Sistema Operacional do Terminal de Acesso de Referência

Celio Hira, Laisa Caroline de Paula Costa, Rogério Pernas Nunes, Lucas C. Villa Real, Marcelo Knörich Zuffo

Laboratório de Sistemas Integráveis
Departamento de Sistemas Eletrônicos
Escola Politécnica da Universidade de São Paulo
Av. Professor Luciano Gualberto, travessa 3, 380, 05508-900
Butantã, São Paulo – SP – Brasil

{chira, laisa, nunes, lucasvr, mkzuffo}@lsi.usp.br

Abstract. *This paper presents the main project aspects and the challenges of the Reference Access Terminal Operating System (SOTAR) conception for the Brazilian Digital TV System. The SOTAR was developed in order to provide manufacturers and embedded software developers with implementing recommendations. Among other requirements, SOTAR was planned to be an open, scalable and flexible system, offering real time applications intelligent use of the access terminal hardware resources.*

Resumo. *O presente trabalho relata os principais aspectos de projeto e os desafios na concepção do SOTAR - Sistema Operacional do Terminal de Acesso de Referência - do Sistema Brasileiro de Televisão Digital. O SOTAR foi desenvolvido para oferecer recomendações a fabricantes de terminais de acesso e desenvolvedores de software embarcado. Dentre outros requisitos, o SOTAR foi projetado para ser um sistema livre, escalável e flexível de forma a oferecer às aplicações de tempo real o uso inteligente dos recursos de hardware do terminal de acesso.*

1. Introdução

Nos últimos anos os sistemas de televisão digital terrestre têm conseguido se estabelecer e se expandir em várias regiões do mundo [Wu, Hirakawa, Reimers e Whitaker 2006]. Dois exemplos de casos de implantação com sucesso são o Japão com o ISDB-T (*Integrated Services Digital Broadcasting Terrestrial*) [DIBEG 1998] e a Itália com o DVB-T (*Digital Video Broadcasting Terrestrial*) [ETSI 2001]. O Japão, com o lançamento do seu sistema de televisão digital terrestre em dezembro de 2003, apresentou vendas na ordem de 18 milhões de terminais de acesso até 2007 (segundo números do JEITA - *Japan Electronics and Information Technology Industries Association*), enquanto que na Itália cerca de 4 milhões de unidades foram vendidas nos últimos dois anos (segundo pesquisa publicada pela GFK Eurisko).

Os dispositivos de recepção dos sinais de televisão digital, aqui denominados como terminais de acesso, passam por uma grande transformação em relação ao sistema analógico, possibilitando o acesso aos usuários finais às novas funcionalidades agregadas pelo sistema de televisão digital, como alta qualidade de som e imagem e a possibilidade de uso de interatividade na TV.

Em 2003 o Brasil deu início ao Sistema Brasileiro de Televisão Digital (SBTVD) através de um projeto para a definição do sistema a ser adotado no país. Este trabalho culminou, em 2007, com as definições das especificações do ISDTV (*Internacional System for Digital TV*), com base nas normas ISDB-T, como sistema de televisão digital a ser implantado no Brasil, com lançamento previsto para dezembro de 2007. Com o objetivo de apoiar as especificações do sistema, o governo brasileiro financiou o desenvolvimento de uma arquitetura de terminal de acesso de referência.

Um sistema completo de televisão pode ser dividido em três componentes: o estúdio, o processo de radiodifusão e o sistema de recepção. O componente estúdio engloba a fase de produção, pós-produção, transmissão de sinais no intra-emissora e armazenamento de conteúdo produzido. Já a componente relativa ao processo de radiodifusão, engloba a transmissão de informações entre emissora e usuários. E, finalmente, o sistema de recepção é composto por uma antena, um televisor e um receptor (podendo estar acoplado ao televisor). O Terminal de Acesso de Referência é o elemento responsável pela recepção e decodificação do sinal digital e pelo oferecimento de interatividade ao usuário. A Figura 1 mostra o componente de rádio difusão e de recepção, indicando a inserção do terminal de acesso na cadeia de um sistema de televisão digital.

