelhantes, abordando também as conjunturas que levaram ao desaparecimento das aplicações que neste momento já não se encontram activas.

Começando pelo Foursquare, o HappyHour apresenta-se como uma aplicação que partilha vários pontos em comum, nomeadamente a detecção de localização e a obtenção de *feedback* para cada estabelecimento. Ambas as aplicações suportam "*push-notifications*", embora o Foursquare permita apenas mensagens entre utilizadores. A nível social, o Foursquare apresenta-se como uma solução mais completa que facilita a realização de "check-ins" com obtenção de "pontos" e "crachás" que incentivam à sua utilização, bem como a criação de "*Tips*" e pesquisa de amigos nas imediações, funcionalidades estas que não estão previstas para a versão protótipo do HappyHour. É possível que algumas destas ideias sejam adaptadas para o conceito do HappyHour em versões futuras, nomeadamente a obtenção de pontos. A nível tecnológico a aplicação HappyHour está a ser desenvolvida de forma a possibilitar a integração de funcionalidades da *Internet of Things* como recolha de informação de sensores ou análise de comportamentos sociais. Este tipo de funcionalidades não são de momento providenciadas pela aplicação Foursquare.

O Facebook Nearby é um serviço em tudo semelhante ao Foursquare sem os "crachás" e as "*push-notifications*". Comparativamente ao HappyHour, é mais focado na componente social em que as pessoas partilham as suas opiniões sobre os espaços e, apesar de ser possível os donos dos estabelecimentos comunicarem com os consumidores através da publicação de mensagens na sua página, esta publicação não despoleta notificações. É também impossível aos administradores saberem se as pessoas com quem estão a comunicar estão perto ou longe. A nível de criação e divulgação de eventos acaba por ser um concorrente de peso ao HappyHour, devido à integração com o Facebook que é a maior rede social a nível mundial com um número avassalador de utilizadores já habituados a usar o sistema.

O Waze não se apresenta como uma aplicação concorrente, pois o HappyHour é uma aplicação direccionada para a diversão nocturna enquanto o Waze é uma aplicação para gestão de tráfego automóvel. Mesmo assim o facto de ser possível planear os nossos planos segundo informações obtidas através de outros utilizadores é um ponto em comum com o HappyHour. No futuro é possível imaginar possíveis pontos de integração, nomeadamente na coordenação de encontros e boleias com amigos, através da partilha de localização e de tempos estimados de viagem.

O GetGlue é uma rede social diferente, baseada em conteúdos e com poucas semelhanças face ao HappyHour uma vez que não tem a mesma finalidade. Foi incluída neste estudo pois a sua componente de partilha de gostos multimédia é interessante e algumas ideias relativas a este tema podem vir a ser integradas no módulo de playlists e músicas disponível no HappyHour.

Comparando o HappyHour ao Yelp, verificamos que existem semelhanças ao nível da disponibilização de informação dos estabelecimentos. Infelizmente existem limitações quanto à cobertura do serviço e este só está disponível em algumas cidades nenhuma destas em Portugal. O serviço HappyHour vai estar disponível em todo o território nacional e com a possibilidade de a sua utilização ser alargada para qualquer local no mundo de forma simples. Apesar de se apresentar como um directório de serviços social, a comunicação estabelecimento - utilizador está apenas limitada a comentários e mensagens assíncronas, enquanto no HappyHour os consumidores podem ser contactados em tempo real.

O GyPSii não tem muito em comum com o HappyHour, pois não se enquadra no conceito de procura de estabelecimentos. A possibilidade de saber onde se encontram os seus amigos num determinado momento é uma funcionalidade que poderá ter interesse em versões futuras do HappyHour.

A serviço GeoCha apresenta uma tecnologia interessante a nível de comunicação com pessoas ao redor de um local. A aplicação foi analisada pois utiliza um sistema semelhante ao que será implementado nas notificações do HappyHour. A diferença é que este último permitirá a comunicação de promoções e ofertas aos utilizadores ao invés de uma comunicação entre utilizadores.

O LocalResponse é um sistema de envio directo de publicidade e das aplicações analisadas é a que mais se assemelha ao princípio básico do HappyHour, no entanto com uma utilização mais específica e limitada. A pesquisa dos gostos do consumidor é algo que tencionamos aplicar mas de uma forma um pouco distinta. Em vez de se ter em conta os locais onde o utilizador esteve e as compras que fez, o HappyHour pretende saber quais os gostos a nível de bebidas e de música de cada utilizador para saber se a promoção de um bar pode ser mais ou menos interessante. A nível do envio de promoções e ofertas directamente ao utilizador, o LocalResponse utiliza as plataformas de redes sociais já existentes para transmitir a antigos clientes informações que lhe possam ser úteis não tendo em conta a localização destes. O HappyHour, com o envio de publicidade apenas a utilizadores que se encontrem nas

proximidades do estabelecimento, garante que quem recebe as informações pode, dentro de minutos, estar a usufruir das promoções que acabou de receber. Enquanto o LocalResponse é um sistema cujo objectivo é unicamente a distribuição de publicidade, o HappyHour irá ter uma componente de serviço aplicada a estabelecimentos de diversão nocturna, permitindo aos utilizadores obter informação sobre eventos, tipos de música e público dos estabelecimentos, em tempo real.

O Loopt partilha algumas das funcionalidades de localização presentes noutras aplicações como o buzzd, e o Brightkite, as quais são também partilhadas pelo HappyHour. No entanto, a rede social é mais limitada, no sentido em que apenas permite a descoberta de locais e de amigos, não permitindo obter informação em tempo real nem notificações sobre estes, como o HappyHour se propõe a fazer.

A grande vantagem do buzzd é que, para além de ter espaços criados por utilizadores, ainda agregava dados de vários directórios de serviços, o que garantia uma grande cobertura a nível mundial. No HappyHour usaremos bases de dados previamente criadas de um simples provedor complementadas com os dados internos do nosso sistema. Ao evitarmos utilizar uma grande diversidade de provedores de bases de dados, evitamos a repetição de dados devido a falhas de posição dos estabelecimentos no mapa. A nível de semelhanças com o HappyHour, o buzzd utilizava também um sistema de avaliação dos espaços por parte dos utilizadores. Esta aplicação deixou de ser desenvolvida pois o LocalResponse estava a ter uma rentabilidade maior provando que um sistema como o HappyHour, que permite o envio de marketing directo, acaba por ser superior.

A rede Gowalla tinha um funcionamento semelhante ao Foursquare. Relativamente ao HappyHour partilhava a detecção de localização e a obtenção de *feedback* para cada estabelecimento. O facto de ter recorrido a um sistema de prémios aos utilizadores acabou por lhe proporcionar um grande sucesso inicial mas que, com o passar do tempo, se foi diluindo. As semelhanças com o Foursquare, que era uma empresa com uma base de utilizadores maior, diminuiu o sucesso da empresa que acabou ao ser comprada pelo Facebook. Na minha opinião, não haver um factor diferenciador em relação à concorrência levou ao seu fim.

O whrrl não tem muito em comum com o HappyHour visto que o seu principal foco era a competitividade entre utilizadores. Ainda assim foi estudado devido às semelhanças nas recomendações consoante os gostos do utilizador. No whrrl essas recomendações eram feitas consoante os grupos em que os utilizadores estavam, enquanto no HappyHour as recomendações terão em consideração os gostos destes. O whrrl acabou por encerrar a sua actividade, não devido ao serviço prestado mas devido a uma aquisição por outra empresa que a incorporou.

O BrightKite tinha um sistema de mensagens também baseado em localização geográfica, mas neste caso para troca de mensagens entre utilizadores. A integração com um sistema de envio de SMS do Brightkite pode ser interessante numa actualização futura da aplicação HappyHour, uma vez que permite estender os serviços de envio de notificações a quem não seja possuidor de um smartphone ou de um acesso à Internet sempre disponível.

Na Tabela 1 pode-se observar uma comparação entre as funcionalidades disponibilizadas tanto pela aplicação HappyHour como as suas concorrentes ainda activas.

	Check in	Localização POIs	Rating	Marketing Directo	Eventos	Rede Social
HappyHour		•		•	•	
Foursquare	•	•	•		•	•
Facebook Nearby	•	•	•		•	•
Waze						•
GetGlue	•					•
Yelp		•	•			
Gypsii		•				•
Geocha		•			•	•
LocalResponse				•		

Tabela 1 - Comparativo Concorrentes Activos

2.1.4. Conclusões

Após uma análise às aplicações apresentadas na secção anterior, chegamos à conclusão que o HappyHour se apresenta como um serviço que se baseia em algumas ideias existentes que já provaram a sua eficácia, mas que se apresenta como inovador no mercado mais específico da diversão nocturna.

Já existem várias ferramentas no mercado que permitem interligar o mundo real com o virtual, algumas apresentando funções bem mais avançadas que as previstas para a versão inicial do HappyHour. Redes sociais como o Foursquare e o Facebook Nearby permitem fazer "*check-ins*", enviar "*push-notifications*" entre utilizadores, ganhar pontos e realizar avaliações e recomendações de estabelecimentos comerciais - redes sociais como o Yelp, fazem destas funcionalidades o seu principal foco. Outras redes sociais, como o GetGlue, jogam com a partilha de gostos culturais entre utilizadores. Estas são características que em muito beneficiariam e que se prevêem existir no futuro, mas que ainda se encontram numa versão muito preliminar no HappyHour.

A versão protótipo prevista para esta fase do projecto ainda é muito simples, e não apresenta muita inovação sobre estas funcionalidades, pois apenas permite o envio de notificações aos utilizadores e a obtenção de informação sobre os estabelecimentos. Apesar de tudo, a maneira como o HappyHour joga com estas informações e funcionalidades é inovadora e não se encontra presente nas outras aplicações. Enquanto o Foursquare, o Facebook

Nearby, o Yelp e o GetGlue são redes sociais colaborativas, continuam a ser estáticas. Ou seja, os utilizadores influenciam os locais que visitam através dos seus comentários e feedback, mas estas influências não são dinâmicas: acontecem quando o utilizador faz "*check-in*", e não se manifestam para além disso, ou seja ficam "estáticas" no sistema. É verdade que o *feedback* de um local se altera ao longo do tempo, pois mais utilizadores continuarão a fazer "*check-in's*" e a deixar comentários. No entanto, estas informações não trazem nenhuma percepção sobre o ambiente de um local num determinado momento. Por outro lado, o HappyHour permite que a própria presença e movimento dos utilizadores afecte a maneira como os estabelecimentos são percepcionados por estes, em tempo real. Se um bar se encontra cheio, o utilizador saberá disso; se as pessoas estão agitadas e em movimento, ou mais paradas e em descanso, o utilizador saberá disso e será informado deste estado, não de uma forma estática e dependente da participação activa dos utilizadores, mas de uma forma dinâmica, em tempo real. Os seus próprios gostos musicais serão uma mais-valia para os donos dos estabelecimentos, podendo influenciar a música que é passada nos mesmos. Assim, o HappyHour não só constrói e tira partido de ideias já existentes e comprovadas, mas inova no sentido de trazer a estas ideias uma componente dinâmica. A única rede que se aproxima deste conceito, das que foram apresentadas, é o Waze, que não é um concorrente ao HappyHour pois não trata do mesmo assunto, focando-se na gestão de tráfego automóvel.

O HappyHour tira partido também de ser uma aplicação orientada, no sentido de ter um objectivo em concreto: enquanto o Foursquare, o Facebook Nearby e o GetGlue são redes sociais generalistas, em que o objectivo é a livre partilha de informação entre utilizadores, o HappyHour assume uma posição mais próxima do Yelp e apresenta-se como um serviço. O seu objectivo é proporcionar informação sobre os estabelecimentos que se encontram nas imediações do utilizador, e transmitir notificações ("*push-notifications"*) sobre os mesmos. Tenhamos em consideração que estas "*push-notifications*" são diferentes das usadas pelo GeoCha e pelo Foursquare, as quais são trocadas entre utilizadores. As notificações do HappyHour são de uma vertente mais publicitária, transmitindo informações úteis aos utilizadores, dadas pelos próprios estabelecimentos nocturnos. De facto, existem várias opiniões que frisam esta necessidade de dar um rumo às aplicações sociais móveis, em termos de objectivo. Numa entrevista para o "*website*" AllThingsD.com, o CEO da Loopt, Sam Altman, justifica a venda da empresa à companhia bancária Green Dot vendo-a como uma oportunidade para tornar a partilha de localização mais útil. Segundo ele, o Loopt andou a experimentar com os conceitos de ofertas consoante lealdade para com estabelecimentos, mas não tinha efectivamente uma forma de traduzir isso em pagamentos. A ligação com uma companhia bancária permite fazer uma ligação directa entre a detecção de localização e o comércio [32]. Tal como sugerido em [48], a combinação de redes sociais, localização e pagamentos permite a criação de aplicações em que o telemóvel do utilizador recebe uma notificação sobre uma promoção no café mais próximo, onde a compra de um café dá direito um pão grátis e em que o utilizador nem precisa de retirar o seu cartão de crédito da algibeira, podendo efectuar o pagamento com o próprio telemóvel. De acordo com a opinião de Josh Martin, director da área de investigação de aplicações para uso na rede "*wireless"* global da empresa [49], este tipo de serviços de "check-in" nunca evoluíram o suficiente para se tornarem ofertas independentes, devendo ser vistos mais como uma funcionalidade em vez de um serviço. Prevê-se que as aplicações que tratam unicamente de "*check-in*" irão tendencialmente desaparecer, sendo integradas em aplicações mais generalistas. O HappyHour vai de encontro a esta previsão, pois é uma aplicação que não se foca exclusivamente na partilha de localização de utilizadores, mas usa esta informação com o objectivo de caracterizar o ambiente de estabelecimentos nocturnos e de partilhar publicidade de interesse aos utilizadores nas imediações.

