Estado de Arte

Vitor Marques - A041449

UNIVERSIDADE
DA MAIA **UMAIA**

2024-05-21

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Sumário

2 Agradecimento

3 Introdução

3.1 Enquadramento Empresarial

A Filmservice International, com sede em Viena, Áustria, é a maior organizadora de festivais de filmes corporativos na Europa. Sob a liderança de Alexander V. Kammel, a empresa organiza vários eventos prestigiosos, incluindo os prémios 'AutoVision Awards' em Munique a cada dois anos, os 'Internationale Wirtschaftsfilmtage' em Viena, os prêmios anuais Cannes Corporate Media & TV Awards em Cannes e os US International Awards em Los Angeles. Além disso, Alexander V. Kammel é o cofundador e diretor do festival Grand Prix CIFFT, que reconhece os melhores filmes de turismo do mundo. Ele também atua como presidente dos Arquivos Austríacos para médias audiovisuais corporativas, educacionais e culturais, e organiza o 'Staatspreis Wirtschaftsfilm'.

Eu fui inserido, no escritório local da empresa, de maneira a incentivar a cooperação entre todos da empresa. Fui a uma entrevista onde falei das minhas capacidades e habilidades e também expliquei a razão por entrar no estágio com eles. Depois de um dia de preparação do material por ambas as partes, eu comecei a estagiar. Todos na empresa tem acesso à sua própria estação de trabalho e foi o mesmo para mim. Foi-me oferecido um computador e um espaço para montar o meu próprio escritório, ao lado dos demais empregados. Tive a liberdade de fazer a organização da forma que considerava melhor. A minha coordenadora e os demais empregados da empresa, estavam sempre disponíveis para resolver as minhas dúvidas e eu as deles. Isto deve-se ao facto de todos na empresa procurarem o trabalho em conjunto, onde cada um pode se focar naquilo que é melhor e também a ajudar os outros. Isto tem grandes resultados não só no ambiente de trabalho como também para com a eficácia do trabalho.

4 Estado de Arte

4.1 Open Source

4.1.1 Definição

Software de open source (Open Source) refere-se a software distribuído ao abrigo de licenças que concedem aos utilizadores acesso ao código-fonte, permitindo-lhes estudá-lo, modificá-lo e redistribuí-lo livremente(Androutsellis-Theotokis, 2010). Este conceito permite a uma comunidade colaborativa de programadores e utilizadores melhorar coletivamente a funcionalidade e a qualidade do software. As licenças de open source variam nos seus requisitos relativamente à distribuição de modificações, com algumas a exigirem que os trabalhos derivados sejam também de open source, enquanto outras permitem modificações proprietárias. O seu aparecimento teve um impacto significativo em vários sectores, incluindo o desenvolvimento de software, a inovação em hardware, o meio académico, a economia em geral e o desenvolvimento de websites(Androutsellis-Theotokis, 2010). Ao democratizar a tecnologia e a inovação, os modelos de open source oferecem oportunidades valiosas para a investigação, o envolvimento dos estudantes e o desenvolvimento de novos modelos empresariais e de software (Androutsellis-Theotokis, 2010).

A origem do software de open source remonta aos anos 50 e solidificou-se nos anos 70 e 80 com os esforços da cultura hacker, em particular através do projeto GNU iniciado por Richard Stallman (Jin, 2018). O conceito de open source gira em torno da ideia de partilhar e modificar livremente o código-fonte do software, em contraste com o software proprietário que restringe o acesso ao código-fonte (Jin, 2018). O movimento open source ganhou força com o desenvolvimento de projetos como o Linux, permitindo o trabalho colaborativo de software em comunidades globais.

- como evoluiu ao longo do tempo

Ao longo do tempo, o software de open source evoluiu para uma força dominante em vários sectores, incluindo a computação integrada, os dispositivos móveis e a nuvem, influenciando tanto o panorama tecnológico como as perceções culturais do desenvolvimento de software.

- Core Principles
- Diferenças para Software Proprietário
- como é desenvolvido, mantido e distribuido

4.1.2 Open Source vs. Free Software

- diferenças nas licenças, ownership e controlo

4.1.3 Benefícios do Open Source

O software de código aberto oferece uma flexibilidade e opções de personalização sem paralelo, permitindo aos utilizadores modificar e adaptar o código às suas necessidades específicas. Isto não só garante que as soluções sejam adaptadas para satisfazer requisitos únicos, como também promove um sentido de propriedade e controlo entre os utilizadores(Prokakis, 2022). Além disso, a comunidade de open source desempenha um papel vital na promoção da inovação, com os colaboradores a participarem ativamente na correção de erros, nos pedidos de funcionalidades e nos melhoramentos.

Uma das vantagens mais significativas do software open source é a sua relação custo-eficácia. Como está disponível gratuitamente, os indivíduos ou as organizações podem aceder-lhe sem incorrer em custos substanciais(Gediya, Singh, Kushwaha, Srivastava, & Wang, 2019). Isto não só reduz os encargos financeiros, como também permite às empresas afetar recursos de forma mais eficiente, conduzindo a um desenvolvimento mais rápido e a uma redução das despesas.

