IPG Politécnico da Guarda
Polytechnic of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

Tércio Paulino Almeida Azevedo

dezembro | 2013

Instituo Politécnico da Guarda

Escola Superior de Tecnologia e Gestão

ALERT BRACELET

Pulseira de Alarme com o Arduíno

TÉRCIO PAULINO ALMEIDA AZEVEDO

Nº 1009380

PROJETO DE INFORMÁTICA DO CURSO DE ENGENHARIA INFORMÁTICA

EM ENGENHARIA INFORMÁTICA

29 de Novembro de 2013

Instituo Politécnico da Guarda

Escola Superior de Tecnologia e Gestão

ALERT BRACELET

Pulseira de Alarme com o Arduíno

TÉRCIO PAULINO ALMEIDA AZEVEDO

Nº 1009380

ORIENTADOR: PROFESSOR LUIS FIGUEIREDO

PROJETO DE INFORMÁTICA DO CURSO DE ENGENHARIA INFORMÁTICA

EM ENGENHARIA INFORMÁTICA

29 de Novembro de 2013

Agradecimentos

Gostaria de expressar o meu agradecimento a todas as pessoas que das mais variadas formas me ajudaram neste projeto. Como não é possível enunciar cada um, vou apenas salientar algumas das pessoas que não poderia deixar de destacar.

Ao Professor Doutor Luís Figueiredo por ter aceitado ser o meu orientador, por toda a disponibilidade e ajuda nas dificuldades que foram aparecendo guiando-me sempre no caminho mais correto assim como a toda a equipa do Magickey.

À Professora Doutora Maria Clara Silveira por toda a ajuda prestada na parte de análise de requisitos e conceção da aplicação, ao Professor Doutor Paulo Vieira que se demonstrou sempre disponível para me ajudar em todos os problemas que iam surgindo, assim como também a todos os outros professores do IPG que mais diretamente ou indiretamente me ajudaram neste projeto e me prepararam para chegar até aqui.

Por fim, mas não menos importantes, gostaria de agradecer a minha família e amigos por todo o apoio que sempre demonstraram. Sem eles tudo teria sido bem mais difícil.

Resumo

Nos últimos anos tem-se assistido a um enorme avanço tecnológico no campo da informática, eletrónica e microeletrónica.

Hoje em dia existem muitos programas para Computador, Tablet, Telemóveis, assim como pulseias eletrónicas que fazem chamadas, com GPS entre outros, mas esquecem-se que estamos em tempos de crise e existem pessoas sem grandes possibilidades que também precisam ser ajudadas e não podem ser esquecidas.

Com base nisso desenvolvi este projeto que tem como objetivo desenvolver uma pulseira simples e barata que alerte a pessoa sempre que precisa de tomar um medicamento.

Para além disso foi desenvolvido um programa em C# para *desktop* que terá como objetivo ser fornecido às farmácias e quando o utente compra os medicamentos pode também comprar a pulseira na farmácia.

O farmacêutico introduz no computador o horário necessário para aquele determinado medicamento comprado e envia os dados para a pulseira.

Este relatório descreve o trabalho que foi realizado no âmbito da unidade curricular Projeto de Informática na Licenciatura em Engenharia Informática da Escola Superior de Tecnologia e Gestão da Guarda.

Palavras-chave

Arduíno, Alarme, Pulseira, App, C#

Abstract

In last year there has been a huge technological advancement in the field of computer science, electronics and microelectronics.

Nowadays there are many programmers for Computer, Tablet, Mobile Phones, as well as electronic bracelets that make calls with GPS among others, but they forget that we are in times of crisis and there are people with a low income who need to be helped and can not be forgotten.

Based on that I developed this project that aims at developing a simple and inexpensive bracelet that alerts the person every time he/she needs to take medication.

Besides, a programme C# for desktop was developed to be used in pharmacies so that when the bracelet with the timer is bought by the user the pharmacist programmers it according to the frequency the medication has to be taken.

This report describes the work carried out within the course of Computer Project in Informatics Engineering, Guarda School of Technology and Management.

Keywords

Arduino, Alarm, Bracelet, App, C#

Conteúdo

Lista de Figuras

Glossário3

C#	É uma linguagem de programação de alto nível orientada a objetos.
Visual Studio	É um conjunto de produtos, ferramentas e tecnologias para desenvolver software.
Arduíno	É um microcontrolador que usa a linguagem de programação C/C++.
API	Application Program Interface. Conjunto de rotinas, funções.
GND	Massa da fonte de alimentação.
LED	Light-Emitting Diode. Díodo semicondutor que quando lhe passa energia emite uma luz vizível.
PC	Personal Computer. Computador pessoal.
MCu	Abreviatura para microcontrolador.
Interface Serial/Porta Serie	Também conhecida como RS-232, é uma porta de comunicação utilizada por alguns equipamentos onde os bits são transferidos em fila.
USB	Universal Serial Bus. Interface padrão, que oferece uma plataforma para estabelecer uma conexão plug-and-play entre diversos dispositivos digitais.
Web Page	Página Web, isto é, de hipertextos acessíveis geralmente pelo protocologo http na internet.
SDK	Software Developers Kit. Pacote que inclui ferramentas como APIs, linguagens de ecripting e interface gráfica necessárias para o desenvolvimento do software.

Capítulo 1

Introdução

O presente relatório descreve o projeto desenvolvido no âmbito da Unidade Curricular Projeto de Informática, na Licenciatura em Engenharia Informática da Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda.

O microcontrolador Arduíno tem a capacidade de armazenamento, realizar processamento, trocar informações e devido ao seu reduzido tamanho, ser transportado facilmente pelo seu utilizador.

O projeto consiste em desenvolver uma aplicação simples e amigável, que permita efetuar as diferentes programações dos medicamentos do dia-a-dia do utente no Arduíno Micro, sendo para tal necessário desenvolver um protocolo de comunicação entre o equipamento e a aplicação.

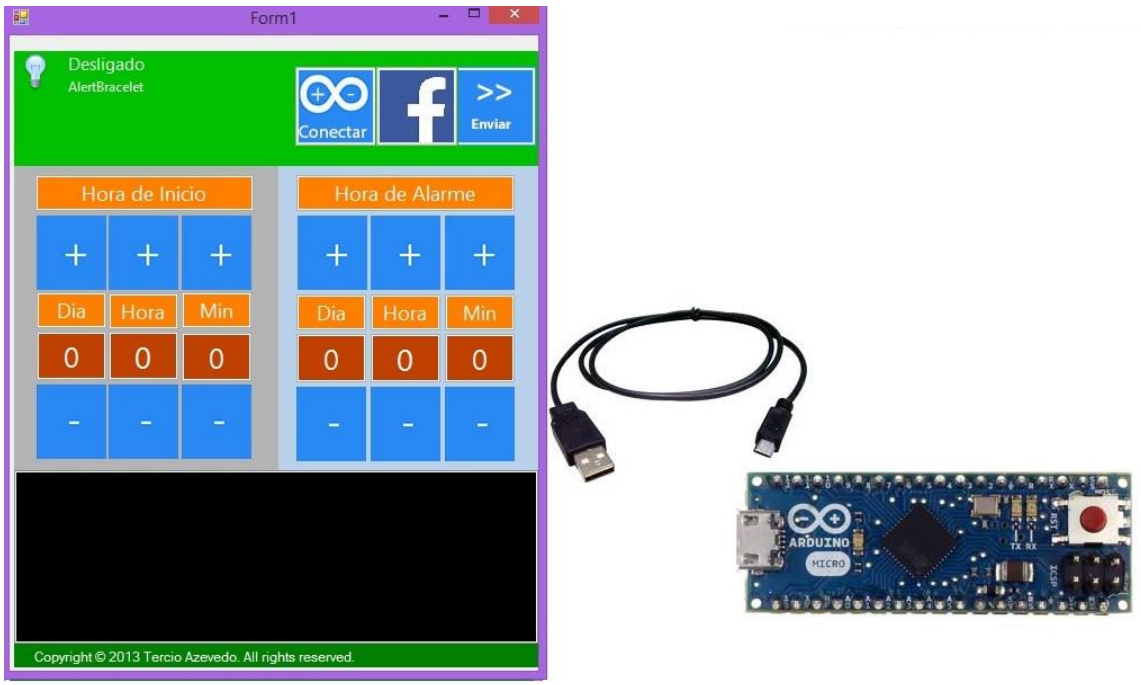


Figura 1: Comunicação

1.1 Motivação

Como a área da eletrónica e microeletrónica é uma área que está a sofrer uma grande evolução, existe uma imensa quantidade de aplicações de projetos inovadores. Para que a aplicação desenvolvida se destaque pensei em algo que pudesse ajudar as pessoas e ainda não existisse nada igual já feito.

Como com o Arduíno ainda não existia nenhum projeto deste tipo, houve aqui uma oportunidade de criar algo novo, comprovado pela pesquisa de aplicações e outros projetos para poder ter a certeza que estava a criar algo diferente e único.

A principal motivação para o desenvolvimento deste projeto é a possibilidade de aplicar os meus conhecimentos na área da programação no projeto e aprender com as novas tecnologias atualmente disponíveis.

Além disso, criar algo que não existe e no futuro ver as pessoas usar a minha pulseira é um sonho e uma grande motivação para mim.

1.2 Objetivos gerais

Criar uma aplicação que se integre bem com o "Look and Feel" do desktop, tátil ou não, que seja rápida e eficaz e simples de utilizar, assim como um programa para o Arduíno que receba os dados, verifique se chegaram corretos e desperte nas horas definidas.

Pretende-se assim que o sistema tenha a capacidade mínima para processar três alarmes diferentes.

1.3 Solução proposta

A solução encontrada para o projeto foi feita de acordo com os requisitos pretendidos. Desenvolver uma aplicação para desktop em C# que visa controlar todos os aspetos da solução pretendida, tais como a introdução da hora de repetição dos medicamentos, assim como a hora que quer iniciar a tomar, bem como o desenvolvimento da aplicação para o Arduíno que permita a receção dos dados e o seu processamento autónomo.

1.4 Contribuição

A contribuição principal deste trabalho é o desenvolvimento, implementação e testes do microcontrolador Arduíno, que possibilita a utilizadores deste projeto, serem alertados sempre que tiverem um medicamento importante a tomar, como já foi descrito acima.

1.5 Estrutura do documento

O documento compreende sete capítulos para além do presente capítulo, e está organizado da seguinte forma:

- No segundo capítulo são descritos alguns dos produtos concorrentes com o mesmo objetivo do meu.
- No terceiro capítulo são descritos todos os objetivos, assim como alguns problemas que surgiram no início do desenvolvimento da aplicação.
- No quarto capítulo é indicada e descrita a metodologia utilizada para desenvolver a aplicação.
- No quinto capítulo são descritas todas as tecnologias e software usados para a realização do projeto.
- No sexto capítulo é feita uma análise do programa em C# e do Arduíno, assim como do pacote usado.
- Por fim no sétimo capítulo é feita uma análise do projeto, bem como objetivos que irão ser futuramente implementados.

Capítulo 2

Soluções Similares

Esta área está em constante evolução e esta sempre a haver novos projetos, sendo para isso apenas necessário ter uma boa ideia.

Eu como pensei ter uma boa ideia, então decidi avançar e fazer o projeto, para isso fiz alguma pesquisa sobre soluções similares e mostro aqui alguns desses exemplos encontrados até ao momento dessa pesquisa.

Hora do Remedio e Medical Alarm

São o exemplo de duas aplicações desenvolvidas para telemóvel pagas, com o mesmo objetivo da minha aplicação, mas com o contra de necessitar da pessoa ter um Smartphone e saber trabalhar com ele, coisa que não acontece com a maior parte dos idosos, e ainda podem não ter o Smartphone consigo e não ouvirem o alarme.