Podemos então concluir que o HappyHour é uma aplicação que usa muitas das ideias já testadas e existentes no mercado. Apesar de se ainda encontrar numa fase inicial, onde a principal funcionalidade serão as "*push-notifications*" enviadas pelos estabelecimentos nocturnos, a ideia do HappyHour que será construída em versões futuras é bastante distinta e inovadora sobre o que já existe, podendo assim ser uma mais-valia no mercado das aplicações sociais móveis.

2.2. Análise SAPO.Mapas.API e Foursquare API

2.2.1. API SAPO.MAPS POI

A API do SAPO.Maps permite integrar os mapas disponibilizados pela SAPO em diversas aplicações. Apesar de a API ser bastante abrangente, neste caso em concreto, o interesse nesta resume-se apenas a uma classe: a SAPO.Maps.Poi. Esta classe da API permite aceder a todos os POIs disponibilizados pela plataforma SAPO.Maps para que possam ser integrados noutras plataformas. Os POIs disponibilizados pela SAPO abrangem todo o território nacional e apresentam uma boa cobertura, incluindo a grande maioria das localidades nacionais.

Após um contacto inicial verificou-se a existência de pouca informação sobre a implementação da API excepto a fornecida oficialmente pela SAPO. Foi encontrada informação no "*website*" dedicado à API mapas e no "*website*" do SAPO.Developers. No entanto esta encontra-se na sua grande parte incompleta devido a erros em ambos os "*websites*" e à falta de documentos referenciados nestes. Como se pode ver nas Figuras 1 a 3 estes erros são frequentes e impossibilitam o acesso a informação relevante relativa à implementação da API.

Figura 1 - Erros nas páginas de documentação da API

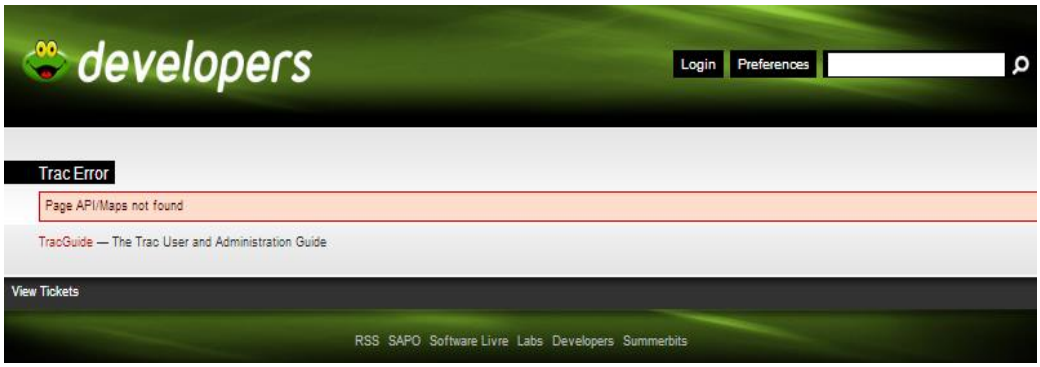

Figura 2 - Página sobre a API não encontrada

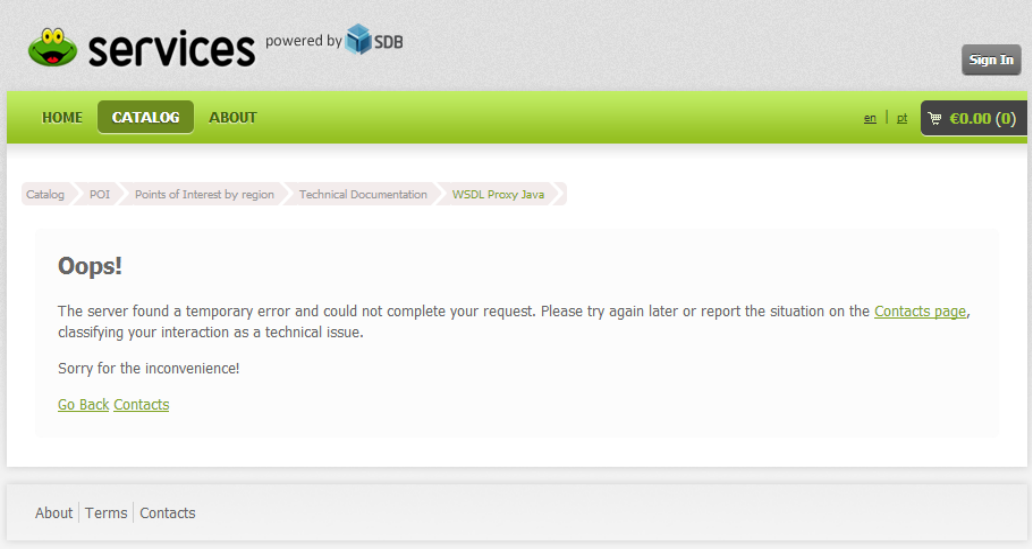

Figura 3 - Client proxy generation indisponível em qualquer linguagem de programação

Mesmo quando essa informação estava disponível continha alguns erros que impediam a sua utilização, como exemplificado na figura 4 onde o código disponível para teste apresenta erros de implementação.

Estas adversidades revelaram-se inultrapassáveis pelo que não foi possível obter uma lista dos POIs presentes no SAPO.Maps. Foi sendo apenas possível obter uma representação de um mapa com funcionalidades básicas de navegação e procura.

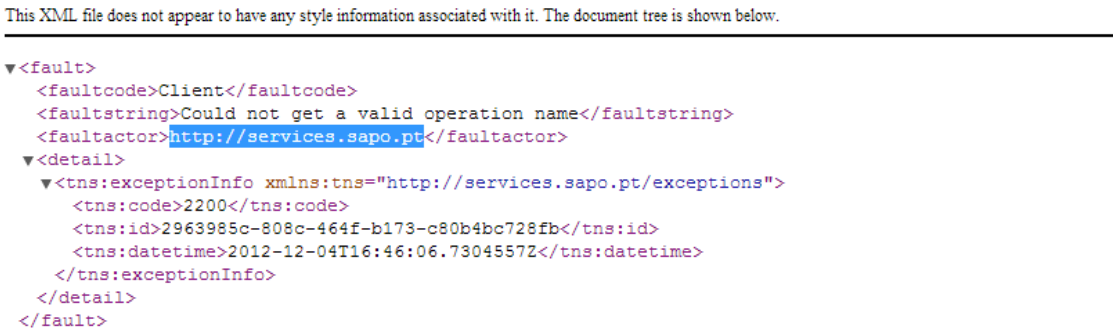

Figura 4 - Código de teste para tutorial com erros de implementação

2.2.2. API Foursquare

A API Foursquare fornece acesso a todas as operações e dados disponibilizados na aplicação móvel Foursquare. Devido ao grande número de utilizadores desta plataforma a API tem uma elevada quantidade de informação disponível. Para além da documentação oficial – que é bastante completa – foi possível encontrar bastantes exemplos e guias em diversos fóruns e "*websites*" dedicados à tecnologia.

A nível de implementação foi efectuado um conjunto de testes de funcionamento em que, com uma simples chamada à API, conseguiu-se obter uma listagem dos POIs numa determinada zona. Como já existia uma implementação no servidor Youth5G optou-se por continuar os testes nesta plataforma de forma a simplificar ainda mais o processo e a facilitar a ambientação com a API.

Durante os testes efectuados foi possível, através da referência de cada POI, obter informações sobre este, como morada e contactos, fotografias e eventos associados. Este módulo de eventos é extremamente interessante para a aplicação HappyHour uma vez que a sua integração com o sistema de gestão de eventos disponibilizado pela nossa aplicação potencia a informação oferecida ao utilizador.

A nível de pontos de interesse a cobertura foi praticamente perfeita uma vez que a quase totalidade de POIs em zonas conhecidas pelos elementos da equipa se encontravam presentes na base de dados disponibilizada pelo Foursquare. Para além disso esta API disponibiliza pontos de acesso actualizados em praticamente todos os locais do mundo.

2.2.3. Conclusões

Devido à dificuldade na obtenção de informação relativa ao funcionamento da API SAPO.Maps foi decidido utilizar a API desenvolvida pelo Foursquare. Comparativamente, a API fornecida pela SAPO foi bastante mais simples de implementar bastando seguir as instruções da documentação oficial. A nível de POIs não foi possível obter uma comparação quantitativa do número de POIs disponíveis numa e noutra aplicação. No entanto, através de uma observação das mesmas zonas, em mapas povoados por POIs de vida nocturna de um e de outro serviço, notou-se uma maior predominância dos POIs fornecidos pelo Foursquare como pode ser observado na **Error! Reference source not found.** e **Error! Reference source not found.**. Além disso o facto de o Foursquare ter uma cobertura mundial traduz-se numa mais fácil adaptação a um possível alargamento do serviço a outros países.

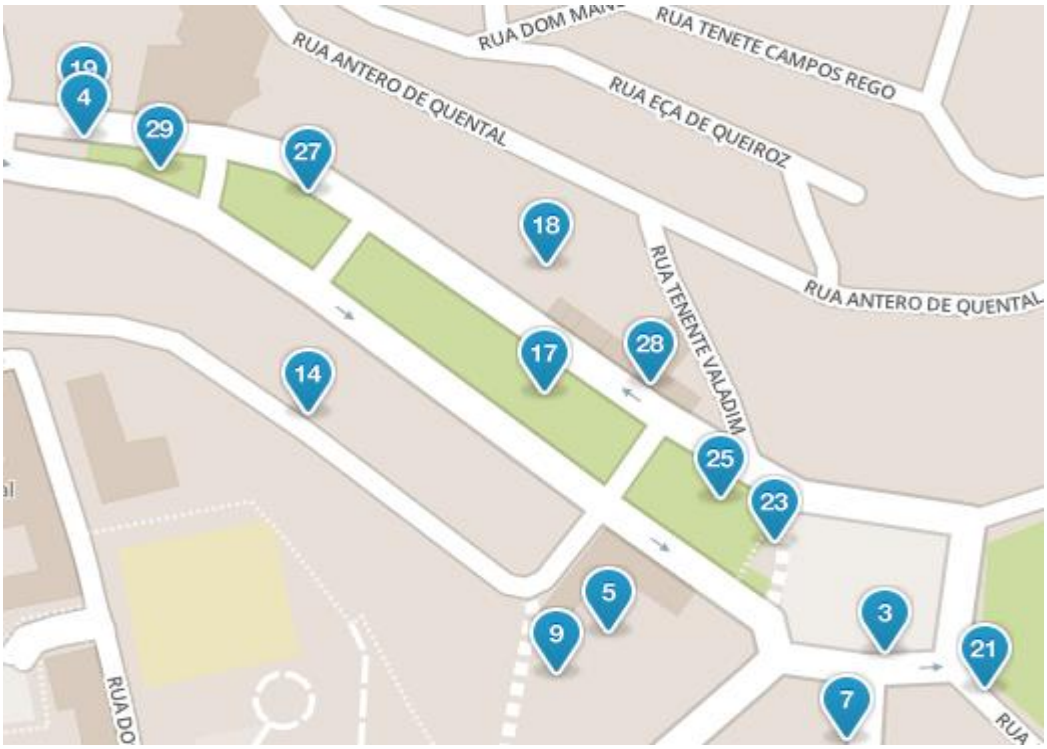

Figura 5 - Pontos de interesse de vida nocturna na zona da Praça da República em Coimbra fornecidos pelo Foursquare

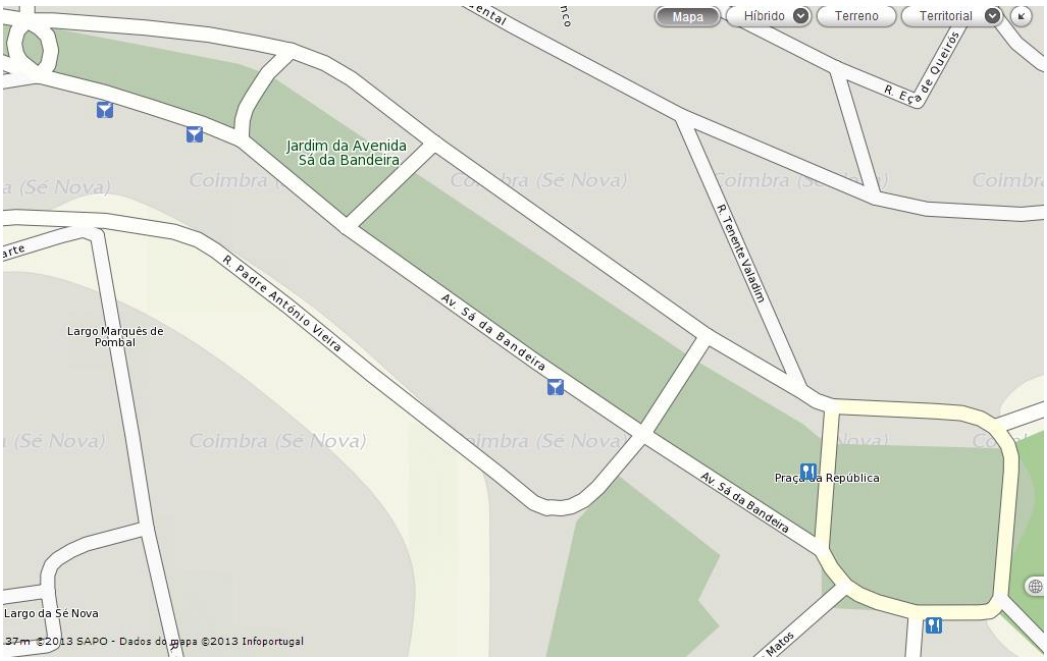

Figura 6 - Pontos de interesse de vida nocturna na zona da Praça da República em Coimbra fornecidos pelo Sapo Mapas

A integração com o Foursquare revelou ainda outros benefícios pois, para além dos dados dos POIs serem mais completos, permite ter acesso à lista de eventos de cada um destes. Ao combinar os dados de eventos presentes na plataforma Foursquare com os que são fornecidos pelos administradores de POIs à aplicação HappyHour conseguimos aumentar a quantidade de informação disponível para os utilizadores.

