Daniela Mendes Fasero

Licenciada em Ciências da Engenharia Electrotécnica e de Computadores

Plataforma IoT de apoio ao ensino de Engenharia baseada em Laboratórios Virtuais e Remotos

Dissertação para obtenção do Grau de Mestre em
Engenharia Electrotécnica e de Computadores

Orientador: João Rosas, Professor Auxiliar Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

Júri:

Presidente: Doutor Rui Alexandre Nunes Neves da Silva – FCT/UNL

Arguente: Doutora Ana Inês da Silva Oliveira – FCT/UNL

Vogal: Doutor João Almeida das Rosas – FCT/UNL

Setembro, 2019

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Plataforma de Suporte ao ensino de Engenharia baseada em Laboratórios Remotos e Virtuais

Para os meus pais e irmão

Agradecimentos

Este é o último passo de uma etapa tão marcante tanto na minha vida académica, como na minha vida pessoal. Existem imensas pessoas a quem agradecer que é provável que me esqueça de alguém, as minhas desculpas desde já por isso.

Queria começar por agradecer ao meu orientador, Professor João Rosas por assim que o abordei pela primeira vez relativamente a procurar orientação para a tese sempre se mostrou disponível e a partir desse momento sempre me apoiou e ajudou em tudo o que estava ao seu alcance, conseguindo sempre motivar-me quando ás vezes as coisas não pareciam tão positivas, desde já o meu enorme obrigado.

Gostaria de deixar, também, o meu maior obrigado a todos os que passaram pelo meu caminho durante os meus anos nesta instituição desde docentes a alunos, sem qualquer dúvida que a minha vida não seria a mesma sem vocês.

Um especial obrigado ao meu namorado, Tiago Ferreira, que melhor que ninguém conseguiu sempre mostrar-me o lado positivo de todas as situações que vivemos nesta instituição, muito em especial nesta última etapa que sem o seu incansável apoio isto teria sido bastante mais difícil.

Por fim, aos meus pais e irmão que são a melhor família que eu poderia ter escolhido, por toda a motivação e infindáveis ensinamentos ao longo da minha vida. Obrigada por toda a paciência, especialmente nestes últimos meses e por me apoiarem em todas as decisões que tomei, espero ser tão boa filha e irmã para vocês como vocês o são para mim.

Ao resto da minha família, tios, primos e aqueles que embora não tenhamos a sorte de partilhar o mesmo sangue fazem parte da minha família do coração, obrigada por me encorajarem sempre a fazer melhor e a ser uma versão melhor de mim mesma.

Sem vocês todos isto não faria sentido. Obrigada.

Resumo

O ensino da engenharia é uma área crucial no desenvolvimento de recém-licenciados que se querem com um perfil de excelência e isso só é possível quando se possibilita a interação desses alunos com equipamentos modernos e de confiança, que estejam num estado de conservação satisfatório.

No entanto, isto nem sempre é possível pois as instituições encontram-se sobrelotadas e onde apresentam escassez de material de laboratório, sendo o existente antigo e pouco prático de utilizar.

As soluções existentes para a resolução dos desafios apresentados anteriormente não são totalmente satisfatórias ou eficazes, pois os paradigmas utilizados na resolução destes problemas são os de Laboratório Remoto e Laboratório Virtual e estes por si só não garantem o acesso dos alunos aos processos práticos de forma eficiente. Pretende-se agilizar estes problemas fazendo uso de uma plataforma que use as tecnologias de Laboratórios Remotos e Laboratórios Virtuais em conjunto com o conceito de IoT, possibilitando assim ultrapassar a escassez de recursos presente.

Esta plataforma irá ter um impacto significativo no desempenho académico dos alunos tornando a aprendizagem mais eficaz, mesmo face a aulas sobrelotadas ou com escassez de equipamentos.

Palavras-chave: IoT, Internet-das-coisas, Laboratório virtual, Laboratório Remoto, Acesso Remoto

Abstract

Engineering education is a crucial area in the development of recent graduates who want to have a profile of excellence and this is only possible when they allow the interaction of these students with modern and reliable equipment, which are in a satisfactory state of conservation.

However, this is not always possible because the institutions are overcrowded and where laboratory material is scarce, the former being old and impractical to use.

Existing solutions for solving the challenges presented above are not entirely satisfactory or effective, as the paradigms used in solving these problems are those of Remote Lab and Virtual Lab and these alone do not guarantee student access. to practical processes efficiently. These problems are intended to be streamlined by using a platform that uses Remote Labs and Virtual Labs technologies in conjunction with the IoT concept, thereby enabling the present resource scarcity to be overcome.

This platform will have a significant impact on students' academic performance by making learning more effective even in overcrowded or equipment-scarce classes.

Keywords: IoT, Internet-of-thing, Remote Access, Remote Lab, Virtual Lab

Conteúdo

Lista de Figuras

Lista de Tabelas

1

Introdução

1.1 Motivação

A massificação do ensino [1] está a tomar proporções nunca antes vistas, cada vez mais as aulas têm mais alunos o que afecta em muito a qualidade do ensino. Alias, segundo [2] acredita-se que salas de aula com muitos alunos estão directamente relacionadas com resultados menos satisfatórios. Fazendo-se uma extrapolação deste problema para o ensino universitário, percebe-se que se esta massificação afecta a qualidade do ensino fornecido aos alunos, pode-se estar a pôr em causa os profissionais de amanhã em áreas fundamentais para a nossa sociedade como a área da saúde, economia, ciência e tecnologia entre outras.

Quando se fala ao nível do ensino da engenharia, é possível verificar estagnação no nível de investimento para a obtenção de recursos didácticos (figura 1.1), afetando a capacidade das instituições em acompanhar o desenvolvimento tecnológico, e assim, poder prestar um ensino de excelência. Muitos dos recursos existentes encontram-se já num estado obsolescência avançado e estes são fundamentais para uma aprendizagem de qualidade. Em conjugação com esta realidade, o tamanho das turmas também tem vindo a aumentar, contribuindo para a degradação da qualidade das aulas.

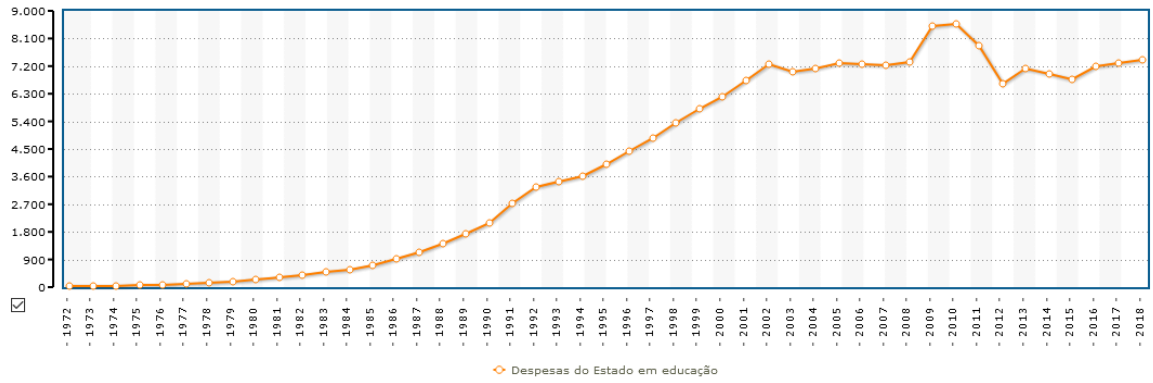

Figura 1.1 -Despesas do Estado em educação até ao ano de 2018 (em Milhões de Euros) [retirado de [3]]

No gráfico da figura 1.2, é ilustrado o número total de alunos que se matricularam no ensino superior desde 1978. No entanto, o mais interessante de notar é que desde o fim dos anos 80/inicio dos anos 90 até aos anos 2000 que é possível verificar em números reais a massificação do ensino a ocorrer. Esta massificação ainda se manifesta actualmente, não estando numa subida tão abrupta aquando do seu aparecimento, mas ainda com uma grande proeminência.

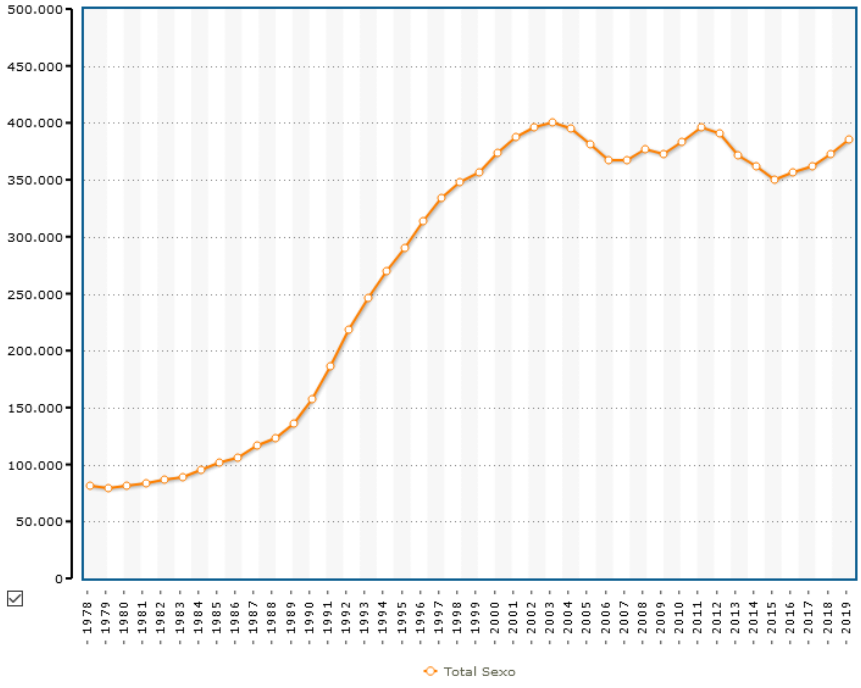

Figura 1.2 - Alunos matriculados no ensino superior em Portugal nas últimas 4 décadas [retirado de [4]]

Embora este problema não tenha surgido só agora, ainda não foi possível colmatar todas as falhas que são proporcionadas por estas dificuldades, apesar de já muito se ter tentado fazer com a implementação de conceitos como o de laboratórios virtuais que visam a emular o processo real de forma digital, isto no entanto não proporciona aos estudantes uma abordagem mais "mãos-na-massa" ao dito processo ou de laboratórios remotos que permitem o acesso ao laboratório físico de forma remota.

1.2 Objectivos

De uma forma mais particular, pretende-se abordar este problema na área do ensino da engenharia, já que o interesse nestas áreas tem vindo a aumentar 6% nos últimos anos quatro anos (figura 1.3). Na área da engenharia, o maior impacto da massificação do ensino, vê-se em aspectos como a disponibilidade de ferramentas nos laboratórios, permitindo assim que se tenha uma experiência real no processo/sistema físico em questão. Pretende-se, por isso, construir uma plataforma que ajude a colmatar estes e tantos outros problemas que derivam desta massificação, para tal utilizar-se-á tecnologias como a tecnologia IoT, serviços web em parceria com conceitos como os de laboratórios remotos e laboratórios virtuais.

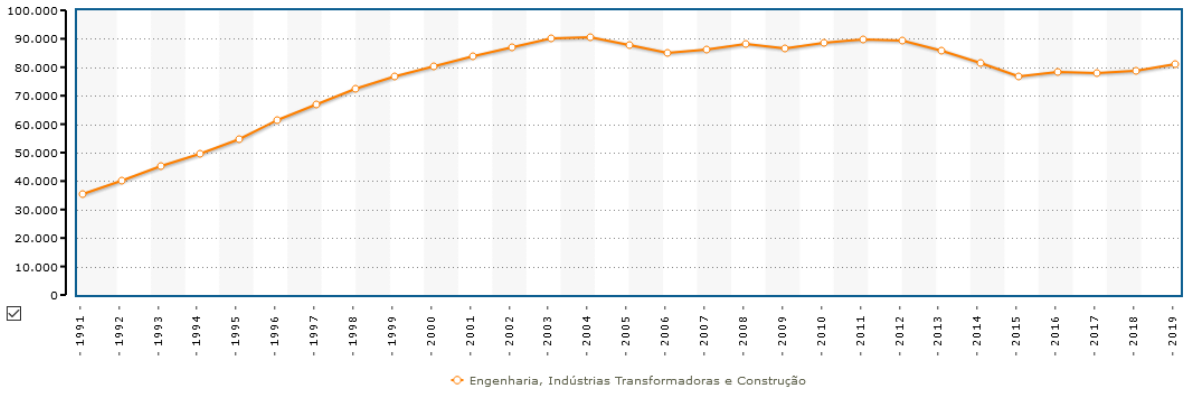

Figura 1.3 - Alunos Matriculados no ensino superior em Portugal nas ultimas 3 décadas na área da engenharia, industrias transformadoras e construção [retirado de [3]]

De forma a implementar a plataforma proposta, foi importante especificar os requisitos necessários ao funcionamento dessa mesma plataforma de onde foram extrapolados os use-cases do respectivo sistema. Seguidamente, passou-se ao desenvolvimento de uma arquitectura conceptual da plataforma, o que colmatou na implementação de dois grandes módulos, o módulo composto pela camada de hardware que ficaria encarregue de providenciar a ligação física entre o kit didáctico real e o dispositivo IoT e o módulo onde estaria incorporada a camada de software, que iria ser responsável por tratar da ligação dos estudantes ao dispositivo IoT.

1.3 Estrutura do documento

A presente dissertação está dividida em 6 capítulos, começando pelo capítulo 2 ,encontramos todos os conceitos relevantes para a implementação da solução proposta anteriormente. Já no capítulo 3, faz-se uma descrição geral do problema, enumerando os requisitos funcionais e não funcionais da plataforma e os seus respectivos use-cases, as abordagens para a resolução deste problema, faz-se também uma referência a como é efectuada a troca entre os vários ambientes presentes nesta plataforma, por fim introduzem-se os dispositivos *IoT* que foram usados durante esta dissertação. No capítulo 4, fala-se da implementação da plataforma por blocos da arquitectura. Seguidamente no capítulo 5, temos os resultados encontrados aquando da verificação dos requisitos funcionais, mas também da validação da solução implementada em cenários hipotéticos, mas bastante reais e por fim, no capítulo 6 encontram-se as conclusões extrapoladas dos resultados alcançados, mas também de todo o trabalho feito e o que fica por fazer como trabalho futuro.

2

Revisão da Literatura

Este capítulo tem como objectivo ilustrar as vantagens da introdução de dispositivos *IoT*, tecnologias e-learning, laboratórios remotos, laboratórios virtuais entre outros num ambiente escolar, neste caso no ensino universitário na área das engenharias.

Dá-se também a conhecer as principais tecnologias de apoio ao e-learning, começando por uma introdução ao conceito de e-learning e depois introduzindo os temas de serviços web e sistemas de aquisição de dados, fazendo questão de mencionar os projectos mais relevantes em cada uma dessas tecnologias.

No fim deste capítulo, faz-se uma breve síntese sobre os temos abordados neste capítulo e a grande questão que se deixa em aberto.

2.1 O futuro do ensino da engenharia

Encontramo-nos numa época especial, tecnologicamente falando, uma época em que o estudo da engenharia se pode fazer de uma forma bastante mais prática e com uma abordagem mais "hands-on". Isto deve-se principalmente ao desenvolvimento tecnológico que se faz sentir na sociedade, neste momento, mas em especial à possibilidade que os dispositivos *IoT* trouxeram, como é o caso de possibilitar acessos remotos aos laboratórios, "remote labs", ou até mesmo a criação de laboratórios virtuais, "virtual labs" (estes dois conceitos estão explicados mais em detalhe nas secções 2.3.1 e 2.3.2). Aliás segundo [4], os "remote labs"

têm tendência a permitir aos alunos acederem aos laboratórios sem saírem do conforto da sua casa através de software de simulação. Por isso, são considerados como um substituto aos laboratórios físicos ou como complemento á experiência real, facilitando por isso o acesso de quem tem de se deslocar consideravelmente para conseguir aceder ao laboratório na sua universidade ou em qualquer outra instituição de ensino.

2.2 Tecnologias de suporte e-learning

O termo e-learning nasceu nos anos 90 e desenvolveu-se juntamente com a expansão do World Wide Web, onde o seu objectivo principal era criar uma comunidade de inquérito independente da localização ou do fuso horário, fazendo uso das tecnologias de comunicação e informação. Hoje em dia, o e-learning deve e pode ser visto, como um complemento da educação presencial, embora este tipo de metodologias possa modificar, mesmo que de forma gradual, o paradigma da educação [5], já que a forma como se encaram as metodologias de e-learning tem sido a mesma há pelo menos mais de uma década. O e-learning passou de uma ideia radical para algo que e considerado comum no nosso conceito de aprendizagem [6].

A forma mais conhecida de e-learning nos dias de hoje são os cursos online, onde a partir destes se desenvolveu uma indústria, passando por fornecedores de recursos, por exemplo o MIT, até aos arquitectos de materiais de aprendizagem. Seguidamente, enumeram-se algumas tecnologias que poderão ajudar na mudança do paradigma do e-learning numa perspectiva educacional.

2.2.1 Serviços Web

Um serviço web é maioritariamente visto como uma aplicação acessível por outras aplicações via web, mas esta definição é bastante abrangente, já que segundo [7] a mesma tudo o que possa ser acessível via URL poderá ser considerado um serviço web, o que nem sempre é verdade. Existe outra caracterização de serviços web feita pelo consorcio UDDI [8], que diz: “serviço web pode ser

definido como aplicações independentes, com um modelo de negócio, abertas para interfaces padrão base que sejam orientadas para a Internet" e esta sim é uma definição bastante mais especifica que a mencionada acima. Existem muitas mais formas de definir um serviço web mas para o âmbito do trabalho aqui apresentado, não faz sentido abordar todas as definições determinadas ao longo dos tempos.

Para se puder usar um serviço web, é preciso ter-se instalado um servidor SOAP, ter conhecimento de WSDL, Web Services Description Language, tipo de linguagem associada a um ficheiro que tem por objectivo descrever onde o serviço está localizado, transporte usado e os parâmetros disponíveis no serviço. Por fim, é necessário ter conhecimentos sobre a framework UDDI, pois esta define um registo global onde está guardada a informação em XML.

Há variadas infra-estruturas que podem ser usadas na implementação de um serviço web. No entanto, a arquitectura clássica [9] de um serviço deste tipo possui as seguintes 3 camadas:

- Service Requester: pessoa/organização que precisa de uma certa funcionalidade implementada num serviço web.
- Service Provider: responsável por criar e disponibilizar o serviço web.
- Service Registry: responsável por guardar a informação relativa á pessoa/organização (requester) e apontar para um ficheiro WSDL que inclui os detalhes necessários de como se uso o serviço.

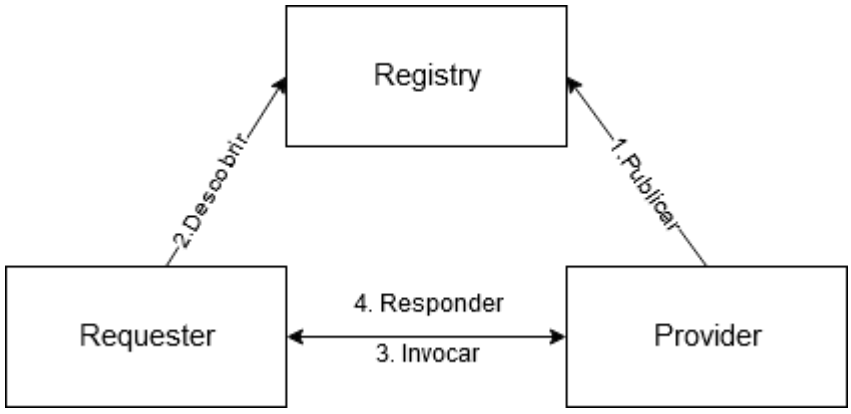

Figura 2.1 - Arquitectura clássica de um serviço web: Web Service Publish & Discovery Architecture inspirado em [9]

De forma bastante resumida, a arquitectura descrita na figura 2.1 tem o seguinte funcionamento [10]: o Service Provider implementa uma colecção de serviços, cujas respectivas descrições são publicadas num Service Registry. O Service Requester, sempre que necessitar dum serviço, executa uma consulta ao Service Registry, fazendo uso de uma operação de busca, e posteriormente usa essa descrição como forma de criar uma ligação com o Service Provider. Assim que é criada essa ligação, é invocado o serviço web.

