Replica: Automação da Reprodução de Ambientes Flatpak por meio da Geração de Artefatos IaC

Adryan N. Reis[1], **Yulle G. S. de Lima**[2]

[1]Instituto Federal de Pernambuco (IFPE) - Campus Igarassu
53659-899 – Igarassu – PE – Brasil

[2]Instituto Federal de Pernambuco (IFPE) - Campus EAD

anr@discente.ifpe.edu.br, yulle.gustavo@ead.ifpe.edu.br

Abstract. *This work presents Replica, a graphical application developed in Python with GTK 4 and libadwaita that automates the generation of IaC artefacts from the current state of a Flatpak system. The tool reads installed applications, their source repositories, and their permissions (overrides), producing artefacts capable of reproducing the environment on a new installation, supporting Ansible Playbook and Shell Script formats. Experiments carried out in a virtualized environment demonstrate that the generated artefacts are able to faithfully restore the set of applications and their permission settings in both homogeneous and heterogeneous Linux distribution scenarios, with the exception of uncatalogued custom remote repositories.*

Resumo. *Este trabalho apresenta o Replica, uma aplicação gráfica desenvolvida em Python com GTK 4 e libadwaita que automatiza a geração de artefatos IaC a partir do estado atual de um sistema Flatpak. A ferramenta realiza a leitura dos aplicativos instalados, seus repositórios de origem e suas permissões (overrides), produzindo artefatos capazes de reproduzir o ambiente em uma nova instalação, suportando os formatos Ansible Playbook e Shell Script. Experimentos realizados em ambiente virtualizado demonstram que os artefatos gerados são capazes de restaurar fielmente o conjunto de aplicações e suas configurações de permissão em cenários homogêneos e heterogêneos de distribuição Linux, com exceção de repositórios remotos customizados não catalogados.*

1. Introdução

A manutenção e reprodução de ambientes de software é um desafio recorrente no cotidiano de desenvolvedores e usuários de sistemas Linux. Eventos como reinstalações do sistema operacional, migrações entre máquinas ou configurações de novos ambientes de trabalho em um ambiente sem uma abordagem declarativa para manutenção destes Flatpaks exigem a reinstalação manual de aplicações e a reconfiguração de suas permissões, sendo um trabalho tedioso e propenso a não replicar corretamente o estado anterior. A relevância desse cenário é evidenciada pelo crescimento do Flatpak, em 2025, o Flathub, repositório primário de aplicações Flatpak [Flatpak 2026], registrou 438,2 milhões de downloads, com um crescimento de 21,6% em relação ao ano anterior, sendo o Brasil o terceiro país com o maior número de downloads, totalizando 14,6 milhões [Flathub 2025].

O Flatpak [Flatpak Team 2026] é um sistema de distribuição de aplicações para Linux que oferece isolamento por meio de *sandboxing*. Cada aplicação é executada em um ambiente controlado, com acesso restrito a recursos do sistema, como sistema de arquivos, dispositivos e variáveis de ambiente. Esse controle de acesso é configurado por meio de *overrides* de permissões, que podem ser personalizados pelo usuário para cada aplicação instalada. Embora o Flatpak facilite a distribuição e o isolamento de aplicações, a restauração dessas configurações após uma reinstalação do sistema não é trivial.

Ferramentas de automação de infraestrutura como o Ansible [Red Hat 2026a] permitem descrever o estado desejado de um sistema em arquivos YAML declarativos, denominados *playbooks*, que podem ser executados para convergir qualquer máquina ao estado especificado [Hochstein and Moser 2017]. Esse paradigma, conhecido como Infraestrutura como Código (IaC) [Kumara et al. 2021], estabelece a prática de versionar e automatizar a configuração de ambientes de software. No entanto, a construção manual de *playbooks* Ansible para ambientes Flatpak existentes exige conhecimento simultâneo da CLI do Flatpak e da sintaxe do Ansible, o que pode ser uma barreira significativa para alguns usuários.

Este trabalho apresenta o Replica, uma aplicação gráfica desenvolvida em Python com GTK 4 que automatiza a criação de artefatos IaC a partir da inspeção do estado atual de um sistema Flatpak. O desenvolvimento seguiu a abordagem de *Design Science Research* (DSR) [Hevner et al. 2004], na qual um artefato de software é projetado para resolver um problema prático identificado e avaliado por meio de experimentos controlados. Diferentemente das soluções existentes, o Replica captura as permissões de forma automática e opera sobre qualquer distribuição Linux com Flatpak instalado, sem exigir a adoção de um sistema operacional ou paradigma específico.

