SAP Inside Track
BARCELONA

ReleaseOwl®
Native DevOps Platform for SAP

Implementing gCTS for Continuous Delivery in Complex SAP S/4HANA Landscapes

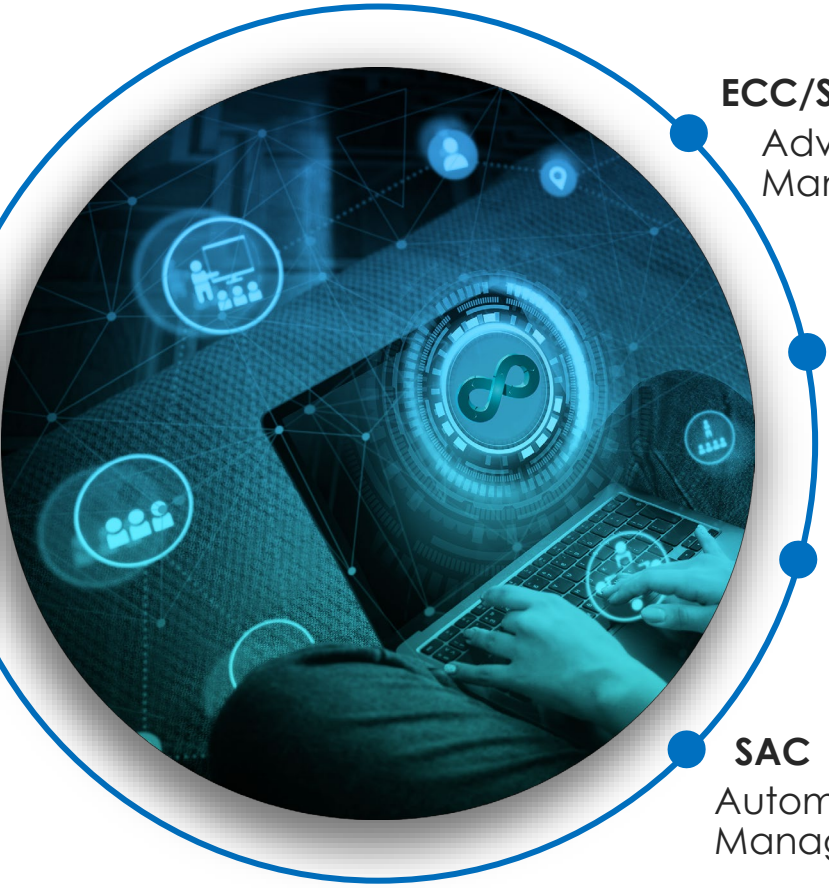
Branching Models, Hotfix Strategies, and the 5-Tier Architecture

Niranjan Gattupalli
Founder & CEO - ReleaseOwl

Agenda

- About ReleaseOwl
- The Role of gCTS in Modernizing SAP DevOps
- Understanding SAP S/4HANA Landscape Architecture
- Branching Models for Continuous Delivery
- Key Operations in gCTS for Robust SAP DevOps
- Key Considerations and Limitations in Implementing gCTS
- Branching Model for Continuous Delivery with gCTS
- HotFix Delivery
- HotFix Strategy with shared environment for BAU/Hotfix Dev and no dedicated test environment

About ReleaseOwl



ECC/S4
Advanced Transport Management

BTP
Continuous Delivery for SAP BTP

IS (CPI)
Automated IS Management

SAC
Automated SAC Management

SAP Certified Built on SAP Business Technology Platform | Available on SAP Store

Native Platform
- No Code DevOps Platform
- Built on SAP. Available on SAP Store

Future Ready
- Only platform for Business Technology Platform (BTP), SAP Integration Suite(CPI), SAP Analytics Cloud (SAC)
- Advanced Transport Management
- No additional IT Investments

Secure
- SAP Certified Architecture, Scalable
- ISO 27001
- Compliance with GxP, GDPR

Lowest TCO
- 65% Lesser Investment
- ROI in ~ 2 years

ROI
- 24X Faster Recovery from Failures
- 3X Lower Change Failure Rates
- 200X more frequent deployments

Integration
- Integration with Jira, Azure, Service Now & 4me
- Integration with Tosca, HCL One test, UiPath..