2.3. Análise de Riscos

Os principais factores que podem levar a percalços na aplicação são maioritariamente externos. A aplicação HappyHour baseia-se bastante em APIs fornecidas por outras empresas pelo que a sua alteração ou descontinuação podem complicar o desenvolvimento da mesma. Também se teve em consideração problemas com perdas de dados, disponibilidade de servidor e problemas com certificados devido à utilização do HTTPS. Na TABELA2 está detalhada a análise feita aos possíveis riscos bem como possíveis soluções.

Descrição	Impacto [0-5]	Plano de contingência
API Foursquare ser alterada/descontinuada	4	Tentar utilizar outros serviços de fornecimentos de POIs.
GoogleMaps API ser alterada/descontinuada	5	Procurar outro serviço de apresentação de mapas como o Foursquare ou o SAPO Mapas apesar de terem funcionalidades mais reduzidas.
API SAPO Mapas não ser possível de implementar	1	Continuar a implementação usando a API Foursquare.
API Facebook ser alterada/descontinuada	2	Encontrar outra solução para confirmação da identidade dos administradores
Bibliotecas DOJO serem alteradas/descontinuadas	3	Utilizar uma versão descontinuada localmente no servidor com a desvantagem de não ser actualizada e de ocupar bastante espaço.
Não ser possível obter certificados HTTPS válidos	1	Criar certificados próprios e quando se passar para a vertente comercial investir em certificados.
Perda de dados durante o desenvolvimento	5	Manter sempre cópias actualizadas na SVN, Dropbox e máquinas de desenvolvimento.
Problemas com o servidor SVN	4	Manter sempre cópias locais actualizadas em várias máquinas bem como na Dropbox..
Impossibilidade de integrar o serviço MusicBox	1	Utilizar outros serviços de comercialização de música com o Google Play Music.
Dificuldades em utilizar a plataforma Youth5G	4	Iniciar uma implementação de raiz com os respectivos custos temporais.

Tabela 2 - Análise de Riscos

Capítulo 3
Arquitecturas

3.1. Tecnologias Utilizadas

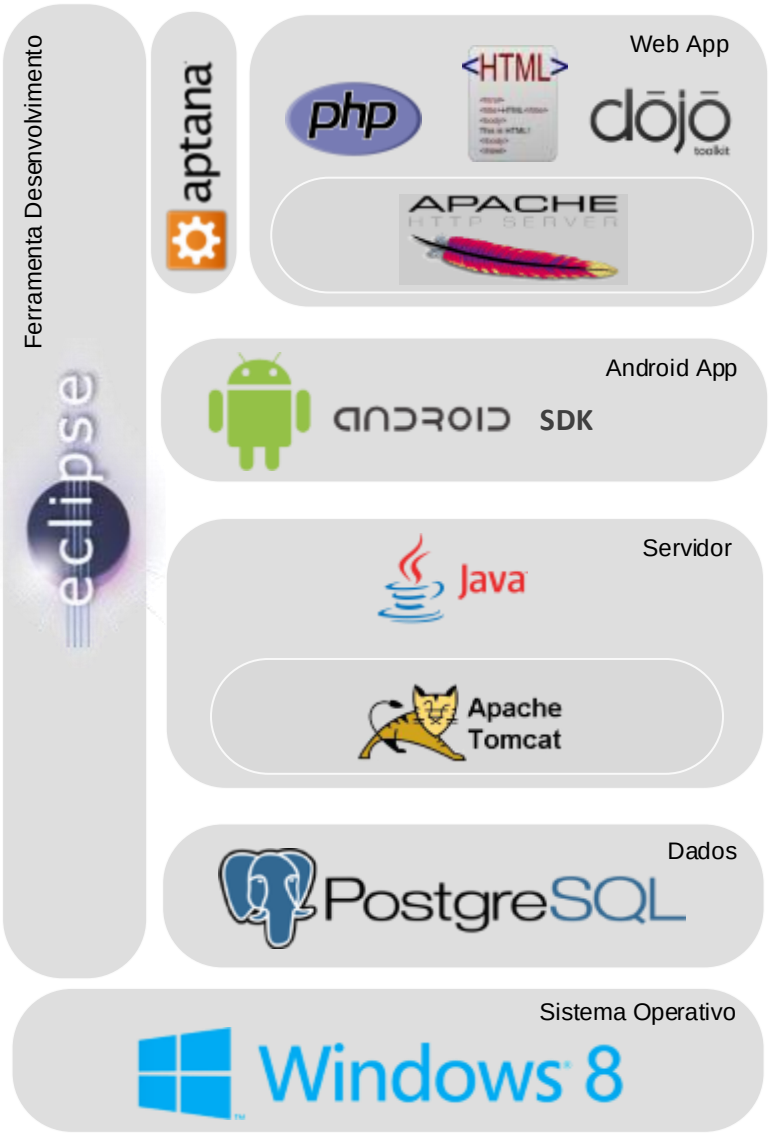

Figura 7 - Ferramentas de Desenvolvimento

Para o desenvolvimento de todos os módulos que compõem a aplicação HappyHour foi utilizado o IDE Eclipse. A escolha recaiu neste IDE pois permite desenvolver aplicações em todas as linguagens de programação que iremos utilizar durante o projecto e ambos os membros da equipa já tiveram contacto com esta plataforma no passado.

No desenvolvimento da aplicação Android foi utilizado como base o SDK da versão 2.1 com o "*Add-On*" Google APIs. Esta opção deveu-se o facto de o cliente Android do Youth5G estar implementado nesta versão e assim ser possível reaproveitar parte do código. Para além desta razão, segundo um estudo efectuado a nível mundial pela Google, em Janeiro de 2013, as versões anteriores à 2.1 do sistema operativo Android representam uma penetração de mercado de 0,2% [50]. Como as versões superiores do sistema operativo Android oferecem retro compatibilidade, garantimos assim que a aplicação é funcional em

99,8% dos telemóveis com este sistema operativo, maximizando o público-alvo. O "*Add-On"* Google APIs teve de ser utilizado pois este permite usar a API do Google Maps necessária para apresentar o mapa no ecrã do utilizador, obter a sua localização e calcular rotas.

A implementação do servidor manteve-se semelhante à utilizada no projecto Youth5G utilizando Webservices disponibilizados por Java Beans a correrem em Apache Tomcat. Para guardar os dados manteve-se a implementação já existente em PostgreSQL uma vez que é um sistema com que ambos os membros já tinham utilizado no passado e tem suporte para grandes quantidades de dados mantendo boas performances.

A aplicação WEB HappyHour Management criada para os gestores dos estabelecimentos poderem inserir e editar eventos. Foi desenvolvida como uma aplicação Facebook usando PHP e Javascript correndo num servidor Apache utilizando o plugin Aptana Studio.

Para controlo de versões no desenvolvimento do "*software"* foi utilizado o servidor de SVN do Departamento de Engenharia Informática. Interligando o IDE Eclipse com este servidor SVN garantimos que todas as alterações ficam registadas e que, em caso de necessidade, seja possível reverter a versões antigas. Para além disso adicionando comentários a cada versão é possível consultar como está a decorrer o desenvolvimento da aplicação.

Para a gestão dos documentos produzidos no âmbito deste projecto é utilizada a Dropbox que também permite "*versioning"* tal como o SVN, mas implementa um sistema de actualizações automáticas dos ficheiros que facilita o processo de escrita. Como a Dropbox não permite adicionar comentários a cada alteração, são utilizadas as funcionalidades do Microsoft Word de registo de alterações e comentários para serem mais facilmente identificadas quais as secções que foram alteradas e o que ainda é necessário fazer ou rever.

3.2. Arquitectura Geral da Aplicação

A plataforma HappyHour vai ser constituída por uma aplicação servidor e duas aplicações cliente.

O servidor será responsável por tratar de todo o processamento de pedidos das restantes aplicações, comunicar com a API do Foursquare e guardar todos os dados necessários numa base de dados.

A aplicação Web vai servir para os administradores de bares gerirem os seus estabelecimentos enviando notificações e definindo eventos.

Aos utilizadores vai ser disponibilizada uma aplicação para o sistema operativo Android capaz de indicar os pontos de interesse ao redor, ver detalhes e eventos, e receber notificações de promoções ao seu redor.

A estrutura geral da aplicação e a forma como os diferentes módulos estão interligados podem ser visualizados na Figura 8.

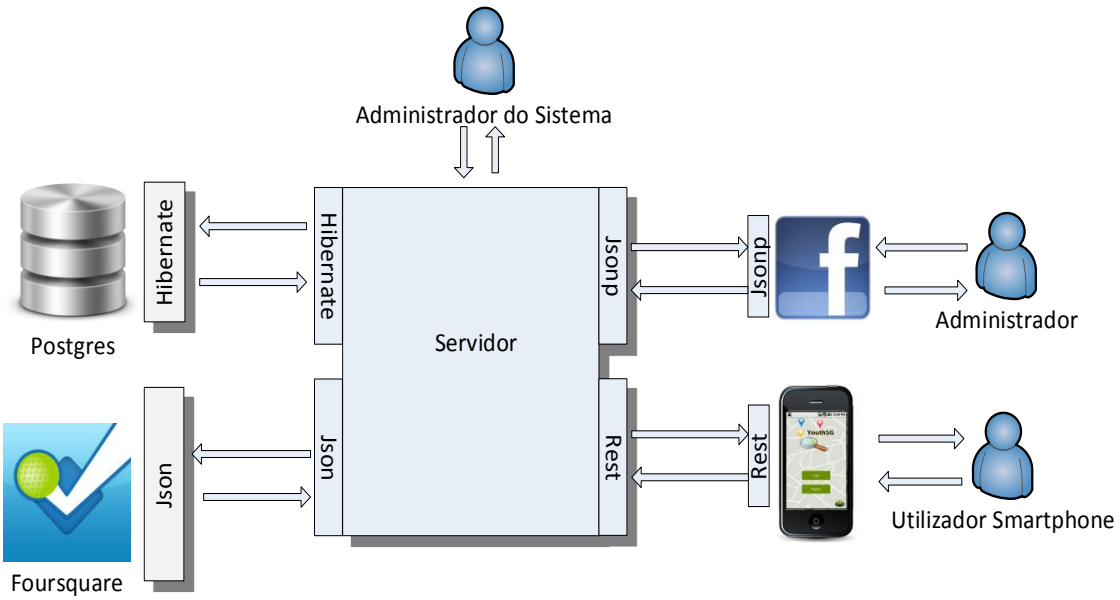

Figura 8 - Visão geral da plataforma HappyHour

3.3. Arquitectura Técnica

3.3.1. Arquitectura de Dados

Todas as informações necessárias ao funcionamento da plataforma HappyHour são guardadas numa base de dados assim que o processamento das mesmas tenha sido concluído pelo servidor. Seguindo este modelo garantimos que em caso de um downtime planeado ou involuntário do servidor as informações dos utilizadores não são perdidas. O facto de as informações estarem centralizadas permite também preparar o sistema para algum eventual ataque para roubo de informação.

A base de dados, implementada usando o sistema de gestão de bases de dados PostgreSQL, é constituída por 17 tabelas relacionadas entre si como se pode ver no modelo físico presente no Anexo B. A estrutura criada promove uma distribuição dos dados pelas várias tabelas para que seja possível aceder-lhes de forma mais simples e directa.

A questão da privacidade dos utilizadores é normalmente um tema sensível principalmente quando estão em causa aplicações de localização de pessoas e, neste caso, não foi descurado. A informação de localização dos utilizadores fica disponível no servidor durante 30 minutos após ter sido recebida. Após este período toda a informação é removida de forma permanente garantido que não é possível repetir os passos de um utilizador. No equipamento móvel são apenas guardados em memória os dados de login, se o utilizador o pretender, para facilitar a entrada na aplicação. Esta informação é guardada na memória "SharedPreferences" que limita o acesso apenas à aplicação que guardou os dados evitando assim que outra aplicação do utilizador tenha acesso aos seus dados de login.

Para garantir a integridade e segurança dos dados todos os pedidos ao servidor incluem os dados do utilizador registado e existem vários níveis de permissões de forma a evitar acessos indesejados.

3.3.2. Arquitectura do Servidor

O servidor da aplicação HappyHour tem por base por três elementos principais: uma base de dados, uma camada de acesso a dados e uma camada de webservices.

Na base de dados estão guardados todos os elementos persistentes como informações de utilizadores, pontos de interesse, eventos e notificações.

Para mediar o acesso a esta base de dados foi criada uma camada de acesso com objectos DAO (Data access object) para cada uma das tabelas da base de dados. Os objectos DAO são objectos abstractos que servem de intermediários entre a base de dados e uma aplicação, permitindo manter uma separação entre ambos. Esta separação é vantajosa pois alterações à base de dados não afectam directamente a aplicação servidor. A comunicação entre os objectos DAO e a base dados é feita através de Hibernate.

A utilizar os dados fornecidos de forma transparente pela camada de acesso a dados teremos a camada de serviço que pode ser dividida em duas sub-camadas. Por um lado temos os interfaces fornecidos através de WebServices que recebem a informação proveniente das aplicações cliente. Por outro temos a implementação dos próprios serviços onde é feito o processamento dos dados.

Assim que um pedido chega a um serviço, a informação que este contém é convertida num objecto Request Data e enviado para a camada de implementação Com onde, através das informações disponibilizadas pela camada DAO, é gerada uma resposta. Essa resposta é compilada num objecto do tipo ResponseData e é reencaminhado para a aplicação cliente que fez o pedido através do Webservice utilizado.