Alguns exemplos de serviços web relevantes são os serviços web da IBM, nomeadamente o IBM JAX – RS, que simplifica o desenvolvimento de aplicações REST e o IBM JAX – WS, que é apresentada como a próxima geração de modelos de programação de serviços web, afirmando possibilitar o desenvolvimento de serviços web usando "proxies" dinâmicos e anotações Java.

2.2.2 Internet das coisas

A internet das coisas, que nos seus primórdios era denominada como “Challenges to the Network”[11] , ou seja, “Desafios da Rede”, e tinha como objectivo inicial propagar informação recolhida pelos dispositivos IoT através da Internet. Actualmente, é mais conhecida como IoT e pode ser caracterizada como uma rede de dispositivos inteligentes onde estes conseguem recolher e partilhar informação entre si, seja essa informação relativa ao ambiente em que se encontram a funcionar, ou como estão a ser usados. Embora o conceito de IoT continue em desenvolvimento, este tem demostrado especial interesse devido á possibilidade que traz de globalização numa rede de objectos físicos.

Para que a interacção com o mundo real seja possível, faz-se uso tanto de sensores como de actuadores a um nível físico - hardware - e de protocolos de internet num nível mais abstracto - software -. Os interlocutores da camada física podem ser encontrados em praticamente todos os dispositivos materiais, como por exemplo: “smartphones”, semáforos inteligentes (semáforos que são accionados por sensores) e em quase tudo com o qual interagimos no dia-a-dia. A informação partilhada por estes dispositivos tem como função principal melhorar,

seja a eficiência ou até mesmo prevenir qualquer imprevisto que possa acontecer.

Alias segundo [12] os "dispositivos IoT são uma evolução radical da Internet que passou de permitir intercomunicação entre humanos para uma rede de dispositivos interconectados" e isto é corroborado pela "tendência" que se tem vindo a desenvolver ao longo dos últimos anos, como é o caso do desenvolvimento em massa de dispositivos pequenos com grandes capacidades de computação e que incluem também capacidades de comunicação sem fios, fazendo estes dispositivos parte do nosso dia-a-dia.

Existem vários tipos de comunicação no mundo IoT [13], dentro destes tipos os mais comuns são: comunicação inter-dispositivos, denominada por D2D, comunicação entre um dispositivo e um ser humano (e vice-versa) e em casos onde a informação recolhida pelo dispositivo tem de ser guardada num local específico. Existe também a comunicação entre dispositivo e centro de recolha. A comunicação entre dispositivos pode ser usada para recolher informação ou para reportar o estado de cada um deles. Já a comunicação com o ser humano é usada quando há necessidade de uma informação/ordem dada pelo humano ou até para avisar que existe algum tipo de situação que carece da atenção do mesmo. Por último, a comunicação com o centro de recolha tem como objectivo fazer uma actualização de dados já existentes no mesmo, para guardar informação nova ou para recolher informação necessária referente a uma acção automática.

No entanto, a tecnologia IoT não está livre de desafios. Seguidamente, tenta-se enumerar os desafios mais proeminentes [11], nomeadamente:

- Interoperabilidade: é a habilidade que um sistema ou plataforma tem de comunicar e posteriormente colaborar com outros sistemas ou dispositivos - esta é considerada um desafio pois podemos estar a falar de sistemas completamente distintos que eventualmente podem precisar de uma plataforma intermédia para que os mesmos possam comunicar/colaborar entre si.
- Qualidade do Serviço: pode ser resumido como o tempo que uma mensagem demora a ser recebida depois de ser enviada - isto pode

representar um desafio, já que dependendo da quantidade de informação que esta a ser transmitida, os sistemas podem não conseguir garantir o mesmo tempo de recebimento das mensagens.

- Escalabilidade: Possibilidade que o ambiente, em que o sistema está incluído, tem de crescer enquanto se mantém o desempenho do mesmo. Isto representa um desafio, pois dependendo do tipo de sistema, este pode ou não conseguir manter o seu desempenho.
- Consumo de Energia: no caso de se fazer uso de dispositivos que incluem módulos wireless, pode-se ter problemas relacionados com a eficiência do sistema caso se faça uso de baterias.

Existem também outros conceitos que estão intimamente ligadas ao conceito de IoT, como é o caso da Rede de Sensores Wireless (figura 2.2), Wireless Sensor Network (WSN). O WSN é tido como um dos blocos fundamentais do IoT. Este consiste num grupo de sensores que são partilhados com certos nós de sensores de forma a monitorizarem eventos importantes ou certos estados, como por exemplo, som ou temperatura.

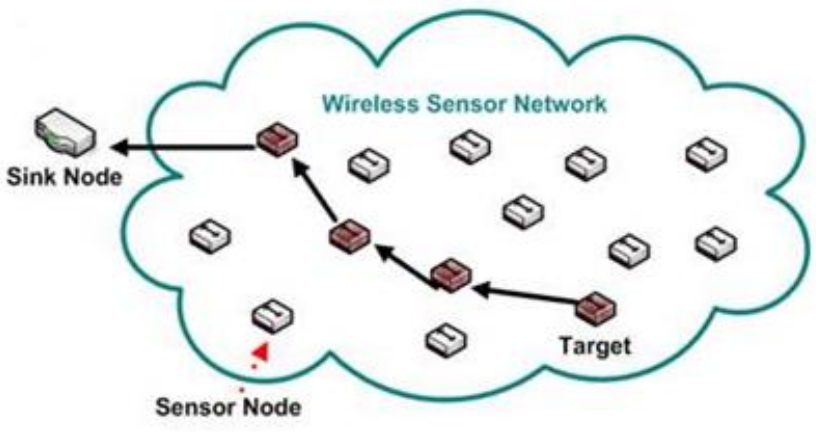

Figura 2.2 - Ilustração de uma rede de sensores (WSN) de [14]

Por fim, dá-se a conhecer um dos projectos ambiciosos relacionados com a Internet das Coisas, este projecto está em desenvolvimento pela IBM juntamente

com a Cisco e pretende possibilitar a aquisição de informação em tempo real em lugares remotos, nomeadamente em plataformas petrolíferas no Mar do Norte ou no Golfo do México fazendo uso de uma tecnologia IoT híbrida, denominada por Watson IoT Platform [15].

2.2.3 Sistemas de aquisição de dados

Perante o tema da educação num nível superior, é bastante comum lidarem-se com sistemas de aquisição de dados, muito em especial em cursos de engenharia, onde estes sistemas são usados. Na maioria dos casos, como forma de monitorizar um ou mais sistemas. O principal objectivo dos sistemas de aquisição de dados, seja ele qual for, é o de recolher informação do mundo físico que depois é enviada para um computador onde pode ser processada.

Os sistemas de aquisição de dados são normalmente formados por uma parte de software, responsável pelo tratamento dos dados recolhidos no mundo real, e por uma componente de hardware, responsável por adquirir os dados no mundo físico como é descrito em [16].

Existem vários exemplos de projectos que incluem sistemas de aquisição de dados e todos eles extremamente interessantes. De seguida, ir-se-á falar sobre dois projectos em particular:

- IceCube [17]: é um observatório de neutrinos, partículas subatómicas sem carga eléctrica que interagem com outras partículas a partir da gravidade e da força nuclear, de elevada energia e onde o seu principal objectivo científico é mapear a energia libertada pelo neutrino no polo Sul, fazendo uso de um sistema de aquisição de dados.
- ANTARES telescópio de neutrino [18]: Este telescópio é usado como forma de estudar recursos astrofísicos a partir da elevada energia emitida pelos neutrinos. O sistema de aquisição de dados neste projecto tem como objectivo converter os sinais analógicos dos tubos foto multiplicadores sensíveis á luz num formato mais apropriado para as análises físicas.

2.2.4 Acesso remoto a sistemas físicos

O acesso remoto na maioria das vezes é efectuado pela internet, usando uma combinação entre interfaces web e uma ligação remota a um computador que posteriormente fará a ligação a um dado servidor.

Existem variadas abordagens que podem ser utilizadas quando se está a falar em acessos remotos de sistemas físicos. Estas podem passar simplesmente pela habilidade de monitorizar a informação vinda de uma única peça de equipamento, até sistemas capazes de alinhar e automaticamente alocar estudantes a um certo equipamento no laboratório.

Numa fase inicial, temporalmente falando, segundo [19], o desenvolvimento do acesso remoto, tinha como preocupação principal a utilização de tecnologias que permitissem o *streaming* de áudio e vídeo em tempo real, tentando-se assim ultrapassar as limitações associadas á largura de banda na altura, enquanto se mantinha a qualidade do serviço e se lidava com os acessos múltiplos de ligações diferentes ao equipamento e/ou laboratórios. Neste momento, as preocupações enumeradas acima foram ultrapassadas, sendo possível sem grande esforço serviços de qualidade, em que se pode exercer uma total confiança. Tanto que neste momento a evolução já não é tão focada numa optimização, mas sim na forma de enriquecer a interacção com o estudante, conforme mencionado em [20].

2.3 Abordagens práticas utilizadas no estudo da engenharia

Como já foi mencionado anteriormente, é conhecido que o estudo da engenharia precisa de uma abordagem prática devido a que no mundo real existem variáveis que não podem ser controlados ou emuladas, o que é também corroborado por [21]. Devido a isso, sempre que possível, é preferível um aluno praticar num laboratório real, ou seja, num laboratório físico. No entanto, a preparação de um certo laboratório tem os seus custos associados, tanto em termos financeiros (compras de material e de espaço), como de tempo já que nunca irá ser um só aluno a trabalhar naquele laboratório, mas sim, na maioria das vezes vários grupos de alunos.

Surgiram então novos conceitos de forma a ajudar na agilização da aprendizagem destes alunos, nomeadamente os *virtual labs* – laboratórios virtuais- e os *remote labs* – laboratórios remotos. Embora sejam conceitos relativamente diferentes entre si, o seu principal objectivo é permitir uma aprendizagem em que os alunos não precisem de se dirigir ao laboratório para fazerem os seus testes e/ou tirarem as suas conclusões.

2.3.1 Laboratórios Remotos

O conceito de laboratório remoto pode ser definido como um sistema que permite controlar, de forma remota, um laboratório físico (figura 2.3). Permitindo assim que tanto docentes como alunos tenham a oportunidade de interagir com certos laboratórios que não estão presentes nas suas unidades de ensino. Este tipo de laboratórios tem custos associados, nomeadamente na sua implementação e manutenção. No entanto, estes custos são bastante mais reduzidos que os custos associados a implementação e manutenção do seu equivalente numa instituição de ensino, existe também a preocupação relativa ao espaço necessário para alocar um laboratório real. Os laboratórios remotos, são na sua generalidade, desenhados e implementados pelos docentes da disciplina/área em si, que irão ser ligados ao processo real onde depois existirá uma camada de ligação entre o utilizador e o software que irá estar a correr num servidor web, como exemplificado durante a elaboração de [20].

Figura 2.3 - Exemplo de laboratório remoto, retirado de [22]

Um dos pacotes de software que pode ser utilizado para a implementação dos laboratórios remotos tem o nome de "*LabView Web Services*" e foi desenvolvido pela National Instruments (figura 2.4). Segundo [23], a utilização deste software reduziu significativamente o esforço utilizado na implementação deste tipo de laboratórios, o que por sua vez proporcionou um aumento significativo na existência de laboratórios remotos.

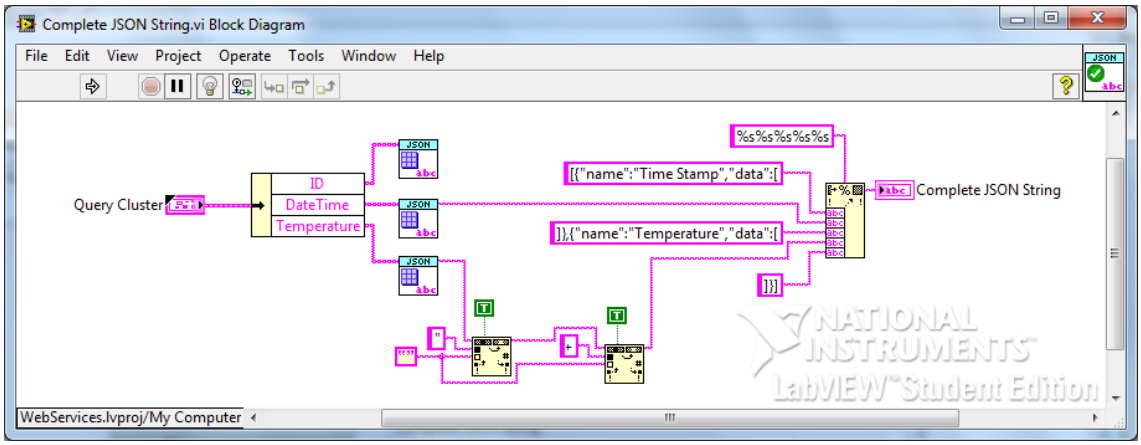

Figura 2.4 - Exemplo da utilização do LabView Web Services, retirado de [24]

E embora os laboratórios remotos sejam maioritariamente vistos como forma de dar acesso ao laboratório a alunos que estejam geograficamente longe [19], estes têm bastantes mais vantagens, como por exemplo: permitir o acesso

de alunos que tenham algum tipo de deficiência e que de outra forma poderiam ter alguma dificuldade em aceder ao processo, promover maior segurança no acesso a certas experiências que possam ser perigosas, permitir o uso de tecnologias de ponta, mas sem dúvida que a vantagem mais proeminente é a possibilidade de obter informação em tempo real e autêntica. Existem também algumas desvantagens relativamente a este tipo de laboratórios, como é o caso de nem sempre poderem ser as melhores representações dos laboratórios reais ou até mesmo de não conseguirem proporcionar ao utilizador um ambiente realístico do facto de se estar a interagir com o processo físico real.

Embora este tipo de laboratório seja, comumente, usado por todo o mundo existe um exemplo em particular que se pretende mencionar. O NetLab [25] foi implementado pela Universidade do Sul da Austrália e tem como principal objectivo proporcionar aos utilizadores uma utilização que permita aos mesmos a sensação de que se está a utilizar o processo real em vez de um laboratório remoto (figura 2.5). Para que isso seja possível, existe uma camara que permite aos utilizadores, visualizarem a experiência e monitorizarem os resultados da experiência à medida que estes vão acontecendo. Este sistema também permite aos alunos comunicarem entre si, usando uma ferramenta de "chat" incorporada no software. Por fim, o NetLab não está só disponível para alunos daquela instituição de ensino, mas também para pessoas externas, desde que estas tenham disponibilidade de usar a ferramenta fora dos espaços de tempo alocados aos alunos.

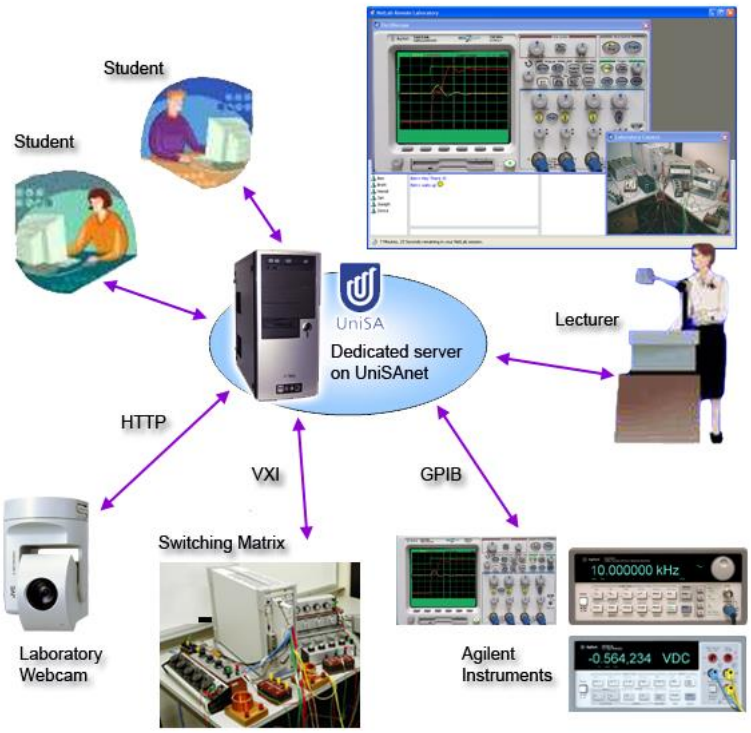

Figura 2.5 - Infra-estrutura do laboratório remoto NetLab [25]

2.3.2 Laboratórios Virtuais

Os laboratórios virtuais são, na maioria dos casos, simuladores de um processo físico que podem ter sido implementados por uma empresa ou por um docente da própria instituição e tem como objectivo possibilitar ao aluno poder estar em qualquer lugar e praticar usando o mesmo, sem a preocupação de estar ou não a danificar o processo real (figura 2.6). No entanto, se este laboratório virtual for a representação de um processo físico, os dados dali retirados podem ser correctos, mas não necessariamente reais, sendo esta a maior desvantagem apresentada pelos laboratórios virtuais.

Os laboratórios virtuais são utilizados, na maioria dos casos, para substituir a utilização dos processos físicos. No entanto, há também a hipótese de estes poderem vir a substituir o papel e a caneta quando relacionados com o trabalho executado pelos alunos em casa como forma de preparação para a utilização dos

laboratórios físicos ou até para a execução de demonstrações como forma de inicialização das aulas de laboratório.

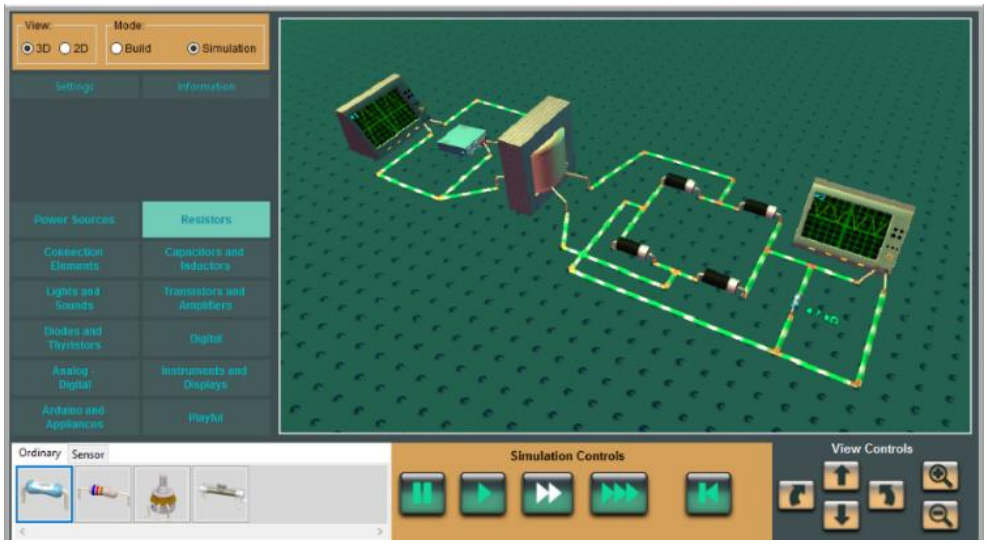

Figura 2.6 – Exemplo de laboratório virtual [26]

Este tipo de laboratórios é bastante útil quando se quer conduzir as experiências passo a passo, sem se precisar de monopolizar o processo real ou até mesmo quando há possibilidade de perigo para a saúde humana. Os laboratórios virtuais também permitem ultrapassar dificuldades que podem ser apresentadas aquando do mau estado dos equipamentos ou da falta de tempo no decorrer das aulas praticas [27].