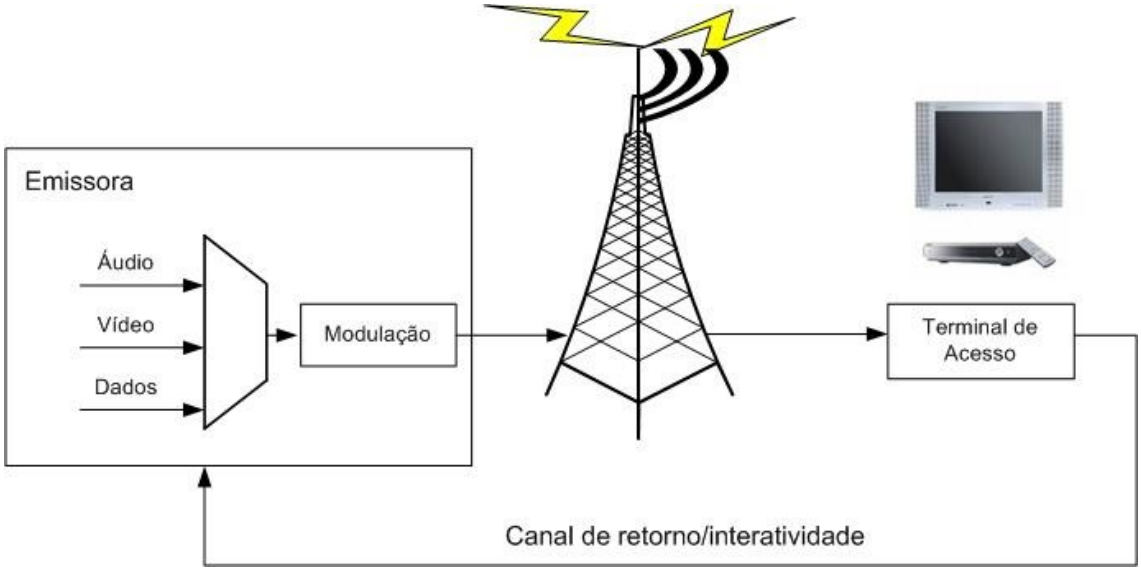

Figura 1. Esquema simplificado de um sistema de televisão digital.

O projeto de recomendação da Arquitetura de Referência do Terminal de Acesso foi coordenado pelo Laboratório de Sistemas Integráveis da Escola Politécnica da Universidade de São Paulo ao longo do ano de 2005. Esta arquitetura foi concebida para contemplar flexibilidade, inclusão digital e a capacidade de incorporação de novas tecnologias, possibilitando implementações que variam em custo, recursos de hardware disponíveis e aplicações.

Tendo em vista tais desafios, a arquitetura de referência inclui a proposta de uma arquitetura de software. A arquitetura proposta insere uma camada de abstração entre a API do *middleware* e o sistema operacional, a IOS (*Interface for the Operating System*), responsável pela uniformização no acesso aos blocos físicos de hardware. Este trabalho apresenta as soluções encontradas para a especificação e desenvolvimento do Sistema Operacional do Terminal de Acesso de Referência (SOTAR).

A primeira parte deste artigo apresenta os requisitos da arquitetura de software do SOTAR, exigidos pelo consórcio, bem como as alternativas tecnológicas empregadas para atender aos mesmos. A arquitetura do SOTAR é apresentada na terceira seção. Por fim, o artigo apresenta suas conclusões e considerações finais.

2. SOTAR

O Sistema Operacional é um gerenciador de recursos. Sua função é gerenciar eficientemente as diferentes partes do sistema, coordenando as tarefas a serem realizadas [Tanembaum 2003]. Em um terminal de acesso, o Sistema Operacional deve controlar os recursos de hardware de um sistema complexo que consiste de processadores, memórias, hardware gráfico, *tuner*, decodificadores de áudio e vídeo e uma ampla variedade de outros dispositivos.

Para controlar tais recursos de hardware, é necessária uma série de características que não são encontradas em um Sistema Operacional convencional. A seguir, apresentaremos os principais desafios na concepção do SOTAR, tendo em vista os requisitos solicitados ao consórcio pelo governo.

Alguns requisitos apontados estão relacionados às ferramentas ou mecanismos computacionais de suporte estrutural, não estando diretamente relacionadas aos recursos essenciais para os terminais de acesso. Já outros requisitos estão voltados para suas funcionalidades primordiais como recepção de sinal digital, decodificação áudio-visual e apresentação do conteúdo, onde alguns ainda seguem as atuais tendências no projeto de sistemas embarcados [Andrews 2005]. As sub-seções a seguir apresentam os requisitos analisados pelo consórcio de universidades envolvidas no projeto.