O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 detalha a arquitetura e a implementação do Replica; a Seção 4 apresenta os experimentos e os resultados; a Seção 5 discute limitações e trabalhos futuros; e a Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

Diversas abordagens existem para o problema da reprodutibilidade de ambientes Linux, cada uma com diferentes níveis de abstração e escopo.

O NixOS [NixOS Foundation 2026] é uma distribuição Linux baseada no gerenciador de pacotes Nix, que adota uma abordagem puramente declarativa para a configuração do sistema, onde pacotes instalados, serviços ativos e configurações são descritos em arquivos de configuração versionáveis, garantindo reprodutibilidade total entre máquinas. Essa filosofia declarativa inspirou o desenvolvimento de módulos específicos para o gerenciamento de aplicações Flatpak dentro do ecossistema NixOS.

O *nix-flatpak* [Modena 2026] é um módulo NixOS que permite o gerenciamento declarativo de aplicações Flatpak, seguindo uma abordagem convergente: o estado descrito na configuração é aplicado ao sistema, mas não é necessariamente exaustivo, permitindo divergências entre builds. O módulo suporta a definição manual de *overrides* de permissões via a opção `services.flatpak.overrides`.

O *declarative-flatpak* [in-a-dil-emma 2026] adota uma abordagem congruente e

transacional, na qual as mudanças são aplicadas atomicamente, de modo que uma falha reverte toda a operação. Assim como o *nix-flatpak*, permite declarar *overrides* manualmente na configuração.

Ambas as ferramentas, contudo, são restritas ao ecossistema NixOS, exigindo a adoção dessa distribuição como sistema base. Além disso, as permissões precisam ser descritas manualmente pelo usuário na configuração. O Replica difere fundamentalmente nos seguintes aspectos: opera sobre qualquer distribuição Linux com Flatpak instalado e realiza a captura automática das permissões já configuradas no sistema, sem exigir intervenção manual do usuário.

O Ansible [Red Hat 2026a] oferece, através da coleção `community.general`, os módulos `flatpak` e `flatpak_remote` para a instalação declarativa de aplicações [Red Hat 2026b]. Esses módulos permitem descrever o estado desejado em um *playbook* YAML e aplicá-lo em qualquer sistema com Ansible instalado. No entanto, a construção do *playbook* é inteiramente manual, pois o usuário deve listar cada aplicação, repositório e permissão explicitamente. O Replica atua durante essa etapa, gerando um arquivo a partir da inspeção do ambiente em execução.

O conceito de Infraestrutura como Código (IaC) [Kumara et al. 2021] estabelece a prática de descrever e versionar o estado de ambientes de software em arquivos declarativos, permitindo sua reprodução automatizada. O Replica se insere nesse paradigma ao gerar automaticamente um artefato IaC a partir da inspeção do sistema, aproximando-se do que a literatura denomina *reverse* IaC ou geração automática de IaC [Chiari et al. 2022]. O *Terraformer* [GoogleCloudPlatform 2026] exemplifica essa abordagem no ecossistema *Terraform* ao inspecionar infraestruturas em núvem já provisionadas e produzir arquivos tf, json ou tfstate equivalentes. O Replica aplica esse mesmo paradigma ao contexto Flatpak, que, em vez de exigir que o operador descreva manualmente o estado desejado, faz com que a ferramenta inspecione o ambiente em execução e gere automaticamente o *playbook Ansible* ou *Shell Script* correspondente.

3. Replica

O Replica é uma aplicação gráfica para sistemas Linux que automatiza a exportação e reprodução de ambientes Flatpak não-declarativos. Desenvolvida em Python com GTK 4[1] e *libadwaita*[2], a ferramenta segue as Diretrizes de Interface Humana do GNOME (HIG) [GNOME Project 2026], que orientam o desenvolvimento de aplicações para o ecossistema. Os princípios de simplicidade e redução de esforço do usuário guiaram as decisões de design, onde a interface apresenta uma única tela com uma única ação principal, removendo a necessidade de conhecimento simultâneo da CLI do Flatpak e da sintaxe dos formatos de exportação suportados. A aplicação é distribuída como Flatpak, o que garante consistência de execução entre distribuições e serve como demonstração prática da própria proposta do trabalho.

A interface, ilustrada na Figura 1, apresenta uma tela principal com um seletor do tipo de artefato, atualmente suportando Ansible Playbook e Shell Script, e um botão de geração. Como o Replica também roda em sandbox Flatpak, ele acessa o sis-

[1] `https://docs.gtk.org/gtk4/`
[2] `https://gnome.pages.gitlab.gnome.org/libadwaita/`

tema hospedeiro via `flatpak-spawn --host flatpak`, viabilizado pela permissão `--talk-name=org.freedesktop.Flatpak`, declarada em seu manifesto. Ao acionar a geração, esse mecanismo invoca os subcomandos `list --app`, `remotes` e `override --show <app_id>`, coletando respectivamente: (1) aplicativos instalados; (2) repositórios configurados; e (3) overrides de permissão de cada aplicação. A Figura 2 apresenta um trecho de *playbook* gerado pelo Replica.

