HERRAMIENTAS PARA PRUEBAS

Ecosistema

- Alrededor de las pruebas y el aseguramiento de la calidad existe todo un ecosistema de herramientas que se utilizan en una o más actividades de soporte de prueba, entre las que se encuentran:
  - Las herramientas que se utilizan directamente en las pruebas, como las herramientas de ejecución de pruebas, las herramientas de generación de datos de prueba y las herramientas de comparación de resultados.
  - Las herramientas que ayudan a gestionar el proceso de pruebas, como las que sirven para gestionar pruebas, resultados de pruebas, datos requisitos, incidencias, defectos …, así como para elaborar informes y monitorizar la ejecución de pruebas.
  - Las herramientas que se utilizan en la fase de reconocimiento, como las herramientas de estrés y las herramientas que monitorizan y supervisan la actividad del sistema.
  - Cualquier otra herramienta que contribuya al proceso de pruebas sin ser especificas del mismo, como las hojas de cálculo, procesadores de texto, diagramadores, …
- Las herramientas pueden clasificarse en base a distintos criterios, tales como el objetivo, comercial/código abierto, especificas/integradas, tecnología utilizada …

Objetivos

- Mejorar la eficiencia de las tareas de pruebas automatizando tareas repetitivas o dando soporte a las actividades de pruebas manuales, como la planificación, el diseño, la elaboración de informes y la monitorización de pruebas.
- Automatizar aquellas actividades que requieren muchos recursos si se hacen de forma manuales (como por ejemplo, las pruebas estáticas, pruebas de GUI).
- Automatizar aquellas actividades que no pueden ejecutarse de forma manual (como por ejemplo , pruebas de rendimiento a gran escala de aplicaciones cliente-servidor).
- Aumentar la fiabilidad de las pruebas (por ejemplo, automatizando las comparaciones de grandes ficheros de datos y simulando comportamientos).

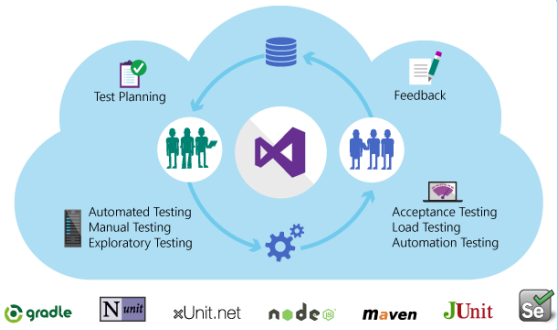
Ventajas

- Reducción del trabajo repetitivo (por ejemplo, la ejecución de pruebas de regresión, la reintroducción de los mismos datos de prueba y la comprobación contra estándares de codificación).
- Mayor consistencia y respetabilidad (por ejemplo las pruebas ejecutadas por una herramienta en el mismo orden y con la misma frecuencia, y pruebas derivadas de los requisitos).
- Evaluación de los objetivos (por ejemplo, medidas estáticas, cobertura).
- Facilidad de acceso a la información sobre las pruebas (por ejemplo, estadísticas y gráficos sobre el avance de las pruebas, la frecuencia de incidencias y el rendimiento).

Desventajas

- Expectativas poco realistas de la herramienta (incluyendo funcionalidad y facilidad de uso).
- Exceso de confianza en la herramienta (sustitución por el diseño de pruebas o uso de pruebas automatizadas cuando seria mejor llevar a cabo pruebas manuales).
- Subestimar:
  - La cantidad de tiempo, coste y esfuerzo necesario para la introducción inicial de una herramienta (incluyendo formación y experiencia externa).
  - El tiempo y el esfuerzo necesarios para conseguir ventajas significativas y constantes de la herramienta (incluyendo la necesidad de cambios en el proceso de pruebas y la mejora continua de la forma en la que se utiliza la herramienta).
  - El esfuerzo necesario para mantener los activos de prueba generados por la herramienta.
- Desprecio del control de versión de los activos de prueba en la herramienta.
- Desprecio de problemas de relaciones e interoperabilidad entre herramientas críticas tales como las herramientas de gestión de requisitos, herramientas de control de versiones, herramientas de gestión de incidencias, herramientas de seguimiento de defectos y herramientas procedentes de varios fabricantes.
- Coste de las herramientas comerciales o ausencia de garantías en las herramientas open source.
- Riesgo de que el fabricante de la herramienta cierre, retire la herramienta o venda la herramienta a otro proveedor.
- Mala respuesta del fabricante para soporte, actualizaciones y corrección de defectos.
- Imprevistos, tales como la incapacidad de soportar una nueva plataforma.

Herramientas de prueba



Test Control	Test Design	Test Execution	Test Environment
Quality reporting Micro Focus ALM Octane, Tricentis qTest manager	Test Design Enterprise Architect Test management, Broadcom test Design Automation, various Open Source Frameworks	Static code analysis SonarQube	Test data management DATPROF Privacy, Subset, IBM Optim, EPI-USE Data Sync Manager, DSM for HCM, Data Secure, BW Test Data, Broadcom Test Data Manager
Testware management SmartBear QAComplete, Tricentis qTest manager		Secure code scan SonarQube, Micro Focus Fortify	Test environment management Azure DevOps, AWS CloudFormation, Micro Focus Deployment Automation
Progress reporting Atlassian JIRA, Zephyr, Micro Focus ALM Octane		Unit test xUnit, Mockito	Service virtualization Parasoft Virtualize, Broadcom Service Virtualization, Smartbear ServiceV
Task management Atlassian JIRA, various Open Source Solutions, Micro Focus ALM Octane		Functional test Micro Focus UFT, Tricentis TOSCA, SmartBear TestComplete/ReadyAPI, Robot Framework, Selenium, Cucumber, Fitnesse	
Anomaly management ServiceNow, Mantis, BugZilla, Atlassian JIRA, Micro Focus ALM Octane		Security test ZAP, Metasploit, NMAP, Micro Focus Fortify Static	
Code coverage SonarQube, Intellij, various Open Source Solutions		Performance test JMeter, NeoTys NeoLoad, Micro Focus LoadRunner	

INTRODUCCIÓN A HP ALM

Introducción

- HP Application Lifecycle Management (ALM) capacita a organizaciones para la gestión del ciclo de vida de aplicaciones básicas, desde los requisitos a la implementación, otorgando a los equipos de la aplicación la visibilidad y la colaboración fundamentales para una entrega predecible, repetible y adaptable de las aplicaciones modernas. HP ALM proporciona una plataforma centralizada para gestionar y automatizar las actividades necesarias para el ciclo de vida principal de la aplicación y para ayudar a impulsar el proceso completo.
- Gestionar el ciclo de vida de las aplicaciones desde su inicio hasta su retirada es un proceso complejo. Tanto si la organización es predominantemente Agile como si se emplean métodos iterativos y secuenciales, el objeto de una gestión de ciclos de vida efectiva es lograr una mayor previsibilidad, una creciente repetibilidad, una calidad mejorada y un pronto ajuste de cambios. Conocer los hitos de un proyecto, las entregas, los recursos y los requisitos presupuestarios y hacer seguimiento del estado del proyecto, los estándares e indicadores de calidad permite a los gestores de entregas alcanzar estos objetivos.
- HP ALM simplifica y organiza la gestión de aplicaciones brindando al usuario el control sistemático del proceso. Ello contribuye a crear un marco de trabajo y bases para el flujo de trabajo de gestión de ciclos de vida de aplicaciones en un repositorio central.

Características

- Rastreo de versiones
  - ALM incorpora un sistema de organización y rastreo de versiones de la aplicación que permite a los gestores alinear las prioridades empresariales y expectativas de calidad con los requisitos, pruebas y defectos del proyecto. ALM permite tomar decisiones sobre versiones mejor fundadas con indicadores de rendimiento clave en tiempo real (KPIs).
- Requisitos y pruebas
  - ALM permite definir y mantener un repositorio de requisitos y pruebas. Los requisitos ayudan a garantizar que las necesidades de pruebas y empresariales estén cubiertas. Las pruebas pueden ser generadas automáticamente a partir de estos requisitos garantizando así que se sometan a pruebas los aspectos apropiados de la aplicación. Para lograr los diversos objetivos de un proyecto, podrás organizar las pruebas del proyecto en grupos exclusivos. ALM proporciona un método para programar y ejecutar las pruebas, recopilar resultados de pruebas y analizar los datos.
- Seguimiento de defectos
  - El análisis de defectos y sus tendencias permite tomar decisiones efectivas del tipo "continuar/finalizar". ALM incluye un sistema de seguimiento de defectos que permite monitorizar los defectos desde su detección inicial hasta la resolución de los mismos. Permite también compartir defectos entre proyectos, lo cual reduce el riesgo, pues permite que los desarrolladores encuentren, prioricen y resuelvan los defectos en una etapa temprana. Un repositorio de defectos centralizado permite asimismo informar del estado de defectos agregados y sobre las tendencias entre proyectos.

Características

- Herramientas de análisis
  - La capacidad de rastrear el progreso durante todo el proceso del ciclo de vida de la aplicación es esencial para una óptima previsibilidad. ALM ofrece herramientas para analizar cada una de las fases del proceso, incluida la instrumentación específica para proyectos Agile (como por ejemplo, gráficos de producto y gráficos de avance).
- Bibliotecas de activos
  - ALM permite compartir y reutilizar las bibliotecas de activos entre proyectos. Las bibliotecas compartibles permiten gestionar iniciativas con varias aplicaciones para asegurar que los cambios que se realizan a una aplicación no afecten negativamente a otra. También contribuye a lograr una mayor consistencia y repetibilidad al permitir la reutilización de activos. Pueden aplicarse cambios específicos a los activos compartidos de cada proyecto sin que ello altere la integridad de la biblioteca.
- Integraciones de ALM
  - ALM ofrece integración con herramientas de HP (como Unified Functional Testing y LoadRunner), así como herramientas de pruebas personalizadas y de terceros, y herramientas de gestión de configuración y requisitos. ALM se comunica con la herramienta de pruebas que se desee, ofreciéndo una solución integral para la realización de pruebas de aplicaciones completamente automatizadas.
- Performance Center
  - ALM incorpora funcionalidad que permite al usuario gestionar todos los aspectos de proyectos de pruebas de rendimiento a gran escala, incluida la programación y asignación de recursos a partir de una ubicación centralizada accesible en la web. ALM ayuda a agilizar el proceso de pruebas, reducir costes de recursos y aumentar la eficiencia operativa.
- Lab Management
  - ALM incluye funcionalidad para gestionar los recursos que permiten ejecutar pruebas en hosts remotos. ALM ayuda a automatizar la implementación y el proceso de pruebas, aumentando la fiabilidad y facilidad de uso.

Ediciones

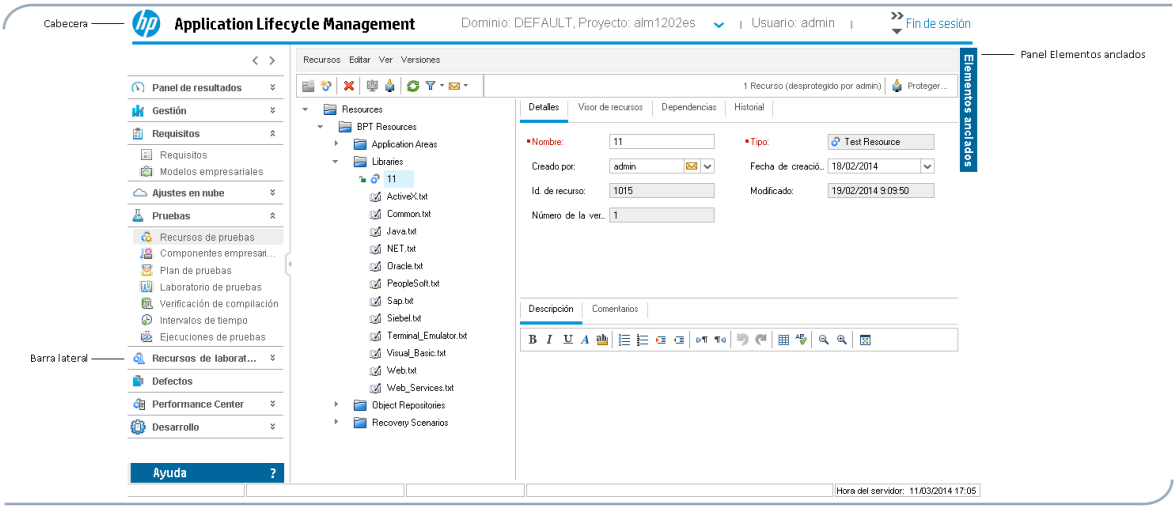
- **HP ALM**: Una plataforma unificada para la gestión y automatización de procesos, actividades y activos para la creación, prueba, entrega y mantenimiento de aplicaciones. Incluye módulos para la gestión de requisitos, pruebas, defectos y de desarrollo, y plantificación general de versiones y proyectos.
- **HP ALM Essentials Edition**: Proporciona un subconjunto de funcionalidades del producto HP ALM y está pensado para ayudar a los equipos más pequeños a ponerse en marcha rápidamente.
- **HP ALM Performance Center Edition**: Funcionalidad para la completa gestión, programación, ejecución y monitorización de secuencias de comandos de prueba de rendimiento. Reside en la misma plataforma que HP ALM y se integra directamente con HP ALM y HP LoadRunner.
- **HP Quality Center Enterprise Edition**: Reside en la misma plataforma unificada que HP ALM. Ofrece una funcionalidad básica para la gestión de calidad. Admite la creación de un centro de calidad de excelencia a través de estrechas integraciones con HP Unified Functional Testing, HP Business Process Testing y HP Sprinter.
- **HP Quality Center Express Edition**: Proporciona un subconjunto de funcionalidades del producto HP ALM y está diseñado para introducir nuevos clientes a HP ALM. Ofrece funcionalidad básica para la gestión de pruebas y de defectos.
- **HP Quality Center Community Edition**: Proporciona un subconjunto de funcionalidades del producto HP ALM y para introducir nuevos clientes a HP ALM, con funcionalidad básica para la gestión de pruebas y de defectos.

