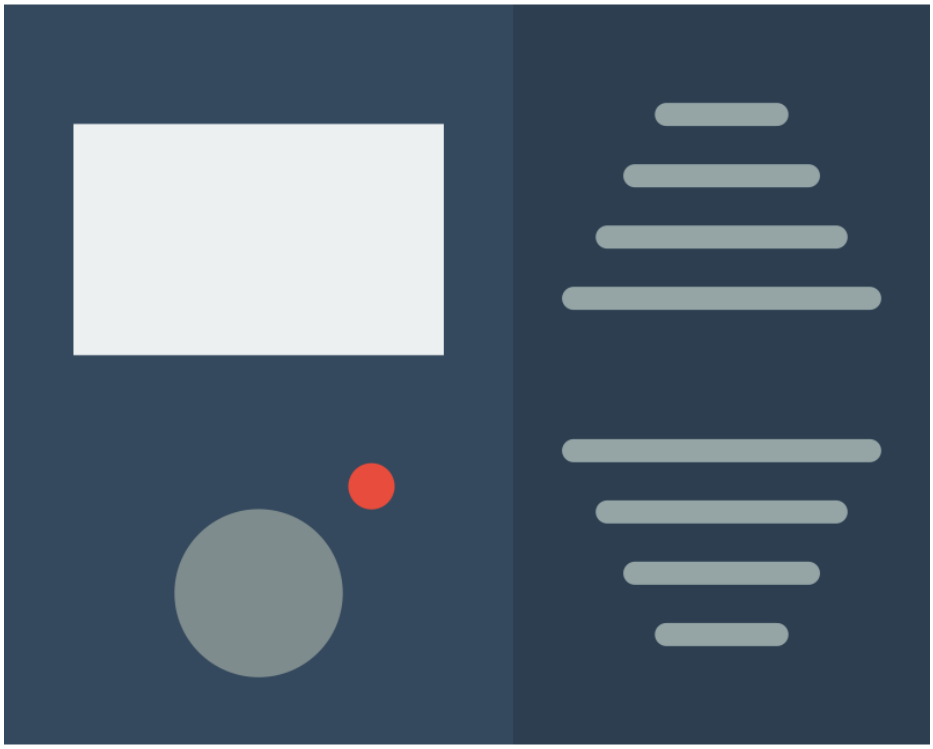

Pedal Pi - Especificação Suplementar

por Paulo Mateus Moura da Silva e Leonardo da Silva Costa
Pedal Controller Projects

Versão 1.3
02/05/2017

1. Restrições

1. O produto deve oferecer um design e configurações simplificadas, focando no uso;
2. Utilizando-se de plugins de áudio *open-source* consolidados e outras ferramentas de código livre, possui restrições em sua manutenção, pois é realizada de forma indireta e colaborativa;
3. Por restrições de tempo e mão-de-obra:
    a. Devem ser desenvolvidas as funcionalidades primordiais. Funções complexas que não primordiais devem ser deixadas de lado;
    b. O produto código-fonte deve ter um cuidado especial, com enfoque em manutenabilidade, de forma que o trabalho possa ser continuado pela comunidade open-source. São necessários assim uma boa documentação e testes. (Veja 2. Intervalos de qualidade)

2. Intervalos de qualidade

2.1 Desempenho

- O processamento dos plugins de áudio deve gerar baixa latência: A configuração da interface de áudio não deve causar uma latência superior a 10 ms;
- A aplicação - que permite o controle do hardware e execução dos plugins de efeito - deve consumir no máximo 10% do poder de processamento e, juntamente com o sistema operacional, ¼ da memória principal em sistemas computacionais com no mínimo 1 gigabyte;
- O tempo de alterações de configurações de efeitos aceitável é entre 20 ms e 100 ms;
- O período de inicialização do sistema não deverá ultrapassar um minuto.

