

DATA SHEET · OPERAÇÃO ASSISTIDA

Operação assistida

Acompanhamento da plataforma GitLab implantada pelo pipeline Pointer, com ajustes por merge request e rotinas de dia 2 versionadas

CÓDIGO

PS-OPS-01

TIPO

Data sheet

VERSÃO

1.2

DATA

1 de outubro de 2026

O3S Consultoria e Tecnologia da Informação Ltda.
contato@pointertech.digital · +55 (61) 99874-4648 · pointertech.digital

GITLAB is a trademark of GitLab Inc. in the United States and other countries and regions.



GitLab
Select
Partner

DATA SHEET • PS-OPS-01

Operação assistida

Acompanhamento da plataforma GitLab implantada pelo pipeline Pointer, com ajustes por merge request e rotinas de dia 2 versionadas

Acompanhamento da plataforma GitLab entregue pelo pipeline de implantação da Pointer, no período acordado: ajustes de configuração por merge request no repositório do engajamento, execução das rotinas de dia 2 do pipeline (backup sob demanda, verificação avulsa, atualização de versão — sem parada na alta disponibilidade Linux e no Cloud Native Hybrid com backends em alta disponibilidade —, diagnóstico, reexecução das migrations do banco e as-built), manutenção da frota de runners do cliente, dos agentes GitLab para Kubernetes e da configuração do GitLab Geo, apoio a upgrades e análise de incidentes da plataforma. Os chamados do cliente são abertos e acompanhados na ferramenta de chamados da Pointer, com prazo de primeira resposta por severidade em horas úteis. Período, horário de atendimento, canais e prazos são definidos na proposta.

1. Para quem

- Clientes com a plataforma GitLab implantada pelo pipeline Pointer, em VMs Linux ou em Kubernetes, existentes ou criadas pelo pipeline, que querem acompanhamento no período seguinte à entrega.
- Clientes que precisam executar backup sob demanda, verificação, atualização de versão e diagnóstico pelas rotinas versionadas do projeto do engajamento.
- Clientes com frota de runners, agentes GitLab para Kubernetes, Container Registry ou GitLab Geo entregues pelo pipeline Pointer.

2. Onde executamos

O mesmo ambiente da implantação: VMs Linux ou cluster Kubernetes do cliente, em data center próprio ou em nuvem, inclusive a infraestrutura criada pelo pipeline na AWS ou no Azure. As rotinas rodam no runner do engajamento com acesso à rede do cliente, com as variáveis protegidas do projeto do engajamento.

3. Escopo do serviço

- **Acompanhamento** da plataforma no período acordado: uso, correções e ajustes de configuração, inclusive dos recursos entregues (Container Registry, escopos da busca global, KAS e Prometheus).

- **Ajustes de configuração por merge request** no repositório do engajamento: revisão, validação automática da configuração e aplicação pela implantação do pipeline, que repete o preflight, executa a verificação pós-implantação e gera o as-built atualizado.
- **Rotinas de dia 2** disponíveis no pipeline, conforme a trilha (tabela abaixo).
- **Apoio a upgrades:** nova versão definida por merge request, respeitando o caminho de upgrade oficial e, no Kubernetes, a matriz de versões do Kubernetes do Helm chart; backup sob demanda antes do upgrade; atualização sem parada (ZDU) no Cloud Native Hybrid com backends em alta disponibilidade ou PostgreSQL externo; na alta disponibilidade Linux, atualização sem parada sob consulta; nas demais topologias, reexecução da implantação com janela de manutenção; verificação avulsa depois da atualização.
- **Frota de runners e agentes GitLab para Kubernetes:** mudanças declaradas por merge request (runners e agentes novos, recriação e remoção) e verificação do estado de cada runner e das conexões de cada agente.
- **GitLab Geo:** reconfiguração da replicação a partir do site primário. O failover não é automatizado pelo pipeline.
- **Análise de incidentes da plataforma:** diagnóstico do cluster (Kubernetes, somente leitura), verificação avulsa, logs dos jobs e resultado da verificação pós-implantação; correção por merge request ou pelas rotinas de dia 2.
- **Atendimento de chamados:** ferramenta de chamados da Pointer dedicada ao cliente (painel web com senha de acesso; qualquer pessoa do cliente abre chamado, sem usuário no GitLab): categoria (incidente, solicitação, dúvida, mudança), severidade pelas definições de impacto da GitLab, histórico completo de cada chamado (comentários, status, responsável, tempo trabalhado, resolução), prazo de **primeira resposta** da Pointer em horas úteis por severidade (padrão: emergência 1 h, alta 4 h, média 1 dia útil, baixa 2 dias úteis, no expediente de 9h às 18h em dias úteis, ou conforme a proposta) e escalonamento ao GitLab Support, com o chamado aberto no Support Portal em nome do contato de suporte cadastrado do cliente e acompanhado pelo técnico.
- **Teste de backup e de restauração.** A restauração não tem rotina no pipeline nesta versão.
- **Ajustes de runners, integrações e políticas** conforme o escopo contratado.

