Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Pedro José Santos Monteiro

Aplicação Android para sistema de Domótica

Junho de 2015

Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Pedro José Santos Monteiro

Aplicação Android para sistema de Domótica

Tese de Mestrado

Engenharia Electrotécnica - Energia e Automação Industrial

Professor Doutor Paulo Rogério Perfeito Tomé

Professor Doutor Daniel Filipe Albuquerque

Junho de 2015

“Nothing happens to anybody which he is not fitted by nature to bear.”

Marcus Aurelius

NOTA PRÉVIA

Esta dissertação deu origem ao artigo: Arquitetura de Sistema de Controlo de Domótica, (Monteiro, P., Tomé, P., & Albuquerque, D. (2015). Domotics Control System Architecture. Apresentado na 10th Iberian Conference, Aveiro, p6).

RESUMO

O Homem sempre procurou abrigo e proteção em cavernas que posteriormente se converteriam nas habitações atuais. Já nesse período de evolução, o Homem aprimorava as suas construções tendo em conta as suas necessidades. Com a evolução tecnológica ao nível dos materiais de construção, redes de comunicação entre outras, o Homem deu origem aos conhecidos "edifícios inteligentes", iniciando-se o processo de mudança do paradigma de habitação. A habitação perdeu a função de local de refúgio ou descanso para se transformar num local de lazer, bem-estar e repouso. Para acompanhar esta mudança de paradigma nasceu a Domótica permitindo transformar uma simples habitação numa habitação inteligente, através de sistemas de automação residencial.

Os sistemas de domótica possibilitam o controlo dos diversos elementos presentes numa habitação moderna, como por exemplo a iluminação, para além de permitir ao utilizador a configuração de macros ou tarefas. A evolução das tecnologias de comunicação, nomeadamente o aparecimento do smartphones e os tablets, possibilitou à domótica atingir um novo grau de flexibilidade até aqui inexistente. Este novo grau é possível uma vez que estes novos aparelhos têm a capacidade de executar o mesmo controlo e gestão através de aplicações, que era até então, efetuado pelos softwares computorizados, fornecidos na aquisição dos sistemas de domótica.

Esta dissertação propõe o desenvolvimento dum sistema de domótica, cuja arquitetura se divide em componente física e lógica. Relativamente à componente física, esta assenta em uma arquitetura distribuída, isto é, possui mais que uma consola de comando que comunicam entre si. No que diz respeito à componente lógica, esta assenta em uma aplicação para plataformas móveis (AndroidTM) que irá permitir o controlo e a gestão de toda o elemento físico do sistema de domótica.

A arquitetura do sistema provou ser flexível, pois a componente lógica (aplicação) consegue controlar e gerir o elemento físico. Quanto à aplicação, uma vez que utiliza Android que, tratando-se de uma plataforma com variadíssimas opções ao nível de desenvolvimento gráfico, permitiu a criação de interfaces simples e intuitivos para o utilizador. No que diz respeito à gestão do sistema, o utilizador poderá interagir com operações mais simples como a alteração de designações ou ícones, e opções mais complexas como a configuração de macros, onde é possível efetuar uma série de operações em cadeia.

ABSTRACT

The Man has always sought shelter and protection in caves that would later become the current dwellings. Already at this period of evolution, Man honed their constructions taking into account their needs. With the technological evolution in terms of building materials, among other communication networks, the man gave rise to the known "intelligent buildings", initiating the process of change in the housing paradigm. Housing lost the local function of refuge or rest to become a place of recreation, wellness and relaxation. To accompany this change of paradigm the Domotics was born allowing the transformation of a simple habitation into a smart housing through home automation systems.

The domotics' systems allow monitoring of various elements present in a modern dwelling, such as lighting, in addition to allowing the user with configuration of macros or tasks. The evolution of the communication technologies, including the arrival of smartphones and tablets, enabled the domotics reach a new level of flexibility until then non-existent. This new step is possible since these new devices have the ability to perform the same control and management through applications that were until then performed by computerized software provided in the acquisition of domotics.

This thesis proposes the development of a domotics' system, whose architecture is divided into physical and logical component. Regarding the physical component, it is based on a distributed architecture, i.e., has more than one control console which communicate with each other. With regard to the logic component, this is based on an application for mobile platforms (AndroidTM) that will allow the monitoring and the management of all the physical element of the domotics' system.

The system architecture proved to be flexible for the logic component (application) is able to control and manage the physical element. As for the application, since it uses Android which in the case of a platform with varied options in terms of graphical development has enabled the creation of simple and intuitive user interfaces. With regard to system management, the user can interact with simpler operations such as changing names or icons, and more complex options such as macros setting, where it is possible to perform a series of chain operations.

PALAVRAS CHAVE

Habitação Inteligente
Domótica
Automação Residencial
Android
ModBus
Dispositivos Móveis

KEY WORDS

Smart House
Domotics
Home Automation
Android
ModBus
Mobile Devices

AGRADECIMENTOS

Em primeiro lugar, um especial agradecimento aos meus orientadores, Professor Paulo Tomé e Professore Daniel Albuquerque, por todo o seu acompanhamento e dedicação ao longo desta dissertação, mesmo quando o seu desenrolar escapava às vossas áreas de conhecimentos.

Gostaria de agradecer à empresa Tecsisel, na pessoa do meu monitor, o Eng.º Rui Ribeiro, e ainda ao Eng.º Miguel Almeida pela vossa disponibilidade para tratarem de assuntos relacionados com esta dissertação e ainda por me facultarem os equipamentos necessários para a execução da mesma.

Deixo um agradecimento à direção do Mestrado em Engenharia Eletrotécnica, mais concretamente ao Professor Paulo Moisés pelo seu apoio ao longo destes anos e por ter disponibilizado o laboratório LIAERSE (Laboratório de Investigação Aplicada em Energias Renováveis e Sustentabilidade Energética).

Ao restante corpo docente do Departamento de Engenharia Electrotécnica, um muito obrigado por me terem dotado de competências intelectuais e profissionais através dos diversos trabalhos propostos no decorrer da licenciatura e mestrado que permitirão abordar os futuros desafios com uma perspetiva diferente que até tinha possuía.

Um grande agradecimento a todos os meus amigos, que diretamente e indiretamente contribuíram para o desenvolvimento desta dissertação, através de conhecimentos, apoio, motivação, principalmente nos períodos mais complicados.

Por fim, o meu maior agradecimento à minha família, especialmente aos meus pais e irmã, que sempre me apoiaram neste percurso de formação, nunca colocando entraves nas minhas escolhas, mesmo quando anteviam que podiam surgir adversidades.

ÍNDICE GERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

1. Introdução

Devido às inovações tecnológicas das últimas décadas o paradigma de habitação encontra-se neste momento em alteração. Atualmente uma habitação, não possui apenas a função de local de refúgio ou descanso, mas também ser um local de lazer, bem-estar e repouso. Para acompanhar esta mudança de paradigma nasceu a Domótica permitindo transformar uma simples habitação numa habitação inteligente.

De modo a se criar uma habitação inteligente, nesta dissertação vai-se desenvolver uma aplicação para dispositivos móveis que se interliga com uma arquitetura física automatizada, criando-se assim um sistema de domótica.

Neste capítulo é introduzido o conceito de domótica bem como as áreas em que esta pretende atuar. Posteriormente são apresentados os objetivos e o enquadramento desta tese, seguidos da estrutura deste documento. A secção 1.1 aborda o conceito de domótica. Esta abordagem irá permitir contextualizar o termo domótica mais detalhadamente no capítulo 2, de modo a se entender a associação do termo domótica a uma habitação inteligente. A secção 1.2 apresenta o enquadramento do tema desta tese, onde introduz-se o problema a resolver nesta dissertação. Na secção 1.3, expõem-se os objetivos para esta tese, tendo em conta os pressupostos apresentados na secção 1.2. Por fim, a secção 1.4 exibe a estrutura deste documento acompanhada de um pequeno resumo sobre os temas dos capítulos que compõem esta exposição escrita.

1.1 Conceitos Introdutórios

O conceito *"Domótica"* deriva da junção da palavra "*Domus*" (casa em Latim) com a palavra "robótica" (realização de uma ação sem intervenção humana) (Barros, 2010; Silva, 2008). Deste modo pode-se deduzir que a domótica não é mais do que o conjunto de tecnologias e sistemas que permitem automatizar as diversas tarefas a realizar no interior de uma habitação. Esta automatização permite reduzir, tanto quanto possível, a intervenção humana. Por outro lado, a domótica também permite o apoio a pessoas com deficiências ou pessoas idosas, aumentando assim a sua qualidade de vida e autonomia, visto que o sistema pode assumir o papel de "mordomo" para este tipo de pessoas.

Em suma, a domótica tem como principal objetivo melhorar o conforto, a segurança, a comunicação e a gestão de energia de uma habitação, como se pode observar pelo esquema apresentado na Figura 1-1.

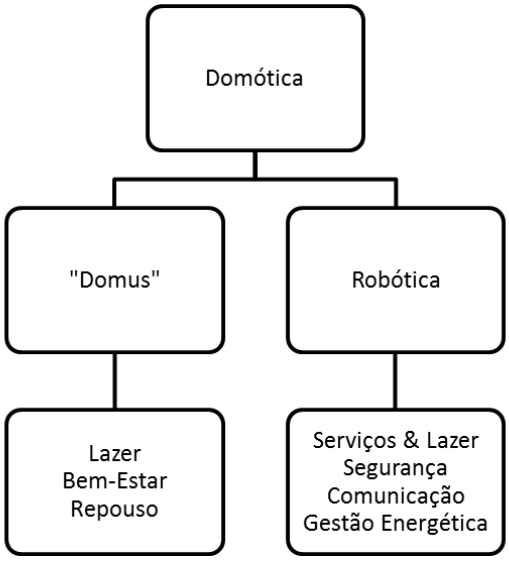


Figura 1-1: Definição de domótica.

Devido à domótica ser uma área bastante abrangente esta pode ser dividida de diversas formas (Barros, 2010; Gouveia, 2009; Silva, 2008). Nesta dissertação é seguida a divisão da domótica em quatro grandes áreas (ver Figura 1-2):

- ***Serviços & Lazer*** – Área responsável pela automatização dos diversos sistemas da residência como climatização, iluminação, entre outros. Permite aumentar os níveis de conforto da residência e libertar o utilizador de determinadas rotinas domésticas;
- ***Segurança –*** Área que permite dotar a residência de sistemas para a deteção de vários cenários indesejados como intrusões, inundações, incêndios e fugas de gás;

- ***Comunicação*** – Área que abrange as comunicações internas (por exemplo entre o utilizador e o sistema de automatização residencial) e comunicações externas, ou seja, comunicar com a habitação via web ou dispositivos móveis;
- ***Gestão Energética*** – Área responsável pela racionalização dos consumos energéticos por parte dos diversos sistemas. Considerada uma área de vital importância no quotidiano atual face aos novos pressupostos energéticos e ambientais a nível mundial.

Foi adotada esta divisão, visto que os parâmetros de serviços e lazer (conforto) e segurança são dois parâmetros de extrema importância no que diz respeito a uma habitação. Tratando-se de um sistema que integra vários subsistemas, é necessário o desenvolvimento de protocolos de comunicação para que os subsistemas comuniquem entre si e para que o sistema possa ser acedido de fontes externas. Por fim, inclui-se a gestão energética devido às conjunturas da sociedade atual para a eficiência energética de uma habitação.

A divisão anterior permitirá classificar e comparar as diversas soluções existentes no mercado atualmente e focar a área de intervenção do sistema físico desenvolvido e a aplicação a integrar neste sistema.

Figura 1-2: Divisão da domótica em quatro grandes áreas.

Relativamente ao funcionamento de um sistema de domótica, genericamente este é composto por uma rede de comunicação dispositivos de recolha de informação e um órgão de supervisão (Gouveia, 2009). A informação recolhida dos diversos pontos da habitação possibilita uma análise do estado do ambiente residencial, que por sua vez permite ao órgão de supervisão atuar sobre o mesmo espaço de forma a exercer o controlo necessário.

No que diz respeito às arquiteturas dos sistemas de automação residencial, estas podem ser divididas em dois grandes tipos (Gouveia, 2009):

- **Centralizadas –** Constituídas por uma unidade central que comanda todo o sistema, desde a recolha de informação até à supervisão do sistema;
- **Distribuídas –** Trata-se de uma arquitetura, onde existem vários dispositivos com capacidade de processamento de informação, que podem comunicar entre si.

Aliado ao desenvolvimento dos sistemas de domótica, existe também um forte desenvolvimento ao nível dos dispositivos móveis inteligentes que, quando associados a estes sistemas, melhoram consideravelmente a interação com a domótica (Anacleto, 2012). Desta forma os sistemas vêm a sua flexibilidade e controlo melhorados, uma vez que com um simples toque no ecrã, poder-se-á desencadear um número infindável de ações.

1.2 Enquadramento

Com o intuito de se gerar um sistema com as características enunciadas na secção anterior, foi proposto a criação da aplicação "HomeTouch" que será integrada no sistema físico "HomeDom", desenvolvido pela empresa Tecsisel (empresa promotora desta dissertação).

Relativamente ao funcionamento da aplicação, esta terá que efetuar o controlo dos parâmetros do sistema físico, que se encontram associados aos tipos de variáveis suportadas por este que são a iluminação, estores, temperatura e som.

Após este controlo ser processado pela aplicação, a informação sobre o mesmo será posteriormente enviada para uma das consolas de comando presentes na habitação, mais concretamente uma por divisão. Aquando da receção da informação, a consola de comando irá processar a informação e atuar de modo a satisfazer os comandos recebidos.

A consola de comando também fornece informação à aplicação quando solicitada pela mesma, permitindo ao utilizador ter conhecimento do estado de cada divisão em tempo real, procedendo às alterações se necessário.

Em suma, o funcionamento global do sistema pode ser representado pelo exemplo apresentado pela Figura 1-3, onde existe uma consola, que representa as diversas consolas presentes na residência, que comunica com a aplicação. Esta comunicação é sempre desencadeada pela aplicação tendo em conta a ação desencadeada pelo utilizador. Essas ações podem passar pela consulta e alteração do valor de um determinado comando, como por exemplo, a iluminação de uma determinada zona da divisão.

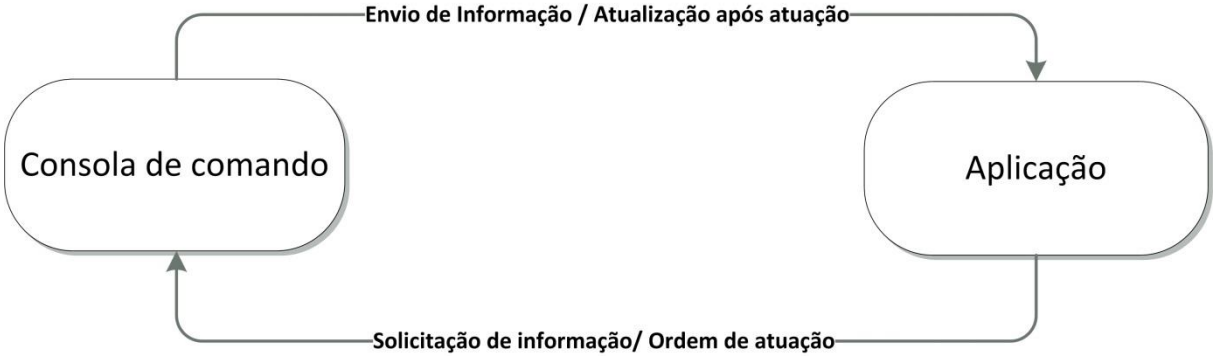


Figura 1-3: Ciclo de funcionamento do Sistema "HomeDom".

1.3 Objetivos

Tendo em conta o ciclo de funcionamento apresentado na Figura 1-3 e que a arquitetura do sistema siga a configuração distribuída, esta dissertação tem os seguintes objetivos:

- Desenvolver uma aplicação para dispositivos móveis que permita controlar e gerir a arquitetura física existente do sistema de domótica.

De modo a satisfazer o objetivo proposto foi criada a aplicação "HomeTouch" capaz de controlar os diversos subsistemas de domótica. Relativamente à plataforma móvel utilizada, a entidade promotora desta dissertação optou pela plataforma Android™. Quanto ao funcionamento da aplicação, esta permite efetuar o controlo dos diversos parâmetros presentes na arquitetura física do sistema, criando um novo ponto de acesso do sistema independente da arquitetura física. Adicionalmente, a aplicação fornece ao utilizador o valor atual das diversas variáveis associadas a cada divisão permitindo ao utilizador ter conhecimento do seu estado sem necessidade de se deslocar à consola de comando.

1.4 Estrutura e Organização da Dissertação

Como se trata de um documento onde são abordados diversos conteúdos relacionados com o tema desta dissertação, é necessário agrupar os mesmos em capítulos, tendo em conta o seu tipo e finalidade.

Deste modo esta dissertação irá ser organizada através de 6 capítulos:

- Capítulo 1 – Introdução
  - No capítulo um é efetuada uma pequena apresentação do conceito de domótica, seguida do enquadramento e objetivos a atingir nesta tese. Por fim é exposta a estrutura e organização deste documento.

- Capitulo 2 – Sistemas de Gestão e Controlo Habitacionais
  - Inicialmente neste capítulo, aborda-se mais detalhadamente o conceito de domótica, mais concretamente a sua origem e evolução. De seguida são apresentados os protocolos existentes para se efetuar a comunicação nos diversos sistemas da automação residencial. Seguidamente apresentam-se os sistemas de automação residencial existentes e comercializados nos mercados internacionais e nacional, neste caso Portugal, de modo que se possa analisar o que já se encontra implementado. Por fim, inicialmente é efetuada uma comparação entre os sistemas comercializados com ao auxílio a parâmetros que serão expostos neste capítulo. Após essa comparação é efetuado um paralelo entre os sistemas comercializados e o sistema da entidade promotora, de modo a concluir-se quais as vantagens que o segundo possui face aos restantes.

- Capítulo 3 – Tecnologias de Suporte
  - Este capítulo aborda as tecnologias que serão utilizadas no desenvolvimento da aplicação. Nomeadamente a plataforma móvel escolhida (Android), protocolo de comunicação (ModBus) e biblioteca de armazenamento de dados (SQLite).

- Capítulo 4 – Arquitetura do Sistema
  - Como o seu nome indica, este capítulo apresenta a arquitetura geral do sistema, isto é, apresenta a componente física do sistema e posteriormente a componente lógica (aplicação desenvolvida).

- Capítulo 5 – Aplicação “HomeTouch“
  - Relativamente a este capítulo, é apresentada a aplicação desenvolvida, isto é, apresentam-se as diferentes interfaces gráficas onde estão inseridas as diversas funcionalidades referidas no Capítulo 4.

- Capítulo 6 – Conclusões Gerais e Trabalhos Futuros
  - Por fim neste capítulo são apresentadas as conclusões gerais referentes a esta dissertação. Adicionalmente são expostos alguns trabalhos a desenvolverem-se no futuro no âmbito desta dissertação.

2. Sistemas de Gestão e Controlo Habitacionais

Devido à expansão do mercado da automação residencial dos últimos anos, atualmente encontram-se diversas ofertas de sistemas de domótica, diferenciados entre si em detalhes ou opções de funcionamento.

Neste capítulo são apresentados alguns dos sistema de gestão e controlo habitacionais já comercializados na área da automação residencial. Esta apresentação permitirá averiguar quais são os métodos de instalação, modos de operação entre outras características. Para além disso, a informação apresentada permitirá compará-los com o sistema desenvolvido de modo a se averiguar se o sistema de domótica da empresa é competitivo comercialmente face aos restantes.

A secção 2.1 aborda detalhadamente a origem e evolução do termo domótica. Com esta abordagem pretende-se expor as inovações que permitiram que este conceito tenha ganho grande relevo na sociedade. Na secção 2.2 são apresentadas as ofertas existentes para o controlo habitacional no panorama internacional bem como nacional. Estas ofertas refletem o quão uma residência se pode tornar autónoma, não se excluindo a intervenção humana necessária para a configuração e atribuição de um maior grau de autonomia. Após a apresentação das ofertas existentes é efetuada uma comparação sistematizada entre todos os sistemas, tendo em conta as áreas da domótica como a segurança, comunicação, entre outras. Por fim, a última secção deste capítulo apresenta os pontos de divergência entre sistema desenvolvido e os demais referenciados na secção 2.2, em termos de funcionamento e implementação. Essa apresentação irá permitir concluir se o sistema desenvolvido possui vantagens para competir com os restantes no mercado da automação residencial.

2.1 Domótica

Como todas as restantes inovações tecnológicas, que não surgiram num ápice, também a domótica passou pelo mesmo processo evolutivo. Processo esse que é mais antigo do que frequentemente se julga, visto que os primeiros passos nesta área foram dados ainda no século XIX, muito antes do nascimento do termo em si.

Os primeiros passos foram dados pela mão de um dos grandes escritores de ficção científica, Júlio Verne, quando no século XIX começou a introduzir o conceito de automação (Larousse, 1999), dando de certa forma, início às inovações tecnológicas responsáveis para a implementação da domótica na atualidade (Tseng, Cheng, Chang, & Wen-Shyang Hwang, 2012). A automação industrial, responsável pela domótica na atualidade, surgiu ainda no final do século XIX quando o professor Warren Johnson inventou o termostato elétrico (Controls, 2009). Este novo equipamento veio permitir o controlo da temperatura em espaços fechados de forma automática (Controls, 2009). Ainda no final desse século, foram concebidos os primeiros atuadores e sensores de controlo pneumático de temperatura pela empresa Johnson Electric Service Company (fundada pelo professor Warren Johnson), dando origem ao primeiro sistema de controlo de temperatura por zonas ou divisões. Este sistema de controlo, para além de reduzir a intervenção humana, permitiu otimizar os consumos energéticos, sendo implementado em locais de grande importância tais como o Capitólio dos EUA, o palácio imperial do Japão e o palácio real de Espanha.

Todo o processo de evolução da automação foi acompanhado por um forte progresso ao nível da construção civil no que diz respeito aos materiais utilizados (Silva, 2008). Na Tabela 2-1 encontram-se as evoluções presenciadas neste ramo.

Tabela 2-1: Evolução dos edifícios ao nível da construção, adaptado de (Neto & Lopes, 2004).

	Fator Evolutivo
Método Tradicional (Até 1900)	Materiais rudimentares (pedra e madeira). Construções rígidas com serviços básicos
Evolução na construção (1900-1945)	Novos tipos de materiais (betão armado). Evolução nos sistemas elétricos.
Evolução Tecnológica (1945-atualidade)	Espaços interiores mais dinamicos e flexíveis. Implementação de invoção técnológicas

Estas evoluções permitiram que os sistemas automatizados, desenvolvidos para a automação industrial, pudessem ser implementados em edifícios públicos, destinados ao setor dos serviços (Barros, 2010). Contudo esta implementação foi lenta ocorrendo com o passar das décadas. A Tabela 2-2 mostra o tipo de sistemas que foram introduzidos em cada década.

Tabela 2-2: Evolução dos edifícios inteligentes, adaptado de (Neto & Lopes, 2004).

	Evolução Técnológica	Implementação Prática
Década 60	Sistemas de controlo centralizado	Sistemas de Climatização
Década 70	Comercialização de microprocessadores	Sistemas de gestão e controlo mais sofisticados e maior quantidade
Década 80	Processamento e transmissão de informação bem como requisitos de segurança e conforto	Sistemas de automação utilizando tecnologias de informação e comunicação

Com a introdução destas tecnologias nos edifícios surgiu o conceito de edifício inteligente, contudo atualmente ainda não apresenta uma definição consensual na comunidade científica (Silva, 2008). Com o aparecimento dos edifícios inteligentes, faltava conquistar o último tipo de edifício onde o ser humana passa o seu tempo diário restante que é a sua habitação residencial.

Agregando todos os conteúdos acima mencionados, obtém-se a Figura 2-1, onde está representada como se tonar uma simples habitação numa habitação inteligente.

Figura 2-1: Diagrama duma habitação inteligente.

2.1.1 Comunicação em sistemas de domótica

Com a massificação dos sistemas de automação residencial na sociedade, levou-se também ao desenvolvimento duma área fundamental que é a comunicação entre controladores e dispositivos de campo. Nessa evolução, surgiram três tipos de comunicação para sistemas de domótica com a seguinte designação:

- **Powerline (Rede elétrica) –** Tipo de comunicação que reutiliza a instalação da habitação para o funcionamento do sistema de automação residencial. De referir que sistemas baseados neste tipo de comunicação trazem impactos mínimos aquando da sua instalação, no que diz respeito a alterações físicas na habitação;

- **Cable (Cabo) –** Como o seu nome indica é necessário recorrer à instalação de cabos na habitação para o funcionamento dos sistemas de domótica. A instalação de sistemas com base neste tipo de comunicação em habitações já existentes, pode provocar uma instalação mais dispendiosa;

- **Wireless (Sem fios) –** Tipo de comunicação que recorre ao meio ambiente para transferir informação entre pontos. Assim como o powerline, também este modo de comunicação traz um impacto mínimo aquando da instalação dum sistema de domótica que o utilize.

2.1.2 Protocolos de comunicação em sistemas de domótica

Com a evolução das redes de comunicação nas suas diversas vertentes e com a alta taxa de aceitação do conceito de domótica por parte da sociedade, surgiram com o passar das décadas diversos protocolos que podem ser implementados nos sistemas de automação residencial. É de referir que já existem protocolos que conseguem combinar dois tipos de comunicação, com o intuito de se alargar o número de habitações onde se pode implementar o sistema de domótica.