No entanto, há sempre que ter em atenção que em alguns casos, dependendo de que tipo de processo que se está a "simular", a informação retirada pode não corresponder aos valores "reais" e é também notório que, embora a experiência retirada nos laboratórios virtuais já seja uma mais valia, esta não se compara a uma experiência num laboratório real.

O exemplo de aplicação mencionado de seguida foi implementado devido á necessidade que havia de os estudantes conseguirem interagir com um processo que não existia na sua instituição de ensino relacionado com eixos de torção e materiais biológicos, proporcionando assim uma expansão das capacidades daquele laboratório [28]. Este laboratório foi construído de forma modular,

possibilitando assim que pudesse ser utilizado durante as aulas de laboratório, mas também como demonstração nas salas de aula, ou como forma de os alunos realizarem os seus estudos relacionados com as matérias envolventes nesta experiência de laboratório [29].

2.4 Experiências e contribuições comunidade

É possível encontrar-se imensos projectos IoT na internet, onde grande parte desses projectos são somente possíveis graças á partilha de outros utilizadores de dispositivos *IoT* dos seus conhecimentos e experiências. Isto deve-se a inúmeros factores: o primeiro sendo a época em que nos encontramos, onde desde que se tenha um computador com acesso à internet se consegue aprender um pouco de tudo. Outro factor que vale a pena mencionar é que o conceito de *IoT* tem por base fazer a ligação entre vários tipos de dispositivos que de outra forma não conseguiriam interagir entre si. Devido a isso, este conceito está intrinsecamente ligado á partilha de informação.

Existe uma grande variedade de comunidades em que o seu principal objectivo é desenvolver e ajudar quem quer desenvolver os seus projectos *IoT*, seja através de vídeos tutoriais ou guias de como reproduzir um projecto já feito pelo autor. Existem também comunidades para quem lida com *IoT* profissionalmente como é o caso do "The *IoT Community of the ng Connect Program*" [30], que tem como objectivo principal juntar empresas para desenvolverem os seus protótipos, casos de estudo, soluções, etc, promovendo uma aceleração na aquisição de respostas e resultados.

2.5 Sistemas ciber-físicos

O conceito de sistemas ciber-fisicos aparece da necessidade crescente que pretende possibilitar o acesso a informação e serviços em qualquer lugar, algo que acaba por ser inevitável no mundo actual. Estes sistemas permitem a ligação entre o mundo real -físico - e o mundo computacional e as suas infra-estruturas de comunicação através de sistemas automáticos [31]. Estes sistemas propõem-

se a transformar a forma como nós interagimos com o mundo físico e a sua implementação requer diferentes tipos de conhecimentos, em que as três maiores áreas são: controlo, sistemas e informática.

O desenvolvimento destes sistemas é uma tendência em crescimento devido a que cada vez mais existe uma produção de sensores com elevada sensibilidade a um custo bastante reduzido, de computadores com capacidades de processamento elevadas a preços apelativos ou até os avanços nas energias alternativas. Todos estes factores juntos, promovem a proliferação deste tipo de sistemas em áreas como os sistemas aeroespaciais, sistemas de defesa, sistemas robóticos, entre outros [32].

Os sistemas ciber-físicos permitem juntar a logica computacional de forma a monitorizar e se necessário, controlar as dinâmicas dos processos físicos (figura 2.7).

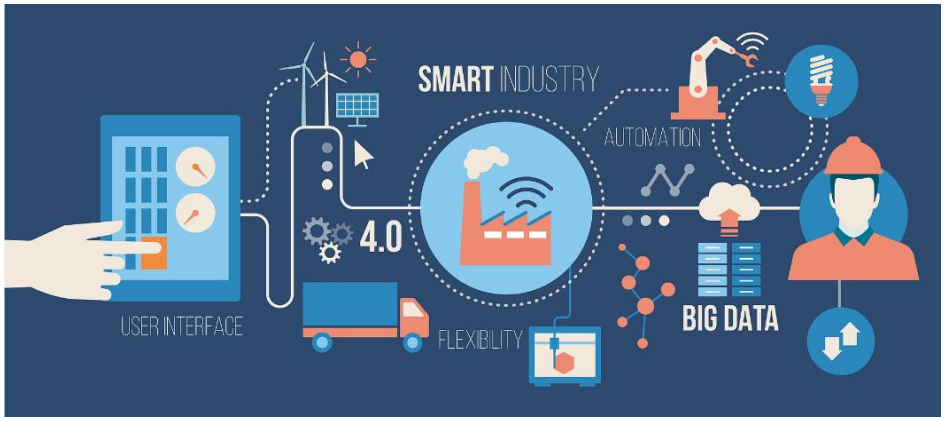

Figura 2.7 - Ilustração sistema ciber-físico [33]

2.6 Síntese e questões em aberto

Até este ponto falou-se das mais emergentes tecnologias dos últimos anos que de alguma forma proporcionam um alívio, seja de recursos físicos (Laboratórios Virtuais, Laboratórios Remotos, Acessos remotos a sistemas físicos, IoT e sistemas de aquisição de dados) seja de recursos teóricos (tecnologias base e-learning), muito preciso no ensino da engenharia no ensino superior. Todas estas tecnologias foram fulcrais para chegarmos ao ponto em que nos encontramos hoje.

A questão que se põe agora é: *"Porque e que nenhuma destas tecnologias consegue satisfazer o problema apresentado nesta dissertação?"*

E a resposta apresentada nesta fase do documento é a seguinte: Cada vez mais os alunos procuram mais o acesso instantâneo a toda e qualquer tipo de informação, nesta era em que se quer tudo "inteligente", desde os nosso telemóveis até aos frigoríficos, e as tecnologias anteriormente apresentadas, quando independentes umas das outras não conseguem proporcionar isso. Pretende-se implementar, nos capítulos seguintes, uma solução que seja tecnologicamente mais interessante para os alunos.

3

Arquitectura do sistema

Neste capítulo inicia-se fazendo um pequeno resumo do propósito do projecto proposto, qual o seu objectivo e o porquê da sua existência. Faz-se também uma pequena comparação entre os conceitos existentes para se satisfazer este problema e as suas vantagens e desvantagens relativamente á abordagem que queremos implementar. Seguidamente, faz-se uma descrição dos requisitos funcionais e dos respectivos use-cases. Aquando da definição destes, cria-se uma arquitectura conceptual, onde se pretende explicar as duas camadas que dividem este projecto, o seu objectivo e muito sucintamente o que se usou para que essas camadas tomassem forma.

Por fim, introduziu-se o tema de troca de ambientes, algo deveras importante na plataforma implementada e também o tema da aquisição e troca de informação, um tema extremamente relevante quando se está a trabalhar com dispositivos IoT.

3.1 Demarcar o Problema

Devido à massificação presente no ensino em geral, mas muito em especial, no ensino da engenharia, começa-se a notar um decréscimo da produtividade no decorrer das aulas práticas, devido à necessidade que existe de as aulas serem

cada vez mais curtas, o que não possibilita aos alunos a interacção necessária com os procedimentos/sistemas, o tempo que se perde na preparação e/ou interacção com os ditos sistemas mas também a inexistência de ferramentas, como o caso dos kits de simulação didácticos, os processo físicos, entre outros, suficientes para que os alunos consigam trabalhar em simultâneo.

O problema que se propõe resolver ajuda a colmatar as lacunas existentes, fazendo uso da tecnologia IoT e do conceito de laboratórios remoto de forma a emular o comportamento dos kits didácticos (figura 3.1) presentes nas salas de aula por todo o mundo.

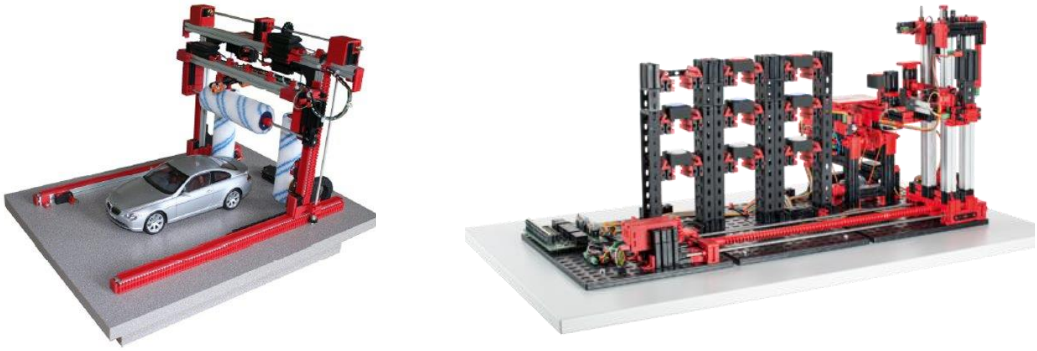

Figura 3.1 - Exemplos dos kits didácticos

Desta forma, pretende-se que o acesso aos kits didácticos seja mais rápido, mais eficaz, mas também que seja mais interessante para os alunos, devido a utilização de uma tecnologia que não temos a oportunidade de ver em "acção" no nosso dia-a-dia, podendo até vir a originar projectos a título próprio pelos alunos num ambiente fora da sala de aula.

À data deste documento, a interacção com os kits reais existentes é feita usando o NI-DAQ (anexo 5 e 6), placa de aquisição de dados, que por um lado é ligado ao computador do aluno e por outro conectada ao kit.

3.2 Abordagens existentes

Na tabela 3.1, pretende-se enumerar as abordagens existentes que permitissem resolver o problema acima descrito, como por exemplo os laboratórios

remotos e os laboratórios virtuais, tal como as suas vantagens e desvantagens relativamente á plataforma que se pretende implementar.

Esta tabela tem especial interesse pois consegue resumir o falado no capítulo 2 de uma forma bastante sucinta, enquanto permite ter uma visão mais focada na informação que realmente importa para este projecto e consequentemente para a solução que se pretende implementar.

Tabela 3.1 - Abordagens existentes e respectivas vantagens e desvantagens

Funcionalidade	Vantagens	Desvantagens
Laboratório Remoto	Permite um acesso remoto aos recursos já existentes; Baixo custo, alta confiança, flexibilidade e conveniência	Embora bastante flexível, não é idealizado para o acesso a recursos que não estejam geograficamente distantes.
Laboratório Virtual	Permite simular os recursos existentes. Permite executar simulações passo-a-passo Permite executar experiências em laboratórios que não estão disponíveis na instituição de ensino	Os alunos não conseguem usufruir de uma abordagem mais "pratica", do ponto de vista da interacção física com o processo.
Laboratório virtual + Laboratório Remoto + Dispositivo IoT (abordagem que se pretende implementar)	Permite utilizar as vantagens mencionadas acima para o LR e LV enquanto possibilita a utilização dentro da sala de aula	Necessidade de mecanismos adicionais de interoperabilidade. Necessidade de coordenar o acesso múltiplo ao sistema físico.

3.3 Requisitos funcionais

Segundo [34] "um requisito funcional tem como objectivo capturar o comportamento esperado do sistema". Os requisitos funcionais são o ponto de partida de qualquer plataforma e se estes não forem realizados com sucesso a plataforma poderá, e provavelmente irá apresentar sérias lacunas.

Seguidamente, é apresentada a tabela 3.2 com os requisitos funcionais e não funcionais referentes à solução proposta.

Tabela 3.2 - Requisitos funcionais e não funcionais

	Requisitos Funcionais
R1	Estabelecer ligação com o kit real
R2	Estabelecer Access Point
R3	Executar diferentes operações de acesso ao kit
R4	Receber informação (servidor e cliente)
R5	Enviar informação (servidor e cliente)
R6	Permitir troca entre plataforma IoT e Simulador
R7	Plataforma funcionar sem quebra (horas/dias)
	Requisitos Não Funcionais
RNF1	Possibilitar acesso singular ao IoT
RNF2	Interface intermédia – possibilidade de comunicação directa com o kit

3.4 Use-cases

Os use-cases estão profundamente ligados aos conceito de requisitos funcionais , aliás segundo [34] "os use-cases são uma forma de representar os

requisitos funcionais. Isto é especialmente verdade para a comunidade de objectos-orientados (...)". Até se pode ir mais longe e frisar o facto de os use-cases não só representarem os requisitos funcionais, mas em particular os requisitos funcionais que os utilizadores conseguem "visualizar", ou seja experienciar. Para este trabalho, os use-case propostos estão ilustrados na figura 3.2.

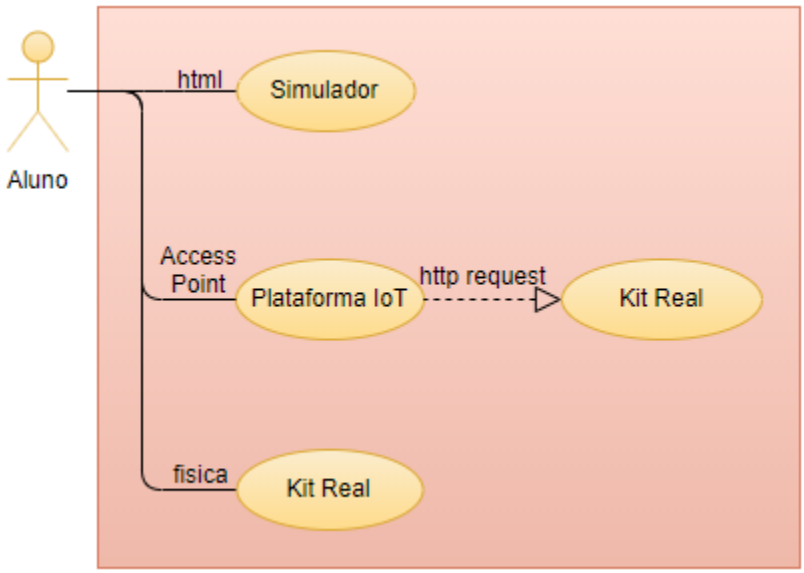

Figura 3.2 - Use-cases da plataforma.

3.5 Arquitectura conceptual

Nesta secção, ir-se-á fazer uma breve referência à visão da arquitectura que se pretende implementar (figura 3.3), introduzindo todos os blocos que são constituintes da mesma e por fim uma breve explicação dos mesmos.

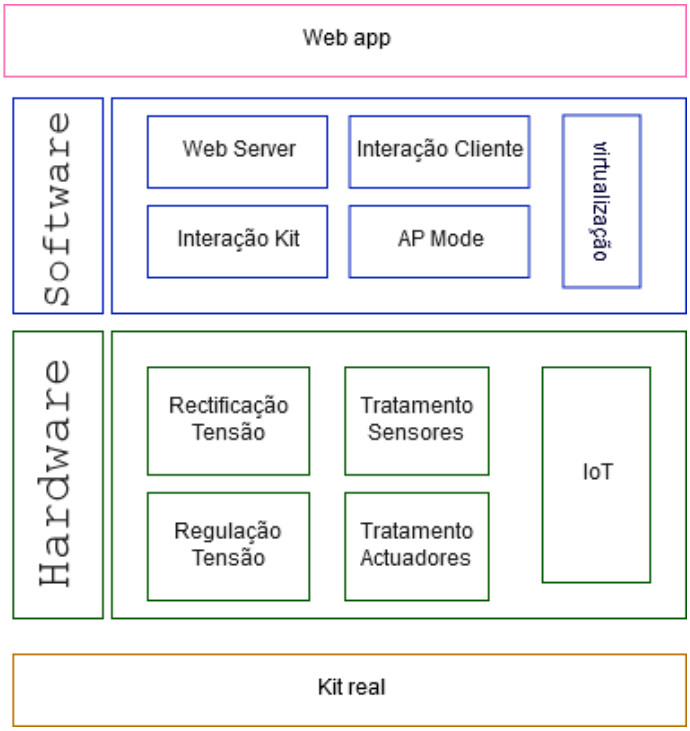

Figura 3.3 - Arquitectura conceptual

3.5.1.1 Camada de Software

1) Web Server: Baseia-se na implementação do web server que permitirá o acesso ao IoT e posteriormente ao kit.
2) Interacção Cliente: Recebimento das mensagens vindas do cliente e posterior envio das mensagens para cliente.
3) Interacção Kit: Tratamentos das instruções provenientes do cliente de forma a que o kit as consiga executar.
4) Ap Mode: Criação de um access point.
5) Virtualização: Inclui o simulador do kit real.

3.5.2 Camada de Hardware

1) Rectificação Tensão: Receber a tensão da rede e transformá-la na tensão de alimentação do kit.
2) Regulador Tensão: A partir da tensão de 24V que alimenta o kit, transformar na tensão de alimentação do dispositivo IoT.
3) Tratamento de Sensores: Trabalhar o sinal dos sensores do kit de forma a poderem ser interpretados pelo IoT.

4) Tratamento de actuadores: Trabalhar o sinal dos actuadores do IoT de forma a que possam ser interpretados pelo kit.
5) IoT: Componente que possibilita a troca de informação entre o kit e o aluno.

3.5.3 Trocas de ambiente

De forma a fazer-se uso das vantagens inerentes aos laboratórios remotos, permitir o acesso remoto a recursos existentes mas não geograficamente próximos, e laboratórios virtuais, permitir executar experiências em laboratórios que não estão disponíveis na unidade de ensino ou simplesmente naquele momento, a plataforma permite uma transição transparente entre o ambiente real, providenciado pelo IoT sob o conceito de laboratório remoto, e o ambiente simulado, sob o conceito de laboratório virtual (figura 3.4). Tornando assim esta plataforma numa tecnologia híbrida dos três conceitos fundamentais desta dissertação: Laboratórios Remotos, Laboratórios Virtuais e Internet-das-coisas.

Esta troca de ambientes é facilitada através de um ficheiro de interface que providência essas trocas.

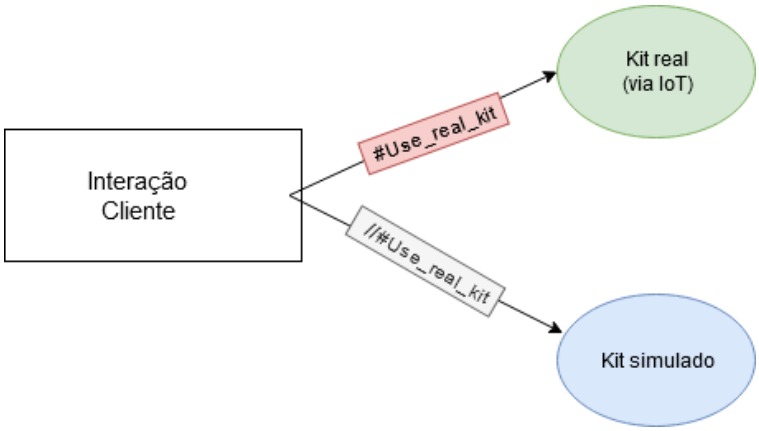

Figura 3.4 - Esquemático relativo às trocas de ambiente de simulação

3.6 Aquisição de informação/trocas de informação

A aquisição e/ou troca de informação é uma parte fulcral deste projecto, podendo até chegar ao ponto de ser a parte mais importante do mesmo, já que sem existirem as ditas trocas de informações, a solução que se está a tentar

implementar não faz qualquer sentido. Se tomarmos em atenção o objectivo principal do que nos propomos aqui, este pode ser resumido em: "fazer a ligação instantânea ao(s) kit(s), permitindo assim uma interacção com o aluno mais fluida e imediata", percebemos que neste contexto fazer "a ligação" pode ser trocado por "adquirir/trocar informação", ficando assim o nosso objectivo mais explicito.

Estas trocas de informação são o que irá impulsionar a interacção com o kit e irá ter como elemento principal a placa IoT, já que este é o elemento de ligação entre as camadas de hardware e de software da arquitectura (figura 3.3). É seguro mencionar que, embora todas as componentes da arquitectura deste sistema estejam envolvidas na aquisição/troca de informação, a camada de hardware toma uma posição mais importante na plataforma, já que sem a sua existência não "haveria" informação adquirida. Relativamente á parte de software, esta irá transmitir as acções pedidas pelo aluno para o modulo de hardware e, caso haja necessidade, transmitirá de volta o valor do(s) porto(s)(explicado na secção 4.4).