2.2 Robustez

- O código gerado deve possuir uma documentação boa o suficiente e bons códigos de exemplo de modo que:
    - O entendimento da arquitetura para o desenvolvimento de novas IHM por desenvolvedores *Seniors* seja inferior a 2 horas;
    - A montagem de alguma das IHM simples desenvolvidas no projeto seja inferior a 4 horas por makers iniciantes;
- Deverá ser possível a utilização de um computador de uso pessoal para a execução dos plugins de áudio, caso o sistema embarcado utilizado não consiga atender os requisitos não funcionais do projeto.

2.3 Tolerância a falhas

- A atualização da aplicação não deve resultar em perda de configurações previamente salvas.
- Outras medidas de tolerância a falhas não serão inicialmente tomadas. Futuros upgrades poderão trazer:
    - Criação de cópias de segurança, de modo automática ou manual;
    - Prevenção ao corrompimento de informações em casos como interrupção de energia.
    - Interrupção por parte do sistema do funcionamento de plugins com consumo de processamento instável (com taxa de variação de 50%) ou gulosos (com taxa de uso superior aos 50% do ofertado pelo hardware).
- Interrupções por completo do funcionamento do sistema são totalmente indesejáveis, de modo em que o limite deve ser de uma vez por hora nas versões iniciais.

2.4 Usabilidade

- Para usuários experientes, caso estes queiram utilizar o equipamento embarcado como um computador de propósito geral, a case que envolve o hardware não deve bloquear entradas HDMI e USB;

- A interface de edição de *pedalboards* deve ser usável de modo em que usuários pouco familiarizados com o sistema, mas acostumados com a abstração de pedais de efeitos e com a interface de dispositivos móveis deverão em menos de 20 minutos compreender a organização dos sistema.

2.5 Segurança

- Deve restringir o acesso ao Sistema Operacional e à aplicação através da troca das senhas padrão e da adição de uma camada de autenticação nos programas ou aplicativos gerenciadores. Além, deve ser possível bloquear o acesso à rede completamente.

3. Outros Requisitos

3.1 Padrões Aplicáveis

- Para a utilização da interface de rede para o gerenciamento remoto dos plugins de áudio, deverão ser utilizados corretamente as especificações e tecnologias TCP/IP (v4 e v6), REST, RFC-6455 WebSocket e Socket;
- O sistema deverá gerenciar plugins de áudio que seguem o protocolo LV².

3.2 Requisitos do Sistema

- Sistema embarcado com:
    - suporte a distribuições do Sistema Operacional Linux;
    - plugins de áudio LV² instalados;
    - interface de rede Ethernet. WiFi e Bluetooth são desejáveis, mas sem enfoque no presente momento;
    - bom poder de processamento: 32 ou 64 bits com mono-core ou multi-core com frequência superior a 800,0 MHz;
    - memória RAM com no mínimo um Gigabyte.
- Computador pessoal para processamento de plugins de áudio com:
    - suporte a distribuições do Sistema Operacional Linux;
    - plugins de áudio LV² instalados;
    - interface de rede Ethernet;
    - bom poder de processamento: 32 ou 64 bits com mono-core ou multi-core com frequência superior a 2,0 GHz;
    - memória RAM com no mínimo dois Gigabytes.
- Dispositivo para gerenciamento e configuração de forma remota:
    - suporte à internet;
    - navegador web moderno: versão com data de lançamento de até seis meses anterior à data deste documento.

3.3 Requisitos de Desempenho

Verifique a subseção **2.1 Desempenho**;

3.4 Requisitos ambientais

- Deve ser possível o usuário controlar o volume resultante do processamento dos plugins de áudio para que este possa atender às restrições ambientais relacionadas ao silêncio e o ruído.