Recognized by CRN Magazine as one of the Top 10 DevOps Platforms of 2023

ReleaseOwl - The #1 Native SAP DevOps Platform

Advanced Transport Management for ECC, S/4HANA

- Transport Management
- Impact Analysis
- Web-Based Retrofit and Conflict Resolution
- Integrations with ATC, Code Inspector & gCTS

CI/CD Pipelines for SAP BTP

- Automated CI/CD Pipelines
- Robust Packaging & Deployments for MTAR, CAP & RAP
- Security Scanner for malware detection

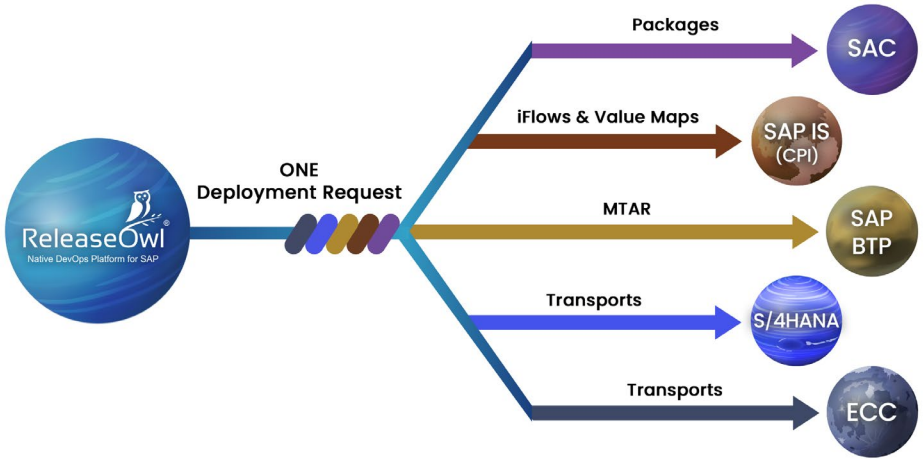
Secured & Accelerated Development for SAP Integration Suite

- Automated Packaging & Deployments
- Security Rule Engine
- CPI Regression Test Generator
- Backup & Rollback

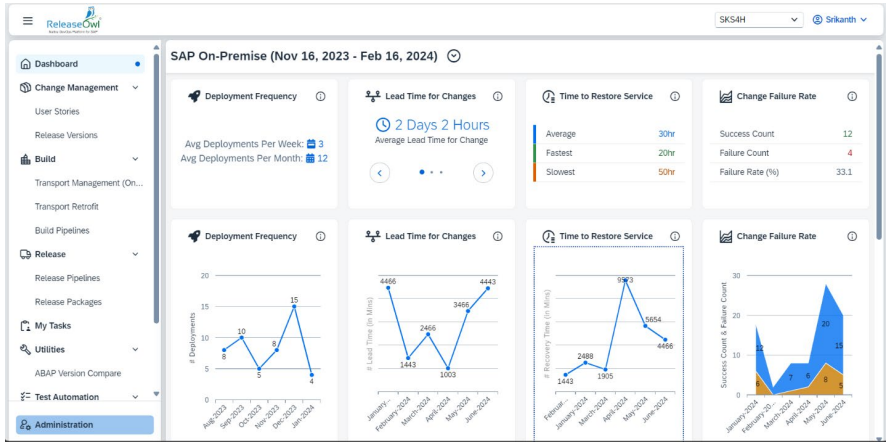
Streamlined Release Management for SAC

- Automated Deployment for SAC Packages
- Versioning and Downgrade Protection

Deployment Multiverse



Metrics Driven DevOps Maturity

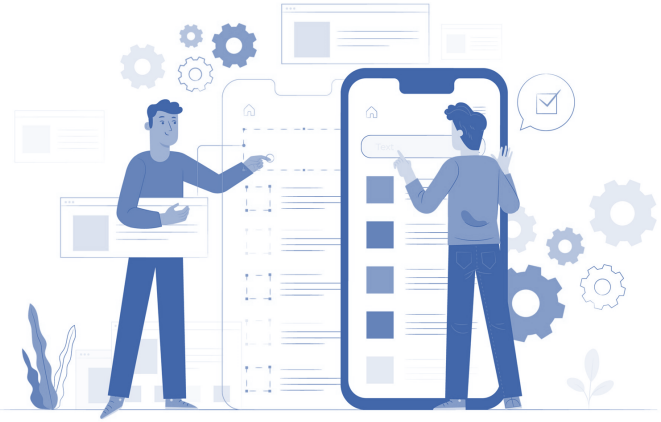


The Role of gCTS in Modernizing SAP DevOps

Key Points:

Bridging Traditional and Modern Practices

- Combines SAP's traditional Change and Transport System (CTS) with Git-based version control.