***Aclaración**: HP ALM empezó siendo TestDrirector de Mercury hasta la v8.0 que se renombra a Mercury Quality Center, hasta la v9.0 donde HP adquiere a Mercury y paso a denominarse HP Quality Center hasta la v10.0, donde ya toma el nombre de HP ALM. Con el cambio de propietarios, HP ALM/Quality Center se ha ido renombrando a Micro Focus ALM/Quality Center y OpenText ALM/Quality Center.*

Interfaz de usuario

- Para iniciar ALM:
  - Abrir el navegador y escribir la URL de ALM.
    - http://<servidor>[:puerto]/qcbin
  - O, solo en Windows, se puede hacer clic en el vínculo de Cliente de escritorio de ALM.
- Salvo que ALM se haya configurado para la autenticación externa, se abrirá la ventana Inicio de sesión de ALM y requerirá la autenticación.
- En la Cabecera se muestra el dominio actual, proyecto y nombre de usuario y proporciona los botones para acceder a otros proyectos, herramientas comunes y documentación.
- La Barra lateral contiene botones para desplazarse entre las vistas de ALM, cambiando entre los módulos de ALM y accediendo a varios recursos en línea.
- El Panel Elementos anclados abre un panel para mostrar los elementos de ALM que están anclados. Se puede hacer clic en un requisito, prueba o defecto anclados para acceder a ellos rápidamente.
- La zona central o zona de trabajo es contextual en función del módulo seleccionado y permitirá consultar y mantener los diferentes artefactos del ALM.

Interfaz de usuario



Información general sobre el interfaz

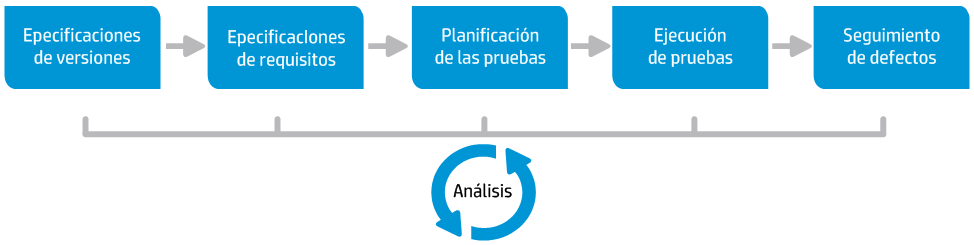
- HP ALM organiza y muestra datos en cuadrículas y árboles. Podrás manipular los datos de diversas maneras, como por ejemplo reorganizando las columnas, filtrándolos, ordenándolos y agrupándolos. Podrás también adjuntar archivos a registros, buscar texto específico en los registros y ver el historial de varias entidades.
- Cuando cambia un requisito, prueba o defecto, HP ALM puede enviar alertas a las entidades asociadas y notificar a los responsables de las entidades asociadas. El administrador del proyecto podrá activar las reglas de alerta en función de las asociaciones que el usuario establezca entre requisitos, pruebas y defectos. Podrás agregar una marca de seguimiento a un requisito, prueba, instancia de prueba o defecto específico para recordarte a sí mismo hacer seguimiento de un tema en particular.
- El usuario puede determinar el aspecto de las ventanas de HP ALM seleccionando ciertas configuraciones y guardarla como una vista favorita que se podrá volver a cargar y usar en el futuro. El usuario decide si se permite a otros el acceso a las vistas de favoritos guardándolas en una carpeta pública o una privada.
- También puede anclar un requisito, un plan de pruebas o un defecto, lo que le permita saltar rápidamente a ese elemento, con independencia del módulo en el que se encuentre.

Control de versiones de entidades

- En proyectos con control de versiones habilitado, podrás crear y gestionar entidades de HP ALM y al mismo tiempo mantener las versiones anteriores de tales entidades. Ello incluye requisitos, pruebas, recursos de pruebas, modelos de procesos empresariales y componentes empresariales. El control de versiones solo se aplica a entidades individuales.
- Para realizar cambios a una entidad en un proyecto con control de versiones habilitado, se deberá primero desproteger la entidad. Al desproteger una entidad, ALM bloquea la entidad con objeto de prevenir que otros usuarios sobrescriban los cambios efectuados. La versión desprotegida de la entidad no se encuentra visible para otros usuarios.
- Cuando se termine de efectuar cambios, se deberá volver a proteger la entidad. La nueva versión de la entidad se encontrará entonces disponible para otros usuarios.

Proyectos

- Un proyecto de ALM permite almacenar y gestionar los datos generados y recopilados por ALM. El proyecto es la base de la gestión del ciclo de vida. Cada proyecto está vinculado a una base de datos que almacena información sobre el proyecto.
- ALM permite gestionar el acceso de usuarios a un proyecto. El grupo de usuarios determina los privilegios que el usuario tiene en un proyecto determinado. Al personalizar el proyecto se podrá indicar a ALM que conserve un registro de valores del sistema y campos de usuario del proyecto.
- El mapa de ruta de Application Lifecycle Management incluye las siguientes fases que se gestionaran en el proyecto:



Mapa de ruta

- **Especificaciones de versiones**: Desarrollar un plan de gestión de ciclos de versiones ayuda a una gestión más eficiente de ciclos y versiones de las aplicaciones. Podrás rastrear el progreso de una versión de aplicación con su plan para determinar si la versión es la correcta.
- **Especificaciones de requisitos**: Definir requisitos para satisfacer las necesidades empresariales y de pruebas. Podrás gestionar los requisitos y realizar seguimiento multidimensional entre requisitos, pruebas y defectos, por múltiples versiones y ciclos. ALM ofrece visibilidad en tiempo real de la cobertura de requisitos y defectos asociados para evaluar la calidad y el riesgo empresarial.
- **Planificación de las pruebas**: En función de los requisitos del proyecto, podrás crear planes de pruebas y diseñar pruebas. ALM ofrece un repositorio para pruebas tanto manuales como automatizadas.
- **Ejecución de pruebas**: Crear subconjuntos de las pruebas de su proyecto con objeto de alcanzar objetivos de pruebas específicos. ALM admite las pruebas avanzadas, de regresión, funcionales y de sanidad. Permite ejecutar pruebas programadas para diagnosticar y resolver problemas.
- **Seguimiento de defectos**: Enviar defectos y hacer seguimiento del progreso de reparación. Analizar defectos y tendencias de defectos permite tomar decisiones efectivas del tipo "continuar/finalizar". ALM presta soporte para el ciclo de vida de defectos en su totalidad (desde la detección inicial de problemas, hasta la corrección de los defectos y la verificación de soluciones).

Versiones y ciclos

- La publicación de versiones de aplicaciones puede resultar desafiante en ocasiones. Requiere consensuar las prioridades empresariales y expectativas de calidad con los requisitos, pruebas y defectos de los distintos proyectos. La mayoría de las aplicaciones se someten a un proceso de pruebas en diferentes plataformas de hardware, diversas configuraciones (ordenadores, sistemas operativos y exploradores) y varias versiones de aplicación. La gestión de todos los aspectos de una versión de la aplicación puede llevar tiempo y resultar difícil. El proceso de gestión de aplicaciones comienza en la definición de versiones.
- HP ALM permite organizar y hacer seguimiento de versiones futuras mediante la definición de versiones y ciclos.
- Una **versión** representa un grupo de cambios que tienen lugar en una o más aplicaciones cuya distribución se produce simultáneamente. Cada versión puede contener diversos ciclos.
- Cada **ciclo** lo conforma un conjunto de esfuerzos de desarrollo y control de calidad que persiguen un objetivo común en función de la línea base de la versión. Tanto las versiones como los ciclos poseen fechas de inicio y fin claramente definidas.

Versiones y ciclos

- Una vez definidos las versiones y ciclos, se definen y examinan los requisitos y se asignan a versiones y ciclos. Los **requisitos** describen en detalle las necesidades de la aplicación y sirven de base para crear el **plan de pruebas**. Las pruebas que se elaboran durante la fase del plan de pruebas deben tener por objeto satisfacer tales requisitos.
- Después de asignar los requisitos a las versiones y ciclos, se crean las carpetas del conjunto de pruebas y se asignan a ciclos. Un **conjunto de pruebas** es un grupo de instancias de prueba de un proyecto de ALM diseñado con el propósito de lograr ciertos objetivos de pruebas. Una vez asignadas las carpetas del conjunto de pruebas a un ciclo, se ejecutan los conjuntos de pruebas de dichas carpetas.
- Si se detecta algún error durante la ejecución de un conjunto de pruebas, podrá enviar el **defecto**. ALM creará automáticamente un vínculo entre el conjunto de pruebas, el ciclo y revisión asociados y el nuevo defecto.

Versiones y ciclos

- Los **defectos** pueden ser enviados a un proyecto de ALM desde cualquier módulo y en cualquier etapa del proceso de gestión de aplicaciones. Mientras se analiza y decide cuáles serán los nuevos defectos que se repararán, los defectos pueden ser asignados a la versión y ciclo apropiados.
- Con posterioridad a las ejecuciones de pruebas, podrás analizar el progreso de las pruebas con objeto de determinar si cumplen los objetivos de la versión correspondiente. También podrá determinar el número de defectos que se resolverán y la cantidad que continuará pendiente. Los resultados podrán ser analizados a nivel de versión o de ciclo.
- Se podrá hacer seguimiento del progreso del proceso de gestión de aplicaciones en tiempo real analizando el árbol de versiones y asegurándose de que se ajuste a los objetivos de versión.
- Para acceder a las versiones y ciclos para administrarlas, desde la barra lateral de ALM, en Gestión, seleccionar Versiones.

Planificación y seguimiento del proyecto

- Planificación y seguimiento del proyecto (PPT) permite a los gestores de QA hacer seguimiento del estado de disponibilidad de la aplicación definiendo objetivos respecto a las actividades de una versión de aplicación.
- Para cada versión, se definen los elementos de ámbito de la versión, que hace referencia a una sección subdividida de una versión, por ejemplo, una función nueva, un cambio a una función existente o un tema nuevo. Para cada elemento de ámbito, se definen los requisitos, las pruebas, los conjuntos de pruebas y los defectos relacionados. Para medir el progreso de los elementos de ámbito de la versión se asocian a hitos.
- Un hito es un punto en la línea temporal de una versión que representa la finalización de una entrega. Permite realizar un seguimiento y validar el progreso de la versión. Un hito se puede asociar a uno o a varios elementos de ámbito de la versión.
- PPT recopila y analiza los datos de los hitos definidos mediante los indicadores de rendimiento clave (KPI). Un KPI es una medida cuantificable diseñada para realizar un seguimiento de una variable de rendimiento importante a lo largo del tiempo y medir el resultado esencial de las actividades de garantía de calidad. Para cada KPI, se definen los niveles del umbral para establecer los límites de advertencia.
- PPT usa los KPI para analizar los datos de disponibilidad del hito y para mostrar la disponibilidad general del estado e implementación de una versión en forma de tarjeta de resultados. La tarjeta de resultados supervisa y hace un seguimiento de cómo se va cumpliendo cada objetivo diariamente. Para analizar más en profundidad los resultados, se puede generar informes y gráficos del panel de resultados.
- En Gestión → Versiones, para una versión se pueden agregar los hitos, asociar los elementos de ámbito de la versión y los KPI, y en la ficha Tarjeta de resultados consultar los resultados.

Requisitos

- El módulo Requisitos permite definir, gestionar y hacer seguimiento de los requisitos en todas las etapas de la gestión del ciclo de vida de la aplicación.
- El módulo Requisitos permite:
  - **Requisitos previos**: Permite determinar el ámbito de los requisitos compilando información como por ejemplo especificaciones técnicas y funcionales, documentos de requisitos empresariales y de marketing, y objetivos de las partes interesadas.
  - **Importación de modelos de procesos empresariales**: Si trabajas con modelos de procesos empresariales, podrás crear un marco de trabajo de requisitos importando modelos creados con herramientas de modelado estándar. El módulo Modelos empresariales permite analizar la calidad de los modelos de procesos empresariales y los flujos empresariales.