Tendo em conta os tipos de comunicação já apresentados, de seguida apresentam-se os protocolos mais relevantes em cada um dos tipos:

- **X10 –** Protocolo mais antigo criado para a domótica, no entanto ainda em utilização. Trata-se dum protocolo que opera sobre a instalação elétrica existente, utilizando uma frequência específica para transmitir a informação (Barry Grussling, 2014). Na Figura 2-2 encontra-se exemplificado este protocolo;

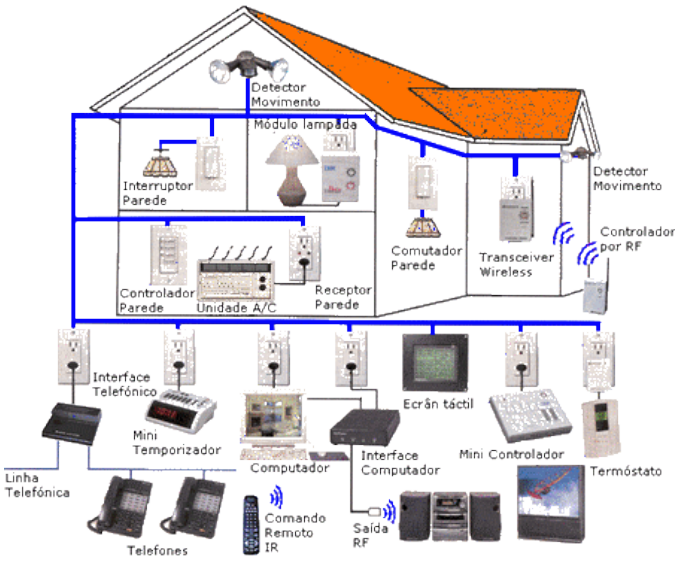


Figura 2-2: Exemplo de instalação do protocolo X10, adaptado de (Eletronica PT, 2014).

- **INSTEON –** Protocolo que foi disponibilizado no ano de 2005 após anos de desenvolvimento. Este protocolo cria uma rede residencial automatizada que combina a comunicação entre os diversos equipamentos por via da instalação elétrica e/ou sem-fios (radiofrequência) como se pode visualizar na Figura 2-3 (INSTEON, 2013).

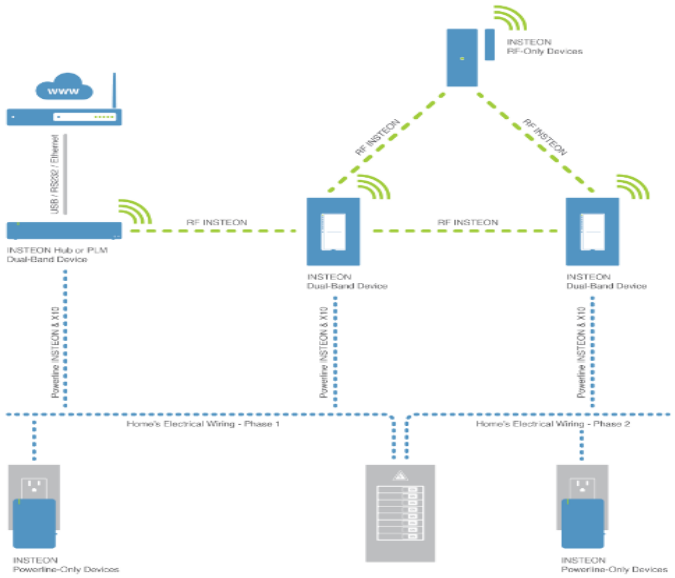


Figura 2-3: Funcionamento da tecnologia INSTEON, adaptado de (INSTEON, 2013).

- **LonWorks –** Protocolo criado na década de 90 pela Echelon Corporation. É também um protocolo que combina dois modos de funcionamento dedicados que é por via da rede elétrica ou por via par entrançado (cabo). Também possui a capacidade de ser adaptado a uma rede TCP/IP já existente podendo estabelecer ligação à internet (Echelon Corporation, 2009).

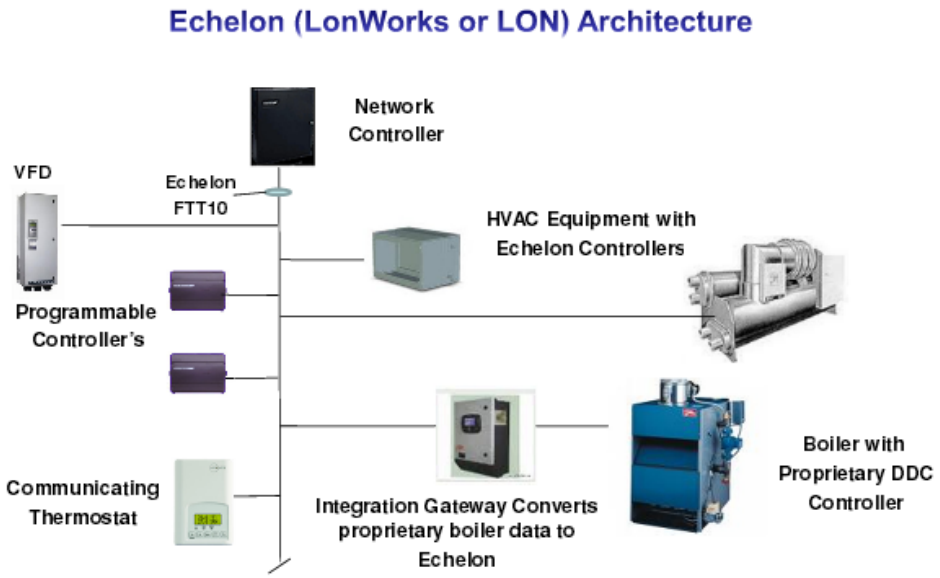


Figura 2-4: Esquema de funcionamento da tecnologia LonWorks.

- **Z-Wave** – O protocolo Z-Wave foi originalmente desenvolvido pela empresa Zensys, mas que devido à sua grande aceitação na área da automação, agora é tutelado pela Z-Wave Alliance (Z-Wave Alliance, 2014). No que diz respeito ao seu funcionamento assenta na comunicação sem fios, não necessitando de nenhum tipo de cablagem para que os equipamentos comuniquem entre si. Na Figura 2-5 está esquematizado um exemplo de comunicação recorrendo ao protocolo Z-Wave.

Figura 2-5: Comunicação através do protocolo Z-Wave (Z-Wave Alliance, 2014).

- **ZigBee** – O protocolo ZigBee, desenvolvido pela ZigBee Alliance (ZigBee Alliance, 2014), é caracterizado por ser um protocolo de muito baixo consumo energético, com comunicação sem fios. O exemplo de uma possível implementação deste protocolo encontra-se representado na Figura 2-6.

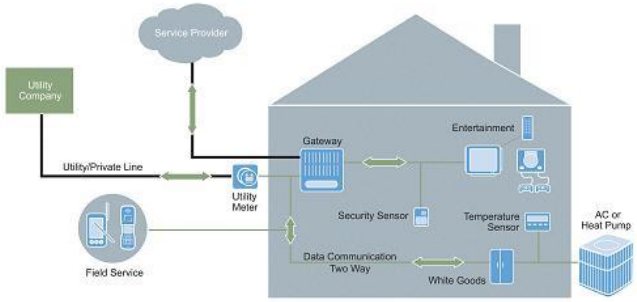


Figura 2-6: ZiggBee Home Automation (Kubismus, 2014).

2.2 Sistemas de domótica comerciais

Sendo os Estados Unidos (EUA) um dos pioneiros deste tipo de sistema e possuindo um mercado com diversas empresas, permitindo o desenvolvimento de múltiplas soluções, optou-se por analisar algumas dessas ofertas de domótica já comercializadas neste país no ano de 2014.

A análise dos sistemas será feita tendo em conta a definição apresentada de domótica e a sua divisão nas quatro grandes áreas: Serviços & Lazer, Segurança, Gestão de Energia e Comunicação, de acordo com o que foi apresentado no capítulo 1.

2.2.1 HomeSeer HS3

O sistema HomeSeer 3 (HomeSeer, 2014a), foi desenvolvido pela empresa HomeSeer (Homeseer, 2014), outrora *Keware Technologies*, fundada no final da década de 90 e considerada uma empresa de topo na área da automação residencial. Este sistema (ver Figura 2-7) potencia a integração de diversos sistemas capazes de controlar parâmetros de uma habitação como a iluminação, temperatura entre outras.

No campo das funcionalidades, pode-se referir que este sistema abrange os quatro campos mencionados anteriormente que regem a domótica. Isso é possível, uma vez que o sistema permite o controlo de todos os sistemas residências, permitindo aos utilizadores libertar-se de certas rotinas, aplicando esse tempo em outras atividades que até aqui não era possível.

Figura 2-7: Software Controlo Habitacional HomeSeer HS3 (HomeSeer, 2014c).

Ao nível da segurança, é um sistema dotado de diversas funcionalidades, além do usual subsistema de anti-intrusão, visto que possui sistemas de deteção de inundações entre outros, dando aos utilizadores uma maior segurança estando estes no interior ou ausentes da residência.

Por outro lado, o HomeSeer também possui mecanismos que permite efetuar a gestão de energia, através da monitorização de determinadas variáveis energéticas como potência consumida, entre outras. Através destas variáveis são elaborados os padrões de consumo energéticos, que permitem reajustar a utilização dos equipamentos domésticos. Tal é possível uma vez que atualmente os equipamentos domésticos são desenvolvidos de modo a receberem comandos de sistemas de automação residencial para iniciarem e interromperem o seu funcionamento tendo em conta o padrão energético desejado.

Para atingir as diversas ações já referidas é necessário dotar o sistema de inteligência e autonomia, visto que o sistema possuiu uma autonomia inicial limitada. Neste caso, essa autonomia ou inteligência limitada provem de pré configurações básicas como execução

certas ações como por exemplo a ativação temporizada de uma lâmpada. Adicionalmente é permitido ao utilizador que configure as designadas macros ou tarefas dotando a habitação com a autonomia desejada pelo mesmo, tornando a casa tão mais inteligente e autónoma conforme a complexidade das macros elaboradas pelo utilizador.

Por último, quanto à comunicação e a nível externo, este sistema, além do ponto inicial de comando que é o computador, permite o acesso remoto por via dispositivos móveis através de aplicação HSTouch (HomeSeer, 2014d). Este tipo de acesso torna o sistema muito versátil nesta categoria, que é um dos pontos além das funcionalidades já apresentadas que permite obter a distinção entre os demais sistemas. Quanto ao manuseamento da aplicação para os dispositivos móveis e ao software para computadores fornecidas para o HS3, é importante mencionar que ambos caracterizam-se por uma fácil usabilidade não havendo a necessidade de conhecimentos adicionais para operar com ambas as possibilidades. No entanto é de salientar que quanto maior a complexidade do sistema físico, mais tempo é necessário para dominar todo o software de supervisão do HS3.

Em relação à compatibilidade do HS3, categoria que se pode encaixar na área da comunicação, este sistema possui uma vasta compatibilidade com as plataformas Microsoft e Apple. Uma das principais características do HS3 passa pela exigência do Windows Media Center para a utilização de algumas das suas funcionalidades. Outras das aplicações suportadas pelo HS3 ao nível de multimédia é o Windows Media Player e iTunes permitindo um vasto campo de ação ao nível de utilizadores. É de realçar que o software HS3 pode ser instalado nas recentes plataformas Windows (ver Figura 2-8) ao contrário da Mac OSX para a qual este sistema não possui essa compatibilidade à data desta análise. Relativamente aos web browsers, o HS3 já possui uma maior diversidade, uma vez que permite o acesso através dos habituais browsers como o Firefox, Safari e Google Chrome.

Figura 2-8: Ecrãs do software HomeSeer HS3 (HomeSeer, 2014c).

No que diz respeito ao hardware, este sistema possui uma loja online, característica única entre os vários sistemas que serão apresentados. Nesta loja podem-se consultar todos os

periféricos já testados no HS3, desde sensores, atuadores, camaras de segurança, entre outros para o HomeSeer's HomeTroller1[1] (HomeSeer, 2014b) que se pode visualizar na Figura 2-9. É de salientar que este sistema possui compatibilidade com sistemas HVAC, responsáveis por arrefecimento e aquecimentos de ambientes fechados.

Figura 2-9: Controlador HS HomeTrollser Series 3 (HomeSeer, 2014b).

No que diz respeito à comunicação interna, ou seja, a comunicação feita entre ponto de comando e seus periféricos, é assegurada pelos seguintes protocolos suportados pelo HS3:

- Insteosn;
- UPB;
- X10;
- Z-Wave;
- Lutron HomeWorks.

É de referir que nos protocolos mencionados, estão inseridas as três vertentes de transmissão de dados que são via cablado, wireless ou sinal elétrico. O utilizador possui assim uma ampla lista de periféricos, tendo em conta as suas necessidades de implementação e o tipo de tenologia a usar, sendo o exemplo dessa lista, a tabela que se encontra no Anexo A.

2.2.2 Control4

O sistema Control4 (Control4, 2013), possui o mesmo nome que a empresa que o projetou, a Control4, estabelecida no início deste milénio. Tem como principal particularidade a par da em empresa HomeSeer, ser uma empresa de eleição no que diz respeito à domótica residencial.

[1] Controlador desenvolvido pela HomeSeer para servir de ponte entre o mundo físico e o ciberespaço.

Figura 2-10: Sistema Control4 (Control4, 2014c).

Em relação ao sistema Control4 (ver Figura 2-10), tem como principais características a sua robustez e a facilidade de uso, sendo que é uma solução que consegue integrar as quatro categorias mencionados inicialmente nesta análise. Relativamente às suas funcionalidades, este sistema consegue efetuar a gestão habitacional da mesma forma que o anterior, mantendo assim o padrão de conforto para os seus utilizadores, permitindo que estes se dediquem a outras iniciativas (ver Figura 2-11). No entanto, este não possui a opção de reconhecimento de voz para desencadeamento de ações no sistema, havendo a necessidade de os utilizadores se desloquem a um ponto de acesso do sistema para efetuar alguma alteração.

Referente à sua autonomia e inteligência, este sistema possui um nível básico, podendo ver esses dois parâmetros evoluir consoante a complexidade que o seu utilizador lhe proporcione ao nível dos mecanismos de gestão e controlo. A complexidade deste sistema também está associada ao tipo de tarefas a executar e o número de dispositivos de controlo e atuação que o utilizador requeira ao instalador.

Na categoria da segurança, esta solução de domótica também se encontra preparada para lidar com os diversos cenários a este nível, possibilitando aos utilizadores que possam estar mais tranquilos quando ausentes da residência, por exemplo aquando de férias.

Figura 2-11: Exemplo do funcionamento do Control4.

No que diz respeito à comunicação, mais concretamente a nível externo, o Control4 oferece a comunicação via dispositivos móveis, além dos tradicionais browsers, através do aplicativo MyHome (Control4, 2014a), tanto para as plataformas Android e IOS. Contudo, este tipo de acesso é limitado às proximidades da casa, sendo que este pode passar a acesso remoto na eventualidade da aquisição do protocolo 4Sight (Control4, 2014a). Este protocolo torna o acesso até agora local para um tipo de acesso remoto, podendo ser o sistema acedido de qualquer parte. Adicionalmente, o utilizador passa a ter a informação do estado do sistema em qualquer um dos seus dispositivos, uma comunicação mais segura, tornando o sistema mais versátil e cómodo para o utilizador.

No capítulo da comunicação interna, esta solução possui um ponto fraco, uma vez que se trata de um sistema muito proprietário a nível de hardware contrariamente ao sistema apresentado pela HomeSeer. O Control4 possui uma lista mais restrita de aparelhos compatíveis com este sistema, passando muito por equipamentos desenvolvidos pela própria empresa. No entanto também permite a utilização de equipamentos não abrangidos pela marca só que em quantidades mais reduzidas. Quanto aos protocolos suportados, consoante o tipo de controlador utilizado (ver Figura 2-12) este sistema possui os seguintes protocolos de comunicação:

- Ethernet (TCP/IP);
- Z-Wave;
- ZigBee.

Com o tipo de protocolos apresentados, pode-se concluir que a instalação do Control4 pode ser feita sem recurso modificações estruturais nas habitações, um fator a ter em conta na instalação desta solução numa habitação já construída.

Figura 2-12: Controlador Control4 HC-800 Controller (Control4, 2014b).

Quanto à sua implementação, o Control4 só é instalado por via de vendedores certificados pela empresa Control4 (Control4, 2015). Este procedimento limita em certa medida este sistema tendo em conta que qualquer modificação seja ela a remoção, alteração de equipamentos ou uma possível expansão do mesmo necessite que um técnico efetue essas alterações. Por outro lado o utilizador apenas necessita de escolher o tipo de gestão e controlo que deseja, sem ter que se preocupar de como o vai ter que instalar, sendo que esse tipo de opereação fica ao cargo do vendedor que tem a responsabilidade de instalar o sistema.

Por fim no campo da gestão energética, pelo que se averiguou, não é fornecida qualquer informação sobre a aplicação deste campo no Control4, sendo que por isso presume-se que não possua essa funcionalidade. Numa eventualidade de uma possível contribuição a este nível, esta tem de ser efetuada pelo utilizador através de configuração das ditas macros de modo a ter-se um perfil energético satisfatório.

2.2.3 Crestron Series 3

Os sistemas *Series 3* (Technical Publications Department, 2013), foi desenvolvido pela empresa Creston fundada nos meados do século passado, sendo também numa empresa líder na área da automação, para residências, escolas ou edifícios de negócios.

Figura 2-13: Sistema Série 3 da Creston (Crestron, 2014b).

Do conjunto de sistemas que constituem a *Series 3* (ver Figura 2-13), vai ser analisado o sistema PRO3 que é a solução mais equipada face às restantes, sendo rotulado por ser um sistema muito dinâmico e elegante. Sobre as suas funcionalidades, pode-se referir que este sistema possui a capacidade de gerir os diversos parâmetros de controlo, contribuindo nesse caso para o lazer dos seus utilizadores que mais uma vez se podem libertar de rotinas, reutilizando esse tempo para outro tipo de atividades.

Uma das funcionalidades únicas desta solução está relacionada com a execução de tarefas personalizadas, que serve para automatizar o sistema, sendo que o PRO3 permite uma execução simultânea de múltiplas tarefas (ver Figura 2-14). Esta funcionalidade é passível de ser executada visto que o PRO3 assenta numa arquitetura distribuída, ou modular, que permite uma maior rapidez de comunicação entre o controlador central e os diversos periféricos, visto que a comunicação é feita através dos diferentes módulos e não na globalidade. Em caso de atualização do sistema, este não necessita de ser desligado completamente mas sim o módulo em questão, uma vez que este tipo de arquitetura permite ter vários programas em vez de um único programa que coordena tudo, sendo que alterações num programa específico não afetam os restantes.

Figura 2-14: Funcionamento da arquitetura distribuída Crestron (Crestron, 2014b).

Relativamente à segurança, de mencionar que os tradicionais cenários estão salvaguardados, sendo que para o caso do subsistema de anti-intrusão, este já não oferece o serviço que monitoriza a residência aquando da ausência de todos os seus residentes.

Ao nível da comunicação externa, este apresenta a mesma facilidade de acesso tanto localmente por via dos usuais sistemas computacionais, isto porque foi dotado de compatibilidade com os grandes sistemas operativos, o Windows e o Mac OSX. A nível remoto, o acesso ao sistema pode ser efetuado através das aplicações das plataformas Android e IOS como se encontra representado na Figura 2-15.

Figura 2-15: Aplicações móveis do sistema Crestron *Series 3* (Crestron, 2014a).

Por ser um sistema que unifica diversos sub-sistemas sobre uma única tutela, a Crestron projetou a *Series 3*, onde se insere o PRO3, de modo a que este possua uma alta taxa de compatibilidade com equipamentos de outras marcas, tornando o sistema mais versátil. Esta característica não é em nada afetada pelo facto de a instalação deste sistema ser feita por via a instaladores certificados pela empresa em questão.

Relativamente à comunicação interna, o PRO3 é dotado de protocolos de comunicação proprietários, isto é, protocolos fechados desenvolvidos pela Crestron e de acesso privado. Mas nesta vertente também foi permitida a compatibilidade deste sistema com os diversos protocolos mais conhecidos na área da automação residencial tornando-o bastante versátil. Assim sendo, os protocolos permitidos no PRO3 são:

- Insteosn;
- UPB;
- X10;
- Z-Wave;
- ZigBee;
- KNX;

Uma particularidade deste sistema, PRO 3, para este tipo de comunicação é a presença do controlo por sub-rede, representada na Figura 2-16, que não é mais que uma rede ethernet gigabit dedicada para dispositivos proprietários Creston. Esta sub-rede permite que todo o sistema possua um endereço de IP único, não afetando a comunicação entre todos os dispositivos da sub-rede com a restante rede da habitação. Para aplicações sensíveis que necessitam de segurança absoluta, todo o controle sub-rede pode ser completamente isolado da rede do cliente usando o modo de isolamento.

Figura 2-16: Esquema de funcionamento da subnet (Crestron, 2014b).

No que diz respeito à gestão energética, é de realçar que a Crestron projetou uma plataforma capaz de fazer a monitorização e gestão dos consumos de grandes complexos através do software Crestron Fusion RV. Este software encontra-se inserido no PRO3, permitindo que os consumos podem ser quantificados e futuramente diminuídos, aquando da análise do perfil energético. Esta análise possibilita alterações no mesmo perfil, por via do agendamento de determinadas tarefas como por exemplo utilizar a máquina de lavar loiça em horários em que o preço da energia é mais reduzido.

No que concerne ao mercado nacional, este encontra-se em evolução, sendo o caso deste progresso o seguinte sistema cuja conceção e desenvolvimento foi efetuado em Portugal.

2.2.4 Domus

Este sistema foi criado pela subsidiária JG Domótica (JG Domótica, 2014a), que pertence ao grupo JG Componentes que opera em Portugal, na área da computação e automação desde início da década de 80. Sistema integralmente concebido e desenvolvido em Portugal, através da parceria da JG Domótica com várias marcas de componentes, entre elas a Matsushita e Mitsubishi entre outras.

Figura 2-17: Sistema Domus (JG Domótica, 2014c).

Pode-se referir que o Domus possui diversos módulos como mostra a Figura 2-17, permitindo a evolução do sistema a partir do módulo base. Estes oferecem aos seus utilizadores uma sensação de conforto, libertando-os das obrigações mais básicas como o acender e o apagar da iluminação por cada divisão que uma pessoa passe. Contudo, na sua forma mais básica, este sistema não apresenta qualquer nível de autonomia ou inteligência, o que significa que o sistema não executa comandos pré-definidos.

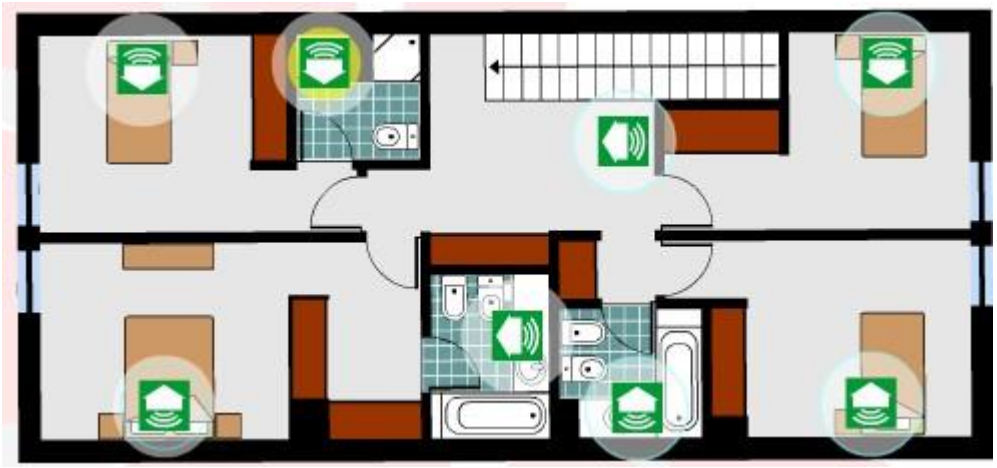


Figura 2-18 – Exemplo dos sensores para medição de temperatura (JG Domótica, 2014b).

Analisando a segurança oferecida por este sistema, pode-se referir que o Domus oferece um nível considerável neste aspeto, uma vez que utilizando os módulos opcionais disponibilizados pela JG Domótica obtém-se o controlo contra os diversos cenários possíveis no interior da habitação como incêndios ou inundações. De salientar que no caso do subsistema de anti-intrusão, o Domus vem equipado com o módulo de telecomunicações, que executa chamadas com conteúdo pré gravado para as autoridades ou outro número a designar pelo utilizador, como se pode visualizar ma Figura 2-19.

Figura 2-19: Exemplo do “modo total” do módulo de segurança do sistema Domus (JG Domótica, 2014d).

Em relação à comunicação, é de salientar que a nível externo, este sistema não possui à data desta análise, acesso remoto por via dispositivos móveis, não podendo ser utilizados os tradicionais smartphones ou tablets. No entanto, é disponibilizado um módulo GSM[2], que permite ao utilizador do sistema enviar mensagens pré-definidas via telemóvel, para execução de determinadas ações no sistema. Adicionalmente para este tipo de comunicação, são fornecidos 2 comandos onde são inseridas as opções de controlo do sistema, de modo a compensar o defeito anterior, pois estes comunicam com os diversos sensores através de infravermelhos. Por fim, referir que este sistema possui o acesso via software fornecido pelos instaladores, que além de permitir o controlo e configuração, também disponibiliza a opção de criação e configuração de macros, o que faz com que o sistema adquira um determinado nível de autonomia consoante as configurações introduzidas pelo utilizador.

Relativamente ao nível de comunicação entres equipamentos, ou seja, interna, quando comparado com os sistemas de automação residencial estrangeiros, verifica-se que o sistema Domus é um sistema mais fechado, ou seja, a comunicação é feita com protocolos proprietários da empresa JG Domótica.

No que diz respeito à gestão de energia, este sistema não foi projetado para lidar com este tipo de controlo como acontece com os anteriores, sendo que inadvertidamente possa contribuir para esta mesma gestão devido a algumas opções que este sistema possui. Um exemplo do que foi mencionado passa pelo modo automático no controlo da iluminação, onde em caso algum, nenhum ponto de luz fica ligado sem a presença física de uma pessoa nessa mesma divisão devido aos sensores de movimento. Na Figura 2-20 encontra-se exposto um exemplo deste modo de funcionamento.

Figura 2-20: Modo automático do modo de iluminação do sistema Domus. (JG Domótica, 2014b).

[2] Sistema Global para Comunicações Móveis (Global System for Mobile Communications), também conhecido como 2G, que permite a comunicação móvel em qualquer parte do globo.