Key Features of gCTS:

- **Version Control:** Tracks all changes and enables collaborative development.
- **Branching and Merging:** Supports feature and hotfix isolation for parallel workstreams.
- **Automated Transports:** Simplifies movement of changes across SAP landscapes.

gCTS in Continuous Delivery:

- Enables faster, incremental deliveries with fewer risks.
- Integrates seamlessly with CI/CD pipelines for automated testing and deployment.
- gCTS provides OData and REST services, enabling seamless integration with external tools and systems.

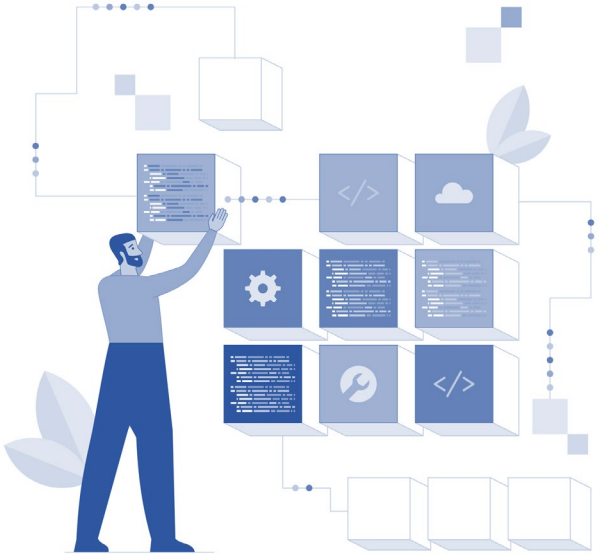
Understanding SAP S/4HANA Landscape Architecture

Key Components of the 5-Tier Architecture:

- **Development (DEV):**
  - Where active development and initial configurations occur.
- **Quality Assurance (QA):**
  - For integration testing and validation of changes.
- **Pre-Production (STAGING):**
  - Simulates the production environment for final testing.
- **Production (PROD):**
  - Live environment for end-users.

Maintenance (HOTFIX):

- Dedicated for critical patches and BAU (Business-As-Usual) fixes to ensure production stability.



Branching Models for Continuous Delivery in SAP Landscapes

Key Strategies for Branching:

- **Feature Branches:**
  - Used for isolated development of new functionality.
  - Promotes better collaboration and reduced conflict.
- **Hotfix Branches:**
  - Dedicated to addressing urgent production issues.
  - Ensures minimal disruption to ongoing development.
- **Master Branch:**
  - Represents the production-ready, stable codebase.
  - All validated changes merge into this branch for deployment.
- **Environment Branches:**
  - Environment-specific branches (e.g., DEV, QA, STAGING, PROD) to track progression.