Requisitos

- **Creación y mantenimieto de requisitos**: Permite definir un marco de trabajo jerárquico respecto al ámbito de sus requisitos creando un árbol de requisitos y definir grupos de requisitos en el árbol de requisitos. Podrás crear una lista de requisitos detallados en el árbol de requisitos para cada grupo de requisitos y asignar un nivel de prioridad al requisito, el cual podrá ser tomado en consideración al crear el plan de pruebas.
- **Seguimiento de requisitos**: Podrás agregar seguimiento entre los requisitos. Al analizar el impacto de un cambio propuesto en un requisito específico, el seguimiento muestra los otros requisitos que pueden verse afectados por el cambio. Para verificar el grado de integridad de las relaciones entre sus requisitos podrás generar una matriz de seguimiento.
- **Cálculo de riesgos**: Utiliza la gestión de calidad basada en riesgos para calcular el nivel al que hacer pruebas a cada requisito en función de la naturaleza del mismo y de los recursos disponibles.
- **Creación de cobertura**: Crea la cobertura entre los requisitos y las pruebas para asegurarte de que todos los requisitos sean implementados en el proyecto. Podrás asimismo crear cobertura convirtiendo los requisitos en pruebas en el árbol del plan de pruebas. Se creará automáticamente la cobertura entre los requisitos y sus pruebas correspondientes.

Requisitos

- **Vinculación a defectos**: Podrás vincular un requisito con determinados defectos. Ello resulta de utilidad, por ejemplo, al crear cobertura entre requisitos y pruebas. Vincular defectos puede contribuir a garantizar la conformidad con los requisitos y necesidades de pruebas. Si un requisito cambia, podrás identificar de inmediato las pruebas y defectos que se han visto afectados y quién es la persona responsable.
- **Asignación a versiones**: Asigna requisitos a versiones o ciclos definidos en el árbol de versiones del módulo Versiones.
- **Análisis de requisitos**: Revisa tus requisitos para asegurarse de que se ajuste al ámbito de requisitos definido. Una vez que el requisito haya sido aprobado, cambia el estado del requisito de No revisado a Revisado. Para facilitar la tarea de revisión de los requisitos podrás generar informes y gráficos.
- **Establecimiento de línea base**: Crea una línea base para cerrar una versión o comparar hitos significativos en su ciclo de vida de aplicación.

Especificación de requisitos

- Los requisitos se registran en el módulo Requisitos creando un árbol de requisitos. El árbol de requisitos es una representación gráfica de las especificaciones de requisitos que muestra la relación jerárquica existente entre varios requisitos. El árbol incluye diferentes grupos de requisitos en función del tipo de requisitos o área funcional.
- Podrás crear una lista de requisitos detallados en el árbol de requisitos para cada grupo de requisitos. Los requisitos del árbol están descritos en detalle, pueden incluir datos adjuntos y vínculos entre requisitos y otras entidades relacionados. Permite, asimismo, calcular y analizar riesgos de requisitos y establecer un nivel de prioridad, para ser tomados en consideración al crear el plan de pruebas.
- Una vez creado el árbol de requisitos, los requisitos podrán usarse como base para definir pruebas en su árbol del plan de pruebas. Con *Convertir en pruebas* podrás reutilizar los requisitos y convertirlos en entidades del árbol del plan de pruebas como asuntos de pruebas, pruebas y pasos o descripciones de pruebas.
- Además de crear los requisitos directamente en ALM, también puedes importarlos al proyecto de ALM desde Microsoft Word o Microsoft Excel.

Especificación de requisitos

- Campos
  - Id. req.
  - Nombre
  - Descripción
  - Tipo de requisito
  - Revisado
  - Prioridad
  - Producto
  - Elemento principal de requisito
  - Id. de seguimiento de requisito
  - Seguimiento de comentario
  - Comentarios
  - Versión y Ciclo de destino
  - Autor
  - Creado por
  - Fecha y Hora de creación
  - Modificado
  - Estado de la cobertura directa
  - Análisis de cobertura
  - Impacto empresarial RBQM
  - Impacto empresarial personalizado RBQM
  - Probabilidad de error personalizada RBQM
  - Complejidad funcional personalizada RBQM
  - Riesgo personalizado RBQM
  - Horas de pruebas personalizadas RBQM
  - Nivel de prueba personalizada RBQM
  - Fecha del último análisis RBQM
  - Impacto empresarial real RBQM
  - Probabilidad de error real RBQM
  - Complejidad funcional real RBQM
  - Riesgo real RBQM
  - Esfuerzo aleatorio estimado RBQM
  - Excluir del análisis RBQM
  - Probabilidad de error RBQM
  - Complejidad funcional RBQM
  - Riesgo RBQM
  - Horas de pruebas RBQM
  - Nivel de prueba RBQM
  - Usar: impacto empresarial personalizado RBQM, probabilidad de error personalizada RBQM, complejidad funcional personalizada RBQM, resultados personalizados RBQM, riesgo personalizado RBQM
- Información complementaria:
  - Datos adjuntos
  - Defectos vinculados
  - Seguimiento de requisitos
  - Cobertura de prueba
  - Vínculo a modelos empresariales
  - Análisis de riesgos/evaluación de riesgos
  - Historial

Seguimiento de los requisitos

- Seguimiento de requisitos define la relación entre dos o más requisitos. Al analizar el impacto de un cambio propuesto en un requisito específico, los vínculos de seguimiento muestran los otros requisitos que pueden verse afectados por el cambio.
- Puedes agregar vínculos de seguimiento al (Rastrear desde) y del requisito (Rastrear a) en la ficha *Seguimiento de los requisitos*. Para revisar las asociaciones y dependencias que existen entre los requisitos usa la ficha *Análisis de impacto*. Puedes generar una matriz de seguimiento para determinar el grado de integridad de las relaciones entre sus requisitos.
- La matriz de seguimiento permite determinar el alcance de las relaciones entre los requisitos con otros requisitos y los requisitos con pruebas. Resulta de utilidad para verificar si se cumplen los requisitos y permite identificar si se han efectuado cambios respecto al ámbito de los requisitos.
- Cuando un requisito cambia, HP ALM puede alertar a los requisitos afectados.

Gestión de calidad basada en riesgos

- Al planificar el modo de hacer pruebas a los requisitos, comúnmente solo se dispone de un número limitado de recursos y no se pueden hacer pruebas integrales a todos los requisitos. Se deben alcanzar compromisos y hacer pruebas parciales de los requisitos que presentan una importancia baja para el negocio o de aquellos cuyo riesgo asociado a la implementación es bajo.
- La función de gestión de calidad basada en riesgos permite calcular el nivel al que hacer pruebas del requisito en función de la naturaleza del mismo y de los recursos de los que se dispone. Podrás entonces proceder a planificar el proceso de pruebas basado en estas recomendaciones.
- Los tipos de requisitos que tengan habilitada la gestión de calidad basada en riesgos admitirán el análisis de riesgos, denominándose requisito de análisis, o una evaluación de riesgos individual, denominándose requisito de evaluación.
- Se entiende por requisito de análisis aquel requisito que pertenece a un tipo que representa niveles más altos en la jerarquía del árbol de requisitos, como el tipo Carpeta.
- El análisis de riesgos de un requisito de análisis se efectúa en función de los requisitos de evaluación secundarios en el árbol de requisitos.
- Se realiza la suma de los resultados de riesgos de varios requisitos de evaluación para obtener un análisis de riesgos global que será de utilidad para determinar el esfuerzo y la estrategia de pruebas.

Gestión de calidad basada en riesgos

- Se entiende por requisito de evaluación aquel requisito que pertenece a un tipo que representa requisitos de detalle de los requisitos de análisis y que por lo tanto se encuentran en un nivel inferior en la jerarquía del árbol de requisitos. Los requisitos de evaluación de un requisito de análisis determinado forman la base del análisis de riesgos de dicho requisito de análisis.
- El riesgo se compone de su transcendencia empresarial, que mide la importancia del requisito para el negocio, y probabilidad de error, que indica la probabilidad de que una prueba basada en el requisito arroje errores. La complejidad funcional indica la complejidad de la implementación del requisito.
- Podrás *Determinar la complejidad funcional y el riesgo* en la ficha *Evaluación de riesgos* del Detalle del requisito. En la ficha *Preguntas de la evaluación* se pueden asignar valores a los criterios de Trascendencia empresarial, Probabilidad de error y Complejidad funcional o, en lugar de asignar valores a cada conjunto de criterios, se puede asignar valores personalizados directamente a cada categoría en la ficha *Resultados de la evaluación*.
- Podrás activar cada uno de los tipos de requisitos de la gestión de calidad basada en riesgos y personalizar la configuración predeterminada de la gestión de calidad basada en riesgos.

Modelos empresariales (BPM)

- El módulo Modelos empresariales de HP ALM aborda la necesidad de una conexión más robusta entre el modelado de procesos empresariales, la gestión de control de calidad y las definiciones de requisitos. El módulo integra los modelos de procesos empresariales en el ciclo de vida de la aplicación.
- Esta integración potencia la colaboración entre las distintas funciones implicadas en el modelado de procesos empresariales y en los ciclos de vida de prueba, facilitando por tanto la comunicación entre usuarios empresariales y las personas en departamentos más técnicos. Esta colaboración facilita unos mejores resultados empresariales identificando actividades de alto nivel, guiando por tanto al gestor de QA para determinar los requisitos de pruebas de alto nivel.
- La integración de modelos de procesos empresariales en ALM implica la importación de modelos de procesos empresariales en ALM y la vinculación de requisitos y pruebas con flujos empresariales de extremo a extremo, modelos y actividades. Una vez ejecutadas las pruebas podrá mostrar vistas de estado de la calidad en el nivel del modelo de procesos empresariales.
- Para trabajar con modelos de procesos empresariales en ALM deberás en primer lugar diseñar modelos con herramientas de modelado estándar e importar los modelos a ALM.
- Además de las entidades estándar del modelo de procesos empresariales, ALM permite efectuar pruebas de calidad en flujos empresariales de extremo a extremo ("rutas").

Plan de pruebas

- El desarrollo de un plan de pruebas claro y conciso es fundamental para el éxito del proceso de pruebas de una aplicación. Un buen plan de pruebas permite evaluar la calidad de la aplicación en cualquier etapa del proceso de gestión de la misma.
- Las aplicaciones suelen ser demasiado grandes para poderles hacer pruebas integrales. El módulo Plan de pruebas permite dividir la aplicación según criterios funcionales.
- Podrás dividir una aplicación en unidades o asuntos creando carpetas en el árbol del plan de pruebas. Con ello se obtiene una representación gráfica del plan de pruebas en la cual se muestran las pruebas según la relación jerárquica de sus funciones. Existen varios métodos para organizar los planes de pruebas por asuntos como:
  - La funcionalidad de la aplicación: como dominios, operaciones, roles, …
  - El tipo de pruebas: como funcionales, de interfaz del usuario, de rendimiento, de carga, …
- Una vez que hayas definido los asuntos del árbol, podrá decidir qué pruebas crear para cada asunto y agregarlas al árbol. Llegado este punto podrás definir información básica sobre la prueba, como su nombre, estado y el diseñador. Para ilustrar una prueba se podrá también adjuntar un archivo, URL, instantánea de la aplicación o información del sistema. A continuación definirás los pasos que contendrá la prueba. Los pasos de la prueba contienen instrucciones detalladas sobre cómo ejecutar una prueba y evaluar los resultados.

Plan de pruebas

- Cada prueba debe tener un objetivo claramente diferenciado como la verificación de una función o requisito del sistema en particular. Las pruebas que definas deberán estar basadas en los objetivos que establezca al principio del proceso de gestión de la aplicación.
- Los campos del plan de pruebas son Id. de prueba, Nombre de la prueba, Descripción, Prioridad de prueba, Tipo, Comentarios, Fecha de creación, Modificado, Diseñador, Tiempo de desarrollo estimado, Estado de la ejecución, Ruta, Estado, Asunto, Plantilla y Esfuerzo de prueba.
- Además de crear el árbol del plan de pruebas en el módulo Plan de pruebas de ALM, también puedes importar datos de planes de pruebas desde Microsoft Word o Microsoft Excel a el proyecto de ALM.
- Se pueden agregar pruebas al árbol del plan de pruebas de ALM desde el software de pruebas admitido, como Unified Functional Testing.

Diseño de pruebas

- Una vez que agregue una prueba al árbol del plan de pruebas, podrá crear la prueba definiendo pasos de diseño. Antes de definir los pasos de pruebas, hay que decidir si realizar la pruebas manualmente o automatizarlas. Automatizar una prueba permite que la prueba sea reutilizable, repetible y la ejecución no supervisada a alta velocidad.
- Un paso de prueba incluye las acciones que deben realizarse en la aplicación, la información que debe introducirse y los resultados esperados.
- En el caso de pruebas manuales deberás definir pasos, ejecutarlos en la aplicación manualmente y registrar los resultados de cada paso. Usa las pruebas manuales en aquellos casos en los que la prueba requiera una respuesta del evaluador, en pruebas de usabilidad, pruebas de un solo uso, pruebas que necesiten ser ejecutadas de inmediato, pruebas que requieran conocimiento sobre la aplicación y pruebas sin resultados predecibles.
- Las pruebas automatizadas requieren que se creen secuencias de comandos y ejecutar la prueba con una herramientas de pruebas de HP (Unified Functional Testing, LoadRunner o Visual API-XP) o de terceros. En el caso de pruebas automatizadas, el experto en pruebas podrá emplear los pasos de diseño como base para crear la secuencia de comandos detallada de la prueba.