2.1. Utilização de Soluções em Software Livre

Um requisito considerado no desenvolvimento do SOTAR foi o uso de software livre, sendo aliado à política do governo brasileiro de apoio a ele. Algumas características favoráveis ao uso do software livre são o custo e a independência em relação a um único mantenedor. O impacto do custo do terminal de acesso na digitalização de um sistema de TV é bastante agressivo devido ao número de unidades necessárias (uma para cada TV no sistema analógico). Considerando que o modelo de software proprietário geralmente requer o pagamento de *royalties* por cada unidade vendida, o uso de soluções livres torna-se bastante atrativo economicamente, desonerando os fabricantes de aparelhos e os usuários finais.

Para atender a este requisito, foram analisados vários sistemas operacionais sob licença de uso livre e em desenvolvimento ativo como o FreeBSD, NetBSD, OpenBSD, eCos, Fiasco e Linux.

O sistema operacional Linux foi escolhido pelos requisitos de software livre, escalabilidade e maior disponibilização de ferramentas. Possui inúmeras distribuições distintas, cada qual com suas particularidades, como facilidade de uso ou foco específico em dispositivos embarcados. Dentre as inúmeras alternativas, optou-se pela utilização de uma versão especializada do GoboLinux [Muhammad e Detsch 2002], uma distribuição brasileira. Entre outras características, o GoboLinux apresenta:

- Estruturação própria e diferenciada dos diretórios, voltada para a facilidade de uso, mantendo ainda compatibilidade com as distribuições tradicionais por um mecanismo desenvolvido em *kernel* [Damasio e Villa Real 2003];

- Facilidade de gerenciamento dos perfis de pacotes, agilizando a portabilidade do sistema para plataformas distintas, sem a necessidade de um gerenciador de pacotes baseado em um banco de dados, contribuindo para a atualização de software do terminal de acesso;
- Sistema automatizado para realizar o seu porte para novas arquiteturas de hardware, incluindo sistemas embarcados como o do terminal de televisão digital;
- Ferramentas eficientes para personalização e compilação de programas a partir de seus códigos fonte;
- Simplificação na descrição de *scripts* de *boot* e de tarefas do sistema.

Dadas as características supra-mencionadas, o GoboLinux mostrou-se uma alternativa bastante viável e eficaz para o projeto.

2.2. Tempo de inicialização e carga do sistema operacional

Um outro requisito importante levado em consideração é o tempo de *boot*. A importância deste trabalho está relacionada ao fato de que os usuários esperam que o produto fique disponível para uso imediatamente após ligá-lo. Tradicionalmente, os sistemas operacionais como Linux e Windows utilizados em desktops e servidores demoram de trinta a sessenta segundos aproximadamente para ficarem disponíveis. Este tempo, entretanto, é bastante alto para um sistema de televisão, que geralmente não demora mais do que 4 segundos até mostrar alguma imagem para o usuário.

Para otimizar o tempo de *boot* do terminal de acesso de referência, optou-se pela utilização do Software Suspend 2 [Machek 2004], um recurso que permite armazenar o estado do sistema operacional em uma área de memória volátil, como a memória RAM. Esta técnica possibilita a restauração do sistema em um tempo menor, visto que não há a necessidade da carga de scripts de inicialização, de drivers de dispositivos e de aplicativos residentes. O tempo de restauração varia de acordo com a plataforma de hardware utilizado.

Outra técnica empregada foi a utilização de *caches* para evitar o cálculo de valores estáticos, como a quantidade de instruções que o processador é capaz de executar por segundo (*bogomips*). A eliminação do processo *init* também permitiu suprimir processos concorrentes desnecessários, como os de *login* interativo, sendo substituído pela execução de um aplicativo residente do terminal de acesso, responsável pela recepção e exibição do conteúdo transmitido na televisão.

2.3. Escalabilidade

A escalabilidade é um requisito fundamental dada a variedade de terminais de acesso. Como os terminais de acesso seguem perfis de hardware de diferentes custos, nem todo recurso presente em um terminal de perfil avançado estará disponível em um terminal de perfil básico. Isto exige que as APIs suportem as diferentes configurações de um terminal de acesso.

