전자정부 프레임워크 기반 프로젝트의 시스템 메뉴 단위 변경 이력 관리를 위한 Git 커밋 로그 활용 연구

엄윤진, 박민호*
숭실대학교*

flow@kakao.com, * mhp@ssu.ac.kr

A Study on the Utilization of Git Commit Logs for the Management of System Menu Unit Change History for E-Government Framework-Based Projects

Um Yoon-Jin, Park Min-Ho*
Soongsil Univ.*

요 약

본 논문에서는 전자정부 프레임워크를 기반으로 하는 프로젝트에서 Git 이력 관리 시스템을 보다 효과적으로 활용하기 위한 시스템 메뉴 단위 변경 이력 관리 방안을 제안한다. 전자정부 프레임워크 기반 프로젝트에서는 변경 이력의 체계적인 관리를 위해 전자정부 프레임워크의 파일 구조와 Git Commit 로그를 활용하여 각 화면의 구성 요소 및 파일 간의 연관 관계를 체계적으로 정리함으로써 변경 이력을 효과적으로 추적할 수 있는 기반을 마련한다. 이 논문은 각 Commit 에 기록된 변경 이력과 관련된 파일을 메뉴 아이디와 연관시켜 특정 작업 또는 변경 사항을 더욱 효과적으로 파악할 수 있도록 한다. 시스템 메뉴 단위 변경 이력 관리의 효율성은 실험적으로 검증되었으며, Git Commit 로그의 중요성을 고려하여 효과적인 메시지 작성 방법도 소개한다. 제안된 방법은 전자정부 프레임워크 기반 프로젝트에서 파일 간의 연관 관계를 체계적으로 관리함으로써 관련 파일을 찾는 데 필요한 시간과 노력을 현저히 감소시킴을 확인하였다. 따라서, 본 연구는 Git 을 통한 버전 관리를 효과적으로 활용하여 프로젝트를 보다 효율적으로 관리하고 유지 보수하는 데에 중요한 의미를 가진다.

주제어 : 전자정부프레임워크, Git, Commit log, 이력관리

Ⅰ. 서 론

전자정부 프레임워크는 국가 표준프레임워크로서 다양한 공공사업 프로젝트에서 널리 사용되고 있으며[1], 이 프레임워크를 기반으로 하는 프로젝트에서의 이력 관리는 프로젝트의 복잡성과 다양한 모듈 간의 의존성이 높아지면서 소스 파일의 변경 이력을 효과적으로 추적하기 어려운 문제가 발생하고 있다. 특히, 시스템 메뉴 단위에서의 변경은 여러 파일 및 모듈 간에 분산되어 있어 이를 효과적으로 관리하는 것이 어렵다. 기존의 이력 관리 방법에서는 변경 이력을 파일 단위로만 파악할 수 있어, 특정 화면에 대한 변경 사항을 정확하게 이해하기 어렵다.

Git 의 커밋 로그는 프로젝트의 변경 이력, 작업 내용, 팀원 간의 협업 내용 등 다양한 정보를 자세하게 담고 있다.[2] 그러나 기존의 이력 관리 방법에서는 Git Commit 로그를 단순히 파일의 변경 사항을 나열하는 정도로 활용하고 있어, 시스템 메뉴 단위의 변경 이력을 명확하게 추적하기 어렵다. 또한, Commit 메시지의 품질은 제각각이며, 특정 작업이나 화면의 변경 내용을 파악하기 어려운 경우가 많다. 이로 인해 개발자 간의 의사소통이 어려워지며, 유지보수 및 버그 수정 작업이 복잡해지는 결과가 나타난다. 본 연구는 이러한 어려움과 한계를 극복하기 위해, 전자정부 프레임워크를 기반으로 하는 프로젝트에서의 Git 을 통한 이력 관리를 시스템 메뉴 단위로 효과적으로 확장하는 방안을 탐구한다. 이를 통해 개발자들은 시스템의 특정 화면에 대한 변경 이력을 보다 명확하게 파악할 수 있을 것으로 기대된다.

