NEORIS

Índice

- Reglamento
- Presentación
- Objetivo del curso
- **Temario**
    - Que es Agile Testing?
    - Quien es el Tester Agile?
    - Cuadrante de Testing
    - Prácticas y técnicas de Agile Testing
    - Gestión de defectos
    - Pruebas de regresión
    - Testing exploratorio
    - Automatización de pruebas
    - TDD (Test Driven Development)
    - ATDD (Acceptance Test Driven Development)
    - BDD (Behavior Driven Development)

NEORIS

Reglamento

- Puntualidad para iniciar y terminar sesión.
- Intégrate en silencio, para evitar interrupciones y mantén el micrófono cerrado.
- Escuchar atenta y respetuosamente.
- Alta Participación, se realizarán actividades.
- Externa tus dudas de manera ordenada, mediante la solicitud de turno para hablar de la herramienta de la sesión o envía tus preguntas por chat y las iremos contestado por tema cubierto.
- Abrir cámara cuando sea solicitado y se encuentren participando de manera directa.
- Las sesiones serán grabadas para que puedan ser consultadas posteriormente.
- Al finalizar las sesiones les compartiremos un examen y encuesta, ésta última para que nos den sus comentarios y mejorar las siguientes sesiones.

Presentación

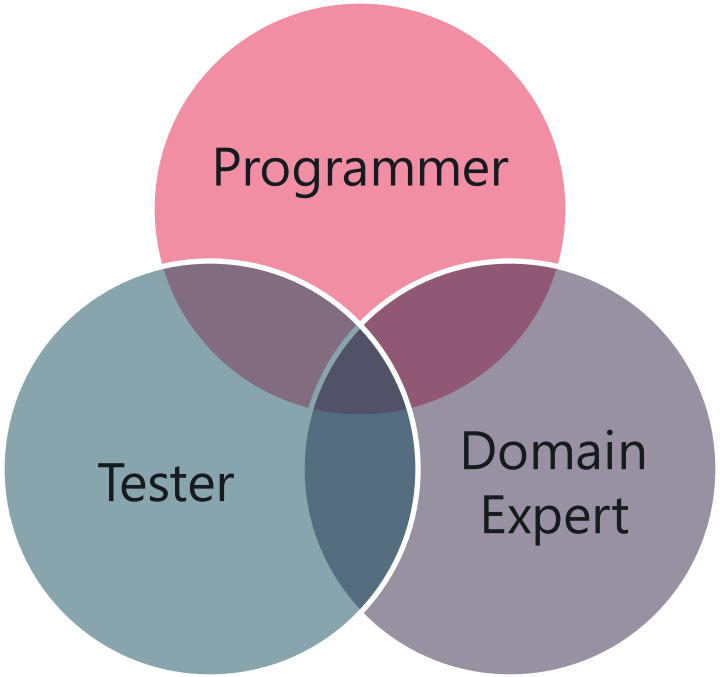
NEORIS

Objetivos del curso

- Comprender los principios ágiles de Testing de software
- Planificar, desarrollar y ejecutar pruebas ágiles
- Diferenciar el papel de las pruebas en proyectos ágiles y no ágiles
- Participar de forma positiva como un miembro mas del equipo ágil centrado en las pruebas
- Comprender los retos y dificultades asociadas a las diferentes actividades no relacionadas con las pruebas llevadas a cabo por el equipo
- Habilidades necesarias para los miembros de un equipo ágil

¿Qué es Agile Testing?

Testing desde una visión ágil:



1.Sigue los principios del desarrollo ágil de software.