Foram adotados procedimentos básicos para a verificação da presença de dispositivos de hardware. Caso o hardware não esteja presente, a abstração provida pela API permite que uma camada de sofware responda pela funcionalidade requerida. Esta metodologia

possibilitou a utilização de soluções híbridas, com diferentes funcionalidades nas camadas de hardware e software.

Como prova de conceito de sua escalabilidade, o SOTAR foi implementado em dois terminais de acesso de arquiteturas de hardware distintas:

- Arquitetura convencional: baseado no chipset Intel i815, este terminal de acesso é considerado uma plataforma de propósito geral, que não apresenta recursos especializados em hardware para televisão digital, como módulos para decodificação de vídeo e áudio, implementados em software;
- Arquitetura especializada: este terminal de acesso é considerado uma plataforma de hardware dedicado, ou seja, possui um processador mais simples, visto que grande parte de suas funcionalidades são implementadas em hardware. A implementação de referência utilizada baseou-se no chipset STi5528 da ST Microelectronics.

2.4. Tempo Real

Normalmente, terminais de acesso precisam executar muitas funções de forma suave e com requisitos de tempo. Pode-se considerar um terminal de acesso reproduzindo um filme, no qual os fluxos de áudio e vídeo devem ser decodificados em sincronia. As informações de cada quadro do filme passam por uma série de transformações antes de serem exibidos na tela. O hardware que controla o sinal de televisão precisa receber cada um destes quadros numa taxa fixa de freqüência para apresentar cada imagem na seqüência correta. Se um processo nesta cadeia não puder ser executado devido à sobrecarga do sistema ou a problemas de escalonamento, um quadro pode ser perdido, resultando na aparência de sobressaltos nas imagens apresentadas ao usuário.

A solução para este requisito foi encontrada no sistema de qualidade de serviço provido pela QoSOS [Moreno et al. 2003], que provê métodos de reserva de recursos de processamento e de rede, possibilitando o atendimento deste requisito. Maiores detalhes são apresentados na seção 3.2.9.

2.5. Tamanho

Outras características com relação direta com o custo do terminal de acesso são as limitações típicas de um eletrônico de consumo, como a necessidade de recursos de processamento e memória. Desta maneira, o sistema operacional desenvolvido deve ser minimalista, apresentando apenas ferramentas específicas para as funcionalidades projetadas para o terminal de acesso.

Além de otimizar e remover símbolos desnecessários para a execução e carga de executáveis e bibliotecas dinâmicas, foram utilizadas implementações minimalistas de ferramentas de manutenção do sistema, com base na ferramenta BusyBox [Lin 2006].

2.6. Capacidade de atualização

A possibilidade de atualização de software, sem intervenção do usuário, requer que o sistema operacional disponibilize suas funcionalidades de forma modular, permitindo que partes críticas do sistema sejam atualizadas sem que o sistema seja reiniciado. Este requisito contempla tanto atualizações individuais de pacotes de software, facilitados pela estrutura de diretórios do GoboLinux, quanto da imagem do sistema operacional como um todo.

3. Arquitetura

A arquitetura de software proposta para o SBTVD está apresentada na figura a seguir:

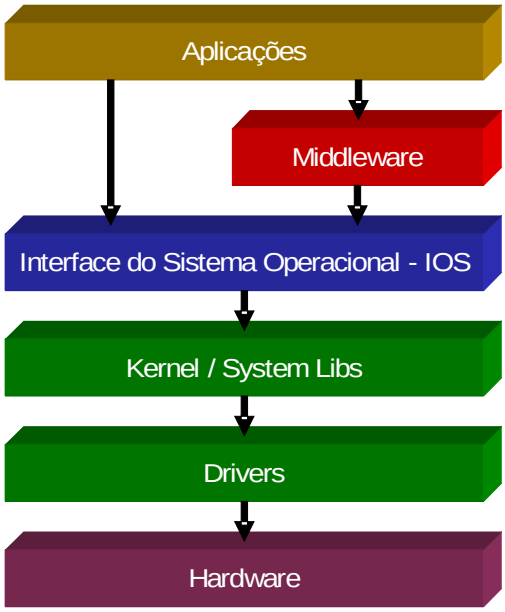

Figura 2. Camadas de software do Terminal de Acesso

Tradicionalmente, as aplicações são desenvolvidas sobre a camada de *Middleware*, que abstrai toda a complexidade e heterogeneidade da plataforma de hardware do Terminal de Acesso.

