

El Estándar IEEE 730

Guía para la planificación y ejecución de un plan de aseguramiento de la calidad del software (SQA)

Autores: GRUPO N

June 19, 2024

- Casquino Ticona Jorge Romel
- Condori García Paul Wenceslao
- García Jiménez Angel Raul
- Phocco Soncco Yeni Rosmirian

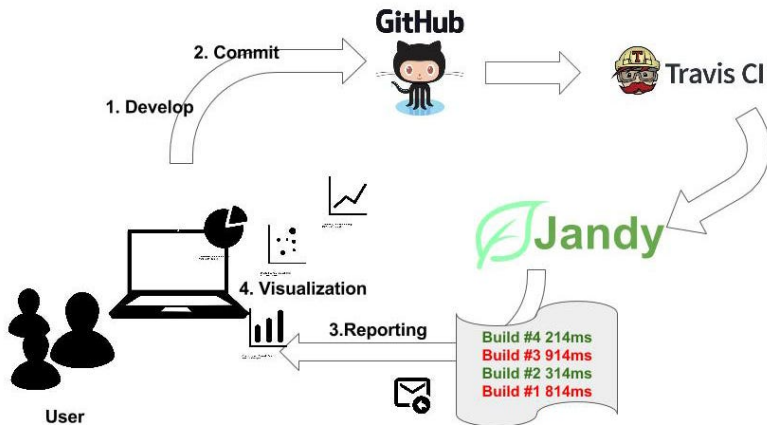
proporciona directrices para la planificación y ejecución de un plan de aseguramiento de la calidad del software (SQA).

- Planificación del SQA: Define los objetivos de calidad, responsabilidades, y recursos necesarios para asegurar la calidad del software.
- Revisión y Auditoría: Realización de revisiones y auditorías independientes para verificar que los procesos y productos cumplen con los estándares establecidos.
- Pruebas y Validación: Implementación de pruebas de software para asegurar que el producto final cumple con los requisitos y especificaciones.
- Informes de Calidad: Documentación y reporte del estado de calidad del software durante todo el ciclo de vida del desarrollo

Cómo Travis CI y GitHub Ayudan al Cumplimiento del IEEE 730

- Travis CI permite configurar pipelines de CI que ejecutan automáticamente pruebas unitarias, de integración y funcionales cada vez que se realiza un commit o se crea una pull request en GitHub.
- Cumplimiento IEEE 730: Automatizar las pruebas garantiza que los cambios en el código son evaluados consistentemente, reduciendo la probabilidad de introducir defectos y mejorando la fiabilidad del software.

Cómo Travis CI y GitHub Ayudan al Cumplimiento del IEEE 730



- Activar Travis CI en el Repositorio: En la interfaz de Travis CI, activa Travis CI para el repositorio que creaste en GitHub.
- Configurar el Archivo `.travis.yml`: Crea un archivo llamado `.travis.yml` en la raíz de tu repositorio. Este archivo define cómo Travis CI debe construir y probar tu proyecto.

Ejemplo de .travis.yml para un Proyecto de Python

```
language: python
python:
  - "3.8"
install:
  - pip install -r requirements.txt
  - pip install flake8
  - pip install pytest
  - pip install coverage
  - pip install pylint
  - pip install mypy
  - pip install safety
script:
  - flake8 . # Análisis de estilo de código
  - pylint my_module/ # Análisis de calidad de código
  - mypy my_module/ # Comprobación de tipos
  - coverage run -m pytest # Ejecución de pruebas unitarias con cobertura
  - coverage report # Informe de cobertura
  - pytest integration_tests/ # Ejecución de pruebas de integración
  - safety check # Análisis de dependencias para vulnerabilidades
notifications:
  email:
    recipients:
      - tu_email@example.com
    on_success: never
    on_failure: always
```

- **Compilación del Proyecto:** Travis CI automáticamente clona el repositorio, instala las dependencias necesarias y compila el proyecto según las instrucciones definidas en el archivo `.travis.yml`.
- **Ejecución de Pruebas:** Travis CI ejecuta las pruebas automatizadas que has definido. Esto incluye pruebas unitarias, de integración y funcionales. Los resultados de las pruebas se registran y se muestran en la interfaz de Travis CI.