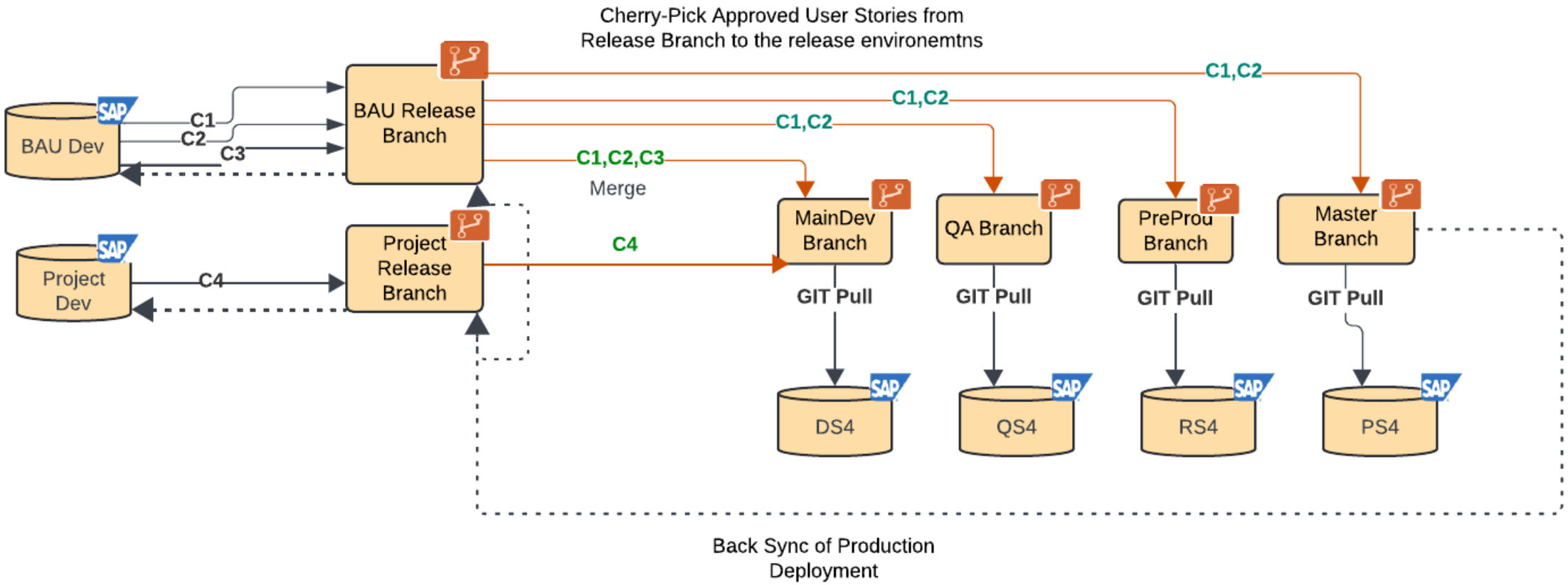
Key Operations in gCTS for Robust SAP DevOps

ReleaseOwl

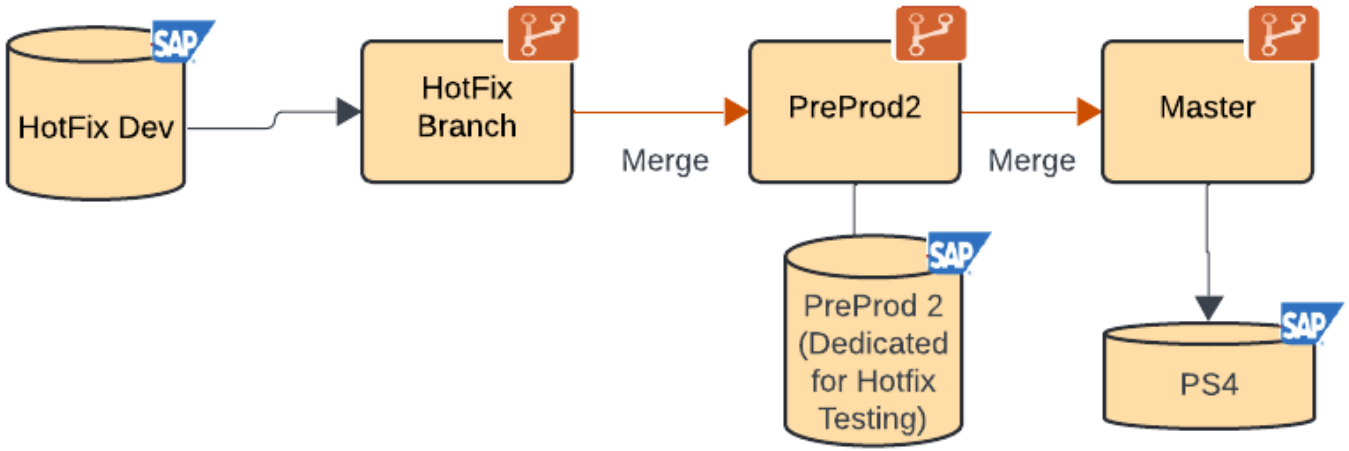
Core Operations

1. **gCTS Switch**
   - **Purpose:** Switch between branches or commits in a repository.
   - **Use Case:**
     - Move to a specific commit for testing or debugging.
     - Shift development focus between feature and hotfix branches.
   - **Key Workflow:**
     - Evaluate locks before switching.
     - Automatically prevent conflicts by listing locked objects.
   - **Outcome:** Seamless transitions across branches with minimized risks.
2. **gCTS Rollback**
   - **Purpose:** Revert a repository to a previous commit to discard changes or resolve conflicts.
   - **Use Case:**
     - Undo unstable deployments.
     - Recover from unexpected failures in PreProd or Production.
   - **Key Workflow:**
     - Identify a stable commit using the repository history.
     - Revert local repositories without affecting remote branches.
   - **Outcome:** Rapid recovery from deployment errors, ensuring landscape stability.
3. **gCTS Reset**
   - **Purpose:** Reset the local repository to a defined state, discarding uncommitted changes.
   - **Use Case:**
     - Clear partial developments.
     - Prepare for fresh feature integration or environment sync.
   - **Outcome:** Clean, consistent repository states ready for new operations.
4. **Cherry Picking (Selective Transport Management)**
   - **Purpose:** Apply specific changes without integrating the full branch.
   - **Use Case:**
     - Migrate critical fixes or isolated improvements between environments.
   - **Outcome:** Greater flexibility in managing specific updates, reducing downstream risks.