Na arquitetura do terminal de acesso de referência, é proposta uma interface disponibilizada pelo Sistema Operacional que abstrai a implementação da plataforma para o *Middleware*. Esta interface é denominada de IOS (*Interface for Operating System*).

Conforme a Figura 2, o Sistema Operacional do Terminal de Acesso de Referência pode ser subdividido em três camadas principais: a Interface do Sistema Operacional (IOS), situada entre o *Middleware* e o sistema operacional; O núcleo do sistema operacional (*Kernel*) em conjunto com bibliotecas de baixo nível (*System Libs*); e a camada de drivers de dispositivos (*Device Drivers*), responsável pela comunicação das aplicações com os diversos componentes de hardware disponibilizados pelo terminal de acesso.

3.1. Núcleo

O Núcleo do Sistema Operacional é responsável por gerenciar os recursos limitados de memória, as aplicações de tempo real, dentre outras funções. Uma das características mais importantes é ter suporte a mecanismos multitarefa para permitir a execução e a convivência de vários processos em intervalos de tempo adequados.

3.2. IOS (*Interface for Operating System*)

A IOS é a interface oferecida pelo Sistema Operacional do Terminal de Acesso, com o objetivo de oferecer as funções para a comunicação entre o software em execução e o hardware, assim como as demais abstrações normalmente definidas internamente em um sistema operacional convencional. Funciona como referência para os desenvolvedores das camadas superiores de software.

A IOS permite a interoperabilidade do sistema operacional com o *middleware*. Fazem parte da IOS as chamadas de sistema, funções de bibliotecas básicas, estruturas de dados e variáveis compartilhadas que viabilizam o acesso a tais funcionalidades.

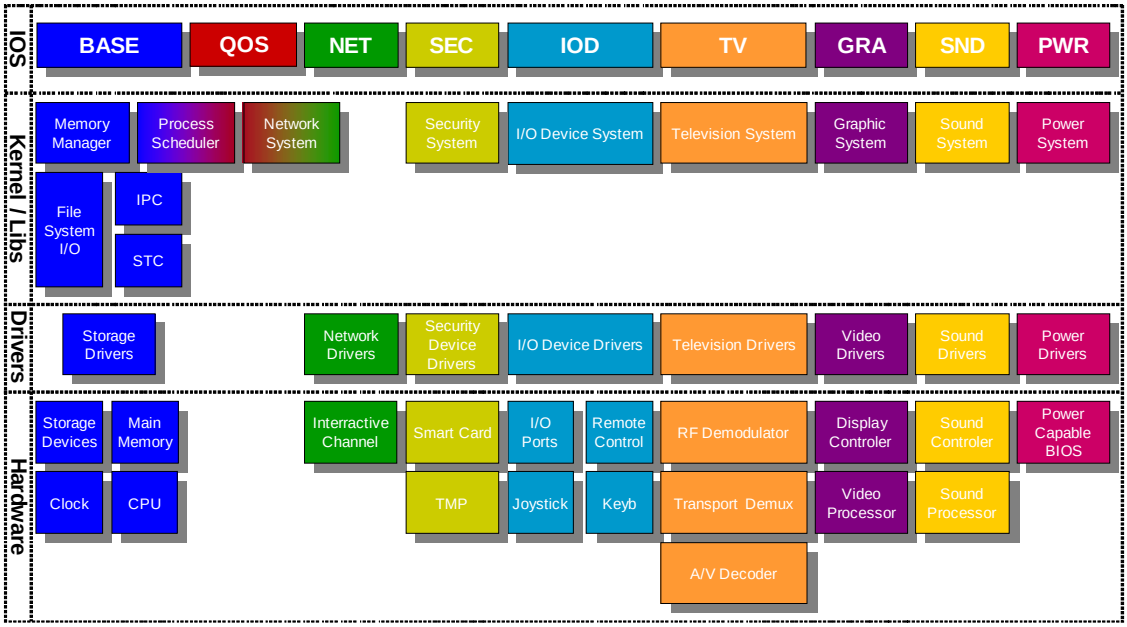

Figura 3. Divisão da IOS

A IOS foi dividida em 9 áreas funcionais, como mostra a Figura 3 e os sub-tópicos a seguir:

3.2.1 IOS-BASE

A IOS-BASE provê um conjunto mínimo de funções para a gerência de processos, acesso a sistemas de arquivos presentes em memória de massa ou memória volátil, gerenciamento de memória principal, comunicação básica entre processos locais e acesso a recursos como o relógio do sistema. Assim, garante acesso ao grupo de Hardware Básico de Computação, composto por:

- Unidade de Processamento Central (CPU);
- Memória principal;
- Memória de massa ou secundária;
- Relógio do sistema.