Diseño de pruebas

- Para incluir instrucciones de uso general en una prueba, se puede crear una prueba (también llamada plantilla de prueba) con dichas instrucciones y con Llamar a la prueba en la ficha Pasos de diseño incluirla como una paso mas de la prueba. Es posible agregar parámetros a las plantillas de prueba pruebas para incrementar su flexibilidad y facilitar su reutilización.
- Una vez que se hayan creado los pasos para una prueba manual, podrás generar un esqueleto de secuencias de comandos que contenga las secuencias de comandos para ejecutar la prueba como prueba automatizada.
- ALM permite usar la misma prueba para hacer pruebas a diferentes casos de uso, cada uno con su propia configuración de prueba. Cada configuración de prueba emplea un conjunto de datos distinto. Podrás definir los datos agregando valores de parámetros de pruebas para cada configuración de prueba. Un parámetro de pruebas es una variable a la que se le puede asignar un valor.

Diseño de pruebas

- Los tipos de pruebas que están disponibles en el módulo Plan de pruebas son:
  - MANUAL: Prueba que se ejecuta manualmente.
  - BUSINESS-PROCESS: Prueba de proceso empresarial (BPM).
  - FLOW: Prueba compuesta por una colección de componentes empresariales en una secuencia fija que realiza una tarea específica (BPM).
  - LR-SCENARIO: Escenario ejecutado por LoadRunner, la herramienta de pruebas de carga de HP.
  - PERFORMANCE-TEST: Prueba de rendimiento.
  - QAINSPECT_TEST: Prueba ejecutada por QAInspect, la herramienta de pruebas de seguridad de HP.
  - QUICKTEST_TEST: Prueba GUI ejecutada por Unified Functional Testing, la herramienta de pruebas empresarial funcional de HP.
  - SERVICE-TEST: Una prueba API, creada en Unified Functional Testing o HP Service Test, la herramienta de HP para crear pruebas para aplicaciones sin interfaz gráfica de usuario, como por ejemplo servicios Web y REST.
  - SYSTEM-TEST: Prueba que indica a ALM que proporcione información del sistema, capture una imagen del escritorio o reinicie un equipo.
  - VAPI-XP-TEST: Prueba creada por Visual API-XP, la herramienta de pruebas API de arquitectura de pruebas abierta de ALM.
  - VuGenScript: Una secuencia de comandos de VuGen ejecutada por LoadRunner.

Diseño de pruebas

- Los campos de la pruebas son Id. de prueba, Nombre de la prueba, Descripción, Prioridad de prueba, Tipo, Comentarios, Fecha de creación, Modificado, Diseñador, Tiempo de desarrollo estimado, Estado de la ejecución, Ruta, Estado, Asunto, Plantilla y Esfuerzo de prueba.
- La información complementaria se muestra en las fichas:
  - Pasos de diseño: Enuncia las instrucciones sobre cómo ejecutar la prueba seleccionada.
  - Secuencia de comandos de la prueba: La secuencia de comandos de pruebas que ejecuta la herramienta de pruebas de la prueba seleccionada.
  - Parámetros: Enuncia los parámetros asociados a la prueba seleccionada.
  - Configuraciones de la prueba: Muestra las configuraciones de la prueba seleccionada.
  - Datos adjuntos: Muestra en una lista datos adjuntos que ofrecen información adicional sobre la prueba seleccionada.
  - Cobertura de requisitos: Enuncia los requisitos que cumple la prueba seleccionada.
  - Defectos vinculados: Enuncia los defectos vinculados a la prueba seleccionada.
  - Dependencias: Muestra las relaciones de dependencia existentes entre entidades como recursos de pruebas y pruebas.
  - Vínculo a modelos empresariales: Muestra en una lista las entidades de modelos empresariales vinculadas a la prueba seleccionada.
  - Criterios: Muestra los criterios de la prueba de proceso empresarial seleccionada (BPM.
  - Historial: Enuncia los cambios efectuados a la prueba seleccionada.
  - Análisis en vivo: Muestra una representación gráfica de datos de pruebas relacionados con la carpeta de asuntos de pruebas seleccionada.
  - Diseño de pruebas: Performance Center: Muestra un resumen detallado de la prueba de rendimiento seleccionada.

Parámetros de pruebas

- Es posible agregar parámetros a las pruebas. Un parámetro de pruebas es una variable a la que se puede asignár un valor fuera de la prueba en la que ha sido definida. El uso de parámetros puede incrementar la flexibilidad de sus pruebas ya que permite ejecutar la misma prueba repetidamente con datos diferentes en cada ocasión. Ello resulta de utilidad cuando existen pasos generales que con frecuencia se desean efectuar como parte de otras pruebas.
- Por ejemplo, podrás crear una plantilla de prueba que permita el inicio de sesión de un usuario cuando se inicie la aplicación. En algunos casos puede que inicie sesión como usuario regular, pero en otros tendrá que iniciar como administrador. Para ello podrás crear dos parámetros entre triples paréntesis angulados: <<<nombre_usuario>>> y <<<contraseña>>>, y cambiar los valores en función del tipo de prueba que llame a la plantilla. Si el inicio de sesión más habitual es el de un usuario regular, podrás configurar los valores predeterminados de los parámetros en base a un nombre de usuario y contraseña habituales. Deberás llamar a la plantilla al principio de cada prueba. Los campos de parámetros de pruebas son: Nombre de parámetro, Descripción, Valor predeterminado, Orden, Modificado y Usado.
- Al trabajar con pruebas manuales podrás agregar parámetros a los pasos de diseño desde la prueba misma o llamando a dichos parámetros de otras pruebas. En una prueba automatizada podrás definir parámetros de una secuencia de comandos de pruebas desde la misma prueba o cargar los parámetros desde un archivo de recursos de pruebas compartido.

Configuraciones de prueba

- HP ALM proporciona capacidades de conocimiento de datos que le permiten determinar qué conjuntos de datos utilizar cuando se ejecuta la prueba. Una ellas es la capacidad de definir configuraciones de prueba para facilitar la reutilización de pruebas y mejorar la cobertura de requisitos.
- Las configuraciones de prueba esencialmente desvinculan los datos de la prueba, convirtiendo la prueba en genérica y facilitando su reutilización. Con las configuraciones de prueba, podrás:
  - Compartir orígenes de datos comunes por las distintas pruebas.
  - Probar diversos casos de uso, cada vez con un conjunto de datos distinto.
- Las configuraciones de prueba proporcionan una granularidad más fina para la cobertura de requisitos. En lugar de únicamente cubrir los requisitos a nivel de prueba, podrá vincular un requisito a una configuración de prueba específica.

Configuraciones de prueba

- HP ALM, en la ficha *Configuraciones*, permite determinar qué datos utilizar cuando se ejecutan configuraciones de prueba. Dependiendo del tipo de prueba, puedes asociar datos estáticos, datos dinámicos o ambos para su configuración de prueba. La ficha Datos refleja con precisión las opciones disponibles.
- Los datos estáticos se suministran para la configuración de prueba mediante la introducción de datos en una cuadrícula, directamente desde dentro de ALM, cuando la cantidad de datos que se pretende asociar es pequeña. Los datos dinámicos se suministran para la configuración de prueba creando un archivo Microsoft Excel externo, cargándolo como un recurso de pruebas en ALM y asociándolo con la configuración de prueba, lo que permite especificar grandes cantidades de datos que son más fáciles de mantener en un archivo externo.
- Para la cobertura, los requisitos se pueden vincular a la prueba o a configuraciones de prueba específicas. Distintos requisitos se pueden vincular a distintas configuraciones de la misma prueba.
- Puedes definir un conjunto de pruebas en el módulo Laboratorio de pruebas. Un conjunto de pruebas puede incluir alguna o todas las configuraciones de prueba definidas para una prueba, o puede incluir configuraciones de prueba basadas en la cobertura de requisitos. Al ejecutar un conjunto de pruebas, los valores de parámetros son recuperados del conjunto de datos en función de los valores definidos para cada configuración de prueba.

Recursos de pruebas

- El módulo *Recursos de pruebas* permite gestionar recursos usados en otras pruebas. Puedes organizar tus recursos definiendo un árbol de recursos de pruebas jerárquico que contenga carpetas de recursos y recursos. Por cada recurso del árbol podrás seleccionar y cargar un conjunto de archivos de recursos en el repositorio de HP ALM. Estos archivos pueden usarse en una o varias pruebas.
- Entonces podrás definir dependencias entre recursos y pruebas. Las dependencias definen las relaciones existentes entre entidades como pruebas, componentes y recursos de pruebas.
- Al analizar el impacto de un cambio propuesto en una entidad específica, las dependencias muestran las otras entidades que pueden verse afectadas por el cambio. Por ejemplo, puedes desear ver dependencias antes de eliminar o copiar una entidad.
- Las relaciones de dependencia se muestran en la ficha *Dependencias*. Esta ficha se encuentra disponible en los módulos *Plan de pruebas, Componentes empresariales* y *Recursos de pruebas*. Podrás ver tanto las entidades utilizadas por la entidad seleccionada como las entidades que están utilizando la entidad seleccionada.

Cobertura de requisitos y pruebas

- Es fundamental que las pruebas del plan de pruebas cumplan los requisitos originalmente establecidos. Para poder hacer seguimiento de la relación entre requisitos y pruebas, podrás agregar vínculos entre ellos.
- En el módulo Plan de pruebas, podrás crear cobertura de requisitos seleccionando requisitos que serán vinculados a una prueba. La cobertura de requisitos facilita la evaluación del impacto de un cambio en la prueba o requisito. Una prueba puede cubrir más de un requisito.
- Asimismo, en el módulo Requisitos, podrás crear cobertura de pruebas vinculando pruebas a un requisito. La cobertura de pruebas facilita la evaluación del impacto de un cambio en la prueba o requisito. Un requisito puede estar cubierto por más de una prueba.
- En lugar de únicamente cubrir los requisitos a nivel de prueba, podrá cubrirlos por configuraciones de prueba. Una configuración de prueba representa un caso de uso específico de una prueba. Las configuraciones de prueba de cobertura con requisitos ofrecen una granularidad más precisa respecto a cobertura de requisitos al permitir la cobertura por diferentes casos de uso de una prueba.
- Si trabajas con el módulo *Modelos empresariales*, podrás vincular las entidades de modelos a pruebas, tanto en el módulo Plan de pruebas como en el de Modelos empresariales.

Pruebas del sistema

- Puedes ejecutar una prueba del sistema para recuperar información del sistema de un equipo, ver una imagen del escritorio capturada en una ejecución de una prueba en un equipo o reiniciar un equipo. Por ejemplo, puedes ejecutar una prueba de limpieza del sistema que reinicie el equipo en el que la prueba automatizada ha dado error. También puedes crear una prueba del sistema para recuperar información sobre el uso de los recursos de un equipo antes o después de una ejecución de pruebas.
- Podrás crear una prueba del sistema agregando una prueba de tipo SYSTEM-TEST a la carpeta de asuntos de pruebas, definiendo la prueba y agregando la prueba a un conjunto de pruebas.
- Al ejecutar una prueba del sistema deberán de crearse los pasos siguientes:
  - SysInfo. Recopilación de información del sistema
  - Instantánea. Captura de imagen de escritorio
  - Reinicio del sistema inicial y Reinicio del sistema final. Reinicio del equipo
- Podrás ver detalles de cada uno de los pasos después de que finalice la ejecución de la prueba del sistema. También podrá ver la información del sistema que ha sido recuperada (como por ejemplo el CPU, la memoria y los procesos que están ejecutándose en el equipo) y una imagen del equipo que está ejecutando la prueba del sistema.

Pruebas VAPI-XP

- La herramienta de pruebas VAPI-XP permite crear secuencias de comandos de pruebas con Microsoft VBScript, Microsoft JavaScript (versión JScript), PerlScript y PythonScript, e integrar estas secuencias en tu proceso de gestión de aplicaciones. Por medio de las secuencias de comandos de pruebas VAPI-XP podrás realizar pruebas de servidores COM/DCOM, servicios Web basados en SOAP, APIs de Java (como por ejemplo clases Java y EJBs) y aplicaciones de consola. Podrás asimismo utilizar VAPI-XP para crear usuarios virtuales de LoadRunner.
- Además, VAPI-XP se encuentra totalmente integrado con HP ALM, lo cual permite diseñar la secuencia de prueba VAPI-XP de manera que llame a cualquier prueba o conjunto de pruebas de ALM y ejecutarlo como parte de la secuencia misma. Ello permite construir un flujo de ejecución de conjuntos de pruebas más avanzado, en el que se pueden filtrar las pruebas del conjunto durante la ejecución, en función del estado o tipo de cada prueba.
- VAPI-XP también ha sido totalmente integrado con la API de arquitectura abierta de pruebas de ALM. Todos los métodos y clases de API de arquitectura abierta de pruebas pueden ser referenciados desde la interfaz de usuario VAPI-XP de manera que puedan incluirse fácilmente en la secuencia de comandos de pruebas.