Por fim, é de realçar que o Domus apresenta uma característica não evidenciada nos restantes que é a presença de uma *Uninterruptible Power Supply* (UPS), cuja função é manter o sistema operacional quando exista uma falha de fornecimento de energia elétrica à habitação.

Para terminar esta análise, na Tabela 2-1 encontra-se a classificação resumidamente dos sistemas escolhidos segundo as vertentes da domótica, mencionadas no início desta secção.

Referente à classificação destes sistemas, esta encontra-se dividida em 5 categorias, onde a de valor mais inferior (um círculo preta) corresponde a uma total ausência das características abordadas, enquanto a contrária (cinco círculos pretas) indica a presença de todos os atributos desejados.

- **Classificação:**

  - ●●●●● – O sistema possui todas as características desejadas;
  - ●●●●○ – O sistema tem pequenos defeitos nas suas características;
  - ●●●○○ – O sistema possui algumas falhas nas suas características;
  - ●●○○○ – O sistema possui muitas falhas nas características desejadas;
  - ●○○○○ – O sistema não possui nenhuma das características desejadas;

Tabela 2-3: Tabela comparativa dos diferentes sistemas de automação residencial.

Categorias \ Sistemas		HomeSeer - HS3 -	Control4 - Control4 -	Crestron - PRO3 -	JG Domótica - Domus -
Serviços & Lazer		●●●●●	●●●●●	●●●●●	●●●●●
Segurança		●●●●●	●●●●○	●●●●○	●●●●●
Comunicação	Externa	●●●●●	●●●●●	●●●●●	●●●○○
	Interna	●●●●●	●●●●○	●●●●○	Sem informação
	Compatibilidade	●●●●●	●●○○○	●●●●●	●●○○○
Gestão Energia		●●●●●	Sem informação	●●●●●	Sem informação

Após uma análise inicial à Tabela 2-3, pode-se constatar que a maioria dos sistemas possui as características desejadas num sistema de automação residencial. Um dos pontos fortes de todos os sistemas são os serviços e lazer, onde se inserem as funcionalidades, que permite concluir que todos oferecem pelo menos as funcionalidades padrão.

No que diz respeito à segurança, é de realçar que alguns dos sistemas apresentam pequenas falhas como é o caso do PRO3 e o Control4, uma vez que revelam algumas lacunas no subsistema de anti-intrusão.

Relativamente à comunicação, conclui-se que o Domus encontra-se alguns níveis abaixo dos restantes, pois trata-se dum sistema muito proprietário na componente interna, o que depois afeta as restantes componentes. Também o Control4 apresenta falhas relativamente à compatibilidade, por se tratar também dum sistema com desenvolvimento restrito que afeta a utilização de equipamentos de outras marcas.

Por fim, sobre a gestão de energia denota-se que tanto o Control4 como o Domus encontram-se novamente uns níveis abaixo dos restantes, uma vez que não possuem ferramentas como os restantes para lidar com uma das vertentes mais importantes na sociedade atual.

2.3 Conclusões

Com base nos sistemas que foram apresentados das diversas empresas e com o ciclo de funcionamento do sistema global desenvolvido, pode-se concluir que existem diferenças entre o que se encontra disponível ao utilizador e a solução que é pretendida implementar.

Relativamente aos pontos de divergência, todos os sistemas já comercializados apresentam um ponto central, sendo que o sistema Creston PRO3 destaca-se dos restantes uma vez que adicionalmente faz o controlo modular das variáveis. Para o sistema desenvolvido, a comunicação, tanto externa como interna, é constituída por vários pontos de acesso ao sistema, isto é, o sistema possui uma arquitetura distribuída e não centralizada. Para uma maior exatidão, deverá ser colocada uma consola de controlo em cada divisão, permitindo ao utilizador não ter de se deslocar a um ponto do sistema para efetuar alterações, podendo alterar qualquer um dos parâmetros na divisão onde se encontra.

Este tipo de instalação permite reduzir a dimensão de cablagem no interior da habitação, tendo em conta que se cada consola monitoriza e controla o que se passa em determinada divisão, a cablagem é mais reduzida se de facto existisse apenas um nó central.

Outra das vantagens que se pode apontar a este tipo de instalação é o facto de em caso de avaria ou mau funcionamento de alguma consola, o sistema não é comprometido na sua globalidade mas sim na divisão onde se verifique a deficiência na consola. Tal não é possível nos restantes sistemas apresentados, onde o nó central embora caraterizado pela sua robustez, podem avariar levando a que o sistema fique fora de operação enquanto o problema persistir.

3. Tecnologias de Suporte

Para se gerar um produto final é necessário recorrer-se a ferramentas para que este possa ser idealizado, estruturado e por fim desenvolvido.

Neste capítulo abordam-se os conteúdos teóricos referentes às ferramentas utilizadas no desenvolvimento da aplicação "HomeTouch". Estes conteúdos irão permitir uma melhor compreensão da componente lógica do sistema "HomeDom" que será apresentado no Capítulo 4. Na secção 3.1 é apresentado o sistema operativo Android™, e as ferramentas que permitem o desenvolvimento de aplicações para este sistema. A secção 3.2 aborda o protocolo ModBus, responsável pela comunicação entre a componente física e lógica do sistema. De seguida são apresentadas as funções deste protocolo que servirão de suporte para esta mesma comunicação. Por fim, a secção 3.3 apresenta a biblioteca SQLite que permite gerir os dados armazenados para o funcionamento da aplicação.

3.1 Sistema operativo para dispositivos móveis - Android™

O sistema operativo Android™ foi criado no final do ano de 2003 pela empresa Android Inc., com o intuito de se produzirem *"... Dispositivos móveis inteligentes mais cientes da localização e preferências do seu dono ..."* (Queirós, 2013). Em julho de 2005, a Android Inc foi adquirida pela Google Inc., sendo que, após dois anos, foi apresentada ao mundo a *Open Handset Alliance*, que reúne operadores como a Telefónica, empresas de software onde se enquadra a Google e por fim fabricantes como a Sony, Samsung, entre outros.

A *Open Handset Alliance* surgiu com o objetivo de desenvolver padrões abertos para os diversos dipositivos móveis, sendo que nesse mesmo ano foi disponibilizada a versão beta do

Android SDK. No decorrer do ano de 2008, foi lançado o primeiro produto comercial, o T-Mobile's G1 HTC Dream (Figura 3-1) que utilizou o sistema Android.

Figura 3-1: T-Mobile G1 HTC Dream com SO Android 1.0.

De referir que com a evolução do Android, foram surgindo diferentes versões cuja cada versão possui uma designação em inglês de uma sobremesa ou bolo, cuja inicial segue a ordem alfabética.

Apesar de o Android ter sido inicialmente projetado para telemóveis, hoje em dia a sua aplicação transcende este tipo de dispositivo. Atualmente o sistema é aplicado em tablets, alguns netbooks, leitores MP4[3] e ainda televisões com serviço de internet (Google Inc., 2014). Com a variedade de dispositivos apresentada, o Android conseguiu adquirir a supremacia no que diz respeito à sua utilização no mercado dos sistemas operativos móveis, como se pode observar no gráfico apresentado na Figura 3-2.

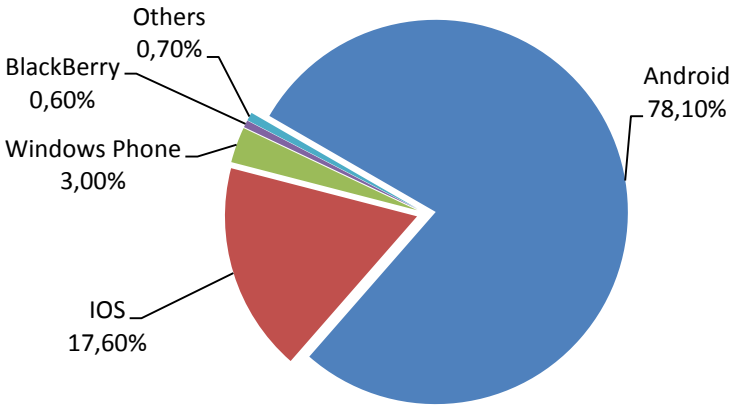


Figura 3-2: Quota de Mercado dos diferentes sistemas operativos móveis em 2014, (IDC Corporate USA, 2015).

O Android é considerado um sistema operativo *"open source"*, isto é, toda a sua arquitetura e o seu funcionamento são do domínio público, permitindo a qualquer pessoa desenvolver

[3] Extensão do formato MPEG-4 Part 14, utilizado usualmente para processamento de ficheiros multimédia.

aplicações, necessitando apenas de possuir as ferramentas de desenvolvimento adequadas para tal procedimento.

3.2 Ferramentas de desenvolvimento de aplicações

Antes da apresentação das ferramentas de desenvolvimento em si, será abordada a arquitetura do sistema Android, de modo a se compreender em que camadas do sistema estas ferramentas irão funcionar. O sistema operativo Android subdivide-se em quatro camadas ou níveis, como a Figura 3-3 apresenta.

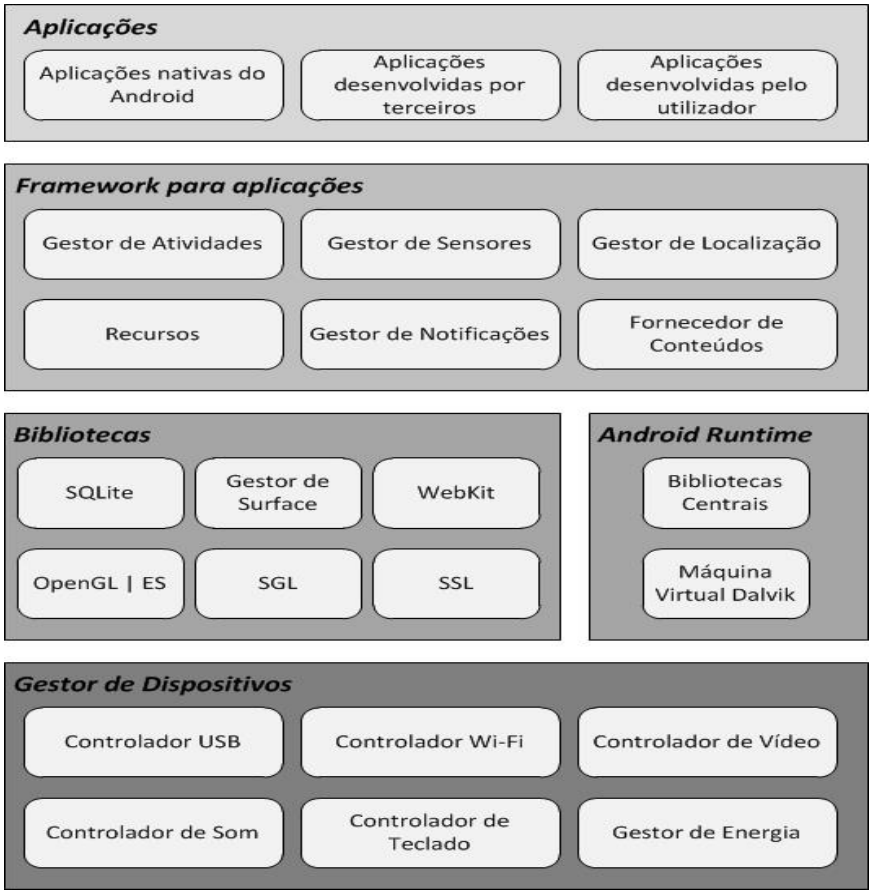


Figura 3-3: Diagrama da arquitetura do SO Android.

Relativamente às funcionalidades das diversas camadas apresentadas na Figura 3-3 tem-se que:

- **Gestor de Dispositivos –** Camada mais baixa de todo o sistema, onde se incorporam todo o tipo de drivers do hardware que compõem o dispositivo móvel;

- **Bibliotecas –** Componente da segunda camada onde estão inseridas as bibliotecas que possuem as características do Android, como por exemplo a capacidade de armazenamento de dados que é gerido pelo SQLite;

- **Android Runtime –** Mecanismo da mesma segunda camada que permite o desenvolvimento de aplicações Android por parte dos programadores. Disponibiliza as bibliotecas nucleares bem como a máquina virtual Dalvik que permite compilar as aplicações. De referir que a linguagem utilizada nesta vertente é a linguagem java;

- **Framework para aplicações –** Camada que permite aos programadores acederem às diversas funcionalidades do sistema, para que estes as possam incluir ao longo do desenvolvimento das suas aplicações;

- **Aplicações –** Camada que integra todas aplicações que já vem com o sistema operativo, bem como todas as aplicações desenvolvidas pelos programadores. Pode-se realçar que é a camada mais alta de todo o sistema e aquela que é exibida ao utilizador final.

No que diz respeito às ferramentas de desenvolvimento, estas podem ser adquiridas via web, mais concretamente nas páginas dos seus fabricantes, uma vez que o sistema em si é de utilização e consulta pública. De seguida enumeram-se as ferramentas base de suporte, que devem ser instaladas num computador para permitirem o desenvolvimento das aplicações.

3.2.1 Android SDK

Android Developement Kit (Android Developers, 2014c) é uma ferramenta que permite criar, testar e compilar e distribuir as aplicações com suporte Android.

Esta ferramenta permite excluir a utilização de dispositivos móveis reais aquando da compilação e teste da aplicação desenvolvida. Isso é possível uma vez que o sistema permite a criação de *Android Virtual Machines* (AVDs), que desempenham o papel de simuladores dos dispositivos reais.

Após a instalação desta ferramenta, o programador pode atualizar a versão do sistema ou API, através do *SDK Manager*, que permite a instalação coletiva ou individual de pacotes.

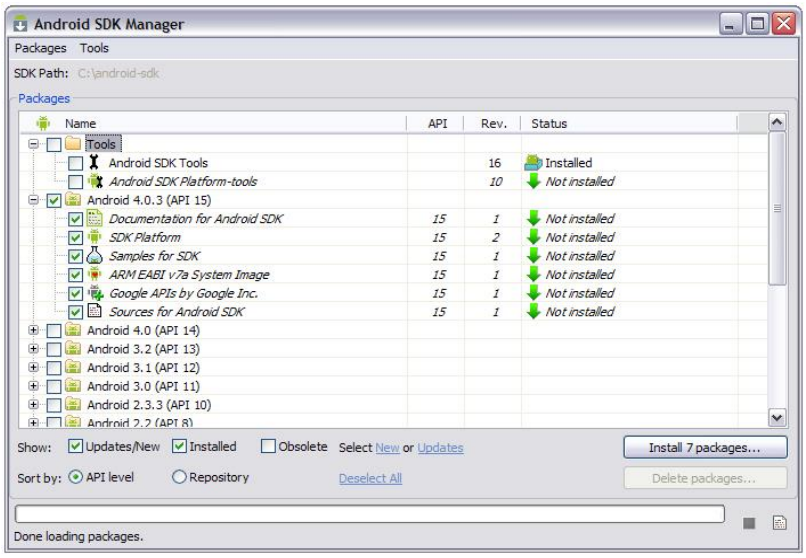


Figura 3-4: Android SDK Manager.

Como se pode visualizar na Figura 3-4, durante a instalação do SDK, é comum instalar-se a versão mais recente do sistema Android. No entanto fica assegurada a retrocompatibilidade com as versões mais antigas, estando limitado o funcionamento nas mesmas pela escolha da versão mínima do SDK por parte do programador.

3.2.2 Eclipse

Ambiente de desenvolvimento integrado ou *Integrated Developement Environment* (IDE), apresentado na Figura 3-5, recomendado para o desenvolvimento de aplicações Android (Eclipse Foundation, 2014). Este poder ser instalado e executado nas plataformas Windows, OSX e Linux.

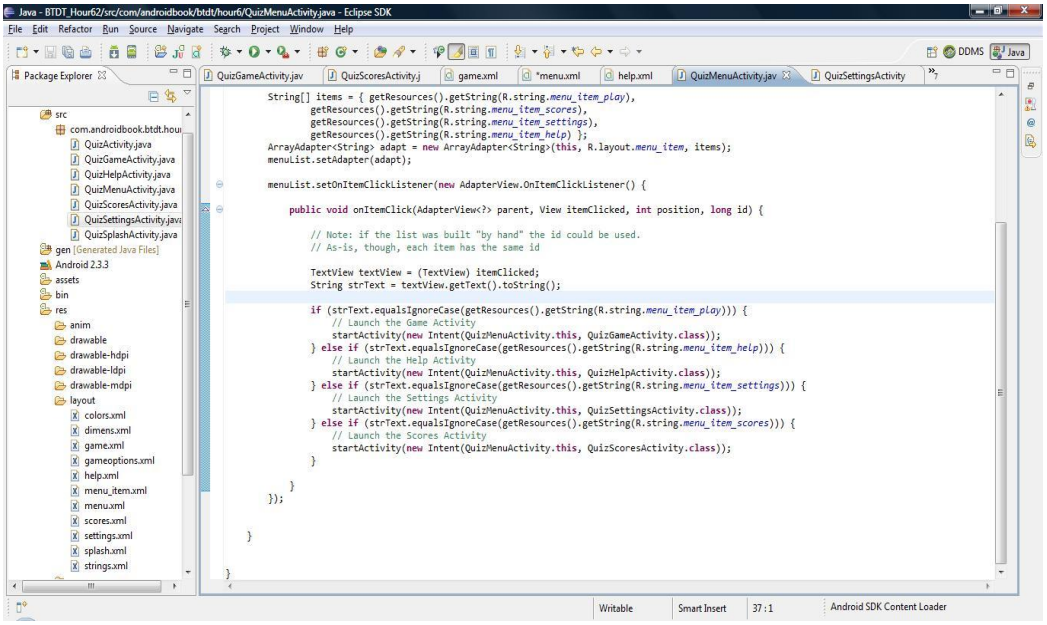


Figura 3-5: Ambiente de desenvolvimento integrado Eclipse.

3.2.3 Android ADT

O Android ADT é um plugin para o *IDE Eclipse* (Android Developers, 2014b) que permite que o eclipse possa desenvolver aplicações, permitindo a criação, compilação e a mais importante, depuração (*"Debug"*). A depuração é um processo que permite ao programador visualizar o funcionamento da aplicação enquanto esta corre no dispositivo ou no simulador. Tal processo possibilita a deteção de erros que podem levar a um mau funcionamento da aplicação ou mesmo ao seu encerramento.

Após a sua instalação, pode-se criar um simulador, AVD, que ficará interligado ao eclipse, simulando o comportamento dum dispositivo real com o Android, sendo um exemplo de um AVD o que se encontra na Figura 3-6.

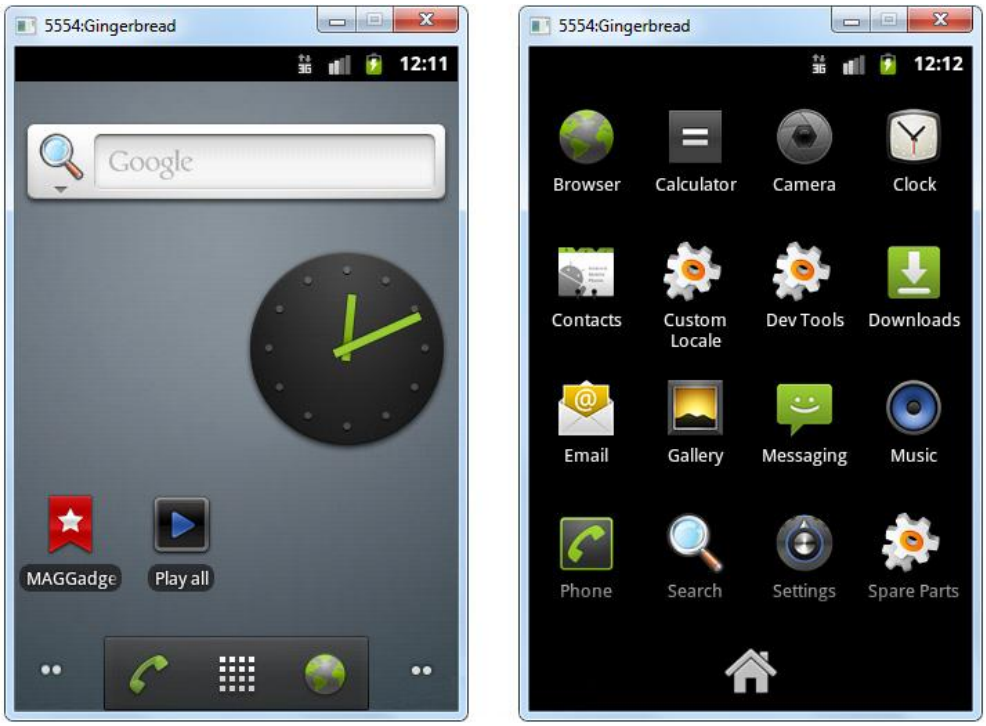


Figura 3-6: Simulador AVD com versão GingerBread.

Com estas ferramentas de desenvolvimento devidamente instaladas, o programador pode começar a desenvolver qualquer tipo de aplicação. Deste modo fica apenas a faltar a aquisição dos conhecimentos para o correto desenvolvimento das aplicações.

3.3 Desenvolvimento de aplicações Android

Nesta secção serão descritos os conteúdos mais importantes no desenvolvimento da aplicação "HomeTouch", tendo em conta a sua funcionalidade no sistema Android.

3.3.1 Activity

Iniciando-se a abordagem pelo componente de maior relevo numa aplicação, a *Activity* tem como função representar os ecrãs onde são inseridas as interfaces gráficas para o utilizador (Android Developers, 2014a). Relativamente ao seu funcionamento, a *activity* possui um conjunto de métodos de retorno *(callback methods)* que correspondem a determinados eventos que estabelecem o ciclo de vida da atividade (ver Figura 3-7).

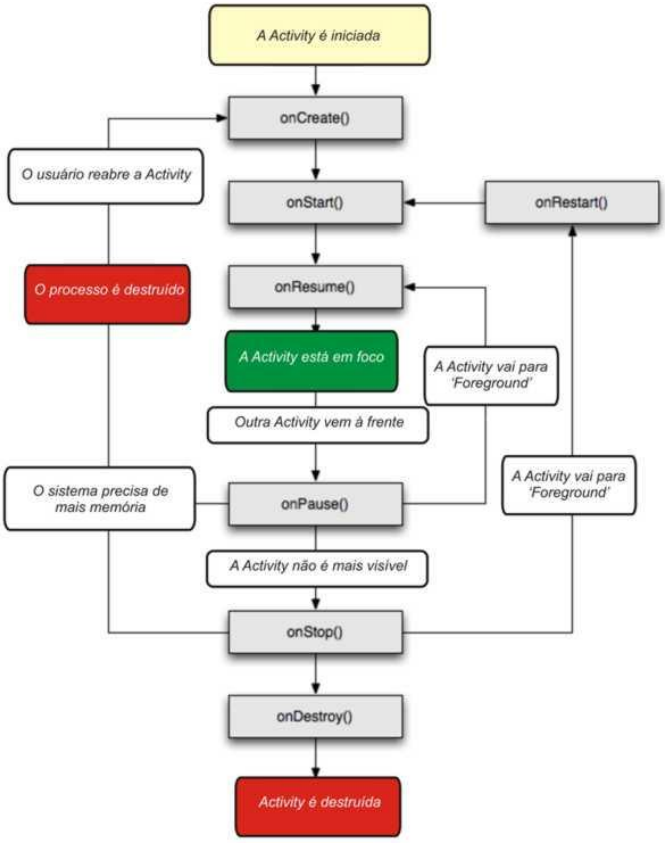


Figura 3-7: Ciclo de vida de uma *Activity*, adaptada de (Android Developers, 2014a).

Como se pode visualizar na Figura 3-7, estão representados os métodos que desencadeiam os eventos do ciclo de vida, que são:

- **onCreate() –** Método que é acionado quando a atividade é desencadeada. Método utilizado por norma para inicialização de variáveis e objetos utilizados pela *activity*;

- **onStart() –** Desencadeado após o onCreate(), permitindo que a atividade fique visível para o utilizados;

- **onResume() –** Quando a atividade permite a interação com o utilizador passando a ficar em foco na aplicação, ou seja, a atividade exibe uma interface gráfica ao utilizador para que este navegue na aplicação;

- **onPause() –** Evento que é iniciado de modo a que a atividade em execução fique em segundo plano enquanto outra atividade ganha o foco da aplicação;

- **onStop() –** Atua quando a atividade perde a visibilidade para o utilizador, isto é, o utilizador já não visualiza a interface gráfica referente a esta atividade;

- **onRestart() –** Desencadeado quando a atividade se encontra interrompida, ou seja quando o método onStop() foi acionado, permitindo reiniciar a atividade em causa;

- **onDestroy() –** Evento que é chamado quando a atividade estás prestes a ser encerrada pelo sistema, sendo que este método é sempre chamado manualmente ou automaticamente.

É de referir que de todos os métodos mencionados anteriormente só três é que possuem um carácter estático, isto é, permanecem em execução por um largo período de tempo. Esses estados são o *resumed* que é o estado que permite que o utilizador interaja com a aplicação, o *paused* onde a atividade fica em pausa, sendo que a duração dessa mesma pausa depende do comando que irá ser executado. Por fim temos o *stopped*, no qual a atividade fica invisível ao utilizador, podendo esta ficar novamente visível quando retomado o estado resumed, tendo que para isso ser reiniciada através do método *onRestart()*, passando pelo estado started que é um estado transitório.

Uma das funcionalidades da aplicação desenvolvida é a comunicação com a componente física. Essa comunicação pode ser efetuada em segundo plano, com recurso a uma classe do sistema operativo Android designada de *Async Task*.

3.3.2 Async Task

A *Async Task* (Android Developers, 2014d) permite efetuar operações de curta duração em segundo plano e publicar os resultados no processo *(thread)* principal, ou seja, permite que a comunicação seja efetuada e que o resultado dessa operação seja reportado posteriormente ao processo principal da aplicação.