4. Rotinas de dia 2 disponíveis no pipeline

• **Disponível** Ofertado nas condições deste documento.

◦ **Sob consulta** Escopo e condições definidos em proposta específica, com execução piloto.

Rotina	Trilha	O que faz	Estado
Backup sob demanda	Linux	Backup da aplicação e da configuração e segredos (/etc/gitlab) no servidor Rails primário; com object storage S3, envio ao bucket de backups	• Disponível
Backup sob demanda	Kubernetes	backup-utility no pod toolbox do release, gravando no bucket de backups do object storage; exclui GitLab Pages e Container Registry quando desabilitados	• Disponível
Verificação avulsa (somente leitura)	Linux	Mesmas checagens da verificação pós-implantação (serviços, versão, integridade da aplicação e do Gitaly, Patroni e Praefect, KAS, GitLab Pages, Prometheus e página de login), sem nova implantação	• Disponível
Verificação avulsa (somente leitura)	Kubernetes	Mesmas checagens da verificação pós-implantação (release Helm ou recurso do GitLab Operator, deployments e rollout, pods, versão, Gitaly, KAS, Prometheus e página de login; no Hybrid, VMs de backend), sem nova implantação	• Disponível
Atualização de versão sem parada (ZDU)	Kubernetes: Cloud Native Hybrid com backends em alta disponibilidade (Patroni e Consul, 3k ou mais) ou PostgreSQL externo	Playbook de zero downtime update do GitLab Environment Toolkit: backends em VM atualizados um a um e o chart com as migrations pós-deploy separadas; exige 2 ou mais réplicas de Webservice e Sidekiq	• Disponível
Atualização de versão sem parada (ZDU)	Linux, alta disponibilidade, com nós GitLab Rails dimensionados conforme a arquitetura de referência	Playbook de zero downtime update do GitLab Environment Toolkit para a versão definida por merge request, com backup sob demanda antes	• Disponível
Atualização de versão com janela de manutenção	Linux, nó único e 2k	Nova versão por merge request, backup sob demanda e reexecução da implantação	○ Sob consulta
Atualização de versão com janela de manutenção	Kubernetes: Cloud Native e Cloud Native Hybrid sem backends em alta disponibilidade nem PostgreSQL externo	Nova versão por merge request, backup sob demanda e reexecução da implantação pelo Helm chart ou pelo GitLab Operator	○ Sob consulta
Diagnóstico do cluster (somente leitura)	Kubernetes	Nós, recursos alocados, pods, jobs, volumes, services, gateway e rotas do namespace, 60 eventos mais recentes, estado e logs dos pods não prontos e logs do Job de migrations	• Disponível
Reexecução das migrations do banco	Kubernetes (instalação pelo Helm chart)	Recria o Job de migrations do release, acompanha até concluir e reinicia os deployments webservice e sidekiq; não se aplica à instalação pelo GitLab Operator	• Disponível