As chamadas de sistema contidas na IOS-BASE seguem, em sua grande maioria, a especificação POSIX [POSIX 2003] para sistemas operacionais.

3.2.2 IOS-NET

A IOS-NET oferece as chamadas para o estabelecimento de um canal de comunicação, nomeado canal de retorno. Fornece funcionalidades para controle do hardware para o estabelecimento de conexões, tais como discagem (como em *modems* ligados à rede pública de telefonia). Assim, garante acesso ao grupo de Hardware de Comunicação em Rede, composto pelo canal de retorno (ou de interatividade).

As funções da área IOS-NET seguem, em sua maioria, a especificação *BSD Sockets*, conforme documentado pelo padrão POSIX [POSIX 2003].

3.2.3. IOS-SEC

A IOS-SEC oferece uma interface de segurança para autenticação e sigilo na troca de informações. Garante acesso ao grupo de Hardware de Segurança, composto por *SmartCards* e pelo TPM (*Trusted Platform Module*).

Grande parte das funções oferecidas por esta interface segue a API disponibilizada pelo projeto NSS (*Network Securities Services*) [NSS 2007].

3.2.4 IOS-IOD

A IOS-IOD oferece interfaces para gerenciar a comunicação com dispositivos de Entrada e Saída que são operadas pelo usuário. Assim, garante acesso ao grupo de Hardware de Interação com o usuário, composto por controle remoto, teclado, *joystick* e portas de entrada/saída.

As funções disponibilizadas pela IOS-IOD foram desenvolvidas pelo consórcio ao longo do projeto.

3.2.5. IOS-TV

A IOS-TV fornece funções para a seleção de canais de TV, acesso a dados sobre programação e dados multiplexados, entre outros. Assim, garante acesso ao grupo de Hardware Especializado de TV, composto por:

- Front End (sintonizador, demodulador e decodificador de canais);
- Demultiplexador de fluxos de transporte (DEMUX);
- Decodificador de Áudio;
- Decodificador de Vídeo

As funções da IOS-TV utilizam, em alguns casos, funcionalidades disponibilizadas pelos padrões Video4Linux2 [Dirks, Verkuil e Rubli 2006] e LinuxTV e pelos *plug-ins* do Gstreamer.

3.2.6. IOS-GRA

A IOS-GRA é constituída por um conjunto de funções que realizam operações gráficas, possibilitando que as aplicações se comuniquem com o Hardware de Vídeo através de uma API. Assim, garante acesso ao grupo de Hardware de Exibição de Gráficos, composto por um controlador de vídeo

As funções da IOS-GRA seguem a especificação da interface DirectFB [DirectFB 2007].

3.2.7. IOS-SND

A IOS-SND é responsável por funções básicas de acesso e controle de mistura do áudio dos aplicativos com o áudio da programação de TV. Assim, garantem acesso ao grupo de Hardware de Processamento de Áudio, composto pelo controlador de áudio.

As funções oferecidas por esta interface seguem, na sua maioria, as APIs disponibilizadas pelo FusionSound [FusionSound 2007] e pelo ALSA [ALSA 2007].

3.2.8. IOS-PWR

A IOS-PWR disponibiliza para aplicações privilegiadas um conjunto de funções para as ações de controle de energia que vão desde a desativação de dispositivos sem utilização até o desligamento do sistema. Assim, garante acesso ao grupo de Hardware de Controle de Energia, composto pelo sistema básico de E/S (BIOS) compatível com gerenciamento de energia.

Para prover este tipo de serviço, o hardware deve ser compatível com algum mecanismo de gerenciamento de energia.

O consórcio responsável pelo SOTAR desenvolveu as funções disponibilizadas pela IOS-PWR.

