MDChange was upgraded in 2025 to resolve the many issues faced by IBM i Git users, while providing powerful automation for the entire DevOps flow.

Automated Git Branch Creation

MDChange automatically creates Git feature branches based on task creation, approval, state transitions, or manual triggers, ensuring every change request gets its own isolated development branch with consistent naming conventions.

Feature-Branch Test Environments

MDChange creates complete isolated IBM i environments for each feature branch, automatically provisioning libraries, IFS folders, test data, and all dependencies—enabling parallel development and testing without conflicts, matching container-based workflows used by cross-platform teams.

Simplified Export of Source Members & IFS Directories into Git

MDChange provides streamlined export capabilities through MDSRC2IFS command and VSCode/RDi integration, automatically converting source members to UTF-8, normalizing line endings, and enabling one-click IFS directory exports to Git with configurable filtering and branch management.

Granular or Bulk Git Commits Supported

MDChange supports flexible commit strategies, allowing developers to commit incrementally or work locally until ready, then automatically comparing changes with origin and creating bulk commits and pushes of all modifications.

Git Workspace Synchronization and Status Visibility

MDChange continuously monitors and compares local workspace, local Git repository, and compiled IBM i objects, providing clear recommendations and preventing deployment disasters from unnoticed environment drift while automatically notifying external tools throughout the development lifecycle.

Huge Git Repositories

MDChange automatically distributes high-volume Git operations across service jobs with intelligent load balancing, enabling parallel processing from multiple repositories to eliminate bottlenecks during large clones, bulk compilations, and multi-developer branch operations.

Automated Pull Requests

MDChange uses bi-directional REST integration to automatically create and manage Git pull requests upon successful environment promotions, with fully customizable titles and configurable target branches for multiple merge levels.

Seamless Project Management Integration

MDChange integrates directly with Azure DevOps, ServiceNow, and Jira to automatically synchronize task status throughout the development lifecycle, eliminating manual updates and providing real-time visibility to project managers and stakeholders.

Post Merge Deployment Triggering

MDChange can be triggered via webhook by successful Git merges to automatically start cross-platform deployment pipelines, creating complete feature branch deployment sets including all non-Git components while tracking the original change request.

Granular, Time-Sensitive Deployments

MDChange enables feature-based deployments on independent schedules rather than monolithic maintenance windows, allowing specific changes to promote independently with maintained dependency tracking and audit trails while keeping production systems live until planned switches.

Automatic Rollback and Risk Mitigation

MDChange provides automatic, object-level rollback capabilities that instantly revert to previous stable states if deployment errors occur, reducing deployment risk & enabling frequent releases with confidence.

Cross-Platform Deployment Guarantee

MDChange coordinates with external CI/CD systems to ensure IBM i and non-IBM i components deploy simultaneously with atomic transactions, automatically rolling back both platforms if either deployment fails.

Bidirectional Enterprise Integration

MDChange's REST APIs enable true bidirectional communication with CI/CD toolchains, receiving triggers from external systems and sending status updates, making IBM i a first-class citizen in enterprise DevOps workflows.

MDChange Bridges the Git Gap - Handling All IBM i Entities

Git excels at managing source code, but has no native understanding of IBM i's unique ecosystem. Traditional Git-based version control systems cannot handle the complexity of IBM i environments

Data Entities
Physical files, Logical files, Multi-member files, DDM files, Data areas, Individual data records

OPM & Message Files
OPM Programs, Message Files, Display Files, Printer Files

SQL Entities
Tables & temporal tables, Partitioned tables, Views & indexes, Functions & procedures, Sequences & constraints, Triggers & UDTs, Field procedures

Job Management
Job descriptions, Subsystems, Job queues, Workload groups

ILE Entities
Modules, Service programs, Binding directories, Programs, SQL packages, Compiler directives

Object Authority
Public authority, Authorization lists, Primary groups, IFS data authority

The MDChange Advantage

Traditional Approach	With MDChange
Manual test environment setup	Automated provisioning
Days to prepare test environment	Minutes to deploy feature branch
Manual data refresh	Automated data transformation
Full production data copies	Intelligent data sub-setting
Shared environments = bottlenecks	Isolated environments = parallel work
"It works on my machine" syndrome	Consistent, reproducible environments
Manual object deployment	Automated, dependency-aware deployment
High risk of missing dependencies	Complete object tracking and management
Hours of data preparation per test	Test-ready data in minutes
Sequential repository operations	Parallel repository processing
Manual pull request creation	Automated pull requests from tasks
All-or-nothing commit strategy	Granular or bulk Git commits supported

Real World Impact

MDChange brings modern DevOps practices to IBM i without sacrificing the platform's legendary reliability and performance. It's time to bridge the gap between Git's power and IBM i's unique requirements.