Se estiverem a usar uma pulseira isso nunca irá acontecer, pois até podem dormir com ela sem terem a necessidade de comprar nenhum Smartphone nem terem que comprar nenhum programa destes e mais a publicidade que traz com o mesmo.

(Marmitt, 2013)

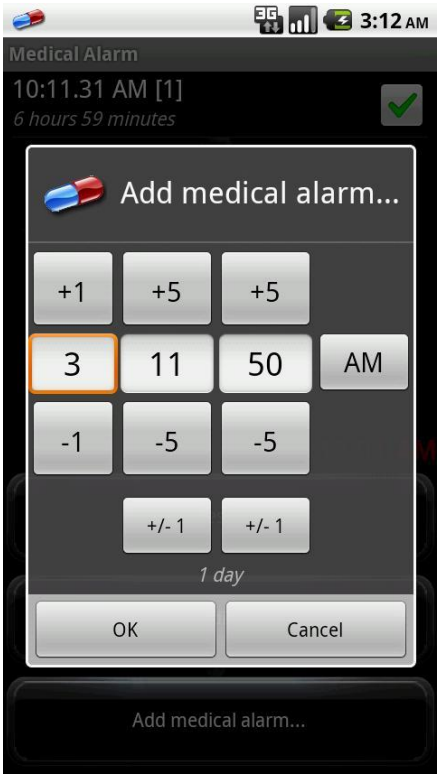


Figura 2: Imagem da aplicação Medical Alarm

Figura 3: Imagem da aplicação Hora do Remédio

Também temos outras soluções no mercado, como por exemplo um recipiente para metermos os nossos medicamentos, com espaço para 3 diferentes como mostra a figura 4, embora também se possa deixar esta caixa num lugar qualquer e não se ouvir o alarme se não nos encontrarmos por perto.

Figura 4: Caixa para medicamentos com alarme

Assim como também temos pulseiras ou relógios com Alarmes e GPS integrados entre outras funções, sendo todas essas soluções caras.

Figura 5: A direita pulseira com GPS e Alarme a esquerda novo relógio do Google

Capítulo 3

Definição do problema e objetivos previstos

O objetivo é criar uma aplicação para alarme de medicamentos.

A aplicação deve poder inserir a hora que queremos iniciar a tomar o medicamento, assim como a hora de repetição desse mesmo medicamento, estado já preparada para ecrãs tátil e ser o mais simples possível e o programa do Arduíno deverá ser capaz de criar até 3 alarmes.

Para que seja possível criar uma aplicação com todo este conteúdo foi necessário:

- Comprar o Arduíno Micro, cabo de ligação ao computador e o speaker para o alarme.
- Criar um programa em C# para desktop que têm a capacidade de comunicar com o Arduíno
- Criar um protocolo de comunicação entre Arduíno e o programa em C#
- Criar um programa para o Arduíno que leia esses dados enviados do Computador e dê um feedback se foi bem recebido.
- Criar uma matriz que guarde até três pacotes de dados para criar até três alarmes e nos informar quantos alarmes estão programados.

3.1 Definição do problema

Quando se está a criar uma aplicação para um microcontrolador, é necessário ter um especial cuidado com a programação e com os recursos que o microcontrolador tem.

Os problemas iniciais que se tiveram que resolver foram os seguintes:

- Comunicação entre C# e Arduíno pela Serial, onde não se estava a conseguir comunicar do Arduíno para o C#, para isso foi encontrada a solução de usar a Serial 1 ligada aos pinos RX e TX do Arduíno.
- As músicas no Arduíno tocadas nas Rotinas de Atendimento de Interrupts, provocavam a paragem da contagem do tempo, ficando este com atrasos nos alarmes, sendo que a solução encontrada foi colocação dessas músicas no programa principal.

3.2 Objetivos previstos

Os objetivos pretendidos nesta aplicação são:

- Definir a hora de início do alarme no programa em C#.
- Definir a hora do alarme no programa em C#.
- Definir um protocolo de comunicação.
- Confirmar a receção dos dados recebidos no programa de Arduíno.
- Gerar até três alarmes com o Arduíno.

Capítulo 4

Metodologia e resultados esperados

4.1 Metodologia

A metodologia escolhida e utilizada para desenvolver, implementar e testar a aplicação foi o desenvolvimento ágil. Esta abordagem iterativa faz com que o utilizador avalie a aplicação, sendo que o programador recebe um feedback constante. Isto torna-se bom para o programador, pois facilita as adaptações ao processo de desenvolvimento.

Esta metodologia valoriza, entre outros, os seguintes princípios:

- Permite efetuar mudanças tardias no projeto sem grandes problemas;
- Software funcional mais do que documentação extensa;
- Simplicidade;
- Responder a mudanças mais do que seguir um plano;
- Garantir a satisfação do utilizador;

O desenvolvimento ágil não descarta os métodos tradicionais, tais como documentação, ferramentas e processos, planeamentos e negociações, mas procura dar-lhes uma cotação secundária perante indivíduos e uma resposta mais eficaz perante mudanças. Uma interação constante da parte do utilizador é uma mais-valia para qualquer projeto. Por estes motivos deve ser um método a utilizar.

4.2 Planeamento

As principais tarefas foram:

- Tarefa 1 – Análise dos requisitos.
- Tarefa 2 – Pesquisa e obtenção de documentação.
- Tarefa 3 – Implementação da aplicação para desktop.
- Tarefa 4 – Implementação da aplicação para Arduíno
- Tarefa 5 – Testar a aplicação a fundo.
- Tarefa 6 – Elaboração do relatório.

O Mapa de Gantt, da Figura 6 mostra como decorreu o desenvolvimento do projeto.

Nome Tarefa	**Duração**	**Início**	**Conclusão**
Análise dos requisitos	2 dias	11/03/13	12/03/13
Pesquisa e obtenção de documentação	6 dias	13/03/13	21/03/13
Implementação da aplicação para desktop	30 dias	01/08/13	09/09/13
Implementação da aplicação para Arduíno	30 dias	10/09/13	18/10/13
Testar a aplicação a fundo	17 dias	21/10/13	08/11/13
Elaboração do relatório	17 dias	11/11/13	28/11/13
Total	102 dias	Cerca de 600 horas	

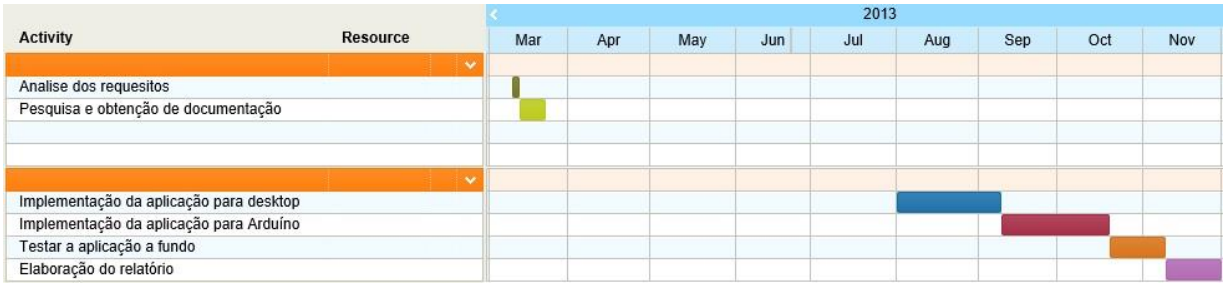


Figura 6: Mapa de Gantt

Capítulo 5

Tecnologias utilizadas

5.1 Introdução

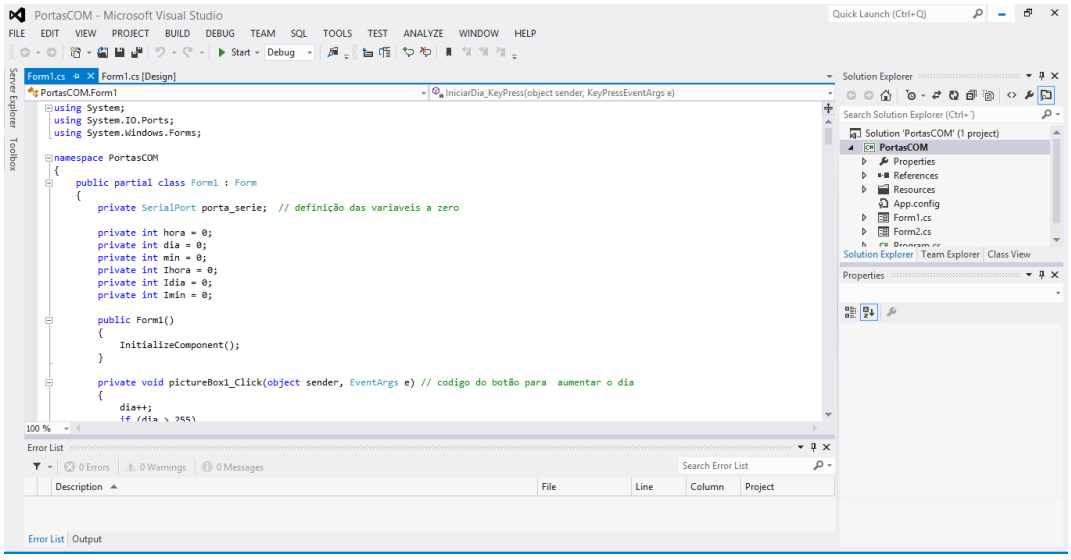
Neste capítulo vão ser descritas todas as tecnologias utilizadas na elaboração deste projeto. Para a aplicação do Arduíno foi utilizado C/C++. O programa de *desktop* foi programado em C#.

5.2 Tecnologias

5.2.1 C#

O C# (CSharp) é uma linguagem de programação orientada a objetos criada pela Microsoft. Faz parte da plataforma .NET e é baseada em C++ e Java. A linguagem C# foi criada junto com a arquitetura .NET. Embora, existam outras linguagens que suportam esta tecnologia (como VB.NET, C++, J#), o C# é considerada a linguagem símbolo do .NET, porque foi criada praticamente do zero para funcionar na plataforma .NET e porque a maior parte das classes do .NET Framework foram desenvolvidas em C#. Esta linguagem de programação foi utilizada para o programa desktop. e tem a seguinte estrutura.

A listagem 1 mostra um exemplo do código C#.



5.2.2 Arduíno

O Arduino foi criado em 2005 para facilitar a programação e a eletrónica, usando uma plataforma aberta de desenvolvimento.

Na prática ligamos componentes nas portas analógicas e digitais e escrevemos programas que usam essas portas.

Tradicionalmente o microcontrolador Arduíno é programado usando o seu IDE proprietário e feito de modo a fazer o upload do código para o microcontrolador de seguida, hoje em dia também temos a opção de usarmos o CodeBender que pretende agora revolucionar a forma de programar para Arduíno no que se refere a plataforma "IDE".

Programar usando o Browser como IDE permite uma flexibilidade muito maior, não sendo estritamente necessário ter instalado o IDE proprietário do Arduíno.

O CodeBender além de codificar permite ainda partilhar o seu código na cloud e colaborar com outros programadores.

Listagem 8 Exemplo Arduíno

```
1: int led = 13;
2:    void setup() {
3:       pinMode(led, OUTPUT);
4:    }
5: void loop() {
6:   digitalWrite(led, HIGH);
7:   delay(1000);
8:   digitalWrite(led, LOW);
9:   delay(1000);
   }
```

Temos que obrigatoriamente programar dois métodos:

void setup( ) {

}

void loop( ) {

}

O setup é executado uma só vez assim que a placa for ligada e o loop terá o código de execução infinita. No setup() definem-se quais pinos que serão de entrada e saída e outros parâmetros de inicialização.

5.3 Software utilizado

5.3.1 Microsoft Visual Studio

O Microsoft Visual Studio é um pacote de programas da Microsoft, para criação de software, é especialmente dedicado ao framework .NET e às linguagens Visual Basic (VB), C, C++, C# (C Sharp). É também um grande produto de desenvolvimento na área web, usando a plataforma ASP.NET.