Branching Model for Continuous Delivery with gCTS

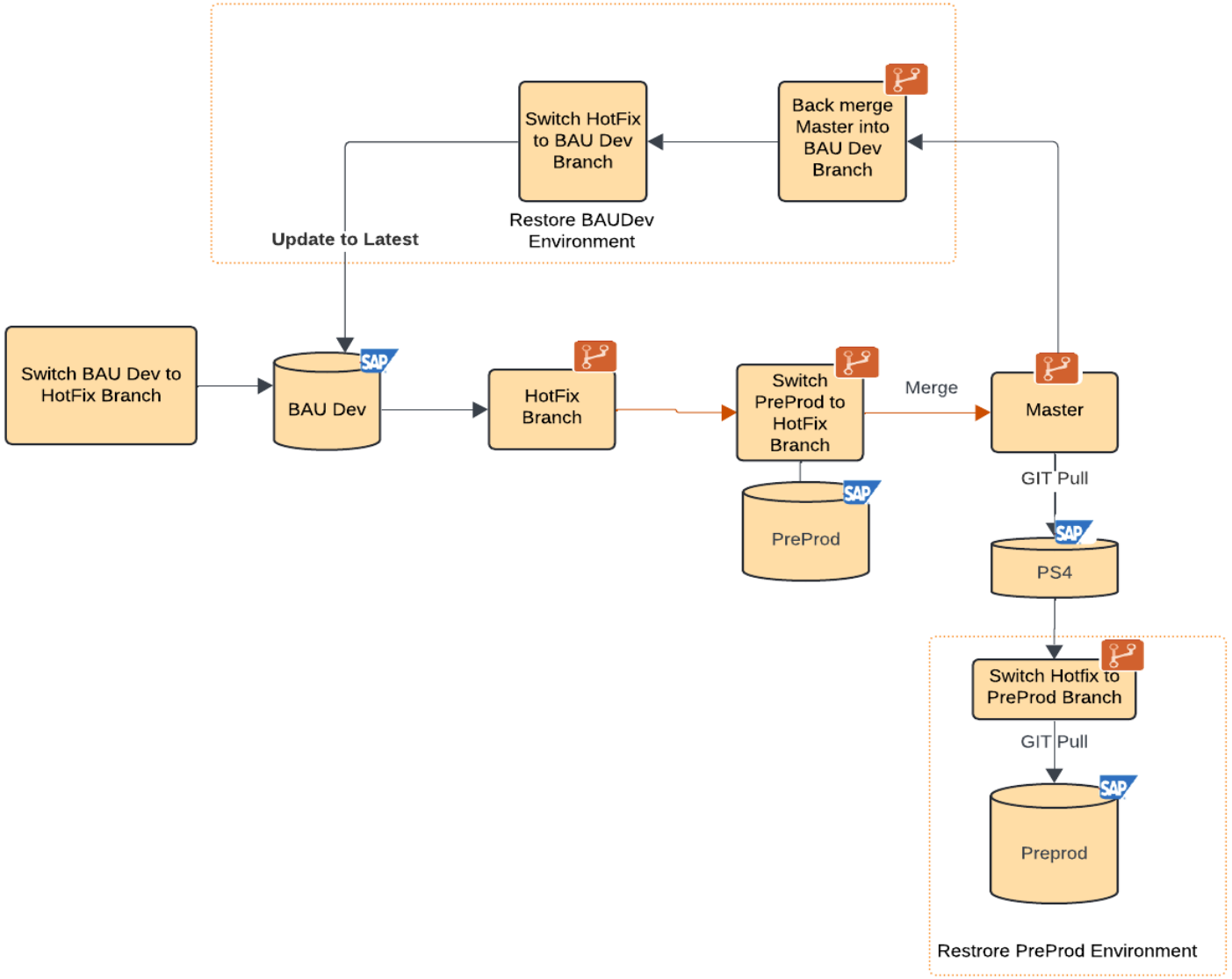


HotFix Delivery



- **Issue Identification:** A critical issue is identified in the production environment.
- **Hotfix Branch Creation:** A dedicated branch is created in the Git repository to isolate the fix.
- **Development in Maintenance Landscape:** The fix is developed and validated in the **Maintenance environment**, ensuring minimal disruption to ongoing projects.
- **Fast-Tracked Deployment:** The hotfix is deployed directly to production via an accelerated transport process.
- **Back-Merge to Development:** The hotfix is merged into the main development stream to keep all environments consistent.

HotFix Strategy with shared environment for BAU/Hotfix Dev and no dedicated test environment



- **Switch BAU Dev to Hotfix Branch:**
  - Transition the BAU Dev environment to the Hotfix branch for urgent fixes, isolating the changes from ongoing development.
- **Hotfix Development & Merge:**
  - Develop and validate the fix in the Hotfix branch.
  - Merge the validated changes into the Master branch for production readiness.
- **Update PreProd for Hotfix Testing:**
  - Switch PreProd to the Hotfix branch and pull the updates to test the fix in a near-production environment.
- **Deploy Hotfix to Production:**
  - Merge the Hotfix branch to Master and Deploy to Prod.
- **Restore Pre-Prod:**
  - After deployment, restore PreProd environments to their respective git Branches
- **Back-Merge to BAU Dev:**
  - Ensure all changes are merged back into the BAU Dev branch to maintain consistency across the landscape.
- **Restore BAU Dev:**
  - Restore PreProd environments to their respective git Branches to continue development

Key Considerations and Limitations in Implementing gCTS

ReleaseOwl

Branch-System Tied Limitation

- **Consideration:** In gCTS, branches are statically tied to SAP systems, restricting flexibility for evolving feature branch strategies and complicating parallel development workflows. This requires careful branch-to-system mapping and innovative solutions for dynamic deployments and shared environments.

Downgrade Protection Challenge:

•**Consideration:** gCTS does not natively support downgrade protection, increasing the risk of older object versions overwriting newer ones during transport execution. This is critical for maintaining system integrity in complex SAP landscapes.