Na camada de serviços existe uma *thread* a correr de cinco em cinco minutos que manipula dados de notificações, e agitação e afluência. Esta *thread* é independente dos restantes elementos da mesma camada interagindo apenas com os objectos DAO correspondentes às tabelas que contêm a informação manipulada por esta.

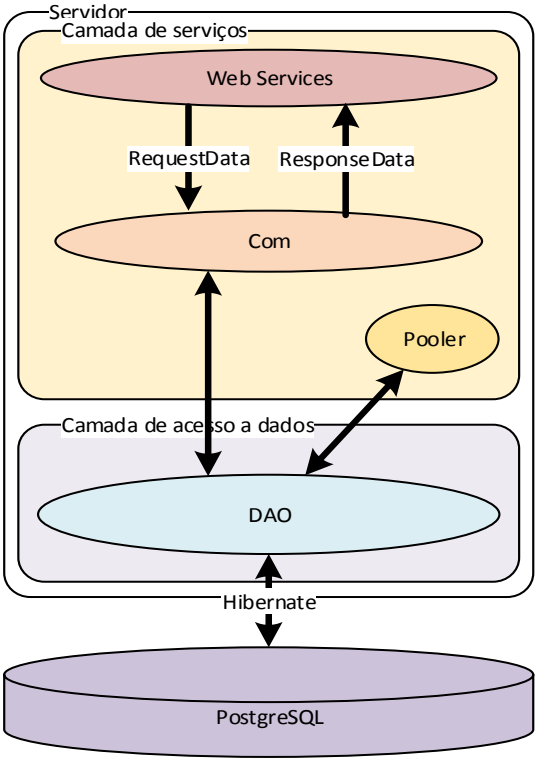

Figura 9 - Arquitectura do Servidor

3.3.3. Arquitectura da Aplicação WEB

A aplicação WEB é constituída por quatro módulos como se pode observar na .

O módulo Página Inicial é constituído por uma página principal – Index – onde é pedido ao utilizador os dados de autenticação ou, caso este não os tenha, é pedido o registo. Quando o utilizador confirma os dados, o Login comunica com o servidor para os validar enquanto, simultaneamente, o Login_as_Page liga-se à API do Facebook de forma a obter os dados das páginas do utilizador.

Após os dados de autenticação terem sido validados o utilizador é redireccionado para a página Main do Menu principal. No Menu o utilizador pode escolher se pretende gerir eventos ou notificações sendo redireccionado para a respectiva página.

O módulo de Notificações é constituído por uma página Notifications onde é mostrada uma lista de todas as notificações existentes para um determinado ponto de interesse e, ao escolher uma destas, são abertos os detalhes através do NotificationEdit. Esta página vai fazer um pedido ao servidor para receber essa notificação e, se o utilizador pretender altera-la, a página envia os detalhes corrigidos para o servidor. No caso de o utilizador pretender criar uma nova notificação é reencaminhado para a página NotificationNew onde preenche os dados e estes são enviados para o servidor.

O funcionamento do módulo Events é semelhante ao módulo notificações.

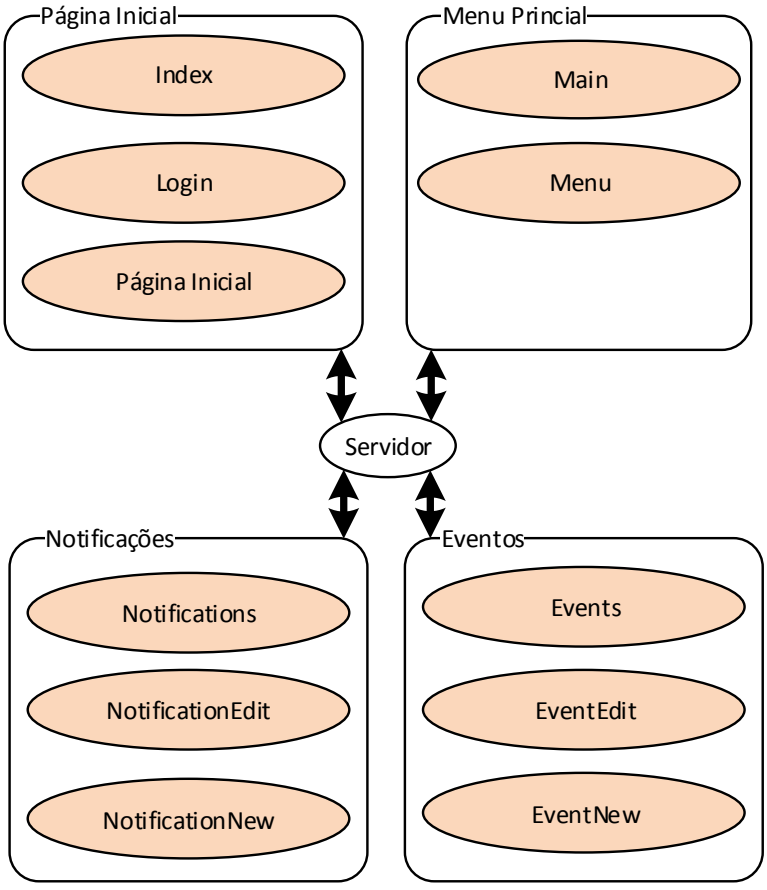

Figura 10 - Arquitectura Aplicação WEB

3.3.4. Arquitectura da Aplicação Android

De forma a aproveitar melhor os recursos dos equipamentos optou-se por distribuir os serviços por *threads* separadas que correm paralelamente. Com esta separação a interacção com o utilizador é melhorada pois evita paragens para processamento de dados na *thread* que apresenta o interface não notando o utilizador quebras de performance. Para além disso com esta abordagem permitimos ao sistema operativo Android distribuir *threads* pelos vários núcleos do processador caso o equipamento suporte.

A aplicação Android é constituída por dois módulos principais. O primeiro, "SplashScreen", tem como responsabilidade obter os dados de login ou de registo do utilizador e de verificar a autenticidade deles através de um pedido ao servidor. Após a autenticação estar concluída todo o processamento da aplicação fica centrado no módulo "MapRouteActivity". Este é responsável por apresentar as informações vindas do servidor ao utilizador. Para isto é utilizado um objecto "GoogleMap" que recorre à "Google Maps API" para desenhar um mapa no ecrã com marcadores que sinalizam os pontos de interesse. Quando o mapa é inicializado em simultâneo é criado um "MapCameraListener", que reage às interações do utilizador com o mapa e o redesenha quando necessário, e um "NotificationService" que corre numa *thread* independente. Esta *thread* é responsável pela leitura dos valores de GPS e do acelerómetro do dispositivo móvel, de os enviar para o servidor e de receber a resposta. Caso a resposta contenha notificações este serviço é também responsável por mostrar as mesmas ao utilizador. Como é um serviço à parte da restante aplicação este pode ser executado em segundo plano permitindo ao utilizador receber notificações e ao servidor HappyHour receber informações.

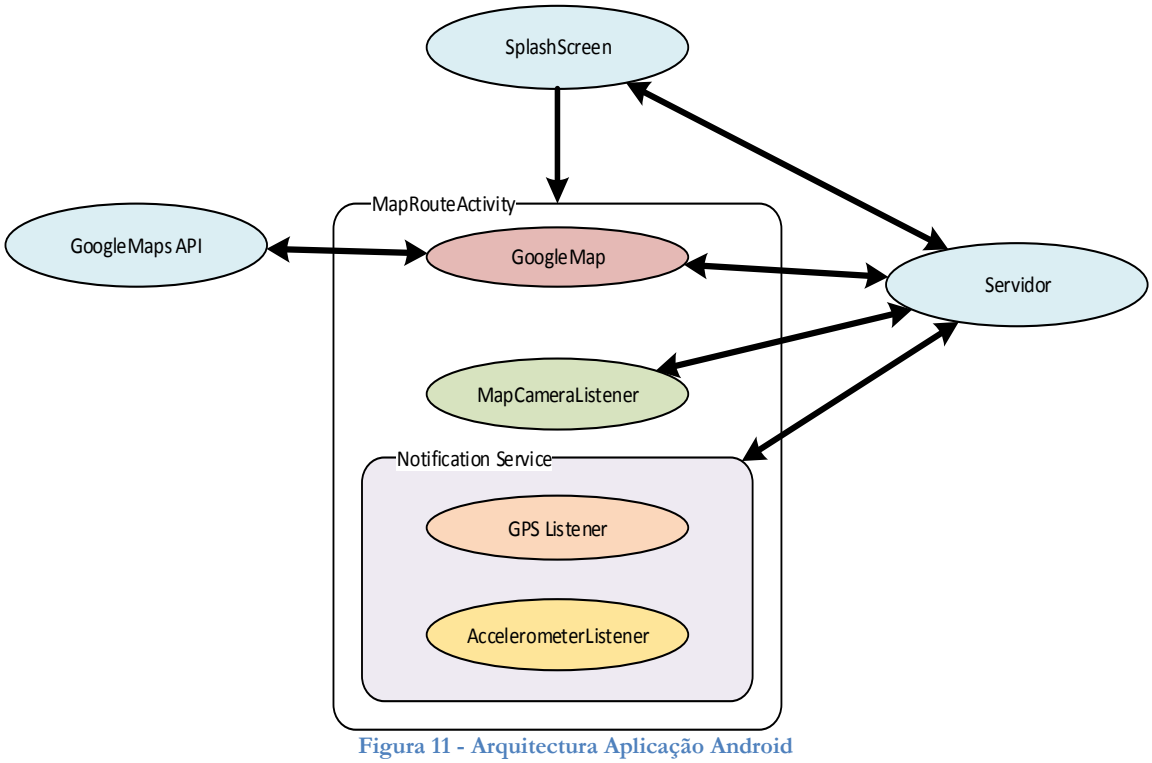

Figura 11 - Arquitectura Aplicação Android

3.3.5. Arquitectura de Comunicação

Como detalhado na Figura 8, apesar de todas as comunicações serem centralizadas no servidor, a comunicação entre as várias aplicações é feita de forma distinta:

- Entre o servidor e a aplicação Android está implementada uma comunicação utilizando WebServices REST;
- A comunicação entre a API Foursquare e o servidor utiliza mensagens JSON uma vez que este é o protocolo utilizado pela API;
- A aplicação WEB utiliza também a formatação JSON para comunicar com o servidor. No entanto, devido a existirem pedidos *"cross-domain"* é necessário codifica-las através de mensagens JSONP;
- As consultas e alterações à base de dados são feitas utilizando mensagens Hibernate.

Capítulo 4
Trabalho Desenvolvido

4.1. Aplicação Youth5G como ponto de partida

Quando integrei o projecto HappyHour este tinha como base um protótipo previamente desenvolvido pelo aluno Duarte Raposo chamado Youth5G. Esta plataforma é constituída por uma aplicação ubíqua que criada para o sistema operativo Android, por um servidor "*backend*" que integra os serviços Youth5G e que apresenta já algumas das funcionalidades previstas para o projecto HappyHour.

A aplicação ubíqua Youth5G permite aos utilizadores observarem um mapa onde podem consultar quais os estabelecimentos ao seu redor bem como os detalhes, contactos e eventos associados a cada um destes. É possível também atribuir pontuações e feedback a cada um destes POIs. A aplicação inclui ainda funcionalidades de navegação que ficaram inutilizadas com as alterações implementadas pelo Google na API Google Maps.

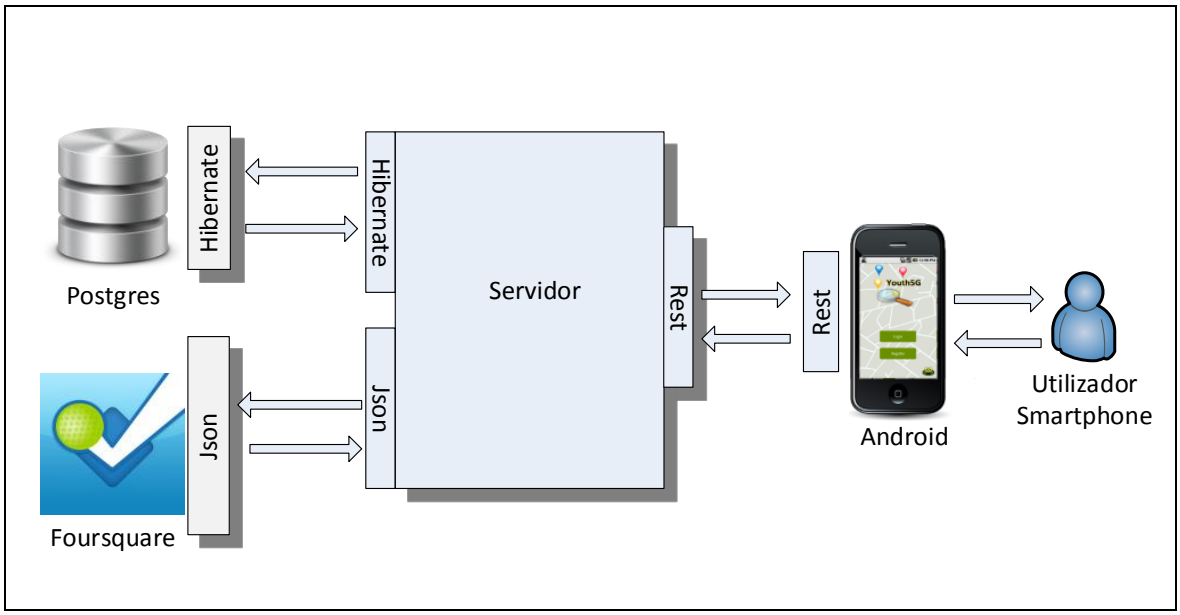

Figura 12 - Visão geral do protótipo Youth5G

O servidor está implementado usando Java Beans a correrem num servidor Apache Tomcat e é o responsável pelo processamento das respostas aos pedidos originários na aplicação ubíqua. Essa troca de informação é feita através de Webservices REST, que realizam a autenticação dos utilizadores e respondem aos pedidos, com base na informação que se encontra na base de dados e na API Foursquare. As trocas de informação entre o servidor e a aplicação ubíqua bem como entre o servidor e a API Foursquare são feitas utilizando o formato JSON. Toda a informação necessária ao funcionamento do sistema Youth5G é guardada na base de dados PostgeSQL da aplicação, estando a comunicação com esta implementada utilizando a biblioteca Hibernate e objectos DAO.