A Figura 7 mostra o interface principal do Microsoft Visual Studio.

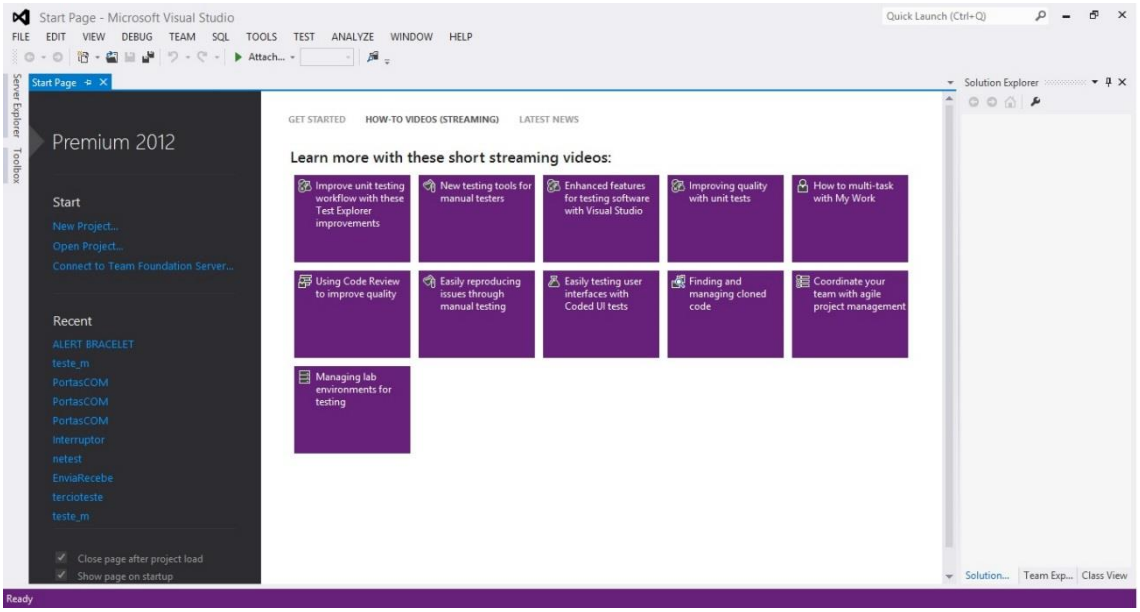


Figura 7: Microsoft Visual Studio.

5.3.2 Microsoft Visio

O Microsoft Visio é um software que permite criar diagramas representativos, ou seja, imagens que possibilitam a análise de informação. Com o Visio pode-se modelar os dados através de diagramas extremamente dinâmicos, tornando assim mais fácil a realização de projetos.

A Figura 8 mostra o interface do Microsoft Visio.

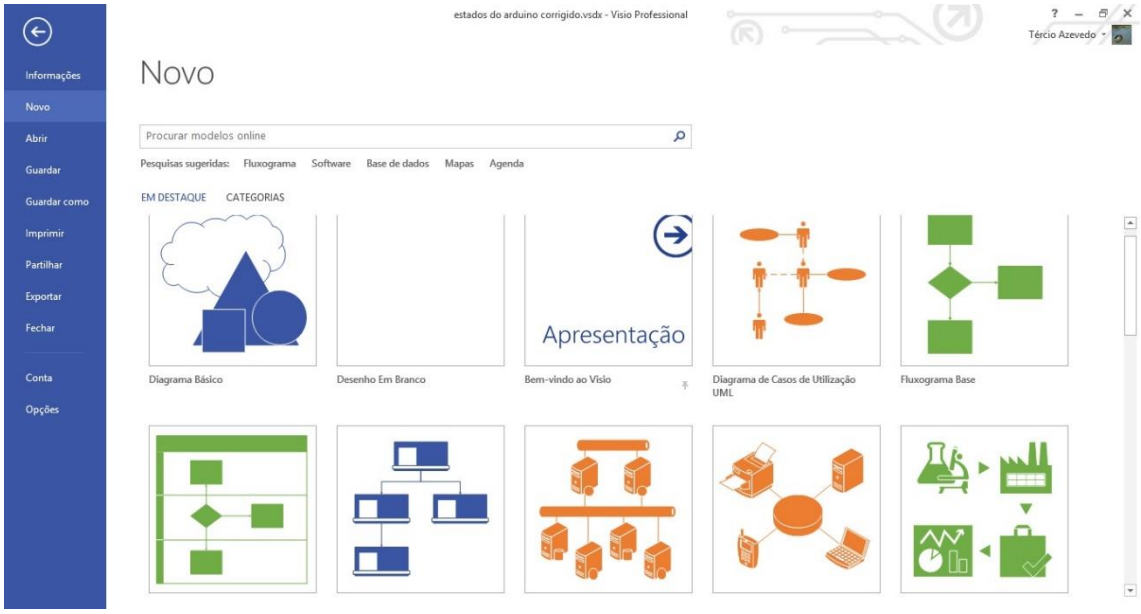


Figura 8: Microsoft Visio

5.3.3 Programa do Arduíno

O ambiente de desenvolvimento do Arduíno, Figura 9 contém um editor de texto para a escrita de código. Tem uma zona de mensagens, uma zona de controlo de informações, uma barra de ferramentas com botões para funções comum e um conjunto de menus, onde podemos também encontrar pequenos exemplos de programas.

Este programa conecta-se ao hardware do Arduíno Micro, via USB para carregar os programas e comunicar com eles.

O ambiente de desenvolvimento do Arduíno possibilita que os programas sejam organizados em torno de duas funções, mas primeiro é necessário declarar as variáveis e fazer o include das API´s a usar.

Os programas escritos com este programa são chamados de sketches.

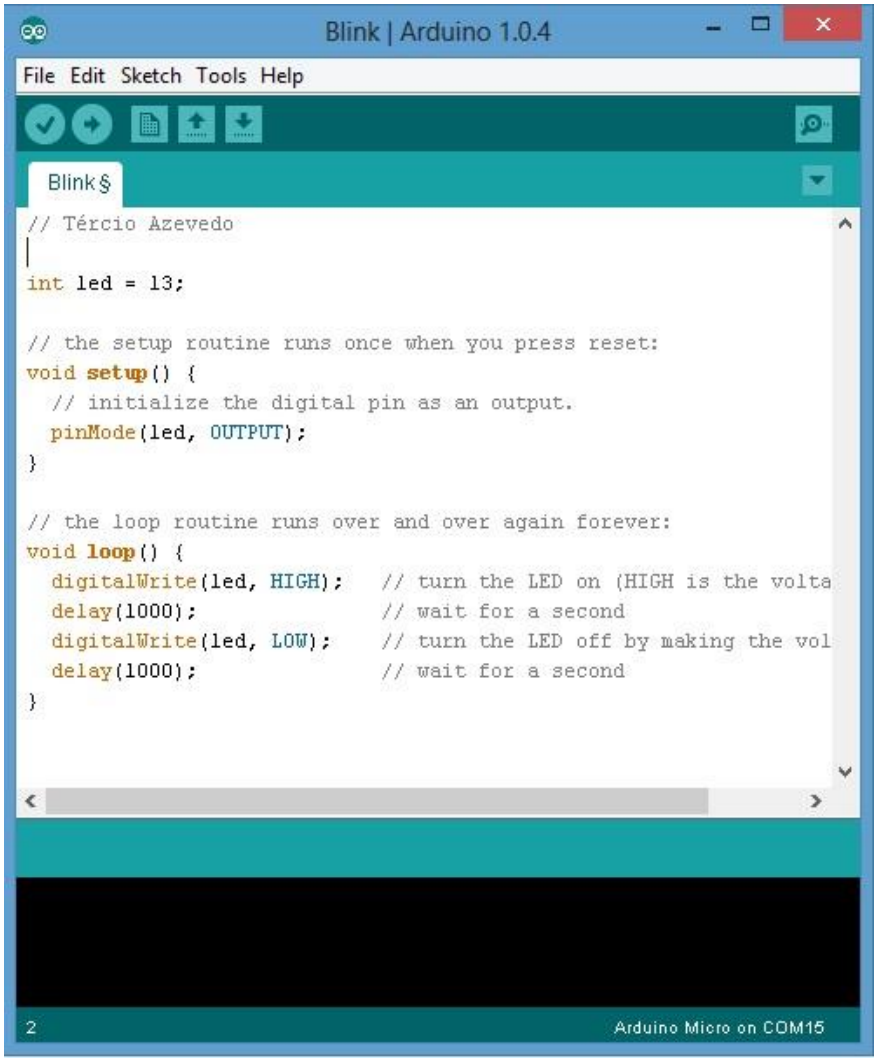


Figura 9: Programa do Arduíno

5.3.4 CodeBender

O CodeBender como já foi citado anteriormente é uma boa opção ao programa do Arduíno, é uma nova ferramenta, ainda pouco conhecida entre alguns programadores mas muito útil para quem programa em Arduíno, pois podemos trabalhar em equipa com outros programadores e ter o nosso código guardado sempre *online*, e apenas partilhamos esse código com quem quisermos, sem a necessidade de ter que instalar o programa no computador, apenas com a limitação de termos que ter internet para podermos trabalhar.

Para aceder a esta plataforma, basta aceder á página: http://codebender.cc

A Figura 10 mostra o interface do CodeBender.

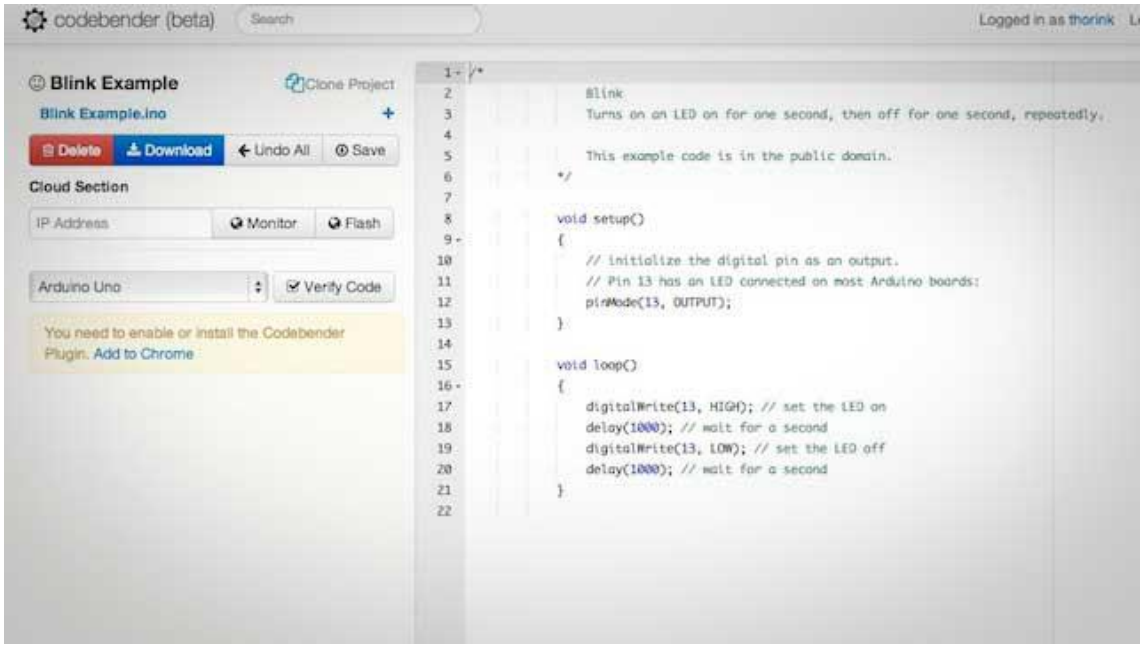


Figura 10: Interface CodeBender

Capítulo 6

Análise de requisitos e Conceção da aplicação

6.1 Fluxograma do Sistema com Estados

Um diagrama de estados é um esquema que mostra todo o comportamento dos objetos, um estado representa uma situação estável de um objeto que se prolonga durante um intervalo de tempo, durante o qual um objeto não sobre estímulos nem os atributos sofrem alterações de valor.