Sobre o seu funcionamento, esta classe é dividida em 4 etapas:

- **onPreExecute() –** Etapa que é invocada antes da *async task* ser executada, que permite ao programador configurar ou indicar alguma informação ao utilizador da aplicação;

- **doInBackground(Params...) –** Etapa primordial nesta classe onde são passados os parâmetros a utilizar pela *async task* e onde vão ser efetuadas todas as operações necessárias para a aplicação. Esta função permite atualizar valores que sejam inicialmente apresentados na função anterior, de modo a que possam ser atualizados na etapa seguinte;

- **onProgressUpdate(Progress...) –** Utilizada quando é necessário atualizar informação na interface do utilizador enquanto a etapa anterior se encontra em execução;

- **onPostExecute(Result) –** Executada após a etapa *doInBackground()*, e que tem como única utilidade passar a informação final para o processo principal de modo a ser utilizada pela aplicação.

Por fim, no que diz respeito às interfaces para o utilizador exibidas no ecrã da *activity*, estes são constituídos por um componente designado de *View* (Android Developers, 2014g), que é o bloco de construção básico de componentes que constitui um interface. Uma *view* pode integrar um ou um grupo de componentes nativos do Android como por exemplo *layout*, *widgets* entre outros tipos de componentes.

3.3.3 Widgets

Os *Widgets* (Android Developers, 2014i) são componentes que permitem a personalização e navegação nas interfaces gráficas, ou entre interfaces da aplicação devido a ações que alguns destes possam desencadear. Tendo em conta os conceitos anteriores, de seguida, são enumerados os *widgets* que serão utilizados na aplicação que permitem ao utilizador interagir com o sistema de automação residencial (Android Developers, 2014i):

- **TextView –** *Widget* utilizado quando se pretende mostrar algum tipo de texto ao utilizador, sendo que o funcionamento deste *widget* é semelhante à caixa de texto da ferramenta Word da Microsoft;

- **ImageView –** Recorre-se a este tipo de *widget* quando se pretende mostrar algum tipo de imagem, como por exemplo um ícone ou outro tipo de imagens, sendo o seu conteúdo carregado através de recursos ou provedores de conteúdo;

- **Button** – Como o seu nome indica, trata-se dum botão que ao ser pressionado ou clicado desencadeia ações previamente programadas;

- **ImageButton –** Combina as funções de um *Button* com o *ImageView*, ou seja, trata-se dum *widget* que apesar de parecer uma imagem quando premido pode desencadear algum tipo de ação;

- **SeekBar –** *Widget* que que permite variar de forma contínua o seu valor entre uma intervalo definido aquando da sua criação, sendo que por defeito possui o intervalo entre o valor mínimo zero e o valor máximo cem;

- **NumberPicker –** Deriva do *widget TimePicker* (Android Developers, 2014h) que é usualmente utilizado nativamente pelo sistema Android para o acerto das horas e data. No caso do *widget* utilizado, possui o mesmo funcionamento que o *TimePicker* só que para um intervalo estabelecido pelo programador da aplicação;

- **Spinner –** Tipo de *widget* que permite selecionar uma opção a partir de um conjunto de opções. No seu estado padrão, este mostra a opção atual escolhida, sendo que pressionando o *spinner* esta apresenta todas as opções para que o utilizador escolha novamente a sua opção;

- **CheckBox** – Componente que permite ao utilizador selecionar uma ou mais opções de um mesmo conjunto. A sua configuração padrão passa por uma lista vertical das opções disponíveis para o utilizador.

3.4 Protocolo de comunicação – ModBus

O protocolo ModBus foi desenvolvido pela empresa Modicon no final da década de 70, com o objetivo de transferir informação de entrada e saída, de forma discreta entre dispositivos e um sistema de supervisão (Acromag Incorporated, 2005). Atualmente é um protocolo implementado em grande escala nos processos industriais com alto índice de aceitação, sendo um protocolo de domínio público, ou seja, pode ser usado e implementado sem custos adicionais.

Relativamente ao funcionamento do ModBus, este protocolo utiliza a comunicação mestre/escravo ou cliente/servidor, isto é, o escravo só envia informações para a rede se o mestre assim o solicitar. No entanto, pode ser solicitado informação a mais que um escravo em modo de mensagem difusa *(broadcast)*. Este modo permite que o mestre envie a mesma requisição a vários escravos, sendo os escravos solicitados respondem individualmente para o mestre.

Na Figura 3-8 encontra-se representado um diagrama de funcionamento deste protocolo.

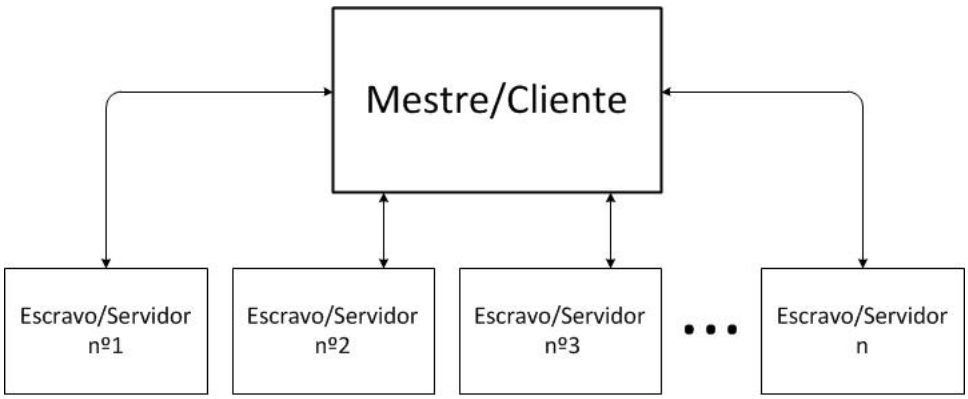


Figura 3-8: Diagrama com protocolo ModBus.

É de salientar que, como indica a Figura 3-8, por cada rede que implemente o ModBus, apenas pode existir um único mestre ou cliente, enquanto para os escravos ou servidores, este número possa ser mais elevado. Quando aos escravos, estes podem ser de qualquer tipo como transdutores, equipamentos de medição entre outros, sendo que o mestre tem de ser um equipamento de processamento de dados, como os tradicionais plc[4] ou computadores com software capaz de implementar o ModBus.

3.4.1 Modos de operação do protocolo modbus

Devido à evolução tecnológica ao nível das redes de comunicação, também este protocolo evoluiu no sentido de se adaptar a esta evolução, surgindo assim as variantes do ModBus (Thomas, 2008)**,**(Acromag Incorporated, 2005) que são:

- **ModBus RTU** – Modo de transmissão que é incorporado na comunicação série. Este tipo de modo é utilizado quando a informação é transmitida em formato binário;

- **ModBus ASCII –** Também um modo utilizado na comunicação série, sendo que diverge para o anterior na codificação da mensagem que passa do formato binário para a codificação ASCII**;**

- **ModBus TCP/IP –** Modo de transmissão que como o seu nome indica assenta na transmissão TCP/IP, onde os dados são encapsulados em formato binário em tramas TCP para a utilização na *ethernet*;

- **ModBus Plus –** Modo que implementa recursos adicionais face aos anteriores, sendo à data ainda um protocolo fechado contrariamente aos anteriores, podendo ser utilizado sob licença do fabricante.

[4] Controlador Lógico Programável (*Programmable logic controller*), que é constituído por um microprocessador que permite desempenhar funções de controlo.

Dos quatro modos apresentados, excluindo à partida o último, é implementado, no desenvolvimento da aplicação em causa, o modo TCP/IP, uma vez que as consolas de controlo possuem compatibilidade com este modo. O facto do protocolo TCP/IP ser implementado nas redes Wi-Fi, permite que a aplicação possa correr em qualquer dispositivo móvel que possua o sistema operativo Android, uma vez que o adaptador Wi-Fi é atualmente um componente padrão em qualquer equipamento que possua o Android.

Relativamente ao ModBus TCP/IP, pode-se referir que este modo surgiu devido ao TCP/IP ser um protocolo robusto mundialmente utilizado nas redes de comunicação. Sobre o seu funcionamento não é mais que a aplicação do modo RTU, em que o transporte é efetuado através do TCP/IP, que visa garantir o tratamento e encaminhamento da informação entre cliente/servidor. Pode-se deduzir neste caso que o protocolo ModBus é responsável pelo tratamento de dados, enquanto o TCP/IP é responsável pelo seu encapsulamento e transmissão (Acromag Incorporated, 2005).

Quanto ao processo de requisição e receção de informação por parte do cliente/servidor, este divide-se nas seguintes quatro etapas que são:

- **ModBus Request (Pedido) –** Quando o cliente envia uma mensagem para a rede para iniciar uma transferência de informação;
- **ModBus Indication (Informação) –** Quando a mensagem é recebida pelo servidor;
- **ModBus Response (Resposta)** – Mensagem proveniente do servidor com a informação pedida;
- **ModBus Confirmation (Confirmação) –** Sinal que a mensagem foi recebida pelo cliente.

Este processo encontra-se esquematizado na Figura 3-9, com a inserção das etapas tendo em conta o sentido da comunicação.

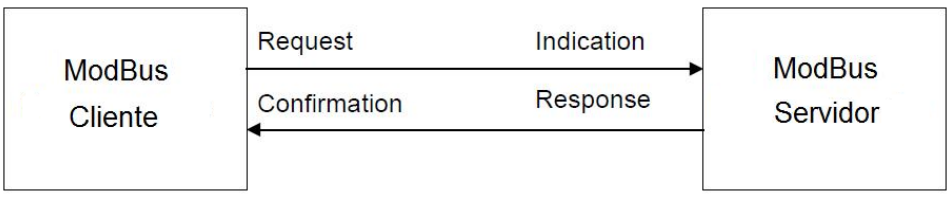


Figura 3-9: Etapas do protocolo ModBus TCP/IP, adaptada de (Modbus-IDA, 2006).

Para se efetuarem as etapas anteriores indicadas na Figura 3-9, é preciso construir a trama ModBus que irá permitir solicitar e recolher a informação desejada. Como o transporte da

informação se faz por via do TCP/IP, nesse caso a trama sofre pequenas alterações surgindo a trama apresentada na Figura 3-10.

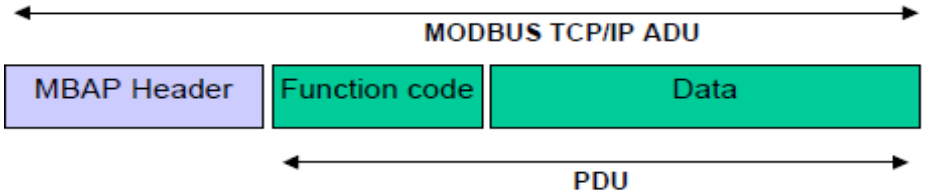


Figura 3-10 – Trama ModBus TCP/IP, adaptada de (Modbus-IDA, 2006).

Relativamente à trama visualizada na Figura 3-10, denota-se que desapareceram os campos *Aditional Address* e *Error Check* da trama standard e que surgiu o campo *MBAP Header*. Esta transformação deve-se à utilização do protocolo TCP/IP que já implementa o *Acknowledge*, isto é, permite saber se a trama enviada chegou ao destino sem corrupção dos dados.
Pode-se deduzir que o *MBAP Header*(ver Figura 3-11) irá possuir informações que permite com que a informação chegue corretamente ao destino.

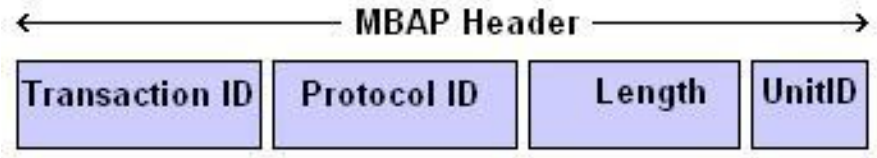


Figura 3-11: Construção da subtrama *MBAP Header.*

Na Figura 3-11 estão representados os quatro campos que compõem o *"MBAP Header"* e que possuem as seguintes funções:

- **Transaction Identifier –** Utilizado no emparelhamento da transferência, uma vez que o escravo/servidor irá copiar este valor para a posterior resposta;

- **Protocol Identifier –** É utilizado para a multiplexação intra-sistema. O protocolo MODBUS é identificado pelo valor 0;

- **Length –** Indica o comprimento dos campos posteriores a este. É utilizado na vertente do TCP/IP para que o destinatário saiba o tamanho máximo da mensagem, mesmo que esta seja dividida em vários pacotes. Este parâmetro permite verificar a integridade da mensagem, ou seja, ver se esta foi corrompida em algum momento;

- **Unit Identifier –** Utilizado para fins de roteamento no interior do sistema, ou seja, permite comunicar com os servidores entre diferentes modos do ModBus. O valor deste campo e definido pelo cliente aquando do *"Request"* e deve ser permanecer igual aquando da *"Response"* por parte do servidor.

3.4.2 Funções do protocolo ModBus

Referente à transferência de informação numa rede ModBus, esta é efetuada com o recurso a funções, que retratam o tipo de operação a efetuar entre cliente e servidor. Estas funções podem ser por exemplo escrever ou ler um registo ou então ativar ou desativar uma saída.

No que diz respeito às funções que incorporam este protocolo, da lista existente (ModBus, 2012), foram escolhidas as seguintes funções para a transferência de informação:

- **Write Single Register (0x06) –** Esta função permite escrever um registo fixo no dispositivo de controlo, sendo que após a escrita deste registo nesse mesmo dispositivo, é enviado para o remetente uma resposta com o conteúdo do registo escrito no destino.

  A utilização desta função deve-se ao facto das consolas de controlo que recebem a informação da aplicação, permitem a receção de registos para ajuste dos valores dos parâmetros disponibilizados ao utilizador.

  No Anexo B pode-se consultar o diagrama que explica todo o processo de funcionamento desta função do ModBus, desde o primeiro momento em que o cliente exerce o "*Request"*.

  Relativamente ao envio da informação, este é executado com o recurso ao campo *"Register Value"*, inserido no campo *"Data"* da trama do ModBus. Os restantes campos mencionados na Figura 3-12 irão permanecer inalterados neste processo.

Request/Response

Function code	1 Byte	**0x06**
Register Address	2 Bytes	0x0000 to 0xFFFF
Register Value	2 Bytes	0x0000 to 0xFFFF

Error

Error code	1 Byte	**0x86**
Exception code	1 Byte	01 or 02 or 03 or 04

Figura 3-12: Campos que incorporam a trama ModBus Write Single Register, adaptado de (ModBus, 2012).

- **Read/Write Multiple Registers –** Esta função permite a leitura ou escrita de um ou mais registos numa única transferência com o ModBus. Para o caso de serem mais que um registo a ser lido ou escrito, é necessário especificar o número de registos afetos a esta operação bem como a posição de qual se vai iniciar ou a leitura/escrita de registos. Para esta função, a resposta proveniente do servidor é constituída pelo número de registos acionados, bem como a informação desses mesmos registos. Esta função irá ser utilizada quando for necessário recolher o

valor dos parâmetros que se encontram na consola aquando do carregar de algumas funções por parte da aplicação.

No que diz respeito ao diagrama de funcionamento desta função, este pode ser consultado no Anexo C.

Também para este caso, têm-se na Figura 3-13, os campos utilizados por esta função para que a informação seja corretamente processada.

Request

Function code	1 Byte	**0x17**
Read Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity to Read	2 Bytes	0x0001 to 0x007D
Write Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity to Write	2 Bytes	0x0001 to 0X0079
Write Byte Count	1 Byte	2 x **N***
Write Registers Value	**N***x 2 Bytes	

***N** = Quantity to Write

Response

Function code	1 Byte	**0x17**
Byte Count	1 Byte	2 x **N'***
Read Registers value	**N'*** x 2 Bytes	

***N'** = Quantity to Read

Error

Error code	1 Byte	**0x97**
Exception code	1 Byte	01 or 02 or 03 or 04

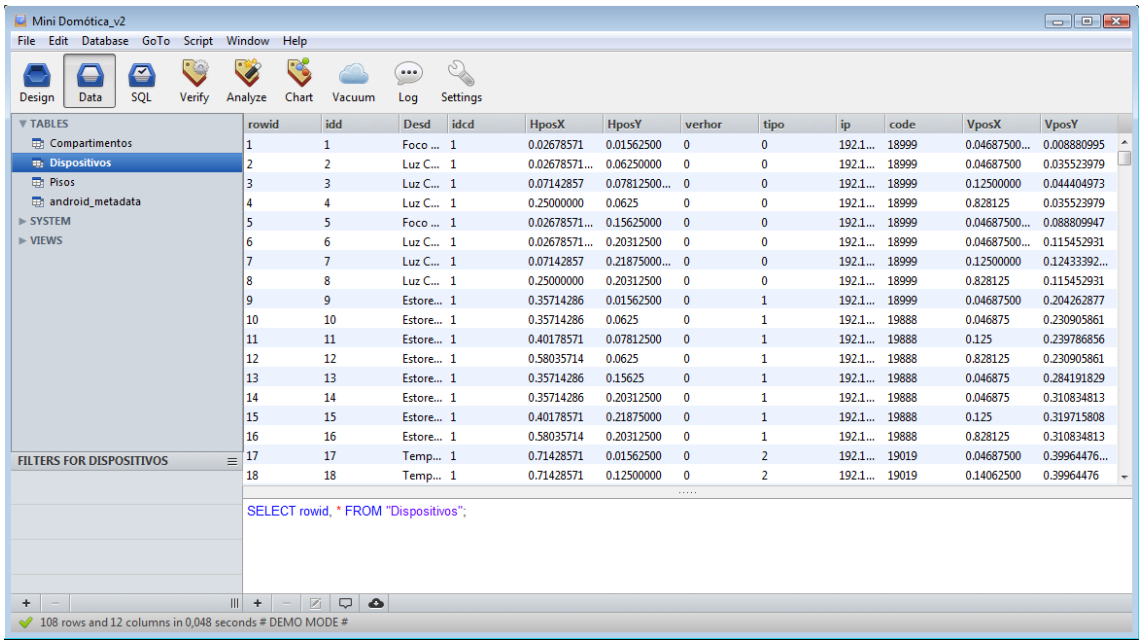
Figura 3-13: Campos que incorporam a trama ModBus Read/Write Multiple Register, adaptado de (ModBus, 2012).

3.5 Armazenamento de dados – SQLite

O SQLite é uma biblioteca que implementa um motor de base de dados SQL, sem a necessidade da existência dum servidor dedicado (SQLite, 2014). Por este motivo é incorporada no sistema operativo Android, podendo ser utilizada quando necessário por parte das aplicações. Aliada à característica anterior, o tamanho da biblioteca no pior dos casos, pode assumir um valor de 500kB, o que o torna esta ferramenta ideal para sistemas com armazenamento limitado como os dispositivos móveis.

O ficheiro criado pelo SQLite designa-se de base de dados, que simplificando, não passa duma compilação de dados estruturados, organizados e armazenados de forma constante (Damas, 2005).

Relativamente a uma ferramenta para utilização do SQLite, é usualmente utilizada o SQLite Manager (SQLabs LLC, 2014) para visualização do conteúdo da base dados, pois é considerada uma das ferramentas de eleição para o tratamento das bases de dados no formato SQLite. Esta aplicação permite exercer ações sobre o conteúdo da base de dados, utilizando para tal a linguagem SQL e os seus comandos.



rowid	idd	Desd	idcd	HposX	HposY	verhor	tipo	ip	code	VposX	VposY
1	1	Foco ...	1	0.02678571	0.01562500	0	0	192.1...	18999	0.04687500...	0.008880995
2	2	Luz C...	1	0.02678571...	0.06250000	0	0	192.1...	18999	0.04687500	0.035523979
3	3	Luz C...	1	0.07142857	0.07812500...	0	0	192.1...	18999	0.12500000	0.044404973
4	4	Luz C...	1	0.25000000	0.0625	0	0	192.1...	18999	0.828125	0.035523979
5	5	Foco ...	1	0.02678571...	0.15625000	0	0	192.1...	18999	0.04687500...	0.088809947
6	6	Luz C...	1	0.02678571...	0.20312500	0	0	192.1...	18999	0.04687500...	0.115452931
7	7	Luz C...	1	0.07142857	0.21875000...	0	0	192.1...	18999	0.12500000	0.12433392...
8	8	Luz C...	1	0.25000000	0.20312500	0	0	192.1...	18999	0.828125	0.115452931
9	9	Estore...	1	0.35714286	0.01562500	0	1	192.1...	18999	0.04687500	0.204262877
10	10	Estore...	1	0.35714286	0.0625	0	1	192.1...	19888	0.046875	0.230905861
11	11	Estore...	1	0.40178571	0.07812500	0	1	192.1...	19888	0.125	0.239786856
12	12	Estore...	1	0.58035714	0.0625	0	1	192.1...	19888	0.828125	0.230905861
13	13	Estore...	1	0.35714286	0.15625	0	1	192.1...	19888	0.046875	0.284191829
14	14	Estore...	1	0.35714286	0.20312500	0	1	192.1...	19888	0.046875	0.310834813
15	15	Estore...	1	0.40178571	0.21875000	0	1	192.1...	19888	0.125	0.319715808
16	16	Estore...	1	0.58035714	0.20312500	0	1	192.1...	19888	0.828125	0.310834813
17	17	Temp...	1	0.71428571	0.01562500	0	2	192.1...	19019	0.04687500	0.39964476...
18	18	Temp...	1	0.71428571	0.12500000	0	2	192.1...	19019	0.14062500	0.39964476

Figura 3-14: SGBD SQLite Manager.

No que diz respeito à utilização desta ferramenta no sistema operativo Android, é necessário recorrer-se a um tipo de classe que não foi referido na secção 3.3. A classe em causa é a *SQLiteOpenHelper* (Android Developers, 2014f), que permite a criação de uma base de dados e adicionalmente possibilita gerir a versão da mesma. Associada a este tipo de classe está a SQLiteDatabase (Android Developers, 2014e), que possui os métodos que permitem por exemplo definir o tipo de acesso à base de dados.

Um dos parâmetros a ter em conta na utilização da base de dados e o modo de acesso à mesma. Este acesso pode ser do tipo leitura, quando se pretende obter algum valor do conteúdo da base de dados, ou do tipo escrita que é utilizado para adicionar, modificar ou apagar informação da base de dados.

4. Arquitetura do Sistema

Para o desenvolvimento de um sistema de automação, é necessário conceber-se a estrutura ao mesmo, de modo a se agruparem todos os componentes que o constituem tendo em conta o seu tipo e função.

Este capítulo apresenta a arquitetura do sistema quer do ponto de vista da componente física e lógica (aplicação desenvolvida para interagir com a componente física).

Na secção 4.1 é efetuada uma breve introdução com o intuito de apresentar o sistema do ponto de vista global quanto ao seu funcionamento. A secção 4.2 apresenta mais detalhadamente a componente física relativamente ao seu funcionamento, e posteriormente a informação necessária para efetuar a interação entre aplicação e esta componente. Por fim na última secção aborda-se a componente lógica, onde é apresentada a arquitetura que permite a aplicação não só comunicar com a componente física, mas também oferecer ao utilizador uma gama de funcionalidades para maximizar a interação entre sistema e utilizador.

4.1 Sistema “ HomeDom “

A arquitetura do sistema, esta encontra-se dividida em duas componentes: a arquitetura física e lógica como se pode visualizar na Figura 4-1 onde está representada, na sua forma mais simplista a arquitetura global do sistema.

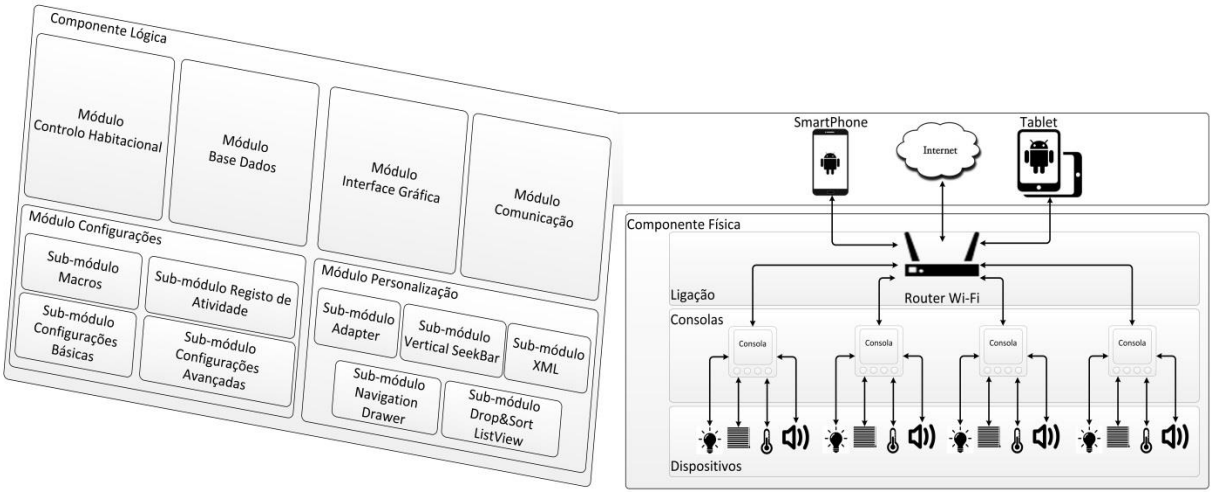


Figura 4-1: Arquitetura do sistema.

Como se pode visualizar na Figura 4-1, a arquitetura física é constituída por uma consola inteligente e independente por divisão. A consola controla diretamente os diversos dispositivos existentes nessa divisão, sem a necessidade de elementos externos.

A componente lógica ou aplicação, vai interagir com as consolas individualmente, solicitando informação ou enviando comandos consoante as instruções recebidas por parte do utilizador.

4.2 Componente Física

A componente física, ilustrada na Figura 4-1, possuiu a particularidade de cada consola se conectar diretamente a um router que posteriormente se encontra interligado com a rede Wi-Fi.

A consola, representada na Figura 4-1, é responsável pela supervisão em cada divisão, sendo que o equipamento escolhido pela empresa promotora é a consola OCS XL4 concebida pela empresa Horner APG ilustrada na Figura 4-2.

Figura 4-2: Consola OCS XL4.