A aplicação servidor está construída segundo um modelo de modularidade com todos serviços a serem independentes, o que permite adicionar ou remover serviços sem comprometer funcionalidades já existentes.

4.2. Aplicação Servidor HappyHour

Como a base a ser utilizada foi o servidor Youth5G começou-se por estudar as funcionalidades que este oferecia e quais seriam uteis para o projecto HappyHour. Todos os Webservices de *"rating"* e *"feedback"* ,bem como algumas bibliotecas, foram eliminadas. A nível de estrutura de comunicação relacionada com POIs e autenticação dos utilizadores foi mantida a estrutura. Relativamente à camada de acesso à base de dados algumas alterações tiveram de ser efectuadas de forma a comportar novas informações que eram necessárias guardar em tabelas já existentes, bem como criar novos DAOs para acesso às novas tabelas.

A pesquisa de informações dos POIs no Foursquare usava a primeira versão da API java v2 (1.0.0) e, apesar de ainda ser compatível com a plataforma actual, esta API limitava certas funcionalidades de pesquisa. Optou-se, então, por proceder à sua substituição pela versão 1.0.2 dessa mesma API mas, após estar implementada, verificou-se que alguns dos campos de pesquisa necessários não estavam ainda funcionais. Foi, portanto, necessário fazer algumas alterações ao código fonte da API produzindo assim uma versão melhorada da mesma.

O sistema de gestão de eventos implicou a criação de uma nova tabela na base de dados que regista todos os detalhes dos eventos bem como a que ponto de interesse estão associados. Como os Webservices relativos aos POIs são utilizados tanto pela aplicação móvel como pela aplicação WEB, foi decidido implementar serviços que sustentem as duas ao invés de criar serviços separados. Foram criados Webservices para obter uma lista de todos os eventos (getListEvent) e os detalhes de um evento específico (getDetailEvent). De forma a aplicar as alterações feitas pelos administradores na aplicação WEB foram criados serviços que permitam criar (setNewEvent), editar (setEvent) ou eliminar eventos (removeEvent). Como a aplicação WEB utiliza algumas chamadas em JSONP foi necessário implementar métodos GET para além dos POST normalmente utilizados pela aplicação Android.

Para o serviço de notificações foram abordadas duas ideias distintas, a possibilidade de utilizar o serviço GCM (Google Cloud Messaging [51]) ou desenvolver um sistema próprio. Apesar de ter a vantagem de as notificações serem instantâneas o GCM implicaria um maior custo energético nos dispositivos móveis pois seria mais um serviço a correr no sistema operativo e mais uma ligação de dados a estar activa. Como a implementação da aplicação móvel implica o envio de dados de posição e acelerómetro para o servidor optou-se por incluir as notificações na resposta a este envio e assim aproveitar a ligação já estabelecida.

Foram criadas três novas tabelas na base de dados para acolherem as notificações não enviadas (unsent_notification), as que estão a ser enviadas (pool_notification) e as já enviadas (sent_notification). Ao servidor foi adicionada uma *"thread"* (pooler) que a cada cinco minutos verifica se há notificações para serem enviadas e, caso existam, move-as para a tabela pool_notification de modo a que fiquem disponíveis para envio. Em simultâneo essa mesma *thread* verifica se alguma das notificações a serem enviadas já estão na tabela há mais de trinta minutos e, move-as para a tabela sent_notification com o campo entregue a *"true"*. Este campo tem um papel importante pois, na eventualidade de um período de inactividade do servidor, permite informar os donos dos estabelecimentos se as notificações foram ou não entregues. Se a *thread* encontrar notificações na tabela unsent_notification com uma data para envio com mais de trinta minutos para a hora actual move-as directamente para a tabela sent_notification com o campo entregue com o valor *"false"*.

O Webservice criado para receber os dados de posição e acelerómetro (SetClosePOIs) passou, assim, a servir uma dupla função. Quando o servidor recebe uma lista com os POIs que estão perto do utilizador, verifica se algum destes tem notificações na tabela

pool_notification e, em caso afirmativo, retorna uma lista com todas as notificações ou uma lista vazia caso não existam.

Para interação com a aplicação HappyHour Management tiveram de ser criados cinco novos Webservices relativos às notificações. Como foi utilizada uma comunicação por JSONP foram implementados métodos POST à semelhança dos serviços para gestão de eventos. De forma a mostrar informação ao utilizador foi criado um Webservice para obter uma lista de todas as notificações (getListNotification) e outro para os detalhes de uma notificação específica (getDetailNotification). A nível de operações do utilizador sobre as notificações foram criados serviços que permitam criar (setNewNotification), editar (setNotification) ou eliminar notificações (removeNotification).

Sendo necessário guardar informações relativas aos dados de posição e acelerómetro dos utilizadores foi criada uma nova tabela (userstatus) que guarda em que POI se encontra o utilizador e quais os valores de acelerómetro enviados. Cada linha desta tabela contém um *"timestamp"* que marca o momento da inserção de forma a poderem ser purgados no caso da aplicação Android deixar de comunicar com o servidor. À tabela pointofinterest, que contém os dados de cada POI, foi acrescentada duas novas colunas que indicam a afluência e a agitação de um estabelecimento. Aproveitando a existência da "*thread*" pooler criada para as notificações foram-lhe acrescentadas funcionalidades ficando esta também responsável por remover valores da tabela userstatus com mais de trinta minutos e por recalcular o grau de afluência e de agitação de cada espaço.

4.3. Aplicação WEB HappyHour Management

De forma a possibilitar a gestão de todas as funcionalidades do serviço HappyHour aos donos de estabelecimentos, foi desenvolvida uma aplicação web integrada na plataforma Facebook chamada "HappyHour Management". Foi decidido integrar esta aplicação nos serviços prestados pelo Facebook, uma vez que a torna mais acessível pelos utilizadores responsáveis pelos estabelecimentos de entretenimento.

Para além da simplicidade de utilização a integração com o Facebook foi pensada de forma a ter mais garantias de que quem está a aceder, é efectivamente o responsável de um dado POI, e a proporcionar algumas funcionalidades extra, como por exemplo, a partilha de eventos também na página da empresa nesta rede social.

A aplicação HappyHour Management foi desenvolvida usando HTML, Javascript e PHP. Para o funcionamento da aplicação foram incluídas bibliotecas Javascript DOJO [52] que tratam de efeitos, animações e apresentação de listagens de elementos. Com a utilização deste sistema a comunicação com o servidor teve de ser feita através do próprio *dojo*, o que impossibilita a utilização de JSON devido a serem pedidos cross-domain. Como tal os pedidos tiveram de ser feitos utilizando o método GET do HTTP com mensagens JSONP. O método GET não tem nenhum mecanismo de segurança previamente implementado pelo que a ligação teve de ser alterada para HTTPS de forma a garantir a confidencialidade dos dados. Toda a restante comunicação entre a aplicação e o servidor foi implementada usando PHP com métodos POST.

Tendo como base o sistema de login da aplicação Youth5G foram feitas alterações que permitam que um utilizador seja marcado como administrador de um ou mais POIs. Desta forma é permitido que o mesmo login que é usado na aplicação WEB para gestão de estabelecimentos possa ser utilizado na aplicação móvel para utilização corrente.

O sistema de registo da aplicação foi a parte mais complicada de implementar uma vez que se procurou minimizar os casos de falsas identidades em que alguém, que não seja o dono do estabelecimento, se consegue registar como administrador do mesmo. Optou-se por um sistema em que o utilizador tem de se registar usando uma conta que tenha privilégios de administração da página Facebook do estabelecimento pretendido de forma a cruzar estes dados com os obtidos através do Foursquare. Para se conseguir aceder a estes dados, a primeira vez que o utilizador iniciar a aplicação, ser-lhe-á pedido para aceitar as permissões necessárias de acesso à sua conta Facebook e apenas após as aceitar terá acesso à aplicação.

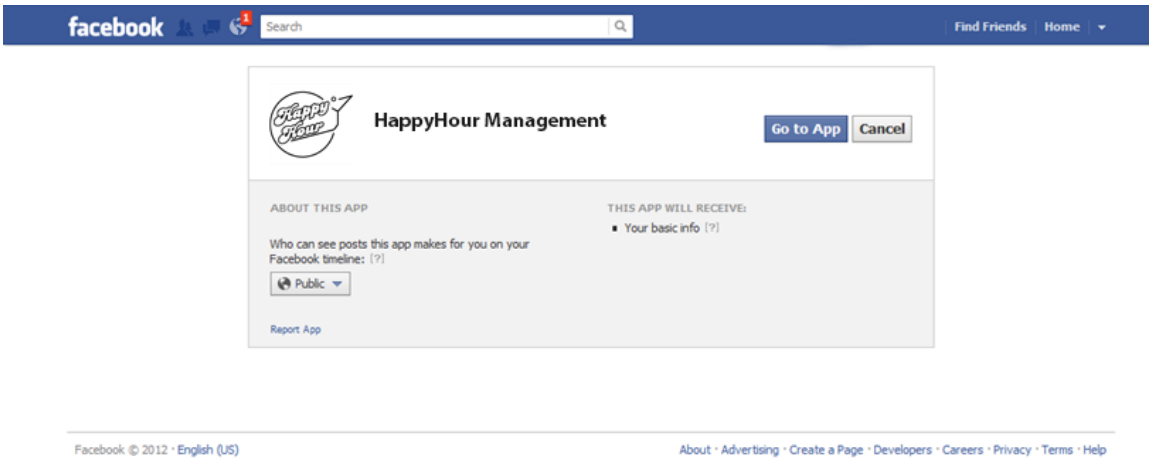

Figura 13 - Aceitação de permissões da aplicação HappyHour Management no Facebook

O Facebook faz o redireccionamento automático para o endereço da aplicação HappyHour Management, e a página apresentada é uma página de "boas vindas" onde o utilizador pode registar-se ou ligar-se com uma conta criada previamente. Quando o utilizador faz login pela primeira vez é direcionado para uma secção de gestão de POIs onde lhe é fornecida uma lista obtida através de um pedido ao Facebook de todas as páginas de que o utilizador é administrador. Este pedido não passa pelo servidor HappyHour e é feito directamente através de PHP uma vez que é necessário o *"token"* de autenticação do utilizador que apenas está disponível como variável de sessão. Após escolher a página que pretende, é feita uma pesquisa através da API do Foursquare no servidor onde se verifica se existe algum POI com o mesmo nome e morada. Em caso afirmativo é pedido ao utilizador para confirmar e fica registado como administrador daquele estabelecimento.

Figura 14 - Gestão de POIs Aplicação HappyHour Management

Caso a aplicação não consiga devolver um POI válido são oferecidas soluções consoante o motivo do erro. Se o POI já estiver atribuído a outra conta de administrador, o utilizador pode contactar a HappyHour com provas de ser o dono do estabelecimento para que a situação seja corrigida. Se o POI não for encontrado com base nos dados fornecidos pelo Facebook, a aplicação pede ao utilizador que confirme que as informações que estão registadas na página do seu estabelecimento coincidem com as que estão inseridas no Foursquare. Após o utilizador corrigir estas informações pode tentar novamente o processo de registo e se, mesmo assim, o sistema não encontrar correspondências, é-lhe sugerido o contacto com os responsáveis do HappyHour.

Se o mesmo utilizador for o dono de vários estabelecimentos deve repetir este procedimento para cada um deles, ficando todos eles associados a uma única conta na plataforma HappyHour. O utilizador pode aceder à aplicação a partir de qualquer uma das contas Facebook, podendo no menu principal do HappyHour escolher qual o estabelecimento que pretende utilizar.