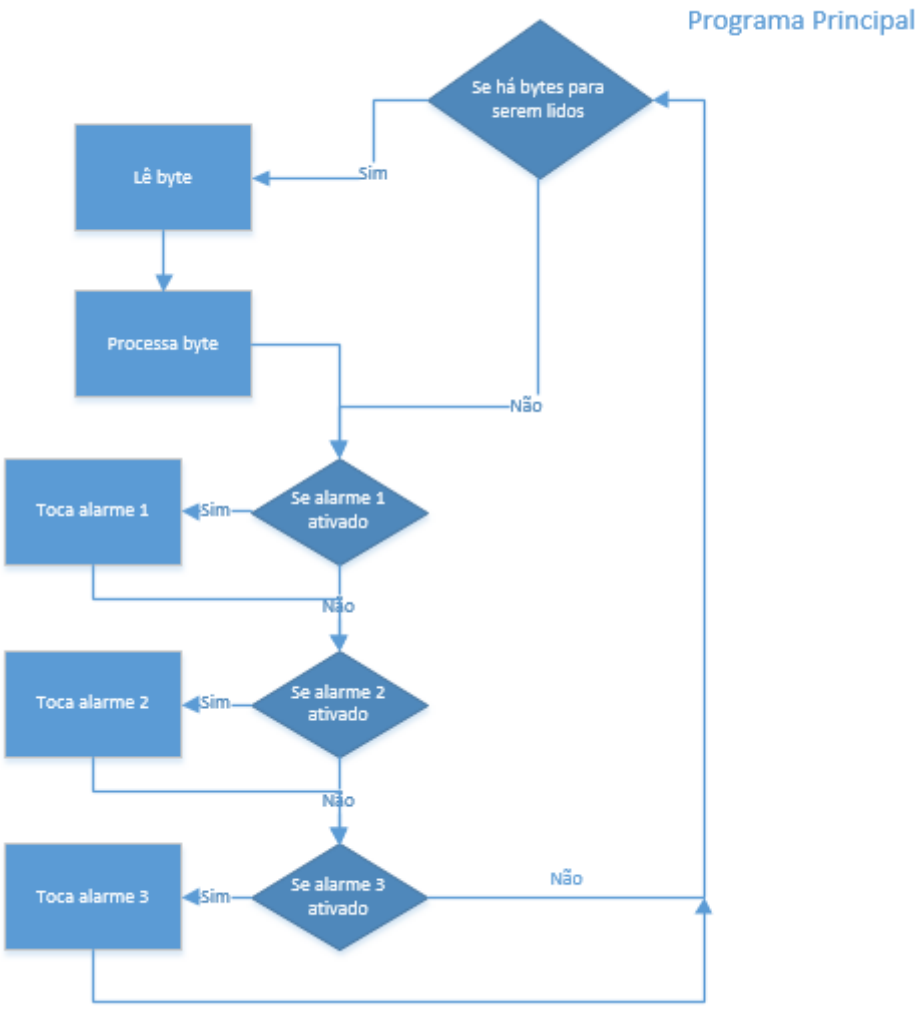


Figura 11: Fluxograma do Sistema com Estados

6.2 Algoritmos

Algoritmo genérico

1. Se há bytes para serem lidos na Porta Serie
   1.1. Ler byte
   1.2. Processa o byte
2. Se alarme 1 ativado
   2.1. Toca o alarme 1
3. Se alarme 2 ativado
   3.1. Toca alarme 2
4. Se alarme 3 ativado
   4.1. Toca alarme 3
5. Volta ao ponto 1

Algoritmo detalhado do ponto 1.2 (processa byte)

1. Se estado igual a zero
   1.1. Se byte igual a zero
      1.1.1. Incrementa estado
   1.2. Senão
      1.2.1. Envia erro
2. Se estado igual a um
   2.1. Se byte igual a 255
      2.1.1. Incrementa estado
   2.2. Senão
      2.2.1. Envia erro
      2.2.2. Estado igual a zero
3. Se estado maior que um
   3.1. Guarda o byte
   3.2. Incrementa o estado
4. Se estado igual a 11
   4.1. Calcula CheckSum
   4.2. Se CheckSum calculado igual a CheckSum recebido
      4.2.1. Envia bem recebido
      4.2.2. Estado igual a zero
      4.2.3. Configura o alarme
   4.3. Senão
      4.3.1. Envia erro
      4.3.2. Estado igual a zero

6.3 Diagrama de casos de uso

Um diagrama de casos de uso permite ver, de uma forma prática, todas as funcionalidades do sistema, assim como todas interações com o ator.

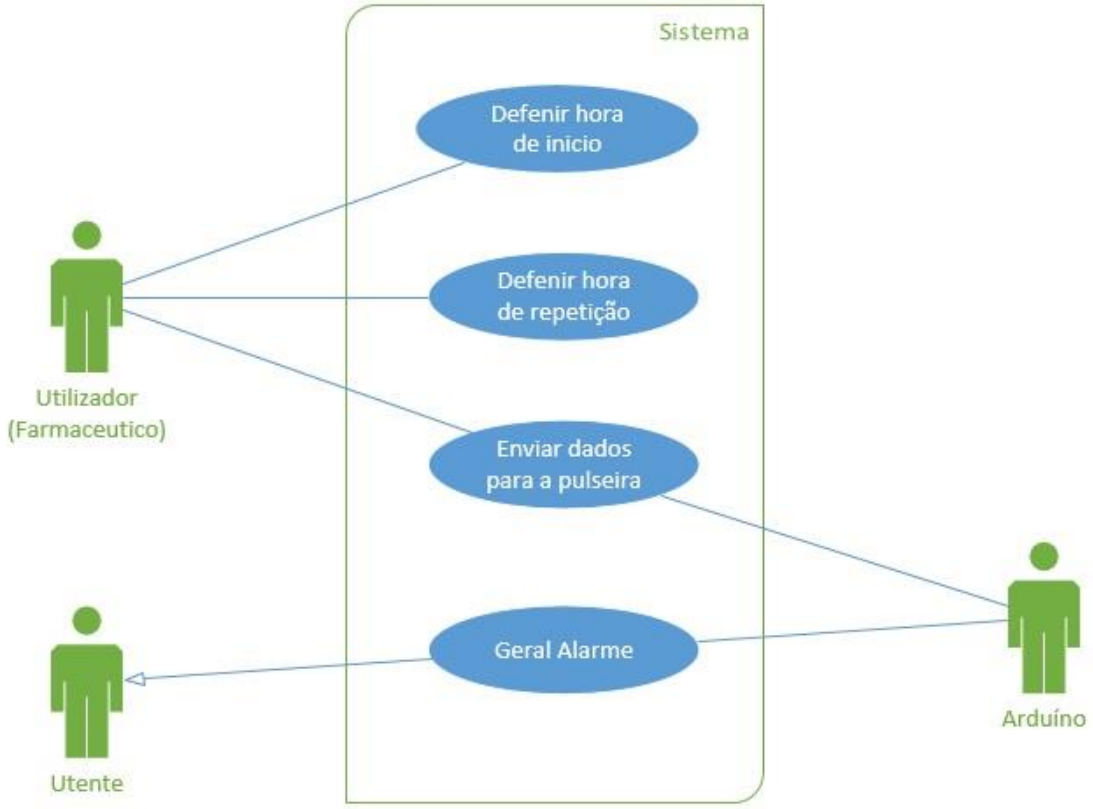


Figura 12: Diagrama de casos de uso da aplicação.

Como mostra a Figura 12, o farmacêutico introduz a hora de início e de repetição no programa em C# e envia os dados para a pulseira (Arduíno), este dá o feedback ao farmacêutico se os dados foram bem recebidos ou não e depois gera o alarme na hora pretendida para o utente.

6.4 Informação sobre o microcontrolador usado (Arduíno)

O Arduíno faz parte do conceito de hardware e software livre e está aberto para uso e contribuição de toda sociedade. O conceito Arduíno surgiu na Itália, em 2005, com o objetivo de criar um dispositivo que fosse utilizado em projetos/protótipos construídos de uma forma menos dispendiosa do que outros sistemas disponíveis no mercado.

O equipamento é uma plataforma de computação física: são sistemas digitais que podem ser ligados a sensores e atuadores, que permitem construir sistemas que percebam a realidade e respondem com ações físicas. Foi desenvolvido com uma biblioteca de funções que simplificam a sua programação, por meio de uma sintaxe similar à das linguagens C e C++.

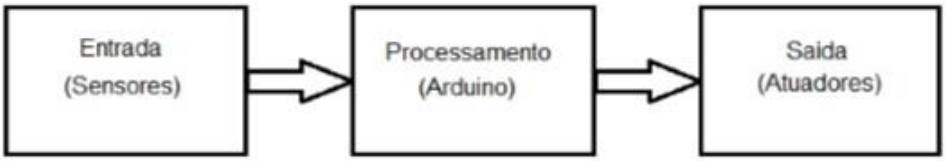


Figura 13: Diagrama de blocos de uma cadeia de processamento, utilizando o Arduíno.

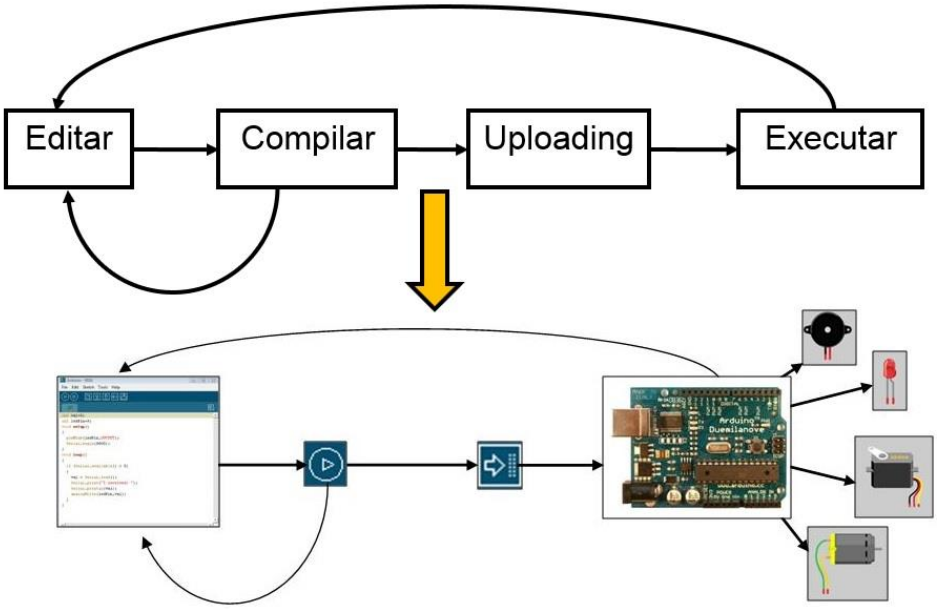


Figura 14: Ciclo de desenvolvimento.

6.5 Arquitetura geral da instalação

Os diagramas de instalação mostram toda a distribuição do *hardware*.

O diagrama seguinte mostra a arquitetura da aplicação, onde usamos uma bateria de testes com um interruptor para alimentar o Arduíno e este liga a um speaker ligado a mesma massa e a uma das saídas do Arduíno.