Sobre a constituição da consola, de referir que esta consola vem equipada com quatro tipos de barramentos (E/S analógicas e digitais) que se encontram apresentados a Figura 4-3. Estes tipos de entradas e saídas permitem ao sistema obter uma maior compatibilidade na componente física no que diz respeito aos dispositivos periféricos. A integração destes barramentos na consola permite a recolha de toda a informação proveniente dos diferentes dispositivos de campo, sejam eles do tipo analógico (sensores de temperatura e posição) ou digital (interruptores, sensores de presença).

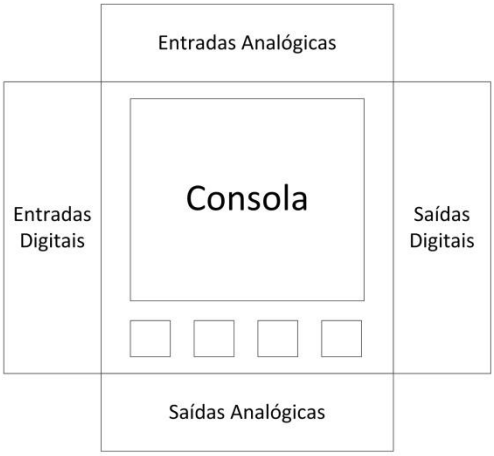


Figura 4-3: Constituição da consola de comando.

Referente à comunicação entre as duas componentes (que se encontra exemplificada na camada de ligação da arquitetura), que como já foi referido no capítulo 3, é efetuada com o recurso ao protocolo de comunicação ModBus TCP/IP. Este protocolo interliga-se perfeitamente com a programação interna da consola adotada pela empresa promotora, uma vez que esta utiliza registos para gestão e controle da informação proveniente dos dispositivos de campo.

A comunicação desenrola-se através de várias operações, quer na aplicação quer durante na transferência da mesma para a consola de comando. Como se trata duma comunicação biderecional, as operações terão em conta o sentido da comunicação, nomeadamente o sentido de escrita (aplicação-consola) e o sentido de leitura (consola-aplicação).

A Figura 4-4 ilustra todas as possíveis operações e funções quer no funcionamento da aplicação e na transferência de informação com o recurso ao ModBus TCP/IP. De referir que a tranferência de informação é efetuada por via de funções também estas ilustradas na Figura 4-4.

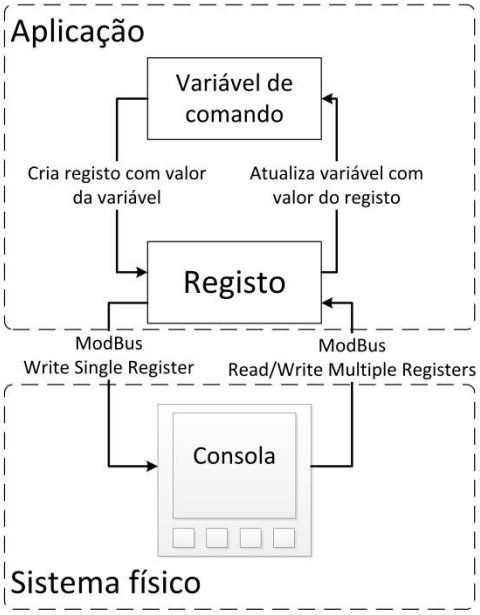


Figura 4-4: Comunicação interna no sistema (aplicação – componente física).

No que diz respeito à comunicação entre os dispositivos e a consola diz respeito, utilizando os barramentos já referidos, na Figura 4-5 encontra-se representada uma possível hipótese de comunicação para os comandos de iluminação, temperatura e estores.

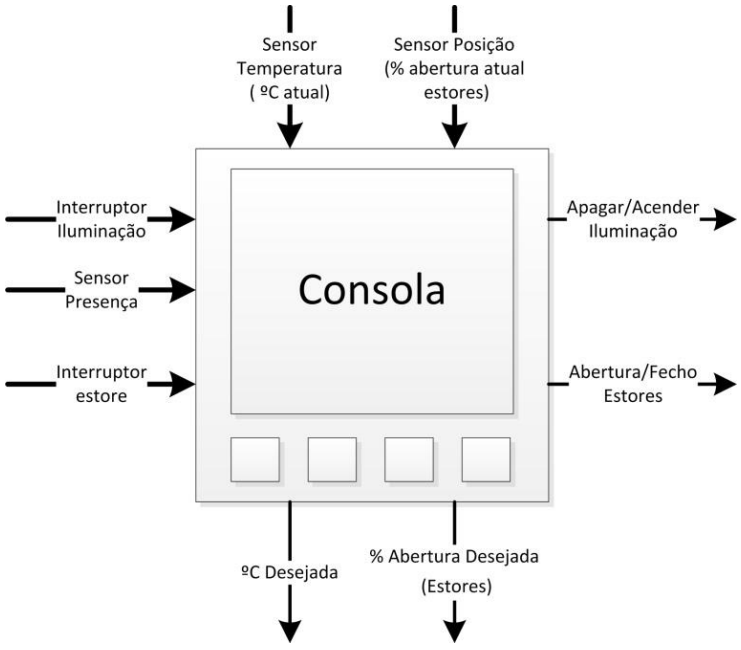


Figura 4-5: Comunicação interna do sistema (consola – dispositivos).

4.3 Componente Lógica

A componente lógica ou aplicação, é desenvolvida tendo em conta o funcionamento do sistema apresentado na secção 4.1, dando origem aos interfaces que serão apresentados no capítulo 5.

Como foi possível observar na Figura 4-1, esta encontra-se subdividida por módulos, onde cada módulo terá uma função específica na arquitetura, possuindo no seu interior as diferentes classes ou atividades que permitem desempenhar as suas tarefas. Na Figura 4-6 está representada a arquitetura da aplicação do ponto de vista dos módulos.

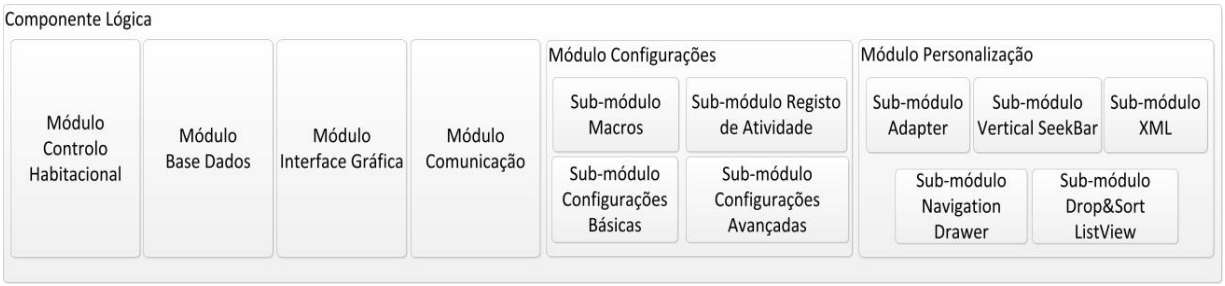


Figura 4-6: Arquitetura da componente lógica (aplicação).

Relativamente aos módulos apresentados na Figura 4-6, o módulo "Controlo Habitacional" caracteriza-se por ser o módulo central da aplicação, que como o seu nome indica permite ao utilizador exercer o controlo sobre a sua habitação no que ao sistema "HomeDom" diz respeito.

O módulo "Base de Dados" é responsável pela gestão da informação necessária ao funcionamento da aplicação e que, como o seu nome indica, essa informação encontra-se numa base de dados que é guardada localmente no dispositivo móvel.

Referente ao módulo "Comunicação", este possuiu a função de efetuar a comunicação entre a aplicação e as consolas do sistema, através do já mencionado protocolo ModBus.

No que diz respeito ao módulo "Interface Gráfica", este módulo integra a classe que é responsável pela criação e configuração das *views* nos *layouts* dinâmicos, mais concretamente no módulo de controlo habitacional onde o layout se adapta às escolhas do utilizador.

O módulo "Configurações", como o seu nome indica, possui todas as configurações e funcionalidades consideradas essenciais para melhorar a experiência de utilização da aplicação, como alteração dos nomes dos pisos, divisões entre outras opções. Devido ao facto das configurações serem compostas por vários tipos, este módulo encontra-se dividido em sub-módulos, como foi possível visualizar na Figura 4-1, sendo que cada sub-módulo possui funções distintas apenas compartilhadas pelo módulo central (controlo habitacional).

Por último, o módulo de personalização integra todos os componentes, sejam eles do tipo *views*, *adapter* ou outro tipo, que a aplicação implemente mas que o sistema operativo Android não possua nativamente.

4.3.1 Módulo Controlo Habitacional

O módulo "Controlo Habitacional", responsável pelo controlo, implementa uma série de comandos que permitem controlar os parâmetros como temperatura, luminosidade, estores e ainda o som. Os comandos presentes também se encontram programados nas diversas consolas de modo a que a informação presente nos dois pontos de acesso no sistema esteja sempre atualizada e em concordância.

Este módulo disponibiliza funções que permitem interagir com os restantes módulos, estar em permanente escuta para deteção de alteração de algum parâmetro do sistema e por fim proceder periodicamente à atualização da informação proveniente da componente física.

Tais funções vão ser incorporadas na atividade *ControloHabitacional* através dos métodos representados na Figura 4-7.

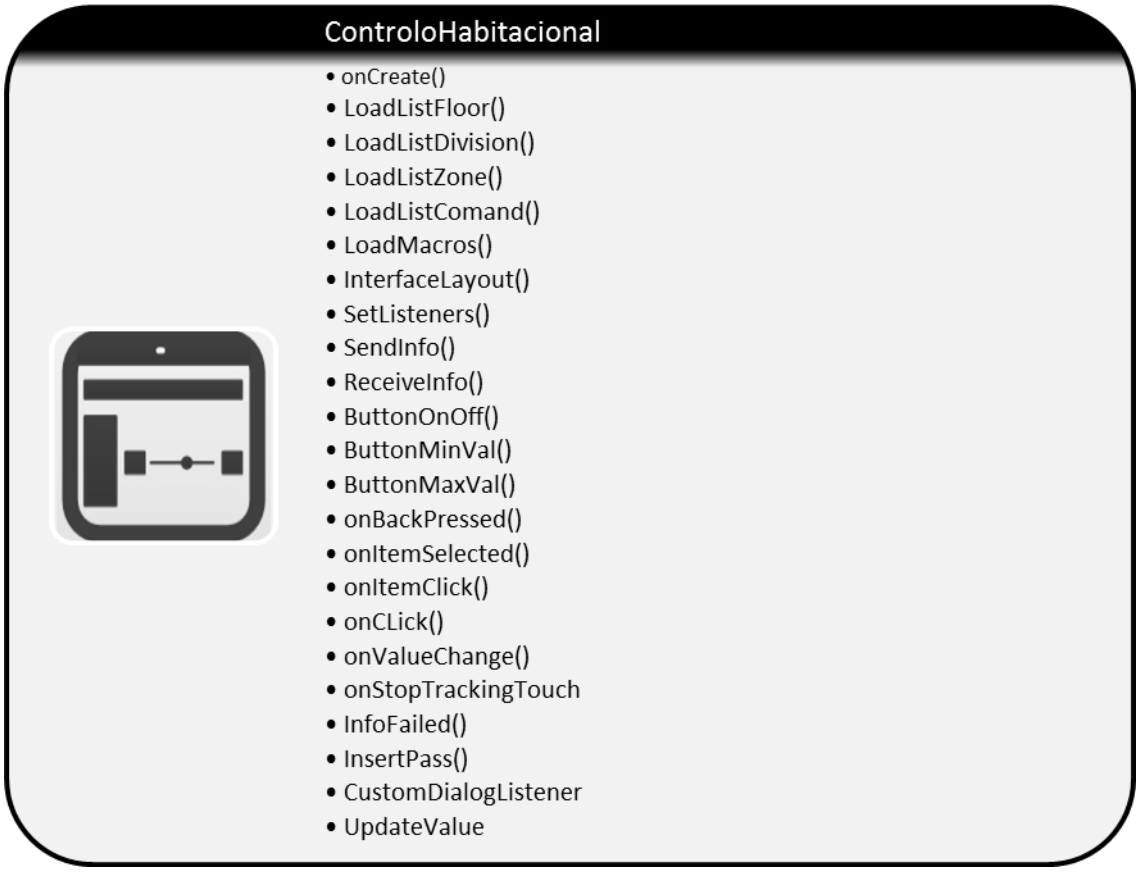


Figura 4-7: Métodos da classe *ControloHabitacional*.

Referentes aos métodos apresentados na Figura 4-7, estes possuem as seguintes funções:

- **Oncreate() –** Método pré-definido duma *activity*, onde são inicializadas as variáveis globais utilizadas nesta classe e ainda objetos para a utilização das classes secundárias;
- **LoadListFloor() –** Método que irá carregar a informação relativamente aos pisos existentes na habitação no *widget* a ser utilizados;
- **LoadListDivision() –** Mesma função que o anterior só que para o caso das divisões, sendo que as divisões são carregadas consoante a seleção do piso;
- **LoadListZone() –** A sua função vai no seguimento dos métodos anteriores, só que para o caso das zonas que também são carregadas tendo em conta a seleção da divisão e do tipo de comando selecionado;
- **LoadListComand() –** Método que carrega os tipos de comandos existentes, sendo eles a iluminação, estores, temperatura, som;
- **InterfaceLayout() –** Responsável pela gestão do layout onde são exibidas as diversas opções de controlo para o utilizador. Essas opções terão em conta o tipo de comando selecionado previamente;
- **SetListeners() –** Tem como função ativar métodos nativos do Android que permitem colocar os diferentes *widgets* à escuta de eventuais alterações ao seu estado inicial;
- **SendInfo() –** Desencadeia o processo de comunicação entre aplicação e a consola respetiva para permitir o envio da informação dos parâmetros a alterar na consola;
- **ReceiveInfo() –** Possui o mesmo propósito que o "SendInfo()", sendo que para este caso a comunicação é iniciada para que a aplicação recolha a informação relativa a um ou mais parâmetros da consola respetiva.
- **ButtonOnOff() –** Método que é desencadeado caso sejam pressionados os botões on ou off no *layouts* de controlo. Dependendo do tipo de botão pressionado e o tipo de comando selecionado, a aplicação desencadeia determinadas ações;
- **ButtonMinVal() –** Como o seu nome deixa prever, este método vais ser chamado quando algum botão de valor mínimo é pressionado, isto é, nos diferentes *layouts* vai existir um botão que coloca um determinado *widget* de controlo no seu valor mínimo automaticamente;
- **ButtonMaxVal() –** Possui a mesma função que o anterior só que para o valor máximo;
- **onBackPressed() –** Método nativo do Android que permite detetar quando o botão "Back" do dispositivo é pressionado, desencadeado a ação de sair da aplicação;
- **onItemSelected() –** Implementado pelo *Spinner* que tem como função detetar quando a seleção do mesmo é alterada, devolvendo a posição da nova seleção;
- **onItemClick() –** Método com mesma função que o anterior só que para o caso da *Listview*;
- **onClick() –** Método utilizado pelo *Button* que é desencadeado sempre que um botão é pressionado. De referir que é o método responsável pelo chamamento dos métodos ButtonOnOff(), ButtonMinVal() e ButtonMaxVal();
- **onValueChange() –** Implementado pelo *NumberPicker* que permite detetar quando o a sua posição muda;

- **onStopTrackingTouch() –** Método utilizado pela *SeekBar* que devolve o valor do seu progresso apenas quando o utilizador retira o dedo do ecrã;
- **InfoFailed() –** Método que é chamado caso o método LoadListFloor() ou LoadListDivision() reporte que não exista informação acerca dos pisos ou divisões respetivamente, bloqueando o funcionamento da aplicação. Neste caso é indicado ao utilizador que entre nas "Configurações Avançadas" para resolver o problema ou então que saia da aplicação, caso não tenha acesso a essas mesmas configurações;
- **InsertPass() –** Desencadeado quando o utilizador pretende aceder ao menu de configurações avançadas que se encontra restrito e só pode ser acedido por via da inserção de uma password;
- **CustomDialogListener –** Classe adicional que tem como função específica lidar com o interface nativo do Android *AlertDialog*. Foi implementada devido ao funcionamento standard do *AlertDialog* que por norma fecha o interface quando pressionado um dos botões possíveis, o que nem sempre é o mais correto para a aplicação desenvolvida.
- **UpdateValue –** Uma segunda classe adicional que como o seu nome sugere, vai ser a responsável pela atualização periódica do parâmetro selecionado, isto se a aplicação estiver em execução e não houver nenhuma alteração nesse período de tempo.

4.3.2 Módulo Base de Dados

O módulo Base de Dados terá como funções detetar a presença da base de dados no dispositivo e em caso de insucesso criar a base de dados. Para além disso permite a atualização da mesma quando necessário e fornece informações aos diferentes módulos quando solicitado.

Relativamente à sua constituição, este módulo possui a classe *SQLiteOpenHelper*, cuja função foi referida no capítulo 3, que é constituída pelos diversos métodos presentes na Figura 4-8.

Figura 4-8: Constituição da classe BaseDados.

De seguida serão explicadas as funções individuais de cada método à semelhança do que foi exposto no módulo de controlo habitacional:

- **CheckBD() –** Método que verifica a existência da base de dados no dispositivo onde se integra a aplicação. Esta verificação é feita com recurso à função *openDataBase*, que verifica a existência do objeto a que está associado. Após este procedimento, se a base de dados existir, a aplicação segue para a próxima etapa sendo que na ausência da base de dados, ela é criada com o recurso à função *openOrCreateDatabase*;
- **CreateTables() –** Método desencadeado aquando da criação da base de dados que cria as diversas tabelas necessárias para o armazenamento da informação;
- **UpdateFloor() –** Tem como função fazer a inserção de informação na tabela dos pisos.
- **UpdateDivision() –** Mesma funcionalidade que o anterior só que para a tabela das divisões;

- **UpdateZone() –** Possui a mesma função que os anteriores, só que insere a informação na tabela das zonas;
- **UpdateMacro() –** Método com o mesmo propósito que os anteriores, carregando a tabela relativa às macros;
- **UpdateMacroVal() –** Permite carregar a tabela das macros com os diferentes comandos;
- **UpdateField() –** Método que pode ser utilizado para diversas operações para atualizar um determinado campo na base de dados. No entanto a sua função principal é a atualização das designações dos espaços físicos quando alteradas pelo utilizador;
- **UpdateIconName() –** Último método deste tipo, sendo que para este caso vai apenas atualizar o nome das imagens a utilizar que estão associadas às diferentes designações dos pisos, divisões ou zonas;
- **GetFloors() –** Primeiro método do conjunto que vai ser responsável pela aquisição da informação no interior da base de dados. Neste caso específico é para a recolha de informação relativa aos pisos;
- **GetAllFloorsInfo() –** Método análogo ao anterior só com a alteração que recolhe toda a informação presente na tabela dos pisos, estando esta informação separada por piso;
- **GetDivisions() –** Função idêntica ao GetFloors(), sendo que para este caso, a informação que será extraída está relacionada com as divisões num dado piso selecionado pelo utilizador;
- **GetAllDivisionsInfo() –** Mesmo princípio que o método GetAllFloorsInfo() só que para as divisões, tendo em conta o piso em questão;
- **GetZones() –** Método com função igual ao GetDivisions() só que para o caso das zonas, onde a variável em causa será a divisão escolhida;
- **GetZonesInfo() –** Possui a mesma função que os GetAllFloorsInfo() e GetAllDivisionsInfo(), só que para uma única zona específica;
- **GetAllZonesInfo() –** Função idêntica aos GetAllFloorsInfo() e GetAllDivisionsInfo() só que a recolha recairá nas zonas existentes de uma dada divisão;
- **GetMacros() –** Método que tem como função recolher a informação relativamente à designação de cada uma das macros existentes, para ser posteriormente ser posta à disposição do utilizador;
- **GetAllMacroInfo() –** Possui a mesma função dos restantes métodos deste tipo;
- **GetMacroValInfo() –** Método que recolhe os valores dos diversos comandos afetos à macro selecionada previamente;
- **GetAllMacroValInfo() –** Utilizado para recolher toda a informação da tabela dos valores das macros;
- **GetComInfo() –** Chamado para adquirir informação que é utilizada pelo módulo de comunicação, para que seja estabelecida a comunicação entre a aplicação e a consola;
- **GetIDP() –** Método que serve apenas para adquirir o valor que identifica o piso que foi selecionado pelo utilizador.
- **GetIDD() –** Obtém a identificação da divisão que se encontra selecionada na aplicação;
- **GetIDZ() –** Semelhante aos anteriores só que para o caso das zonas;

- **GetIDCEN –** Método que pede à base de dados a identificação duma determinada macro;
- **GetLog() –** Último dos métodos de recolha de informação, sendo que para o este caso, recolhe-se toda a informação referente à atividade da aplicação para que posteriormente seja colocada à disposição do utilizador;
- **DeleteMacro() –** Permite a remoção da base de dados de toda a informação referente à macro eliminada;
- **ReplaceMacroContent() –** Método que permite substituir informação sobre os valores de uma dada macro;
- **MacroNameValidation() –** Valida a existência dum dado nome introduzido pelo utilizador na criação de uma macro na base de dados;

4.3.3 Módulo Comunicação

O módulo de comunicação utiliza uma libraria adicional para o sistema operativo Android chamada Jamod, que implementa o protocolo ModBus (Charlton & Wimberger, 2010). Esta libraria já contém as funções referidas na secção deste protocolo no capítulo 3, sendo apenas necessários alguns procedimentos para acondicionamento da informação a enviar e a receber.

Tendo em conta os conteúdos referidos no capítulo 3 sobre a função e funcionamento da classe implementada, *AsyncTask*, de seguida na Figura 4-9 tem-se a sua constituição.

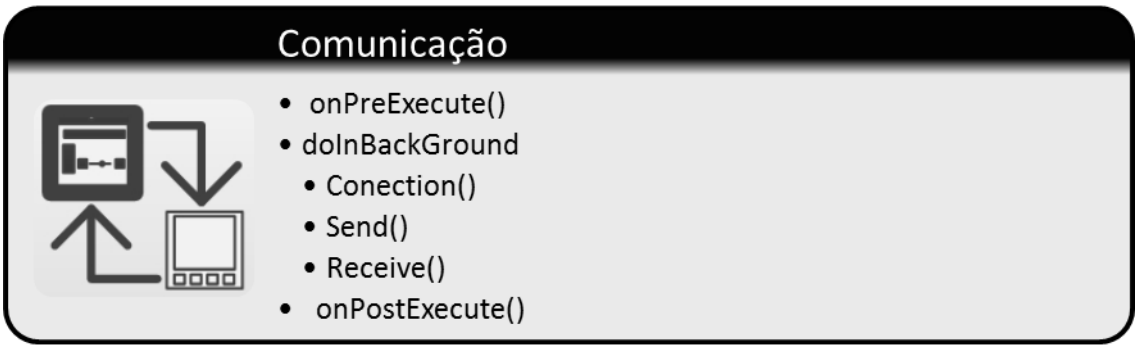


Figura 4-9: Constituição da classe Comunicação.

Relativamente aos métodos apresentados, as suas funções são:

- **Connection() –** Método responsável por criar e manter a ligação entre a aplicação e a consola para a divisão selecionada previamente pelo utilizado;
- **Send() –** Método que emprega a função *Write Single Register* , que tem como parâmetro de entrada o código proveniente do tipo de comando a alterar, tendo em conta se é iluminação, estores, temperatura ou som. Terminado este processo de transferência entre aplicação e consola, a função referida retorna o conteúdo do registo previamente enviado no processo de transferência, de modo a que ambos os valores possam ser comparados e poder-se verificar se houve corrupção de dados.

- **Receive() –** Método mais simples que o anterior que apesar de implementar a função *Read/Write Multiple registers*, apenas tem como objetivo aceder ao registo correspondente do código recebido da aplicação e ler o seu conteúdo, transferindo essa informação para a aplicação para ser colocada à disposição do utilizador.

4.3.4 Módulo Interface Gráfica

Para a construção dos diferentes *layouts*, e tendo em conta o tipo de comando selecionado, o módulo de interface gráfica vai selecionar do conjunto de *views* já apresentados no capítulo 3, as *views* necessárias para que o utilizador exerça o seu controlo sobre o ambiente residencial. É de referir que os *layouts* refletem as opções programadas nas diferentes consolas.

De seguida apresentam-se os métodos que constituem a classe inserida neste módulo (ver Figura 4-10) que faz a gestão de todas as *views* dinâmicas da aplicação.

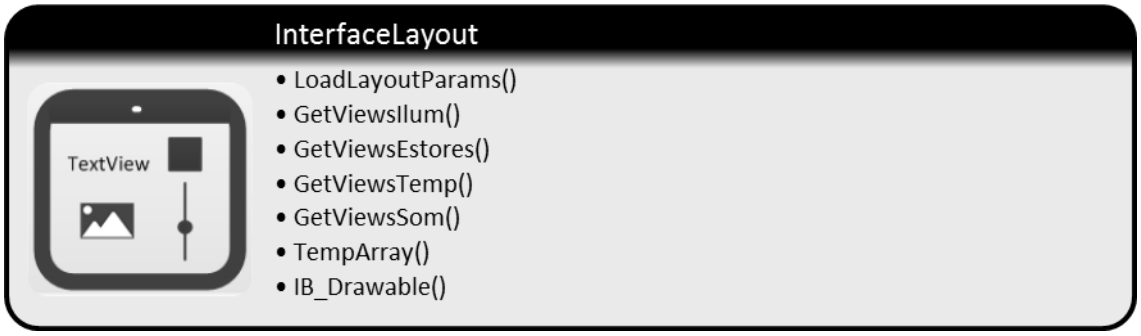


Figura 4-10: Constituição da classe InterfaceLayout.

Sobre os métodos apresentados na figura anterior, as suas funções são:

- **LoadLayoutParams() –** Método que declara na classe o tamanho das *views* a utilizar, ou seja, permite definir o seu comprimento e largura atendendo ao seu conteúdo. Tendo em conta a função de cada *view* e o seu tipo, esta possuirá parâmetros específicos.
- **GetViewsIlum() –** Primeiro de quatro métodos responsáveis por desenharem as *views* no layout definido. O seu funcionamento passa por configurar os diversos *widgets* na interface gráfica para que o utilizador possa efetuar o controlo da iluminação.
- **GetViewsEstores() –** Exatamente o mesmo procedimento que o método anterior, sendo que para este método, a restrição imposta será o tipo de *widgets* utilizados pois o tipo de comando seleciona é os estores.
- **GetViewsTemp() –** Segue o mesma principio de funcionamento que os anteriores mas para o caso da temperatura.
- **GetViewsSom() –** O mesmo comportamento só que para o caso do som.