Além disso, o software de fonte aberta aumenta a privacidade e a transparência do utilizador, reduzindo a assimetria de informação entre os criadores e os utilizadores. A natureza colaborativa da comunidade de software open source garante que as soluções são fiáveis, seguras e continuamente melhoradas, o que resulta numa maior qualidade global do software.

Em geral, as vantagens do open source tornam-no uma opção atrativa para indivíduos e organizações que procuram soluções flexíveis, económicas e seguras que satisfaçam as suas necessidades específicas. Por exemplo, seguindo uma perspetiva de editores de vídeo, quando se utiliza um software proprietário como o Adobe Premiere Pro, este é vinculado aos seus termos de serviço. Tem um controlo limitado sobre os seus dados e podemos ser forçados a partilhá-los com empresas terceiras. Isso pode levar a preocupações sobre privacidade e segurança.

Por outro lado, o Shotcut é um software de edição de vídeo de código aberto que dá controlo total sobre os seus ficheiros e dados. É livre de o utilizar sem quaisquer restrições ou limitações. Com o Shotcut, podemos:

- **Manter os nossos vídeos privados:** Uma vez que o Shotcut não recolhe nem partilha dados do utilizador, dá para ter a certeza de que o trabalho criativo permanece confidencial.
- **Personalizar o software:** Como um projeto de código aberto, o Shotcut é orientado para a comunidade e permite que os utilizadores contribuam com código. Isto significa que existe a possibilidade de adaptar o software às necessidades específicas e ao fluxo de trabalho de cada um.

4.1.4 História do Open Source

- primeiros dias com a criação do GNU do Torvald e o Linux com o Linus

- ganhou popularidade entre 1990 e 2000 com o surgimento e popularidade das comunidades online e dos projetos colaborativos

> Texto do Antigo Relatório

A história do software de fonte aberta remonta aos primórdios da computação e ao aparecimento da cultura hacker. Na década de 1970, o conceito de software de código aberto começou a tomar forma, embora ainda não fosse formalmente designado ou amplamente reconhecido. Durante este período, o desenvolvimento de software era frequentemente um processo fechado e proprietário, com as empresas a guardarem o seu código e a restringirem o acesso ao mesmo. (Bretthauer, 2001)

Uma das figuras centrais no desenvolvimento inicial do software de código aberto foi Richard Stallman, fundador da Free Software Foundation. A defesa de Stallman da liberdade do software e a sua criação do Projeto GNU lançaram as bases do movimento do software de código aberto. Stallman acreditava na importância dos direitos dos utilizadores para aceder, modificar e distribuir software, o que levou ao desenvolvimento da Licença Pública Geral GNU (GPL). (DiBona, Ockman, & Stone, 1999)

O Projeto GNU, iniciado por Stallman nos anos 80, tinha como objetivo criar um sistema operativo livre e de código aberto do tipo Unix. Este projeto marcou um afastamento significativo do modelo de software proprietário prevalecente e enfatizou os princípios de colaboração, desenvolvimento orientado para a comunidade e liberdade de software. A visão de Stallman de um mundo em que os utilizadores tinham a liberdade de controlar a sua experiência de computação teve eco em muitos programadores e entusiastas, lançando as bases para o movimento de código aberto. (Stallman, 2002)

À medida que o Projeto GNU ganhava força, atraía uma comunidade crescente de programadores que partilhavam o compromisso de Stallman com os princípios do código aberto. Esta comunidade não só contribuiu com código para o projeto como também abraçou o espírito de partilha, colaboração e transparência que definia a cultura de código aberto. O Projeto GNU serviu de catalisador para o desenvolvimento de outras iniciativas de código aberto e inspirou uma nova geração de programadores a abraçar os valores da liberdade do software. (*What Is Copyleft? - GNU Project - Free Software Foundation*, n.d.)

Em paralelo com os esforços de Stallman, a cultura hacker desempenhou um papel crucial na formação do desenvolvimento inicial do software de código aberto. Os hackers, no sentido positivo do termo, eram programadores qualificados e amadores apaixonados que procuravam fazer avançar a ciência informática através da inovação colaborativa. O espírito dos hackers de partilhar conhecimentos, mexer no código e ultrapassar os limites da tecnologia estava estreitamente ligado aos princípios do software de fonte aberta. (Raymond, 1999)

> Texto Novo

O software de código aberto tem uma longa história que remonta aos primórdios da computação. No entanto, foi apenas na década de 1970 que o conceito de software de código aberto começou a tomar forma.

4.1.4.1 A Fundação da Free Software Foundation

Richard Stallman, fundador da Free Software Foundation, é uma das figuras centrais no desenvolvimento do software de código aberto. Ele defendeu a liberdade do software e criou o Projeto GNU, que lançou as bases do movimento do software de código aberto. A Licença Pública Geral GNU (GPL) foi criada para garantir que os utilizadores tivessem os direitos de acesso, modificação e distribuição do software. (DiBona et al., 1999)

4.1.4.2 O Projeto GNU

O Projeto GNU, iniciado por Stallman nos anos 80, teve como objetivo criar um sistema operativo livre e de código aberto do tipo Unix. O projeto marcou uma mudança significativa no modelo de software proprietário prevalecente e enfatizou os princípios de colaboração, desenvolvimento orientado para a comunidade e liberdade de software. (Stallman, 2002)

4.1.4.3 A Comunidade de Código Aberto

À medida que o Projeto GNU ganhava força, atraía uma comunidade crescente de programadores que partilhavam o compromisso de Stallman com os princípios do código aberto. Esta comunidade não só contribuiu com código para o projeto como também abraçou o espírito de partilha, colaboração e transparência que define a cultura de código aberto.