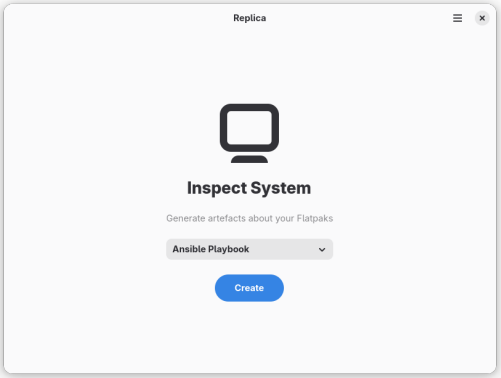


Figura 1. Tela inicial do Replica

```yaml
- name: Replica - Restore Flatpak Environment
  hosts: localhost
  become: false
  tasks:
  - name: Add remote flathub
    community.general.flatpak_remote:
      name: flathub
      state: present
      flatpakrepo_url: https://flathub.org/repo/flathub.flatpakrepo
      method: system
  - name: Add remote flathub
    community.general.flatpak_remote:
      name: flathub
      state: present
      flatpakrepo_url: https://flathub.org/repo/flathub.flatpakrepo
      method: user
  - name: Install Heroic
    community.general.flatpak:
      name: com.heroicgameslauncher.hgl//stable
      state: present
      remote: flathub
      method: system
  - name: Install Alpaca
    community.general.flatpak:
      name: com.jeffser.Alpaca//stable
      state: present
      remote: flathub
      method: system
```

Figura 2. Exemplo de *playbook* gerado

A inspeção é executada em uma *thread* separada para não bloquear a interface durante o processamento. Ao concluir, um diálogo de salvamento é apresentado ao usuário, que pode escolher o local e o nome do arquivo. O artefato gerado é então escrito em disco, e uma notificação confirma o sucesso ou fracasso da operação.

A arquitetura interna utiliza um sistema de registro de geradores baseado em decoradores Python, permitindo a adição de novos formatos de exportação sem alteração do código principal. O Replica está disponível como software livre sob a licença GPL-3.0[3].

4. Experimentos e Resultados

Para validar o Replica, foram conduzidos experimentos em quatro máquinas virtuais provisionadas com o Virtual Machine Manager, cada uma configurada com 2 GB de RAM, sendo duas executando Fedora 44, uma executando Ubuntu 24.04 e uma executando Arch Linux. O ambiente de origem (Máquina 1, Fedora 44) possuía 25 aplicações Flatpak instaladas, distribuídas em 5 repositórios remotos (Flathub, Flathub Beta, Fedora Flatpaks, Gnome Nightly e um repositório remoto customizado, não catalogado publicamente, criado para fins de avaliação), com 12 *overrides* de permissão configurados.

O experimento foi conduzido em três cenários: (1) homogêneo, com a reprodução do ambiente da Máquina 1 na Máquina 2, ambas executando Fedora 44; (2) heterogêneo, com a reprodução do ambiente na Máquina 3, executando Ubuntu 24.04; e (3) heterogêneo, com a reprodução do ambiente na Máquina 4, executando Arch Linux. Em todos os casos, o artefato gerado pelo Replica foi aplicado em uma instalação limpa do sistema

[3]`https://github.com/Nasc1mento/replica`

operacional, sem nenhuma aplicação Flatpak previamente instalada, sendo a única configuração feita a instalação do pacote *flatpak*, ausente por padrão no Ubuntu e no Arch Linux.

A Tabela 1 apresenta os resultados obtidos nos três cenários, comparando o estado do ambiente de origem com o estado restaurado após a execução do artefato gerado.

Tabela 1. Resultados da reprodução do ambiente Flatpak

Cenário	Apps instalados	Repositórios	Overrides
Homogêneo (Fedora → Fedora)	24	4	12
Heterogêneo (Fedora → Arch Linux)	24	4	12
Heterogêneo (Fedora → Ubuntu)	24	4	12

Após a aplicação do artefato nas máquinas alvo, a saída dos comandos `flatpak list --app`, `flatpak remotes` e `flatpak override --show <app_id>` nos ambientes restaurados foi comparada, por meio do utilitário `diff`, com a saída correspondente coletada no ambiente de origem. A comparação confirmou a ausência de divergências para 24 das 25 aplicações, 4 dos 5 repositórios remotos e os 12 overrides de permissão, em todos os cenários. A divergência corresponde a 1 repositório remoto customizado, que não constava na lista de repositórios públicos conhecidos utilizada pelo Replica e à aplicação nele hospedada, ambos não restaurados automaticamente, exigindo intervenção manual do usuário.