Ⅱ. 연구방법

2.1. Git Commit Convention 및 탬플릿 작성

Git 은 현존하는 버전 관리 시스템 중 하나로, 코드의 변경 이력을 체계적으로 관리하는 데 사용된다. 특히 시스템 메뉴 단위 변경에 있어서는 명확하고 간결한 커밋 메시지가 필수적이다. Git 커밋 메시지 컨벤션을 따르면 커밋 로그의 가독성이 향상되고, 코드 유지보수가 더 쉬워진다.[3] 프로젝트를 효율적이고 안정적으로 관리하기 위해 Git 커밋 메시지 템플릿을 설정하는 규칙을 정하고 이를 준수하는 것이 중요하다.[4]

2.2. 프로젝트 구조 분석

전자정부 프레임워크를 기반으로 하는 프로젝트에서의 시스템 메뉴 단위 변경 이력을 효과적으로 관리하기 위해서는 프로젝트의 구조를 명확하게 이해하는 것이 필수적이다. 이를 위해 먼저 MVC(Model-View-Controller) 파일 구조(그림 1)를 분석하여 각 메뉴 화면이 어떤 Controller, Service, ServiceImpl, SQL xml 파일과 관련되어 있는지를 각 파일의 역할과 협력 관계를 정리한다. xml 파일이 화면과 사용자 입력 간의 상호 작용을 처리하는 Controller 를 호출하고, Service 및 ServiceImpl 이 비즈니스 로직을 담당하며, SQL xml 파일은 데이터베이스와의 상호작용을 정의한다. 이때 각 화면(xml 파일)은 관리자가 메뉴 아이디를 부여하여 프로젝트에서 관리하며 특정 기능을 식별하게 된다.

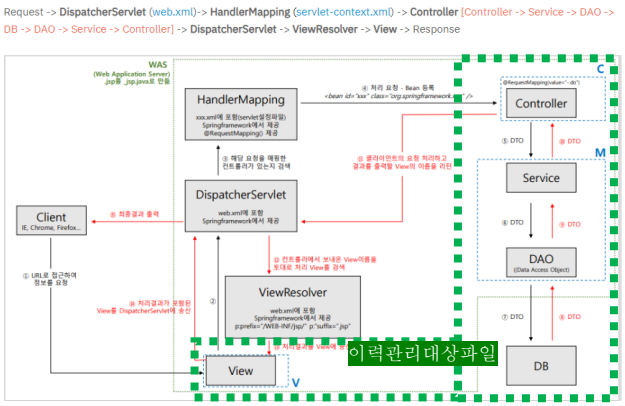

그림 1. Spring MVC 전체 동작 순서[5]

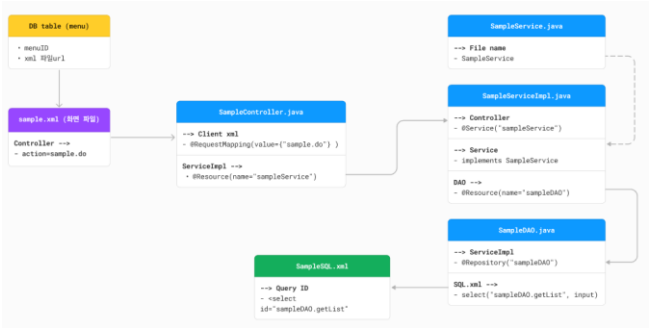

그림 2. 파일간 연결 관계

2.3. 메뉴 아이디와 연관 파일 맵핑하기

첫번째, 프로젝트의 구조를 파악해서 분석하고 두번째, 메뉴 아이디가 부여된 각 화면과 Controller, Service, ServiceImpl, SQL xml 파일 목록을 추출한다. 세번째, Spring Framework 에서의 파일들이 서로 구조적으로 갖는 상관 관계를 로직으로 구현하여 파이썬 코드를 작성한다. 마지막으로 코드를 실행하여 파일마다 메뉴 아이디를 맵핑한다.