- **TempArray() –** Método que gera um *array* com os valores para o intervalo de temperatura, que é definido aquando da instalação do sistema de automação residencial. Este *array* é posteriormente utilizado pelos *widgets* que necessitarem, nomeadamente o *NumberPicker*.
- **IB_Drawable() –** Método que implementa uma configuração adicional para *views* do tipo *ImageButton*. Tal configuração permite combinar o fundo que pode ser uma imagem com os diferentes tipos de estado de seleção de um *ImageButton*, o que não é nativamente implementado. Tal combinação permite ao utilizador saber se se encontra a pressionar um determinado botão.

4.3.5 Módulo Configurações

O módulo de configurações, ao contrário dos anteriores que se encontram interligados com o módulo de controlo habitacional, é acionado através do *layout* do anterior mas corre em paralelo com o módulo central. Qualquer alteração efetuada nos sub-módulos que interajam com o módulo de controlo habitacional, como criação ou remoção de macros, são reportadas a este módulo para que a informação seja atualizada quando o utilizador retorne à interface deste.

4.3.5.1 Sub-Módulo Macros

A configuração de macros é considerada uma das funcionalidades essenciais no que à automação residencial diz respeito como foi mencionado no capítulo 2. Usualmente todos os sistemas comercializados possuem esta funcionalidade, permitindo aumentar assim o conforto a autonomia da habitação.

No que diz respeito à sua função, esta estará relacionada com a criação, edição e possível remoção de macros. Relativamente às macros, estas como seriam de esperar não se encontram restritas a uma única divisão ao comandos, podendo ser constituídas por múltiplas divisões e comandos.

Quanto à sua constituição, na Figura 4-11 encontram-se listados os métodos que compõem a atividade *Macros*.

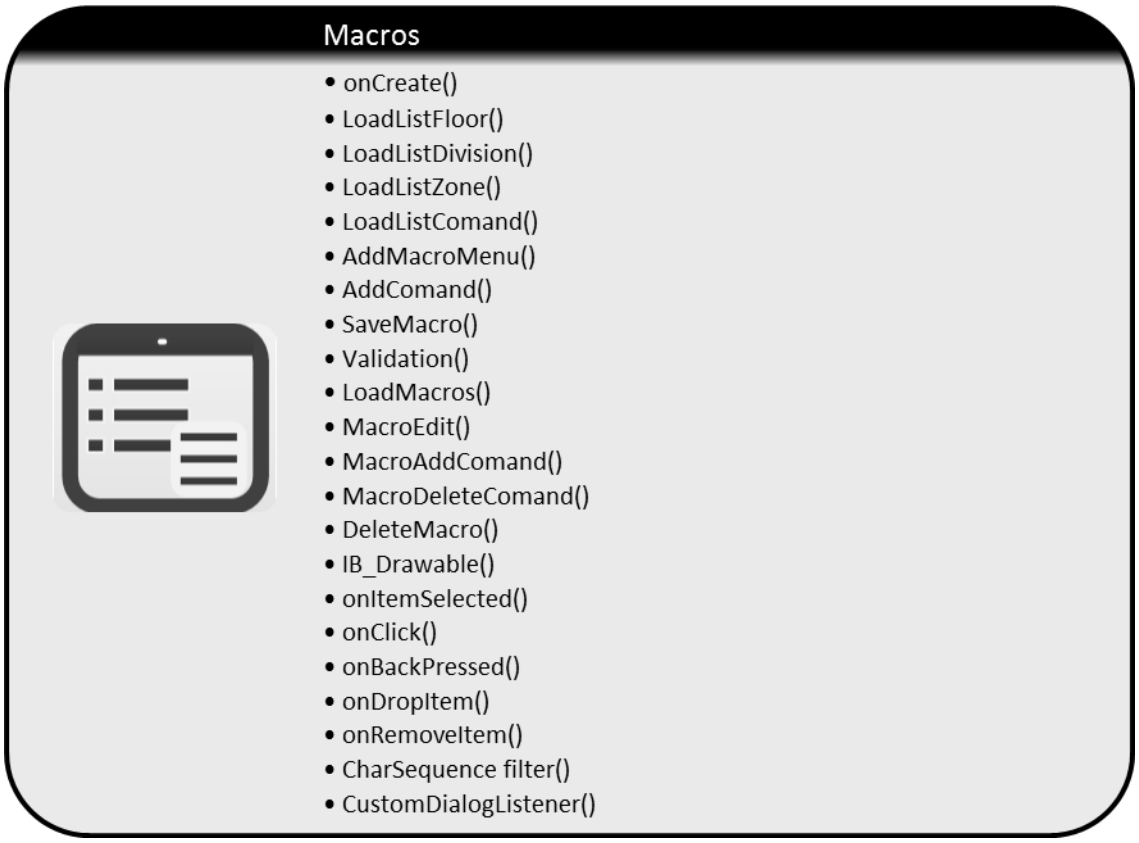


Figura 4-11: Constituição da classe Macros.

Sobre os métodos referenciados na Figura 4-11, é de salientar que os métodos onCreate(), LoadListFloor(), LoadListDivision(), LoadListZone(), LoadListComand(), onItemSelected(), onClick() e CustomDialogListener() possuem as mesmas funções apresentadas no módulo de controlo habitacional. Quanto aos restantes métodos presentes na Figura 4-11, estes possuem as seguintes funções:

- **AddMacroMenu() –** Método utilizado para iniciar o layout onde se pode configurar os comandos a adicionar numa macro e posteriormente fazer a sua criação;
- **AddComand() –** Permite armazenar vários comandos, para o caso de o utilizador pretender construir uma macro que incluía várias divisões ou comandos. A adição de comandos terá um limite de 25 comandos por macro para preservar a rapidez de execução quando solicitadas;
- **SaveMacro() –** Como o seu nome indica, é o método responsável pela criação da macro, onde é pedido que o utilizador insira o nome da macro, para que posteriormente toda a informação relativa à mesma seja armazenada na base de dados da aplicação;
- **Validation() –** Método desencadeado durante a execução do anterior e que vai verificar se existe ou não o nome na base de dados. Caso exista, é pedido ao utilizador que insira um novo nome;

- **LoadMacros() –** Utilizado para carregar o menu de macros existentes, possibilitando ao utilizador além de saber quais as macros já existentes, proceder a alterações das mesmas como adição de comandos, edição ou remoção de determinada macro;
- **MacroEdit() –** Como o seu nome indica, vai abrir o menu de edição onde o utilizador pode alterar a ordem ou remover algum comando da macro selecionada;
- **MacroAddComand() –** Método que reabre o menu de configuração das macros e que permite ao utilizador adicionar mais comandos a uma macro existente, caso não supere o valor máximo de comandos;
- **MacroDeleteComand() –** Método consequente do onRemoveItem() que é chamado caso o utilizador pretenda remover um ou mais comandos da macro selecionada;
- **DeleteMacro() –** Executado quando selecionada a opção de apagar macro, apagando toda a informação da macro escolhida da base de dados;
- **IB_Drawable() –** Possui a mesma função que o método com o mesmo nome no módulo de interface gráfica;
- **onBackPressed() –** Método que neste módulo tem como função retornar ao módulo de controlo habitacional e não sair da aplicação;
- **onDropItem() –** Um dos dois métodos de escuta implementados pelo sub-módulo *Drop&SortListView* que será apresentado no decorrer deste capítulo. Este método tem como função retornar a nova posição se algum comando for arrastado da sua posição inicial;
- **onRemoveItem() –** Segundo método implementado pelo *Drop&SortListView*, que tem como função remover o comando da lista de comandos duma dada macro;
- **CharSequence filter() –** Método que regista o número de caracteres quando o utilizador introduz o nome aquando da criação de uma macro. A sua utilização visa limitar o número de caracteres a serem introduzidos;

4.3.5.2 Sub-Módulo Configurações Básicas

O sub-módulo "Configurações Básicas" tem como objetivo fornecer ao utilizador a possibilidade de este alterar as designações bem como os ícones associados aos espaços físicos que sejam previamente inseridos na aplicação por parte da entidade promotora ou utilizador aquando da instalação do sistema "HomeDom".

As designações que o utilizador insira neste sub-módulo são automaticamente atualizadas na base de dados e posteriormente enviadas para a respetiva consola, mantendo sempre neste caso a informação atualizada nos dois pontos de acesso do sistema.

Sobre a sua constituição, mais uma vez na Figura 4-12 estão listados os métodos que compõem a atividade *BasicsConfig*.

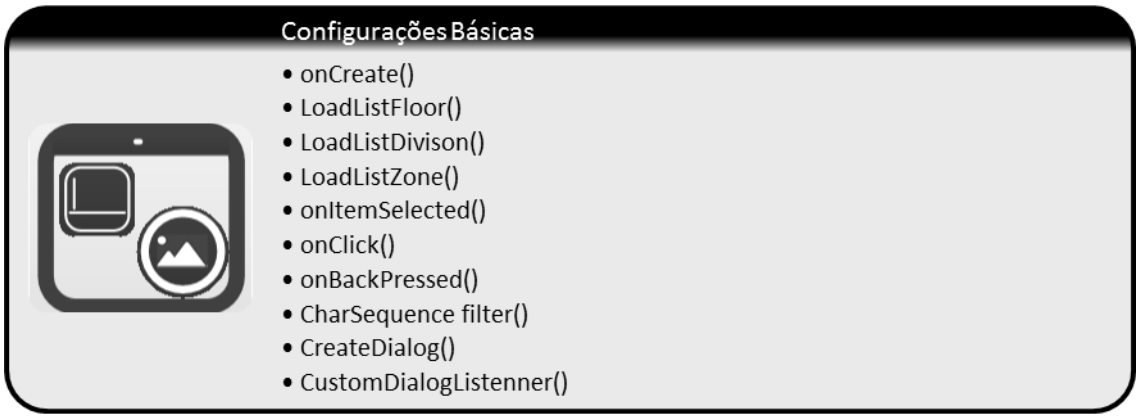


Figura 4-12: Constituição da atividade *BasicsConfig.*

À semelhança do que aconteceu no sub-módulo anterior, na Figura 4-12 encontram-se métodos cujas funções já foram descritas no módulo Controlo Habitacional ou no sub-módulo Macros, sendo que para o caso do "CharSequence filter()", este vai limitar a 20 carateres a introdução de um novo nome para os pisos, divisões ou zonas.

Sobre o método "CreateDialog()" e como o seu nome indica, este método visa a criação dos interfaces *AlertDialog*, uma vez que neste sub-módulo podem ser feitas alterações ao nível das designações e ícones dos pisos, divisões e zonas.

4.3.5.3 Sub-Módulo Registo Atividade

Sub-módulo que implementa o registo de atividade ou *log* como é usualmente conhecido numa aplicação, e que tem como objetivo registar todas as comunicações efetuadas entre componente lógica e física do sistema.

Esse registo permite que em caso de mau funcionamento ou falhas de comunicação entre componentes, esses eventos fiquem registados na base de dados, de modo a que se o problema persistir, o fornecedor do sistema consiga resolver o problema num curto espaço de tempo.

Relativamente ao funcionamento do "Registo Atividade", tratando-se apenas dum sub-módulo de recolha de informação, a atividade que o integra, *Log*, só possui o método *getLog()*. A função deste método é obter da base de dados toda informação da tabela *Log*, onde ficam registadas todas as operações de transferência de informação entre as duas partes do sistema

4.3.5.4 Sub-Módulo Configurações Avançadas

O módulo de configurações avançadas fornece as opções consideradas de risco que um utilizador usual não deverá utilizar com regularidade. É de salientar que estas opções

disponibilizadas quando mal utilizadas, podem comprometer o bom funcionamento da aplicação.

Uma das funções deste sub-módulo passa por carregar a base de dados numa primeira utilização da aplicação com a informação sobre a constituição do sistema físico. Posteriormente, esta função pode ser utilizada para carregar as configurações de macros para o caso de a aplicação se encontrar instalada em dois dispositivos.

Tratando-se de um sub-módulo que possui opções de risco, este é protegido por via de inserção de uma credencial válida para ser acedido, visando limitar que seja utilizado por algum utilizador que não se encontre familiarizado com as opções inseridas.

De seguida, encontram-se descritos na Figura 4-13 os métodos que constituem este sub-módulo.

Figura 4-13: Constituição da classe *AdvancedConfig.*

Como se pode visualizar na Figura 4-13, a classe "AdvancedConfig" possui os métodos onCreate(), onClick(), onItemSelected(), onBackPressed() e CustomDialogListener(), cujas funções já foram mencionadas anteriormente. Sobre os restantes métodos referidos na Figura 4-13, as suas funções são:

- **ExportDialog() –** Método que cria um AlertDialog(), permitindo ao utilizador escolher que tipo de informação a extrair da base de dados, isto é, se pretende extrair a informação relativa à estrutura do sistema ou relativa às macros e seus conteúdos;
- **ExportBD() –** Invocado no seguimento do anterior e que vai extrair a informação utilizando o sub-módulo XML que será descrito no decorrer deste capítulo.
- **ImportDialog() –** Possui a mesma função que o ExportDialog(), mas com o intuito de informar o utilizador dos passos subsequentes da importação da base de dados.
- **ImportBD() –** Método que é chamado pelo anterior caso o utilizador aceite as condições de importação expostas pelo método anterior. Também este método utiliza

o sub-módulo XML para fazer a importação, podendo importar a informação sobre a constituição do sistema como a informação sobre as macros, caso exista.

- **LoadSpinners() –** Método invocado para proceder ao carregamento dos *spinners* que permitem a escolha do piso,divisão,zona e comando para a nova zona a adicionar.
- **AddZone() –** Recolhe a informação proveniente do método "onItemSelected()" e a designação da nova zona e introduz essa informação na base de dados.

4.3.6 Módulo Personalização

Há semelhança do que a aconteceu no módulo "Configurações", também este módulo vai estar dividido em sub-módulos como se pôde visualizar na Figura 4-6, consoante o tipo de componente criado e a função que este vai desempenhar.

4.3.6.1 Sub-Módulo Vertical SeekBar

Sub-módulo que implementa a versão vertical da *seekbar*. Optou-se pela implementação deste tipo de *widget* em detrimento de um atributo da *seekbar* que permite a rotação da mesma, mas com a particularidade de o parâmetro relativo ao seu comprimento após a rotação manter-se. Do ponto de vista de ocupação deste *widget* no *layout*, esta rotação fazia com o que a *seekbar* ocupasse espaço desnecessário tendo em conta a sua nova orientação.

Para criar este *widget* recorreu-se a uma classe abstrata que estende a classe *SeekBar*, isto é, vai possuir a herança desta última possuindo assim todas as características e atributos que a *seekbar* possui.

Assim, só resta alterar neste caso os parâmetros que permitem fazer a rotação da *seekbar* da horizontal para a vertical que se encontra na função representada na Figura 4-14. Este processo é efetuado no momento da sua criação e não como atributo posteriormente, permitindo que com esta ação, o *widget* criado ocupe unicamente o espaço que precise.

```
protected void onDraw(Canvas c) {
    c.rotate(-90);
    c.translate(-getHeight(), 0);

    super.onDraw(c);
}
```

Figura 4-14: Método *OnDraw()* que auxilia a criação da VerticalSeekBar.

No método da Figura 4-14, pode-se verificar que se efetua uma rotação de 90º para se por a *seekbar* na posição vertical e que a variável de espaço que vai ser modificada é a altura e não o comprimento que acontece no caso da *seekbar* nativa.

No que diz respeito aos restantes atributos, como a deteção da variação bem como o seu valor, este *widget* mantém o seu funcionamento, uma vez que como já foi mencionado herdou as características da *seekbar*, ou seja, tanto os seus atributos como os métodos que essa classe implementa.

Figura 4-15: Interface gráfica da VerticalSeekBar.

4.3.6.2 Sub-Módulo Adapter

Face ao facto de o sistema operativo Android não possuir na constituição dos seus *adapters* nativos a opção de inserção de imagens, foi necessário personalizar um *adapter* que permite além do tradicional texto, colocar uma imagem.

Do mesmo modo que foi feito para a *VerticalSeekbar*, também para este caso recorreu-se a uma classe abstrata do Android que estende a classe ArrayAdapter<>, herdando neste caso especifico, os atributos e métodos dum tipo de adapter já existente.

Após o processo anterior, falta apenas editar o método getView(), representado na Figura 4-16, que permite obter ou criar uma view com os requisitos que se pretendem que é o caso de texto e imagem. Para tal é necessário criar um *layout* que incorpore uma *TextView* e *ImageView* para que depois com recurso ao método referido se possa adicionar o texto e as imagens que se pretende

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    View row = convertView;
    ItemHolder holder = null;

    if(row == null)
    {
        LayoutInflater inflater = ((Activity)context).getLayoutInflater();
        row = inflater.inflate(layoutResourceId, parent, false);

        holder = new ItemHolder();
        holder.imgIcon = (ImageView)row.findViewById(R.id.icon);
        holder.txtTitle = (TextView)row.findViewById(R.id.label);

        row.setTag(holder);
    }
    else
    {
        holder = (ItemHolder)row.getTag();
    }

    Item item= data[position];
    holder.txtTitle.setText(item.text);
    holder.imgIcon.setBackgroundResource(item.icon);

    return row;
}
```

Figura 4-16: Método *getView()* que auxilia a criação do adapter personalizado.

4.3.6.3 Sub-Módulo Navigation Drawer

Sub-módulo que implementa o Navigation Drawer (Android Developers, 2015), componente nativo do sistema Android que permite à aplicação ter um menu de navegação quando pressionado o ícone referente à aplicação (ver Figura 4-17).

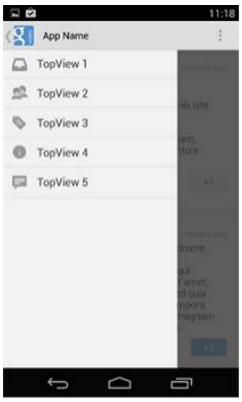


Figura 4-17: Interface gráfica genérica do Navigation Drawer, adaptado de (Android Developers, 2015).

Apesar de ser um componente nativo, foi feita a sua personalização uma vez que para a aplicação desenvolvida, foi personalizado o seu formato de apresentação e a posição de abertura devido às características da aplicação. Quanto à alteração da posição de abertura, foi necessário introduzir o código que se encontra representado na Figura 4-18 na classe que estende o tipo *activity*. Com esta extensão pretende-se assegurar que todos os métodos e

atributos nativos da *activity* possam ser utilizados, uma vez que esta classe é implementada nos diferentes módulos ou sub-módulos da aplicação que são constituídos pela classe *activity*.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int teste = item.getItemId();
    if (item != null && item.getItemId() == 0) {
        if (mDrawerLayout.isDrawerOpen(Gravity.RIGHT)) {
            mDrawerLayout.closeDrawer(Gravity.RIGHT);
        } else {
            mDrawerLayout.openDrawer(Gravity.RIGHT);
        }
    }
    return false;
}
```

Figura 4-18: Método que personaliza abertura do *Navigation Drawer* do lado esquerdo.

Através da Figura 4-18, pode-se verificar que o parâmetro responsável por esta alteração é *Gravity.RIGHT*, que tem como função colocar o componente na posição mais à direita da interface.

Relativamente à sua função, este componente disponibiliza ao utilizador a seleção dos sub-módulos do módulo de configuração, isto é, as opções de escolha inseridas neste menu serão os sub-módulos de macros, configurações básicas, registo de atividade e por fim as configurações avançadas.

4.3.6.4 Sub-Módulo XML

O sub-módulo "XML" implementa a classe responsável que permite à aplicação importar e exportar informação de um ou para um ficheiro do tipo XML. Para efetuar tais processos, esta classe vai implementar métodos como OpenDoc(), ImportBD e ExportBD, que possuem as funções e as variáveis nativas do Android para leitura e escrita de documentos XML.

Sobre o método OpenDoc(), este implementa funções e variáveis do tipo *DOM Parser* e *I/O* (entrada/saída), que permitem a abertura de um ficheiro XML e a leitura do seu conteúdo se o ficheiro existir.

No método ImportBD() são implementadas variáveis do tipo *NodeList* e *xPath* que permitem obter a informação dos diferentes nós que constituem um documento XML. A informação recolhida é posteriormente organizada pela aplicação tendo em conta a constituição da base de dados.

Em relação ao método ExportBD(), este invoca variáveis do tipo *XmlSerializer* e *IO*, onde as variáveis *IO* permitem criar um ficheiro do tipo XML para onde vai ser exportada a informação da base de dados. Posteriormente, o conteúdo deste ficheiro vai ser adicionado com o recurso ao *XmlSerializer* que permite criar os diferentes nós que constituem o XML bem como o adicionar do seu conteúdo.

Por fim, é de referir que este sub-módulo é invocado pelo sub-módulo Configurações Avançadas, onde são implementadas as opções de importação e exportação da base de dados por parte da aplicação.

4.3.6.5 Drag&SortListView

O sub-módulo "Drag&SortListView" implementa um tipo de *ListView* que não é nativa no sistema operativo Android. Este visa dar resposta as opções de edição de macros fornecidas no sub-módulo "Macros" como a alteração da posição dos comandos por via do arrastamento do mesmo. Outra das opções fornecidas por este sub-módulo é a eliminação de comandos da lista existente.

Sendo um sub-módulo personalizado, também se encontram implementados os métodos para lidar com alteração posição e remoção do comando, sendo os métodos utilizados o onDropItem() e onRemoveItem(), cujo funcionamento já foi mencionado no sub-módulo Macros. Na Figura 4-19 encontra-se a interface gráfica deste sub-módulo para o utilizador.

Figura 4-19: Interface gráfica da Drop&Sort Listview.

Como se pode visualizar na Figura 4-19, a opção para alterar a posição do comando encontra-se à esquerda do texto, enquanto a opção de remoção do comando se encontra à direita do mesmo, evitando assim possíveis enganos na seleção. De mencionar que pressionando o texto, não vai ser executada nenhuma ação.

5. Aplicação “ HomeTouch”

Uma vez apresentada a arquitetura da componente lógica (aplicação), é necessário converter toda a sua programação para um resultado gráfico para que utilizador final da aplicação consiga interagir coma a mesma.

Neste capítulo é apresentada a aplicação do ponto de vista da interface gráfica para o utilizador. A interface faz uso dos diversos módulos ou sub-módulos apresentados no capítulo 4, tendo em conta o seu funcionamento e as opções disponibilizadas ao utilizador.

Relativamente à sua estrutura, este capítulo é constituído por várias secções, onde cada uma dessas subsecções apresenta uma interface gráfica e um possível caso de uso. Esse caso de uso reflete os passos necessários para que o utilizador consiga executar determinada ação como por exemplo aumentar o valor da luminosidade.

Na secção 5.1 é apresentada a interface de entrada, ou ecrã de boas vindas. Na secção seguinte é exibida a interface relativa ao módulo de controlo habitacional e também apresentado um caso de uso à semelhança do caso anterior. A secção 5.3 apresenta a interface e caso de uso para o módulo de configurações de macros. Por fim, a ultima secção deste capítulo expõe o mesmo processoque os antrriores para o caso das configurações avançadas.

5.1 Interface Entrada

O ecrã de boas vindas é a primeira interface da aplicação que o utilizador irá presencia. Nesta são apresentados alguns detalhes genéricos sobre a aplicação (nome e versão da aplicação, logotipo da empresa) como se pode vizualizar na Figura 5-1.

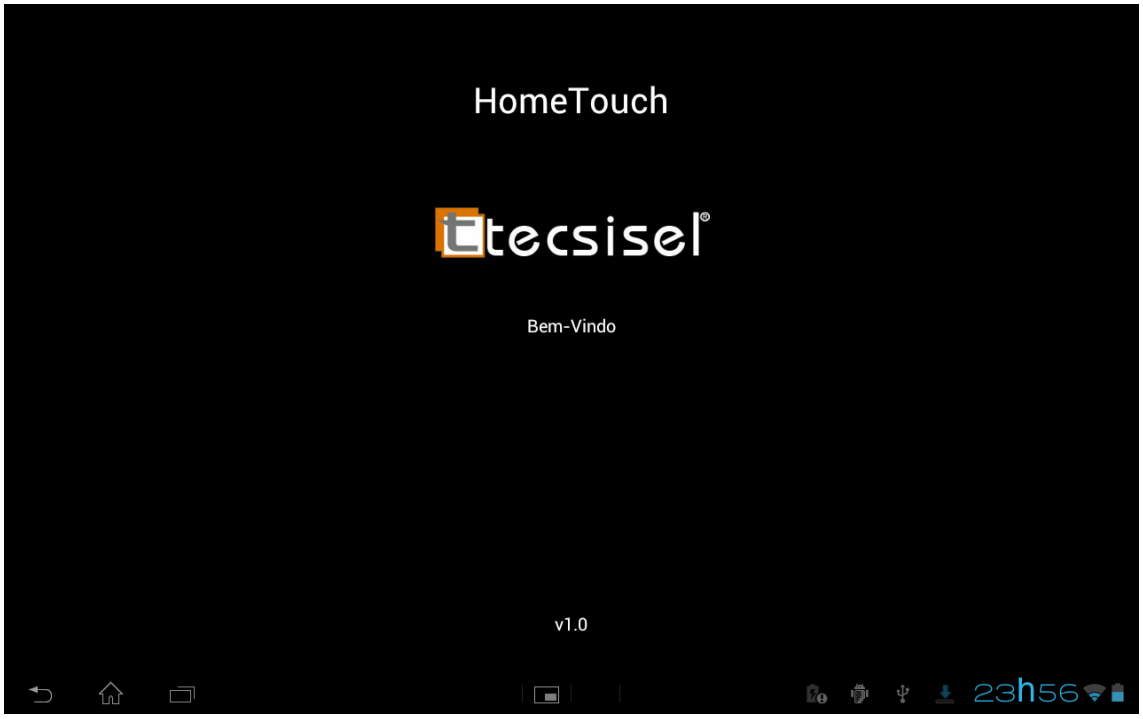


Figura 5-1: Ecrã de entrada da aplicação “ TecsiselApp“.