No entanto, não seria possível haver qualquer tipo de aquisição ou troca de informação, sem a existência de dois elementos fulcrais nesta plataforma: os sensores, que permitem percepcionar o que está a acontecer no kit e os actuadores, que são os elementos responsáveis por accionar a parte física do sistema, pondo-a em funcionamento.

3.6.1 ESP8266 vs ESP32

Ainda internamente ligado ao conceito de IoT, temos o ESP8266 e o ESP32 que são dois dos dispositivos IoT mais utilizados em projectos "amadores" quando nos referimos a providenciar comunicação entre diferentes tipos de electrodomésticos/componentes electrónicos em geral. Isto deve-se principalmente à facilidade que ambos tem em serem programáveis, nomeadamente usando a aplicação *Arduino IDE* que tem uma interface bastante apelativa e simples.

Ambos os dispositivos foram usados durante o desenvolvimento desta dissertação e seguidamente faz-se uma comparação entre ambos, explicando o que levou à escolha de um em vez do outro.

Estes dispositivos têm em comum o seu fabricante: a *Espressif* sendo o ESP32 o sucessor do ESP8266. Estes dois dispositivos partilham bastantes similaridades, como por exemplo: ambos terem um processador de 32-bit e um modulo de WIFI, o que faz que para o utilizador comum seja indiferente qual IoT escolher. No entanto, caso seja necessária mais capacidade de processamento ou mais GPIOs (General *Purpose Input/Output*), entre outros, então já se tem de fazer uma escolha mais informada. Na tabela 3.3 encontram-se as principais diferenças entre estes dois dispositivos.

Tabela 3.3 - Principais diferenças entre o ESP8266 e ESP32

	ESP8266	*ESP32*
Processador	Single core (160MHz)	Dual core (80Mhz - 240MHz)
Rapidez de WIFI	Até 72.2 Mbps	802.11n até 150 Mbps
GPIOs	17	36
BLE	Não	Sim
DAC	Não	2 canais de 8-bit
ADC	10-bit SAR	12-bit SAR
Sensor de Temperatura	Não	Sim
Sensor de efeito de Hall	Não	Sim
Protocolos de Rede	IPv4, TCP/UDP/HTTP/MQTT	IPv4, IPv6, SSL, TCP/UDP/HTTP/FTP/MQTT
Temperaturas de funcionamento	40°C ~ 125°C	–40°C ~ +85°C

Conforme ilustrado na figura 3.5, os dispositivos mencionados são similares em termos de layout físico, tendo facilitado a troca de um para o outro, neste caso, o ESP32.

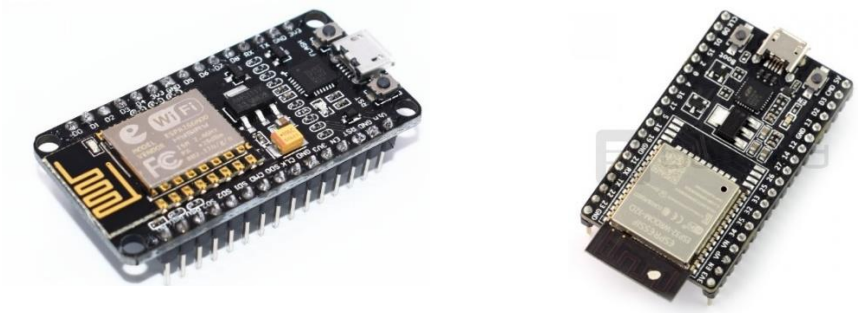

Figura 3.5 - Esp8266 (á esquerda) e Esp32(á direita)

Seguidamente, na tabela 3.4, estão enumeradas as principais funcionalidades presentes no ESP32.

Tabela 3.4 - Descrição das funcionalidades dos componentes presentes no Esp32 [35]

Componente	Descrição
ESP32-WROOM-32	Modulo com a placa ESP32 no seu interior
EN	Botão de reset
Boot	Botão de Download
USB-to-UART Bridge	Chip USB-UART que permite ritmos de transferência até 3Mbps
Micro USB Port	Interface USB. Alimentação da placa e interface de comunicação entre o computar e o modulo ESP32
5V Power On LED	Fica activo quando a placa esta ligada ao cabo USB ou a uma fonte externa de 5V
I/O	Pinos do modulo ESP

Ambos os dispositivos podem funcionar em dois modos (Figura 3.6): o primeiro modo onde é criado um "Access Point"(AP) ao qual todos os que queiram interagir com a placa IoT terão de se ligar a essa rede ou o modo STA(Station), no qual o dispositivo terá de efectuar a ligação a uma rede wireless pré-existente e posteriormente qualquer interacção a ser feita por terceiros, terá como requisito primário fazer a conexão a essa mesma rede.

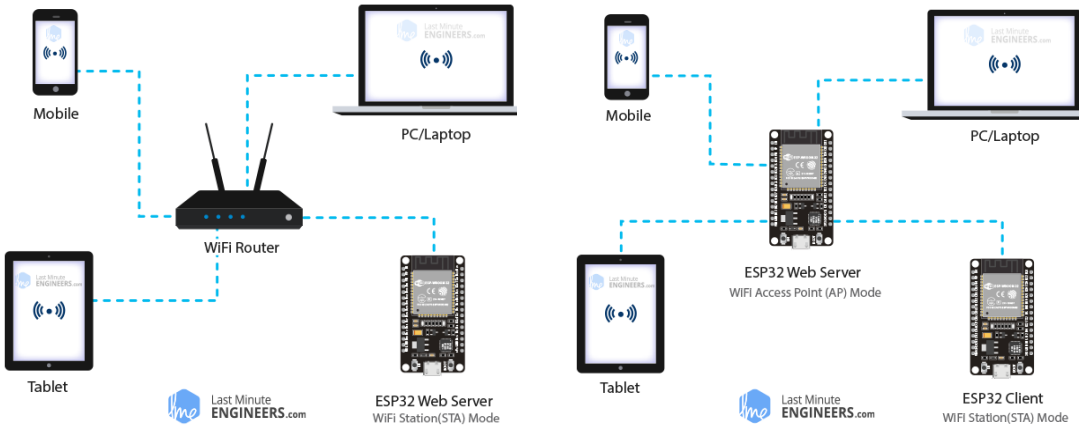

Figura 3.6 - Á esquerda: ilustração referente ao modo STA e á direita: ilustração referente ao modo AP [36]

Depois de efectuadas estas comparações, foi necessário ponderar o que iria ser realmente necessário na elaboração da plataforma, já referida neste capítulo. A grande desvantagem encontrada relativamente ao ESP8266 foi realmente a falta de GPIOs suficientes, e embora este tenha sido o factor fulcral para se escolher o ESP32 em vez do ESP8266, a verdade é que o ESP32 no seu todo traz mais vantagens para a nossa plataforma.

4

Implementação

Neste capítulo, descreve-se em mais detalhe a implementação do módulo de hardware e do módulo de software. Relativamente ao módulo de hardware, ir-se-á definir não só os componentes utilizados, o porquê das escolhas dos mesmos, mas também de certos problemas que foram encontrados e não estavam previstos durante a implementação deste modulo. É também mencionado neste capítulo o porquê da implementação de uma aplicação web durante o desenvolvimento do modulo de hardware.

Após a finalização desta primeira fase da implementação, passa-se para a implementação dos blocos da camada de software. Durante esta etapa, faz-se também a menção da criação de uma biblioteca de código dinâmico (DLL), necessária para incorporar as funções do cliente no programa em Java, que serviria para comandar o kit didáctico.

É importante referir que a plataforma que se pretende implementar serve para qualquer kit didáctico com a mesma complexidade. Aquando desta implementação estava-se a usar o kit presente na figura 4.2, linha de transporte e triagem de peças compacto.

4.1 Rectificação Tensão da Tensão de alimentação da placa

Começou-se com o dimensionamento do circuito que iria ser utilizado (figura 4.1). Este circuito teria de incluir uma parte de rectificação da tensão de alternada para contínua e de um transformador que conseguisse transformar da tensão da rede, que é de 230V 50Hz, para 24V contínuos, necessários na alimentação do kit e consequentemente na alimentação dos sensores e actuadores.

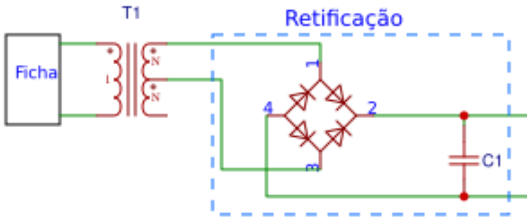

Figura 4.1 - Circuito exemplificativo da rectificação

No entanto, adicionar um circuito, mesmo que fosse o mais simples possível de forma a que o mesmo fizesse a rectificação da tensão iria ocupar espaço desnecessário e que poderia mais tarde fazer falta numa parte do circuito no qual não se pudessem fazer concessão como neste caso, por isso optou-se por utilizar um transformador que já tivesse incluído um modulo de rectificação.

Figura 4.2 - Kit didáctico utilizado durante a implementação da plataforma

4.2 Tratamento dos sensores

Seguidamente, teve de se idealizar como se iria proceder relativamente às diferenças de potencial dos 3V para os 24V e vice-versa.

Passar dos 24V para os ~3V é relativamente fácil, bastando aplicar o conceito de divisor de tensão. Foram assim obtidos os valores de 67KΩ e de 10KΩ para as resistências R1 e R2, respectivas ao divisor de tensão (figura 4.3).

No circuito físico final (Anexo 3), se se prestar atenção repara-se que não havia resistências de 67KΩ suficientes para os 10 portos que se queriam implementar. Isto deve-se a que as resistências de 67KΩ não tem um valor comum e na altura comprou-se o número certo das mesmas, quando se estragou uma durante a implementação, foi preciso improvisar com resistências de 68KΩ, que neste caso não fez qualquer diferença na tensão resultante, pois esta ainda estava dentro dos valores de funcionamento do dispositivo IoT, Esp32.

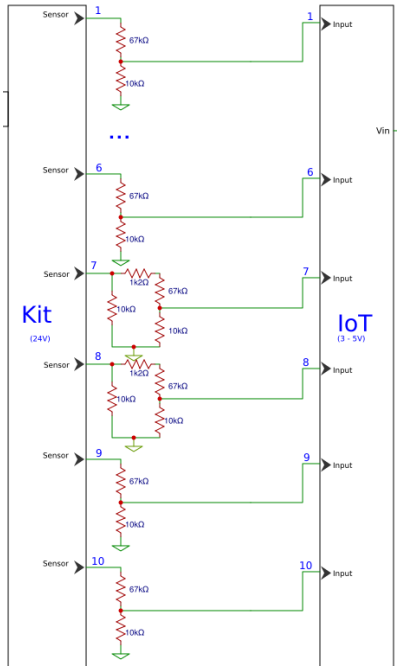

Figura 4.3 - Parte do circuito relativa ao divisor de tensão (sensores)

É possível ainda verificar que nos sensores 7 e 8, ilustrados na figura 4.4, foi necessário adicionar-se mais duas resistências, uma de 10KΩ e outra de ~1KΩ. Isto deveu-se ao facto de estes sensores corresponderem aos sensores que verificam se existe uma peça a passar em frente ao Cilindro B e Cilindro C, respectivamente, e de nestes sensores existir um pequeno led figura 4.5.

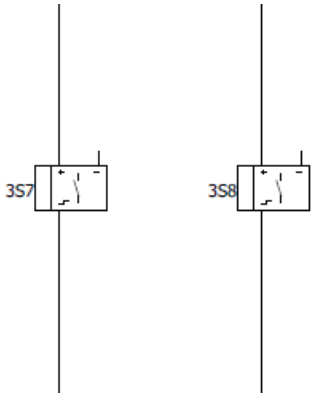

Figura 4.4 - Simbologia referente aos sensores 7 e 8 no manual de instruções do kit [37]

Este led polarizado com a configuração divisor de tensão não recebe corrente suficiente para ficar activo, foi por isso necessário a adição das duas resistências mencionadas acima.

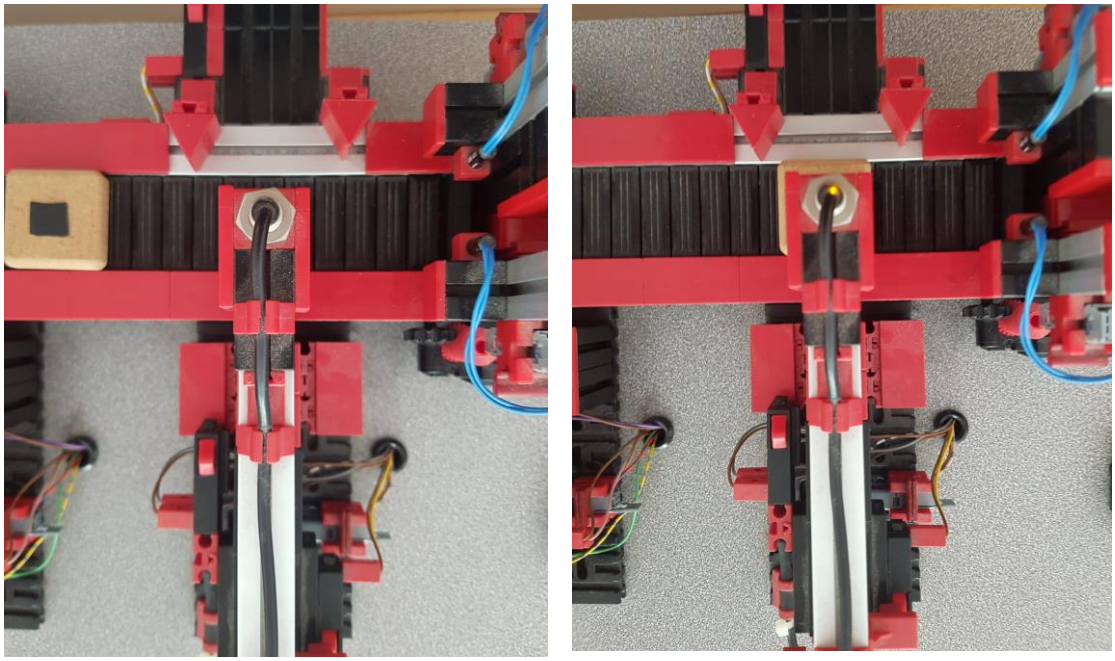

Figura 4.5 - Á esquerda: não foi detectada peça; Á direita: foi detectada peça (led ligado)

4.3 Tratamento dos actuadores

Relativamente á diferença de potencial dos ~3V para os 24V, poder-se-ia ter utilizado componentes "básicos" como díodos, transístores e resistências para implementar a diferença de potencial necessária. No entanto, o circuito iria tomar proporções maiores do que as desejadas para este projecto. De forma a

simplificar o circuito, preferiu-se usar um optocoupler (figura 4.6 – lado direito), que faz uso de um foto-díodo e de um transístor em configuração de Darlington para proporcionar a tensão final necessária.

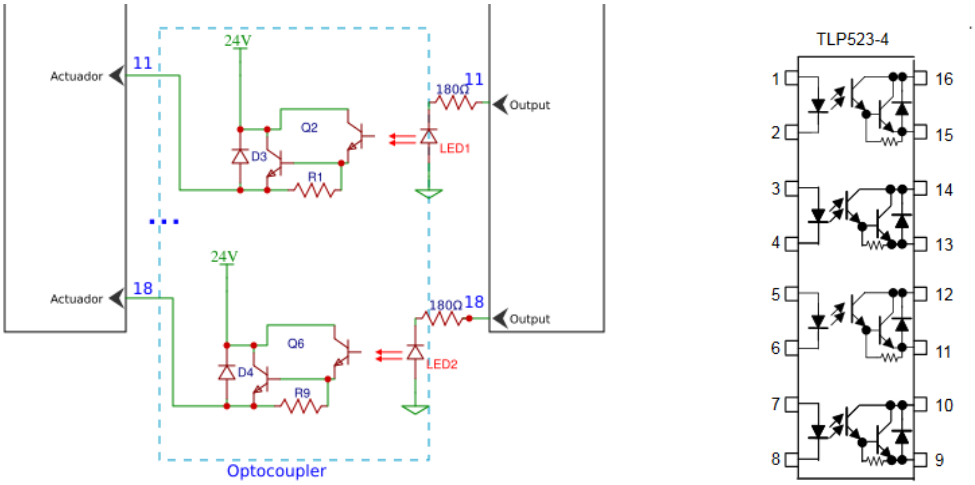

Figura 4.6 - Parte do circuito relativa a implementação com o optocoupler (esquerda) e esquemático do optocoupler (usado neste projecto) da Toshiba TLP523-4 (direita)

Nesta parte da implementação, devido a estarmos a utilizar um componente "externo", o *optocoupler,* foi importante ter em consideração vários factores, como por exemplo a tensão no foto-díodo (VF) não exceder o 1.3V.

Sabia-se que o dispositivo IoT estaria a trabalhar a ~3V, e devido a isso foi necessário dimensionar e adicionar a resistência RQ vista na figura 4.6 de 180Ω.

Havia outros factores que foram tidos em consideração antes de se poder testar o circuito com segurança, como por exemplo a corrente que passa no co-lector (IC) em momento algum poder exceder os 40mA ou a tensão de alimenta-ção(VCC) não poder exceder os 24V e por fim a corrente que passa no foto-dí-odo(IF) não ultrapassar os 20mA, embora isto tenha sido tido em conta no passo a cima. Por fim, a dissipação de energia era um factor que poderia também ser um entrave á implementação deste circuito. No entanto, isso não se verificou já que os valores estariam todos abaixo dos valores recomendados pelo fabricante,

tanto para a dissipação de energia do colector como a dissipação total do circuito.

4.4 Regulação de Tensão

Por fim (na parte referente ao hardware), ficava a faltar a alimentação do dispositivo IoT já que até este ponto a alimentação era realizada utilizando um cabo USB que fazia a ligação ao computador. Embora esta ligação ainda fosse necessária até á realização, com sucesso, da parte da implementação referente ao software, neste ponto era necessário ponderar como se iria proceder á alimentação deste componente quando findo toda a implementação.

Foi escolhido utilizar um regulador de tensão, o LM7805, que ao receber uma tensão de 24V, adquirida anteriormente, e com a ajuda de dois condensadores de 10uF, regula a tensão para os 5V, tensão máxima de alimentação do Esp32 (Figura 4.7).

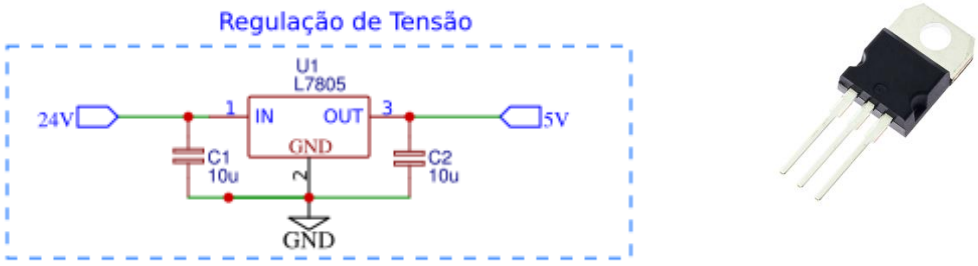

Figura 4.7 - Á esquerda: parte do circuito referente á regulação de tensão e á direita: LM7805

Ficou-se assim com o circuito total, como presente na figura 4.8. É possível encontrar-se nos anexos, uma versão do circuito final feito no software EAGLE, possibilitando assim uma mais fácil passagem para PCB no futuro.