4.1.4.4 A Cultura Hacker

A cultura hacker desempenhou um papel crucial na formação do desenvolvimento inicial do software de código aberto. Os hackers, no sentido positivo do termo, eram programadores qualificados e amadores apaixonados que procuravam fazer avançar a ciência informática através da inovação colaborativa. (*What Is Copyleft? - GNU Project - Free Software Foundation*, n.d.)

4.1.4.5 Consequências

O movimento do software de código aberto inspirou uma nova geração de programadores a abraçar os valores da liberdade do software. Hoje em dia, o software de código aberto é amplamente utilizado e contribui para a inovação e a evolução da tecnologia.

4.1.5 Evolução da Filosofia Open Source

- como o termo open source evoluiu ao ser um termo mais neutro do que free software
- falar dos business models open source como a red hat, mysql e o wordpress conseguiriam monetizar os seus projetos
- mudanças nas licenças
- rise of github
- impacto do cloud computing no desenvolvimento open source

> Texto Antigo

A evolução da filosofia do software livre ao longo do tempo reflecte uma mudança significativa na abordagem ao desenvolvimento de software, dando ênfase à colaboração, transparência e envolvimento da comunidade. Esta evolução tem sido moldada pelas expectativas culturais e filosóficas que sustentam os projetos de software livre, bem como pela distinção entre software livre e software de código aberto.

O conceito de software de fonte aberta surgiu como uma resposta ao modelo tradicional de software proprietário, em que o código era mantido fechado e o acesso era restrito. A filosofia do software de fonte aberta defende a partilha do código-fonte, permitindo aos utilizadores ver, modificar e distribuir software livremente. Este espírito de abertura e transparência promove uma cultura de colaboração e inovação no seio da comunidade de desenvolvimento de software.

No centro da filosofia de código aberto está o princípio do desenvolvimento orientado para a comunidade. Os projectos de software livre baseiam-se nos esforços colectivos de um grupo diversificado de programadores, entusiastas e utilizadores que contribuem para a melhoria e evolução do software. Esta abordagem colaborativa não só acelera o ritmo da inovação, como também garante que o software se mantém acessível e adaptável a uma vasta gama de utilizadores.(Fuggetta, 2003)

Em contraste com o software de fonte aberta, o software livre coloca uma maior ênfase nas liberdades dos utilizadores e em considerações éticas. A Free Software Foundation, fundada por Richard Stallman, defende as quatro liberdades essenciais do software: a liberdade de executar o programa para qualquer fim, a liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades, a liberdade de redistribuir cópias e a liberdade de distribuir cópias das suas versões modificadas a outros.(*Free Software*, n.d.)

Embora o software livre e o software de código aberto partilhem objetivos comuns de promoção da liberdade de software e da colaboração, diferem nos seus fundamentos filosóficos. O software livre dá prioridade aos direitos dos utilizadores e a considerações éticas, enquadrando o software como uma questão de justiça social e de liberdades individuais. Por outro lado, o software de código aberto concentra-se mais nos benefícios práticos do desenvolvimento colaborativo, enfatizando as vantagens de eficiência, fiabilidade e segurança das metodologias de código aberto. (*Free Software*, n.d.)

A evolução da filosofia do software de código aberto tem sido marcada por um reconhecimento crescente da importância da transparência, do envolvimento da comunidade e da inovação partilhada no desenvolvimento de software. Ao adotar os princípios do software livre e de código aberto, tanto os programadores como os utilizadores contribuem para um ecossistema vibrante de criatividade, cooperação e avanço tecnológico.(*Why "Free Software" Is Better Than "Open Source" - GNU Project - Free Software Foundation*, n.d.)

4.1.6 Impacto no Desenvolvimento de Software ???

- aumentou a colaboração
- acelarou inovação
- democratização ao acesso a tecnologia (programas open source em africa?)
- muitas empresas adotam tecnologias open source para se manter competitivas
- mais qualidade e reliability - como os programas open source são mais estáveis e reliable devido ao feedback e teste da comunidade

4.2 CMS

https://www.oracle.com/pt/content-management/what-is-cms/

https://scandiweb.com/blog/what-is-cms-guide-to-content-management-systems/

4.2.1 Definição

Um Sistema de Gestão de Conteúdos (CMS) é uma ferramenta poderosa que permite criar, editar e gerir conteúdos digitais em websites sem a necessidade do utilizador ter conhecimentos técnicos avançados(Wilson, 2023). Exemplificando, podemos considerar um CMS como uma caixa de ferramentas virtual que ajuda a criar e organizar conteúdos online com facilidade. Imaginemos que estamos a construir uma casa nova. Não precisamos de ser arquitetos ou carpinteiros para desenhar a disposição e a estrutura da casa, certo? Da mesma forma, um CMS permite criar páginas Web, armazenar imagens e gerir outros ativos digitais sem escrever código de raiz. Isto significa que podemos nos concentrar na criação de conteúdos interessantes para o público, deixando o trabalho técnico pesado para o CMS. Um Sistema de Gestão de Conteúdos (CMS) distingue-se de outros softwares pela sua capacidade de facilitar a criação, edição e gestão de conteúdos digitais em webistes sem exigir conhecimentos técnicos especializados, ao contrário de outros tipos de software(Wilson, 2023). Esta característica que define os CMSs é o que lhes distingue dos demais softwares e isso os torna uma excelente opção para gerir websites, com as suas interfaces intuitivas.