5. Limitações e Trabalhos Futuros

A principal limitação está na resolução de repositórios remotos customizados. A ferramenta resolve automaticamente os repositórios mais comuns, utilizando como base uma lista de repositórios públicos conhecidos [boredsquirrel 2026], incluindo Flathub, Flathub Beta, GNOME Nightly, Fedora Flatpaks, ElementaryOS, KDE Nightly, entre outros. No entanto, repositórios não catalogados são incluídos no artefato sem URL, exigindo intervenção manual para completar a configuração. Além disso, cenários como a indisponibilidade de uma aplicação ou branch específica no repositório de destino, ou divergências entre instalações `--user` e `--system` do mesmo aplicativo, não foram tratados pela ferramenta nem avaliados experimentalmente neste trabalho. Adicionalmente, a comparação com as abordagens declarativas discutidas na Seção 2 é argumentativa, não tendo sido conduzida uma avaliação empírica direta entre as ferramentas.

Como trabalhos futuros, pretende-se: (1) Implementar a resolução automática de URLs de repositórios customizados; (2) Publicar a aplicação no Flathub; E (3) dar suporte à exportação para formatos alternativos ao *playbook* Ansible e *shell script*, como módulos do *nix-flatpak* [Modena 2026] e do *declarative-flatpak* [in-a-dil-emma 2026], ampliando a compatibilidade da ferramenta com diferentes ecossistemas de gerenciamento declarativo de ambientes.

6. Conclusão

Este trabalho apresentou o Replica, uma aplicação gráfica que automatiza a exportação e reprodução de ambientes Flatpak via geração de Ansible Playbooks e Shell Scripts. Os experimentos em ambientes virtualizados validaram a proposta, demonstrando que os

artefatos gerados restauram fielmente o ambiente original em cenários homogêneos e heterogêneos de distribuição Linux, com exceção de repositórios remotos não catalogados.

O Replica está disponível como software livre sob a licença GPL-3.0, configurando-se como uma base extensível para evoluções futuras e contribuições da comunidade.

Referências

boredsquirrel (2026). Flatpak-remotes: A list of public flatpak repositories. https://codeberg.org/boredsquirrel/Flatpak-remotes. Acessado em: junho de 2026.

Chiari, M., De Pascalis, M., and Pradella, M. (2022). Static analysis of infrastructure as code: a survey. In *2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*, pages 218–225.

Flathub (2025). Flathub 2025 year in review. https://flathub.org/en/year-in-review/2025. Acessado em: junho de 2026.

Flatpak (2026). Repositories — flatpak documentation. https://docs.flatpak.org/en/latest/repositories.html. Acessado em: junho de 2026.

Flatpak Team (2026). Flatpak – the future of application distribution. https://flatpak.org.

GNOME Project (2026). Gnome human interface guidelines. https://developer.gnome.org/hig. Acessado em: junho de 2026.

GoogleCloudPlatform (2026). Terraformer: CLI tool to generate terraform files from existing infrastructure. https://github.com/GoogleCloudPlatform/terraformer. Acessado em: junho de 2026.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research1. *MIS quarterly*, 28(1):75–106.

Hochstein, L. and Moser, R. (2017). *Ansible: Up and Running – Automating Configuration Management and Deployment the Easy Way*. O'Reilly Media.

in-a-dil-emma (2026). declarative-flatpak: Declaratively manage flatpak installations in nixos and your $home. https://github.com/in-a-dil-emma/declarative-flatpak. Acessado em: junho de 2026.

Kumara, I., Garriga, M., Romeu, A. U., Di Nucci, D., Palomba, F., Tamburri, D. A., and van den Heuvel, W.-J. (2021). The do's and don'ts of infrastructure code: A systematic gray literature review. *Information and Software Technology*, 137:106593.

Modena, G. (2026). nix-flatpak: Install flatpaks declaratively. https://github.com/gmodena/nix-flatpak. Versão 0.7.0. Acessado em: junho de 2026.

NixOS Foundation (2026). Nix & NixOS: Reproducible builds and deployments. https://nixos.org.

Red Hat (2026a). Ansible documentation. https://docs.ansible.com.

Red Hat (2026b). community.general.flatpak – manages flatpak applications. Ansible Documentation.