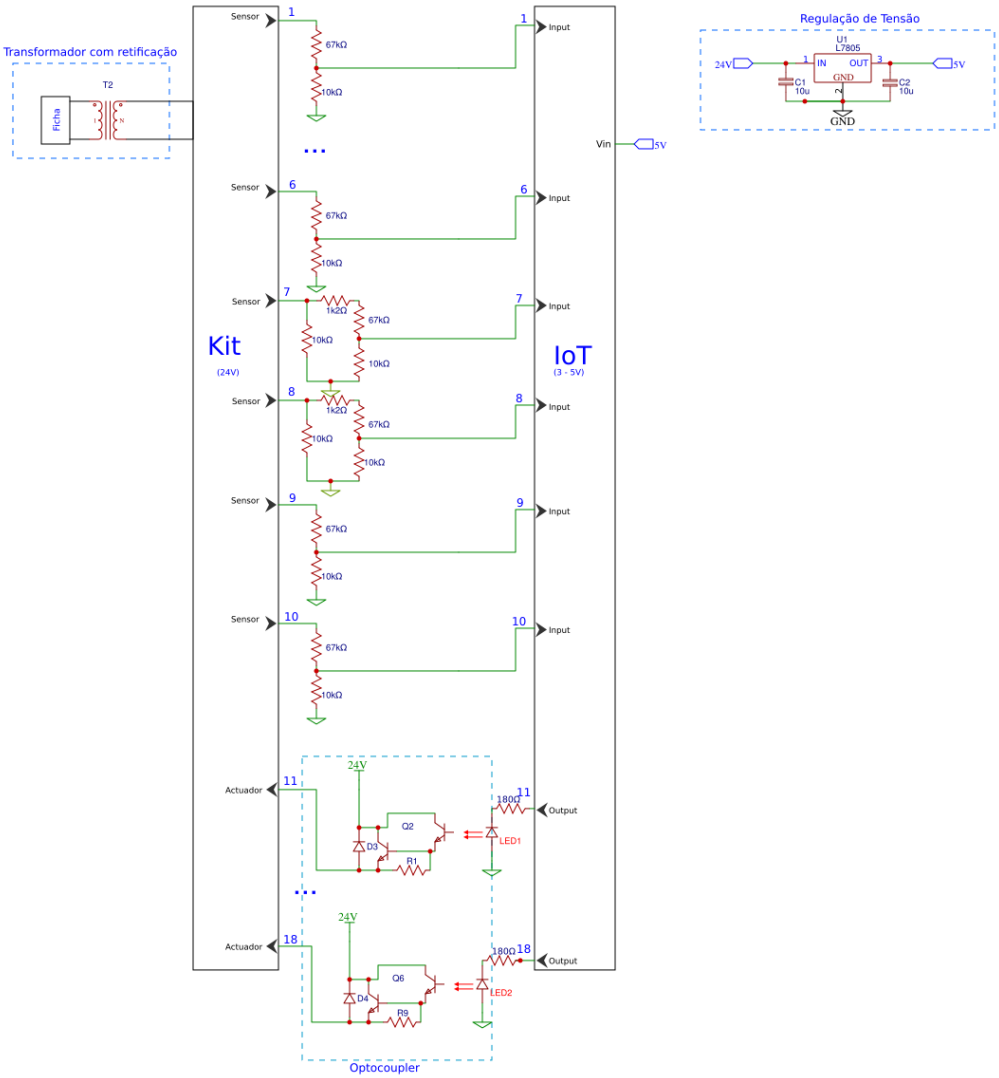

Figura 4.8 -Circuito do modulo de hardware completo

Depois de finalizada a implementação do circuito acima descrito, fez-se um pequeno programa que incluía a preparação do componente de Wifi incluído no Esp32 e posteriormente a criação de uma web application (figura 4.9) que permitiria o controlo do kit didáctico usando a placa Esp32. Esta aplicação web serviria apenas para se verificar se os actuadores conseguiriam ser todos activos quando ligados ao Esp32, e a partir daí se conseguiríamos ler os sensores com os valores esperados. Ficando assim concluído o objectivo definido para a parte relativa ao hardware.

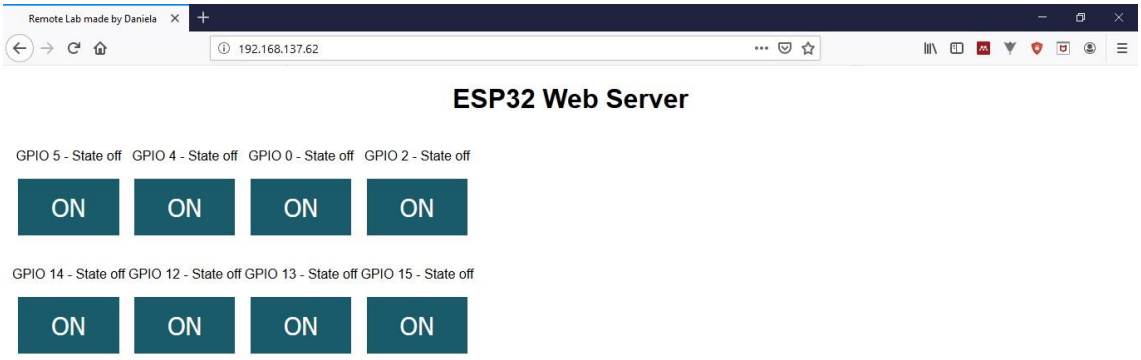

Figura 4.9 – Aplicação web desenvolvida para testar os actuadores da camada de hardware

Findo a implementação da parte relativa ao hardware, restava apenas a implementação da parte de software. Como já vem sendo referido esta fase do projecto implica a implementação da ligação entre o aluno e o kit didáctico, usando como intermediário o dispositivo IoT. É fulcral mencionar que tal como aconteceu na implementação da parte do hardware em que se vê os blocos como entidades individuais que estão todas implementadas no mesmo circuito, o mesmo acontece com a parte da implementação que se segue: embora sejam blocos separados a maioria deles está na mesma "folha" de código e à vista desarmada, se se olhar para o código implementado, não é possível ver onde começa um bloco e acaba o outro.

É importante mencionar que todos os kits disponíveis na sala de aula tem como objectivo poderem ser acedidos via portos múltiplos, o que significa que existindo 3 portos possíveis, estes iram estar a representar 1 byte com 8 portos na realidade, ou seja, o porto[0] ={0,1,2,3,4,5,6,7} e assim sucessivamente.

Seguidamente começa-se com a parte da implementação, feita exclusivamente em software.

4.5 AP Mode

A criação do bloco "AP Mode" da arquitectura tinha como objectivo a implementação de um “access point” ou ponto de acesso que serviria como posição de partida para a interacção por parte dos utilizadores com o kit (figura 4.10).

```
WiFi.softAP(ssid, password);
WiFi.softAPConfig(local_ip, gateway, subnet);
```

Figura 4.10 - Código relativo á criação de um "access point" e posterior configuração

Para se configurar o ponto de acesso, é necessário fornecer um *ssid* e uma palavra pass (Figura 4.10). O *ssid* é o nome pela qual a rede irá ser conhecida, neste caso deu-se o nome de "ESP32" e a palavra pass será a chave necessária para o utilizador se conseguir conectar a essa mesma rede (figura 4.11 - Direita). São também necessários três parâmetros para configurar o ponto de acesso. Estes três parâmetros tomam todos a forma de endereços IP. Estes representam:

- local_ip – o endereço IP relativo ao ponto de acesso
- gateway – o endereço IP relativo ao gateway
- subnet – a máscara da sub-rede

O gateway e o subnet são informações essenciais para a configuração de qualquer dispositivo que pretenda comunicar numa rede com protocolo IPv4, como é descrito em [38] e como se utilizaram valores genéricos para ambos os parâmetro, não se sentiu necessidade de aprofundar este tópico. Neste caso, o local_ip utilizado foi: *http://192.168.137.194/*.

Sem a criação deste bloco, continuaria a ser possível a ligação com o IoT, conforme descrito no capítulo 3, só que teria de se definir o modo de ligação como STA (Explicado na secção 3.6.1).

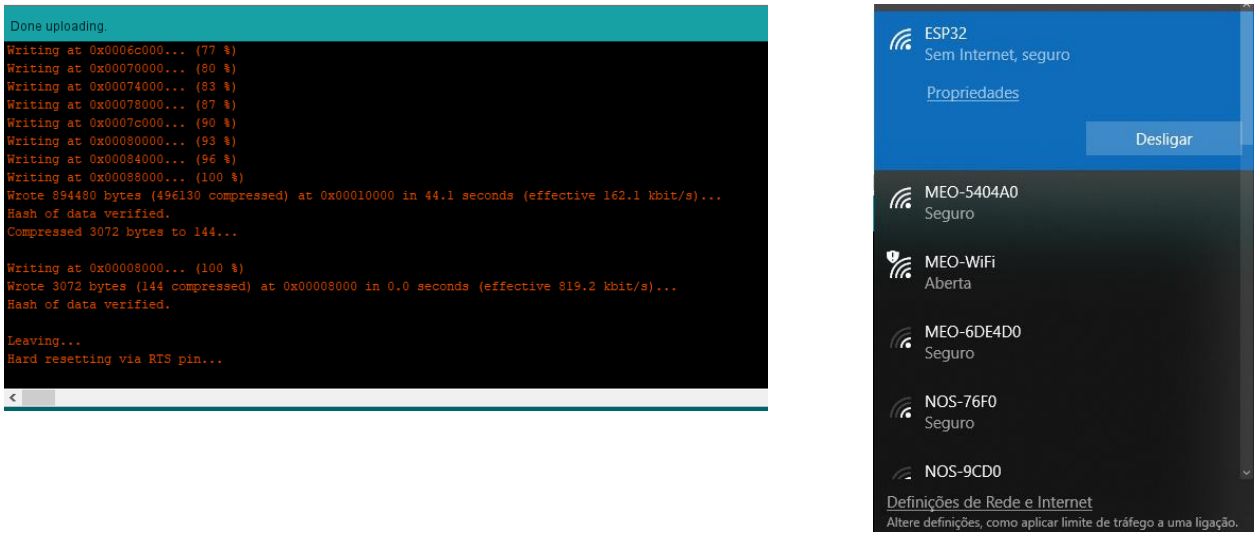

Figura 4.11 - Á esquerda: Imagem ilustrativa de um upload feito com sucesso do código para o dispositivo IoT e á direita: representação da criação do ponto de acesso e posterior ligação ao mesmo

Por fim, na figura 4.11, está ilustrada a criação com sucesso do ponto de acesso, como é possível verificar pela figura do lado direito. No lado esquerdo encontra-se o carregamento efectuado com sucesso do código mencionado anteriormente para o dispositivo IoT.

4.6 Web Server

A criação de um servidor web é o que irá permitir ao ESP32 receber os pedidos proveniente do cliente. A criação deste servidor quando estamos a lidar com um dispositivo IoT, resume-se á linha de código seguinte ilustrada na figura 4.12.

```
WebServer server(80);
```

Figura 4.12 - Criação do servidor web

Pode-se até denominar como inicializar o servidor em vez de fazer a criação do mesmo. Seguidamente (Figura 4.13), encontra-se a instrução relativa á

informação a ser mostrada aquando do carregamento inicial da mesma, usando o seguinte url: "*http://192.168.137.194/*"

```
server.on("/", handle_OnConnect)
```

```
void handle_OnConnect() {
  server.send(200, "text/html", SendHTML())
}
```

Figura 4.13 - Código relativo á instrução da página inicial

A página inicial do servidor web do ESP32 encontra-se ilustrada na Figura 4.14. Esta página serve principalmente para que se consiga verificar se o servidor está em funcionamento. No futuro, poder-se-á adicionar mais componentes a esta página, como por exemplo a criação de um menu de configuração do access point ou até mesmo a configuração dos portos consoante o kit a ser utilizado, entre outros.

Figura 4.14 - Página inicial do servido web

4.7 Interacção Kit

Uma das partes mais sensíveis durante toda a implementação encontra-se neste bloco. Este bloco tem como principal objectivo tratar das instruções que se encontram nas mensagens enviadas pelo cliente, como por exemplo: *"http://192.168.137.194/createDO?porto=2"*.

```
void handle_createDO() {
  Serial.println("Handle_CreateDO");

  String porto = server.arg(0);
  int *portos_temp;

  server.send(200, "text/plain", "Entrou a mensagem de Create DO");

  Serial.println("Porto: ");
  Serial.println(porto);

  portos_temp = retorna_portoMultiplo(porto);

  for (int i = 0; i < 8; i++) {
    if (portos_temp[i] != 0) {
      pinMode(portos_temp[i], OUTPUT);
    }
  }
}
```

Figura 4.15 - Código relativo á recepção da mensagem do cliente

É possível verificar, na figura 4.15, que quando se recebe uma mensagem, qualquer que seja o seu tipo, ir-se-á sempre "descodificar", em primeiro lugar, o porto múltiplo em portos singulares e depois é que se toma a acção necessária, seja criar um pino do tipo output (como exemplificado na figura 4.15) ou ler o valor de um porto especifico.

Na implementação deste bloco tomou-se especial atenção pois era bastante fácil conseguir-se danificar o ESP32 com uma instrução errada, neste caso por exemplo se houvesse uma tentativa de definir um porto que vinha de origem do tipo "output" para um pino do tipo "input", haveria possibilidade de se danificar aquele pino, podendo torna-lo inutilizável.

4.8 Interacção Cliente

Este bloco da arquitectura é especialmente importante pois tem como objectivo receber as mensagens vindas do cliente e quando há necessidade, também, tem o dever de enviar as respostas.

```
server.on("/createDI", handle_createDI);
server.on("/createDO", handle_createDO);
server.on("/writeDigitalU8", handle_write);
server.on("/readDigitalU8", handle_read);
```

Figura 4.16 - Código relativo às mensagens provenientes do cliente

Neste pedaço de código (figura 4.16), podemos visualizar que dependendo do tipo de mensagem proveniente do cliente, irá ser executada uma função diferente, podendo ou não ser necessário enviar uma resposta de volta para o cliente. Quando há necessidade de enviar mensagens de volta para o cliente, é executada a instrução ilustrada na figura 4.17. Neste momento, só há necessidade de informar o cliente de uma resposta quando o mesmo anteriormente pediu para ler o valor dos sensores de um dado porto.

```
server.send(200, "text/plain", resultado_enviar);
```

Figura 4.17 - Código relativo ao envio de mensagens para o cliente

Para terminar este bloco, só falta exemplificar o tipo de código executado no lado do cliente referente às mensagens que são enviadas para o IoT.

```
void create_DO(uint32_t porto) {
    WinHttpInit();

    int recvlen = 0;
    wchar_t size[50] = L"";
    char* pszPostData = "";
    char recv[10240] = { 0 };

    wchar_t* url = (wchar_t*)malloc(42 * sizeof(wchar_t));

    wchar_t* ptr = (wchar_t*)malloc(2 * sizeof(wchar_t));
    wcscpy(ptr, L"");

    int port_transf = porto;

    swprintf_s(ptr,3, L"%d", port_transf);

    wcscpy(url, L"http://192.168.137.194/createDO?porto=");

    wcsncat(url,ptr,2);

    //wprintf(L"%ls\n", url);
    recvlen = SendHttpRequest(L"GET", url, pszPostData, strlen(pszPostData) + 1, recv, FALSE);
}
```

Figura 4.18 - Código do lado do cliente

Na figura 4.18, é possível verificar-se que quando é chamada a função "create_DO", vai-se "juntar" todas as informações necessárias para identificar o tipo de mensagem e o porto respectivo num único *url* que depois será enviado para o servidor, ficando algo parecido com *"http://192.168.137.194/createDO?porto=2"*. Esta função é homologa de todas as outras que enviam mensagens para o IoT, só mudando o tipo de mensagem, o porto e eventualmente se se quiser modificar algum valor no kit.

4.8.1 Interface

Bloco incluído no modulo de "Interacção Cliente", que tem como principal objectivo comutar o modo de funcionamento da plataforma de simulador para kit ou sistema físico real. Para tal, no início deste ficheiro temos uma macro `#define _USE_REAL_KIT_` que se estiver comentada, aquando da compilação deste ficheiro, irá possibilitar a interacção com um simulador do kit real (figura 4.19), caso contrário irá ser usado o kit real.

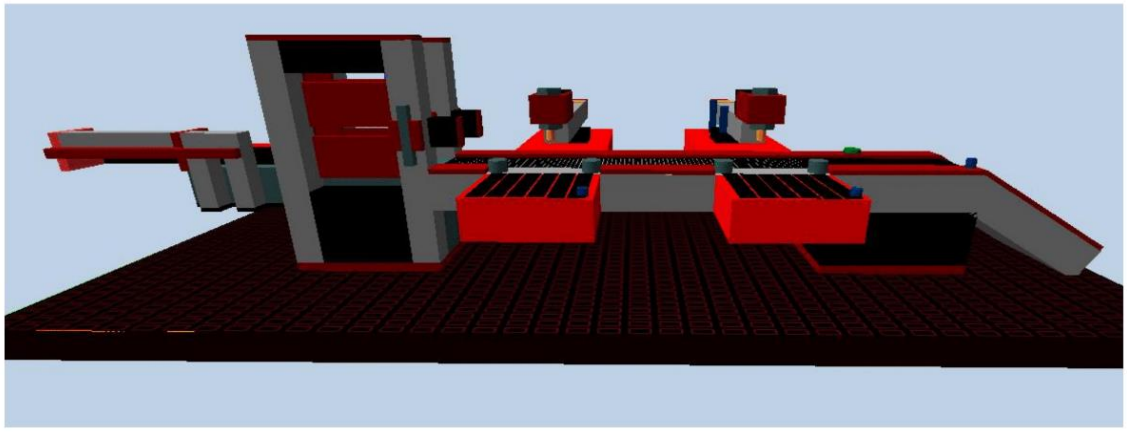

Figura 4.19 - Ambiente de simulação do kit

Esta macro (ilustrada na figura 4.20) possibilitará que sempre que forem chamadas as funções de interacção com o kit, seja ele real ou simulado, ir-se-á ter uma parte do código como o seguinte:

```
void create_DO_channel(uInt32 port)
{
#ifdef _USE_REAL_KIT_
    create_DO(port);
#else
    sim_create_DO_channel(port);
#endif

}
```

Figura 4.20 - Código relativo á mudança de interface

Possibilitando assim o uso das funções adequadas para cada um dos ambientes.

Por último, falta só mencionar que foram criados dois ficheiros: um referente a todo o código necessário para a implementação do servidor - WebServerEsp32APMode.ino e o segundo, referente a implementação do cliente – ClienteAPmode.c.

4.9 Construção da DLL

De forma a possibilitar a implementação de um programa em Java, fazendo uso do acesso remoto implementado a acima, foi necessário criar-se uma DLL. Para tal, foi usado o mecanismo de interoperabilidade JNI, *Java Native Interface*, o qual permite que um programa em Java consiga utilizar recursos que estão em níveis mais abaixo, incluindo o de hardware. Desta forma, as funções de interacção com o kit didáctico, e posteriormente o acesso remoto, que estão programadas em C, podem ser chamadas a partir do código em Java.

Numa primeira fase, foi necessário configurar o sistema, mais especificamente uma variável de ambiente do sistema, denominada *Path* que iria conter o caminho para a pasta onde estaria instalada a versão mais recente do java. Depois foi necessário criar um projecto em Visual Studio que fosse do tipo DLL, denominado Hardware onde se utilizou os especificadores JNIEXPORT e JNICALL, para se implementar os métodos (figura 4.21) que iriam ser usados em Java, mas com as funções implementadas em C.

```
JNIEXPORT void JNICALL Java_Hardware_create_1di(JNIEnv *, jobject, jint port)
{

    create_DI_channel(port);
}
JNIEXPORT void JNICALL Java_Hardware_create_1do(JNIEnv *, jobject, jint port)
{
    create_DO_channel(port);
}
JNIEXPORT void JNICALL Java_Hardware_write_1port(JNIEnv *, jobject, jint port , jint value)
{
    WriteDigitalU8(port, value);
}
JNIEXPORT jint JNICALL Java_Hardware_read_1port(JNIEnv *, jobject, jint port)
{
int v = ReadDigitalU8(port);
return(v);
}
}
```

Figura 4.21 - Métodos implementados na DLL

Assim que os métodos estivessem implementados, era necessário incluir certas directorias na solução de forma a poderem ser usados os métodos mencionados em cima, nomeadamente a directoria onde se encontra o ficheiro *jni.h.* Para este caso, também foi necessário incluir as directorias onde se encontravam

os ficheiros: interface.c, já reescrita com as funções criadas para o acesso remoto ao kit, o mongoose.c , o ClienteAPmode.c e os seus respectivos headers (.h).