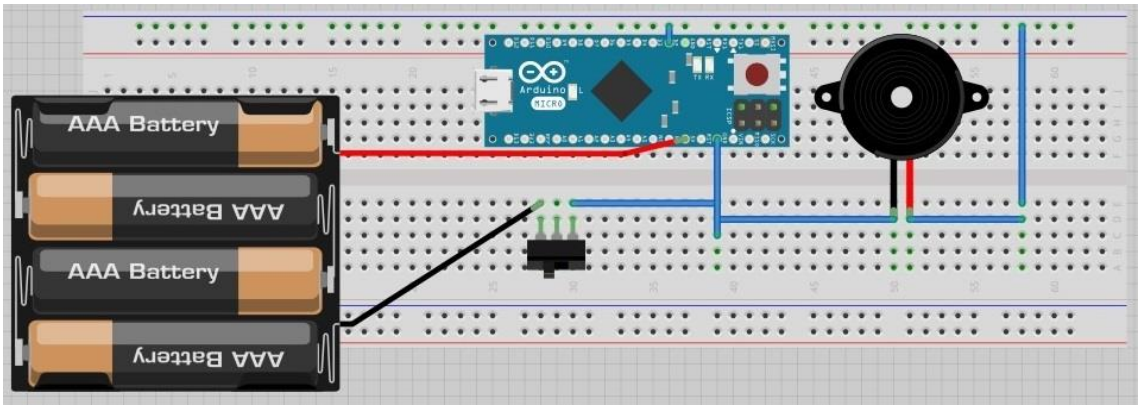


Figura 15: Arquitetura da aplicação

6.6 Características técnicas

O Arduíno Micro tem uma série de facilidades para se comunicar com um computador, outro Arduíno ou outros microcontroladores. O ATmega32U4 fornece UART TTL (5V) de comunicação serial, que está disponível nos pinos digitais 0 (RX) e 1 (TX). O 32U4 também permite a série (CDC) de comunicação através de USB e aparece como uma porta COM virtual para o software no computador, neste caso estamos a usar os pinos RX e TX para a comunicação com o computador. (Arduino, 2013)

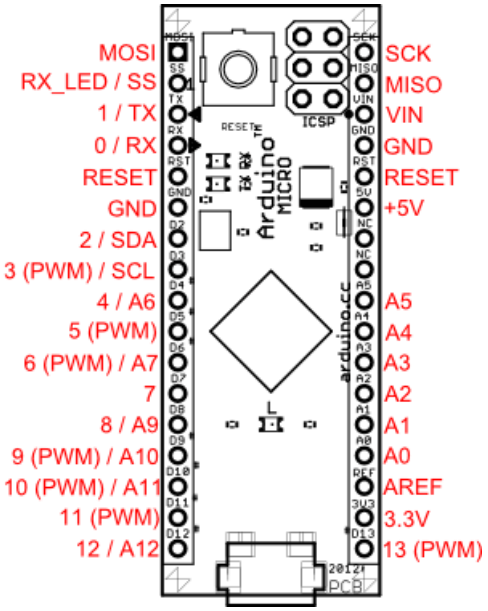


Figura 16: PinOut do Arduíno micro.

Capítulo 7

Implementação da solução

7.1 Introdução

Depois de fazer toda a análise de requisitos tornou-se mais fácil a realização da aplicação. Neste capítulo serão apresentadas algumas das partes mais importantes da aplicação, assim como algum código relevante.

7.2 Alert Bracelet

A aplicação desenvolvida tem o nome "Alert Bracelet", onde o "Alert" é de Alerta e "Bracelet" de pulseira. Ao desenvolver a aplicação teve-se o cuidado de utilizar o estilo do Windows 8, seguindo por isso todas as boas regras que estão disponíveis no *website* da Microsoft.

Algumas das principais características da aplicação:

- Ligação com o Arduíno através de um único botão, sem necessidade de definir nada primeiro.
- Introdução da hora de início
- Introdução da hora de repetição do alarme
- Verificação de todos os dados enviados
- Ler a resposta do Arduíno

Para conectar com o Arduíno apenas e só é necessário carregar no botão conectar sem ser necessário definir a COM ou a velocidade, estando já tudo definido no programa apenas terá que definir a hora de início e de repetição e carregar no botão de evitar e esperar pelo feedback se foi enviado sem erros ou se tem que enviar novamente.

Nesse feedback o Arduíno envia uma mensagem pela Porta Serial dizendo se os dados foram bem recebidos ou não, para o programa em C#, e esse programa vai mostrar essa mensagem numa listBox ao fundo preta como mostra a Figura 17.

Para implementar esta solução surgiu um problema, eu estou a usar o Arduíno Micro e este microcontrolador tem a Porta Serial virtualizada e não estava a conseguir enviar os dados corretos para o C#, para isso foi encontrada uma solução que foi utilizar um segundo cabo USB e ia ligar aos pinos do Arduíno RX e TX utilizando no programa do Arduíno a Porta Serial 1, e assim desta maneira já é possível enviar e receber todos os dados corretos.

Numero 1	Botão para conectar com o Arduíno
Numero 2	Botão para entrar na página do Facebook
Numero 3	Botão para enviar o código para o Arduíno
Numero 4	ListBox onde aparece o feedback do Arduíno

Figura 17: Funções do programa

7.3 Página Web

Foi criado um botão no programa que ao carregar la o utilizador entra numa página do facebook onde pode tirar qualquer dúvida, encontrar novas atualizações do programa ou encontrar mais informações sobre o mesmo.

Figura 18: Página de apoio.

7.4 Som no Arduíno

Para fazer o som com o Arduíno, estou a usar um speaker usados pelos velhos computadores, ligado ao Pin 12 do Arduíno, para ligar e desligar o som, só tenho que utilizar o "tone" e "noTone", dando diferentes valores a frequência do som é possível até fazer músicas com o Arduíno.

Figura 19: Speaker

Exemplo de um som criado no Arduíno:

```
tone (12,1400);  // 12 número do pin de saída, 1400 frequência do som

delay(150);  // durante 150 milissegundos

noTone (12);  // desliga o som da saída 12
```

Inicialmente estava a tocar a música fora do programa principal do Arduíno, e eram tocadas numa rotina de atendimento de interruepts, mas depois de testar, vim a verificar que durante o tempo que estava a música a tocar dentro dessa rotina a contagem do tempo no Arduíno parava, e se a música fosse um pouco maior podia ter atrasos significativos nos próximos alarmes.

O objetivo deste projeto era que fosse o mais preciso possível, onde as pessoas pudessem confiar, e isso não estava a ir de encontro aos meus objetivos, para isso encontrei outra solução onde mudei o lugar onde ia ser tocada a música passando esta agora para o programa principal e nas rotinas de atendimento de interrupts apenas é feita a ativação dos alarmes, metendo estes a true e todo o resto é feito dentro do programa principal.

7.5 Pacote de dados

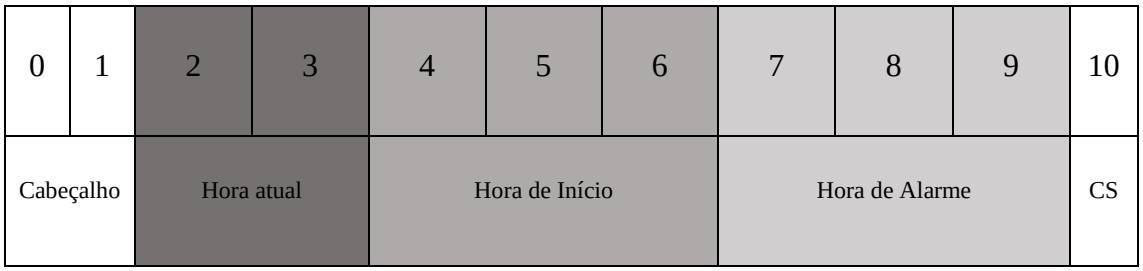


Figura 20: Pacote de dados enviados do C# para o Arduíno

Este esquema mostra-nos os dados que constituem os pacotes enviados do C# para o Arduíno formado por 11 Bytes.

Do Arduíno para o C# será enviado texto dizendo se foi bem recebido ou não assim como também o número de alarmes programados no Arduíno.

CS (CheckSum) é a soma de todos os bytes da hora Atual, com a soma de todos os bytes da hora de Início e da hora de Alarme. Este número é calculado no programa de C#, enviado para o Arduíno, calculado novamente no Arduíno e comparado com o valor recebido para termos uma maior garantia que todos os bytes recebidos estão corretos.

Com o seguinte código verificamos se estamos a enviar bem e a receber bem, enviando o "0" e o "255" vamos comparar esse valor e enviar novamente para o computador uma mensagem de erro, caso os valorem tenham sido mal recebidos.

A mensagem de "Bem recebido" só será enviada no fim de serem feitas as contas com o CheckSum para verificar se todos os dados foram bem recebidos.

Verificar cabeçalho

```
1:  if (Serial1.available() > 0) // Se há bytes para serem lidos na porta série
2:    {
3:      c = Serial1.read();
4:      if (c == 0 && estado == 0)    // Foi recebido o primeiro byte de sincronismo
5:      {
6:         estado++;
7:       }
8:     else if (c != 0 && estado == 0) // Recebido um byte de sincronismo diferente
   do esperado
9:      {
10:        Serial1.println("Erro de envio");
11:     }
12:    else if (c == 255 && estado == 1) // Foi recebido o segundo byte de sincronismo
13:     {
14:        estado++;
15:      }
16:    else if (c != 255 && estado == 1) // Recebido um byte de sincronismo diferente
   do esperado
17:     {
18:         Serial1.println ("Erro de envio");
19:         estado = 0; //Se der erro, voltamos ao estado 0 para voltar a receber os dados
   do inicio
20:      }
21: }
```

7.6 Envio do Pacote

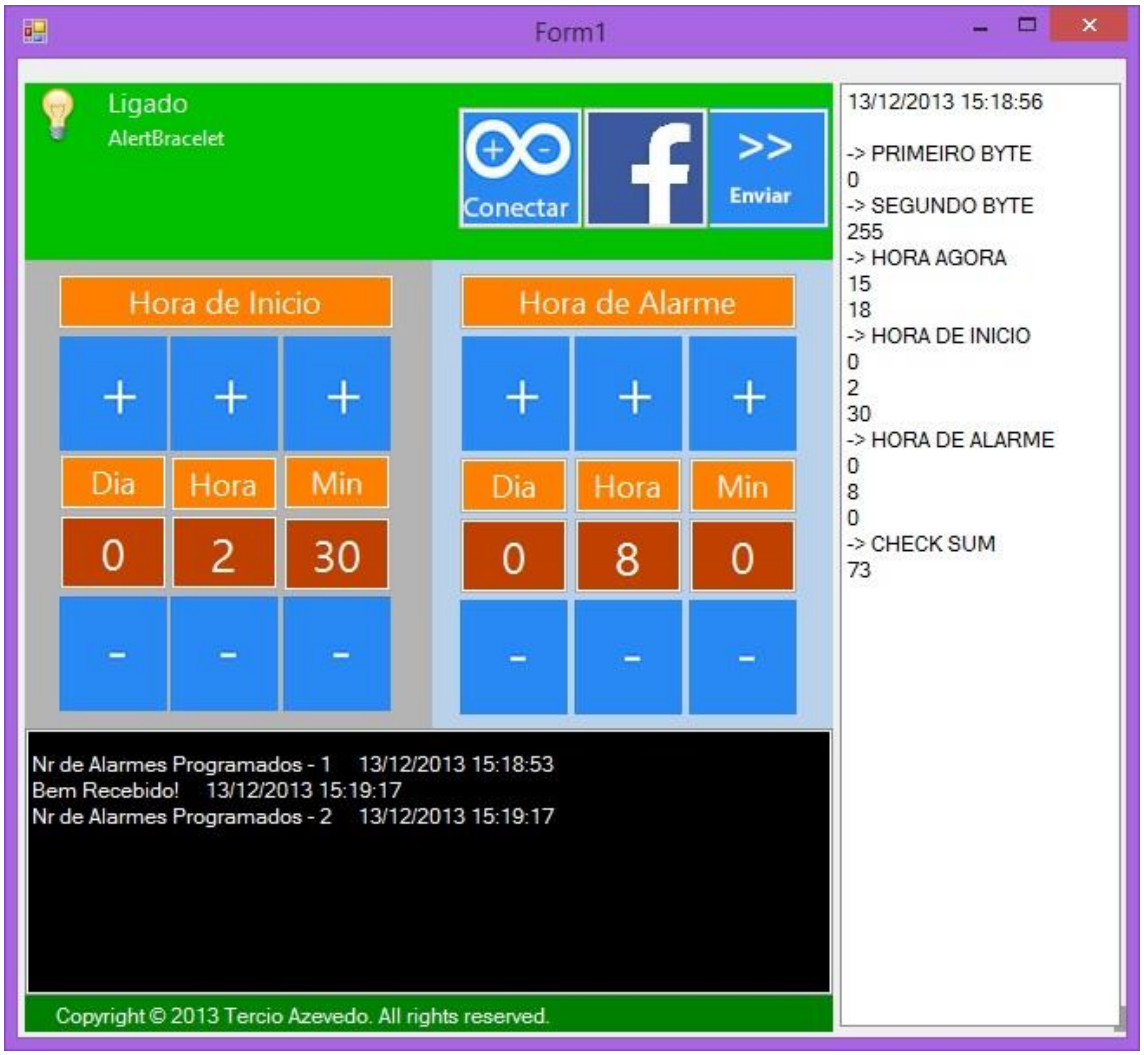
Supondo que daqui por uma hora e meia queríamos começar a tomar um antibiótico de 8 em 8 horas, para isso só é necessário inserir o 8 na hora de alarme e na hora de início o 2 e o 30 no respetivo campo para horas e para os minutos. De seguida é só carregar no botão enviar e esperar que não dê erro nenhum.

