Pedal Pi - Multi-processador de plugins de áudio DIY

Paulo Mateus M. da Silva, Leonardo da S. Costa, Sandro C. S. Jucá

Campus Maracanaú – Instituto Federal de Educação, Ciência e Tecnologia do Estado do Ceará (IFCE)
CEP: 61.939-140 – Maracanaú – CE – Brasil

mateus.moura@hotmail.com, {leonardoscifce, sandro.juca}@gmail.com

Abstract. *The lack in the music market of audio plugins multi-processor solutions that are customizable and accessible affect instrumentalists when they need to add or adapt resources, forcing them to attempt to replace or incorporate new equipment. The project Pedal Pi is an open-source software solution and embedded hardware that aims to fulfill the needs of the above points for instrumentalists beginners. It is presented in this article the project, its software architecture, a cost analysis, the development schedule of work and the results.*

Resumo. *A carência no mercado musical de soluções de multi-processadores de plugins de áudio acessíveis e customizáveis afeta instrumentistas quando necessitam agregar ou adaptar recursos, forçando-os a realizar a substituição ou incorporar novos equipamentos. O projeto Pedal Pi é uma solução de software open-source e hardware embarcado que visa atender os pontos supracitados para instrumentistas iniciantes. É apresentado neste artigo o projeto, sua arquitetura de software, uma análise de custo, o desenvolvimento do trabalho e os resultados obtidos.*

1. Introdução

Instrumentistas musicais iniciantes buscam geralmente equipamentos versáteis para iniciar os estudos. Sem poder investir muito, produtos que oferecem uma gama de possibilidades sonoras por um custo baixo costumam ser mais atrativos. Nesse sentido, o mercado mundial abastece essa demanda ao oferecer soluções básicas, de baixo custo e com qualidade.

Entretanto, o modelo de mercado adotado pelas empresas de processamento de áudio é direcionado à troca de equipamentos por outros mais recentes ou mais potentes. Para os dispositivos de entrada, atualizações e incrementos de plugins de áudio raramente são providos e a interface homem-máquina (IHM) não extensível limita a performance musical do usuário.

Iniciativas *open-source*, por meio do compartilhamento de experiências e de código-fonte, incentivam a otimização e o reaproveitamento, como também propõem alternativas a produtos comerciais [Paulson *et al.* 2004], [Hars e Ou 2002]. Nesse sentido, o AudioPint [Merril *et al.* 2007] oferece uma plataforma robusta para manipulação de efeitos e sintetização com Pure Data; o Zynthian provê uma plataforma

aberta para sintetizadores [Zynthian 2016]; e o MOD [Ceccolini 2013] concede a utilização e o controle de plugins de áudio aos pés de instrumentistas.

De forma similar, o projeto proposto Pedal Pi é uma plataforma acessível que visa proporcionar uma solução DIY (*Do it yourself*) – ou seja, faça você mesmo – para uso de plugins de áudio com qualidade já consolidada. Essa ferramenta, composta por hardware acessível e softwares *open-source*, busca dispor um sistema de processamento multi-efeitos integrável com interfaces de áudio de baixo custo suportáveis pelo GNU/Linux. É projetado para também permitir a configuração e persistência de bancos, de *pedalboards* e dos plugins utilizados através de sua extensão, seja por uma IHM que atenda as necessidades do usuário, como por programas que consumam um serviço *web* a ser oferecido futuramente pela plataforma.

O desenvolvimento deste trabalho científico foi composto pelas seguintes etapas: (i) realização de uma pesquisa de mercado para análise de custo e obtenção dos equipamentos necessários, enfocando a viabilidade econômica; (ii) integração entre os recursos de hardware, softwares já concebidos pela comunidade *open-source*; (iii) implementação de recursos de controle e extensão; e, (iv) testes para comprovação da funcionalidade e da usabilidade.

Como idealização do projeto, almeja-se que o mesmo seja replicado e otimizado pela comunidade *open-source* e pelos próprios usuários. Desta forma, com a difusão dos recursos do Pedal Pi, pode ocorrer a consolidação e a participação do projeto no mercado musical.

2. Análise de mercado

Com o intuito de oferecer um produto DIY economicamente acessível, uma análise de mercado foi realizada para estabelecer um limite máximo para o custo do projeto.