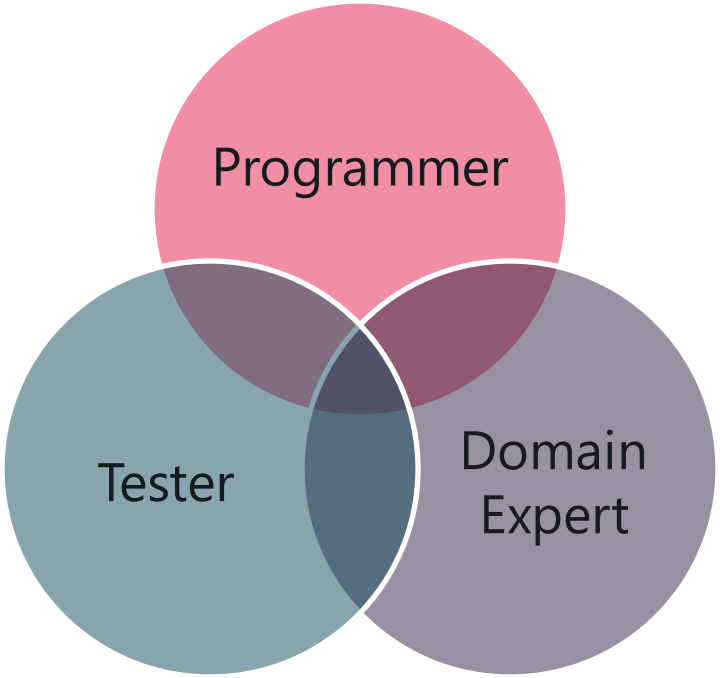
2.Involucra a todos los miembros de un Equipo Ágil Multifuncional.

3.Las metodologías ágiles no ven al software testing como una fase separada.

NEORIS

¿Qué es Agile Testing?

Testing desde una visión ágil:



1. Agile Testing, incorpora una serie de prácticas

- Testing de "todo el equipo".
- Testing independiente (opcional).
- Integración continua.
- Desarrollo guiado por pruebas (TDD).
- Desarrollo guiado por comportamiento (BDD).
- Desarrollo guiado por pruebas de aceptación (ATDD), entre otros.

NEORIS

Agile Testing Manifiesto

Testing durante **SOBRE** Testing al final.

Prevenir bugs **SOBRE** encontrar bugs.

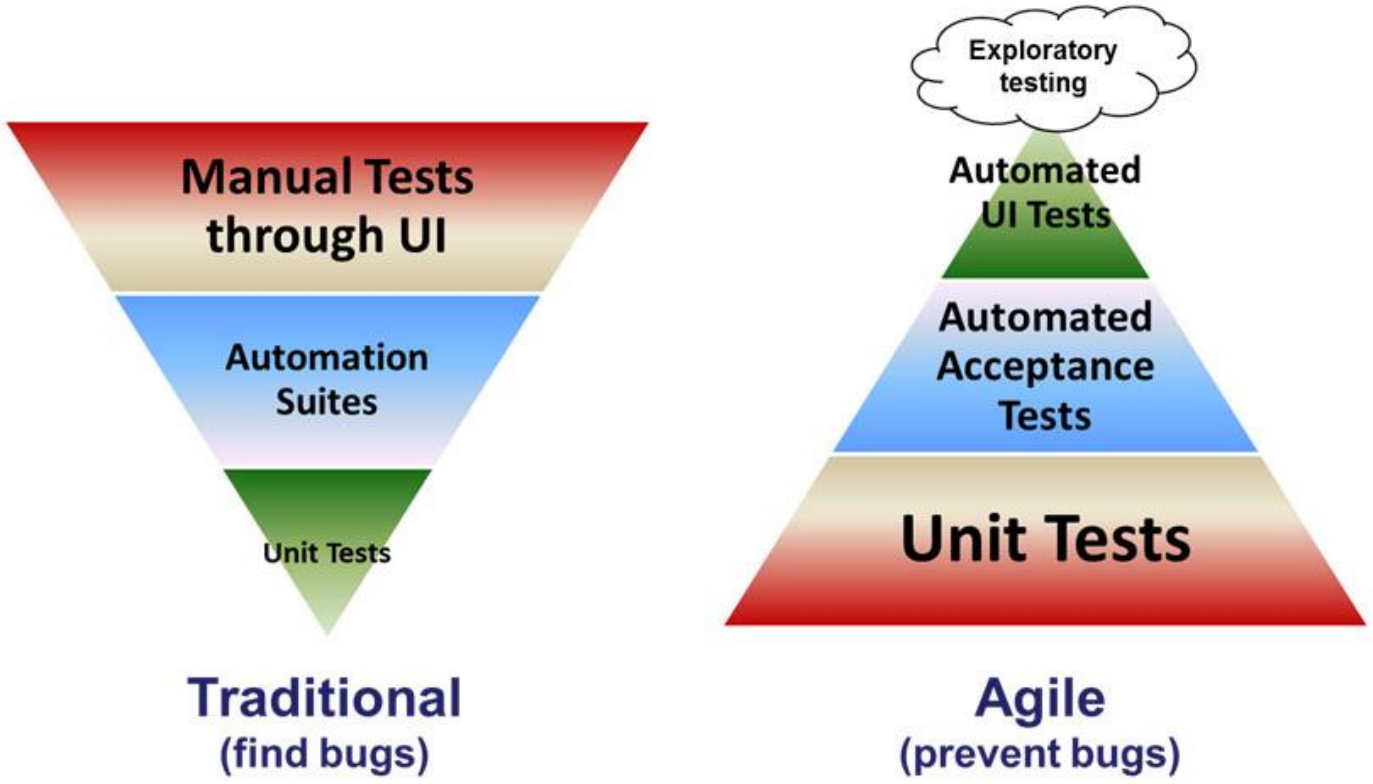
Entender lo que se está probando **SOBRE** verificar funcionalidad.