Depois de tudo isto ser incluído foi possível fazer o build da solução. Este quando executado com sucesso devolve a directoria(figura 4.22) onde foi guardada a DLL.

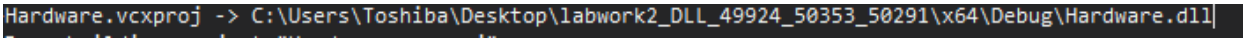

Figura 4.22 - Directoria onde é guardada a DLL

5

Resultados obtidos

5.1 Verificação dos requisitos funcionais

Seguidamente, apresentam-se os testes efectuados de modo a validar os requisitos funcionais definidos no capítulo 3. Algumas destas validações foram feitas durante a implementação (capítulo 4). No entanto, pretende-se reproduzir esses resultados mais uma vez nesta secção.

Nas interacções com o kit, ir-se-á usar o kit de uma linha de transporte e triagem de peças, apresentado na figura 5.1.

Figura 5.1 - Kit real - linha de transporte e triagem de peças compacto

5.1.1 Estabelecer ligação com o kit real

Para se verificar se a ligação com o kit didáctico era estabelecida, bastou enviar-se um pedido para que um actuador se movimentasse. Neste caso, preferiu-se movimentar a passadeira rolante (figura 5.2), pois era o actuador que só precisava de uma instrução para activar e outra para se parar, sem haver preocupações com chegar a limites de hardware e afins.

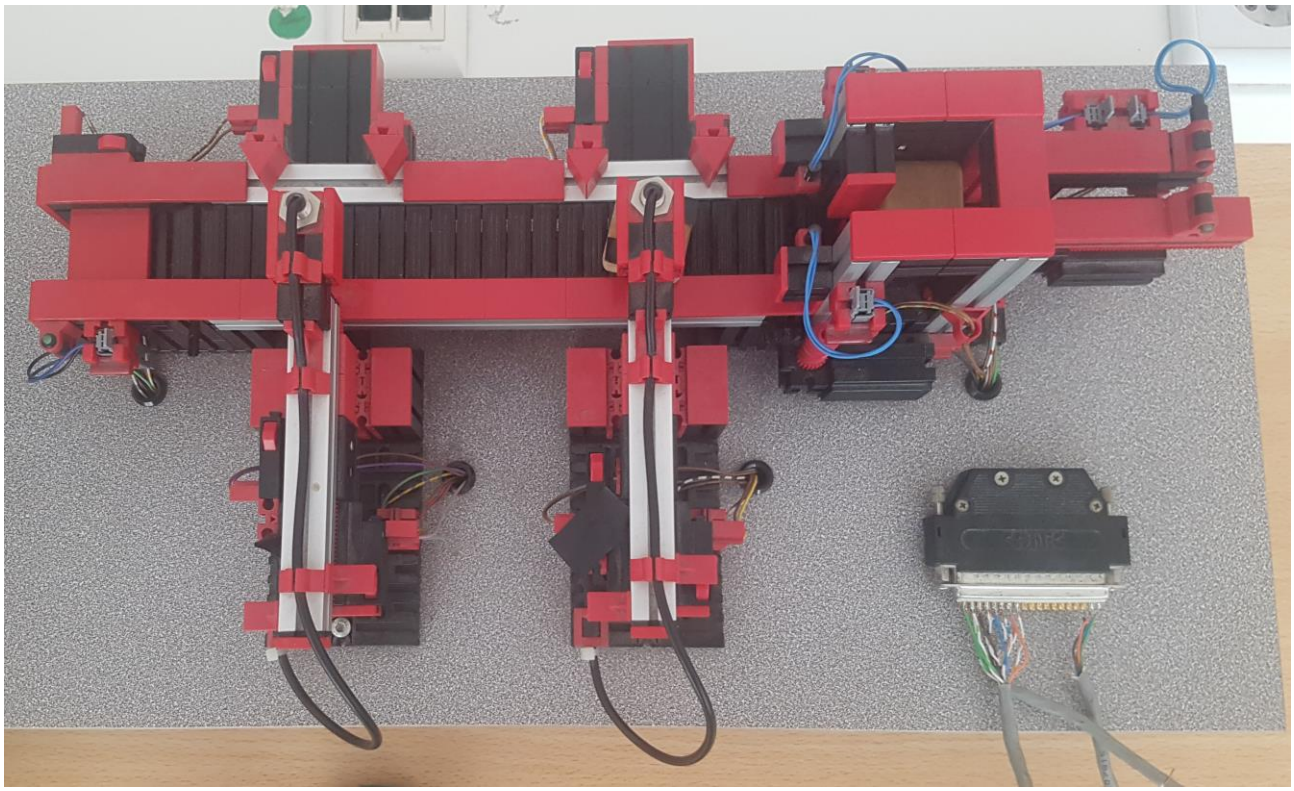

Figura 5.2 - Kit real durante verificações

O mesmo não aconteceria se se movimentasse qualquer um dos 3 cilindros existentes neste kit. Para tal usou-se a seguinte instrução no browser:

http://192.168.137.194/writeDigitalU8?porto=2&valor_port=4.

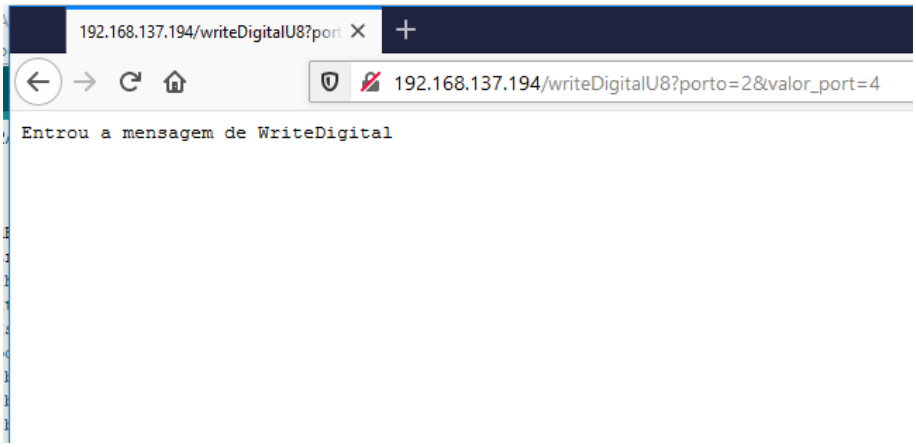

Figura 5.3 - Mensagem enviada com sucesso para o kit

A instrução foi completa com sucesso (figura 5.3), onde se viu depois a passadeira rolante em movimento. Seguidamente, para se cessar o movimento da mesma, bastou enviar a seguinte instrução:

http://192.168.137.194/writeDigitalU8?porto=2&valor_port=0.

5.1.2 Estabelecer Access Point

Como já tinha sido mencionado anteriormente no capitulo 4, o Access Point é o ponto de partida para que se consiga interagir com o kit usando a plataforma implementada nesta dissertação e o mesmo deve estar disponível a partir do momento em que é executado com sucesso o upload do código para o ESP32 (figura 5.4).

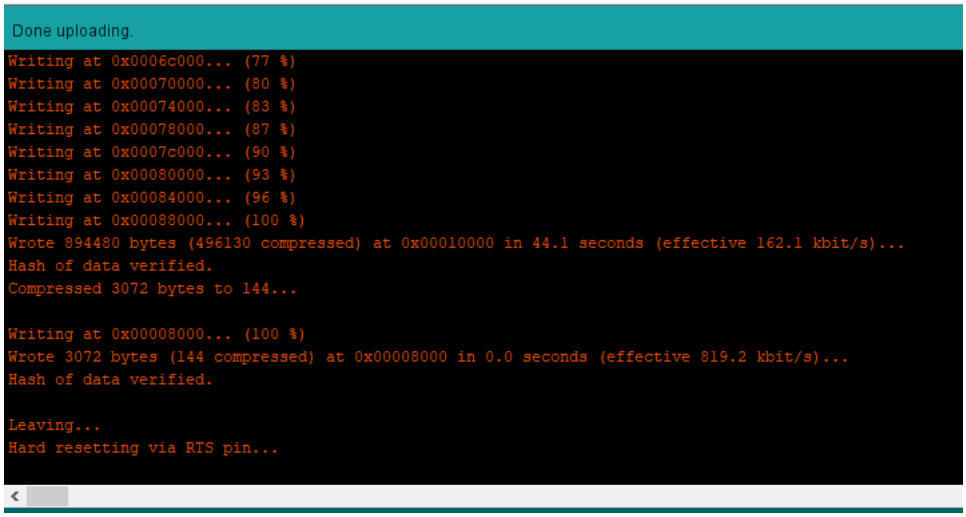

Figura 5.4 - Upload finalizado com sucesso

Figura 5.5 - Access Point disponível

Depois de estar disponível, o mesmo tem de permitir uma ligação a sua rede, o que é visível na figura 5.5.

Ficando assim este requisito como verificado.

5.1.3 Executar diferentes operações de acesso ao kit

O requisito número 3, executar diferentes operações de acesso ao kit, pretende verificar que se pode executar as 3 interacções necessárias com o kit didáctico:

1. Definição do tipo de portos
2. Modificar os valores do kit de forma a activar os actuadores
3. Ler os valores dos sensores

A definição dos portos consiste em dizer ao kit que o porto 1 é um porto de input e o porto 2 é um porto de output, cada um com 8 bits. Para se definir o tipo de porto usa-se uma das seguintes funções, dependendo do tipo de porto que se quer criar: *handle_createDI* ou *handle_createDO*, estas funções são acessíveis enviando o seguinte request: *http://192.168.137.194/createDI?porto=1* ou

http://192.168.137.194/createDO?porto=2 e estas foram executadas com sucesso.

Modificar os valores do kit, permitindo a activação de um ou mais actuadores envolve a actuação da função *handle_write,* que é acessível enviando um "request" como o seguinte: *http://192.168.137.194/writeDigitalU8?porto=2&valor_port=6.*

Com o request definido acima, pede-se ao kit real que active o actuador referente a passadeira rolante e ao movimento para a frente do cilindro A, visualizado na figura 5.6.

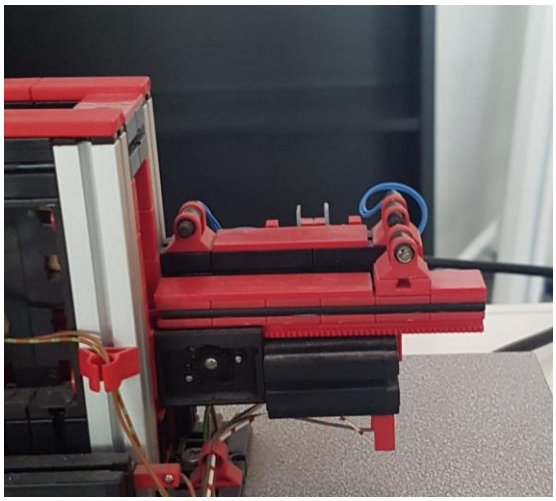

Figura 5.6 - Cilindro A na posição 0

Depois de realizado este "request" com sucesso, e de forma a pedir uma paragem dos actuadores que foram activos em cima, é necessário enviar um request de paragem que tem a seguinte forma:

http://192.168.137.194/writeDigitalU8?porto=2&valor_port=0.

Por fim, e de forma a finalizar a validação deste requisito, só falta averiguar os estados de um certo porto. Para tal, usa-se a função *handle_read* que é activa fazendo um request com a seguinte forma: *http://192.168.137.194/readDigitalU8?porto=2*. Aquando da leitura bem-sucedida do estado do porto, é devolvido um número decimal que representa quais os portos que estão a 0 e a 1.

Tendo sido todas as interacções concluídas com sucesso, pode-se concluir que este requisito foi alcançado com sucesso.

5.1.4 Receber e enviar informação

Tanto este requisito de receber informação como o de enviar informação estão internamente ligados, já que por exemplo a recepção de informação do servidor, pode ser vista como o envio de informação por parte do cliente e vice-versa. Devido a isso, vai-se efectuar a verificação destes dois requisitos em simultâneo.

É importante mencionar que a forma como se está a utilizar esta plataforma, o servidor só envia informação numa situação muito especifica, que é quando há necessidade de ler os estados de um ou mais portos. Para que isso aconteça, o cliente tem de enviar um pedido de leitura (figura 5.7).

```
hardware.SafeReadPort(2);
```

Figura 5.7 - Pedido referente á leitura do estado de um porto

Que depois de recebido no servidor, este envia de volta em número decimal correspondente aos bits que estão activos no porto 2.

O cliente é a parte da nossa arquitectura que realmente faz um número considerável de requisitos de informação, sendo, portanto, quem envia mais informação. O servidor por sua vez é quem recebe mais informação.

Ambos estão a cumprir os requisitos funcionais de receber e enviar informação quando esta é requisitada.

5.1.5 Permitir troca entre plataforma IoT e Simulador

Este requisito é extremamente importante, já que permite a mudança de paradigma entre o modo virtual e remoto. Esta troca é feita usando um ficheiro denominado interface, onde está definida uma macro (figura 5.8) que quando está activa, permite que todas as instruções enviadas pelo cliente sejam executadas no kit didáctico, caso contrário irá fazer a ligação ao simulador e executar as instruções no mesmo.

```
'#define _USE_REAL_KIT_
```

Figura 5.8 - Macro utilizada para trocar entre o simulador e a plataforma IoT

A troca entre plataformas é bastante simples de utilizar, necessitando só que haja uma activação, ou não, da macro e posteriormente que se volte a compilar a DLL (explicada no capítulo 4).

5.1.6 Plataforma a funcionar sem quebras

Este requisito é fulcral para toda a solução implementada, já que sem ele a solução torna-se banal e sem qualquer interesse para o problema que nos estamos a propor resolver. Isto deve-se a que esta plataforma está idealizada para ser algo que está sempre pronto a ser usado, e caso isso não aconteça pode trazer constrangimentos, não realizando assim o que se propõe. Devido a isso, foram realizadas várias verificações de modo a tentar perceber quais os limites desta plataforma. Não foi, no entanto, possível verificar qualquer quebra no funcionamento. Mesmo ao fim de um dia inteiro ligada, a plataforma esteve sempre disponível a efectuar qualquer pedido.

5.2 Validação por cenários

De forma a executar a validação por cenários criou-se um problema hipotético que teria de ser resolvido usando a plataforma IoT em contraste com a abordagem que tem vindo a ser utilizada até agora (paradigma antigo).

O desafio consistia em: receber várias peças, independentemente do tipo e alternadamente colocá-las na estação 1 ou na estação 2 (figura 5.9).

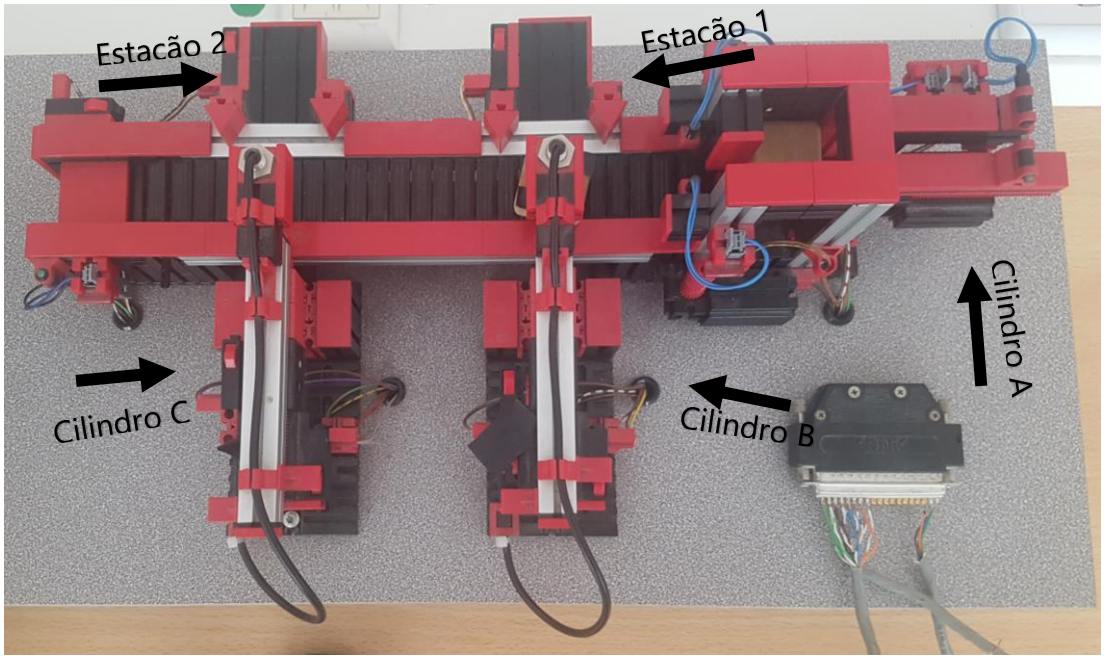

Figura 5.9 - Kit real durante validação por cenários

Existem 2 cenários, conforme ilustrado na figura 5.10. O cenário A, que pretende exemplificar como é utilizado actualmente o kit e como é feita a sua ligação e o cenário B que pretende exemplificar a utilização da plataforma IoT implementada nesta dissertação.

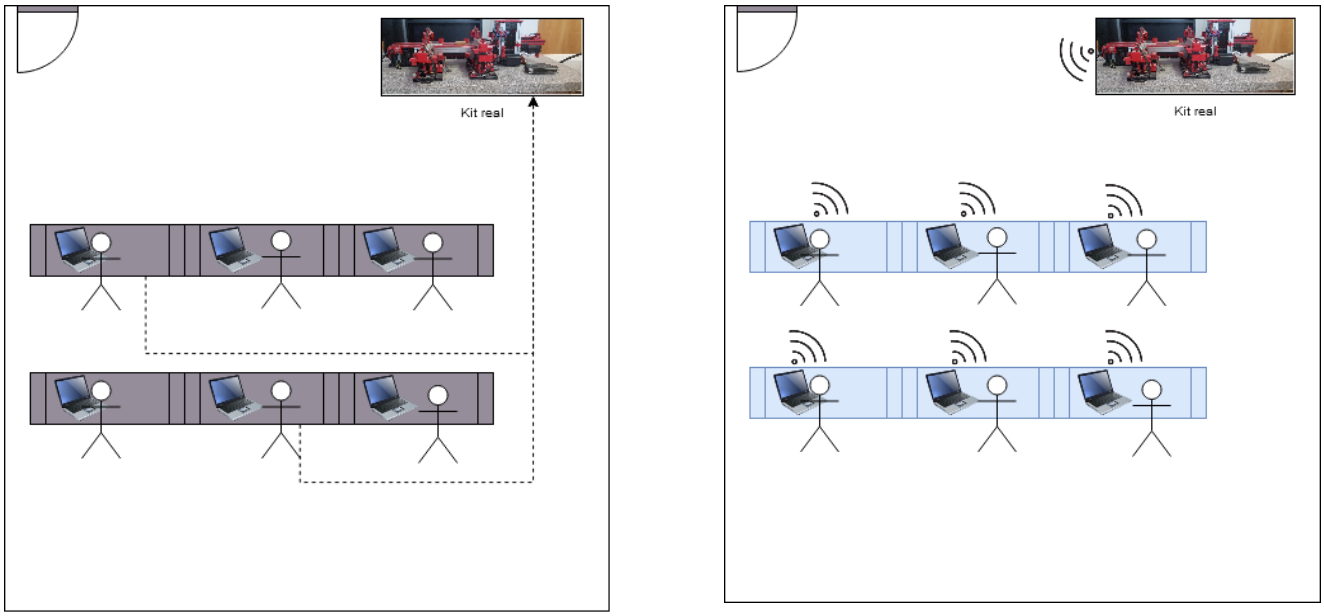

Figura 5.10 - Cenários definidos para a validação por cenários. Esquerda: cenário A; Direita: cenário B

5.2.1 Ambiente simulado

Numa primeira fase é implementado o programa (em Java) e testado no ambiente de simulação (figura 5.13). Este ambiente permite testes por módulos, algo que não é propriamente possível no ambiente real. Para tal ser possível, é necessário começar por criar a DLL, tendo em atenção que a macro `#define _USE_REAL_KIT_` (na Figura 5.11 encontram-se as diferenças relativas ao código nas duas abordagens de utilização do kit didáctico) tem de se encontrar comentada no ficheiro interface.c e compilar a mesma.

```
#ifdef _USE_REAL_KIT_
    char taskName[100];
    char portName[100];
    TaskHandle task_handle;
    sprintf(taskName,"task_di__%d",port);
    sprintf(portName,"Dev1/port%d",port);
    DAQmxCreateTask(taskName,&task_handle);
    DAQmxCreateDIChan(task_handle,portName,"",DAQmx_Val_Cha
    DAQmxStartTask(task_handle);
    tasks[port]=task_handle;
```

```
#ifdef _USE_REAL_KIT_
    create_DI(port);
```

Figura 5.11 - Código referente à interacção com o kit no cenário A (esquerda) e código de interacção com a plataforma IoT (cenário B) (direita)

Primeiramente, testa-se o movimento dos cilindros (figura 5.9) no simulador, Cilindro A - movimenta as peças para o tapete rolante, Cilindro B - empurra as peças para a estação 1 e o Cilindro C - empurra as peças para a estação 2.