Recursos de laboratorio

- La ejecución de las pruebas automatizadas consumen muchos recursos y requieren servidores adicionales.
- HP ALM Lab Management permite gestionar recursos para pruebas funcionales y de rendimiento del lado servidor. Los módulos Recursos de laboratorio de ALM se utilizan para ver y gestionar los recursos de prueba remotos y para automatizar esquemas de implementación.
  - Con el módulo Hosts de pruebas podrás ver y modificar los hosts de pruebas.
  - Con el módulo Entornos de AUT podrás ver y modificar los parámetros de entorno usados por los hosts de AUT. Si usas un entorno conectado a servidores de CDA (HP Continuous Delivery Automation), podrás también vincular sus configuraciones de entorno a CDA para automatizar los esquemas de implementación.

Recursos de laboratorio

- El módulo **Hosts** de prueba de ALM se usa para ver y modificar las propiedades de los hosts de pruebas. Los host de pruebas de la agrupación de hosts de su proyecto se muestran en la cuadrícula del módulo Hosts de pruebas.
- Se pueden usar los hosts de pruebas para las pruebas de rendimiento y funcionales del lado servidor. En lugar de iniciar una prueba desde la consola del equipo, las pruebas se pueden controlar por el servidor de ALM. La ejecución en el lado servidor está disponible para conjuntos de pruebas funcionales y de rendimiento. A cada host de pruebas se le asigna una ubicación, propósito y atributos. Los hosts de pruebas se pueden localizar en el laboratorio de pruebas o se pueden aprovisionar en la nube según se necesite.
- Al programar una prueba se reservará para la prueba el host de pruebas adecuado, y este host no podrá reservarse para otra prueba a menos que exista otro host idóneo para su prueba. La función *Intervalos de tiempo* de ALM permite reservar recursos por adelantado para garantizar que los recursos necesarios estén disponibles cuando se esté preparado para realizar las tareas.
- Para agregar un host de pruebas a ALM, debes crear primero el host en el módulo Hosts de prueba y registrarlo luego mediante Lab Service de HP ALM.

Recursos de laboratorio

- Los **entornos de AUT** permiten a los usuarios hacer que la ejecución de pruebas de Lab Management sea más dinámica al parametrizar los datos de entorno usados para las pruebas.
- Un entorno AUT es un contenedor de un conjunto de parámetros de entorno de AUT. Se puede proporcionar un valor predeterminado para cada parámetro de entorno de AUT.
- Con un entorno de AUT se crea un conjunto de configuraciones de entornos de AUT. Cada configuración de entorno AUT contiene un conjunto de parámetros de entorno de AUT que se puede sobrescribir. En lugar de definir y ejecutar varias pruebas distintas que usan la misma lógica pero necesitan diferentes parámetros de entorno de AUT, puedes simplemente proporcionar una configuración de entorno de AUT determinada que ALM inserte en tu prueba en tiempo de ejecución. ALM usará entonces los valores de parámetro que definiste en la configuración de entornos de AUT al implementar el entorno y ejecutar la prueba.
- Las configuraciones de entorno de AUT son un componente clave en la solución Continuous Delivery de ALM. Permiten la implementación de extremo a extremo automatizada y proporciona un marco de pruebas óptimo para una implementación de aplicaciones más eficiente, fiable y rápida.
- La vinculación de proyectos a HP Continuous Delivery Automation (CDA) permite implementar y aprovisionar dinámicamente los entornos de pruebas. Para hacer dinámicas las definiciones de valores de parámetros de entorno, vincula a CDA las configuraciones de entorno de AUT.

Ejecución de pruebas

- HP ALM permite automatizar íntegramente el complicado proceso de implementar y probar la compilación de una aplicación. ALM puede usarse junto con las funciones de Lab Management para programar implementaciones y conjuntos de pruebas de manera que se ejecuten por la noche o una vez por hora. Dichas implementación y prueba se ejecutan sin intervención del usuario y pueden programarse para ser ejecutadas inmediatamente después de que finalice una compilación. También puede hacer que la compilación se implemente en un entorno determinado definido por usted, o incluso integrar con HP Continuous Delivery Automation (CDA) para implementarlo dinámicamente en una nube pública o privada.
- ALM y Lab Management ofrecen los componentes de prueba que permiten a un equipo de aplicación lograr un estado de Entrega continua, en el cual el software se crea, empaqueta, implementa y prueba de manera automatizada, lo cual permite ofrecer un software fiable, eficiente y a gran velocidad.
- ALM ofrece varios tipos de pruebas automatizadas para probar la funcionalidad de la aplicación que se somete a prueba (AUT). Las dos principales categorías de tipos de pruebas son Funcional y Rendimiento:
  - Las pruebas funcionales permiten comprobar si la aplicación funciona como debe.
  - Las pruebas de rendimiento permiten comprobar si la aplicación puede hacer frente a la carga y la demanda.

Ejecución de pruebas

- Para ejecutar pruebas es necesario crear en primer lugar **conjuntos de pruebas** y seleccionar las pruebas que se incluirán en cada conjunto. Un conjunto de pruebas contiene un subconjunto de pruebas de un proyecto de HP ALM diseñadas con el propósito de lograr ciertos objetivos de pruebas. A medida que tu aplicación cambia, ejecutarás pruebas manuales y automatizadas en el proyecto con objeto de identificar defectos y evaluar la calidad.
- ALM admite distintos tipos de conjuntos de pruebas:
  - Predeterminado: Para ejecutar pruebas funcionales del lado cliente, controladas e iniciadas localmente. Puedes agregar pruebas funcionales tanto manuales como automatizadas a este conjunto de pruebas y que se ejecuten de manera ad hoc.
  - Funcional: Para ejecutar pruebas funcionales desatendidas del lado servidor. Solo se pueden agregar pruebas funcionales automatizadas a este conjunto de pruebas.
  - Rendimiento: Para ejecutar pruebas de rendimiento remotas no supervisadas. Solo se pueden agregar pruebas de rendimiento a este conjunto de pruebas.

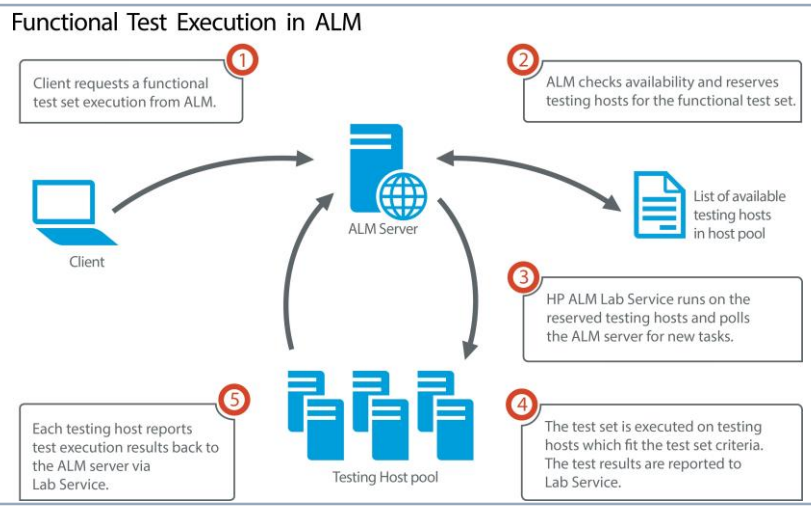
Ejecución de pruebas

- Las pruebas de los **conjuntos de pruebas funcionales** se ejecutan mediante ejecuciones del lado servidor. Ello significa que no hay que estar en los alrededores para iniciar y controlar las pruebas. Los conjuntos de pruebas funcionales se ejecutan por medio de intervalos de tiempo, de modo que se puede programar un conjunto de pruebas para que se ejecute inmediatamente o transcurrido un tiempo si hay hosts disponibles para la prueba en ese momento. Una vez se programe la prueba, ALM se asegura de que se reserven los recursos necesarios para el conjunto de pruebas. El conjunto de pruebas se inicia sin intervención del usuario y se ejecuta en secuencia con la entrada que se haya proporcionado previamente.
- Puedes organizar un conjunto de pruebas funcionales para que se ejecute en el servidor programando un *intervalo de tiempo*. Un intervalo de tiempo contiene un conjunto de pruebas, los detalles de los hosts de pruebas en los que se ejecuta el conjunto de pruebas, así como la hora y la duración de la ejecución del conjunto de pruebas.
- Las pruebas funcionales se ejecutan en hosts de pruebas que están configurados en *Lab Management*. Para ejecutar pruebas en un conjunto de pruebas funcionales, el proyecto deberá disponer de hosts de pruebas. Al programar una prueba se reservará para dicha prueba el host de pruebas adecuado y este host no podrá reservarse para otra prueba a menos que exista otro host idóneo para dicha prueba. ALM gestiona la asignación de hosts dinámicamente.

Ejecución de pruebas

- Puedes controlar la ejecución de las instancias de pruebas en un conjunto de pruebas y crear un *Calendario de ejecución de la prueba*.
- Cuando uses la ficha *Flujo de ejecución* del módulo Laboratorio de pruebas, puedes programar las ejecuciones de la prueba: especificar la hora y fecha, así como configurar las condiciones para la ejecución de una instancia de prueba. Puedes cambiar a modo diagrama la forma en que se muestra el Flujo de ejecución.
- Una **condición** se basa en los resultados de otra instancia de prueba especificada en el Flujo de ejecución. Al configurar las condiciones, puedes indicar al módulo Laboratorio de pruebas que posponga la ejecución de la instancia de prueba hasta que la instancia de prueba especificada haya terminado la ejecución o se haya validado.
- También puedes configurar la secuencia en la que ejecutar las instancias de pruebas.

Ejecución de pruebas



Ejecución de pruebas

- Las pruebas de los **conjuntos de pruebas predeterminadas** se ejecutan mediante ejecuciones del lado cliente. Tu controlas directamente la prueba en tu equipo local. Podrás ejecutar conjuntos de pruebas predeterminadas manual o automáticamente en ALM.
- Para ejecutar manualmente las pruebas se utiliza *HP Sprinter*, que proporciona funcionalidad mejorada, o el *Ejecutor manual*. Cuando ejecutes una prueba manualmente, sigues los pasos de pruebas y realizas las operaciones necesarias en la aplicación sometida a prueba. Marcaras como realizado correcta o incorrectamente un paso dependiendo de si los resultados reales de la aplicación coinciden con los resultados esperados.
- Para ejecutar automáticamente las pruebas se utiliza el *Ejecutor automático*. Cuando se ejecute automáticamente una prueba automatizada, ALM abrirá la herramienta de pruebas seleccionada automáticamente, ejecutará la prueba en el equipo local o hosts remotos y exportará los resultados a ALM.
- También podrás ejecutar automáticamente pruebas manuales. Cuando se ejecuta automáticamente una prueba manual y se especifica un host remoto, ALM notificará por correo electrónico al especialista en pruebas designado para que ejecute la prueba en un host determinado.

Ejecución de pruebas

- Una vez hayan tenido lugar las ejecuciones de pruebas, procederás a revisar y analizar los resultados de las pruebas. Tu objetivo será identificar los pasos que han arrojado error y determinar si se ha detectado algún defecto en la aplicación o si los resultados esperados de las pruebas necesitan ser actualizados. Podrás validar resultados de pruebas regularmente visualizando datos de ejecuciones y generando informes y gráficos. Si se detecta un defecto, podrás crear un nuevo defecto y vincularlo al conjunto de pruebas, instancia de prueba, ejecución de pruebas o paso de ejecución, o a un defecto existente.
- Las **ejecuciones de borrador** permiten probar las pruebas mientras se encuentran en la fase de desarrollo o después de haber sido modificadas. La definición de una prueba como ejecución en modo borrador indica a ALM que ignore los resultados de la ejecución: el resultado de la ejecución no influirá ni en el estado de ejecución de la prueba, el estado de la instancia de prueba, ni en el estado de cobertura. ALM también ignorará las ejecuciones de borrador al calcular el número restante de instancias de prueba que serán ejecutadas y al mostrar los resultados en gráficos de cobertura, progreso y análisis en vivo.
- Los conjuntos de pruebas se crean y definen en el módulo *Laboratorio de pruebas*. Una vez creados los conjuntos de pruebas, puedes asignar las carpetas de conjuntos de pruebas a los ciclos definidos en el árbol de versiones en el módulo Versiones. Con *Performance Center* podrás ejecutar pruebas de rendimiento con objeto de crear carga en una aplicación y probar su rendimiento.