Plataformas de gestão de conteúdo são uma solução robusta que ajuda, empresas a gerir o seu conteúdo digital em sites sem necessidade de habilidades técnicas avançadas. Elas funciona como uma caixa de ferramentas virtual, facilitando a criação e organização de conteúdo online (Wilson, 2023).

O sistema de gestão de conteúdo é um local único para armazenar informações e oferece processos automatizados para gestão e criação colaborativa, utilizando fluxos de trabalho incorporados. Isto permite que equipas trabalhem juntas de forma eficaz, com diferentes privilégios e responsabilidades atribuídos a indivíduos com base em funções (Wilson, 2023).

Por exemplo, autores podem publicar e armazenar o seu trabalho, enquanto editores podem modificá-lo e publicá-lo. Administradores podem realizar todas as tarefas, incluindo conceder permissão a outros membros da organização para atualizar ou rever conteúdo.

Com um sistema de gestão de conteúdo, empresas podem criar e gerir websites e conteúdo em sites com despesas técnicas mínimas, permitindo que criem melhores conteúdos sem precisar agir como gerente de projetos ou tráfego. Além disso, um CMS permite às empresas gerir e distribuir conteúdo sem investir numa equipa de desenvolvimento a tempo inteiro.

O melhor é que não é necessário ter habilidades técnicas avançadas para criar e gerir conteúdo digital, utilizando uma CMS. Isto significa que podemos nos concentrar na criação de conteúdo interessante para o público, deixando o trabalho técnico pesado para o sistema. Com um sistema de gestão de conteúdo, empresas podem se concentrar em melhorar a experiência do utilizador e aumentar a sua visibilidade online, sem precisar se preocupar com as complexidades técnicas.

- o que é um cms
- para que serve
- como se distingue dos outros softwares
- como ajuda a dar manage ao conteúdo do website
- falar que existem cms open source, proprietárias, hybrid????
- Uma breve história do desenvolvimento do CMS, destacando os principais marcos e inovações.
- Discussões sobre os prós e os contras da utilização de um CMS específico, tais como as compensações entre a facilidade de utilização e as opções de personalização.
- Análise da forma como as diferentes plataformas CMS se adaptam a sectores ou nichos específicos (por exemplo, comércio eletrónico, blogues).
- Uma secção sobre o futuro do desenvolvimento de CMS, incluindo tendências, tecnologias emergentes e potenciais alterações.

4.2.2 Vantagens de Usar um CMS

Na era digital atual, os sistemas de gestão de conteúdos (CMS) tornaram-se uma ferramenta essencial para a gestão e apresentação de conteúdos digitais. Ao fornecer uma plataforma de fácil utilização para criar, editar e armazenar conteúdos, os CMS revolucionaram como partilhamos conhecimentos e colaboramos com os outros(Wilson, 2023).

4.2.2.1 Facilidade de Criação e Gestão de Conteúdos

Uma das principais vantagens da utilização de um CMS é a sua facilidade de utilização. Com um CMS, podemos criar e gerir conteúdo digital sem precisar de ter grandes conhecimentos técnicos(Wilson, 2023). Por exemplo, digamos que é um programador Web a trabalhar num projeto para um museu. Pode utilizar um CMS como o WordPress ou o Drupal para criar exposições online interessantes que mostrem a coleção do museu. Com um CMS, pode carregar facilmente imagens, vídeos e texto, e personalizar a disposição e o design da sua exposição.

4.2.2.2 Escalabilidade e Flexibilidade

Outra vantagem da utilização de um CMS é a sua escalabilidade e flexibilidade(Wilson, 2023). Um CMS pode crescer com o seu projeto ou organização, permitindo-lhe adicionar novas características e funcionalidades conforme necessário. Por exemplo, se a exposição do seu museu online se tornar popular e receber inúmeros visitantes, pode facilmente aumentar o seu CMS para lidar com o aumento do tráfego.

4.2.2.3 Segurança e Manutenção Melhoradas

Embora as aplicações CMS como o WordPress, Drupal e Joomla tenham facilitado a criação e gestão de conteúdos digitais por parte das organizações, também apresentam riscos de segurança. Para mitigar estes riscos, é essencial manter o seu CMS atualizado com atualizações e manutenção regulares. Além disso, a implementação de práticas recomendadas, como a utilização de palavras-passe fortes, a atualização de plug-ins e temas e a criação regular de cópias de segurança dos dados, pode ajudar a evitar o acesso não autorizado e as violações de dados.

4.2.2.4 Melhor Colaboração e Fluxo de Trabalho

Um CMS também pode melhorar a colaboração e o fluxo de trabalho numa equipa ou organização. Por exemplo, pode utilizar um CMS para criar identificadores de projetos, armazenar referências entre projetos e itens de projetos e apresentar dados relacionados com projetos numa interface de fácil

utilização. Isto pode aumentar a produtividade e a organização na gestão de projetos, facilitando o trabalho eficaz das equipas.