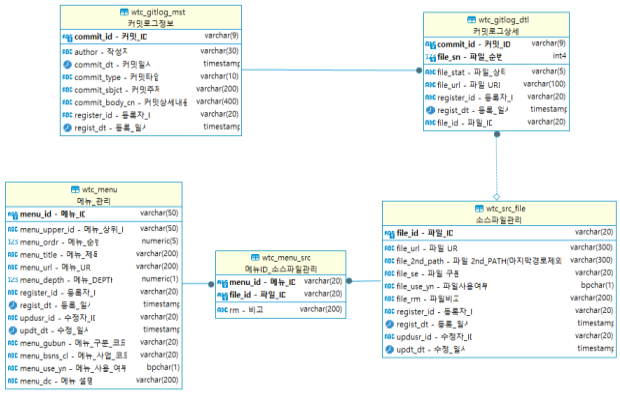

그림 3. 메뉴, 소스파일, Git Commit Log 관계 모델

2.4. Git Commit 로그의 활용

프로젝트의 구조를 파악한 후, Git API 를 활용하여 커밋 로그 데이터 자동 수집하고[6], 커밋 메시지에서 필요한 항목을 추출하여 이를 통해 작업 내용과 변경 내역을 데이터베이스에 저장한다. 저장한 Git Commit 로그를 효과적으로 활용하여 변경 이력을 관리한다. 각 Commit 에는 어떤 파일이 어떤 변경 내용을 포함하고 있는지가 기록되어 있다. 이 정보를 기반으로 각 화면에 대한 변경 이력을 추적하고자 한다

Ⅲ. 시스템 구현

Jupyter Lab 을 사용하여 코드 실행 및 이력 조회 작업을 웹에서 수행할 수 있도록 시스템을 개발하였다. 각 메뉴 아이디를 부여한 파일 목록 정보와 추출한 Git Commit Log 정보들을 서로 관계를 가질 수 있도록 테이블을 설계하고, 이를 데이터베이스에 저장한다. 메뉴 아이디별로 Git commit log 를 쉽게 확인할 수 있도록 정보들 간의 연관관계를 RDBMS 쿼리로 조회하여 구현하였다. 시스템 메뉴별 변경이력을 검색하기 위해 IDE Tool 에서 메뉴에 해당하는 파일을 찾고, 관련 파일의 변경 이력을 Git 시스템에서 검색하는 것과 비교하여 웹에서 구현된 기능을 통해 소스 파일을 검색하는 횟수와 시간이 감소되었다.

그림 4. 메뉴 별 변경이력 조회 구현 화면

Ⅳ. 결론 및 향후 연구

본 논문에서 제안한 방안은 전자정부 프레임워크를 기반으로 하는 프로젝트에서의 Git 을 통한 시스템 메뉴 단위 변경 이력 관리에 적합한 모델을 제시하였다. 향후 연구에서는 다음과 같은 방향으로 진행할 계획이다.

첫째로, 제안한 방안을 다양한 프로젝트와 기관에 적용하여 실제 활용성을 검증하는 것이 중요하다. 다양한 규모와 복잡성을 가진 프로젝트에서의 적용 가능성과 효과를 확인함으로써 제안한 방법의 실제적인 유용성을 입증할 필요가 있다. 둘째로, Git Commit 메시지의 자동 생성 및 품질 향상을 위한 도구나 기술에 대한 연구가 필요하다. 특히, 메뉴 아이디와 변경 내용을 자동으로 추출하고 명시하는 방법을 보완하여, 개발자들이 보다 쉽게 효과적인 Commit 메시지를 작성할 수 있도록 도움이 되는 방법을 연구하고자 한다.

참 고 문 헌

[1] 표준프레임워크 포털. https://www.egovframe.go.kr

[2] https://git-scm.com/docs/git-log

[3] https://cbea.ms/git-commit/

[4] https://yozm.wishket.com/magazine/detail/1974/

[5] Spring 기본동작순서/구조. https://server-engineer.tistory.com/253

[6] https://git-scm.com/docs/pretty-formats