Rotina	Trilha	O que faz	Estado
Documento as-built	Linux e Kubernetes	Gerado a cada implantação verificada: versões, arquitetura, servidores ou componentes, infraestrutura criada pelo pipeline, recursos habilitados, nomes das variáveis secretas e resultado de cada checagem	• Disponível
Frota de runners do cliente: aplicação de mudanças e verificação	Linux e Kubernetes	Runners novos criados e instalados; recriação e remoção declaradas por merge request; relatório do estado de cada runner na instância (status, tags, proteção, timeout e versão dos managers)	• Disponível
Agentes GitLab para Kubernetes: aplicação de mudanças e verificação	Linux e Kubernetes	Agentes novos registrados e instalados; token novo e reinstalação ou remoção declarados por merge request; relatório das conexões de cada agente	• Disponível
GitLab Geo: configuração da replicação	Linux, dois sites	Playbook de Geo do GitLab Environment Toolkit executado a partir do site primário, com a autoridade certificadora do primário nos nós do secundário; pode ser executado de novo depois de ajustes	• Disponível

- **Sob consulta:** atualização de versão com janela de manutenção sobre ambiente em uso; escopo e condições definidos em proposta específica, com execução piloto.
- Atualização de versão sem parada na alta disponibilidade Linux: nós GitLab Rails dimensionados conforme a arquitetura de referência; o playbook espera cada nó Rails voltar antes de seguir.
- Atualização de versão sem parada no Kubernetes: somente Cloud Native Hybrid com backends em alta disponibilidade (Patroni e Consul, 3k ou mais) ou com PostgreSQL externo, com 2 ou mais réplicas de Webservice e Sidekiq; o pipeline recusa a rotina no Hybrid 2k com PostgreSQL em VM e na Cloud Native.
- O backup do Kubernetes não inclui o Secret gitlab-rails-secrets; no Linux, o as-built aponta a cópia do gitlab-secrets.json fora do servidor. As duas cópias ficam fora do servidor ou do cluster.
- As rotinas com falha não bloqueiam o pipeline; a análise é feita pelo log do job, e a rotina pode ser executada de novo.

5. Recursos configurados

- Nativo — recurso da plataforma GitLab
 - Requer configuração
- Integração externa — depende de serviço do cliente ou de terceiro

Recurso	Classificação	Detalhe
Backup do Linux package (gitlab-backup e gitlab-ctl backup-etc)	• Nativo	Comandos oficiais executados pelo pipeline no servidor Rails primário.

Recurso	Classificação	Detalhe
Envio do backup ao object storage	• Requer configuração	Com object storage S3 configurado na implantação, o backup vai ao bucket de backups.
Backup do Helm chart (backup-utility no toolbox)	• Nativo	Ferramenta oficial do chart, executada pelo pipeline no pod toolbox do release.
Verificação avulsa	• Requer configuração	Playbook de verificação da Pointer, o mesmo da verificação pós-implantação, executado sob demanda em modo somente leitura.
Zero downtime update do GitLab Environment Toolkit	• Requer configuração	Playbook oficial do GET, na alta disponibilidade Linux e, no Kubernetes, no Cloud Native Hybrid com backends em alta disponibilidade ou PostgreSQL externo.
API de runners da instância	• Nativo	Criação, consulta e remoção dos runners da frota pela API da instância (fluxo de tokens da versão 16 ou superior da plataforma GitLab), com o token de API do cliente.
API de agentes para Kubernetes da instância	• Nativo	Registro, tokens e consulta das conexões dos agentes pela API REST e GraphQL da instância.
Playbook de Geo do GitLab Environment Toolkit	• Requer configuração	Configuração da replicação entre os dois sites Linux package, executada a partir do site primário.
Environment protegido com aprovação	• Requer configuração	Recurso nativo configurado no projeto do engajamento: implantação, atualização de versão e aplicação de mudanças na frota e nos agentes exigem aprovação de pessoa diferente de quem executa.
Grupo de recursos por ambiente	• Requer configuração	Recurso nativo de CI/CD: duas execuções nunca alteram o mesmo ambiente, a mesma frota ou os mesmos agentes ao mesmo tempo.
Ferramenta de chamados da Pointer	• Requer configuração	Painel em GitLab Pages do projeto de chamados do cliente, com os dados cifrados e acesso por senha; cada chamado é um arquivo versionado no projeto, gravado pelo pipeline; relatório mensal gerado pelo pipeline.
GitLab Support (Priority Support)	• Nativo	Suporte do fabricante da assinatura do cliente: primeira resposta de 30 minutos (24x7) a 24 horas (24x5) conforme a severidade, para o chamado aberto no Support Portal pelo contato cadastrado do cliente; não é prazo de resolução.