Depois experimenta-se o tapete rolante e a detecção de peças, que são de 3 tipos (figura 5.12).

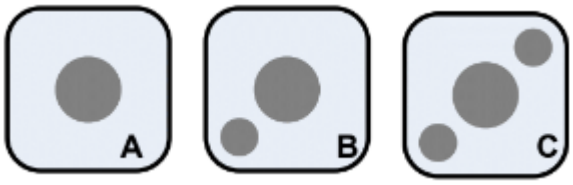

Figura 5.12 - Tipos de peças existentes na linha de montagem

Por fim, faltava só testar neste ambiente o decorrer integral do programa, assegurando que há detecção da peça e que alternadamente irá ser encaminha para

a estação correspondente, que neste caso será a estação 1 se for uma peça de número ímpar e para a estação 2 se for uma peça de número par.

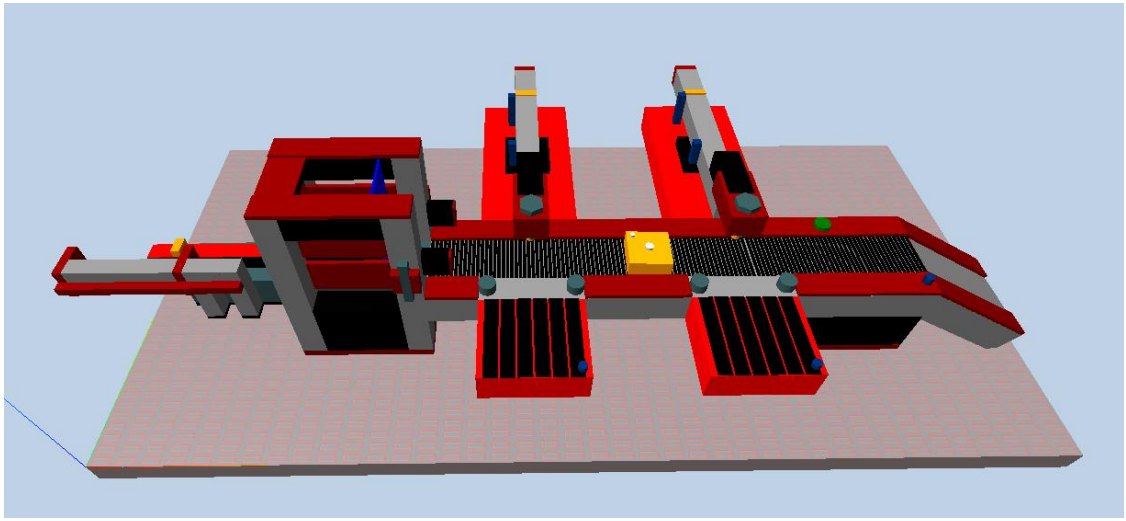

Figura 5.13 - Ambiente de simulação

5.2.2 Ambiente Real

5.2.2.1 Cenário A

Neste cenário, é necessário começar por fazer a ligação ao dispositivo da *National Instruments* (anexo 5) e fazer a instalação dos drivers. Para se fazer a ligação a este equipamento, o aluno tem de se deslocar até ao mesmo e esperar a instalação dos drivers.

Seguidamente, há a necessidade de se voltar a recompilar a DLL, desta vez com a macro `#define _USE_REAL_KIT_` e com as funções referentes á utilização do NiDAQ (figura 5.14), e incluí-la no programa implementado em Java.

Depois, é necessário abrir o *NI Device Monitor* e verificar que se consegue detectar o kit.

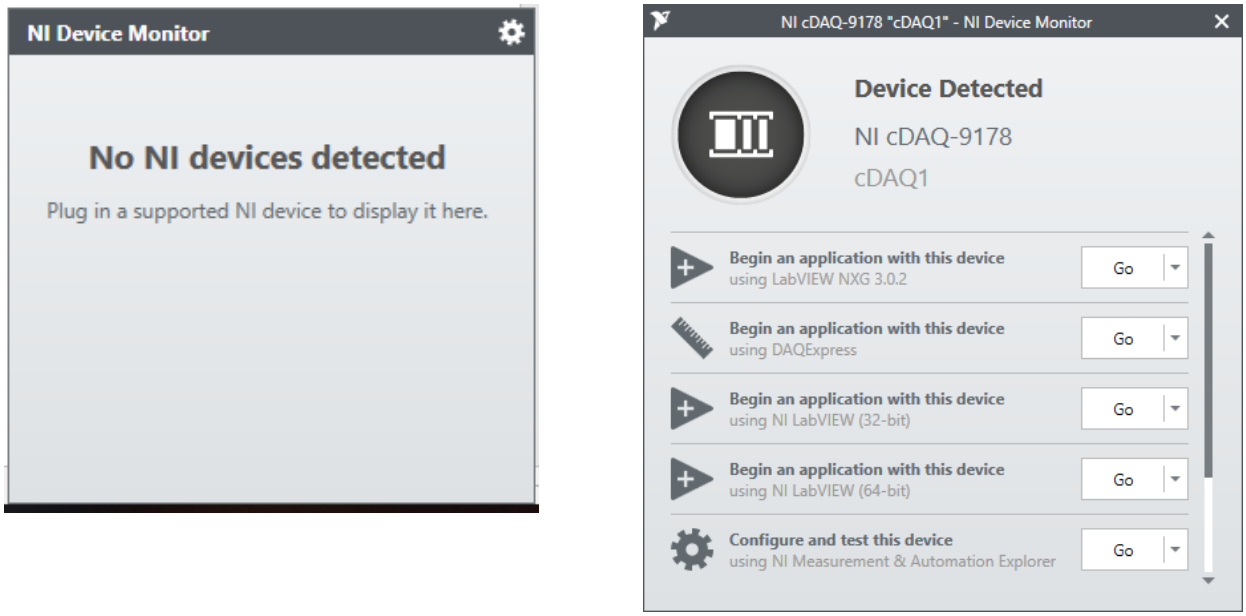

Figura 5.14 - *NI Device Monitor* software quando há um kit desconectado vs quando existe um kit conectado

A partir deste momento, foi possível confirmar a implementação do programa (em Java) usando o ambiente real (kit didáctico).

5.2.2.2 Cenário B

Neste cenário, onde é utilizada a plataforma IoT implementada durante esta dissertação, para se passar à utilização do ambiente real é necessário, analogamente ao cenário anterior, recompilar a DLL de forma a permitir macro `#define _USE_REAL_KIT_` (com as funções referentes a plataforma IoT (figura 5.11 - direita)).

Estabelecer-se a ligação ao ponto de acesso estabelecido pelo dispositivo IoT, neste momento denominado "Esp32". Para, tal o aluno só tem de se ligar à rede "Esp32" como se de uma rede wireless se tratasse.

A partir deste momento, é possível o aluno interagir com o kit real utilizando a plataforma IoT.

Findo a validação por cenários, é possível verificar que a utilização da plataforma IoT, ajuda a colmatar a falta de tempo que era recorrente durante as aulas de laboratório, já que a inclusão desta plataforma permite uma ligação instantânea ao kit real.

6
Conclusões e trabalho futuro

6.1 Conclusões

Todo o trabalho efectuado nesta dissertação tinha um objectivo bastante concreto, ajudar a colmatar as dificuldades que surgem aquando da massificação do ensino que se faz sentir neste momento. De uma forma mais particular, no ensino superior e nos cursos de engenharia, onde se vê cada vez mais a parte pratica destes cursos a não possibilitar uma aprendizagem no seu melhor, tanto por falta de tempo para aceder às experiências, como por falta de material disponível.

O que se propôs implementar foi uma plataforma IoT, que permitisse uma ligação instantânea aos kits didácticos nas salas de aula um pouco por todo o mundo. Concluiu-se que usar um dispositivo IoT e todo o modulo de hardware mais software que fazem parte da plataforma, descrita nesta dissertação para substituir a ligação física que agora existe, de forma a permitir aos alunos interagirem com o kit real. Esta tem bastantes vantagens, como o caso de a ligação ser, literalmente, instantânea e de dar a conhecer aos alunos tecnologias mais modernas do que as que tem vindo a ser usadas até agora. Permitindo assim concluir que a abordagem utilizada para combater as dificuldades que estão a surgir, devido ao excesso de alunos que entram no ensino superior foi bem-sucedida e

sem qualquer dúvida que vai em muito facilitar o trabalho tanto de alunos como de docentes.

Embora não mencionadas neste documento, foram implementadas outras soluções antes de se chegar à solução aqui proposta no capítulo 4. Estas soluções trouxeram uma enorme quantidade de aprendizagem o que possibilitou a solução final. A implementação feita com recurso aos *http requests*, é bastante mais simples e de uma versatilidade completamente diferente relativamente às implementações anteriores. Esta última implementação torna muito mais fácil reproduzir esta plataforma noutros kits didácticos que tenham o mesmo nível de complexidade, já que basta só fazer um mapeamento dos sensores e actuadores desses novos kits e modificar os portos no servidor web.

6.2 Trabalho futuro

Deixa-se algum trabalho futuro para ser executado como por exemplo a reprodução desta plataforma IoT implementada nesta dissertação noutros kits didácticos com a mesma complexidade e eventualmente a implementação de uma única plataforma IoT que seja versátil o suficiente para servir para todos os kits presentes numa sala de aula, desde que tenham o mesmo tipo de complexidade.

Relativamente à plataforma implementada nesta dissertação, existe a necessidade de a parte de hardware ser recriada em PCB, e na parte de software adicionar o acesso singular ao Kit via plataforma IoT, que é mencionado na arquitectura do sistema como requisito não-funcional.

Bibliografia

[1] S. Guri-Rosenblit, H. Šebková, and U. Teichler, "Massification and diversity of higher education systems: Interplay of complex dimensions," *High. Educ. Policy*, vol. 20, no. 4, pp. 373–389, 2007.

[2] D. J. Hornsby and R. Osman, "Massification in higher education : large classes and student learning," 2014.

[3] Fundação Francisco Manuel dos Santos, "PORDATA - Alunos matriculados no ensino superior: total e por área de educação e formação." [Online]. Available: https://www.pordata.pt/Portugal/Alunos+matriculados+no+ensino+superior+total+e+por+área+de+educação+e+formação-1026-8240. [Accessed: 25-Nov-2019].

[4] C. A. Jara, F. A. Candelas, S. T. Puente, and F. Torres, "Hands-on experiences of undergraduate students in Automatics and Robotics using a virtual and remote laboratory," *Comput. Educ.*, vol. 57, no. 4, pp. 2451–2461, 2011.

[5] C. Proceedings, "in the 21st Century in the 21st Century," *Small*, vol. 41, no. April, pp. 229–246, 2004.

[6] S. Downes, "E-learning 2.0," *Self Published*. 2013.

[7] Gustavo Alonso, Fabio Casati, Harumi Kuno, "Web Services: Concepts, Architectures and Applications," *Springer-Verlag Berlin Heidelb.*, vol. 9081, no. Chapter 5, 2004.

[8] Uddi Org, "UDDI Executive White Paper," *Int. Bus.*, 2001.

[9] G. Gray and K. O'Connor, "Web Services Technology Infrastructure," *ITB J.*, vol. 3, no. 2, 2015.

[10] W. Services and C. Architecture, "Web Services Conceptual Architecture (WSCA 1.0)," *Architecture*, vol. 5, no. May, pp. 6–7, 2001.

[11] Z. H., H. A., and M. M., "Internet of Things (IoT): Definitions, Challenges and Recent Research Directions," *Int. J. Comput. Appl.*, vol. 128, no. 1, pp. 37–47, 2015.

[12] Y. S. Chen and J. S. Hong, "Intelligent Device-to-Device Communication in the Internet of Things," *IEEE Syst. J.*, vol. 7, no. 1, pp. 77–91, 2013.

[13] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," *J. Comput. Commun.*, vol. 03, no. 05, pp. 164–173, 2015.

[14] b105, "PFC: Design and implementation of a dynamic Wireless Sensor Network | B105 lab." [Online]. Available: http://elb105.com/pfc-design-and-implementation-of-a-dynamic-wireless-sensor-network/. [Accessed: 27-Oct-2019].

[15] IBM, "IBM Watson IoT Platform - Overview." [Online]. Available: https://www.ibm.com/pt-en/marketplace/internet-of-things-cloud. [Accessed: 20-Nov-2019].

[16] E. Koutroulis and K. Kalaitzakis, "Development of an integrated data-acquisition system for renewable energy sources systems monitoring Koutroulis, E. and Kalaitzakis, K. Renewable Energy, 2003, 28, (1), 139–152," *Fuel Energy Abstr.*, vol. 44, no. 3, p. 163, 2003.

[17] R. Abbasi *et al.*, "The IceCube data acquisition system: Signal capture, digitization, and timestamping," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 601, no. 3, pp. 294–316, 2009.

[18] J. A. Aguilar *et al.*, "The data acquisition system for the ANTARES neutrino telescope," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 570, no. 1, pp. 107–116, 2007.

[19] D. Lowe, S. Murray, E. Lindsay, and D. Liu, "Evolving remote laboratory architectures to leverage emerging internet technologies," *IEEE Trans. Learn. Technol.*, vol. 2, no. 4, pp. 289–294, 2009.

[20] J. M. Sebastián, D. García, and F. M. Sánchez, "Remote-access education based on image acquisition and processing through the internet," *IEEE Trans. Educ.*, vol. 46, no. 1, pp. 142–148, 2003.

[21] J. Ma and J. V. Nickerson, "Hands-on, simulated, and remote laboratories: A comparative literature review," *ACM Comput. Surv.*, vol. 38, no. 3, p. 1, 2006.

[22] Universidade Politécnica de Catalunya, "Virtual Lab for Remote Studies — Center Innovation Electronics. Motion Control and Industrial Applications. MCIA — UPC. Universitat Politècnica de Catalunya." [Online]. Available: https://mcia.upc.edu/en/facilities/virtual-lab-for-remote-studies. [Accessed: 18-Nov-2019].

[23] N. Zoricu, Nedic; Jan, Machotkd; Andrew, "Remote laboratories versus virtual and real laboratories," *Education*, 2003.

[24] G. Payne, "Labvolution | LabVIEW: Web Services Part 3." [Online]. Available: http://www.labvolution.com/labview-web-services-part-3/. [Accessed: 18-Nov-2019].

[25] N. Jan, Machotka;Zorica, "NetLab - The Online Remote Laboratory." [Online]. Available: http://netlab.unisa.edu.au/index.xhtml. [Accessed: 18-Nov-2019].

[26] QuaSci Labs LLC, "ElectricVLab - 3D Virtual Lab for Electricity & Electronics." [Online]. Available: https://electricvlab.com/. [Accessed: 18-Nov-2019].

[27] C. Tüysüz, "The effect of the virtual laboratory on students' achievement and attitude in chemistry," *Int. Online J. Educ. Sci.*, vol. 2, no. 1, pp. 37–53, 2010.

[28] Cornell, "Torsion: Engineered and Biological Materials." [Online]. Available: https://courses.cit.cornell.edu/virtual_lab/intro.shtml. [Accessed: 18-Nov-2019].

[29] P. Bhargava, J. Antonakakis, C. Cunningham, and A. T. Zehnder, "Web-based virtual torsion laboratory," *Comput. Appl. Eng. Educ.*, vol. 14, no. 1, pp. 1–8, 2006.

[30] "The Program - IoT Community." [Online]. Available: http://iotcommunity.com/the-program/. [Accessed: 10-Sep-2019].

[31] L. Miclea, IEEE Computer Society Test Technology Technical Committee, Annual IEEE Computer Conference, Q. and T. IEEE International Conference on Automation, AQTR 19 2014.05.22-24 Cluj-Napoca, and THETA 19 2014.05.22-24 Cluj-Napoca, "Cyber Physical Systems in the Context of Industry 4.0," pp. 2–4, 2014.

[32] R. Rajkumar, "Cyber-Physical Systems: The Next Computing Revolution," pp. 1–6, 2010.

[33] "Cyber-Physical Systems (CPS)." [Online]. Available: http://www.imm.dtu.dk/~jbjo/cps.html. [Accessed: 27-Oct-2019].

[34] R. Malan, D. Bredemeyer, and B. Consulting, "Functional requirements and use cases," *functreq. pdf, 39k June*, pp. 1–8, 1999.

[35] "Development Board | Espressif Systems." [Online]. Available: https://www.espressif.com/en/products/hardware/development-boards. [Accessed: 27-Oct-2019].

[36] Last Minute Engineers, "In-depth: Create A Simple ESP32 Web Server In Arduino IDE." [Online]. Available: https://lastminuteengineers.com/creating-esp32-web-server-arduino-ide/. [Accessed: 27-Oct-2019].

[37] S. GMBH, "Compact Transport and Sorting Line Guide," vol. 1, no. 4, p. 53, 2000.

[38] A. Froehlich, “IP Addresses, Subnet Masks, and Default Gateways | IT Infrastructure Advice, Discussion, Community - Network Computing.” [Online]. Available: https://www.networkcomputing.com/network-security/ip-addresses-subnet-masks-and-default-gateways. [Accessed: 21-Nov-2019].