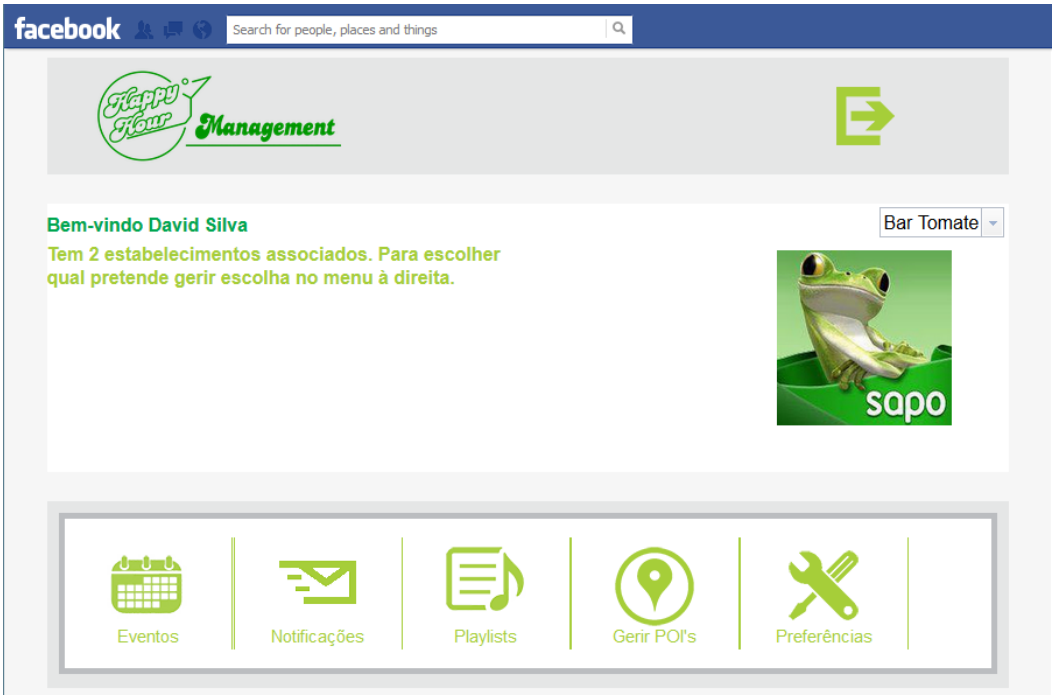

Figura 15 - Menu da aplicação HappyHour Management

O menu principal está associado à manutenção de eventos e notificações do sistema, e encontra-se estruturado da seguinte forma:

- Eventos – Abre uma lista de todos os eventos associados a um estabelecimento os quais podem ser editados ou eliminados. Nesta vista podem ser também criados novos eventos;
- Notificações – Permite agendar notificações para estas serem enviadas a todos os utilizadores da aplicação HappyHour Android numa determinada data e hora, ou cancelar ou editar as mesmas antes de serem enviadas;
- Opções – O utilizador pode desassociar a sua conta HappyHour de um POI, acrescentar outros POIs ou consultar as opções de ajuda;
- Estabelecimentos – O utilizador pode escolher qual o estabelecimento que pretende através de um menu "*dropdown*".

As páginas de gestão de eventos e de notificações foram criadas seguindo um modelo de usabilidade, permitindo ao utilizador de forma intuitiva ter acesso a toda a informação e editá-la facilmente. Assim, as listagens aparecem à esquerda e sempre que utilizador selecciona um elemento a aplicação abre um novo elemento à direita onde os detalhes podem ser editados.

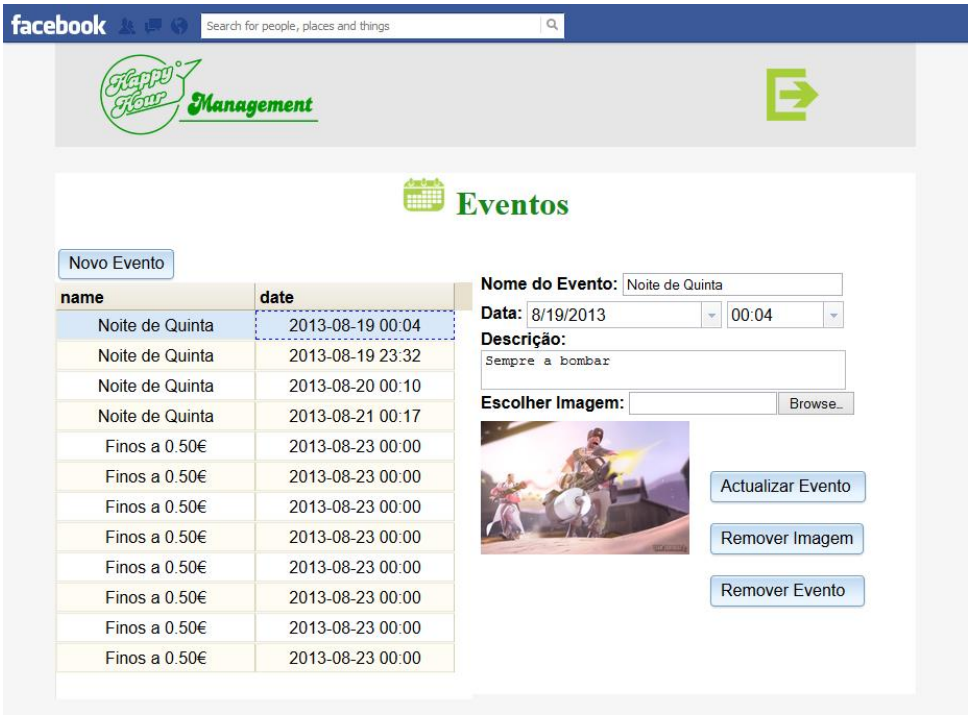

Figura 16 - Gestão de Eventos Aplicação HappyHour Management

No caso de o utilizador pretender criar um novo evento ou notificação basta seleccionar essa mesma opção e preencher os dados no mesmo formulário que usa para edição.

4.4. Aplicação Android HappyHour

Com a redistribuição de tarefas ocorrida no inicio do segundo semestre devido à entrada de novos elementos na equipa, a minha participação no desenvolvimento desta passou a ser de *"debug"*, desenvolvimento de algoritmos e discussão das informações a serem comunicadas ao servidor ao invés de criar código.

Tomando como base a aplicação Android Youth5G, foram removidas todas as funcionalidades que não eram relevantes para a nossa aplicação como *ranking*, *feedback* e *refresh* manual do mapa. A secção de ajuda foi alterada bem como o sistema de navegação até aos POIs.

Durante o desenvolvimento da aplicação Android o mapa fornecido pela API do Google Maps Android deixou de aparecer no ecrã e após alguma pesquisa descobriu-se que a "key" que permita utilizar a API tinha expirado. Quando se tentou obter uma extensão da licença verificou-se que esta versão da API tinha sido descontinuada em Dezembro de 2012 pelo que não seria possível obter uma "key" que substituísse a antiga. Para se obter uma nova licença foi necessário actualizar a API para a versão V2. Mas, infelizmente, apesar de ser dada uma ideia de continuidade, a nova versão nada tem a ver com a anterior. Foi portanto necessário reescrever toda a parte de apresentação do mapa e marcadores bem como todas as funcionalidades de interacção com os mesmos.

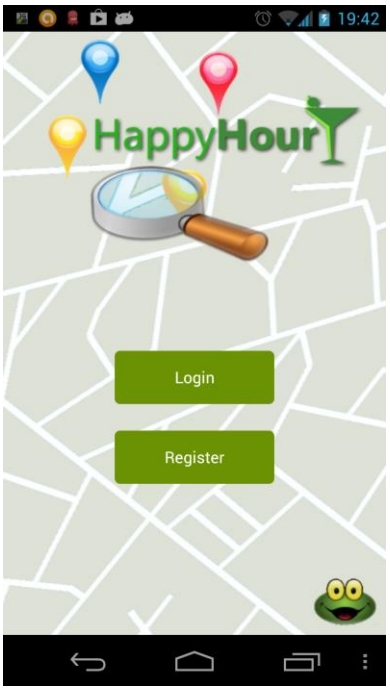

Figura 17 - Aplicação Android HappyHour - Splash Screen

Para não aparecerem demasiados POIs em simultâneo no ecrã quando se está com um nível de *zoom* reduzido optou-se por fazer "*clustering*" quando estes estão próximos, como se pode ver na Figura 18. Desta forma aumenta-se a usabilidade do mapa evitando a saturação da informação fornecida utilizador. Foi também implementada uma optimização nos pedidos de POIs ao servidor variando estes consoante o nível de *zoom*. Assim diminuiu-se a quantidade de informação em cada pedido bem como o número de pedidos caso as informações já estejam disponíveis na memória do dispositivo móvel.

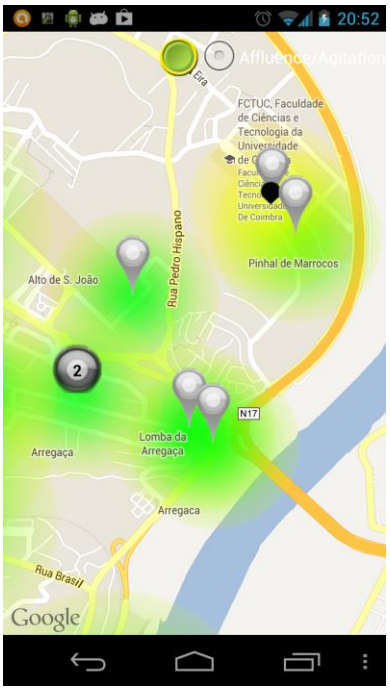

Figura 18 - Aplicação Android HappyHour - Clustering

Para apresentar informação sobre afluência e agitação nos POIs foi criada uma opção que permite adicionar uma *layer* ao mapa. Esta *layer* contém um *heatmap* que pode representar o grau de afluência ou de agitação de um local consoante a escolha do utilizador, como pode

ser visto na Figura 19. A cor do *heatmap* difere consoante certos níveis sejam atingidos numa escala de 1 a 3, em que 1 é representado pela cor verde, 2 é representado pela cor amarela e 3 é representado pela cor vermelha. No desenvolvimento desta *layer* começou-se por usar a função de desenho de círculos do Google Maps mas verificou-se que o impacto na performance era demasiado elevado. Assim optou-se por aplicar imagens PNG com círculos ao redor dos POIs, o que implica um menor custo computacional. Como a medida dos círculos é em metros o seu tamanho variava com o nível de *zoom*. Para evitar esta situação foi feito um aumento proporcional dos círculos consoante a variação do *zoom*.

Figura 19 - Aplicação Android HappyHour - Heatmap

Como se pretendia que a leitura dos valores de GPS e acelerómetro, bem como a recepção de notificações, pudessem ser executados sem a aplicação estar activa, esta foi implementada através de um serviço independente, o "NotificationService". Ao correr numa *thread* à parte o serviço não pára de ser executado quando a aplicação é encerrada, mas sim apenas quando o utilizador escolhe essa opção. A leitura dos dados é feita através de dois *Listeners*, um de localização e outro de agitação.

Para leitura dos valores de GPS foi utilizado o método "onLocationChange" do "LocationListener" que é parte integrante dos serviços de localização do sistema operativo Android. Quando a localização do utilizador muda, esta é guardada memória e substitui o valor anterior. No caso do acelerómetro foi criado um novo *Listener* que acorda a cada 100 segundos para fazer uma leitura dos valores de acelerómetro durante 5 segundos.

Após um período de espera de 300 segundos o "NotificationService" calcula uma média entre as três leituras dos valores de agitação que, juntamente com o último valor de posição, são enviados para o servidor. Na resposta a esta mensagem são recebidas, caso existam, as notificações para os POIs ao redor do utilizador sendo o "NotificationService" responsável por mostrar as mesmas ao utilizador através de uma chamada ao sistema de notificações do sistema operativo Android.

Como referido no início da secção o sistema de navegação até um determinado POI foi alterado uma vez que não foi possível convertê-lo para a versão 2 da GoogleMapsAPI. Quando o utilizador escolhe nos detalhes de um POI a opção "Go to POI", é enviado um

pedido ao sistema operativo para começar uma "*navigation activity*" e este abre o programa de navegação predefinido pelo utilizador no seu equipamento com o POI como destino.

Figura 20 - Aplicação Android HappyHour - Go to POI

4.5. Testes

Para a garantir o correcto funcionamento da plataforma HappyHour foram feitos testes após cada iteração. De cada vez que uma funcionalidade era dada como totalmente concluída todas as funcionalidades prévias eram testadas tanto na aplicação WEB como na aplicação Android. Desta forma foi possível descobrir alguns erros em módulos que não pensávamos serem afectados.

O facto de os testes serem constantes ao longo do tempo permitiu também encontrar erros que apenas se dão num período de tempo específico, nomeadamente o facto de as tabelas que estávamos a usar tanto para eventos como notificações não terem em consideração os ajustes da hora segundo o fuso horário. Isto levava a que durante o período de inverno as notificações e eventos não apareciam à hora correcta. A integridade dos dados na base de dados foi também tida em atenção e, de cada vez que um novo serviço que alterasse valores nas tabelas foi criado, as tabelas eram inspeccionadas para garantir que os dados estavam a ser inseridos segundo os padrões previstos.

Para testar o sistema de localização da aplicação Android foi criado um ponto de interesse na sala do laboratório G6.2 ficando um dos elementos a verificar os dados recebidos no servidor e dois elementos da equipa foram para o exterior de forma a verificar que, dentro de um raio de 700 metros, as notificações eram recebidas. Os testes foram feitos usando 2 equipamentos Android, um com a versão 4.0.4 e outro com a 4.1.2, ambos com a opção de localização por GPS activada de forma a reduzir a margem de erro. Ambos os dispositivos receberam as notificações quando os telemóveis assumiam estar a menos de 700m do POI e apareciam como estando dentro do ponto a menos de 150m.

Os testes de acelerómetro foram efectuados no laboratório utilizando dois equipamentos Android equipados com acelerómetro. A aplicação foi executada em ambos e foram testados os vários níveis de agitação. Começou-se por verificar os casos em que o utilizador está parado ou a utilizar o telemóvel normalmente na mão. Posteriormente testou-se o caso de uma pessoa estar a andar em passo normal, e por fim, casos de grande movimentação como estar a dançar ou a correr. Após verificação dos dados obtidos em cada leitura chegou-se à conclusão que os patamares de agitação necessitavam ser revistos. Após se corrigirem os valores os testes foram executados novamente tendo sido obtidos resultados satisfatórios.

4.6. Desafios encontrados

O facto de começar o projecto com quase um mês de atraso face ao planeado acabou por colocar alguns obstáculos no início, uma vez que o estudo da aplicação Youth5G teve de ser mais teórico baseado na documentação e não tanto um estudo exploratório com uma análise mais aprofundada da implementação existente na aplicação. Esta diferença na abordagem do estudo torna-se mais notória quando é necessário desenvolver novos módulos para integrar na aplicação existente, uma vez que, apesar de saber como esta funciona, algumas especificações principalmente a nível das mensagens trocadas, requerem estudo e experimentação.