6. Ferramentas

FERRAMENTA OFICIAL DA GITLAB

GitLab Environment Toolkit (GET)

Reaplica a configuração nas VMs ou no cluster a cada ajuste, executa o zero downtime update na alta disponibilidade Linux e no Cloud Native Hybrid com backends em alta disponibilidade e a configuração do GitLab Geo.

PACOTE OFICIAL DA GITLAB

Linux package (Omnibus)

Backup da aplicação (gitlab-backup) e da configuração (gitlab-ctl backup-etc) nas VMs.

CHART OFICIAL DA GITLAB

GitLab Helm chart

Backup pelo backup-utility do toolbox e Job de migrations do banco no Kubernetes.

FERRAMENTA OFICIAL DA GITLAB

GitLab Runner

Frota do cliente: pacote oficial nas VMs e chart oficial gitlab-runner no Kubernetes, instalados e verificados pelo componente de runners do pipeline.

CHART OFICIAL DA GITLAB (GITLAB-AGENT)

Agente GitLab para Kubernetes (agentk)

Instalado e reinstalado em cada cluster do cliente pelo componente de agentes do pipeline.

FERRAMENTA POINTER

psctl

Valida a configuração de cada merge request, avalia o preflight, cria e verifica os runners e os agentes pela API e gera o as-built.

7. Metodologia

Assessment → Implantação → Otimização → Transferência de conhecimento.

Fase 1 Assessment • Kick-off: período, horário de atendimento, canais e prazos de primeira resposta definidos na proposta; papéis e aprovadores das mudanças. • Habilitação da ferramenta de chamados: projeto de chamados do cliente, expediente, feriados e prazos da proposta, contatos cadastrados no GitLab Support, senha de acesso entregue ao ponto focal do cliente e chamado de teste. • Revisão do as-built e do repositório do engajamento; conferência do runner e das variáveis do projeto do engajamento.	Fase 2 Implantação • Ajustes aprovados por merge request e aplicados pela implantação do pipeline, com preflight, verificação e as-built. • Execução das rotinas de dia 2 sob demanda; implantação, atualização sem parada e mudanças na frota e nos agentes passam pela aprovação do environment protegido.	Fase 3 Otimização • Acompanhamento de uso e análise de incidentes da plataforma, com verificação avulsa e diagnóstico. • Atendimento dos chamados do cliente na ferramenta de chamados, com primeira resposta dentro do prazo por severidade e escalonamento ao GitLab Support quando é defeito da plataforma. • Apoio a upgrades: backup sob demanda, atualização de versão e verificação avulsa depois da atualização. • Teste de backup e de restauração. • Ajustes de runners, integrações e políticas conforme o escopo contratado.	Fase 4 Transferência de conhecimento • Sessão de transferência sobre as rotinas de dia 2 e o fluxo de mudança por merge request. • As-built atualizado e relatório de encerramento. • Encerramento conforme o escopo: remoção do runner do engajamento, revogação das variáveis e tokens e arquivamento do projeto do engajamento, ou entrega do repositório ao cliente.
--	---	--	--

8. Entregáveis

Entregável	Descrição	Formato
Relatório periódico de acompanhamento	Atividades do período, ajustes aplicados, rotinas executadas e incidentes analisados, com a seção de chamados do período (abertos, resolvidos, por categoria e severidade, primeira resposta no prazo, chamados no GitLab Support); periodicidade definida na proposta.	Documento

Entregável	Descrição	Formato
Ferramenta de chamados do cliente	Painel dedicado ao cliente para abrir e acompanhar chamados, com o histórico de cada um, o prazo de primeira resposta à vista e o número do chamado no GitLab Support quando houver escalonamento; entregue com senha de acesso no início do período.	Painel web (GitLab Pages)
Registro de mudanças	Merge requests aprovados no repositório do engajamento, com o pipeline que aplicou cada ajuste.	Repositório
Evidências das rotinas	Logs dos jobs, resultado da verificação pós-implantação e da verificação avulsa (retenção de 1 ano), arquivo de diagnóstico do cluster (retenção de 30 dias), relatórios da frota de runners e dos agentes (retenção de 1 ano) e registro dos backups executados.	Artefatos do pipeline
As-built atualizado	Documento gerado pelo pipeline depois de cada implantação verificada.	Markdown
Relatório de encerramento	Resumo do período e pendências.	Documento