Construir el mejor sistema **SOBRE** romper el sistema.

El equipo es responsable de la calidad **SOBRE** el Tester es responsable de la calidad.

NEORIS

Testing Tradicional vs Agile Testing



NEORIS

Los 10 Principios de Agiles Testing

- Provee feedback de forma continua
- Entrega valor al cliente
- Posibilita la comunicación cara a cara
- Ten coraje
- Mantenlo simple
- Practica mejora continua
- Responde al cambio
- Auto-organízate
- Céntrate en las personas
- Disfruta

¿Quién es el Tester Ágil?

Un Tester ágil debe seguir el manifiesto ágil(individuos e interacciones, software funcionando, colaboración con el cliente y respuesta al cambio).

El Testing ágil se centra en añadir valor al negocio y en entregar la calidad que el cliente solicita, diferenciándose así del Testing tradicional, centrado en cumplir unos requisitos.

En caso de duda volver a los valores y principios ágiles.

NEORIS

Rol del Tester Ágil

- Es el de un experto, que garantice la entrega con valor de negocio deseado por el cliente a un ritmo sostenible y continuo.
- Para ello, utiliza la "especificación mediante ejemplos" para capturar los comportamientos deseados y no deseados para guiar la codificación.
- En un entorno ágil el rol de Tester es el de mayor especialización, considerando que él debe manejar herramientas de automatización, gestión ágil y metodologías.
- Además, el rol también posee mayor interacción con otras personas como por ejemplo el cliente o los desarrolladores, por lo que también necesitará habilidades soft de comunicación, orientación al cliente, negociación, entre otras.