Em conclusão, a utilização de um sistema de gestão de conteúdos oferece inúmeras vantagens, incluindo a facilidade de criação e gestão de conteúdos, escalabilidade e flexibilidade, segurança e manutenção melhoradas e melhor colaboração e fluxo de trabalho. Ao escolher o CMS adequado às suas necessidades e ao implementar as melhores práticas de segurança e manutenção, pode libertar todo o potencial dos conteúdos digitais e melhorar a sua presença em linha.

- facilidade de criação de contéudo e de management do mesmo
- Scalability and flexibility
- melhora segurança e maintenance e update
- melhor colaboração e worflow

4.2.3 Tipos de CMSs

Um Sistema de Gestão de Conteúdo (CMS) é composto por dois componentes essenciais: a área frontal e a área posterior.

A **área frontal**, também conhecida como frontend, é o ponto de interação do utilizador com o site. É aqui que a estrutura e o estilo do website são definidos, utilizando HTML, CSS e JavaScript para apresentar conteúdo rico e personalizado. A área frontal é responsável por fornecer uma experiência visual atraente e fácil de navegar.

Por outro lado, a **área posterior**, também conhecida como backend, é a aplicação que publica novos conteúdos no site. O processo começa quando o utilizador acessa uma interface web fácil de usar para adicionar, criar e publicar conteúdo na área frontal do CMS. Em vez de precisar de habilidades em HTML, CSS e JavaScript, o utilizador pode criar conteúdo num ambiente semelhante ao Microsoft Word.

A área posterior armazena o conteúdo no banco de dados e a pública na área frontal do website. Isso permite que os utilizadores publiquem conteúdo sem precisar entender tecnologias web ou construir a sua aplicação web desde o início.

Juntos, esses dois componentes formam o CMS, oferecendo uma solução escalável e flexível para gerir conteúdo em diferentes plataformas. A combinação da área frontal e da área posterior permite que os utilizadores criem e publiquem conteúdo de forma rápida e eficiente, sem precisar de habilidades técnicas especializadas.

4.2.3.1 CMSs Tradicionais

Um sistema de gestão de conteúdos (CMS) acoplado, ou tradicional, é uma escolha popular para aqueles que querem ter controlo total sobre o design e a funcionalidade do seu site. Este tipo de CMS oferece uma interface de fácil utilização que permite aos administradores gerir facilmente o conteúdo do website, a base de dados e o desempenho geral.

Uma das principais vantagens de um CMS tradicional é a sua capacidade de fornecer um back end totalmente acessível que se liga à base de dados do site e a modifica, permitindo a publicação contínua de conteúdos num front end com estilo (*Who, What, and Types of Content Management Systems?*, n.d.). Isto faz com que seja a escolha ideal para aqueles que querem ter controlo criativo sobre o design e a disposição do seu website.

No entanto, existem alguns inconvenientes na utilização de um CMS acoplado. Por exemplo, estes sistemas requerem frequentemente um alojamento Web dedicado para funcionar, o que pode aumentar o custo global de manutenção do website. Além disso, os administradores também têm de instalar e manter tecnologias específicas para tornar o software funcional, o que pode ser demoroso e exigir conhecimentos técnicos(*Who, What, and Types of Content Management Systems?*, n.d.).

Também, a instalação e configuração de um CMS acoplado pode ser um processo complexo que exige que o administrador tenha algum nível de conhecimento técnico. O WordPress é um exemplo popular de um CMS tradicional, oferecendo um pacote completo para os utilizadores instalarem, lançarem um website e publicarem conteúdos. Em geral, embora um CMS tradicional ofereça muitos benefícios, é essencial considerar as potenciais desvantagens antes de decidir se deve ou não utilizar este tipo de sistema.

Em conclusão, um CMS tradicional pode ser uma excelente escolha para aqueles que querem ter controlo total sobre o design e a funcionalidade do seu website. No entanto, os administradores devem estar conscientes dos potenciais inconvenientes e estar dispostos a investir tempo e recursos na manutenção do sistema.

4.2.3.2 Software as a Service CMSs

Um Sistema de Gestão de Conteúdos (CMS) baseado em SaaS oferece uma abordagem simplificada à criação e gestão de websites. Esta solução alojada na nuvem elimina a necessidade de configuração, instalação ou alojamento web pré-configurado, tornando-a uma opção atrativa para as empresas que procuram uma presença online simples(*Who, What, and Types of Content Management Systems?*, n.d.).

Ao tirar partido do poder da nuvem(*Who, What, and Types of Content Management Systems?*, n.d.), o SaaS CMS fornece uma solução completa que simplifica a criação, a gestão e a distribuição de conteúdos. Com a sua interface intuitiva, os utilizadores de várias origens podem criar rapidamente sites, gerir conteúdos e partilhá-los em canais digitais(Contentful, 2023). Esta flexibilidade é particularmente

útil para as empresas que procuram estabelecer uma forte presença online sem incorrer nos custos e complexidades associados às soluções CMS tradicionais.