9. Premissas

- A plataforma foi implantada pelo pipeline Pointer e o projeto do engajamento (configuração, variáveis e runner com acesso à rede do cliente) segue ativo no período: as rotinas de dia 2 dependem dele.
- Toda mudança de configuração, de versão, da frota de runners e dos agentes entra por merge request; a versão dos componentes do pipeline é fixa e só muda por merge request.
- A atualização de versão respeita o caminho de upgrade oficial da GitLab, uma versão minor por vez na atualização sem parada; no Kubernetes, também a matriz de versões do Helm chart.
- Os tokens de API da frota de runners e dos agentes e o acesso aos clusters seguem válidos no período.
- Período, horário de atendimento, canais e prazos definidos na proposta. O prazo da Pointer é de primeira resposta em horas úteis por severidade, não de resolução; o SLA do GitLab Support vale para o chamado aberto no Support Portal pelo contato cadastrado do cliente.
- O acesso à ferramenta de chamados é por senha entregue ao ponto focal do cliente, que a distribui a quem abre chamados; quem envia se identifica com nome e e-mail.

Fora do escopo

- Desenvolvimento de aplicações e dos pipelines das aplicações do cliente.

- Suporte do fabricante: o suporte oficial da GitLab é o da assinatura GitLab do cliente.
- Operação da infraestrutura: VMs, sistema operacional, cluster Kubernetes, rede, DNS, certificados, balanceador do cliente e serviços gerenciados (PostgreSQL, Redis, object storage).
- Rotinas fora do pipeline nesta versão: restauração automatizada e failover do GitLab Geo.
- Atualização de versão sem parada com nós GitLab Rails abaixo do dimensionamento da arquitetura de referência.

Responsabilidades do cliente

- Manter válidos os acessos do projeto do engajamento (chave SSH, kubeconfig, credenciais de nuvem, tokens de API e senhas) e o runner com acesso à rede do cliente.
- Aprovar as mudanças e as atualizações de versão pelos aprovadores definidos no kick-off.
- Prover janela de manutenção para atualização de versão nas topologias sem atualização sem parada disponível.
- Infraestrutura, rede e segurança: VMs ou cluster, DNS, certificados, balanceador, regras de firewall e contas de serviço.
- Guardar fora do servidor ou do cluster a cópia do gitlab-secrets.json (Linux) ou do Secret gitlab-rails-secrets (Kubernetes).

10. Duração

Período definido na proposta. A duração de cada rotina de dia 2 depende da topologia e do volume de dados da plataforma. Limites de tempo definidos nos jobs: atualização sem parada até 6 h, backup no Kubernetes até 3 h, reexecução das migrations até 2 h (a espera pelo Job de migrations é de até 60 minutos), configuração do GitLab Geo até 4 h, aplicação da frota de runners até 2 h e dos agentes até 1 h.

11. Documentos relacionados

- PS-00 · Catálogo de Serviços
- PS-IMP-01 · Implantação da plataforma GitLab em servidores Linux
- PS-IMP-02 · Implantação da plataforma GitLab em Kubernetes
- PR-IMP-01 · Pré-requisitos: Implantação da plataforma GitLab em servidores Linux
- PR-IMP-02 · Pré-requisitos: Implantação da plataforma GitLab em Kubernetes
- PS-ONB-01 · Onboarding do cliente na plataforma GitLab
- PS-CAP-01 · Capacitação na plataforma GitLab

O3S Consultoria e Tecnologia da Informação Ltda. · contato@pointertech.digital · +55 (61) 99874-4648 ·
pointertech.digital

PS-OPS-01 · versão 1.2 · 1 de outubro de 2026. GITLAB is a trademark of GitLab Inc. in the United States and other countries and regions. A Pointer é parceira GitLab Select.