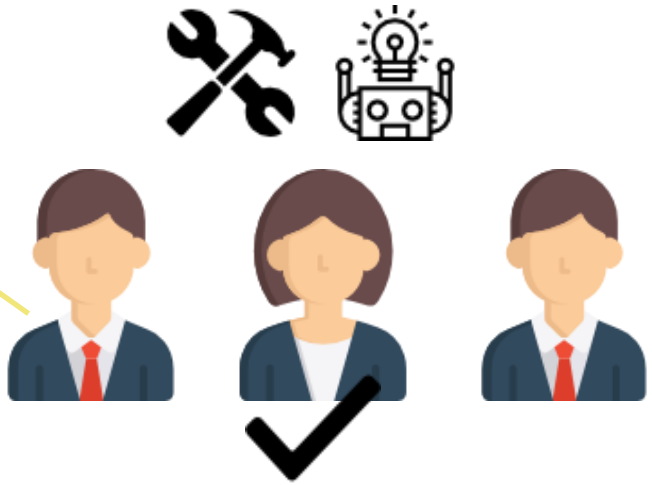
Responsabilidades del Tester Ágil

- Darle prioridad a una aplicación funcionando a una aplicación documentada **"Código que funciona"**
- Comunicarse a menudo.
- Entender todos los criterios de aceptación.
- Colaborar con los desarrolladores y con el Product Owner (PO) de cerca.
- Aplicar las pruebas los más tempranas posibles.
- El tester debe empezar a escribir las pruebas tan pronto como la planificación de sprint se termina.
- Evaluar el costo-beneficio en la automatización.
- Aplicar testing exploratorio.

Características de un Tester Ágil

Auto Organizado

Multi-Disciplinario



De tiempo completo

Encontrarse en el mismo lugar

Estable

Actividad 1

Instrucciones:

- Navega el Código QR que se muestra a continuación o en un navegador captura https://kahoot.it/

