
정보시스템 분야 Stack 통합 Test 결과보고서 [Trac]

2015. 07.

목 차

I. Stack 통합 테스트 개요	1
1. 목적	1
II. 테스트 대상 소개	2
1. Trac 소개	2
2. Trac 주요기능 및 특징	5
3. Trac 시스템 요구사항	7
III. Stack 통합 테스트	8
1. 테스트 환경	8
2. 테스트 접근 방법	10
3. 테스트 수행	12
4. 테스트 수행 결과	13
IV. 종합	14
※ 참고자료	15
[별첨1] Trac 테스트 케이스	

I. Stack 통합 테스트 개요

공개SW Stack 통합테스트는 여러 공개SW들의 조합으로 시스템 Stack을 구성한 후 Stack을 구성하는 공개SW의 상호 운용성에 중점을 두고 기능 및 성능테스트 시나리오를 개발하여 테스트를 진행한다.

본 통합테스트를 통해 안정적인 Stack 정보를 제공하여 민간 및 공공 정보시스템 개발 및 도입 시 활용될 수 있도록 한다.

1. 목적

□ 공개SW Stack 통합 테스트 수행 목적

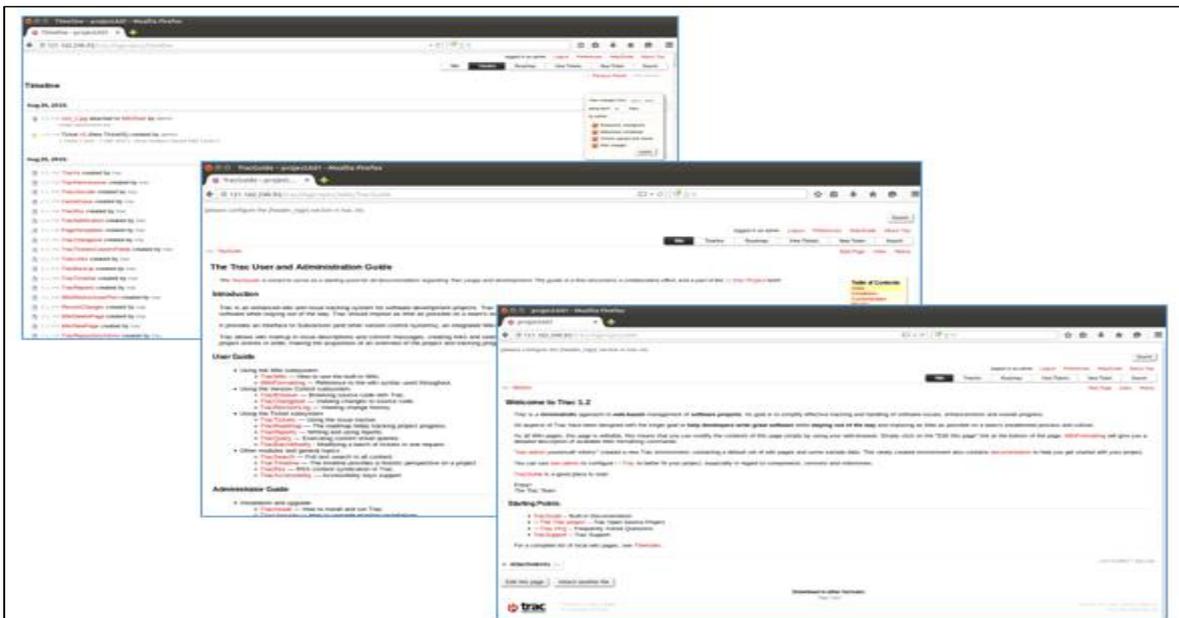
- 공개SW로 구성된 Stack이 유기적으로 잘 동작함을 확인
- 다양한 Stack 구성에 기반을 둔 테스트를 통해 안정된 Stack 조합 규명
- 공개SW 시스템 도입을 위한 Stack 참조모델의 신뢰성 정보로 활용
- 공개SW의 신뢰성과 범용성에 대한 사용자 인식 제고

II. 테스트 대상 소개

1. Trac 소개

Trac은 오픈소스 프로젝트 활성화의 일환으로 Python 기반으로 개발되었으며, 소프트웨어 개발 프로젝트의 관리에 특화된 이슈 트래킹 시스템이다. Trac은 이슈 Description과 Commit 메시지에 대한 Wiki Markup 및 버그, Task, Changeset 정보와 파일, Wiki 페이지 간의 참조 링크 기능을 포함하는 향상된 Wiki 기능과 함께 Timeline 및 Roadmap 기능을 근간으로 팀의 정립된 개발 프로세스 및 정책을 효율적으로 반영하여 관리할 수 있는 웹 기반의 소프트웨어 개발 관리 환경을 제공한다.