3.2.9. IOS-QOS

A IOS-QOS estabelece funções para a negociação de qualidade de serviço envolvendo recursos como CPU e canal de retorno, visando garantir a qualidade de operação de aplicações com fortes requisitos de processamento e/ou comunicação. Dessa forma, tais recursos passam a ser compartilhados com maior justiça e menor probabilidade de ocorrência de sobrecargas, as quais levariam à violação da qualidade de operação das várias aplicações concorrentes. Trata-se de um grupo adicional, sem vínculo específico a um bloco de hardware, mas fundamental para o uso de mecanismos de reserva de recursos como processamento e rede em aplicações de tempo real. Esta IOS segue o padrão conhecido como QoSOS [Moreno et al. 2003].

4. Conclusões

O artigo apresentou uma visão geral do projeto SOTAR, dando ênfase à seleção do Linux como base do SOTAR e da arquitetura proposta, com a inserção da IOS. A escolha do Linux traz a vantagem do uso do software livre, juntamente com uma série de ferramentas já existentes e funcionais, além da ampla gama de desenvolvedores experientes neste sistema atualmente. Já a IOS definida realiza o intermédio de toda a comunicação necessária por parte das aplicações residentes e do *middleware* para o uso dos recursos controlados pelo núcleo do sistema operacional.

Como o terminal de acesso para um sistema de televisão digital sofre influência direta dos requisitos de todos os outros elementos que compõem o sistema, como, por exemplo, modulação, codificação de áudio e vídeo e interatividade, a principal contribuição desse projeto foi concentrar todos esses requisitos e organizá-los em recomendações para os fabricantes e desenvolvedores. Nesse sentido, o sistema operacional é determinante para o atendimento dos requisitos e praticamente essencial para qualquer tipo de solução, pois embora terminais para recepção básica, ou seja, sem nenhum nível de interatividade, possam dispensar um sistema operacional, na prática vemos que todas as soluções recentes (tanto integralmente baseadas em processamento genérico quanto em SoCs) necessitam de um.

Referências

Wu, Y., Hirakawa, S., Reimers, U.H., Whitaker, J. (2006). "Overview of digital television development worldwide". In: Proceedings of the IEEE, Volume 94, Issue 1, p. 8-21.

ETSI (2001). "Digital Video Broadcasting; Framing structure, channel coding and modulation for Digital Terrestrial Television", ETSI EN 300 744, v.1.4.1.

DIBEG (1998). "Terrestrial intelligent services digital broadcasting (ISDB-T) – Specification of channel coding, framing structure and modulation", http://www.dibeg.org/techp/Documents/Isdb-t_spec.PDF.

Tanembaum, A. (2003). "Sistemas Operacionais Modernos", Prentice-Hall, 2ª. edição.

Andrews, D. et al. (2005). "Impact of Embedded Systems Evolution on RTOS Use and Design", In: Workshop on Operating System Platforms for Embedded Real-Time Applications, Palma de Mallorca.

Damasio, Felipe W, Villa Real, Lucas Correia (2003). GoboHide: Uma Solução Flexível e Escalável para Inodes Ocultos no Kernel Linux, IV Workshop sobre Software Livre - WSL2003, Porto Alegre/RS.

Dirks, B., Verkuil, H., Rubli, M. (2006) "Video for Linux Two API Specification", Draft 0.21.

Moreno, M, Soares Neto, C, Gomes, A.T., Colcher, S, Soares, L.F. (2003) "QoSOS: An adaptable architecture for QoS provisioning in network operating systems", Journal of the Brazilian Telecommunications Society, Special Issue, v.18, n.2, p.118-131.

POSIX (2003). ISO/IEC 9945-1:2003 Information technology - Portable Operating System Interface (POSIX) - Part 1: Base Definitions.

NSS (2007). "Network Securities Services", http://www.mozilla.org/projects/security/pki/nss/, Maio.

Muhammad, H, Detsch, A. (2002). "Uma nova proposta para a árvore de diretórios UNIX". In: Proceedings of the III WSL – Workshop em Software Livre, Porto Alegre.

Machek, P. et al. (2004). "Power and thermal management". In: OpenWeekend Conference.

Lin, Y. (2006). "Turn key – Image building", Wireless/Wireline Convergence Networks Lab.

FusionSound (2007). "FusionSound Reference Manual", http://www.directfb.org/docs/FusionSound_Reference/index.html, Maio.

ALSA (2007). "Advanced Linux Sound Architecture", http://www.alsa-project.org/, Maio.

DirectFB (2007). "DirectFB Overview (v0.2 for DirectFB 0.9.21)", www.directfb.org/docs/DirectFB_overview_V0.2.pdf, Maio.