2.1. Análise do mercado de equipamentos musicais

Os equipamentos escolhidos para a comparação de preços pertencem à categoria de entrada, ou seja, são considerados equipamentos com custo baixo, visando atingir a parcela de instrumentistas que estão iniciando no meio musical. Entre eles, foram escolhidos dois processadores multi-efeitos – Zoom G1On e Line 6 Pocket Pod – e um pedal de guitarra – Boss DS 1 – como forma de representarem a gama de equipamentos disponíveis presentes aos músicos.

Na busca realizada em sites *e-commerce* americanos, o Zoom G1On e o Boss DS 1 apresentaram um valor médio de USD 50.00. Para o Line 6 Pocket POD, o valor médio era de USD 130.00. Como resultado dessa pesquisa, foi estipulado um custo máximo de USD 70.00 para o Pedal Pi. O valor deve ser dividido entre um Sistema Embarcado que execute alguma distribuição do Sistema Operacional (SO) GNU/Linux, interface de áudio, fios, botões, displays e alimentação de energia.

2.2. Seleção dos equipamentos utilizados

Em busca de um equipamento de pequeno porte capaz de embarcar um Sistema Operacional GNU/Linux e processar plugins de áudio em tempo real, foi escolhido o Raspberry Pi 3 (RPi 3). Seguindo o lema de fomentar a educação e o desenvolvimento

tecnológico [Raspberry Foundation], o esforço da Raspberry Foundation gerou, além de seus produtos, resultados em diversas áreas. Especificamente, para Processamento Digital de Sinais (DSP), projetos como o de Llopis (2015), o de Terán (2015) e Zynthian nortearam a escolha desse equipamento.

Carecendo de uma interface de áudio contendo uma entrada de áudio, foi necessário também buscar uma interface que ofereça uma boa qualidade de áudio. *HiFiBerry DAC*, o equipamento possivelmente mais conhecido, naturalmente seria a primeira escolha, mas, pelas limitações de custo e da ausência de entrada de áudio, preferiu-se utilizar uma interface de áudio genérica. Assim, foi escolhido o *USB - Guitar Link*, um equipamento genérico facilmente encontrado em sites *e-commerce*.

Com base nos materiais escolhidos, uma lista foi formulada (Tabela 1). O valor estimado está baseado nos preços obtidos em sites *e-commerce* internacionais.

Tabela 1. Lista estimada de materiais. Estimativa de preço em USD.

Quantidade - Item	Preço	Quantidade - Item	Preço
1 Raspberry Pi 3	35.00	2 Displays de sete segmentos	0.70
2 Footswitches DPDT ou 3PDT	12.00	1 Interface de áudio com entrada e saída p10 genérica	5.00 ~ 10.00
1 Cartão USB 8 Gb	3.00	1 Fonte para alimentação	5.00
	Total	**65.70**	

3. Arquitetura

O sistema foi desenvolvido em módulos que comunicam entre si. Sua arquitetura permite que os *nodes* do sistema sejam distribuídos em máquinas distintas (Figura 1). A seguir, são apresentados os módulos presentes nos *nodes*.

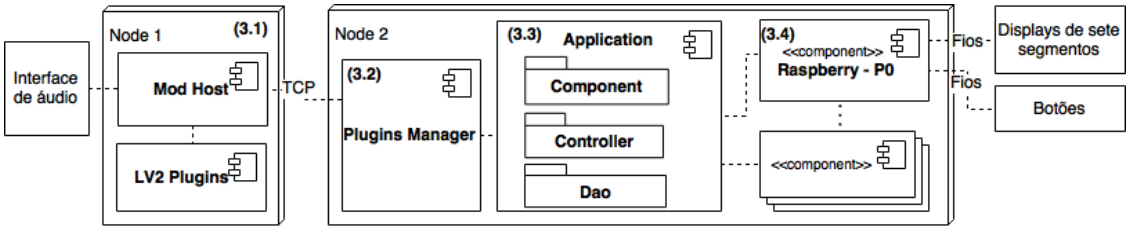

Figura 1. Visão geral da arquitetura

3.1. Plugins de áudio *Lv2* e *mod-host*

A tecnologia escolhida para o fornecimento de plugins de áudio foi a LV2 (acrônimo de *Linux Audio Developer's Simple Plugin API* - LADSPA - version 2). Possuindo uma gama de plugins de áudio já consolidados – como Calf Plugins[1] e Guitarix[2] – o projeto Pedal Pi poderá dar um suporte para instrumentistas iniciantes e mais experientes.