프로젝트 관리도구로서 Trac은 현재, Modified BSD 라이선스가 적용되고 있다. 또한, Subversion, Git, 머큐리얼(Mercurial), Bazaar 등 버전 관리 소프트웨어와 연동을 통해 기능 확장성을 제공하며, 개발 프로세스에 따른 발생 이슈 관리 및 버그 추적 관리 기능을 제공한다.



[그림 1. Trac 주요 사용화면 구성]

□ 관련 기술 및 개념 이해

> 머큐리얼(Mercurial)

- 소프트웨어 개발을 위한 크로스 플랫폼 분산 버전 관리 도구
- Revision 구분 관리하는 데 SHA-1 해시 사용
- ssh 상에서 실행 가능
- Microsoft Windows, MAC OS X, Unix 계열의 크로스 플랫폼 지원
- GPL 라이선스 적용

> Bazaar

- 캐노니컬(Canonical Ltd.)이 지원하는 분산 버전 관리 도구
- GNU/리눅스 배포판과 Microsoft Windows, MAC OS X 지원 패키지 지원
- GPLv2 또는 이후 버전 라이선스 적용

> Git

- 소스코드 관리를 위한 분산 버전 관리 도구
- 폴더 관리 체계로 구성되어 있으며, 각 폴더는 전체 기록과 각 기록을 추적할 수 있는 정보를 포함
- Microsoft Windows, MAC OS X, Linux, POSIX 운영체제 지원
- GNU 일반 공중 사용 허가서 v2 (GPLv2) 적용

> SVN(Subversion)

- 자유 소프트웨어 버전 관리 도구
- 제한이 있는 CVS 대체위해 2000년부터 콜랩넷에서 개발
- 서버, 클라이언트 모델을 따르며, http, https, svn, svn+ssh 의 규약 따름
- 크로스 플랫폼 운영체제 지원
- Apache 라이선스 적용

> CVS (Current Version System)

- 소프트웨어 개발을 위한 동시 버전 관리 도구로 파일 단위의 모든 작업 내용과 변화 히스토리 추적 관리 기능 제공
- Microsoft Windows, Unix 계열 운영체제 지원
- GNU GPL 라이선스 적용

2. Trac 주요기능 및 특징

Trac은 버전 관리 소프트웨어의 웹 인터페이스를 제공하며, 프로젝트 진행에 따른 각종 개선점과 개발 과정에서의 발생 버그 관련한 이슈 트래킹, 그리고 Wiki를 통한 문서 관리 및 각 리소스 연동을 주 기능으로 한다.

□ Trac의 특징

- > Ticket 정보 처리 단위로 프로젝트 별 계정 권한에 따른 업무 분배 및 스케줄링 기능 제공
- > 버전관리 소프트웨어 연동 및 Plugin 설치, 설정에 따른 기능 확장성 제공
- > Wiki를 기반으로 한 문서관리 및 Customized Report 기능 제공

□ Trac의 주요기능 카테고리

- > Project management 기능 (Roadmap, Milestones, etc.)
- > Account Manager 기능 (Account Manager Plugin 적용)
 - Web 관리자 및 각 역할 별 계정 권한 설정
- > Ticket system 기능 (bug tracking, tasks, etc.)
 - 업무 프로세스에 따른 발생 이슈 및 이벤트 처리
- > Search engine 기능
 - 작업 처리 단위인 Ticket 정보 선택적 조회
- > Email notifications 기능
 - 등록 계정에 대한 발생 이벤트 및 업무 정보 알림
- > Fine-grained permissions 기능 (since version 0.11)
 - 계정 별 역할 구분에 따른 기능 제약 설정 관리
- > Timeline 모니터링 기능
 - 최근 발생한 모든 활동 내역에 대한 관리
- > RSS feeds 알림 기능

-
- 공지 사항 알림 및 관리 기능
 - > Multiple project 관리 기능 지원 (planned for post-1.0 versions)
 - > iCalendar export 기능 지원
 - Mac OS X용 Apple iCal 기능 지원
 - Mozilla Calendar/ cross-platform 지원
 - Evolution, Microsoft Outlook 호환성 지원
 - Google Calendar 호환성 지원
 - > Multiple repository 관리 기능 지원 (since version 0.12)
 - > Interface Localizations 지원 (since version 0.12)
 - > Spam-Filtering 기능 제공
 - > Blog/ Discussion/ Forum 기능 (Plugin 지원)
 - > Guestbook 기능 (plugin 지원)