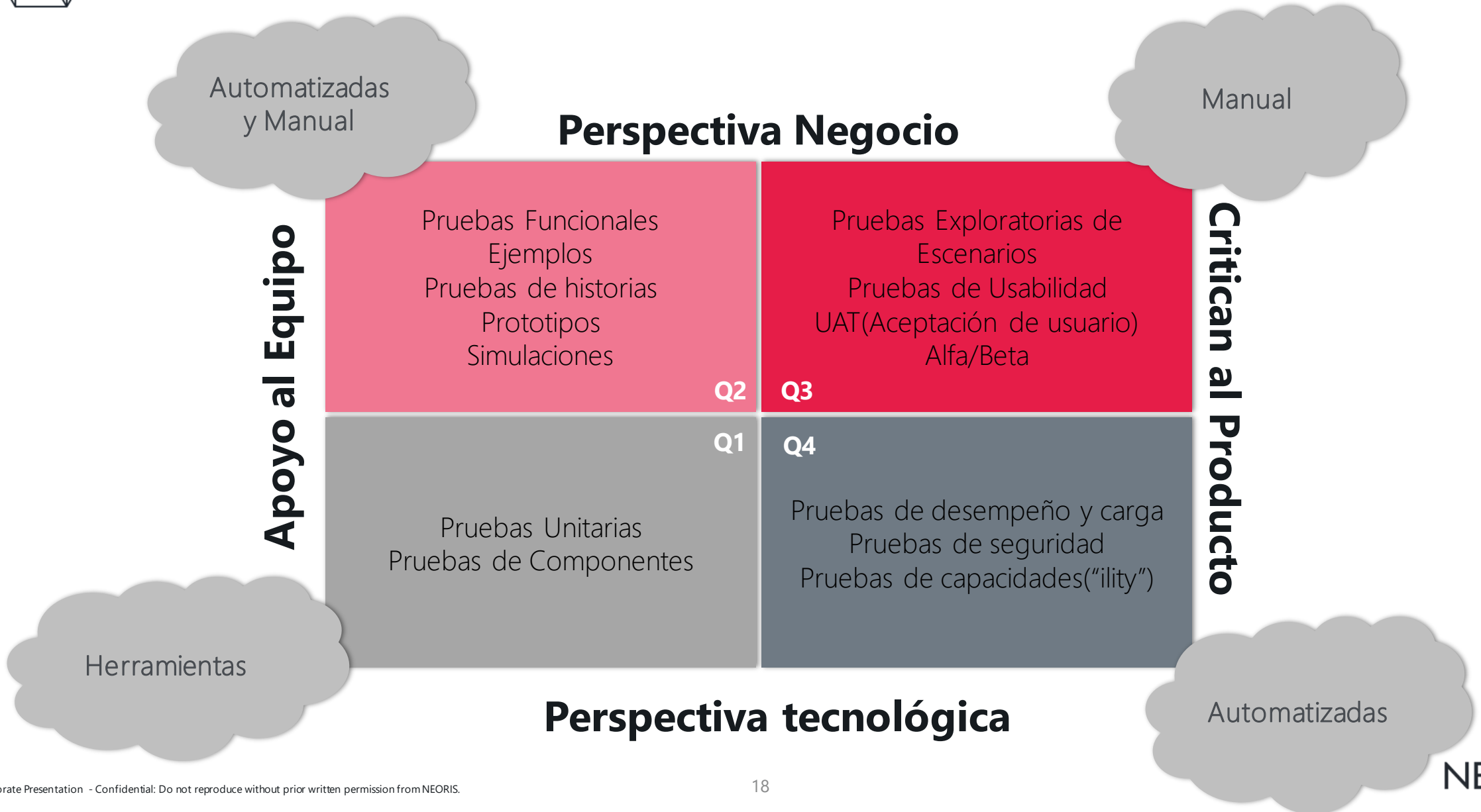
QR

- Capturar el Game PIN: **9093969**

- Una vez entrando, no proporciones datos personales: Nombre completo, fecha de nacimiento, correo electrónico, teléfono, RFC, Nombre de la compañía, datos de tarjetas, puedes utilizar un nickname

NEORIS

Planificación “Cuadrante de Testing”