Ejecución de pruebas

- ALM ofrece un modo de agrupar varios conjuntos de pruebas, independientemente de su tipo, creando un **Conjunto de verificación de compilación**. El conjunto de verificación de compilación permite comprobar el estado global de la compilación y puede contener tanto conjuntos de pruebas funcionales como de rendimiento.
- Puedes crear varios conjuntos de verificación de compilación para comprobar la integridad de la aplicación en varios niveles. Puede crearse y programarse un conjunto de verificación de compilación voluminoso para que se ejecute siempre por la noche, y puede crearse y programarse otro que solo contenga los conjuntos de pruebas criticas de manera que se ejecute cada hora, o manualmente cada vez que se cree una compilación.
- ALM ofrece un modo de definir un conjunto de parámetros de entorno que se puede agrupar a conjuntos de configuración de compilación y conjuntos de pruebas llamado Entorno de AUT.
- ALM permite programar mediante intervalos de tiempo la implementación y las pruebas que se realizarán a su aplicación en un futuro. ALM se asegurará de que se reserven con antelación los recursos para pruebas que el intervalo de tiempo requiera.

Resultados de las ejecuciones de pruebas

- Después de ejecutar las pruebas, puedes ver los resultados en el módulo **Ejecuciones de pruebas** en HP ALM. El módulo contiene fichas que permiten estudiar los resultados de las ejecuciones de instancias de pruebas, ejecuciones de conjuntos de pruebas y ejecuciones de los conjuntos de verificación de compilación.
- En la ficha Ejecuciones de pruebas, puedes ver:
  - Los resultados de las pruebas manuales que constan del estado general validado/con error de la prueba, así como el estado validado/con error de cada paso de una ejecución de prueba.
  - Los resultados disponibles para las pruebas automáticas que varían en función del tipo de prueba.
- Estos resultados ayudarán a determinar si se ha detectado un defecto en la aplicación. En algunos casos, puedes decidir que un paso no es correcto porque los resultados esperados ya no son válidos y tienen que actualizarse.
- Después de la ejecución de pruebas manuales y automatizadas, puedes ver los resultados e información general de las ejecuciones deseadas, consultar los detalles de la ejecución de la prueba, comparar los resultados de la ejecución de la prueba más reciente, gestionar datos adjuntos, ver y editar información sobre la configuración de la ejecución de la prueba, gestionar los defectos vinculados, así como ver un historial de cambios de la ejecución de la prueba. Los resultados se pueden mostrar en una cuadrícula y filtrar las ejecuciones que cumplen ciertos criterios definidos.

Seguimiento de defectos

- La búsqueda y reparación de los defectos de las aplicaciones de forma eficaz resulta fundamental para el proceso de desarrollo. Gracias al **módulo Defectos** de HP ALM, podrás informar sobre los errores en el diseño de la aplicación y realizar un seguimiento de los datos extraídos de los registros de defectos a lo largo de todas las etapas del proceso de gestión de la aplicación.
- El módulo Defectos se usa para crear los defectos de la aplicación para un proyecto de ALM y realizar un seguimiento de los defectos hasta que los desarrolladores y el evaluador de la aplicación determinen que se han resuelto los defectos.
- Los registros de defectos proporcionan información a los miembros de los equipos de control de calidad y de desarrollo de la aplicación sobre los nuevos defectos descubiertos por otros miembros. A medida que se monitoriza el progreso de la reparación de defectos, se actualiza la información del proyecto.
- Puedes vincular un defecto a requisitos, pruebas, conjuntos de pruebas, pruebas de proceso empresarial, flujos, instancias de pruebas, ejecuciones, pasos de ejecuciones, así como otros defectos. Puedes compartir y sincronizar defectos en varios proyectos de ALM mediante HP ALM Synchronizer.

Análisis

- HP ALM le ofrece las herramientas de análisis que te permiten analizar y mostrar los datos de ALM en varios formatos.
- En los módulos **Panel de resultados**, se pueden analizar los datos de ALM mediante la creación de gráficos, informes de proyectos e informes de Excel y páginas del panel de resultados que muestran varios gráficos en paralelo. El panel de resultados contiene los siguientes módulos y herramientas:
  - Módulo Vista de análisis: Contiene el árbol de análisis en el que organiza todos los elementos de análisis. Los elementos de análisis pueden ser de cualquiera de los siguientes tipos de análisis: gráficos, informes de proyectos e informes de Excel. Los usuarios con los permisos de administrador requeridos también tienen acceso a la ficha Menú de análisis. Esta ficha permite gestionar los elementos de análisis que se generan en el menú Análisis de determinados módulos, como Requisitos y Laboratorio de pruebas.
  - Módulo Vista de panel de resultados: Contiene el árbol de panel de resultados en el que se organizan las páginas del panel de resultados. En las páginas del panel de resultados puedes organizar varios gráficos que has creado en el árbol de análisis y mostrarlos en una vista única.
  - Herramienta Gráficos de análisis en vivo: Permite crear y mostrar gráficas dinámicas de los datos relacionados con los planes de pruebas y los conjuntos de pruebas.

Análisis

- Los gráficos de HP ALM ayudan a analizar y ver las relaciones entre los distintos tipos de datos. En ALM se pueden crear los siguientes tipos de gráficos:
  - Gráficos Vista de negocio: Un gráfico basado en una vista de negocio representa una única entidad o varias entidades y refleja únicamente información del valor empresarial.
  - Gráfico de entidades: Cada gráfico está basado únicamente en una entidad, como requisitos o defectos, y permite ver cualquiera de los atributos de la entidad.
- Al crear un gráfico de entidades, puedes usar varios tipos de gráficos. Cuando estés consultando los gráficos de entidades, puedes ver los detalles adicionales de los registros representados por cada barra o segmento. Una vez creados los gráficos en el módulo Vista de análisis, puedes seleccionar y organizar varios gráficos, y verlos en paralelo en una página del panel de resultados.
- Cuando trabajes con PPT, puedes crear y personalizar gráficos en el módulo Vista de análisis, en relación a los datos de KPI del módulo Versiones.
- Puedes crear gráficos que incluyan datos de varios proyectos de ALM.

Análisis

- Los informes de proyecto permiten diseñar y generar informes integrales que contienen la información de proyecto de HP ALM. Se pueden generar formato HTML, Microsoft Word o PDF.
- En un informe de proyecto, se definen las secciones y subsecciones, y cada una presenta los registros de una entidad de ALM determinada. Puedes elegir si se muestran los datos de una línea base seleccionada. Para cada sección del informe, se asigna una plantilla que determina los campos y el diseño de la sección. Se asignan también las plantillas de estilo y de documento que determinan el aspecto general del informe. Los informes se pueden crear con parámetros, lo que te permite crear un informe más flexible que solo se tienes que crear una vez, pero que puede ser utilizado en un número de contextos diferentes.
- Puedes exportar los datos de HP ALM a Microsoft Excel. Esto permite analizar los datos usando cualquiera de las capacidades disponibles en Excel. Un informe de Excel consta de un conjunto de datos definido por las consultas SQL en la base de datos del proyecto. También puedes ejecutar una secuencia de comandos de Visual Basic en los datos extraídos para procesarlos y analizarlos. En ALM 12.00, no se pueden crear informes de Excel pero se pueden consultar y editar informes de Excel existentes de versiones anteriores.

TESTLINK & MANTIS

TestLink - Instalación

- Instalar XAMPP (si no está instalado)
- Descomprimir Testlink al directorio \xampp\htdocs y renombrar quitando versión.
- En \xampp\htdocs\testlink:
  - Renombrar custom_config.inc.php.example por custom_config.inc.php
  - Editar custom_config.inc.php y al final antes del ?> añadir:
    - $tlCfg->log_path = TL_ABS_PATH . 'logs/';
    - $g_repositoryPath = TL_ABS_PATH . 'upload_area/';
- En caso de que de el error Fatal error: Call to undefined function plugin_get_current() in C:\xampp\htdocs\testlink\lib\functions\lang_api.php on line 67
  - Editar el fichero y añadir al principio, detrás de <?php
    - require_once('plugin_api.php');

TestLink - Instalación

- En el navegador: localhost\testlink
- Marcar New installation
- Marcar “I agree to the terms …” y continuar
- Si al final pone “Your system is prepared for TestLink configuration (no fatal problem found).” pulsar Continue
- En “Database admin login” introducir “root” y dejar password en blanco.
- Introducir en “TestLink DB login”, por ejemplo, “admin” y en password también “admin”.
- Pulsar en “Process TestLink Setup” y esperar a que aparezca la página de login.
- Entrar con:
  - login name: admin
  - password : admin

TestLink

- http://testlink.org/
- TestLink es una herramienta web de gestión de pruebas que ayuda a gestionar las pruebas funcionales de un proyecto permitiendo realizar las tareas relacionadas con el proceso de aseguramiento de calidad de software tales como: gestión de requerimientos, diseño de casos de prueba, planes de pruebas, ejecución de casos de prueba, seguimiento de informes de resultados del proceso de pruebas.
- Es una herramienta gratuita que permite crear y gestionar casos de pruebas y organizarlos en planes de prueba. Estos planes permiten a los miembros del equipo ejecutar casos de test y registrar los resultados dinámicamente, generar informes, mantener la trazabilidad con los requerimientos, así como priorizar y asignar tareas.
- Permite gestionar tantos proyectos de pruebas como sean necesarios, accesibles con diferentes roles de usuario con distintos permisos para los integrantes de un equipo de desarrollo.
- Los proyectos pueden definir diferentes plataformas de pruebas y mantener un inventario de las mismas.

TestLink

- Los planes de pruebas organizan y controlan la ejecución de los casos de pruebas y representan una prueba en un punto determinado en el tiempo.
- Un plan de pruebas debería ser una tarea o conjunto de tareas claramente definidas con unos plazos y un contenido. Podría contener una actividad de testeo para una nueva versión del producto o simplemente una petición de cambio para un cliente.
- Los planes se estructuran en Builds (construcciones o versiones) del producto, a las que se añaden los casos de prueba indicando el usuario responsable y las prioridades.
- Los planes permiten establecer Hitos, establecer los porcentajes de cobertura de casos de prueba por prioridad para un determinado periodo de tiempo.
- Los Hitos se alcanzan cuando se alcanzan todos los niveles de cobertura establecidos, el estado de hito se puede consultar en las Métricas Generales del Plan de Pruebas.
- Los planes pueden establecer las plataformas de ejecución donde se van a ejecutar los casos de prueba del plan.

TestLink

- La ejecución de los casos de prueba se realiza dentro de los planes de pruebas por versión del software (build) bajo prueba y plataforma de ejecución.
- Para la ejecución del caso de prueba muestra los pasos a ejecutar y el resultado esperado, permitiendo registrar paso a paso el resultado (pasado, fallado, bloqueado) y unas observaciones.
- La ejecución concluye al asignar un resultado (pasado, fallado, bloqueado) al caso de prueba para una generación específica, opcionalmente se puede indicar la duración de la prueba y observaciones.
- Un caso está bloqueado cuando no es posible probarlo por algún motivo (por ejemplo, un problema de configuración no permite ejecutar la función a probar).
- En caso de fallo permite la integración con herramientas de gestión de incidencias como Mantis para el seguimiento de los defectos.
- Dispone de una amplia colección de informes y gráficos con los resultados y métricas del plan de pruebas.

Automatización de pruebas en TestLink

- En TestLink:
  - Mi Configuración → Interfaz API → Generar una nueva key
  - Gestión del Proyecto de Pruebas → Mi proyecto → Habilitar Automatización de Pruebas (API keys)
- El fichero custom_config.inc.php.example
  - Se renombra a custom_config.inc.php
  - Se añade:
  - $tlCfg->exec_cfg->enable_test_automation = ENABLED;
  - $tlCfg->api->enabled = TRUE;
- Se rearranca el servidor web para que utilice la nueva configuración.
- Descargar (https://code.google.com/archive/p/dbfacade-testlink-rpc-api/downloads):
  - testlink.api.java.client:testlink-api-client-2.0.jar
  - org.apache.ws.commons.util:ws-commons-util:jar:1.0.2
  - org.apache.xmlrpc:xmlrpc-client:jar:3.1.3
  - org.apache.xmlrpc:xmlrpc-common:jar:3.1.3

Automatización de pruebas en TestLink

```
public class testlinkTest {
    public static String DEVKEY="4989f97897a30e5cb6ad2eb09654ab90"; // API key del proyecto.
     // URL del RPC
     public static String URL= "http://localhost/testlink/lib/api/xmlrpc/v1/xmlrpc.php";
     String testProject="Mi proyecto"; // Nombre del proyeto
     String testPlan="Mi plan";// Nombre del plan
     String build="Mi build"; // Nombre de la build

    @Test
    public void test() {
        if(true) {
            updateTestLinkResult("Mi caso", null, TestLinkAPIResults.TEST_PASSED);
        } else {
            updateTestLinkResult("Mi caso", null, TestLinkAPIResults.TEST_FAILED);
        }
    }
    public void updateTestLinkResult(String testCase, String exception, String result) throws TestLinkAPIException {
        TestLinkAPIClient testlinkAPIClient = new TestLinkAPIClient(DEVKEY, URL);
        testlinkAPIClient.reportTestCaseResult(testProject, testPlan, testCase, build, exception, result);
    }
}
```


Mantis

- https://www.mantisbt.org/
- Mantis Bug Tracker es un sistema de gestión de incidencias, control de cambios y control de errores, es una aplicación web OpenSource multiplataforma que permite gestionar las incidencias de la empresa, sistemas o proyectos.
- Es un sistema fácil de usar y adaptable a muchos escenarios, tanto para incidencias técnicas, peticiones de soporte o bugs de un sistema.
- Es una aplicación realizada con php y mysql que destaca por su facilidad y flexibilidad de instalar y configurar.
- Mantis es una aplicación que permite a distintos usuarios crear tickets de cualquier tipo.
- A la hora de notificar una incidencia, el usuario tiene muchas opciones y campos a rellenar con el fin de hacer más fácil el trabajo del encargado de resolver el ticket.
- Mantis permite notificar a los usuarios novedades por correo electrónico.
- Se puede especificar un número indeterminado de estados para cada tarea (nueva, se necesitan más datos, aceptada, confirmada, asignada, resuelta, cerrada) y configurar la transición de estados.
- Permite la carga de plugins específicos de la plataforma que añaden funcionalidades extra.