O erro pode ser de dois tipos: O Arduíno não esta conectado ao computador ou está mal conectado aparecendo uma mensagem com essa informação ou então surgiu algum problema no envio com o Arduíno e o Arduíno enviou uma mensagem de erro para o computador e é pedido ao utilizador para voltar a enviar de novo.

A Figura 20 mostra para este exemplo que tipo de informação vai ser enviada para Arduíno.

O CheckSum neste caso é de 73, que é a soma os bytes anteriores:

15 + 18 + 0 + 2 + 30 + 0 + 8 + 0 = 73



1º Byte	0
2º Byte	255
3º Byte	15
4º Byte	18
5º Byte	0
6º Byte	2
7º Byte	30
8º Byte	0
9º Byte	8
10º Byte	0
11º Byte	73

Figura 21: Informação do Pacote

7.7 Tempo

Para fazer a contagem do tempo estou a usar o #include<Time.h> e #include <TimeAlarms.h>.

Para isso foi necessário fazer download desta libraria e adiciona-la na pasta "libraries" do nosso programa para poder fazer os includes.

Para ter a hora atualizada no Arduíno utilizo o setTime e depois a hora de início atribuo à função Alarm.timerOnce que só toca uma única vez e de seguida uso a função Alarm.timerRepeat para repetir o alarme.

Estas funções têm como parâmetros de entrada: (Horas, Minutos, Segundos, Função).

Horas, Minutos e Segundos, são variáveis do tipo inteiro e a Função é o nome da função que pretendemos que seja chamada quando for atingida a hora do alarme. Esta função sendo chamada dentro de uma rotina de atendimento de interrupt deverá ter um código o mais curto possível, que demore o menos tempo possível a ser executado para evitar atrasos na contagem do tempo do Arduíno.

setTime – define a hora do Arduíno para a hora pretendida .

Alarm.timerOnce - Cria um timer que irá chamar uma função apenas uma vez em um determinado momento.

Alarm.timerRepeat - Cria um timer que irá chamar uma função várias vezes em um determinado momento.

Alarm.delay – os alarmes e timers são apenas verificações das funções chamadas quando se usa esta função de atraso. Podemos passar 0 para o mínimo de atraso. Este atraso deve ser usado em vez do atraso Arduíno normal (), para processamento em tempo útil de alarmes e timers.

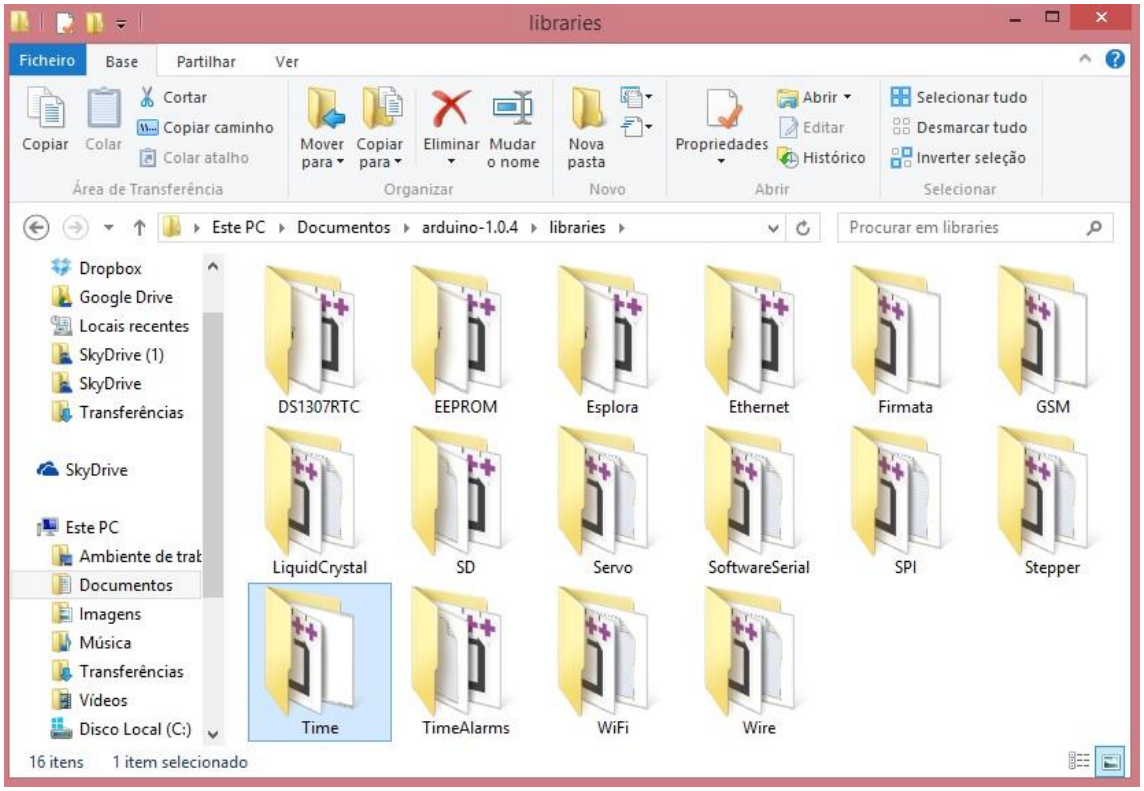


Figura 22: Libraria do Time e TimeAlarms.

Para fazer uma simulação do alarme usei o Serial.println e enviei a hora atual do microcontrolador, assim como os alarmes e fiz a simulação para dois medicamentos, o primeiro alarme foi de 5 em 5 segundos e o segundo alarme foi de 8 em 8 segundos, a partir da hora de início, e o resultado foi o que se segue.

Alarme Exemplo Arduíno

```
1: setTime(horaPc,minPc,0,1,1,13);
2:    // Cria os alarmes
3: Alarm.timerOnce(horaAlarm1, OnceOnly);
4: Alarm.timerRepeat(horaAlarm2, Repeats);
5:   Alarm.timerRepeat(horaAlarm3, Repeats2);
6:   }
7: void OnceOnly (){
8: Serial.println("Alarme, Hora de Inicio");
9: }
```

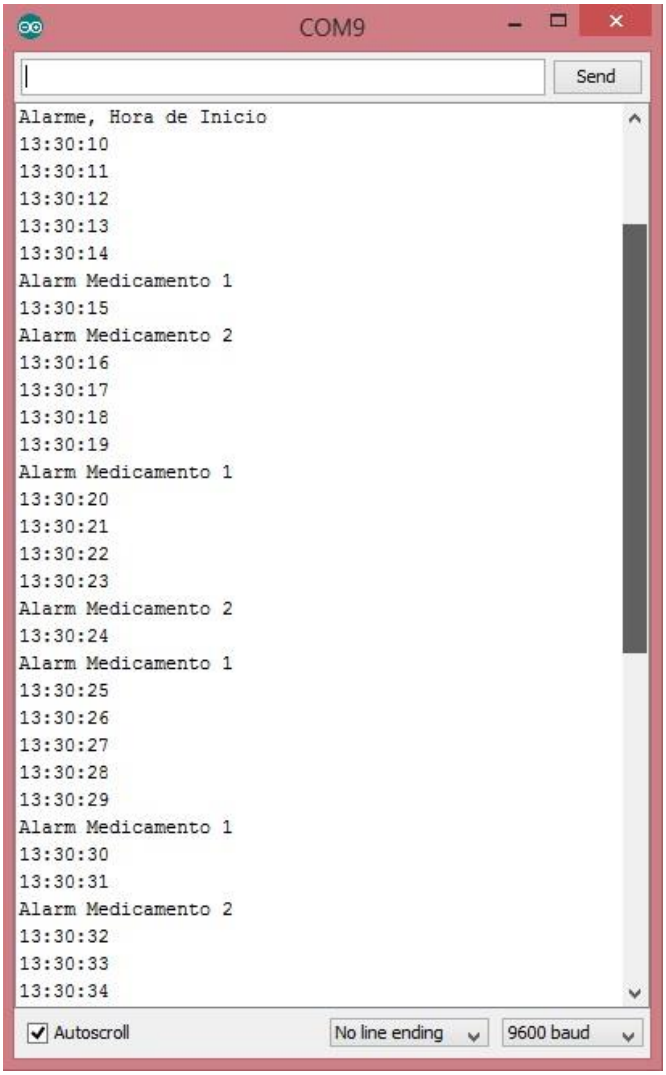


Figura 23: Serial Monitor do Arduíno

7.8 Programa de C#

No programa em C#, temos duas possibilidades de inserir os números, podemos usar os botões "+ " e "- " mais direcionado para ecrãs tátil, sem a necessidade do uso do teclado, como também a possibilidade de escrever diretamente o número pretendido no teclado numérico, estando todos os campos validados, só deixando escrever números.

No campo dos minutos só aceita números até 59, nas horas até 23 e nos dias até 255, se forem inseridos números superiores a estes não deixa, aparecendo uma mensagem a informar que o número é demasiado grande para ser inserido como mostra a Figura 24.

