SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA DEPARTAMENTO DE COMPUTAÇÃO – DCOMP

Departamento de Computação/UFS

SISTEMAS OPERACIONAIS
PROF. CALEBE
CONCEIÇÃO

Aula 4 - Exercícios sobre o documentário

Revolution OS - Documentário sobre GNU/Linux - Legendado em PT-BR

Atividade avaliativa, conforme plano de ensino apresentado. Atente-se aos prazos descritos no AVA.

Aluna(o): Guilherme Menezes de Azevedo Turma: T02

Questão 1) O termo free software (software livre) é sinônimo de domínio público? Explique. (3:12) - (3:55) - (06:40) - (11:00) - (12:30) - (15:10) - (17:00) - (17:12) - (18:25)

Resp.: Antes de começar respondendo essa pergunta devemos definir o que é domínio público? O domínio público é uma condição jurídica que permite o uso livre de obras intelectuais, artísticas, culturais, científicas, folclóricas e cinematográficas. [1] Segundo o documentário Revolution OS, o termo Free Software é referente ao software que é distribuído aos usuários a liberdade de executar, estudar, modificar e distribuir o mesmo. Sendo assim, o termo Free Software não pode ser considerado como domínio público, pois não está associado ao movimento do Software livre devido ao fato de não haver um controle sobre como ele é modificado e redistribuído.

A Free Software Foundation (FSF) é uma organização fundada por Richard Stallman em 1984 logo após se demitir do MIT, o qual tem como objetivo promover a liberdade de uso, estudo, modificação e distribuição do software. [Início do Projeto GNU (Gnu not Unix)]

Algo a mais que foi pesquisado sobre o FSF, complementando ao apresentado no documentário:

As quatro liberdades essenciais do software livre, conforme definidas pela FSF, são: [2]

1. **A liberdade de executar o programa para qualquer propósito.**
2. **A liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades.**
3. **A liberdade de redistribuir cópias do programa.**
4. **A liberdade de distribuir cópias das suas versões modificadas a outros.**

A licença pública Geral GNU (GPL) foi uma forma de criar um direito autoral reverso, segundo Stallman. Nos termos da GPL teria que a pessoa ao redistribuir o software modificado garantir também os direitos essenciais da FSF. OBS: Linus Torvalds usou essa licença ao desenvolver o Linux.

Questão 2) Enumere os argumentos usados por Bill Gates para sustentar seu ponto de vista na carta endereçada à comunidade de lobistas da Califórnia, o Homebrew Computer Club. (07:30)

Resp.:

1 - Esforço e Custo de Desenvolvimento: “O desenvolvimento de software de qualidade envolve um esforço significativo e custos substanciais. Ele destacou que programadores dedicam muito tempo e recursos para criar software, e que eles deveriam ser compensados por seu trabalho.”

2 - Desincentivo à Inovação: “A distribuição gratuita de software desincentiva os desenvolvedores de criar novos programas. Se os programadores não pudessem obter retorno financeiro pelo seu trabalho, eles não teriam motivação para inovar e desenvolver software de alta qualidade.”

3 - Equidade e Justiça: “É injusto para os desenvolvedores verem seu trabalho sendo distribuído gratuitamente sem nenhuma compensação. Ele comparou a situação ao roubo, afirmando que as pessoas que copiam software sem pagar estavam essencialmente roubando dos desenvolvedores.” (Mesmo tendo “copiado” em 1985 a ideia do Projeto Macintosh kkkkkk)

4 - Sustentabilidade da Indústria: “A importância de um modelo de negócios sustentável para a indústria de software. Ele acreditava que, sem um mercado onde o software fosse vendido, a indústria não poderia crescer e se desenvolver adequadamente.”

Questão 3) É correto afirmar que antes do lançamento do Linux não existia kernel disponível para a comunidade simpática ao software livre usar os programas disponibilizados pela GNU Project, da Free Software Foundation?

Resp.: Antes de responder essa pergunta será apresentado pontos importantes apresentados no documentário.