Após este ecrã, o utilizador entra na interface gráfica do módulo de controlo habitacional sem que necessite de executar qualquer ação adicional.

5.2 Interface de controlo do sistema

A interface de controlo do sistema é responsável pelo controlo do sistema propiamente dito. Neste ecrã pode ser efetuada a alteração das diversas variáveis de comando, isto é, a variação dos valores da luminosidade, abertura de estores, temperatura e volume de som. Na Figura 5-2, encontra-se o diagrama de funcionamento indicando os passos necessários para se proceder à variação das variáveis mencionadas.

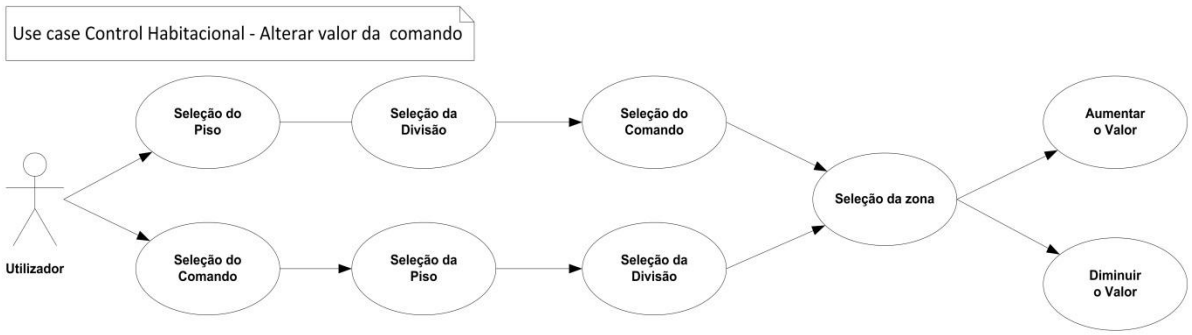


Figura 5-2: Diagrama de funcionamento do módulo Controlo Habitacional.

Como se pode observar no diagrama da Figura 5-2, existem dois caminhos possíveis possibilidades para executar a mesma ação. O utilizador pode começar por selecionar o piso, piso seguida da escolha da divisão e posterior opção do comando a alterar (iluminação, estores, temperatura ou som). Outra possibilidade seria escolher primeiro o comando a alterar, seguido da seleção do piso e divisão e por fim a escolha da zona.

Na Figura 5-3 encontra-se representado a interface com os diferentes passos que o utilizador precisa necessita de para alterar neste caso o valor da iluminação.

Figura 5-3: Interface gráfica para o módulo de controlo habitacional.

Relativamente à Figura 5-3, é importante salientar que, aquando da seleção da zona, a aplicação pede à consola respetiva que lhe envie o valor atual do parâmetro. Esta operação visa permitir ao utilizador ter conhecimento do valor atual, evitando discrepâncias entre o valor apresentado pela aplicação pela consola. Neste caso específico, o valor que a consola enviou a pedido da aplicação foi 80%, isto é, a iluminação da zona selecionada encontra-se a 80% da sua luminosidade máxima. Este valor pode ser confirmado na Figura 5-4, onde se encontra o ecrã da consola de comando que exibe todos os valores das diferentes zonas de iluminação para a sala do R/C.

Figura 5-4: Consola de comando da divisão “Sala”.

Ainda na Figura 5-3, no último passo, procedeu-se a uma alteração da luminosidade da zona “Iluminação Zona 1” para o valor de 15% na aplicação. Esta alteração vai desencadear uma nova operação de comunicação entre as componentes, sendo que neste caso vai ser do tipo de escrita. Este processo vai informar a consola que tem que atualizar o valor para a zona selecionada na aplicação para o valor introduzido pelo utilizador, processo este que pode ser visualizado na figura seguinte (ver Figura 5-5).

Figura 5-5: Consola de comando da divisão “Sala” após alteração do valor de luminosidade na aplicação.

O ecrã do controlo do sistema também permite o acesso às restantes funcionalidades da aplicação, como a configurações de macros ou as configurações avançadas, basta ao utilizador pressionar o ícone  que se encontra no canto superior direito da interface. Pressionando este ícone, é disponibilizado o menu de configuração da aplicação onde se inserem as configurações mencionadas anteriormente e as restantes abordadas no capítulo 4 (ver Figura 5-6).

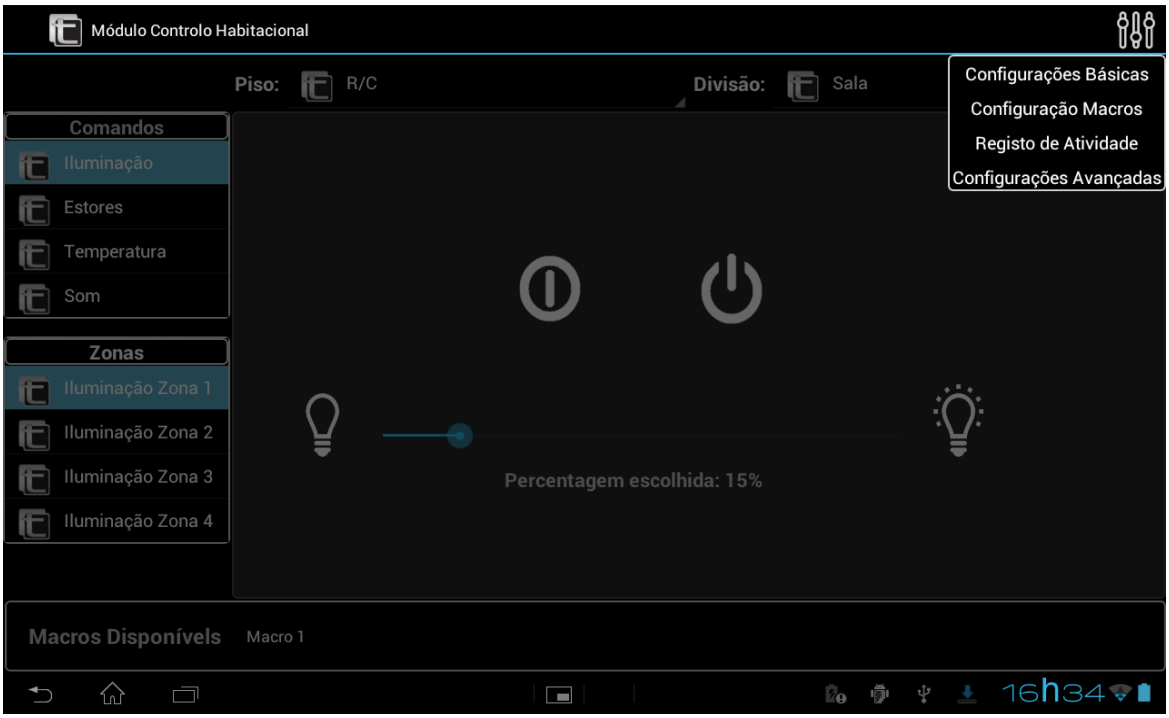


Figura 5-6: Menu de configurações da aplicação “HomeTouch“.

De referir que o menu de configurações está disponível nas diversas interfaces da aplicação, ou seja, também encontra-se disponível em todos os menus configuração. A sua presença permite ao utilizador aceder aos vários menus sem que necessite retornar à interface de controlo do sistema, sendo que se o utilizador pressionar o botão *Back*, , todos os menus de configurações retornam ao módulo de controlo habitacional.

Por fim, este módulo ainda está configurado para atualizar periodicamente (10 em 10 minutos) o valor do parâmetro que se encontra selecionado. Este procedimento possibilita que em caso de alteração manual do valor na consola de comando, a aplicação possua sempre o valor atualizado, não existindo a necessidade de selecionar novamente a zona para que o valor do parâmetro seja atualizado.

Selecionando, no menu de configurações, as “Configurações Macros”, uma das funcionalidades essenciais no que diz respeito à automação residencial, acede-se ao sub-módulo Macros.

5.3 Configurações Macros

No menu de configurações de macros, e como já foi mencionado no capítulo 4, são disponibilizadas ao utilizador várias opções que passam pela criação, edição e eliminação de macros. A Figura 5-7 apresenta o caso de uso que reflete o funcionamento dos processos já referidos.

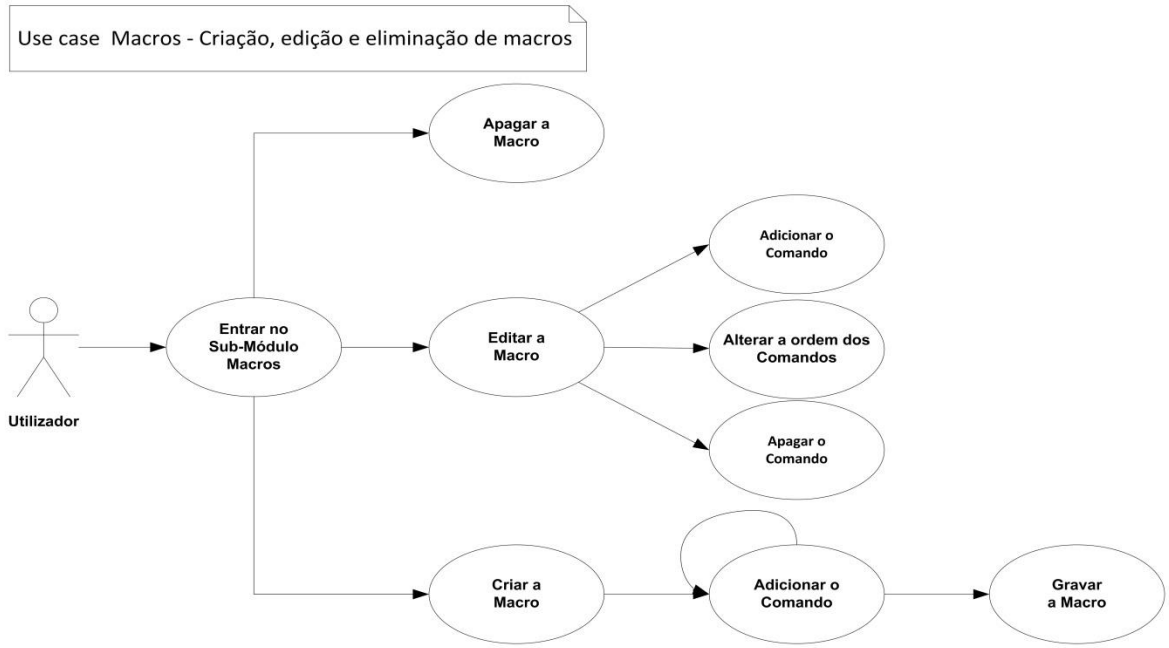


Figura 5-7: Diagrama de funcionamento do sub-módulo Macros.

Relativamente ao diagrama da Figura 5-7, este encontra-se dividido em 3 opções, tendo em conta a existência ou não de alguma macro configurada na aplicação. Caso não se verifique a sua existência, este menu apenas oferece a opção para criar uma macro. Selecionando essa opção é permitido ao utilizador escolher os comandos que pretende adicionar à macro, e como já foi referido no capítulo 4, os comandos não se encontram restringidos a uma divisão ou tipo de comando. Após a inserção de um comando é finalmente dada ao utilizador a opção para gravar a informação da macro, onde este só precisa de introduzir um nome para identificar a macro a gravar. No processo de gravação da macro, a aplicação vai validar o nome introduzido pelo utilizador, uma vez que não são permitidas macros com a mesma designação, evitando deste modo que macros com o mesmo nome desempenhem funções diferentes. Para o caso de o nome introduzido pelo utilizador se encontrar em utilização, a aplicação pede novamente a este que reintroduza um novo nome, passando novamente por um novo processo de validação. Este procedimento pode ser observado na Figura 5-8 onde está apresentada a opção de criação de uma macro, com todos os passos descritos.

Figura 5-8: Interface gráfico para sub-módulo Macros (criação macro).

Caso se verifique a existência de uma macro, para além da opção para criação de uma nova macro, o utilizador tem acesso às opções de edição e eliminação da macro que selecionar. Quanto à edição, esta opção oferece ao utilizador a possibilidade de alterar a ordem de algum comando existente na macro, apagar determinado comando do conteúdo da mesma e ainda possibilita adicionar mais comandos à macro. Estas operações só são permitidas caso o utilizador não tenha atingido o número máximo de comandos permitidos por macro (25

comandos). Quanto à opção da eliminação da macro, permite ao utilizador remover todo o conteúdo de determinada macro da aplicação.

No que diz respeito às interfaces gráficas, a Figura 5-9 apresenta a opção de edição de uma macro, mais concretamente a alteração da ordem dos comandos ou eliminação dos mesmos.

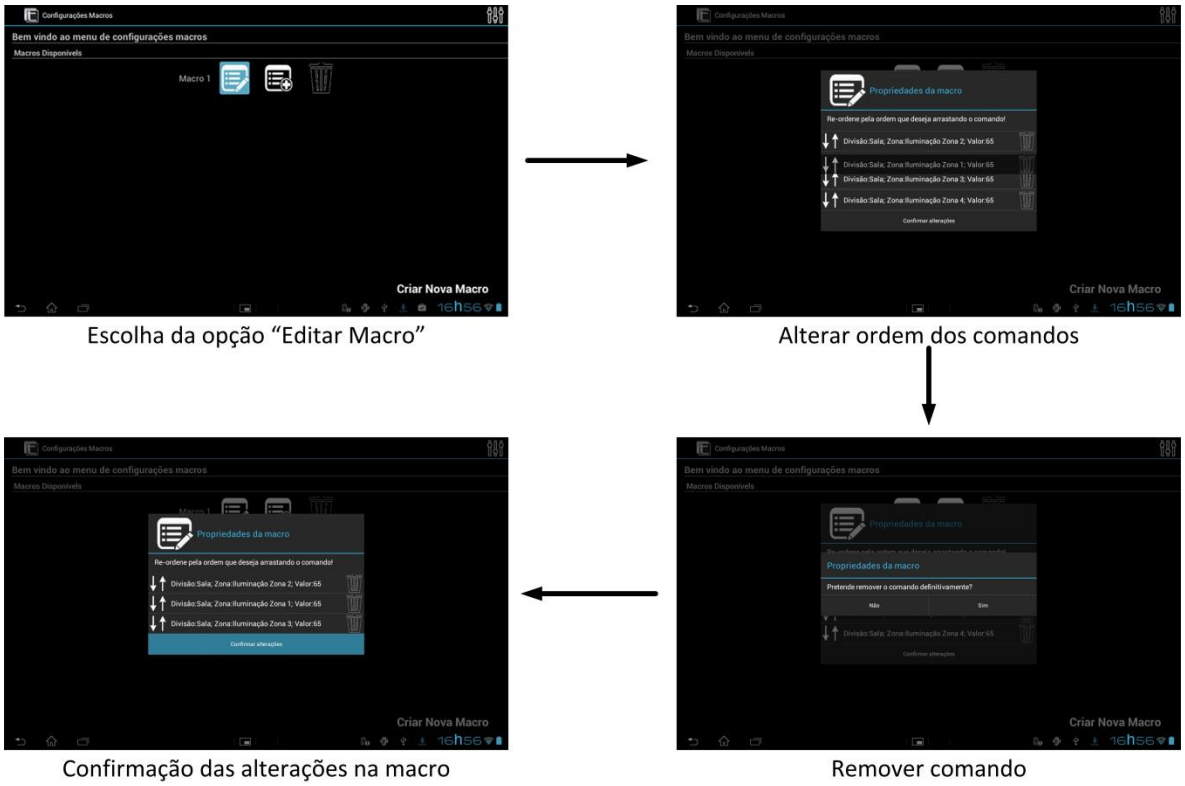


Figura 5-9: Interface gráfico para sub-módulo Macros (edição macro).

É de referir que para efetuar as operações de arrasto e de eliminação dos comandos, é necessário que, no primeiro caso, o utilizador pressione o ícone do lado esquerdo enquanto no segundo caso, pressione do lado direito.

Por outro lado, na Figura 5-10 encontra-se a opção de adição de mais comandos ao conteúdo de uma macro já existente.

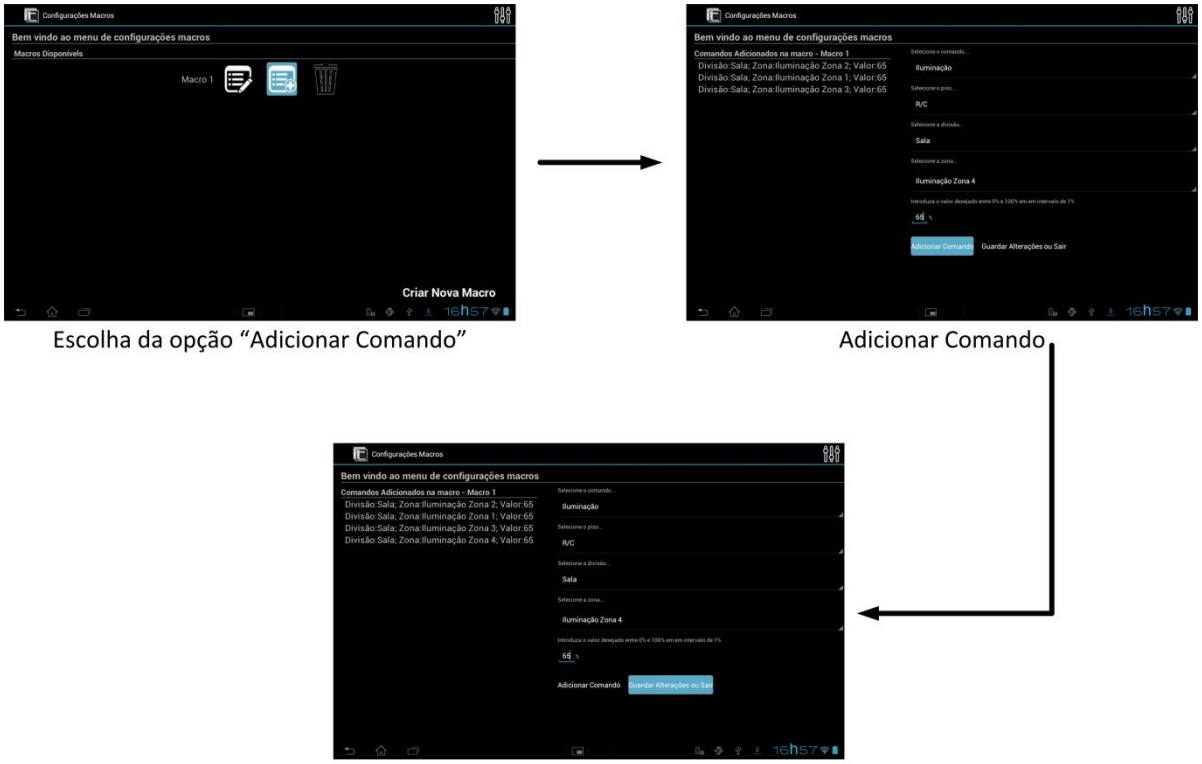


Figura 5-10: Interface gráfico para sub-módulo Macros (edição macro).

5.4 Configurações Avançadas

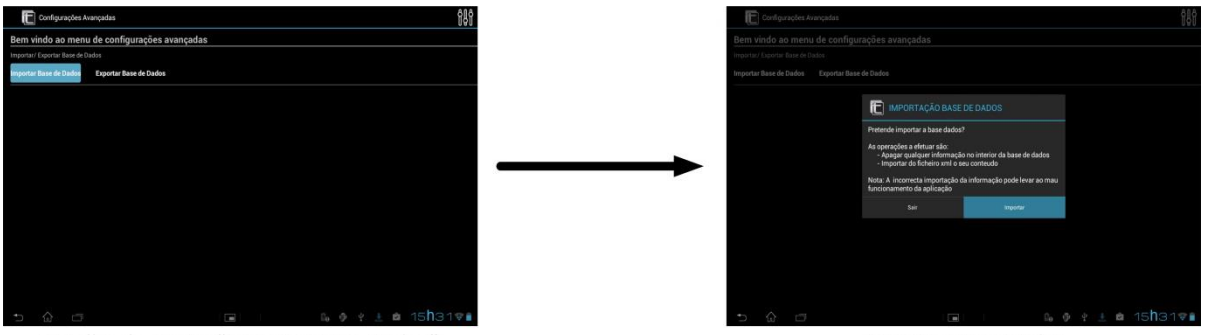
No sub-menu “Configurações Avançadas”, o utilizador tem acesso às opções de importação e exportação da base dados, ou seja, este poderá carregar a base de dados com a informação necessária recorrendo à opção de importação. Para o caso de o utilizador adquirir outro dispositivo móvel, a aplicação dispõe da opção de exportação que descarrega a informação da base de dados para um ficheiro do tipo XML, que depois pode ser utilizado para carregar a base de dados num segundo dispositivo móvel.

Relativamente ao funcionamento deste sub-módulo, na Figura 5-11 está representado o diagrama com as opções disponibilizadas ao utilizador.

Figura 5-11: Diagrama de funcionamento do sub-módulo Configurações Avançadas.

Como se pode visualizar no diagrama da Figura 5-11, sobre a importação da base de dados, a aplicação executa de forma autónoma todo o processo, enquanto no processo de exportação da informação, o utilizador tem que indicar que informação pretende exportar. Quanto à informação que se pode exportar, esta é dividida em dois tipos, a informação geral onde se encontra toda a informação referente à constituição do sistema físico e a informação das macros, onde esta incorporada toda a informação sobre as macros existentes na aplicação.

No que diz respeito às interfaces gráficas, a Figura 5-12 apresenta os passos para efetuar o processo de importação



Escolha da opção “Importar Base Dados” Importar Base Dados

Figura 5-12: Interface gráfico para sub-módulo Configurações Avançadas (importação).

De citar que o aparecimento do alerta para confirmação da importação base de dados visa informar o utilizador para o facto de que se este processo for mal efetuado pode levar ao mal funcionamento da aplicação. Tal sucede-se uma vez que em caso de se escolher esta opção com a base de dados carregada, o primeiro passo deste processo consiste na eliminação de toda a informação existente da base de dados antes de iniciar a importação da informação. Se por algum motivo o utilizador não possuir o ficheiro com a informação correta, a aplicação

poderá não importar a informação na sua globalidade, acontecendo o mesmo para o caso de o ficheiro ter a sua estrutura mal formada.

De forma análoga à anterior, a Figura 5-13 exibe a interface para o processo de exportação.

Figura 5-13: Interface gráfico para sub-módulo Configurações Avançadas (exportação).

Relativamente à exportação, é importante mencionar que o ficheiro gerado ficará acessível numa pasta, criada aquando da instalação da aplicação, e de acesso público.

Por fim, como se pode constatar ainda na Figura 5-11, este sub-módulo oferece ainda a opção “Adicionar Zona”. Esta opção permite adicionar uma zona de qualquer comando suportado em qualquer divisão, caso o utilizador pretenda expandir o sistema físico, sem que para isso seja necessário a utilização de softwares ou aplicações adicionais. Esta opção encontra-se representada na Figura 5-14.

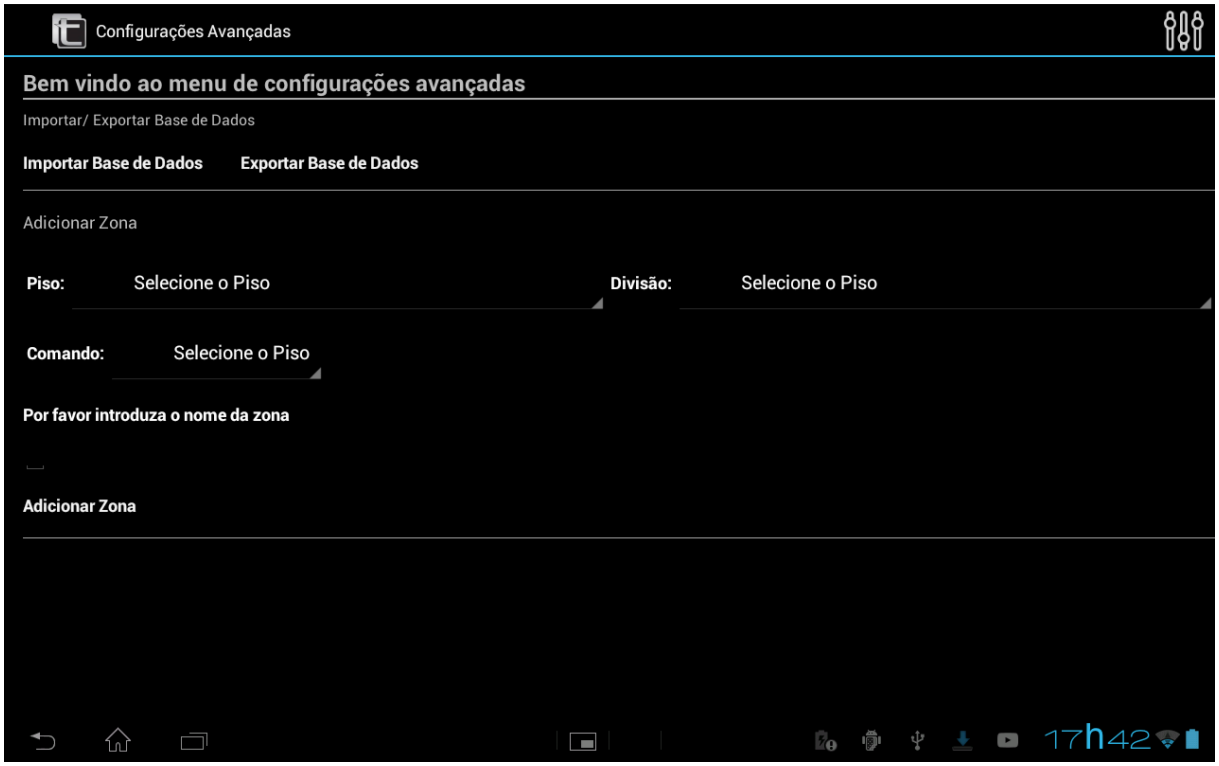


Figura 5-14: Interface gráfico para sub-módulo Configurações Avançadas (adicionar zona).