O *mod-host*[3] – *host* LV2 para o servidor de som JACK – foi utilizado para o controle e gerência dos plugins de áudio e do servidor de som JACK. A possibilidade de

[1] http://calf-studio-gear.org
[2] http://guitarix.org/
[3] https://github.com/moddevices/mod-host

controlá-lo por *sockets* TCP permite uma maior modularidade e abstração, facilitando o desenvolvimento e a manutenção.

3.2. Plugins Manager

A biblioteca Plugins Manager desenvolvida neste projeto abstrai a comunicação com o *mod-host*. Com o auxílio da biblioteca *liblilv*[4], disponibiliza uma forma de controle e organização lógica de plugins de áudio através de uma API escrita em *python 3*.

Em sua abstração, instâncias dos plugins de áudio com seus parâmetros configurados e suas conexões estabelecidas ficam armazenadas em *pedalboards* – conhecidos também como *patch*, *preset* ou *set* de pedais. *Pedalboards* ficam agrupados em bancos – uma relação muitos para um.

3.3. Application

Application é a parte orquestradora do projeto. Com o papel de *aplicação*, é responsável por carregar seus módulos, especificar e gerenciar a comunicação entre os componentes terceiros (pacote *component*), fornecer uma API para acesso ao que é ofertado por Plugins Manager (pacote *controller*) e persistir e carregar o seu estado (pacote *dao*).

Sendo concebida para permitir a extensão do Pedal Pi, possibilita a adição de novos componentes para controle do equipamento. Inspirado no padrão de projeto *Observer* [Gamma *et al.* 1995], os componentes registrados em Application são notificados quando uma mudança for realizada.

Por exemplo, a alteração do *pedalboard* em uso por um aplicativo controlador resultaria na atualização da numeração exibida em displays de sete segmentos. Nesse exemplo, o aplicativo consumiria um serviço *web* provido por um componente e os displays atualizados seriam gerenciados pelo componente Raspberry - P0.

3.4. Componente Raspberry - P0

Raspberry - P0 dispõe um gerenciamento simplificado de parte dos recursos oferecidos por *Application* ao usuário do Pedal Pi através de uma Interface Homem Máquina composta por displays de sete segmentos e botões (Figura 2): Por meio dos displays, o usuário poderá visualizar qual o número do *pedalboard* que está sendo utilizado; a partir dos botões, poderá alterar o *pedalboard* atual para o próximo ou para o anterior.

O enfoque da IHM desenvolvida foi a simplicidade de montagem. A biblioteca *gpiozero* foi utilizada por proporcionar um controle amigável de pinos físicos e componentes conectados [Nuttall 2015].

4. Resultados obtidos

Esta seção descreve os resultados obtidos nesta pesquisa até o momento.

4.1. Modularidade

A fim de testar a modularidade da arquitetura, o sistema foi distribuído em um conjunto de três computadores e um tablet conectados em uma mesma rede:

[4] https://github.com/moddevices/lilvlib

- Computador **X**: *Mod-host* na Live-MOD[5] – um sistema operacional fornecido pela *MOD Devices* contendo *plugins* de áudio a partir de uma *live ISO*. Uma guitarra estava conectado ao computador através do USB - Guitar Link;
- Computador **Y**: Application e componente de dispunha um serviço *web*;
- Computador **Z**: Sistema web que consumia o serviço *web*.

No ambiente preparado, o tablet, por intermédio do sistema web em **Z**, recebeu informações de *pedalboards* e suas configurações do **Y** e realizou ações de mudança do *pedalboard* em uso e dos valores de parâmetros de instâncias de *plugins* do *pedalboard* em uso através de comandos enviados do **Y** para o **X**.

No teste, o conjunto se comportou de forma parcialmente desejada, apresentando raras vezes problemas no processamento dos plugins de áudio. Acredita-se que se deva ao uso de um sistema operacional sendo executado em uma *live ISO*, ao invés do uso do sistema instalado no computador.

4.2. Construção física do projeto

Utilizando o *Raspbian*, SO gratuito distribuído pela Raspberry Foundation, foram instalados os softwares desenvolvidos. A montagem dos equipamentos pode ser vista na Figura 3. São da esquerda para direita, (a) displays de sete segmentos, (b) Raspberry Pi 3, (c) botões (substituindo os *footswitches*), e (d) interface de audio USB.