4. Tabela de fatores arquitetural

Fator	Cenário de qualidade	Variabilidade	Impacto	Prioridade	Dificuldade
Segurança - Restrição de acesso					
Evitar acesso indevido do SO ou da aplicação	A adição das camadas de segurança não devem impactar no resto do sistema.	**Flexibilidade atual** Não exigida no momento **Evolução** Nenhuma	Impacto baixo/médio no web-service e na aplicação controladora (camada de segurança)	A	M
Adaptabilidade - Suporte à outras tecnologias de plugins de áudio					
Possibilitar implementar o suporte de outras tecnologias de plugins de áudio	A implementação deve o mínimo de efeitos colaterais, de forma que menos de 10% dos testes implementados devem quebrar	**Flexibilidade atual** conforme descrita no Fator **Evolução** Nenhuma	Baixo impacto após implementar variação protegida	M	B

5. Memorandos Técnicos

5.1 Segurança - Restrição de acesso (Não implementado)

- **Fatores**:
  Evitar acesso indevido do Sistema Operacional ou da aplicação;
- **Solução**: Adicionar uma chave liga-desliga para habilitar e desabilitar o wireless e um led para indicar a transmissão de dados.
  Deixar o sistema preparado para adicionar funcionalidades para autenticação no web-service (variação protegida através de adição de um middleware de autenticação);

Nintendo 3Ds - Controle para habilitar Wireless

- **Motivação ou raciocínio**: É necessário uma forma de restringir o acesso ao gerenciamento do Pedal Pi. Alterações indevidas podem comprometer performances ao vivo. Ataques de segurança podem ocorrer através de acesso *ssh* indevido ou pelo uso da API web-service.

  A solução pensada se concentra em dois pontos:
  - Habilitação/desabilitação da rede por uma chave seletora;
  - Indicação de tráfego da rede através de um led.

  No futuro, operações para alteração da senha do *ssh* e a adição de uma camada de autenticação no web-service deverão ser implementadas;

- **Alternativas consideradas e fundamentos de rejeição dessas alternativas**:
  Foi estimado que o desenvolvimento das funcionalidades para alterar a senha do ssh e da adição de uma camada de autenticação no web-service gastariam um tempo de desenvolvimento grande e não necessariamente resolveria o problema: usuários poderiam colocar senhas fracas.

  O ideal é a combinação dos dois métodos, mas como a solução escolhida evita ataques ao bloquear o acesso, foi escolhida implementar esta primeiro.

5.2 Adaptabilidade - Suporte à outras tecnologias de plugins de áudio

- **Fatores**:
  Possibilitar implementar o suporte de outras tecnologias de plugins de áudio;
- **Solução**: Obter variação protegida através de polimorfismo e baixo acoplamento (*Observer*):
  - Permitir a especificação de outros plugins de áudio na cada de modelo através do polimorfismo;
  - Permitir a substituição ou complemento do host de áudio por outro* através de um baixo acoplamento entre a classe *façade* que se comunica com o host e o sistema. O baixo acoplamento se dá através do padrão de projeto *Observer*: Alterações realizadas em objetos do modelo são repercutidas no host através das notificações (*subject*: objetos do modelo. *observer*: *façade*).

    * Exemplo: Substituir *mod-host* por *carla*.
- **Motivação ou raciocínio**: Sabendo que pode haver uma demanda futura para adicionar o suporte de outros plugins de áudio, era necessário que o sistema possuísse meios não tão custosos de atualizar, mas que as escolhas tomadas não custassem muito tempo de desenvolvimento.

  Para adicionar suporte a outros plugins de áudio, além de implementar devidamente novas classes da camada de domínio (como VST2Effect, AUEffect) é necessário também trocar o host de áudio. Sendo cada host implementado independentemente e não havendo um padrão conhecido, a comunicação com o host deveria estar completamente isolado do resto do sistema.

  A escolha tomada preparou o sistema para atualizações sem haver uma superengenharia.
- **Alternativas consideradas e fundamentos de rejeição dessas alternativas**:
  No esboço de implementação inicial havia um acoplamento forte entre *façade* do host de áudio e o sistema. Foi percebido que isso não auxiliaria na manutenção no futuro.