A nível da API SAPO Mapas foram encontrados bastantes problemas na sua implementação, maioritariamente por falta de documentação disponível. Como só foi possível implementar algumas das suas funcionalidades - as quais não eram relevantes para o nosso projecto - e com as vantagens apresentadas pela API do Foursquare, optámos por continuar a utilizar esta última.

Quanto à aplicação HappyHour Management, como foi referido anteriormente, a sua integração com o servidor já existente revelou alguns problemas a nível de comunicação, principalmente devido à necessária segurança na troca de mensagens. Esta teve de ser implementada através de JSONP uma vez que os "*browsers"* interpretam os pedidos entre o servidor Apache e o servidor Apache Tomcat como "*cross-domain"*, apesar de estarem na mesma máquina. Os pedidos "*cross-domain"* são normalmente bloqueados por razões de segurança eliminando qualquer tipo de comunicação. A única forma de contornar esta limitação é utilizando mensagens JSONP. No entanto, estas estão limitadas ao método GET que não implementa qualquer tipo de encriptação, sendo portanto necessário alterar a ligação ao servidor de forma a usar HTTPS. A configuração do HTTPS, apesar de ser um processo relativamente simples, revelou alguns problemas com a validação dos certificados e os pedidos não estavam a chegar ao servidor. Após algumas alterações na forma como é estabelecida a ligação HTTPS conseguimos estabelecer uma comunicação segura entre o cliente web e o servidor.

A nível de disponibilização de dados ao utilizador tivemos também alguns problemas com as bibliotecas DOJO em que algumas funções de apresentação de tabelas que funcionavam correctamente numa página de teste se mostrarem inutilizáveis quando implementadas na nossa aplicação.

Durante os testes, após a introdução dos serviços de notificações no servidor, verificou-se que estas estavam desfasadas uma hora da hora actual. Após alguma pesquisa verificou-se que as tabelas não estavam a suportar as alterações de hora no fuso horário e quando ia ser feita uma pesquisa na base de dados as notificações apareciam como mais antigas e eram marcadas como não enviadas. Este problema apenas se revela no horário de Inverno pelo que se a aplicação fosse desenvolvida durante o horário de Verão provavelmente não seria

detectado. Para superar esta situação optou-se por converter a data actual para a *timezone* da tabela e assim fazer a pesquisa.

A API Foursquare V1 utilizada no servidor encontrava-se desactualizada pelo que foi necessário proceder à sua actualização para suportar as novas funcionalidades. Infelizmente mesmo a última versão (V2 1.0.2) ainda não incluía as funcionalidades necessárias para limitar as pesquisas como *"radius"* ou localidades perto. Foi então necessário recorrer ao código fonte da API, que está disponível como *"open source"*, e fazer as alterações necessárias. Ao tentar compilar novamente a API verificou-se que esta necessitava de uma grande variedade de bibliotecas pelo que foi necessário encontrar cada uma de modo a que a API ficasse compilada correctamente.

O processo de conversão da aplicação Android de modo a suportar a versão 2 da API do Google Maps não estava prevista e a sua adaptação ocupou uma grande quantidade de tempo. Para além do custo temporal de reescrever todo o código relacionado com a interface gráfica tem também de se ter em contra o tempo necessário na aprendizagem do funcionamento desta nova API que em nada se assemelhava à anterior.

No trabalho desenvolvido pelos alunos de LC optou-se por criar um *"heatmap"* para mostrar os dados referentes à agitação e afluência usando circunferências desenhadas à volta dos POIs. Depois de alguns testes verificou-se que o desenhar círculos revela-se muito pesado computacionalmente o que levou a uma grande quebra de performance na apresentação do mapa. Para evitar essas quebras optou-se por aplicar imagens com círculos que são menos pesadas computacionalmente.

Capítulo 5
Plano de Trabalho e Implicações

5.1. Planeamento

O plano de trabalho proposto para o projecto HappyHour consistiu nas seguintes tarefas definidas no início do semestre:

1- **Estudo da Ferramenta YOUTH5G** – Estudo da arquitectura da ferramenta YOUTH5G e todos os seus componentes, previamente desenvolvidos pelo aluno Duarte Raposo.

   a. **Tempo estimado** – 1 mês

   b. **Início** – 15 Outubro 2012

   c. **Prazo** – 15 Novembro 2012

2- **Implementar uso de POIs com os serviços SAPO** – Nesta etapa tentou-se adaptar o sistema de forma a utilizar o serviço de "Points of Interest" da SAPO. Caso tal não seja possível, será novamente usada a API do Foursquare.

   a. **Tempo Estimado** – 1 mês

   b. **Início** – 15 Novembro 2012

   c. **Prazo –** 15 Dezembro 2012

3- **Implementar mecanismos de introdução de eventos associados a um estabelecimento por parte dos donos dos estabelecimentos / bares.**

   a. **Tempo Estimado** – 1 mês

   b. **Início** – 15 Dezembro 2012

   c. **Prazo** – 20 Janeiro 2013

4- **Implementação do sistema "*push-notifications*"** – Este sistema permitiu a divulgação de promoções e actividades em tempo real a utilizadores que se encontrem nas imediações do local

   a. **Tempo Estimado** – 1 mês

   b. **Início** – 20 Janeiro 2013

   c. **Prazo** – 20 Fevereiro 2013

5- **Integração do sistema com Facebook** – A integração da aplicação HappyHour com o Facebook será feita com o objectivo de tentar utilizar os gostos do utilizador para melhorar a eficácia das "*push-notifications*". Também proporcionará à SAPO uma base de dados completa com perfis, gostos e comportamento de utilizadores.

   a. **Tempo Estimado** – 2 meses

   b. **Início** – 20 Fevereiro 2013

   c. **Prazo** – 20 Abril 2013

6- **Implementar as funcionalidades de grau de afluência**

   a. **Tempo Estimado** – 2 meses

   b. **Início** – 20 Abril 2013

   c. **Prazo** – 20 Junho 2013

7- **Implementar um algoritmo para estimar o nível de agitação de um local**

   a. **Tempo Estimado** – 1 mês

   b. **Início** – 20 Junho 2013

   c. **Prazo** – 20 Julho 2013

8- **Integrar o HappyHour com o serviço SAPO.MusicBox**

   a. **Tempo Estimado** – 1 mês

   b. **Início** – 20 Julho 2013

   c. **Prazo** – 20 Agosto 2013

9- **Testes e Melhorias**

   a. **Tempo Estimado** – 1,5 meses

   b. **Início** – 20 Agosto 2013

   c. **Prazo** – 30 Setembro 2013

5.2. Redistribuição de Tarefas

Durante o decorrer do projecto novos membros integraram a equipa responsável pela aplicação HappyHour, o que levou a alterações no planeamento e distribuição de tarefas. A entrada de novos elementos implicou a criação de um guia de desenvolvimento que serviu como peça importante na ambientação de novos colaboradores.

Em Janeiro de 2013 o aluno de Licenciatura João Pratas integrou o projecto em regime de voluntariado. Como o seu objectivo era adquirir conhecimentos na programação para Android foi-lhe atribuído o desenvolvimento do sistema de apresentação de notificações do lado da aplicação móvel, tanto dentro da aplicação como no sistema operativo.

No âmbito da cadeira Laboratório de Comunicações pertencente ao plano curricular do Mestrado de Engenharia Informática foram disponibilizadas duas vagas a alunos para colaboração neste projecto no início do segundo semestre. Estas vagas foram preenchidas pelos alunos Bruno Antunes e Elói Almeida que ficaram encarregues de desenvolver o sistema de leitura e tratamento de valores de GPS e acelerómetro na aplicação Android.

Com a entrada destes elementos foi decidido que as restantes tarefas de desenvolvimento da aplicação Android bem como alterações necessárias à base de dados ficariam a cargo do aluno de doutoramento David Nunes. A finalização da aplicação WEB bem como desenvolvimento do servidor com a criação de novos serviços ficaram a meu cargo. Esta divisão de tarefas foi decidida tento em conta o à vontade com cada um dos módulos que cada elemento da equipa tinha.

5.3. Mapa de Gantt do 1º Semestre

Na **Error! Reference source not found.** pode ser observado o mapa de Gantt do primeiro semestre. A vermelho estão assinalados os períodos de tempo previstos para cada tarefa e a verde o tempo que cada uma destas ocupou na realidade.

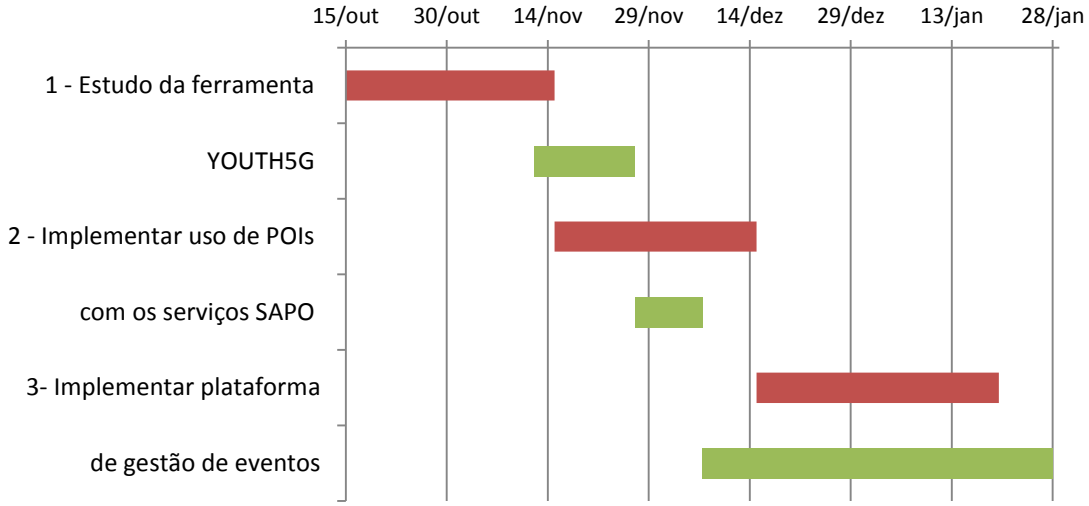

Figura 21 - Mapa de Gantt do 1º Semestre

5.4. Mapa de Gantt do 2º Semestre

Na Figura 22 está especificado o mapa de Gantt para o 2º Semestre com o planeamento do tempo ocupado por cada tarefa assinalado a vermelho.

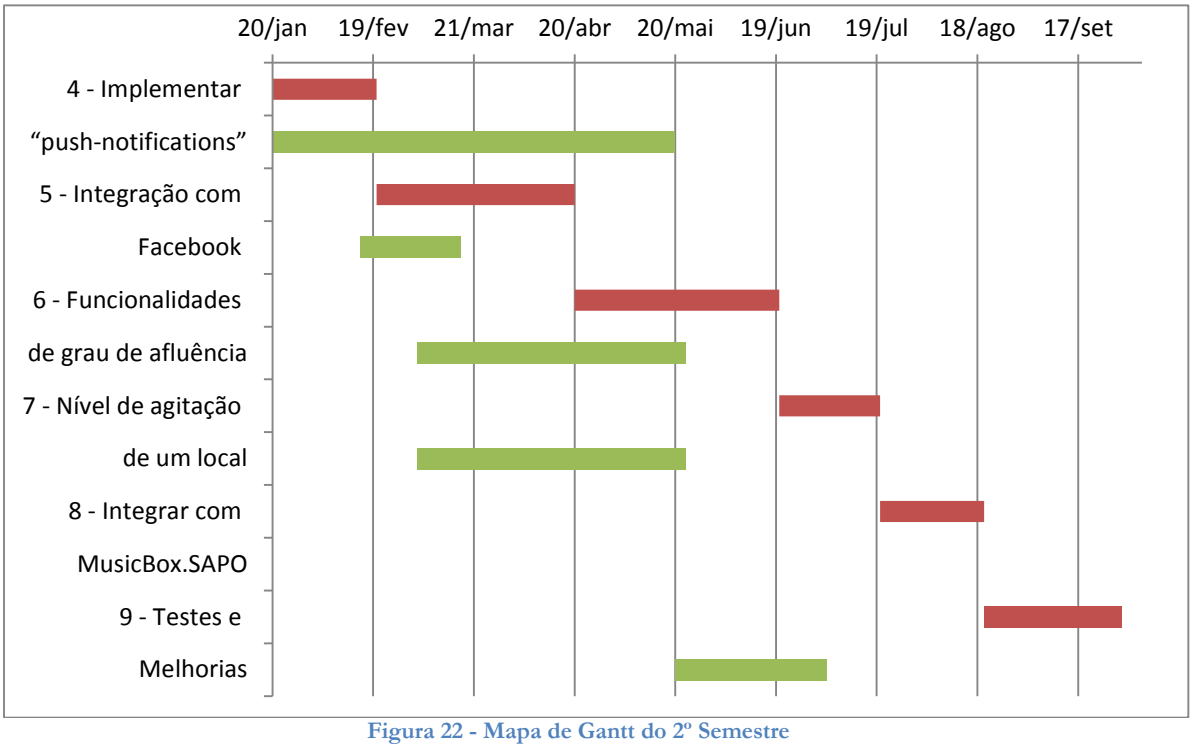

Figura 22 - Mapa de Gantt do 2º Semestre

5.5. Desvios

Com a minha integração tardia no projecto, que só acabou por se verificar no dia 12 de Novembro, o projecto sofreu imediatamente um atraso inicial. Felizmente, devido a já possuir alguns conhecimentos da tecnologia utilizada e a ter dedicação exclusiva ao projecto permitiu que houvesse uma recuperação face a este cenário inicial.

A segunda fase foi concluída no dia 6 de Dezembro de 2012, ou seja, com 10 dias de avanço face ao inicialmente estipulado. Este "*buffer*" de tempo ganho veio a revelar-se crucial mais à frente uma vez que a terceira tarefa que tinha uma carga de trabalho de um mês demorou mais do que o previsto devido a alguns atrasos na criação e integração da aplicação web HappyHour management com o servidor. Como esta aplicação é também responsável pela criação das "*push-notifications*" optou-se por implementar também a tarefa 4 em simultâneo pelo que o atraso registado não se revelou preocupante.