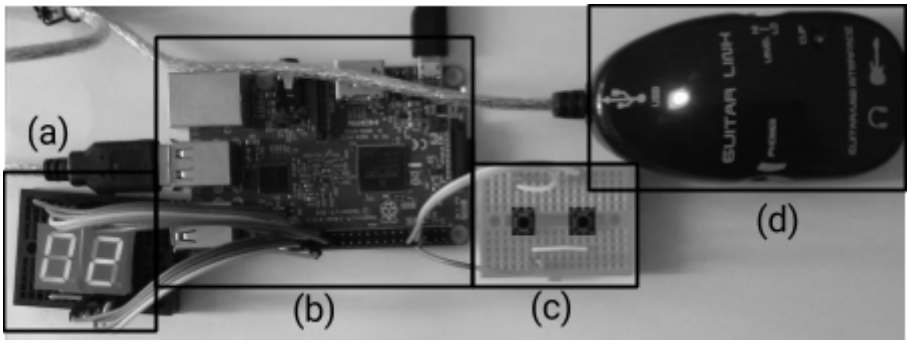

Figura 3. Equipamentos do projeto conectados.

Nos testes realizados, o processamento de plugins de áudio demonstrou ser instável para configurações no JACK em que a latência ficava menor que 0.20 *ms*. Para latências maiores, apesar de se comportar mais estável, o atraso era suficientemente alto para atrapalhar execuções musicais.

Uma alternativa encontrada para contornar o problema foi delegar o processamento dos plugins de áudio para um computador pessoal. Com essa configuração foi possível aproveitar a estabilidade do processamento do áudio com a extensibilidade propiciada pelo sistema embarcado utilizado.

5. Conclusões e trabalhos futuros

Como visto na análise de mercado, o protótipo proposto apresentou um custo equivalente a equipamentos disponíveis comercialmente. Contudo, possui vantagens de ser aberto e customizável às características do usuário, viabilizando a adição de novos

[5] http://wiki.moddevices.com/wiki/Live-MOD

plugins de áudio, a utilização em hardwares para os quais o GNU/Linux oferece suporte e o desenvolvimento de IHMs que correspondam às necessidades dos instrumentistas.

Apesar de ocorrerem problemas no processamento de áudio no sistema embarcado utilizado, a combinação com um computador pessoal apresentou um nível de estabilidade desejado. Nesse sentido, um suporte ao projeto juntamente com um breve desenvolvimento tecnológico tende a permitir que o projeto torne-se mais robusto, atendendo assim tanto instrumentistas iniciantes quanto mais experientes.

Como sugestões para trabalhos futuros, são propostos o desenvolvimento de aplicativos para a configuração em sistemas *mobiles*, o desenvolvimento de IHM adaptadas para pessoas com deficiência motora e um estudo de distribuição do processamento de plugins de áudio em equipamentos conectados em rede.

Referências

Ceccolini, G. e Germani, L. (2013). "MOD – An LV2 host and processor at your feet". Institute of Electronic Music and Acoustics, University for Music and Performing Arts Graz, Austria May 2013. Editores: IOhannes m zmölnig e Peter Plessas All ISBN 978-3-902949-00-4. p. 159-161.

Gamma, E., Helm R., Johnson, R. e Vlissides, J. (1994). "Design Patterns: Elements of Reusable Object-oriented Software". Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA ISBN:0-201-63361-2

Hars, A., Ou, S. (2002). "Working for free? Motivations for participating in Open-Source projects". International Journal of Electronic Commerce 6, 25–39.

Llopis, H. S. (2015). "Implementación de efectos de audio en la tarjeta Raspberry Pi B+", Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación, España.

Merril, D., Vigoda, B. e Bouchard, D. (2007). "Audiopint: A Robust Open-Source Hardware Platform for Musical Invention". Em: Pd Convention 2007. Montréal, Québec, Canada.

Nuttall, B. (2015). "GPIO Zero: A Friendly python api for physical computing", https://goo.gl/Y3UvSJ. Acessado em Setembro de 2016.

Paulson, J. W., Succi, G. e Eberlein, A. (2004). "An empirical study of open-source and closed-source software products". IEEE Transactions on Software Engineering, 30, pp. 246-256, 2004.

Pedal Pi (2017). "Pedal Pi - DIY audio plugins multi-processor", http://github.com/PedalPi. Acessado em Janeiro de 2017.

Raspberry Foundation. "About us", https://www.raspberrypi.org/about/. Acessado em Outubro de 2016.

Terán, M. e Antonio, M. (2015). "Diseño y Construcción de una pedalera de guitarra digital basada en RaspberryPi", Universidad San Francisco de Quito, Ecuador.

Zynthian. (2016). "Zynthian: The Open Synth Platform". http://zynthian.org/. Acessado em Agosto de 2016.