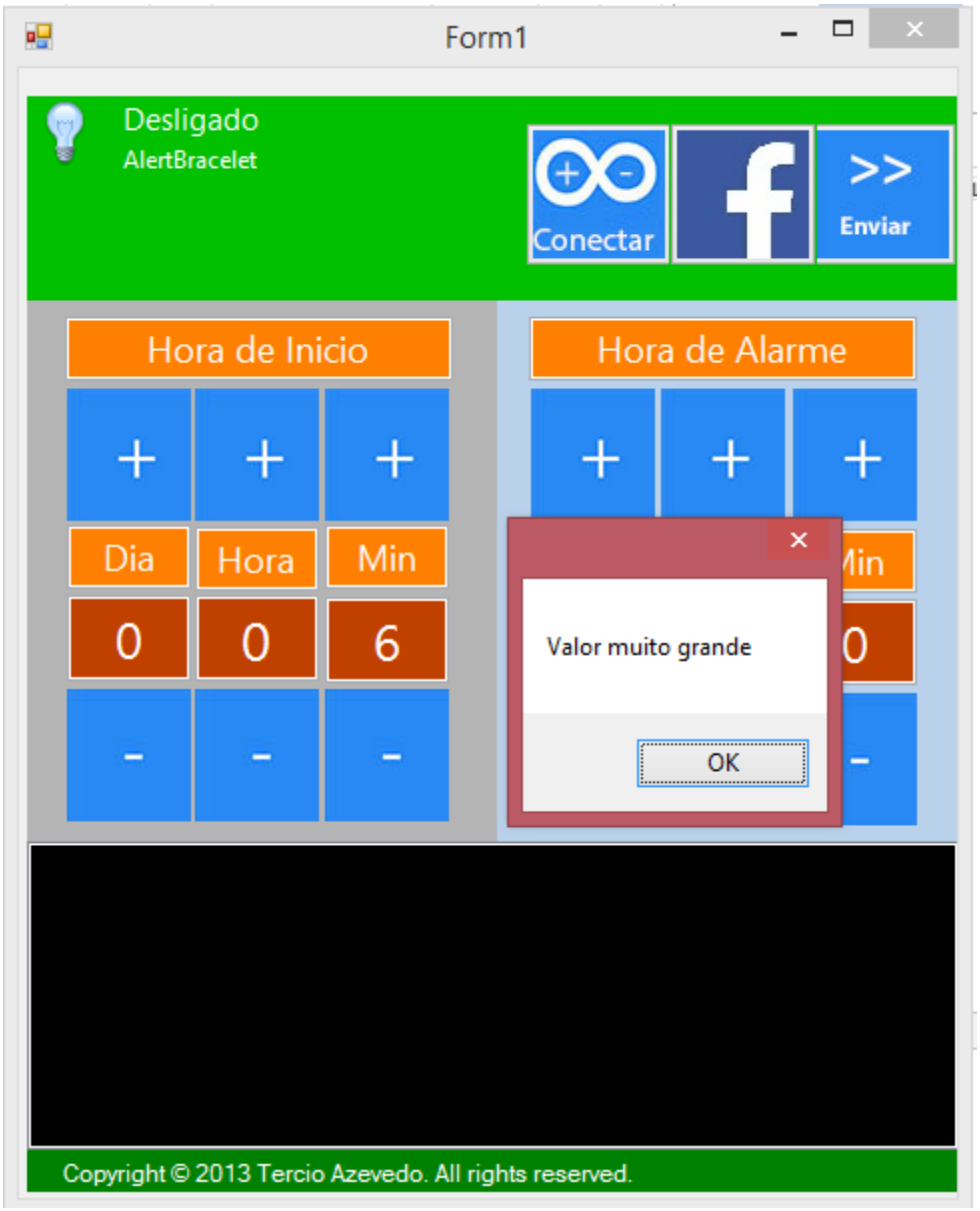


Figura 24: Validação dos campos

Capítulo 8

Conclusões e trabalho futuro

8.1 Conclusões

Penso que este trabalho que eu propus ao meu orientador no âmbito do projeto final de curso, foi muito pertinente pois ajudou-me muito a preparar como futuro profissional.

Durante a elaboração do projeto foram surgindo problemas, pois foi também a primeira vez que trabalhei com o Arduíno mas todos os problemas foram ultrapassados com sucesso. Com a elaboração deste projeto, foi possível aprender e aprofundar os meus conhecimentos em diversas áreas, nomeadamente na área do Arduíno como programar um microcontrolador, comunicar com ele, mete-lo a tocar ou até simplesmente a acender um led entre mil e uma coisas que se pode fazer com o Arduíno.

Este projeto foi um grande desafio para mim, principalmente para estabelecer um protocolo que comunicasse corretamente entre C# e o microcontrolador, pois neste tipo de projeto não posso ter falhas, apenas um número mal recebido podia alterar tudo, mas esse desafio foi superado com sucesso e além disso ainda consegui não só fazer para um alarme, mas para três com músicas distintas para cada um deles, podendo assim o utente associar cada música ao medicamento que precisa tomar.

As músicas criadas para os alarmes são simples, mas distintas pois o meu objetivo também não era a criação de músicas, embora que no Arduíno seja possível tocar qualquer melodia, assim como o aspeto final do projeto ainda não ser de uma pulseira, mas espero que num futuro próximo possa ver este projeto inovador numa farmácia.

De um modo geral, achei que este trabalho foi muito bom uma vez que alcancei os objetivos pretendidos e aprofundei vários temas importantes do curso sempre com a supervisão do meu orientador e agora ver todo este projeto funcionar corretamente deixa-me orgulhoso.

8.2 Trabalho futuro

Este projeto não fica por aqui. Existe planos de novas atualizações com novas funcionalidades.

Lista de funcionalidades previstas para breve:

- Avisar quando a bateria estiver fraca
- Guardar os dados recebidos na memória EPROM para quando falhar a bateria não haver perda de dados.
- Implementar o projeto numa pulseira
- Comercializar o projeto

Bibliografia

Arduino. (2013). Obtido de http://arduino.cc/

codebender. (2013). Obtido de http://codebender.cc

Marmitt, J. (2013). *Hora do Remédio*. Obtido de Google Play: https://play.google.com/store/apps/details?id=fontes.principal.horadoremedio&hl=pt_PT

Silveira, J. A. (2013). *Arduino Cartinha para programação em C*. Obtido de http://www.revistadoarduino.com.br/

Capítulo 9

Apêndice

9.1 Apêndice A

Programa do Arduíno

```
#include <Time.h>
#include <TimeAlarms.h>

//declaração das variaveis
int recebido[4][10];
int c;
int estado=0;
int frequencia = 0;
byte checksun = 0;
long int hInicio1;
long int hInicio2;
long int hInicio3;
long int hAlarme1;
long int hAlarme2;
long int hAlarme3;
boolean tocar1=false;
boolean tocar2=false;
boolean tocar3=false;
int posicaoVazia = -1;//a variavel é iniciada a -1 porque depois é incrementada e
passa a zero

 void setup(){
  Serial1.begin(9600);// velocidade da porta Serie
}

void loop() {
        if (Serial1.available() > 0) // Se há bytes para serem lidos na porta
série
        {
            c = Serial1.read();
            if (c == 0 && estado == 0)    // Foi recebido o primeiro byte de
sincronismo
            {
                estado++;
            }
            else if (c != 0 && estado == 0) // Recebido um byte de sincronismo
diferente do esperado
            {
                Serial1.println("Erro de envio 1");
            }
            else if (c == 255 && estado == 1) // Foi recebido o segundo byte de
sincronismo
            {
                posicaoVazia++;
```

```
                if (posicaoVazia == 3)
                {
                    posicaoVazia = 0;
                }
                estado++;
            }
            else if (c != 255 && estado == 1) // Recebido um byte de sincronismo
diferente do esperado
            {
                Serial1.println("Erro de envio 2");
                estado = 0;   //Se der erro, voltamos ao estado 0 para voltar a
receber os dados do início

          }
 //######################### FIM DO CABEÇALHO ##################################

            else if (estado > 1 && estado < 12) // queremos apenas os bytes de
dados e não os de cabeçalho e checksun
            {
                recebido[posicaoVazia][estado - 2] = c;

                estado++;

                if (estado == 11) // quando estado igual a 11 significa que os
numeros ja foram recebidos e colocados no array
                {
                    checksun = 0;
                    for (int i = 0; i < 8; i++) //aqui vai ser calculado o
checksum para verificarmos os dados recebidos
                    {
                        checksun = checksun + recebido[posicaoVazia][i];
                    }
                    if (checksun == recebido[posicaoVazia][8]) // compara o valor
da soma com o valor do byte 8 para saber se recebemos bem os dados
                    {
                        Serial1.println("Bem Recebido!");
                        Serial1.print("Nr de Alarmes Programados - ");
                        Serial1.println(posicaoVazia + 1); // aqui é feito +1
porque começa no 0 a 2 e eu quero de 1 a 3
                    } else {
                        Serial1.println("Erro de envio 3"); //houve erro no
checksum
                    }
                    setTime(recebido[posicaoVazia][0], recebido[posicaoVazia][1],
0, 1, 1, 13);
                    //aqui estou a defenir a hora no arduino, quando enviu
segunda vez a hora ja esta defenida,
                    // por isso se estiver 0 ou se estiver posicaoVazia vem a ser
a mesma coisa

                    hInicio1 = recebido[0][2] * 24 + recebido[0][3];
                    hAlarme1 = recebido[0][5] * 24 + recebido[0][6];
                    hInicio2 = recebido[1][2] * 24 + recebido[1][3];
                    hAlarme2 = recebido[1][5] * 24 + recebido[1][6];
                    hInicio3 = recebido[2][2] * 24 + recebido[2][3];
                    hAlarme3 = recebido[2][5] * 24 + recebido[2][6];

                  // Uma vez que também estou a enviar o numero de dias e a
função do Arduíno não recebe dias, apenas recebe horas minutos e segundos,
```

```
                // estou a pegar no numero de dias a multiplicar por 24 e a
somar ao numero de horas.
                // Depois esta variável é atribuída ás funções de alarme no
lugar das horas.

                  switch (posicaoVazia) {
                      case 0: {
                          Alarm.timerOnce(hInicio1, recebido[0][4], 0,
Inicio1);  // Cria um alarme sem repetição, atribui horas, minutos e segundos e
irá chamar uma função Inicio1
                      }
                      break;

                      case 1: {
                          Alarm.timerOnce(hInicio2, recebido[1][4], 0,
Inicio2); // Criar um alarme sem repetição, atribui horas, minutos e segundos e
irá chamar uma função Inicio2
                      }
                      break;

                      case 2: {
                          Alarm.timerOnce(hInicio3, recebido[2][4], 0,
Inicio3); // Criar um alarme sem repetição, atribui horas, minutos e segundos e
irá chamar uma função Inicio3
                      }
                      break;
                  }

                  estado = 0;
              }
          }
      }
      // Na rotina de atendimento de interrupts quando chega a hora de inicio
ou de alarme, as variaveis tocar1, tocar2 ou tocar3,
      // são postas a true e a musica é tocada dentro do programa principal
para não parar a contagem do tempo e podendo ser tocados
      // 2 ou os 3 alarmes em simultaneio (um a seguir ao outro) não correndo o
risco de nenhum deles ser perdido.

      if (tocar1 == true)// Foi ativado o alarme 1 através da rotina de
interrupt. O som e a mensagem é feita aqui
      {
          Serial1.println("Alarme um!");
          tone(12, 900); // com o tone e noTone faço um som, sendo 12 numero do
pin de saida e 900 valor da frequência
          delay(400);
          noTone(12);
          delay(300);
          tone(12, 900);
          delay(400);
          noTone(12);
          tocar1 = false;
      }
      if (tocar2 == true)
      {
          Serial1.println("Alarme dois!");
          tone(12, 1300); //// com o tone e noTone faço um som diferente, sendo
12 numero do pin de saida e 1300 valor da frequência
          delay(2000);
          noTone(12);
```

```
            tocar2 = false;
        }
        if (tocar3 == true)
        {
            Serial1.println("Alarme tres!");   // aqui é feita uma variação da
frequencia para nos dar um tipo de som diferente
            for (frequencia = 150; frequencia < 1800; frequencia += 1) {
                tone(12, frequencia);
                delay(1);
            }
            for (frequencia = 1800; frequencia > 150; frequencia -= 1) {
                tone(12, frequencia);
                delay(1);
            }
            noTone(12);
            tocar3 = false;
        }
        Alarm.delay(0); //os alarmes e timers são apenas verificações das funções
chamadas quando se usa esta função de atraso.
       // Podemos passar 0 para o mínimo de atraso. Este atraso deve ser usado em
vez do atraso Arduíno normal (), para processamento em tempo útil de alarmes e
timers.

    }
// ######################## FIM DO PROGRAMA PRINCIPAL############################

//****INICIO*****
    void Inicio1() {
        Alarm.timerRepeat(hAlarme1, recebido[0][7], 0, Alarme1); // Cria um
alarme com repetição, atribui horas, minutos e segundos e irá chamar uma função
Alarme1
        tocar1 = true;
    }

    void Inicio2() {
        Alarm.timerRepeat(hAlarme2, recebido[1][7], 0, Alarme2); // Cria um
alarme com repetição, atribui horas, minutos e segundos e irá chamar uma função
Alarme2
        tocar2 = true;

    }

    void Inicio3() {
        Alarm.timerRepeat(hAlarme3, recebido[2][7], 0, Alarme3); // Cria um
alarme com repetição, atribui horas, minutos e segundos e irá chamar uma função
Alarme2
        tocar3 = true;
    }

// ****ALARMES*****
    void Alarme1() {
        tocar1 = true;
    }
    void Alarme2() {
        tocar2 = true;
    }

    void Alarme3() {
        tocar3 = true;
   }
```