Uma das principais vantagens do CMS SaaS é a sua escalabilidade. À medida que o seu negócio cresce, pode facilmente atualizar o seu plano para acomodar o aumento do tráfego ou das exigências de conteúdo. Esta escalabilidade também permite uma maior flexibilidade em termos de colaboração entre equipas, uma vez que vários utilizadores podem aceder e contribuir para o website em simultâneo("Headless CMS Explained in One Minute," n.d.).

Para além da sua facilidade de utilização e escalabilidade, o CMS SaaS também oferece funcionalidades de segurança robustas para proteger informações sensíveis. Com atualizações e cópias de segurança automáticas, pode ter a certeza de que os seus dados estão seguros e são facilmente recuperáveis em caso de um evento inesperado.

Para ilustrar as vantagens do CMS SaaS, considere uma pequena empresa de comércio eletrónico que pretende criar uma loja online. Ao tirar partido de uma solução baseada em SaaS, pode criar rapidamente um website com aspeto profissional sem necessitar de grandes conhecimentos técnicos. Isto permite-lhes concentrarem-se no que é mais importante - fornecer um excelente serviço ao cliente e construir a sua marca.

Em conclusão, o CMS SaaS representa uma abordagem revolucionária à criação e gestão de sítios Web. A sua facilidade de utilização, escalabilidade e características de segurança robustas tornam-no uma opção atrativa para as empresas que procuram uma presença online simples sem a necessidade de soluções técnicas complexas.

4.2.3.3 Decoupled CMSs

Num sistema de gestão de conteúdos (CMS) decoupled, a camada de apresentação é separada do backend(*Who, What, and Types of Content Management Systems?*, n.d.). Um sistema de entrega atua como intermediário entre o frontend e o backend, acedendo a este último via uma interface de programação de aplicações (API)[1].

Esta abordagem proporciona uma maior flexibilidade na interação com o conteúdo criado no backend. Por exemplo, considere uma organização que procura utilizar a sua biblioteca de conteúdos para um novo fim, como aplicações móveis(Masini, n.d.). Neste cenário, um decoupled CMS é uma solução atrativa porque permite o desenvolvimento de aplicações múltiplas e adaptáveis no frontend, mantendo a consistência no backend.

[1] Uma API liga programas ou componentes de computador, oferecendo um serviço a outro software. Trata-se de um documento que descreve como construir ou utilizar esta ligação. Ao contrário das interfaces de utilizador, as APIs ligam computadores ou software entre si. Não se destinam à interação humana direta, mas sim aos programadores que as integram no seu software. Uma API tem frequentemente diferentes partes que atuam como ferramentas ou serviços disponíveis para os programadores(Reddy, 2011).

Um exemplo desta flexibilidade é visto na utilização de APIs para integrar serviços de terceiros com um website ou aplicação. Ao tirar partido das API, os programadores podem incorporar sem problemas conteúdos e funcionalidades existentes nos seus projetos sem comprometer a integridade do material de origem original.

Além disso, as soluções decoupled empregam frequentemente princípios de conceção modular, permitindo a criação de componentes reutilizáveis que podem ser facilmente integrados em várias aplicações. Esta abordagem permite que as organizações se adaptem rapidamente às condições de mercado em mudança ou às necessidades dos clientes, modificando módulos específicos em vez de reconstruir sistemas inteiros.

Também, uma arquitetura dissociada proporciona uma maior escalabilidade e facilidade de manutenção, uma vez que os componentes individuais podem ser atualizados ou substituídos de forma independente sem afetar outras partes do sistema. Isto resulta num processo de desenvolvimento mais eficiente, tempo de inatividade reduzido e melhor desempenho geral do sistema. Ao adotar uma estratégia decoupled CMS, as organizações podem desbloquear novas possibilidades de reutilização de conteúdos, adaptabilidade e inovação, mantendo o controlo sobre os seus ativos digitais.

4.2.3.4 Headless CMSs

Os Headless CMSs são uma categoria mais recente de CMSs que se concentram apenas na gestão de conteúdos, sem estarem ligados a uma camada de apresentação específica . Contentful é um exemplo de um CMS headless. Neste tipo de configuração, o CMS fornece APIs para recuperar e manipular conteúdos, que podem ser consumidos por qualquer estrutura ou aplicação front-end. Esta abordagem oferece maior flexibilidade e escalabilidade.

Um Headless CMS oferece às organizações uma abordagem única à gestão de conteúdos. Ao contrário dos sistemas CMS tradicionais, não possui uma interface de front-end e, em vez disso, baseia-se em aplicações personalizadas para aceder e apresentar conteúdos(Contentful, 2023).

Uma das principais vantagens de um CMS sem interface é a sua flexibilidade. Ao dissociar o sistema back-end da aplicação front-end, os programadores têm um maior controlo sobre como o seu conteúdo é apresentado. Isto permite um elevado grau de personalização, tornando-o uma opção atrativa para organizações com necessidades de conteúdo complexas ou específicas(“Headless CMS Explained in One Minute,” n.d.).

Por exemplo, uma empresa que pretenda criar uma aplicação móvel e um website com capacidade de resposta pode utilizar um CMS sem interface para armazenar o seu conteúdo de forma centralizada, ao mesmo tempo que fornece diferentes camadas de apresentação para cada aplicação. Esta abordagem também permite que os programadores reutilizem conteúdos em várias plataformas, reduzindo a necessidade de esforços duplicados.