Na Figura 5-14 pode-se visualizar que para adicionar uma zona, só é necessário identificar o espaço físico da mesma, ou seja, indicar em que piso e divisão se encontra. Após o utilizador prestar essas informações, só falta introduzir o nome da nova zona, sendo que esta etapa possui a mesma restrição referida no sub-menu das macros.

De salientar que esta opção só pode ser acedida pela introdução de uma credencial especial aquando da seleção do menu de configurações avançadas. É exercido este bloqueio uma vez que esta opção quando mal utilizada, fornece à aplicação informações erradas e que pode levar ao seu mau funcionamento, visto que a consola de comando não possui a zona adicionada.

6. Conclusões e Trabalhos Futuros

Tratando-se do último capítulo desta dissertação, neste capítulo vão ser apresentadas as conclusões e os possíveis trabalhos futuros a serem desenvolvidos após esta dissertação.

Numa primeira secção, abordam-se as conclusões gerais no que diz respeito ao desenvolvimento e funcionamento da aplicação “ ”, bem como à temática em si. Na secção 6.2 apresentam-se os possíveis trabalhos futuros a serem desenvolvidos que visam melhorar o funcionamento da aplicação e ainda potenciar as capacidades do sistema em geral.

6.1 Conclusões Gerais

Relativamente à automação residencial, pôde-se constatar que se trata de uma área em crescimento e expansão na sociedade, devido à alteração do paradigma do conceito de habitação. Essa alteração associada às inovações tecnológicas, possibilitaram não só aumentar os níveis de conforto e segurança nas habitações, mas também de dotar as habitações de mecanismos para que estas se tornem autónomas, quando devidamente programadas.

No que diz respeito ao conteúdo desta dissertação, este apresentou uma solução versátil para o controlo e gestão dum sistema de domótica existente, utilizando meios tecnológicos já existentes e amplamente utilizados pela sociedade. Essa solução passou pelo desenvolvimento de uma aplicação para a plataforma móvel Android™, que permite ao seu utilizador controlar os parâmetros físicos do sistema, e fazer adicionalmente a gestão do mesmo sistema.

Quanto ao resultado final do desenvolvimento da aplicação, esta provou ser uma aplicação intuitiva no controlo dos parâmetros, visto que fornece sempre qual o ponto em que o utilizador se encontra a efetuar a alteração necessária. Ainda no controlo, a aplicação possui a capacidade de realizar a atualização do valor do parâmetro selecionado em caso de inatividade da aplicação quando esta se encontra em execução, ou quando este é selecionado. Esta atualização permite ao utilizador ter conhecimento do valor atual do parâmetro, caso este, por exemplo, seja alterado na consola de comando que se encontra em cada divisão. Adicionalmente, a aplicação possui ferramentas como as configurações básicas que visam potenciar a gestão do sistema, através de alterações das designações ou ícones dos pisos, divisões e zonas. Outra ferramenta importante disponibilizada nesta aplicação foi a configuração das macros, componente vital neste tipo de sistemas, tendo em conta que permite configurar cenários específicos por parte do utilizador tendo em conta as funções a desempenhar por estes.

6.2 Trabalhos Futuros

Tendo em conta que todos os objetivos propostos foram atingidos, não existiria a necessidade da execução de trabalhos futuros. No entanto aquando do desenvolvimento da aplicação e a análise das capacidades do sistema, surgiram possíveis inovações para a aplicação e o sistema já implementado, que são:

- Otimização do funcionamento das interfaces gráficas, de modo a que o utilizador possa ter acesso a um menu de controlo simples de todos os parâmetros de controlo inseridos na mesma divisão. Este menu podia permitir que o utilizador possa executar ações simples como desligar/ligar a luz em determinadas zonas sem que tenha de efetuar a seleção individual das mesmas.
- Implementar a configuração de eventos em paralelo com a configuração de macros, isto é, permitir configurar eventos que não necessitem que a aplicação “ “ esteja a ser executada, para serem realizados pelo sistema.
- Incorporar um nó no sistema com a função de armazenar a informação relativa à base de dados do sistema, ficheiros de configurações, entre outros. Tal nó evita a necessidade dos utilizadores possuírem, nos seus dispositivos, uma pasta adicional com essa mesma informação.
- Desenvolvimento da aplicação “ “ para a plataforma IOS, de modo a aumentar a compatibilidade do sistema “ “.
- Implementação dum módulo de comandos por voz, uma vez que se trata de uma opção cada vez mais adotada pelos sistemas de automação residencial e suportada nativamente pelas plataformas Android™ e IOS.

REFERÊNCIAS BIBLIOGRÁFICAS

Acromag Incorporated. (2005). Introduction to ModBus TCP/IP. Obtido de http://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf

Anacleto, J. A. da C. (2012, Julho). *Desenvolvimento de uma aplicação web para dispositivos móveis-Monitorização e controlo de uma rede digital signage.* (Tese mestrado). Universidade do Minho, Guimarães. Obtido de Repositório Institucional da Universidade do Minho.

Android Developers. (2014a, Agosto). Activity. Obtido 9 de Dezembro de 2014, de http://developer.android.com/reference/android/app/Activity.html

Android Developers. (2014b, Agosto). ADT Plugin. Obtido 20 de Agosto de 2014, de http://developer.android.com/tools/sdk/eclipse-adt.html

Android Developers. (2014c, Agosto). Android SDK. Obtido 20 de Agosto de 2014, de http://developer.android.com/sdk/index.html

Android Developers. (2014d, Agosto). AsyncTask. Obtido de http://developer.android.com/reference/android/os/AsyncTask.html

Android Developers. (2014e, Agosto). SQLiteDatabase. Obtido 5 de Setembro de 2014, de http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html

Android Developers. (2014f, Agosto). SQLiteOpenHelper. Obtido 25 de Agosto de 2014, de http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html

Android Developers. (2014g, Agosto). TextView. Obtido 26 de Agosto de 2014, de http://developer.android.com/reference/android/widget/TextView.html

Android Developers. (2014h, Agosto). TimePicker. Obtido 26 de Agosto de 2014, de http://developer.android.com/reference/android/widget/TimePicker.html

Android Developers. (2014i, Dezembro). Widget. Obtido 11 de Dezembro de 2014, de http://developer.android.com/reference/android/widget/package-summary.html

Android Developers. (2015, Março). Navigation Drawer. Obtido 5 de Março de 2015, de https://developer.android.com/design/patterns/navigation-drawer.html

Barros, A. (2010, Dezembro 10). *Edifícios Inteligentes e a Domótica Proposta de um Projecto de Automação Residencial utilizando o protocolo X-10* (Tese). Universidade Jean Piaget de Cabo Verde. Obtido de Biblioteca digital da UniPiaget.

Barry Grussling. (2014). X10 Home Automation. *X10 Home Automation*, 7.

Charlton, J., & Wimberger, D. (2010). JaMod - Java ModBus Library [Informativo]. Obtido 23 de Agosto de 2014, de http://jamod.sourceforge.net/kb/protocol.html

Control4. (2013, Outubro). System User Guide. Obtido de http://www.control4.com/docs/product/control4-system/user-guide/latest/control4-system-user-guide-rev-t.pdf

Control4. (2014a, Agosto). Control4 - Aplicativo. Obtido 5 de Agosto de 2014, de http://www.control4.com/products/app

Control4. (2014b, Agosto). Control4 - Sistema. Obtido 5 de Agosto de 2014, de http://www.control4.com/products/system-overview

Control4. (2014c, Agosto). Control4 - Sistema Operativo. Obtido 7 de Agosto de 2014, de http://www.control4.com/products/software/control4-operating-system

Control4. (2015). Control4 - Representante Certificado. Obtido 28 de Maio de 2015, de http://www.control4.com/dealer_locator

Controls, J. (2009). residential heating solutions: Efficient & comfortable heat for your home. Johnson Controls, Inc. Obtido de ttp://www.johnsoncontrols.co.uk/unitary

Crestron. (2014a, Agosto). Crestron Mobile - Aplicações. Obtido 6 de Agosto de 2014, de http://www.crestron.com/products/crestron_mobile_apps/

Crestron. (2014b, Agosto). Sistemas Séries 3. Obtido 7 de Agosto de 2014, de http://www.crestron.com/products/control_systems/3series/

Damas, L. (2005). *SQL - Structured Query Language* (6.ª ed.). FCA.

Echelon Corporation. (2009). Introduction to the LonWorks Platform - revision 2. Echelon Corporation. Obtido de http://downloads.echelon.com/support/documentation/manuals/general/078-0183-01B_Intro_to_LonWorks_Rev_2.pdf

Eclipse Foundation. (2014, Agosto). Eclipse. Obtido 20 de Agosto de 2014, de https://www.eclipse.org/

Eletronica PT. (2014, Setembro). X10 - Casa Inteligente, Domótica. Obtido 8 de Setembro de 2014, de http://www.electronica-pt.com/content/view/70/

Google Inc. (2014, Agosto). Android. Obtido 8 de Abril de 2015, de https://www.android.com/

Gouveia, P. E. F. (2009). *DOMUS-A: automação de ambientes residenciais* (Tese mestrado). Universidade de Aveiro, Aveiro. Obtido de Repositório Institucional da Universidade de Aveiro.

HomeSeer. (2014a, Abril). HomeSeer 3 - End User Documentation. HomeSeer Technologies. Obtido de http://homeseer.com/support/homeseer/HS3/HS3Help.pdf

Homeseer. (2014, Agosto). Homeseer - Companhia. Obtido 4 de Agosto de 2014, de http://homeseer.com/company/index.htm

HomeSeer. (2014b, Agosto). HomeSeer- Controlador HomeTroller™. Obtido 4 de Agosto de 2014, de http://www.homeseer.com/products/hardware/hometroller.htm

HomeSeer. (2014c, Agosto). HomeSeer HS3 - Software. Obtido 4 de Agosto de 2014, de http://store.homeseer.com/store/HomeSeer-HS3-Home-Automation-Software-Download-P1524.aspx

HomeSeer. (2014d, Agosto). HomeSeer - HSTouch Server Software. Obtido 4 de Agosto de 2014, de http://store.homeseer.com/store/HomeSeer-HSTouch-Designer-Software-P1753.aspx

IDC Corporate USA. (2015, Abril). IDC: Smartphone OS Market Share 2014, 2013, 2012, and 2011. Obtido 8 de Abril de 2015, de http://www.idc.com/prodserv/smartphone-os-market-share.jsp

INSTEON. (2013). INSTEON - WhitPaper Compared. INSTEON. Obtido de http://www.insteon.com/pdf/insteondetails.pdf

JG Domótica. (2014a, Agosto). Domus - Empresa. Obtido 7 de Agosto de 2014, de http://jgdomotica.com/apresentacao.htm

JG Domótica. (2014b, Agosto). Domus - Exemplo moradia. Obtido 7 de Agosto de 2014, de http://jgdomotica.com/jgdomselplace.htm

JG Domótica. (2014c, Agosto). Sistema Domus. Obtido 7 de Agosto de 2014, de http://jgdomotica.com/jgd-f-por.htm

JG Domótica. (2014d, Agosto). Sistema Domus - Alarme. Obtido 8 de Setembro de 2014, de http://jgdomotica.com/domus%20sistbas%20alarme.htm

Kubismus. (2014, Setembro). ZigBee Home Automation. Obtido 8 de Setembro de 2014, de http://kubismus.cz/index.php?option=com_content&view=article&id=65:zigbee-home-automation-&catid=35:company-news

Larousse. (1999). *Nova Enciclopédia Larousse* (4479.ª ed., Vol. 22). Círculo de Leitores.

ModBus. (2012, Abril). ModBus Application Protocol Specification. ModBus. Obtido de http://www.modbus.org

Modbus-IDA. (2006, Outubro). ModBus messaging on TP/IP implementation guide. Modbus-IDA. Obtido de http://www.Modbus-IDA.org

Neto, C., & Lopes, A. (2004, Novembro). Sistema de Controlo de Edifícios Inteligentes Domótica. Instituto Politécnico de Castelo Branco. Obtido de http://www.domus.areadeservico.com/

Queirós, R. (2013). *Android™ – Introdução ao Desenvolvimento de Aplicações* (Vols. 1–8). FCA.

Silva, L. F. G. da. (2008). *Automação em ambientes residenciais* (Tese mestrado). Universidade de Aveiro, Aveiro. Obtido de Repositório Institucional da Universidade de Aveiro.

SQLabs LLC. (2014, Dezembro). SQLiteManager. Obtido 11 de Dezembro de 2014, de http://www.sqlabs.com/sqlitemanager.php

SQLite. (2014, Agosto). SQLite. Obtido 21 de Agosto de 2014, de http://www.sqlite.org/about.html

Technical Publications Department. (2013). Crestron AV3 & PRO3 3-Series Control Systems Operations Guide. Creston. Obtido de http://www.crestron.com/downloads/pdf/product_manuals/av3_pro3.pdf

Thomas, G. (2008, Agosto). Introduction to the Modbus Protocol. *Contemporary Controls*, p. 4.

Tseng, P.-C., Cheng, R.-S., Chang, Y.-C., & Wen-Shyang Hwang. (2012, Julho 1). Journal of Computers. *Toward Ubiquitous Networking: QoS-aware Residential Gateway with Embedded ZigBee-based Network*, *23*(2), 15.

ZigBee Alliance. (2014, Agosto). ZigBee Alliance. Obtido 22 de Agosto de 2014, de http://www.zigbee.org/Home.aspx

Z-Wave Alliance. (2014, Agosto). Z-Wave Alliance. Obtido 22 de Agosto de 2014, de http://www.z-wavealliance.org/

Anexos

ANEXO A. LISTA DISPOSITIVOS SUPORTADOS PELO SISTEMA HOMESEER

Z-Wave Hardware

Category	Company	Device	Purpose
PC Interfaces	HomeSeer	Z-Troller™	Z-Wave Serial PC Interface / Primary Controller
		SmartStick™	Z-Wave USB PC Interface
		ZU0100-001	Z-Wave USB PC Interface
	ACT / HomePro	All "ZCU" Series	Z-Wave USB PC interface
		All "ZCS" Series	Z-Wave RS232 (serial) PC interface
	AEON Labs	USB Stick	Z-Wave USB PC Interface
	ControlThink	USB Stick (Wayne Dalton Firmware)	Z-Wave USB PC Interface
Controllers	AEON Labs	Minimote	Z-Wave handheld remote control
	ACT / HomePro	ZTH100	Z-Wave handheld remote control
		ZTH200 (Euro)	Z-Wave handheld remote control
		All "ZTW" Series	Wall-mounted one-button Z-Wave controller (Decora-style)
	Cooper Wiring	Aspire RF® Series (all)	Handheld remote and wall-mounted scene controllers
	EGuestControls	Evolve ZTM Series	Wall-mounted controller
	GE/Jasco	All	Handheld remote and wall-mounted controllers
Wall Switches	ACT / HomePro	All "ZDW" Series	Z-Wave dimming wall switch
		All "ZRW" Series	Z-Wave relay wall switch
		AS101	Z-Wave companion switch (for Z-Wave 3-way circuits)
	Aeon Labs	All Micro Switches	Z-Wave micro switch and illuminators
	Cooper Wiring	Aspire RF® Series (all)	Z-Wave wall switches
	EGuestControls	Evolve Series	Z-Wave wall switches
	Enerwave	Elegance Z-Wave	Z-Wave wall switches

UPB (Universal Powerline Bus) Hardware

Category	Company	Device	Purpose
PC Interfaces	HAI	36A00-1	UPB RS232 PC Interface **requires HomeSeer plug-in**
	PCS	PIM	UPB RS232 PC Interface **requires HomeSeer plug-in**
	Simply Automated	UMC	UPB RS232 PC Interface **requires HomeSeer plug-in**
Controllers	HAI	38A00-1	UPB room controller
		38A00-2	UPB House controller
	PCS	WM6	UPB wall mount controller
		DTC6	UPB desktop controller
		IR Controller	IR controller
Wall Switches	HAI	35A00-1	UPB dimming wall switch (600 wt)
		35A00-3	UPB non-dimming wall switch (600 wt)
		35A00-2	UPB dimming wall switch (1000 wt)
		35A00-4	UPB non-dimming wall switch (1000 wt)
		40A00-1	UPB relay wall switch (15a)
		37A00-1	UPB aux switch (for 3-way circuits)

INSTEON (by SmartLabs)

Category	Company	Device	Purpose
PC Interfaces	Smarthome	2414U (PLC)	USB PC Interface, **requires free HomeSeer plug-in**
		2412S (PLM)	Serial PC Interface, **requires free HomeSeer plug-in or** 3rd Party Plug-in
		2412U (PLM)	USB PC Interface, **requires free HomeSeer plug-in or** 3rd Party Plug-in
		2413U (PLM)	USB PC Interface, **requires free HomeSeer plug-in** or 3rd Party Plug-in
Controller	Smarthome	KeyPadLinc	Multi-button Wall Controller
Wall Switches	SmartHome	All INSTEON Lamp and Appliance Modules	
Other	SmartHome	Non-Lighting Devices	Requires 3rd Party Plug-in

X-10 Hardware

Category	Company	Device	Purpose
PC Interfaces	ACT / HomePro	TI103, TI213, TI223	X10 RS232 Interface
	SmartHome	1132U, 1132CU	X10 USB Interface (U & CU supported, downloading events not supported for CU version)
	Marrick, LTD	LynX10-PLC	2-Way X-10 Transceiver
	X10 Inc.	CM11A/HD11A/CM12U (220V)	2-Way X10 (downloading events not supported)
		CM15	2-Way X10 with RF (downloading events not supported)
		CM17A (Firecracker)	X10 RS232 - Wireless (RF) Interface
Controllers	Applied Digital Inc.	Ocelot/CPU-XA	2 way X10, 2 way Infrared learning, 2 way input/output relays (downloading events not supported)
	Custom Solutions Inc.	HomeVision	2 way X10, infrared, I/O controller (downloading events not supported)
	X10 Inc.	Palmpad	Handheld RF remote control for X10 devices
	SmartHome	HouseLinc	2 way X10, 1 way infrared non learning (downloading events not supported)

Figura A -1: Lista de dispositivos suportados pelo HomeSeer.

ANEXO. B FUNÇÃO MODBUS WRITE SINGLE REGISTER

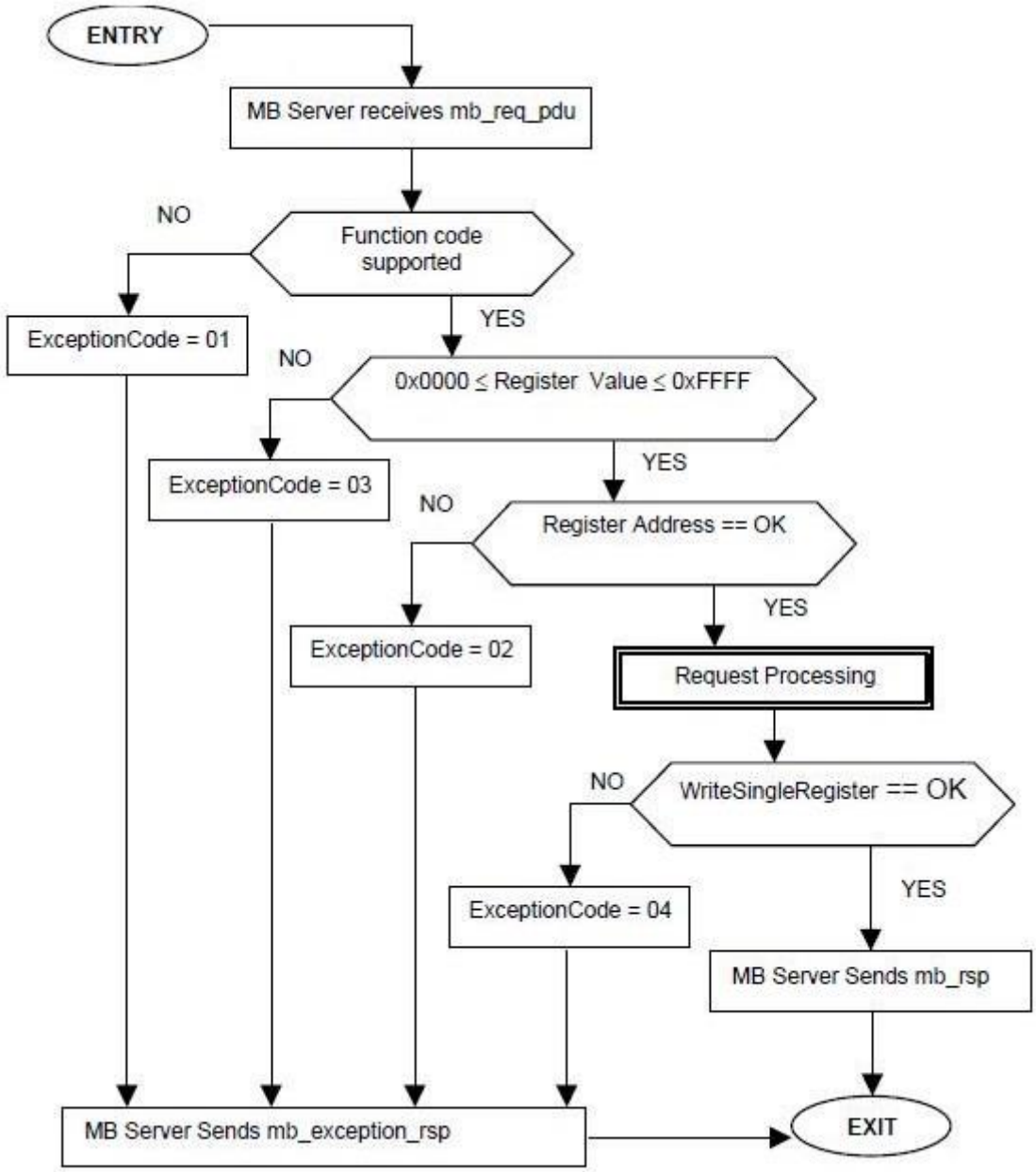


Figura B-2: Diagrama de funcionamento da função *Write Single Register*.

ANEXO. C FUNÇÃO MODBUS READ/WRITE MULTIPLE REGISTERS

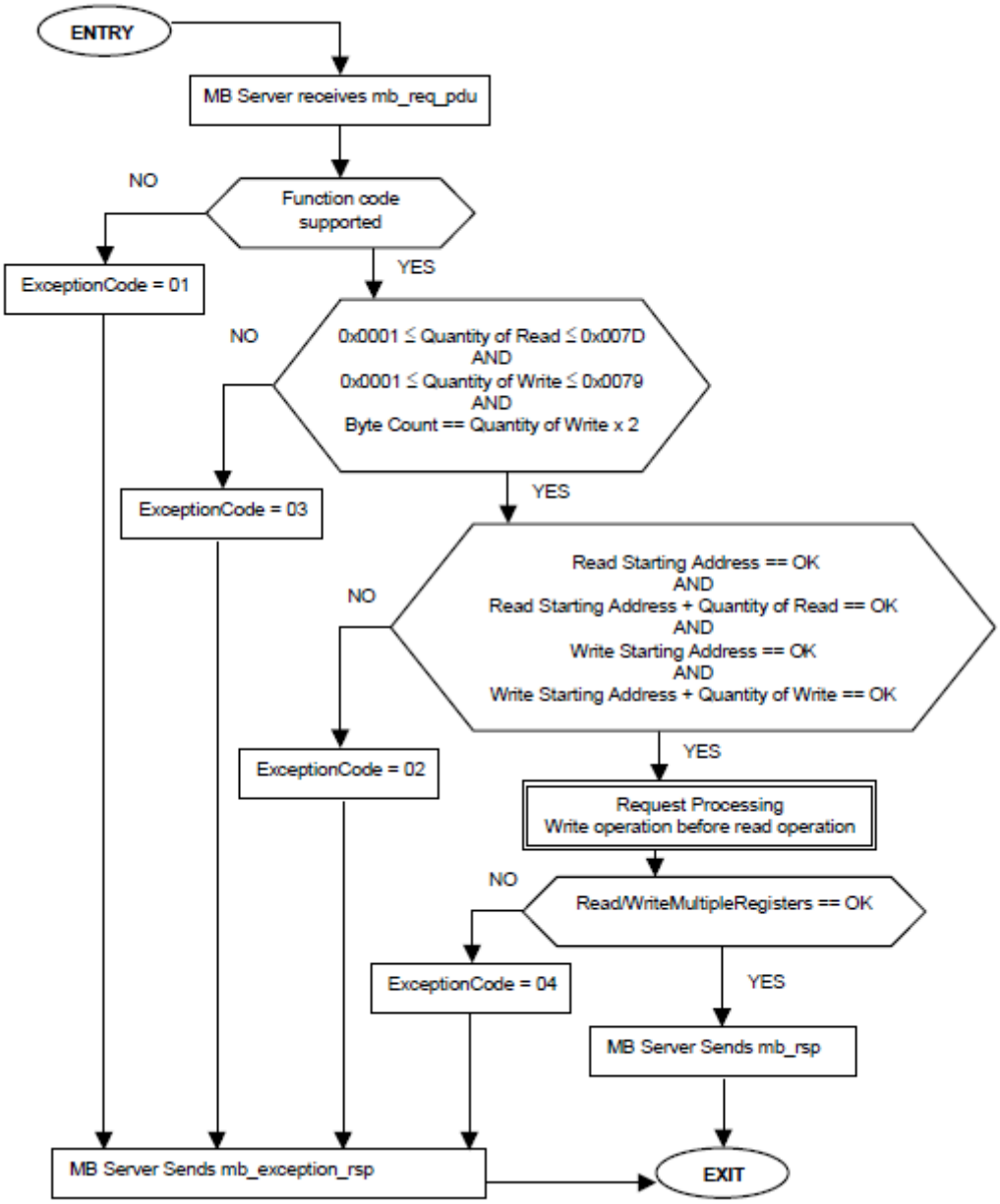


Figura C-3: Diagrama de funcionamento da função Read/*Write Multiple Registers*.