3. Trac 시스템 요구사항

□ Trac 시스템 요구사항

항목	정보
OS	- Windows, Linux 지원
Python	- 버전 2.5 이상 (최신 안정버전 권장)
setuptools	- 버전 0.6 이상 (최신 안정버전 권장)
Genshi	- 버전 0.6 이상 (최신 안정버전 권장)
DB	- MySQL or MariaDB (최신 안정버전 권장) - MySQLdb (버전 1.2.2 이상) - PostgreSQL (버전 8.0 이상) - Psycopg2 (버전 2.0 이상)

[표 1. Trac 시스템 요구사항]

※ 상기 시스템 요구사항 중 Linux 환경에서 MySQL을 DB로 적용하여 시스템 구성할 경우, Trac이 MyISAM 테이블을 지원하지 않는 트랜잭션 메커니즘을 사용하므로 모든 테이블은 InnoDB or TokuDB타입의 테이블로 생성되어야 한다. 예를 들어, MySQL의 프로젝트 연동 데이터베이스 생성 시, Default Character 타입에 utf8 타입을 설정하고, collate 옵션으로 utf8_bin 타입을 추가 설정하여 이슈를 해결할 수 있다.

※ 관련 상세 정보는 아래의 링크 내역 참조

[1] 기능/특징 세부

- <http://trac.edgewall.org/wiki/PluginList>

[2] 설치/ 유저가이드

- <http://trac.edgewall.org/wiki/TracInstall>

- <http://trac.edgewall.org/wiki/TracGuide>

[3] 공식 Repository/ 다운로드

- <http://trac.edgewall.org/wiki/TracDownload>

III. Stack 통합 테스트

1. 테스트 환경

Trac SW 환경

SW	Version
Ubuntu	12.04 LTS (64비트)
Trac	1.0.8 (최신 안정버전)
Python	2.7.3
setuptools	0.7
Genshi	0.7
DB (MySQL)	Ver 14.14 Distrib 5.5.44, for debian-linux-gnu (x86_64)
Apache	최신 안정버전

[표 2-1. 테스트 SW 환경]

- ※ Ubuntu 환경에서 Trac 구성 시, 필수 의존성(Dependencies) 기능 요소로 Python, setuptools 및 Genshi가 설치되어야 함
- ※ 상기 Trac 환경 구성 시, TracMercurial plugin 기준 이미지 및 설정 적용함

Stack 환경

Stack	OS	네트워크 정보 (IP)
A (Server)	Ubuntu 12.04 LTS	121.162.249.93
A (Client)	Ubuntu 12.04 LTS	121.162.249.94

[표 2-2. Stack 환경]

□ HW 환경

제조사	모델명	CPU	MEM	Disk	NIC
HP	dc7900 CMT	Quad-Core 2.66Ghz~4P	3.6GiB	265GB	Gigabit 1Port

[표 2-3. HW 환경]

※ 동일 사양 HW조건외 PC 2대로 서버, 클라이언트 환경 구성하여 테스트 진행

2. 테스트 접근 방법

Trac은 버전 관리 소프트웨어의 웹 인터페이스를 제공하며, 프로젝트 진행에 따른 각종 개선점과 개발 과정에서의 발생 버그 관련한 이슈 트래킹, 그리고 Wiki를 통한 문서 관리 및 각 리소스 간 연동을 주 기능으로 한다.

본 테스트는 서버, 클라이언트 환경을 구성하여 Trac의 설치 및 프로젝트 운용에 따른 사용자 시나리오를 토대로 Trac 주요 기능 요소의 기능 신뢰성을 확보하며, 환경 구성의 안정성과 Web UI 환경의 호환성을 확인 하는 데 주된 테스트 방향성과 목적을 두고 수행되었다.

□ 탐색적 테스트(Exploratory Testing)

탐색적 테스트는 테스트 엔지니어의 지적 능력을 최대한 공유, 활용하는 것을 목적으로 하는 테스트 접근법으로 테스트를 수행할 대상을 실행시켜 사용함과 동시에 사용 측면에서 문제가 되는 부분에 집중하여 테스트를 설계 및 계획한다.

이러한 과정은 효율적 진행을 위한 Time Boxing을 통해 수행되므로 테스트 케이스 작성을 최소화할 수 있고, 상대적으로 적은 시간에 집중적인 테스트를 가능하게 한다.

□ 리스크 기반 테스트(Risk based Testing)

리스크 기반 테스트 기법은 테스트 대상에 비해서 테스트 자원이 부족한 경우 효과적이고, 효율적인 테스트 수행을 위해 적용 될 수 있다. 해당 기법은 크게 리스크 식별과 리스크 분석, 그리고 리스크 계획의 세 단계로 구분 진행된다.

리스크 식별 단계에서는 제품 품질관점에서 테스트 대상이 될 항목을 식별하고, 프로젝트나 제품에 대한 리스크 요소를 식별한다.

리스크 분석 단계에서는 장애 발생가능성과 장애로 인한 영향을 식별하고 리스크 우선순위를 결정한다.