Durante o desenvolvimento foi dado a conhecer à equipa um concurso fomentado pela empresa Ericsson, o *Application Awards* [53]. O tema para o concurso deste ano foi o desenvolvimento de aplicações para a vida na cidade, área onde se insere a aplicação HappyHour, pelo que decidimos participar. Apesar de a participação ter requerido algum tempo para preparação de documentação e criação de um vídeo [54] optou-se por não a incluir no plano de trabalho pois não afectou o desenvolvimento.

A entrada de novos elementos levou a alguns desvios pois foi necessário criar guias de documentação para os novos elementos bem como despender tempo em reuniões com os mesmos para os colocar a par do que estava a ser feito. O tempo dispendido nestas tarefas acabou por atrasar o desenvolvimento das tarefas 4 e 5 mas foi compensado pois com o aumento do número de elementos da equipa as tarefas 6 e 7 decorreram em paralelo com as anteriores terminando antes do previsto. A mudança de versão da API Google Maps acabou por atrasar a implementação das tarefas 4, 6 e 7 que dependiam da aplicação Android. O impacto apesar de notório acabou por não se revelar relevante no plano geral uma vez que, como referido no ponto anterior, a entrada de novos elementos acelerou o processo de desenvolvimento dessas tarefas.

Em reunião com os responsáveis pelo SAPO.Labs fomos informados que o serviço MusicBox.Sapo não é desenvolvido por nenhuma empresa pertencente ao grupo PT Comunicãoes mas sim por uma empresa subcontratada – nmusic. Com este cenário fomos informados que o Sapo.Labs não dispõe de ferramentas que nos permitam integrar a aplicação HappyHour com a plataforma MusicBox.Sapo pelo que esta tarefa foi descartada. Como esta tarefa estava definida como *"should"* no documento de requisitos o impacto na versão final da aplicação HappyHour não foi relevante.

Capítulo 6
Considerações Finais e Trabalho Futuro

O projecto em que fui inserido foi aliciante tendo características que, mesmo não estando envolvido na equipa de desenvolvimento, me fariam ser um utilizador do produto final. Num balanço geral e tendo em conta os objectivos a que me propus posso considerar o resultado como bastante positivo.

Este estágio foi fundamental no desenvolvimento das minhas capacidades de programação, organização e gestão de projectos. O desenvolvimento da plataforma HappyHour implicou a utilização de várias aprendizagens obtidas durante o meu percurso académico mas, acima de tudo, implicou um aprofundamento dos conhecimentos que tinha adquirido. A aplicação Youth5G previamente criada revelou-se um grande passo para facilitar a implementação. No entanto obrigou a um período inicial de estudo de forma a entender a maneira de pensar de quem a desenvolveu. "Reverse Engeneering" é algo não abordei durante o plano curricular pelo que foi uma boa experiência tentar pegar em algo sobre o qual pouco conhecemos e entendê-lo ao pormenor.

Como pontos positivos a salientar deste ano de trabalho consideraria a utilização de uma metodologia bem definida no desenvolvimento e a existência de metas que ao serem atingidas dão um sentimento de cumprimento a quem para elas trabalha. Como pontos menos positivos tenho apenas a referir que a entrada de novos elementos não estava precavida e foi necessário desenvolver nova documentação já preparada para novas entradas na equipa.

Apesar de os objectivos a que me propus no inicio do ano lectivo terem sido atingidos o desenvolvimento da aplicação HappyHour continuará, movido por novas ideias e conceitos que foram surgindo ao longo do desenvolvimento deste projecto. Como pontos fulcrais para a continuidade do projecto durante o próximo ano foram escolhidas as seguintes ideias:

- Implementação de funcionalidades na aplicação Web de gestão para partilha de eventos em várias redes sociais como Facebook, Google+ e Twitter;
- Criação de "*beacons*" *wireless* baseados em Android mini PCs capazes de fornecer funcionalidades de *checkin* automático e partilha de Internet para os utilizadores do serviço HappyHour;
- Utilização desses "*beacons*" para reconhecimento de músicas e inserção automática de *playlists* na base de dados de cada estabelecimento;
- Gráfico com estatísticas de agitação e afluência para os administradores;
- Implementação de mecanismos que minimizem a conectividade intermitente associada à transição de redes de dados móveis e redes wireless.

Para além destas ideias a serem implementadas outras oportunidades poderão surgir com o a migração para versões mais recentes do sistema operativo Android, bem como novas funcionalidades suportadas por melhorias das API's tanto da Google como do Foursquare. No plano de trabalhos está também agendada a participação numa feira de mostra de tecnologia organizada pela Sapo na qual vão ser demonstradas as potencialidades da plataforma HappyHour.

Por fim tenho de agradecer ao meu colega Eng.º David Nunes e ao professor Jorge Sá Silva que foram incansáveis e tudo fizeram para que este projecto fosse levado a bom porto.

Referências

[1] D. Raposo. (2012, September) Aplicações socialsense. LCT-WSN. [Online]. Available: http://www.youtube.com/watch?v=pkBVxjNNn5o&t=3m20s

[2] M. Weiser, "The computer for the twenty-first century," *Scientific American*, vol. 265, no. 3, p. 94–104, 1991.

[3] (2012, December) Foursquare website. Foursquare. [Online]. Available: https://-foursquare.com/

[4] (2012, September) About foursquare. Foursquare. [Online]. Available: https://-foursquare.com/about/

[5] (2013) Facebook website. Facebook. [Online]. Available: http://-www.facebook.com/

[6] R. Kim. (2012, December) Facebook muscles in on yelp and foursquare with nearby upgrades. Gigaom. [Online]. Available: http://gigaom.com/2012/12/17/facebook-muscles-in-on-yelp-and-foursquare-with-nearby-upgrades/

[7] (2013) Facebook mobile. Facebook. [Online]. Available: http://-www.facebook.com/FacebookMobile

[8] T. A. Press. (2012, October) Number of active users at facebook over the years. Yahoo! Finance. [Online]. Available: http://finance.yahoo.com/news/number-active-users-facebook-over-years-214600186–finance.html

[9] J. Constine. (2012, December) Hands on with facebook nearby, a new local biz discovery feature that challenges yelp and foursquare. techcrunch. [Online]. Available: http://techcrunch.com/2012/12/17/facebook-nearby/

[10] S. Young. (2012, December) New ways for people to discover your business with nearby. Facebook. [Online]. Available: http://www.facebook-studio.com/news/item/new-ways-for-people-to-discover-your-business-with-nearby

[11] (2012, August) Waze. waze Mobile. [Online]. Available: http://www.waze.com/

[12] (2012) EnglishGetglue website. GetGlue. [Online]. Available: http://getglue.com/

[13] L. Dubois. (2010, October) Getglue: The foursquare of entertainment? [Online]. Available: http://www.inc.com/news/articles/2010/10/getglue-the-foursquare-of-entertainment-checkin.html

[14] (2012, November) Getglue has agreed to merge with viggle and create 4.5m users strong leader in social tv! GetGlue. [Online]. Available: http://blog.getglue.com/?p=11651

[15] J. Nededog. (2011, September) Getglue and directv partner for integrated on-screen check-in. Hollywood Reporter. [Online]. Available: http://www.hollywoodreporter.com/-news/getglue-directv-partner-integrated-screen-242268

[16] D. Chartier. (2011, March) Getglue for iphone gets foursquare, sports check-ins. Macworld. [Online]. Available: http://www.macworld.com/article/1158924/-getglue_foursquare_sports.html

[17] (2012, December) About yelp. Yelp. [Online]. Available: http://www.yelp.com/-about

[18] (2012, December) More places to yelp. Yelp. [Online]. Available: http://-www.yelp.com/locations

[19] J. Porter. (2008, June) Social design patterns for reputation systems: An interview with yahoo's bryce glass (part ii). bokardo. [Online]. Available: http://bokardo.com/-archives/social-design-patterns-for-reputation-systems-two/

[20] N. GONZALEZ. (2008, April) Yelp lets businesses fight back. Tech Crunch. [Online]. Available: http://techcrunch.com/2008/04/28/yelp-lets-businesses-fight-back/

[21] (2013, January) Gypsii products. GyPSii. [Online]. Available: http://-www.gypsii.com/products.php

[22] D. Nations. (2013, January) What is gypsii? About.com. [Online]. Available: http://-webtrends.about.com/od/profiles/fr/Gypsii-profile.htm

[23] (2013) Geocha website. GeoCha. [Online]. Available: http://www.geocha.com/

[24] (2012, August) Geocha: Start conversations with people around you. makeuseof. [Online]. Available: http://www.makeuseof.com/dir/geocha-start-conversations-with-people-around-you/

[25] (2012, August) Geocha: Online local chat tool. webwarehub. [Online]. Available: http://www.webwarehub.com/geocha-online-local-chat-tool/

[26] A. Saqib. (2012, July) Geocha: Location-based chat & information through interactive map. addictivetips. [Online]. Available: http://www.addictivetips.com/web/-geocha-local-chat-location-information-through-interactive-map/

[27] (2013) Geocha app. GeoCha. [Online]. Available: https://app.geocha.com/

[28] (2013) Localresponse website. LocalResponse. [Online]. Available: http://-www.localresponse.com/

[29] (2012, December) Twiter website. Twitter. [Online]. Available: https://twitter.com/

[30] (2012, December) Instagram website. Instagram. [Online]. Available: http://-instagram.com/

[31] A. Ha. (2012, October) Localresponse can now target ads based on historical tweets and check ins. TechCrunch. [Online]. Available: http://techcrunch.com/2012/10/10/-localresponse-historical-targeting/

[32] R. Kim. (2012, Março) Location-based service loopt bought for 43.4*m by green dot corp*. *Gigaom*. [*Online*]. *Available*: *http://gigaom.com*/2012/03/09/*location-based-service-loopt-bought-for*-43-4*m-by-green-dot-corp*/

[33] (2012, Julho) Loopt shutdown faqs and information. Loopt. [Online]. Available: https://loopt.zendesk.com/home

[34] (2013) buzzd website. buzzd. [Online]. Available: http://www.buzzd.com/

[35] (2012, September) Buzzd. CrunchBase. [Online]. Available: http://-www.crunchbase.com/company/buzzd

[36] (2013) buzzd. linkedin. [Online]. Available: http://www.linkedin.com/company/-buzzd

[37] (2011, April) Buzzd pivots to become localresponse, helping merchants make sense of check in data. BETABEAT. [Online]. Available: http://betabeat.com/2011/04/-localresponse-helps-merchants-master-check-in-data/

[38] J. Swartz. (2010, December) The latest from gowalla is worth checking out. USA Today. [Online]. Available: http://content.usatoday.com/communities/technologylive/-post/2010/12/the-latest-from-gowalla-is-worth-checking-out/1#.UOeS_G_ZZ8E

[39] D. Chartier. (2010, December) Gowalla 3.0 unites facebook, foursquare check-ins. Macworld. [Online]. Available: http://www.pcworld.com/article/212251/-gowalla_30_unites_facebook_foursquare_checkins.html

[40] J. Cabalona. (2012, March) Gowalla is officially shut down. [Online]. Available: http://mashable.com/2012/03/11/gowalla-shuts-down/

[41] M. Siegler. (2010, March) Whrrl 3 wants to kill farmville. not foursquare. not gowalla. farmville. TechCrunch. [Online]. Available: http://techcrunch.com/2010/03/10/whrrl-3/

[42] (2010) Whrrl. CrunchBase. [Online]. Available: http://www.crunchbase.com/-product/whrrl

[43] A. Tsotsis. (2011, April) Groupon acquires whrrl creator pelago. TechCrunch. [Online]. Available: http://techcrunch.com/2011/04/18/groupon-acquires-whrrl-creator-pelago/

[44] J. O'Dell. (2010, May) Brightkite brings together location sharing and group texting. Mashable. [Online]. Available: http://mashable.com/2010/05/03/brightkite-mobile-apps/

[45] ——. (2010, July) Brightkite takes branded badges to the next level. MAshable. [Online]. Available: http://mashable.com/2010/07/20/brightkite-badges/

[46] M. Siegler. (2010, December) Brightkite kills its check-in functionality to focus on group texting. TechCrunch. [Online]. Available: http://techcrunch.com/2010/12/10/-brightkite-check-in-dead/

[47] R. Wauters. (2011, December) Brightkite winds down, says it will come back with 'something better' (again). TechCrunch. [Online]. Available: http://techcrunch.com/2011/-12/20/brightkite-winds-down-says-it-will-come-back-with-something-better-again/

[48] E. Eldon. (2012, Março) How green dot will use loopt to go after mobile payments. [Online]. Available: http://techcrunch.com/2012/03/09/how-green-dot-will-use-loopt-to-go-after-mobile-payments/

[49] C. Tode. (2012, Março) Is it game over for mobile check-ins? Mobile Marketer. [Online]. Available: http://www.mobilemarketer.com/cms/news/social-networks/-12337.html

[50] (2013, January) Platform versions. Google. [Online]. Available: http://-developer.android.com/about/dashboards/index.html

[51] (2013) Google cloud messaging. Google. [Online]. Available: http://-developer.android.com/google/gcm/index.html

[52] (2013) Dojo toolkit. dojo. [Online]. Available: http://dojotoolkit.org/

[53] (2013, January) Ericsson application awards overview. ERICSSON. [Online]. Available: http://www.ericssonapplicationawards.com/overview

[54] D. Nunes. (2013) Happyhour movie. DEI UC. [Online]. Available: http://-www.youtube.com/watch?v=sl7Gj-ewuQk