Integración

- En Testlink: En el proyecto: Gestión del Gestor de Defectos → Crear → Tipo: Mantis SOAP

```
<issuetracker>
<username>administrator</username>
<password>curso</password>

<uribase>http://host.docker.internal:9081/</uribase>
<uriwsdl>http://host.docker.internal:9081/api/soap/mantisconnect.php?wsdl</uriwsdl>
<uriview>http://host.docker.internal:9081/view.php?id=</uriview>
<uricreate>http://host.docker.internal:9081/</uricreate>

<project>Curso</project>
<category>General</category>
<resolvedstatus>
<status><code>80</code><verbose>resolved</verbose></status>
<status><code>90</code><verbose>closed</verbose></status>
</resolvedstatus>
</issuetracker>
```

- Editar proyecto → Integración del Gestor de Defectos: Activo, Seleccionar el gestor creado

GESTIÓN DEL SOFTWARE, COLABORACIÓN Y EL CONTROL DE VERSIONES

Evolución del software

- Durante el desarrollo
  - El desarrollo del software siempre es progresivo, incluso en el ciclo de vida en cascada
  - El desarrollo evolutivo consiste, precisamente, en una evolución controlada (ciclo de vida espiral, prototipos evolutivos)
- Durante la explotación
  - Durante la fase de mantenimiento se realizan modificaciones sucesivas del producto
- Además de ello, el desarrollo de software se realiza normalmente en equipo, por lo que es necesario integrar varios desarrollos en un solo producto

Control de versiones

- Un control de versiones es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.
- Los sistemas de control de versiones son herramientas de software que ayudan a los equipos de software a gestionar los cambios en el código fuente a lo largo del tiempo.
- A medida que los entornos de desarrollo se aceleran, los sistemas de control de versiones ayudan a los equipos de software a trabajar de forma más rápida e inteligente. Son especialmente útiles para los equipos de DevOps, ya que les ayudan a reducir el tiempo de desarrollo y a aumentar las implementaciones exitosas.

Características

- Mecanismo de almacenamiento de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación...).
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos).
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos (normalmente pudiendo volver o extraer un estado anterior del producto).

Ventajas

- Un completo historial de cambios a largo plazo de todos los archivos.
  - Esto quiere decir todos los cambios realizados por muchas personas a lo largo de los años. Los cambios incluyen la creación y la eliminación de los archivos, así como los cambios de sus contenidos. Que, quien y cuando se realizaron los cambios. Facilita volver a una versión anterior.
- Creación de ramas y fusiones.
  - La creación de una "rama" permite mantener múltiples flujos de trabajo independientes los unos de los otros al tiempo que ofrece la facilidad de volver a fusionar ese trabajo, lo que permite que los desarrolladores verifiquen que los cambios de cada rama no entran en conflicto. Facilita el trabajo colaborativo.
- Trazabilidad.
  - Ser capaz de trazar cada cambio que se hace en el software y conectarlo con un software de gestión de proyectos y seguimiento de errores, además de ser capaz de anotar cada cambio con un mensaje que describa el propósito y el objetivo del cambio, no solo ayuda con el análisis de la causa raíz y la recopilación de información.

Conceptos

- Versión
  - "Versión" es la forma particular que adopta un objeto en un contexto dado.
- Revisión
  - Desde el punto de vista de evolución, es la forma particular de un objeto en un instante dado. Se suele denominar "revisión".
- Configuración
  - Una "configuración" es una combinación de versiones particulares de los componentes (que pueden evolucionar individualmente) que forman un sistema consistente.
  - Desde el punto de vista de evolución, es el conjunto de las versiones de los objetos componentes en un instante dado

Conceptos

- Línea base
  - Llamaremos "línea base" a una configuración operativa del sistema software a partir del cual se pueden realizar cambios subsiguientes.
  - La evolución del sistema puede verse como evolución de la línea base
- Cambio
  - Un cambio representa una modificación específica a un archivo bajo control de versiones. La granularidad de la modificación considerada un cambio varía entre diferentes sistemas de control de versiones.
  - Un "cambio" es el paso de una versión de la línea base a la siguiente
  - Puede incluir modificaciones del contenido de algún componente, y/o modificaciones de la estructura del sistema, añadiendo o eliminando componentes

Conceptos

- Branch:
  - Una “ramificación”, bifurcación o variante es una bifurcación de la línea base u otra ramificación que a partir de ese momento evoluciona y se versiona por separado.
  - Las ramificaciones representan una variación espacial, mientras que las revisiones representan una variación temporal.
- Repositorio
  - El repositorio es el lugar en el que se almacenan los datos actualizados e históricos de cambios, a menudo en un servidor.
  - El repositorio permite ahorrar espacio de almacenamiento, evitando guardar por duplicado elementos comunes a varias versiones o configuraciones

Sistemas de Control de Versiones Manuales

- Un método de control de versiones, usado por muchas personas, es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son ingeniosos).
- Este método es muy común porque es muy sencillo, pero también es tremendamente propenso a errores.
- Es fácil olvidar en qué directorio te encuentras y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.
- Consume mucho espacio al duplicar elementos que no se han modificado entre versiones.

Sistemas de Control de Versiones Centralizados

- El siguiente gran problema con el que se encuentran las personas es que necesitan colaborar con desarrolladores en otros sistemas. Los sistemas de Control de Versiones Centralizados (CVCS por sus siglas en inglés) fueron desarrollados para solucionar varias personas que necesitan colaborar.
- Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.
- Esta configuración ofrece muchas ventajas, especialmente frente a VCS locales, permite trabajar de forma exclusiva: para poder realizar un cambio es necesario comunicar al repositorio el elemento que se desea modificar y el sistema se encargará de impedir que otro usuario pueda modificar dicho elemento, hasta que sea liberado.
- Sin embargo, esta configuración también tiene serias desventajas:
  - Es el punto único de fallo que representa el servidor centralizado.
  - Sin conexión al servidor nadie podrá colaborar o guardar cambios en archivos en los que hayan estado trabajando.

Sistemas de Control de Versiones Distribuidos

- Los sistemas de Control de Versiones Distribuidos o DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio, existe un repositorio local y otro central.
- De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo. Cada clon es realmente una copia completa de todos los datos.
- Se puede trabajar de forma local. Permite operaciones más rápidas.
- Cada usuario modifica la copia local y cuando el usuario decide compartir los cambios el sistema automáticamente intenta combinar las diversas modificaciones. El principal problema es la posible aparición de conflictos que deban ser solucionados manualmente o las posibles inconsistencias que surjan al modificar el mismo fichero por varias personas no coordinadas.

https://git-scm.com/
https://git-scm.com/book/es/v2

GIT

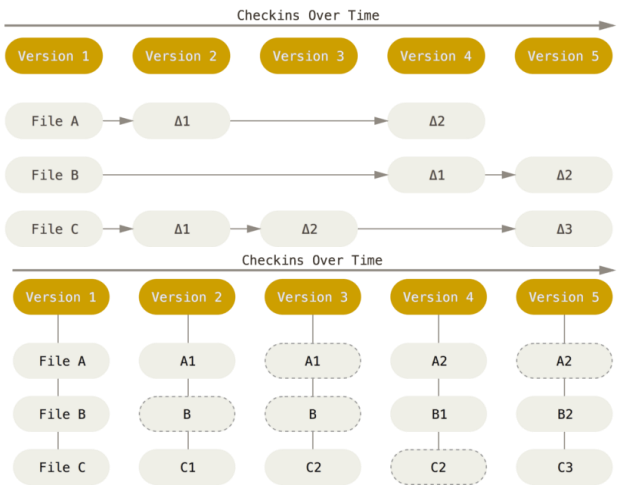
GIT

- Git (pronunciado "guit") es un sistema de control de versiones distribuido de código abierto y gratuito diseñado para manejar todo, desde proyectos pequeños a muy grandes, con velocidad y eficiencia.
- Git es, con diferencia, el sistema de control de versiones moderno más utilizado del mundo. Git es un proyecto de código abierto maduro y con un mantenimiento activo que desarrolló originalmente Linus Torvalds, el famoso creador del kernel del sistema operativo Linux, en 2005.
- Git es fácil de aprender y ocupa poco espacio con un rendimiento increíblemente rápido. Supera a las herramientas SCM como Subversion, CVS, Perforce y ClearCase con características como bifurcaciones locales económicas, áreas de preparación convenientes y múltiples flujos de trabajo. Funciona a la perfección en una amplia variedad de sistemas operativos e IDE (entornos de desarrollo integrados).
- No confundir con GitHub, que es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git, siendo la plataforma más importante de colaboración para proyectos Open Source.
- Instalación:
  - https://git-scm.com/

Fundamentos de Git

- La principal diferencia entre Git y cualquier otro VCS es la forma en la que manejan sus datos. Conceptualmente, la mayoría de los otros sistemas almacenan la información como una lista de cambios en los archivos, manejan la información que almacenan como un conjunto de archivos y las modificaciones hechas a cada uno de ellos a través del tiempo.
- Git maneja los datos como un conjunto de copias instantáneas de un sistema de archivos miniatura. Cada vez que se confirma un cambio o guarda el estado del proyecto en Git, él básicamente toma una foto del aspecto de todos los archivos en ese momento y guarda una referencia a esa copia instantánea. Para ser eficiente, si los archivos no se han modificado Git no almacena el archivo de nuevo, sino un enlace al archivo anterior idéntico que ya tiene almacenado. Git maneja los datos como una secuencia de copias instantáneas.
- La mayoría de las operaciones en Git sólo necesitan archivos y recursos locales para funcionar. Por lo general no se necesita información de ningún otro equipo de tu red.
- Todo en Git es verificado mediante una suma de comprobación (checksum) antes de ser almacenado, y es identificado a partir de ese momento mediante dicha suma. Esto significa que es imposible cambiar los contenidos de cualquier archivo o directorio sin que Git lo detecte.
- Cuando realizas acciones en Git, casi todas ellas sólo añaden información a la base de datos de Git. Es muy difícil hacer algo que no se pueda enmendar.

Instantáneas, no diferencias



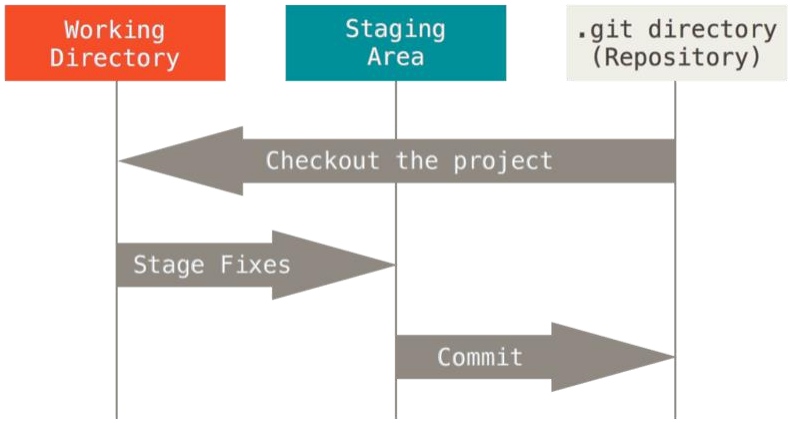
Estados

- Git tiene tres estados principales en los que se pueden encontrar tus archivos:
  - Modificado (modified): significa que has modificado el archivo pero todavía no lo has confirmado a tu base de datos.
  - Preparado (staged): significa que has marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.
  - Confirmado (committed): significa que los datos están almacenados de manera segura en tu base de datos local.
- Esto nos lleva a las tres secciones principales de un proyecto de Git:
  - El directorio de Git (git directory) es donde se almacenan los metadatos y la base de datos de objetos para tu proyecto. Es la parte más importante de Git, y es lo que se copia cuando clonas un repositorio desde un repositorio central.
  - El área de preparación (staging area) es un archivo, generalmente contenido en tu directorio de Git, que almacena información acerca de lo que va a ir en tu próxima confirmación. A veces se le denomina índice ("index"), pero se está convirtiendo en estándar el referirse a ella como el área de preparación.
  - El directorio de trabajo (working directory) es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que los puedas usar o modificar.