- Projeto GNU e a Necessidade de um Kernel: Richard Stallman começou o Projeto GNU em 1983. Ele pretendia criar um sistema operacional completo e gratuito para substituir o Unix. Nos anos seguintes, construíram muitos componentes importantes, desde compiladores até editores de texto e utilitários. No entanto, o projeto falhou em um componente crítico: um kernel. O GNU Hurd seria a alternativa de kernel para o sistema operacional GNU. No entanto, Hurd enfrentou muitos atrasos e problemas técnicos, então não estava pronto para uso prático.

- Kernels Disponíveis Antes do Linux: antes do Linux, a comunidade de software livre tinha

acesso a alguns kernels que poderiam ser usados com os programas GNU. No entanto, alguns deles não eram inteiramente gratuitos ou úteis para todos os propósitos possíveis. Minix: esse sistema operacional era parecido com o Unix, foi desenvolvido pelo educador Andrew S. Tanenbaum para fins educacionais, apesar de ter sido pequeno e eficiente, ele não era totalmente gratuito, uma vez que a implementação não podia ser modificada ou redistribuída.[3] BSD : em meados da década de 1980, tornou-se a forma de distribuições de sistemas operacionais Unix inteiros, uma vez que as versões do sistema BSD Unix permitem aos usuários a quantidade máxima de software possível.[4] No entanto, as versões iniciais eram ainda parte da licença da AT&T e, portanto, era mais complicado usar o sistema.

- Em relação ao Linux: No ano de 1991, Linus Torvalds lançou a versão de seu kernel Linux, que se tornou rapidamente o kernel favorito da comunidade de software livre. O Linux é livre, sob a Licença Pública Geral GNU é, combinado com os programas do GNU, um sistema operacional completamente funcional. O lançamento do Linux foi fundamental porque, ao fornecer à comunidade de software livre um kernel verdadeiramente livre e robusto, possibilitou a utilização prática e eficaz do sistema operacional GNU.[5]

  Podemos então responder que antes do linux já existia kernels disponíveis, entretanto não eram tão eficientes como o sistema desenvolvido pelo Torvalds.

Questão 4) Explique o modelo de negócios das primeiras empresas baseadas em software livre. (19:30) - (20:36) - (22:25)

Resp.: Será apresentado os modelos de negócios mencionados no documentário e em leituras complementares.[6]

- **Serviços e Suporte**: Muitas das primeiras empresas de código aberto, como a Red Hat e a Cygnus Solutions, baseiam seus modelos de negócios na venda de serviços e suporte técnico. Fornecem serviços como instalação, configuração, manutenção e suporte técnico especializado para suas distribuições de software livre. Isso permitiu que as empresas gerarem receita enquanto mantinham o software livre e aberto.

- **Consultoria**: Além do suporte técnico, essas empresas prestam consultoria para ajudar outras empresas a implementar soluções baseadas em software de código aberto. Os serviços de consultoria incluem análise de requisitos, planejamento de infraestrutura, customização de software e treinamento da equipe de TI.

- **Software Customizado**: Empresas como a Cygnus Solutions também se especializaram no desenvolvimento de software personalizado. Elas eram contratadas por outras empresas para adicionar funcionalidades específicas a softwares livres existentes ou

desenvolver novos softwares de acordo com as necessidades do cliente.

- **Documentação**: Algumas empresas, como a Red Hat, criaram distribuições de sistemas operacionais baseados em Linux, que incluíam não apenas o software livre, mas também documentação abrangente, manuais de usuário, e ferramentas de instalação simplificadas. Essas distribuições eram vendidas em pacotes, muitas vezes acompanhadas por assinaturas que incluíam atualizações e suporte.

- **Parcerias e Colaborações**: Essas empresas frequentemente estabeleciam parcerias com outras empresas de tecnologia, governos, e instituições educacionais. As parcerias poderiam envolver colaborações em projetos de desenvolvimento de software, programas de pesquisa, ou iniciativas educacionais.

Questão 5) Explique as diferenças entre o kernel Linux no seu lançamento e a arquitetura do GNU Hurd que estava em desenvolvimento no contexto do GNU Project. Consulte o Capítulo 2 do livro do Silberschatz para aprofundar sua resposta, explicando as diferenças em mais detalhes.