Outra vantagem de um Headless CMS é a sua escalabilidade. À medida que as necessidades de conteúdo de uma organização evoluem, esta pode facilmente adicionar ou modificar aplicações front-end sem afetar o sistema back-end subjacente(Contentful, 2023). Isto faz com que seja uma solução ideal para as empresas que registam um crescimento rápido ou exigências variáveis.

Para além da sua flexibilidade e escalabilidade, um Headless CMS também proporciona um maior controlo da segurança e do acesso. Ao separar o sistema back-end da aplicação front-end, os programadores podem implementar medidas de segurança mais rigorosas para proteger dados sensíveis("Headless CMS Explained in One Minute," n.d.). Isto é particularmente importante para as organizações que lidam com informações confidenciais ou que trabalham com sectores regulamentados. Embora um Headless CMS ofereça muitas vantagens, também requer mais trabalho do que os sistemas CMS tradicionais. Os programadores têm de conceber, criar e ligar aplicações front-end personalizadas, o que pode ser demorado e exigir muitos recursos. No entanto, a flexibilidade e a escalabilidade oferecidas por esta abordagem fazem com que seja um investimento que vale a pena para as organizações que procuram um maior controlo sobre o seu conteúdo. Em conclusão, um Headless CMS é uma solução poderosa para as organizações que necessitam de controlo e flexibilidade totais sobre como o seu conteúdo é acedido e apresentado. Ao fornecer armazenamento de conteúdo centralizado, capacidades organizacionais e a liberdade de conceber aplicações front-end personalizadas, oferece benefícios sem paralelo para as empresas que procuram elevar a sua presença digital.

4.2.3.5 Diferenças de Cada Tipo de CMS

Tabela 1: Diferenças de Cada Tipo de CMS

Funcionalidade	CMS Tradicional	CMS Headless	Decoupled CMS
Gestão de Conteúdos	Acoplamento estreito com a camada de apresentação	Centra-se exclusivamente na gestão de conteúdos	Gere a criação e a edição de conteúdos
Camada de Apresentação	Estreitamente ligado ao CMS	Separada do CMS, utiliza APIs para recuperação de conteúdos	Camada de apresentação separada para renderização e interface do utilizador
Flexibilidade	Flexibilidade limitada em termos de apresentação	Elevada flexibilidade em termos de apresentação	Elevada flexibilidade em termos de conteúdo e de apresentação

Funcionalidade	CMS Tradicional	CMS Headless	Decoupled CMS
Escalabilidade	Escalabilidade limitada devido ao acoplamento estreito	Escalável devido à separação de interesses	Escalável devido a uma camada de apresentação separada
Personalização	Opções de personalização limitadas para a apresentação	Opções de personalização elevadas para a apresentação	Opções de personalização elevadas para conteúdo e apresentação
Integração de API	Capacidades limitadas de integração de API	Fortes capacidades de integração de API	Fortes capacidades de integração de API

Legenda

- **Definição**: Breve descrição do tipo de CMS.
- **Desenvolvimento**: Requisitos de conhecimento e habilidades para desenvolver o CMS ou aplicação.
- **Implantação**: Complexidade e tempo necessários para implantar e configurar o CMS ou aplicação.
- **Manutenção**: Frequência e complexidade da manutenção do CMS ou aplicação.

4.2.3.6 CMSs de código aberto, proprietários e híbridos

Outra consideração importante ao selecionar um CMS é o facto de ser de código aberto, proprietário ou híbrido. Os CMS de código aberto, como o WordPress, oferecem flexibilidade de personalização e apoio da comunidade, enquanto os CMS proprietários, como o Adobe Experience Manager, oferecem funcionalidades e apoio empresarialmente. Os CMS híbridos combinam as vantagens de ambos os mundos.

- diferentes categorias de CMS
- tradicional (wordpress - talvez arranjar outro exemplo)
- headless CMS (contentful)
- decoupled cms (adobe experience manager)
- talvez falar de open source e proprietary cms e hybrid?

4.2.4 Alternativas cms open source no mercado atual

- wordpress

- joomla
- drupal
- magento (e-commerce)
- ghost

organizar numa tabela

- licensign
- ease of use
- customization
- community support
- security
- scalability
- integration
- development
- cost
- documentation and tutorials

4.2.5 porquê wordpress?

> A TABELA ESTA NO Users-Miraiy-Zotero-storage-QU5IZE73 PAG.37

- populariedade
- suporte da comunidade
- scalability, and flexibility
- facilidade de uso para iniciantes
- biblioteca enorme com plugins e temas
- updates constantes e melhorias pela comunidade

4.3 WordPress

4.3.1 Definição e História do Wordpress

- origens do wordpress
- fundadores do wordpress
- como evoluiu ao longo dos anos
- core features (php based e integração com database MySQL)
- o que é o wordpress (pedir para não repetir aquilo que já foi falado)

4.3.2 Arquitetura do wordpress

- aspetos técnicos do wordpress
- request-response cycle
- mecânicas de cache
- como trabalha os inputs do utilizador

4.3.3 Customização e Extensibilidade

- plugins
- temas
- código php
- importância de otimização de código para melhorar performance
- deixar responsivo??