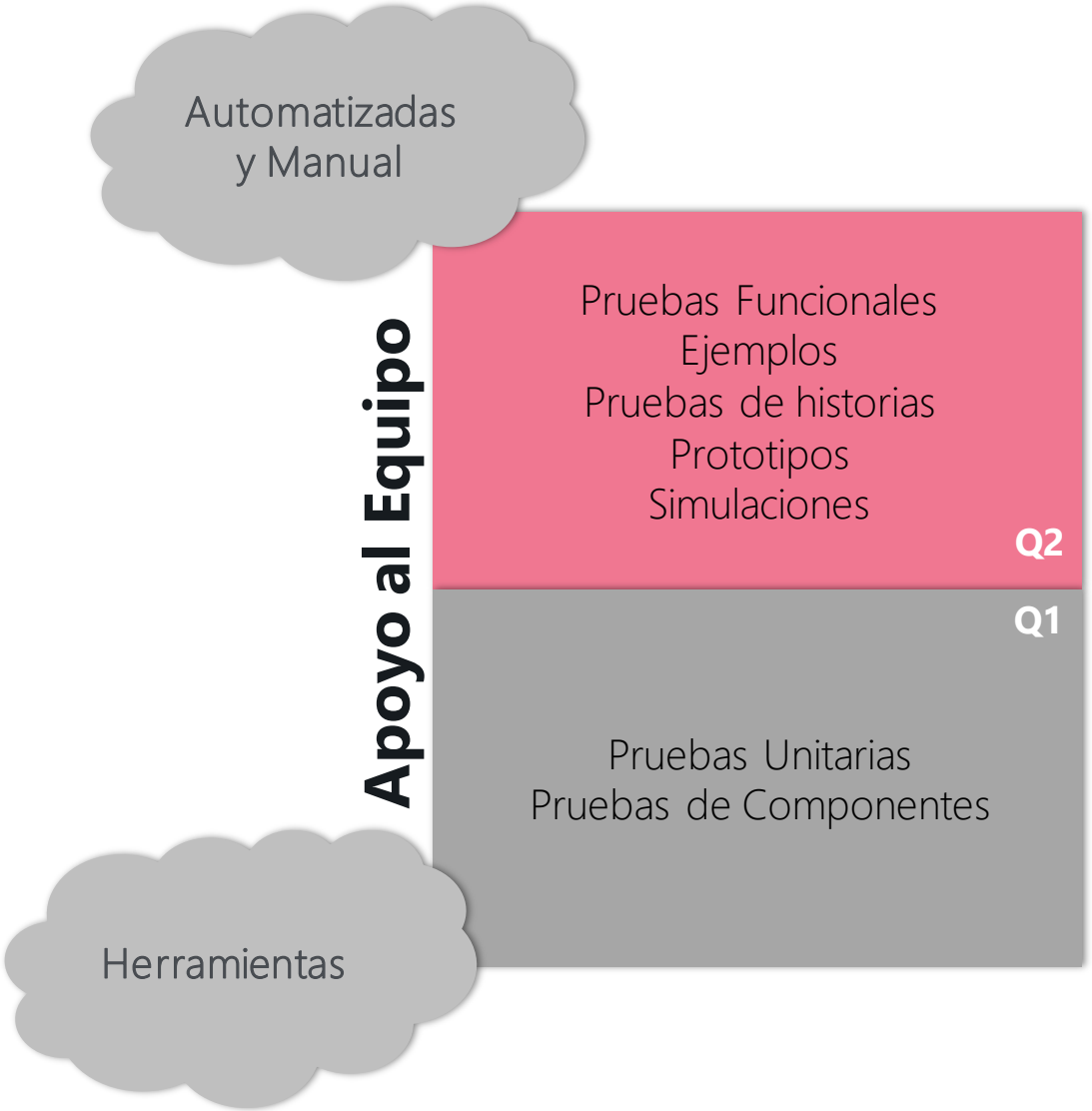


NEORIS

Pruebas de Apoyo al Equipo(Supporting the Team)

- Destinadas a apoyar al equipo de desarrollo en la medida en que este desarrolla el producto.

- Estas pruebas primeramente guían el desarrollo de la funcionalidad, y luego cuando se automatizan sirven para apoyar la **refactorización** y la inclusión de un nuevo código sin causar resultados inesperados en el comportamiento del sistema.



NEORIS

Cuadrante 1:
Pruebas de Apoyo al Equipo de Cara a la Tecnología

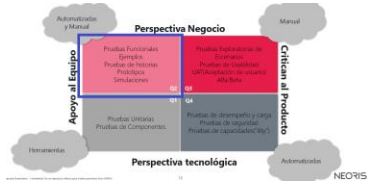
•También se le suele llamar prueba de programador, pruebas de cara a tecnología. Suelen automatizarse usando alguna herramienta de la familia de xUnit, en el mismo lenguaje de programación usado para fabricar la aplicación.

•Estas son pruebas de calidad interna. La calidad interna no es definida por el cliente(son de naturaleza técnica) sino por los programadores para logar cumplir los requerimientos funcionales y no funcionales solicitados.

•Las pruebas unitarias y de componentes pasarán a formar parte del proceso de los Check-in ejecutado cada vez que se incluyen cambios en el código, otorgando así feedback inmediato.

NEORIS

Cuadrante 2:
Pruebas de Apoyo al Equipo de Cara al Negocio.



Las pruebas de cuadrante 2(funcionales, ejemplos, pruebas, de historias, etc.)

•Se derivan de ejemplos que suministra el equipo del cliente.

•Describen los detalles de cada historia de usuario.

•Se ejecutan a un nivel funcional y verifican condiciones de satisfacción definidas por el negocio.

•Se escriben usando un lenguaje que los expertos en el negocio(no necesariamente informáticos de profesión)puedan entender. De hecho los expertos del negocio usan estas pruebas en las definiciones de calidad externa y participan en su elaboración.

•Pueden duplicar pruebas ya realizadas en el cuadrante 1, pero a un nivel superior(nivel funcional).

•Están orientadas a ilustrar y confirmar el comportamiento deseado del sistema.

NEORIS

Cuadrante 2:
Automatización de Pruebas del Cuadrante 2.

Las pruebas del cuadrante 2 también pueden ser automatizadas.

- La automatización permite identificar y solucionar errores rápidamente.
- Deben ejecutarse con frecuencia para dar alerta temprana al equipo
- Si es posible, es recomendable ejecutarlas directamente en la capa de lógica de negocio.
- Aún así algunas de estas pruebas deben ser ejecutadas desde la perspectiva del cliente, es decir desde la interfaz de usuario.