Anexos

Anexo 1 - Tabela Actuadores

Tabela 8.1 - Tabela referente aos actuadores do kit linha de transporte e triagem de peças

Pinos da ficha	**Objectivo**	**Pino da placa IoT**
20	Cilindro A - Movimento trás	23
21	Cilindro A - Movimento frente	21
22	Passadeira rolante	19
23	Cilindro B - Movimento trás	18
24	Cilindro B - Movimento frente	5
25	Cilindro C - Movimento frente	16
26	Cilindro C - Movimento trás	17
27	Luz de fim de curso	4

Anexo 2 - Tabela Sensores

Tabela 8.2 - Tabela referente aos sensores do kit linha de transporte e triagem de peças

Pinos da ficha	**Objectivo**	**Pino do IoT a que esta ligado (normalmente)**	**Tipo de activação**
1	Cilindro A – totalmente para trás	26	HIGH
2	Cilindro A – totalmente para a frente	32	HIGH
3	Cilindro B – totalmente trás	35	LOW
4	Cilindro B – totalmente para a frente	34	LOW
5	Cilindro C – totalmente trás	33	LOW
6	Cilindro C – totalmente frente	25	LOW
7	Cilindro B - detecta peça	27	LOW
8	Cilindro C - detecta peça	14	LOW
9	identificação tipo de peça 1	2	HIGH
10	identificação tipo de peça 2	15	HIGH

Anexo 3 – Circuito final

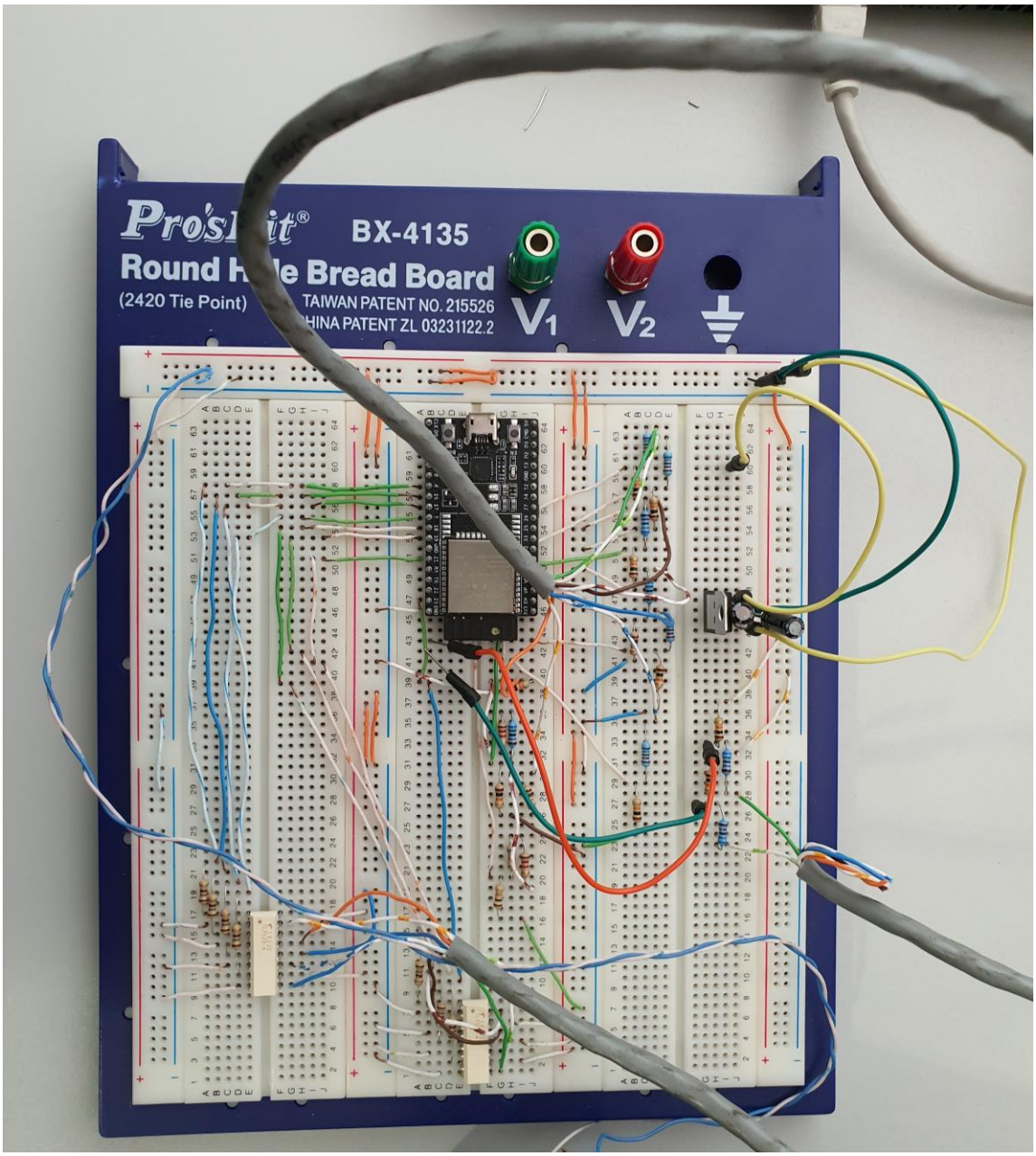

Figura 8.6.1 - Circuito final em Breadboard

Anexo 4 – Esquemático completo em Eagle

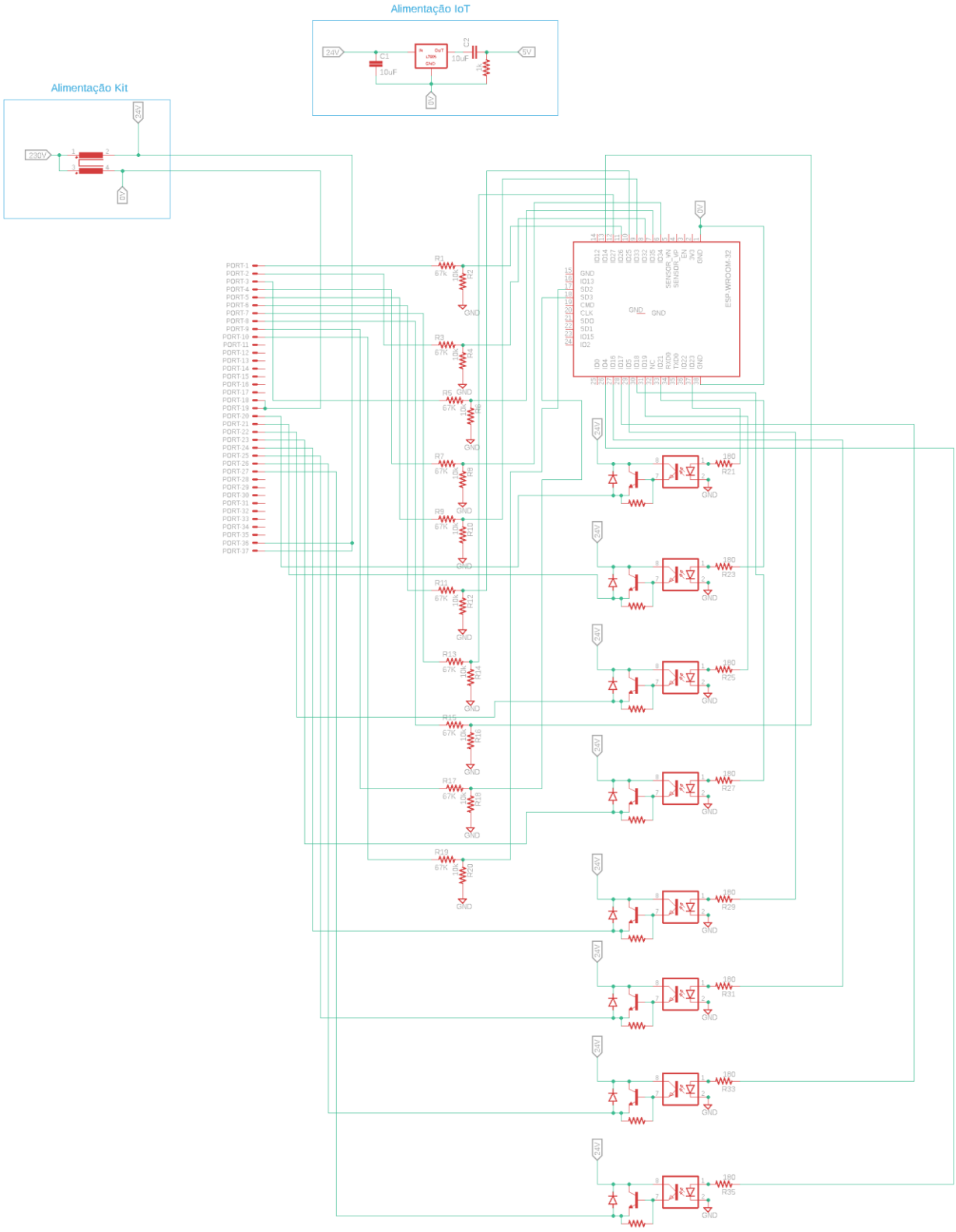

Figura 8.6.2 - Esquemático final usando o programa Eagle

Anexo 5 – NI- DAQ (1)

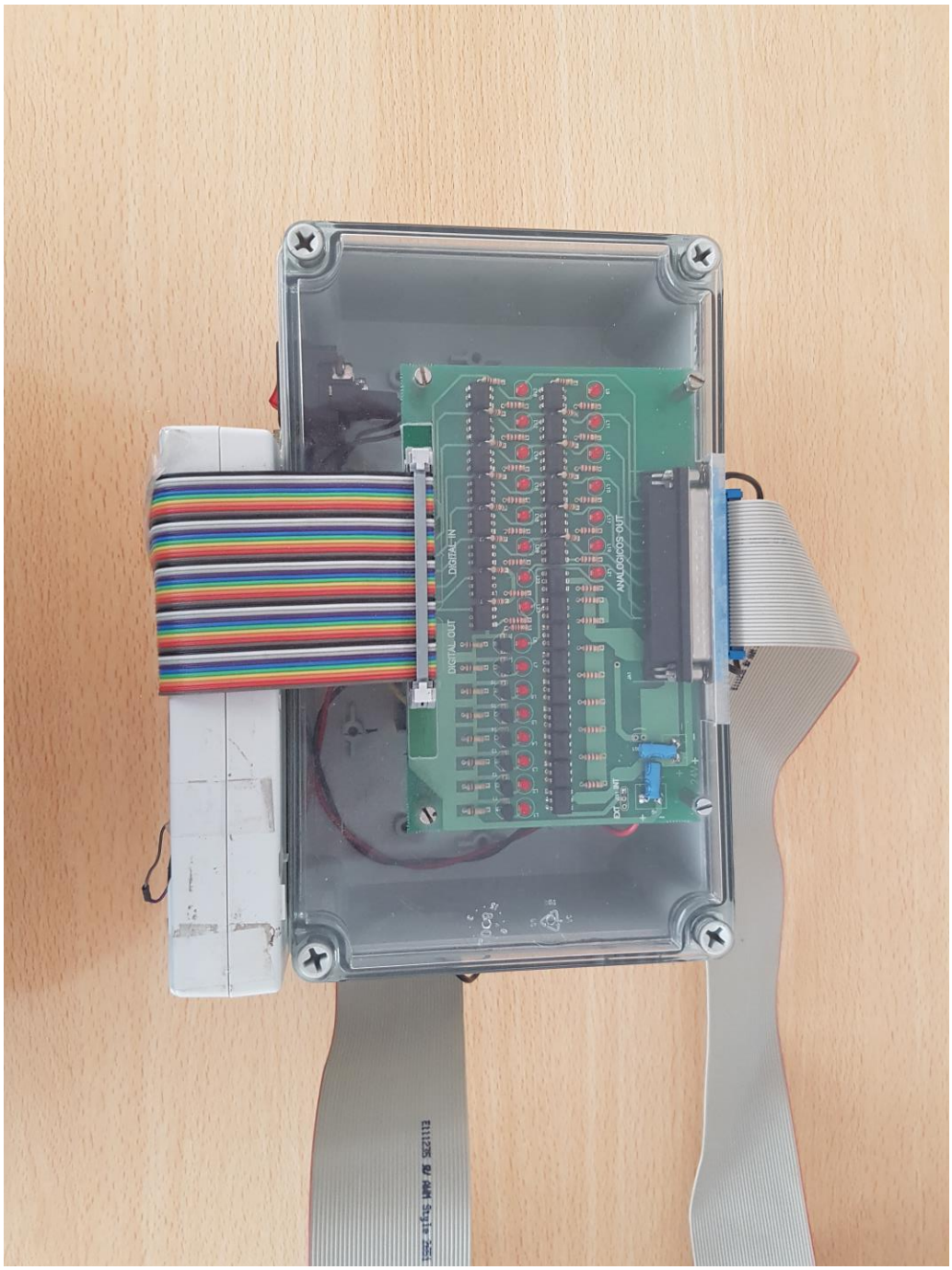

Figura 8.6.3 - Data Acquisition Device (DAQ) da National Instruments (visão superior)

Anexo 6 – NI- DAQ (2)

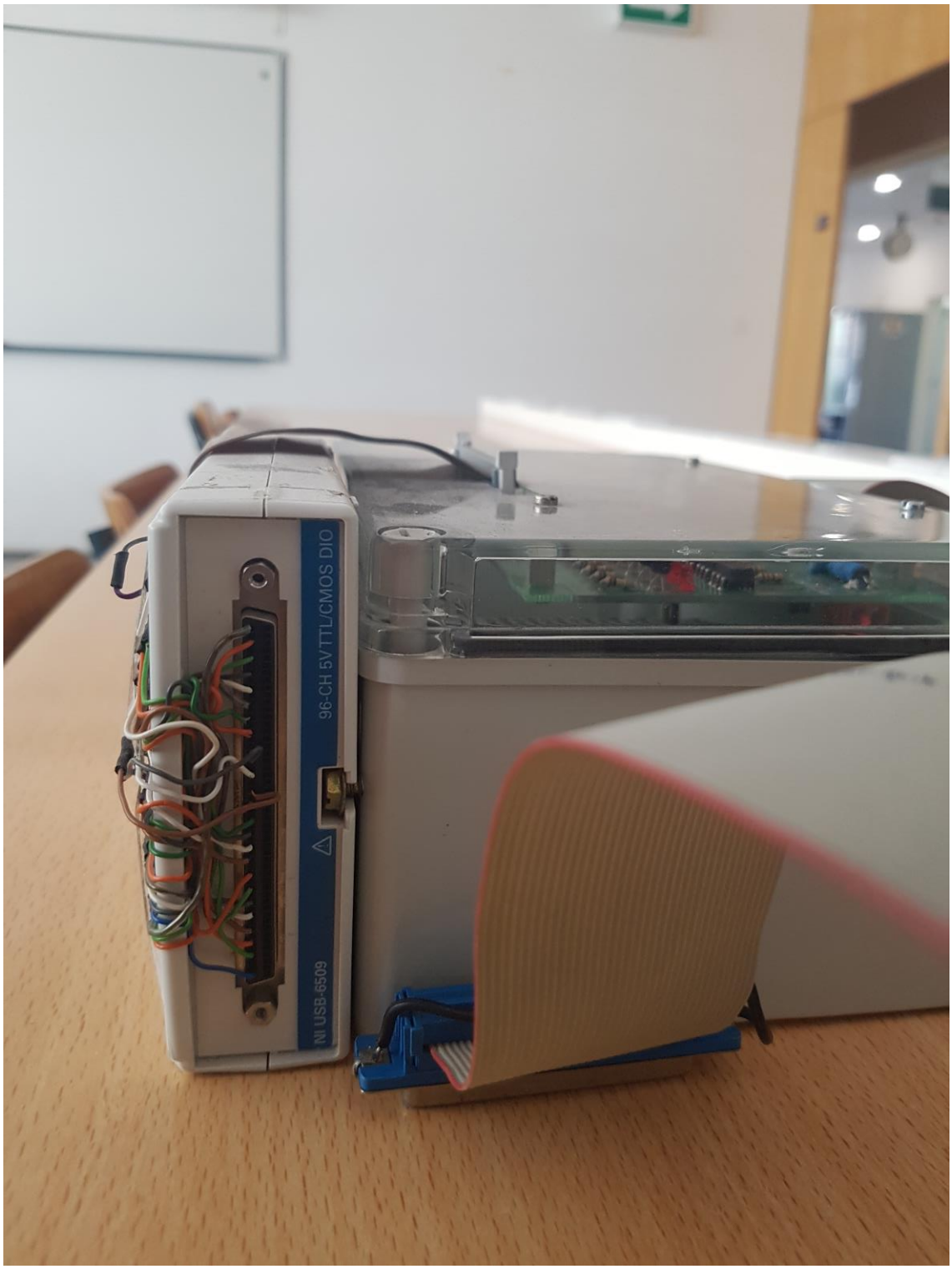

Figura 8.6.4 - Data Acquisition Device (DAQ) da National Instruments (visão lateral)

Anexo 7 - Ligação plataforma IoT ao Kit real

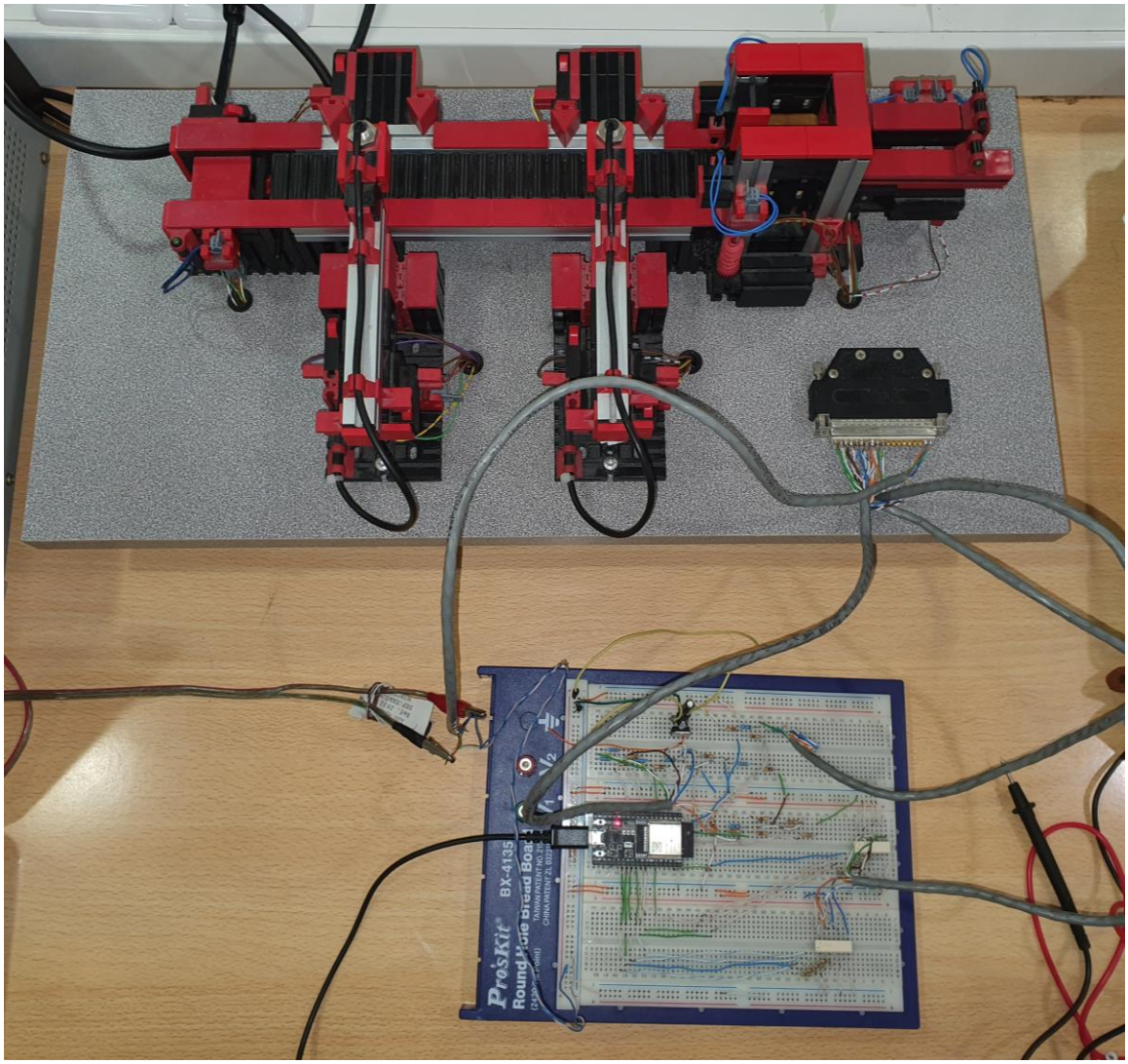

Figura 8.6.5 - Ligação Plataforma IoT ao kit real