4.3.4 Melhores Práticas de Segurança

- proper password management
- updates regulares
- plugins de segurança como wordfence e malcare

4.3.5 Melhores Práticas para Construir um Website Open Source com WordPress

- design responsivo (mobile first e deve adaptar em diferentes ecrãs)
- otimização de imagens (compressão de imagens, lazy loading, formatos de imagem otimizado)
- organização do conteúdo (estrutura organizada de contéudo, com tags, categorias, descrições metas. ajuda os search engines a crawl pelo site melhorando a user experience
- caching e otmização (cache de páginas, posts, e outros conteúdos para reduzir load times. também falar de minifying code, comprimir ficheiros e leveraging browser caching)
- segurança e updates (importância de updates regulares, patches de segurança e plugin management. desabilitar plugins datados, usar passwords fortes e monitorizar malware)
- performance monitoring (GTmetrix, Pingdom, ou WebPageTest. como identificar bootlenecks e otimizar a performance do website)
- acessibilidade e SEO (WCAG guidelines, SEO. Dicas de como criar conteúdo acessível, usar semantic HTML, otimizar tags meta)
- backups e recovery (importância de backups regulares, guia de estratégias de backup (3,2,1 rule?) plugins como updraftPlus e VaultPress para automatizar backups e data recovery em casos de emergência)

- version control e git (conceito de controlo de versão com git e como pode ajudar devs a dar manage em mudanças de código, track revisions e colaborar com outros)

4.3.6 Exemplos reais de websites open source bem sucedidos criados com o WordPress

- mostrar features únicas
- sucessos
- desafios

Androutsellis-Theotokis, Stephanos (2010). Open Source Software: A Survey from 10,000 Feet. *Foundations and Trends® in Technology, Information and Operations Management*, *4*(3-4), 187–347. https://doi.org/10.1561/0200000026

Bretthauer, David (2001). Open Source Software: A History. *Published Works*. Retrieved from https://digitalcommons.lib.uconn.edu/libr_pubs/7

Contentful (2023). *Headless CMS explained in 1 minute | Contentful*. Retrieved from https://www.youtube.com/watch?v=bGMo1XCNQMo

DiBona, Chris, Ockman, Sam, & Stone, Mark (Eds.) (1999). *Open sources: Voices from the open source revolution* (1st ed). Beijing ; Sebastopol, CA: O'Reilly.

Free Software: Freedom and Cooperation - GNU Project - Free Software Foundation (n.d.). Retrieved from https://www.gnu.org/philosophy/rms-nyu-2001-transcript.html

Fuggetta, Alfonso (2003). Open source software––an evaluation. *Journal of Systems and Software*, *66*(1), 77–90. https://doi.org/10.1016/S0164-1212(02)00065-1

Gediya, Jagdish, Singh, Jaswinder, Kushwaha, Prabhakar, Srivastava, Rajan, & Wang, Zening (2019). 7 - Open-Source Software. In Robert Oshana & Mark Kraeling (Eds.), *Software Engineering for Embedded Systems (Second Edition)* (pp. 207–244). https://doi.org/10.1016/B978-0-12-809448-8.00007-2

Headless CMS explained in one minute (n.d.). Retrieved from https://www.contentful.com/headless-cms/

Jin, Zhi (2018). Open Models: Beyond the Open Source Software Development. *ACM SIGSOFT Software Engineering Notes*, *43*(4), 9–12. https://doi.org/10.1145/3282517.3282522

Masini, Jason (n.d.). *What is a decoupled CMS?* Retrieved from https://www.sitecore.com/knowledge-center/digital-marketing-resources/what-is-a-decoupled-cms

Prokakis, Emmanouil (2022). Free and Open-Source Software: Freedom, Transparency and Efficiency in the Digitalization Era. *Journal of Politics and Ethics in New Technologies and AI*, *1*(1), e31230–e31230. https://doi.org/10.12681/jpentai.31230

Raymond, Eric (1999). The cathedral and the bazaar. *Knowledge, Technology & Policy*, *12*(3), 23–49. https://doi.org/10.1007/s12130-999-1026-0

Reddy, Martin (2011). *API design for C++*. Amsterdam Heidelberg: Morgan Kaufmann.

Stallman, Richard (2002). *Free software, free society: Selected essays of Richard M. Stallman*. Retrieved

from https://books.google.com/books?hl=pt-PT&lr=&id=UJlNAgAAQBAJ&oi=fnd&pg=PA1&dq=Free+Software,+Free+Society&ots=bPpl3ZrMlA&sig=QBWGVtQACqBNYemf8T2cVNnzvq4

What is Copyleft? - GNU Project - Free Software Foundation (n.d.). Retrieved from https://www.gnu.org/licenses/copyleft.html

Who, what, and types of content management systems? (n.d.). Retrieved from https://www.oracle.com/pt/content-management/what-is-cms/

Why "Free Software" is better than "Open Source" - GNU Project - Free Software Foundation (n.d.). Retrieved from https://www.gnu.org/philosophy/free-software-for-freedom.html

Wilson, Kevin (2023). Content Management Systems. In Kevin Wilson (Ed.), *The Absolute Beginner's Guide to HTML and CSS: A Step-by-Step Guide with Examples and Lab Exercises* (pp. 181–201). https://doi.org/10.1007/978-1-4842-9250-1_9