NEORIS

Cuadrante 2:
Pruebas de Prototipo de Interfaces con el Usuario

•Las pruebas destinadas a validar la interfaz de usuario propuestas(prototipos) también pertenecen a ese cuadrante.

•En estas pruebas, los encargados de diseñar la interfaz con el usuario elaboraran las pantallas de ejemplos o wireframes, los cuáles validan con el cliente(usuario) antes que comience el desarrollo.

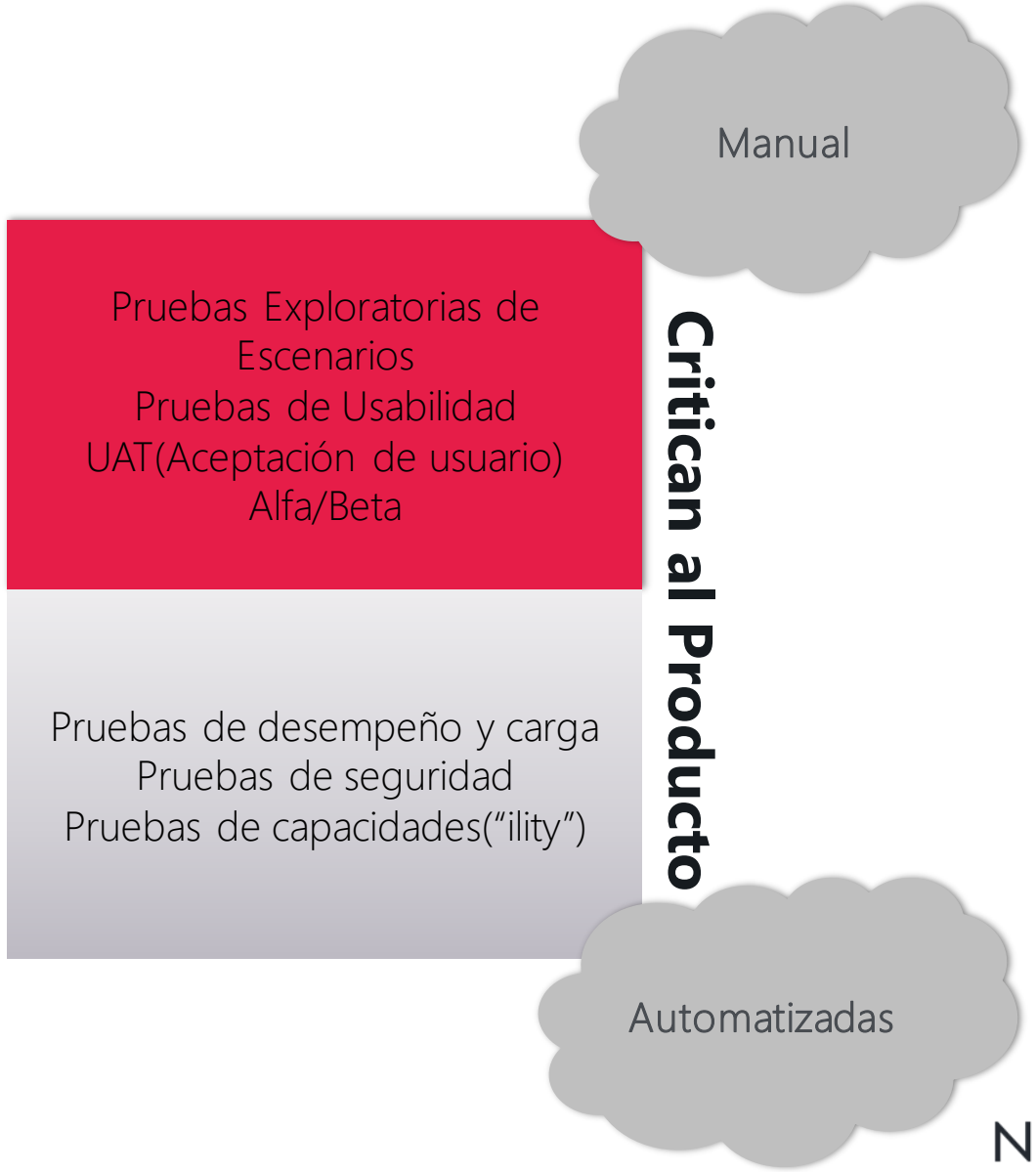
•Una vez que comienza el desarrollo, los testers son los encargados de comunicar a los desarrolladores de software esos diseños y resolver sus dudas que puedan presentarse.