Changes in Remote Repository - Syntax Validation is Not Done

- **Consideration:** When changes are made directly in the remote Git repository (e.g., through manual updates or external integrations), syntax validation is not performed at the time of the commit or push.

Cherry Picking - Risks of Inconsistencies

- **Consideration:** Cherry picking involves selecting specific commits to apply them to another branch. While this allows flexibility in transporting selective changes, it can lead to inconsistencies if dependencies between objects or changes are overlooked.

User and Team Synchronization:

- **Consideration:** Synchronizing users and teams between ABAP systems and gCTS requires attention. User IDs may differ, and while gCTS can synchronize teams by creating them and adding existing users, discrepancies can occur if not managed properly.

gCTS: Transforming SAP Development with Modern DevOps Practices

ReleaseOwl

- **Version Control Meets SAP:**
  - gCTS integrates Git's powerful version control with SAP's traditional Change and Transport System, enabling modern, agile development workflows.

- **Enabling Continuous Delivery:**
  - By automating transport management and leveraging branching strategies, gCTS simplifies deployments and enhances development speed and reliability.

- **Flexibility and Extensibility:**
  - With OData and REST APIs, gCTS allows seamless integration into CI/CD pipelines, enabling custom workflows and advanced transport validations.

- **Overcoming Challenges:**
  - While gCTS requires addressing considerations like downgrade protection and static branch-to-system mappings, these can be mitigated with strategic planning, automation, and extensions.

- **Future-Proof SAP Landscapes:**
  - gCTS positions organizations to adopt DevOps best practices, improve collaboration, and achieve operational excellence in SAP landscapes.

"*gCTS is not just a tool but a pathway to modernizing SAP DevOps workflows. Its integration of Git with traditional SAP landscapes brings a level of agility, traceability, and automation that aligns with the demands of today's fast-paced business environments. By addressing its challenges and leveraging its extensibility, organizations can unlock the full potential of their SAP systems, setting the stage for continuous innovation and scalability.*"

SAP Inside Track
BARCELONA

ReleaseOwl®
Native DevOps Platform for SAP

Thank You!

success@releaseowl.com