Flujo de trabajo

- Git plantea una gran libertad en la forma de trabajar en torno a un proyecto. Sin embargo, para coordinar el trabajo de un grupo de personas en torno a un proyecto es necesario acordar como se va a trabajar con Git. A estos acuerdos se les llama flujo de trabajo. Un flujo de trabajo de Git es una fórmula o una recomendación acerca del uso de Git para realizar trabajo de forma uniforme y productiva. Los flujos de trabajo más populares son git-flow, GitHub-flow y One Flow.
- El flujo de trabajo básico en Git es algo así:
  1. Modificas una serie de archivos en tu directorio de trabajo.
  2. Preparas los archivos, añadiéndolos a tu área de preparación.
  3. Confirmas los cambios, lo que toma los archivos tal y como están en el área de preparación y almacena esa copia instantánea de manera permanente en tu directorio de Git.
  4. Periódicamente, se sincronizan el repositorio local con el repositorio central para que los cambios estén disponibles para el resto de los colaboradores, actuando también como copia de seguridad del repositorio local.
- Si una versión concreta de un archivo está en el directorio de Git, se considera confirmada (committed). Si ha sufrido cambios desde que se obtuvo del repositorio, pero ha sido añadida al área de preparación, está preparada (staged). Y si ha sufrido cambios desde que se obtuvo del repositorio, pero no se ha preparado, está modificada (modified).

Flujo de trabajo



Flujo de trabajo

- Git-Flow (Creado en 2010 por Vincent Driessen)
  - Es el flujo de trabajo más conocido. Está pensado para aquellos proyectos que tienen entregables y ciclos de desarrollo bien definidos. Está basado en dos grandes ramas con infinito tiempo de vida (ramas master y develop) y varias ramas de apoyo, unas orientadas al desarrollo de nuevas funcionalidades (ramas feature-*), otras al arreglo de errores (hotfix-*) y otras orientadas a la preparación de nuevas versiones de producción (ramas release-*).
- GitHub-Flow (Creado en 2011 por el equipo de GitHub)
  - Es la forma de trabajo sugerida por las funcionalidades propias de GitHub. Está centrado en un modelo de desarrollo iterativo y de despliegue constante. Está basado en cuatro principios:
    - Todo lo que está en la rama master está listo para ser puesto en producción
    - Para trabajar en algo nuevo, debes crear una nueva rama a partir de la rama master con un nombre descriptivo. El trabajo se irá integrando sobre esa rama en local y regularmente también a esa rama en el servidor
    - Cuando se necesite ayuda o información o cuando creemos que la rama está lista para integrarla en la rama master, se debe abrir una pull request (solicitud de integración de cambios).
    - Alguien debe revisar y visar los cambios para fusionarlos con la rama master
    - Los cambios integrados se pueden poner en producción.
  - GitHub intenta simplificar la gestión de ramas, trabajando directamente sobre la rama master y generando e integrando las distintas features directamente a esta rama.
- One Flow (Creado en 2015 por Adam Ruka)
  - En él cada nueva versión de producción está basada en la versión previa de producción. La mayor diferencia con Git Flow es que no tiene rama de desarrollo.

Terminología

- commit: un objeto Git, una instantánea de todo su repositorio comprimido en un SHA
- branch (rama): un puntero móvil ligero a una confirmación
- clone: una versión local de un repositorio, incluidas todas las confirmaciones y ramas
- remote: un repositorio común en un servidor central (como GitHub) que todos los miembros del equipo usan para intercambiar sus cambios
- fork: una copia de un repositorio del servidor central propiedad de un usuario diferente
- pull request (solicitud de extracción): un lugar para comparar y discutir las diferencias introducidas en una rama con revisiones, comentarios, pruebas integradas y más
- HEAD: que representa el directorio de trabajo actual, el puntero HEAD se puede mover a diferentes ramas, etiquetas o confirmaciones cuando se usa git checkout

Crear un repositorio Git

- Se puede obtener un proyecto Git de dos maneras: La primera es tomar un proyecto o directorio existente e importarlo en Git. La segunda es clonar un repositorio existente en Git desde otro servidor.
- Inicializando un repositorio en un directorio existente (esto crea un subdirectorio oculto llamado .git que contiene todos los archivos necesarios del repositorio local):
  - git init
- Añadiendo los ficheros al repositorio local:
  - git add .
- Para confirmar los cambios:
  - git commit -m 'initial project version'
- Para vincular el repositorio local con el repositorio central y sincronizar los cambios:
  - git branch -M main
  - git remote add origin https://github.com/myuser/myrepository.git
  - git push -u origin main
- Clonando un repositorio existente:
  - git clone https://github.com/myuser/myrepository localdir

Trabajar con un repositorio Git

- Enumera todos los archivos nuevos o modificados de los cuales se van a guardar cambios
  - git status
- Muestra las diferencias entre archivos que no se han enviado aún al área de espera
  - git diff
- Registra los cambios del archivo permanentemente en el historial de versiones
  - git commit -m"[descriptive message]"
- Sube todos los commits de la rama local al repositorio central
  - git push
- Descarga los cambios del repositorio central a la rama local
  - git pull

Ramificaciones

- Git permite dividir la rama principal de desarrollo (master) y a partir de ahí continuar trabajando de forma independiente: solución de problemas, nueva versión, experimentación … Los cambios en una rama no afectan a las otras ramas.
- Crear una Rama Nueva:
  - git branch hotfix
- Para saltar de una rama a otra se utiliza el comando git checkout.
  - git checkout hotfix
- Los cambios, confirmaciones y otras se realizan sobre la rama actual. Una vez terminados los trabajos en la rama se pueden fusionar con la rama principal (si hay cambios en la principal se pueden producir conflictos que habrá que solucionar):
  - git checkout master
  - git merge hotfix
- Es importante borrar la rama fusionada que ya vamos a necesitar
  - git branch -d hotfix

Solicitud de incorporación de cambios

- Las pull requests, solicitud de incorporación de cambios, son una funcionalidad que facilita la colaboración entre desarrolladores.
- En su forma más sencilla, las solicitudes de incorporación de cambios son un mecanismo para que los desarrolladores notifiquen a los miembros de su equipo que han terminado una función. Una vez la rama de función está lista, el desarrollador realiza la solicitud de incorporación de cambios mediante el repositorio central. Así, todas las personas involucradas saben que deben revisar el código y fusionarlo con la rama principal.
- Pero la solicitud de incorporación de cambios es mucho más que una notificación: es un foro especializado para debatir sobre una función propuesta. Si hay algún problema con los cambios, los miembros del equipo pueden publicar feedback en las solicitudes de incorporación de cambios e incluso modificar la función al enviar confirmaciones de seguimiento. El seguimiento de toda esta actividad se realiza directamente desde la solicitud de incorporación de cambios.
- Cuando realizas una pull request, lo que haces es solicitar que otro desarrollador (el mantenedor del proyecto) incorpore (o haga un pull) una rama de tu repositorio (fork) al suyo.

Merge Conflicts

- Un merge conflict se produce cuando hay discrepancias al sincronizar versiones que entran en conflicto y Git no puede determinar automáticamente qué es correcto:
  - La versión original modificada es distinta de la versión actual.
  - Difieren las versiones en diferentes ramas.
  - Se va a sobre escribir un cambio local pendiente.
- Podemos sacar más información ejecutando el comando git status.
- Git agrega algunas líneas adicionales en el archivo para marcar el conflicto:

```
<<<<<<< HEAD
=======
>>>>>>> commit_name
```

- La línea ======= es el "centro" del conflicto. Todo el contenido entre el centro y la línea <<<<<<< HEAD es contenido que existe en la rama principal actual a la que apunta la referencia HEAD. Por el contrario, todo el contenido entre el centro y >>>>>>> commit_name es contenido que está presente en nuestra rama de fusión.
- La forma más directa de resolver un conflicto de fusión manualmente es editar el archivo conflictivo, eliminar las marcas, dejando una de las dos secciones, las dos o ninguna. Las utilidades de los entornos pueden ayudar a resolver los conflictos.
- Una vez resuelto el conflicto o conflictos, con git add se preparar el nuevo contenido fusionado y con git commit se confirma la solución.

Ignorar Archivos

- A veces, tendrás algún tipo de archivo que no quieres que Git añada automáticamente o más aun, que ni siquiera quieras que aparezca como no rastreado. Este suele ser el caso de archivos generados automáticamente como trazas o archivos creados por el sistema de compilación. En estos casos, se puede crear un archivo de texto llamado .gitignore que liste patrones de los archivos a excluir.

  reame.md
  /node_modules
  doc/**/*.md
- Las reglas usan patrones globales y puedes incluir en el archivo .gitignore las siguientes:
  - Ignorar las líneas en blanco y aquellas que comiencen con #.
  - Emplear patrones glob estándar que se aplicarán recursivamente a todo el directorio del repositorio local.
  - Los patrones pueden comenzar en barra (/) para evitar recursividad.
  - Los patrones pueden terminar en barra (/) para especificar un directorio.
  - Los patrones pueden negarse al añadir al principio el signo de exclamación (!).

Comandos

- Configurar: Configura la información del usuario para todos los respositorios locales
  $ git config --global user.name "[name]"
  Establece el nombre que estará asociado a tus commits
  $ git config --global user.email "[email address]"
  Establece el e-mail que estará asociado a sus commits
- Crear repositorios: Inicializa un nuevo repositorio u obtiene uno de una URL existente
  $ git init [project-name]
  Crea un nuevo repositorio local con el nombre especificado
  $ git remote add origin [url]
  Especifica el repositorio remoto para su repositorio local
  $ git clone [url]
  Descarga un proyecto y toda su historial de versiones
- Suprimir el seguimiento de cambios: Un archivo de texto llamado .gitignore suprime la creación accidental de versiones para archivos y rutas que concuerdan con los patrones especificados
  $ git ls-files --others --ignored --exclude-standard
  Enumera todos los archivos ignorados en este proyecto

Comandos

- Efectuar cambios
  - $ git status
    - Enumera todos los archivos nuevos o modificados de los cuales se van a guardar cambios
  - $ git diff
    - Muestra las diferencias entre archivos que no se han enviado aún al área de espera
  - $ git add [file]
    - Guarda el estado del archivo en preparación para realizar un commit
  - $ git diff --staged
    - Muestra las diferencias del archivo entre el área de espera y la última versión del archivo
  - $ git reset [file]
    - Mueve el archivo del área de espera, pero preserva su contenido
  - $ git commit -m"[descriptive message]"
    - Registra los cambios del archivo permanentemente en el historial de versiones
  - $ git rm [file]
    - Borra el archivo del directorio activo y lo pone en el área de espera en un estado de eliminación
  - $ git rm --cached [file]
    - Retira el archivo del historial de control de versiones, pero preserva el archivo a nivel local
  - $ git mv [file-original] [file-renamed]
    - Cambia el nombre del archivo y lo prepara para ser guardado

Comandos

- Sincronizar cambios: Registrar un marcador para un repositorio e intercambiar historial de versiones
  - $ git fetch [bookmark]
    - Descarga todo el historial del marcador del repositorio
  - $ git merge [bookmark]/[branch]
    - Combina la rama del marcador con la rama local actual
  - $ git push [alias] [branch]
    - Sube todos los commits de la rama local al repositorio central
  - $ git pull
    - Descarga el historial del marcador e incorpora cambios
- Branches (cambios grupales): Nombra una serie de commits y combina esfuerzos ya completados
  - $ git branch
    - Enumera todas las ramas en el repositorio actual
  - $ git branch [branch-name]
    - Crea una nueva rama
  - $ git checkout [branch-name]
    - Cambia a la rama especificada y actualiza el directorio activo
  - $ git merge [branch-name]
    - Combina el historial de la rama especificada con la rama actual
  - $ git branch -d [branch-name]
    - Borra la rama especificada

Comandos

- Repasar historial: Navega e inspecciona la evolución de los archivos de proyecto
  - $ git log
    - Enumera el historial de versiones para la rama actual
  - $ git log --follow [file]
    - Enumera el historial de versiones para el archivo, incluidos los cambios de nombre
  - $ git diff [first-branch]...[second-branch]
    - Muestra las diferencias de contenido entre dos ramas
  - $ git show [commit]
    - Produce metadatos y cambios de contenido del commit especificado
- Rehacer commits: Borra errores y elabora un historial de reemplazo
  - $ git reset [commit]
    - Deshace todos los commits después de [commit], preservando los cambios localmente
  - $ git reset --hard [commit]
    - Desecha todo el historial y regresa al commit especificado

Comandos

- Guardar fragmentos: Almacena y restaura cambios incompletos
  - $ git stash
    - Almacena temporalmente todos los archivos modificados de los cuales se tiene al menos una versión guardada
  - $ git stash pop
    - Restaura los archivos guardados más recientemente
  - $ git stash list
    - Enumera todos los grupos de cambios que están guardados temporalmente
  - $ git stash drop
    - Elimina el grupo de cambios más reciente que se encuentra guardado temporalmente

GitHub

- echo "# Mi repositorio" >> README.md
- git init
- git add .
- git commit -m "initial commit"
- git remote add origin https://github.com/myuser/myrepository.git
- git branch -M main
- git push -u origin main

- https://github.com/github/gitignore