Automatizar el Proceso de Evaluación

- Generación de Informes de Pruebas: Travis CI genera informes sobre el estado de las pruebas, indicando si han pasado o fallado. Estos informes son esenciales para el aseguramiento de la calidad según IEEE 730.
- Ejecuta pruebas automatizadas y genera informes.

Automatizar el Proceso de Evaluación

The screenshot displays the Travis CI web interface for the repository `mklabs/nabe`. The interface includes a navigation bar with links to Home, Stats, Blog, and Docs, and a sign-in button for GitHub. A sidebar on the left shows recent repositories: `mklabs/nabe` (build #7, finished about 24 hours ago) and `mklabs/cakes` (build #1, finished 4 days ago). The main content area shows the build history for `mklabs/nabe`, with tabs for 'Current' and 'Build History'. The build history table lists several builds with their status, commit hash, message, duration, and completion time. The right sidebar shows the 'Workers' section with a list of worker nodes and their status, and a 'Queue: Builds' section indicating no jobs are currently in the queue. A yellow warning box states 'This stuff is alpha.' and a 'Join us and help!' section provides information about the Travis CI project.

Travis CI interface showing the build history for the repository `mklabs/nabe`.

The interface includes a sidebar with recent repositories and a main section displaying the build history table.

Recent Repositories:

- `mklabs/nabe` (Build #7): Duration: -, Finished: about 24 hours ago
- `mklabs/cakes` (Build #1): Duration: -, Finished: 4 days ago

Build History Table:

Build	Commit	Message	Duration	Finished
7	<code>95f8bfe (0.1.0)</code>	fix json response with buffers instead of post content	-	about 23 hours ago
6	<code>4a82abc (0.1.0)</code>	archives: ensure we check the date parsed from model.all instead of grepping in case of archives #12	-	about 23 hours ago
5	<code>895b0bf (0.1.0)</code>	switch to git-fs 0.0.7, fixing utf8 rendering. #11	-	-
4	<code>7bed277 (master)</code>	npm install should be enough and prevent install submodule devDependencies	-	4 days ago
3	<code>596651f (master)</code>	Merge branch 'coffee' into master	-	4 days ago

Workers:

- erlang.tty.nl
- rails1.worker.travis-ci.org
- rails2.worker.travis-ci.org
- ruby1.worker.travis-ci.org
- ruby2.worker.travis-ci.org
- ruby3.worker.travis-ci.org

Queue: Builds

No jobs

Queue: Rails

No jobs

This stuff is alpha.

Please do not consider this a stable service. We're still far from that! More info [here](#).

Join us and help!

Travis is an open, distributed build system for the open source community. Find us at:

Evaluar Resultados y Realizar Revisión de Código en GitHub

- Revisión de Resultados de Pruebas en Travis CI: Revisa los resultados de las pruebas en la interfaz de Travis CI. Si alguna prueba falla, se puede ver un registro detallado del error para diagnosticar y solucionar el problema.
- Revisión de Código en GitHub: Utiliza pull requests (PR) para realizar revisiones de código colaborativas. En cada PR, Travis CI ejecuta las pruebas automatizadas y muestra el estado de la construcción (build status) directamente en GitHub. Los revisores de código pueden ver los resultados de Travis CI y aprobar o solicitar cambios basados en estos resultados.

Evaluar Resultados y Realizar Revisión de Código en GitHub

The screenshot displays the GitHub homepage with a dark theme. On the left, the 'Top Repositories' section features a search bar and a list of repositories, including 'kyo3773pw/datos', 'kyo3773pw/software-engineering', 'kyo3773pw/sd_repo', 'frdtorres/Teaching2024', 'kyo3773pw/proyecto-chatbot', 'kyo3773pw/Teaching2024', and 'kyo3773pw/kyo3773pw.github.io'. A green 'New' button is visible above the search bar. The main content area is divided into two columns. The left column shows 'Trending repositories' with 'fudan-generative-vision/hallo' (Python, 3.1k stars) and 'anthdm/superkit' (Go, 336 stars). The right column shows 'Recommended for you' with 'fajarghifar/happybirthday'. On the far right, there are two announcement boxes: one for 'The GitHub Enterprise Server 3.13 is generally available' and another for 'Explore repositories' featuring 'Kampus-Merdeka-Software-Engineering / km-feb24-balikpapan-17'.

Despliegue Continuo (Opcional)

- Configura despliegue automático si todas las pruebas pasan.
- Añade pasos de despliegue en el archivo `.travis.yml`.