Resp.: Em 1991, Linus Torvalds desenvolveu o kernel Linux com design monolítico. Nesta configuração, o kernel funciona como um bloco substancial de código no espaço do kernel, integrando recursos cruciais como sistemas de arquivos de gerenciamento de memória e drivers de dispositivos. Apesar de fornecer desempenho eficiente, esta estrutura complica a depuração e expansão do kernel. Linux e Solaris são monolíticos, ocupando um único espaço de endereçamento para maior eficiência, mas são também modulares para permitir a adição dinâmica de novas funcionalidades. O Windows é principalmente monolítico por razões de desempenho, mas incorpora características de microkernel, suportando subsistemas separados executados como processos de usuário e módulos do kernel carregáveis dinamicamente.[7]

Desde o início, uma equipe global de desenvolvedores trabalhou de forma colaborativa no aprimoramento do Linux. Linus Torvalds supervisionou seu desenvolvimento online, resultando em rápidas melhorias no kernel. O Linux rapidamente se tornou funcional, preenchendo o vazio deixado pelo Projeto GNU em termos de um kernel funcional.

Por outro lado, o GNU Hurd, desenvolvido pelo GNU Project desde 1990, segue uma arquitetura de microkernel. Nesse modelo, o microkernel (Mach) fornece apenas os serviços básicos, como gerenciamento de memória e comunicação entre processos. Funcionalidades adicionais, como sistemas de arquivos e gerenciamento de dispositivos, são implementadas em servidores no espaço de usuário, em vez de serem incorporadas diretamente no kernel.

A principal função do microkernel é fornecer comunicação entre o programa cliente e os diversos serviços que também estão sendo executados no espaço do usuário. A comunicação é fornecida por **transmissão de mensagens.**

O Tru64 UNIX (antes conhecido como Digital UNIX) fornece uma interface UNIX para o usuário, mas é implementado com um kernel Mach. O kernel Mach mapeia chamadas de sistema UNIX em mensagens enviadas aos serviços de nível de usuário apropriados. O kernel do Mac OS X (também conhecido como Darwin) também se baseia, em parte, no microkernel Mach.

O QNX é um sistema operacional de tempo real para sistemas embutidos. Seu microkernel, QNX Neutrino, fornece serviços de transmissão de mensagens, escalonamento de processos, manipulação de comunicação de rede de baixo nível e interrupções de hardware. Outros serviços do QNX são executados como processos padrão em modo de usuário, fora do kernel.[8]

A abordagem modular do Hurd permite que componentes individuais sejam desenvolvidos e substituídos independentemente, oferecendo maior flexibilidade e segurança. No entanto, o Linux continua sendo a escolha predominante devido à sua eficiência e ampla adoção.

A metodologia atual mais eficaz para o projeto de sistemas operacionais envolve o uso de módulos de kernel carregáveis. Nesse modelo, o kernel mantém um conjunto de componentes principais e adiciona serviços adicionais por meio de módulos, tanto durante a inicialização quanto em tempo de execução. Isso é comum em sistemas modernos como UNIX, Solaris, Linux, Mac OS X e Windows.

No Solaris, a estrutura do kernel inclui sete tipos de módulos carregáveis: classes de scheduling, sistemas de arquivos, chamadas de sistema, formatos executáveis, módulos STREAMS, módulos diversos e drivers de dispositivos e bus.[9]

Questão 6) Aqui não precisa responder. Apenas recomendo que atente-se à importância histórica de ferramentas como o Apache Web Server, e o Netscape para a evolução do Software Livre, e os vetores que levaram a essa convergência. O documentário fala sobre isso.

Bons estudos.

- REFERÊNCIAS COMPLEMENTARES

[1]www.olhardigital.com.br/2024/02/12/internet-e-redes-sociais/dominio-publico-o-que-e-e-como-funciona/

[2] FSF History

[3]www:documentation:read-more [Wiki]

[4]https://diolinux.com.br/video/sistema-de-codigo-aberto-baseado-no-unix.html

[5]History of Linux

[6]https://www.ibm.com/br-pt/topics/open-source

[7] Livro Fundamentos de Sistemas Operacionais (2.7.5 - Sistemas Híbridos)

[8] Livro Fundamentos de Sistemas Operacionais (2.7.3 - MicroKernels)

[9] Livro Fundamentos de Sistemas Operacionais (2.7.4 - Módulos)