9.2 Apêndice B

Programa para Desktop

```
using System;
using System.IO.Ports;
using System.Windows.Forms;

namespace PortasCOM
{
    public partial class Form1 : Form
    {
        private SerialPort porta_serie;  // definição das variaveis a zero

        private int hora = 0;
        private int dia = 0;
        private int min = 0;
        private int Ihora = 0;
        private int Idia = 0;
        private int Imin = 0;

        public Form1()
        {
            InitializeComponent();
        }

        private void pictureBox1_Click(object sender, EventArgs e) // codigo do
botão para  aumentar o dia
        {
            dia++;
            if (dia > 255)
                dia = 0;
            ShowText();
        }

        /// <summary>
        /// Mostra os valores na caixas de texto
        /// </summary>
        private void ShowText()
        {

            horas.Text = Convert.ToString(hora);
            dias.Text = Convert.ToString(dia);
            minutos.Text = Convert.ToString(min);

            IniciarMin.Text = Convert.ToString(Imin);
            IniciarHoras.Text = Convert.ToString(Ihora);
            IniciarDia.Text = Convert.ToString(Idia);
        }

        /// <summary>
        /// Função para guardar os dados e enviar para o arduino
        /// </summary>
        private void ShowResult()
        {
            listboxInfo.Items.Clear();
```

```
        // Definir a hora atual
        DateTime horaAtual = DateTime.Now;
        listboxInfo.Items.Add(horaAtual.ToString());
        listboxInfo.Items.Add("");



        // POR OS VALORES TODOS EM APENAS UM ARRAY DE BYTES
        byte[] bytesEnviar = new byte[11];

        byte checkSum = 0;

        bytesEnviar[0] = 0;
        bytesEnviar[1] = 255;
        bytesEnviar[2] = (byte)horaAtual.Hour;
        bytesEnviar[3] = (byte)horaAtual.Minute;
        bytesEnviar[4] = Convert.ToByte(dias.Text);
        bytesEnviar[5] = Convert.ToByte(horas.Text);
        bytesEnviar[6] = Convert.ToByte(minutos.Text);
        bytesEnviar[7] = Convert.ToByte(IniciarDia.Text);
        bytesEnviar[8] = Convert.ToByte(IniciarHoras.Text);
        bytesEnviar[9] = Convert.ToByte(IniciarMin.Text);

        // Somar os bytes ao checkSum
        for (int i = 2; i < bytesEnviar.Length - 1; i++)
        {
            checkSum += bytesEnviar[i];
        }
        bytesEnviar[10] = checkSum;


        // MOSTRAR UM INFO DO QUE ESTA GUARDADO NO ARRAY DE BYTE
        listboxInfo.Items.Add("-> PRIMEIRO BYTE");
        listboxInfo.Items.Add(bytesEnviar[0]);

        listboxInfo.Items.Add("-> SEGUNDO BYTE");
        listboxInfo.Items.Add(bytesEnviar[1]);

        listboxInfo.Items.Add("-> HORA AGORA");
        listboxInfo.Items.Add(bytesEnviar[2]);
        listboxInfo.Items.Add(bytesEnviar[3]);

        listboxInfo.Items.Add("-> HORA DE INICIO");
        listboxInfo.Items.Add(bytesEnviar[4]);
        listboxInfo.Items.Add(bytesEnviar[5]);
        listboxInfo.Items.Add(bytesEnviar[6]);

        listboxInfo.Items.Add("-> HORA DE ALARME");
        listboxInfo.Items.Add(bytesEnviar[7]);
        listboxInfo.Items.Add(bytesEnviar[8]);
        listboxInfo.Items.Add(bytesEnviar[9]);

        listboxInfo.Items.Add("-> CHECK SUM");
        listboxInfo.Items.Add(bytesEnviar[10]);

        SendBytes(bytesEnviar);
    }
```

```
        private void btnDiasMenos_Click(object sender, EventArgs e)  // codigo do
botão para diminuir o dia
        {
            dia--;

            if (dia < 0)
                dia = 255;
            ShowText();
        }

        private void btnHorasMais_Click(object sender, EventArgs e) // codigo do
botão para aumentar a hora
        {
            hora++;
            if (hora > 23)
            {
                dia++;
                hora = 0;
            }
            ShowText();
        }

        private void btnHorasMenos_Click(object sender, EventArgs e) // codigo do
botão para diminuir a hora
        {
            hora--;
            if (hora < 0) hora = 23;
            ShowText();
        }

        private void btnMinMais_Click(object sender, EventArgs e) // codigo do
botão para aumentar os minutos
        {
            min++;
            if (min > 59)
            {
                min = 0;
                hora++;
            }
            ShowText();
        }

        private void btnMinMenos_Click(object sender, EventArgs e) // codigo do
botão para diminuir os minutos
        {
            min--;
            if (min < 0) min = 59;
            ShowText();
        }

        private void pictureBox8_Click(object sender, EventArgs e)
        {
            ShowResult();
        }


        private void SendBytes(byte[] bytesToSend)
        {
            // Verificar se a variavel da porta seria já foi construida
            if (porta_serie == null)
```

```
            {
                MessageBox.Show(this, "IMPOSSIVEL ENVIAR OS BYTES\n\n Arduino não
está conectado!", "ERRO", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                // Verificar se a porta serie esta connectada ao dispositivo
                if (porta_serie.IsOpen == true)
                {
                    porta_serie.Write(bytesToSend, 0, bytesToSend.Length); //
Enviar todos os bytes para o arduino
                    MessageBox.Show("ENVIADO! :)");
                }
                else
                {
                    MessageBox.Show(this, "IMPOSSIVEL ENVIAR OS BYTES\n\n Arduino
não está conectado!", "ERRO", MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
        }
        // verifica as teclas precionadas para os minutos se o numero foi maior
que 59
        private void minutos_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
            {
                int numero = Convert.ToInt32(minutos.Text + e.KeyChar);

                if (numero > 59)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
            catch { }
        }
        // verifica as teclas precionadas para as horas se o numero foi maior que
23
        private void horas_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
            {
                int numero = Convert.ToInt32(horas.Text + e.KeyChar);

                if (numero > 23)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
            catch { }
        }
        // verifica as teclas precionadas para os minutos do Iniciar se o numero
foi maior que 59
        private void IniciarMin_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
```

```
            {
                int numero = Convert.ToInt32(IniciarMin.Text + e.KeyChar);

                if (numero > 59)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
            catch { }
        }
        // verifica as teclas precionadas para as horas se o numero foi maior que
23
        private void IniciarHoras_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
            {
                int numero = Convert.ToInt32(IniciarHoras.Text + e.KeyChar);

                if (numero > 23)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
            catch { }
        }
        // verifica as teclas precionadas para os dias se o numero foi maior que
255
        private void dias_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
            {
                int numero = Convert.ToInt32(dias.Text + e.KeyChar);

                if (numero > 255)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
            catch { }
        }
        // verifica as teclas precionadas para os dias no Iniciar se o numero foi
maior que 255
        private void IniciarDia_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = isDecimal(e);
            try
            {
                int numero = Convert.ToInt32(IniciarDia.Text + e.KeyChar);

                if (numero > 255)
                {
                    MessageBox.Show("Valor muito grande");
                    e.Handled = true;
                }
            }
```

```
            catch { }
        }
        private bool isDecimal(KeyPressEventArgs tecla)
        {
            if ((tecla.KeyChar < (char)Keys.D0 || tecla.KeyChar > (char)Keys.D9)
&& tecla.KeyChar != (char)Keys.Back)
                return true;
            else
                return false;
        }

        private void BtHMais_Click(object sender, EventArgs e) // codigo do botão
para aumentar os minutos do iniciar
        {
            Imin++;
            if (Imin > 59)
            {
                Imin = 0;
                Ihora++;
            }
            ShowText();

        }

        private void BtMMenos_Click(object sender, EventArgs e) //codigo do botão
para diminuir os minutos do iniciar
        {

            Imin--;
            if (Imin < 0) Imin = 59;
            ShowText();
        }

        private void BtMMais_Click(object sender, EventArgs e) // codigo do botão
para aumemtar as horas do iniciar
        {
            Ihora++;
            if (Ihora > 23)
            {
                Ihora = 0;
                Idia++;
            }
            ShowText();

        }

        private void BtHMenos_Click(object sender, EventArgs e) // codigo do
botão para diminuir as horas do iniciar
        {
            Ihora--;
            if (Ihora < 0) Ihora = 23;
            ShowText();

        }

        private void BtDMais_Click(object sender, EventArgs e) // codigo do botão
para aumentar o dia
        {
            Idia++;
            if (Idia > 255)
```

```
                Idia = 0;
            ShowText();
        }

        private void BtDMenos_Click(object sender, EventArgs e) // codigo do
botão para diminuir o dia
        {
            Idia--;

            if (Idia < 0)
                Idia = 255;
            ShowText();
        }

        string indata;
        void PortaSerial_DataReceived(object sender, SerialDataReceivedEventArgs
e)
        {
            // função para ler a porta serie e meter na listBox
            if (porta_serie.BytesToRead != 0)
            {
                indata = porta_serie.ReadLine();

                // Correr em sincronização a instrução para adicionar dados a
lista
                this.BeginInvoke((Action)(() =>
                    {
                        listBox1.Items.Add(indata + "     " +
System.DateTime.Now);
                    }));
            }
        }


        private void button1_Click(object sender, EventArgs e)
        {
            ShowResult();
        }
        // codigo do botao para conectar ao arduino

        private void btnLigar_Click(object sender, EventArgs e)
        {
            if (porta_serie == null)
            {
                LigarPortaSerie();
            }
            else if (porta_serie.IsOpen == false)
            {
                LigarPortaSerie();
            }
        }

        /// <summary>
        /// Função para ligar a porta serie
        /// </summary>
        private void LigarPortaSerie()  // define os parametros da porta serie
        {
            porta_serie = new SerialPort("COM10", 9600);
            porta_serie.Parity = Parity.None;
            porta_serie.DataBits = 8;
```

```
            porta_serie.Handshake = Handshake.None;
            porta_serie.DataReceived += PortaSerial_DataReceived;

            try
            {
                porta_serie.Open();
                MessageBox.Show("LIGADO COM SUCESSO!");
                picLight.Image = Properties.Resources.Lightbulb_32;
                lblConnection.Text = "Ligado";
            }
            catch (Exception ex)
            {
                MessageBox.Show(this, "OCURREU UM ERRO AO LIGAR AO ARDUINO\nTENTE
OUTRA VEZ!!!\n\n" + ex.Message, "ERRO", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                picLight.Image = Properties.Resources.Lightbulb_Off_32;
                lblConnection.Text = "Desligado";
            }
        }

        // vai para a pagina oficial do facebook
        private void button1_Click_1(object sender, EventArgs e)
        {

System.Diagnostics.Process.Start("https://www.facebook.com/pages/Alert-
Bracelet/597036387022786");
        }


    }
}
```