마지막으로 리스크 계획 단계에서는 리스크의 우선순위에 따른 대처 방안 및 완화 정책을 수립하며, 이후 테스트 수행 시 커버리지를 고려하여 선택과 집중을 통해 테스트를 수행하게 된다.

□ 시나리오 테스트

시나리오 테스트 기법은 단일 기능에 대한 결함 여부를 확인하는 것이 아니라, 서로 다른 컴포넌트 사이의 상호작용과 간섭으로 발생할 수 있는 결함을 발견하기 위한 기법이다.

3. 테스트 수행

테스트 수행 관련 세부 내역은 별첨 「Trac 테스트케이스」 문서를 참고한다.

□ 탐색적 테스트 현황

이하, 테스트 현황의 차터 항목 구분은 각 기능 아이템 항목 별 세부 구분 항목으로서 사용자 시나리오를 기반으로 정의되었다.

기능 아이템	기본 차터	테스트 아이디어
Trac 기본환경 관리	1	12
Trac 업무 단위 관리	2	20
Trac 운용 관리	2	10
합 계	5	42

[표 3. 테스트 아이디어 현황]

- 기본환경 관리 카테고리에서는 대상 SW의 설치 및 기본 환경구성 측면에서 안정성 확인과 계정 생성에 따른 Web 기반 운용환경 별 접속 기능 확인을 중심으로 테스트 진행
- 업무 단위 관리 카테고리에서는 개발 프로젝트 진행에 따른 주요 프로세스를 기준으로 프로젝트의 업무 단위 관리 기능과 테스트 관련 이슈 관리 기능을 중심으로 테스트 진행
- 운용 관리 카테고리에서는 시스템의 Web 서버 연동에 따라 제공되는 Web UI 상에서 주요 기능 카테고리 별 기능 항목의 동작 정상성 확인 및 Web 환경 호환성 확인을 중심으로 테스트 진행

4. 테스트 수행 결과

Trac 테스트를 수행한 결과 기능 카테고리 별 정리된 내용은 아래와 같다.

분류		PASS	FAIL	N/T	N/A
기능	테스트 아이디어				
Trac 기본환경 관리	12	12	0	0	0
Trac 구성 관리	20	20	0	0	0
Trac 운영 관리	10	10	0	0	0

[표 4. 테스트 결과]

- 상기 테스트는 프로젝트 운용 형태에 따라 Standalone 서버를 적용한 시스템 카테고리과 Web 서버를 적용한 시스템 카테고리를 구분하고, 사용자 시나리오를 반영하여 Trac 시스템의 기능 정상성 확보를 중점으로 수행함
- Trac 시스템 구성으로 서버, 클라이언트 환경을 구축하고 주요 관리 환경인 Web 브라우저 UI 상에서 각 기능 카테고리에 따라 관리 시스템의 구동 관련 주요 기능 동작들을 확인한 결과 환경 구성 및 기능 관련한 심각한 오류 및 치명적 장애 이슈는 발견되지 않음

IV. 종합

- Trac 테스트 수행 결과, 공개 SW로 구성된 Stack 환경에서 치명적 결함, 이슈 발생 없이 설치 및 설정 관리됨을 확인하였으며, 개발 프로젝트의 관리 절차 관련 제반 기능들이 각 기능 단위 resource들과 유기적으로 동작함을 확인하였다.

- Trac은 티켓(Ticket) 단위의 정보 생성 관리 기능을 제공하며, 각 Ticket 정보는 Subversion과 같은 버전 관리 시스템과 연동되어 체계적인 업무 프로세스 관리를 가능케 한다.

- Trac은 Python을 근간으로 개발된 오픈소스 이슈관리 도구로 Plugin 설정을 통해 다양한 기능 및 확장성을 제공하며, SVN 등의 형상관리 도구와 접목이 가능하여 소프트웨어 개발 프로젝트 관리 환경에 유용한 솔루션으로 적용될 수 있다.

※ 참고 자료

[1] Trac 공식 사이트

- <http://trac.edgewall.org/>

[2] 설치 가이드/ 유저가이드

- <http://trac.edgewall.org/wiki/TracInstall>
- <http://trac.edgewall.org/wiki/TracGuide>
- <http://trac.edgewall.org/wiki/TracPlugins#InstallingaTracPlugin>

[3] 배포/ 개발 관련

- <http://trac.edgewall.org/wiki/TracDownload>
- <http://trac.edgewall.org/roadmap>
- <http://trac.edgewall.org/browser>
- <http://trac.edgewall.org/wiki/TracDev>

[4] 포럼/ 커뮤니티

- <http://trac.edgewall.org/wiki/MailingList>
- <http://trac.edgewall.org/wiki/TracUsers>