NEORIS

Pruebas que Critican al Producto(Critique the Product)

Cuando nos referimos a "críticas de software", no tienen necesariamente un sentido negativo, pues éstas pueden ser inclusive para resaltar aspectos positivos o sugerir mejoras.

Para un cliente es muy difícil saber de antemano lo que quiere hasta verlo plasmado en un producto de características mínimas, esta es una realidad con toda metodología de desarrollo de software debe lidiar.



NEORIS

Cuadrante 3:
Pruebas que Critican el Producto de Cara al Negocio

•En el cuadrante 3, las pruebas se ejecutan en su conjunto(es una prueba integral) para determinar si cumple o no las expectativas.

•Cuando se realizan estas pruebas, se trata de emular lo mas posible el entorno real en que será ejecutadas por lo tanto, son pruebas funcionales manuales y sólo las pueden ejecutar personas(no máquinas)

NEORIS

Cuadrante 3:
Pruebas que Critican el Producto de Cara al Negocio

•Se puede recurrir al algún grado de automatización por ejemplo para preparar los datos de prueba, sin embargo al ejecutarla debe utilizarse la intuición para saber si el producto cumple con las expectativas y proporciona valor al área de negocio.

•Usualmente estas pruebas son ejecutadas por el equipo de cliente(usuarios finales), en la forma de pruebas de aceptación(UAT).

•Las pruebas de usabilidad forman parte de éste cuadrante y representan en si mismas todo un campo de estudio.

•En éste cuadrante , el Testing exploratorio es un aspecto central, en el Testing exploratorio, el tester diseña y ejecuta la prueba al mismo tiempo. Ambos procesos se retroalimentan, es decir la ejecución de la prueba y análisis críticos de los resultados, se aprende más sobre el negocio y la aplicación, se rediseña la prueba y se repite el proceso.

NEORIS

Cuadrante 4: Pruebas que Critican el Producto

•Las pruebas del cuadrante 4 son enteramente de naturaleza técnica (no funcional) y son tan críticas para el Agile Testing como cualquier metodología de desarrollo de sistemas.

•La intención es analizar y someter a pruebas de extremo a características no funcionales como el desempeño o la robustez.

•Las pruebas del cuadrante 4 tienen mucho que ver con validar que se cumplan los requerimientos no funcionales.

NEORIS

Uso de los 4 Cuadrantes: Importante

•Los 4 cuadrantes son una taxonomía para ayudar en la planificación del agile Testing.

•La idea de usarlos es asegurar que se tomen en cuenta todos los recursos y métodos necesarios para logar la producción de software de calidad.

•Los 4 cuadrantes no son reglas rígidas ni en su contenido ni orden, cada equipo de desarrollo de software debe adaptarlos a su situación particular con amplia libertad.

•Los 4 cuadrantes no deben ejecutarse en orden del 1-4,pues eso sería aplicar una metodología predictiva(En cascada) y no Agile.

•La numeración de los cuadrantes es arbitrario y es para que, por ejemplo un interlocutor pueda referirse al "Cuadrante 1" en lugar de "pruebas de tecnología de cara a cliente".

NEORIS

Actividad 3 - Cuadrantes

En una dinámica grupal, completar los cuadrantes y explicarlo:

Perspectiva del negocio

Manules y Automatización

Manuales

Apoyo al equipo

Funcionales, Automatizacion,, Pruebas de Historias Simulaciones., proto

orias, UAT,Integrales

Critican

Pruebas Unitaria componentes

dad, desempeño, performance, seguridad

Herramientas

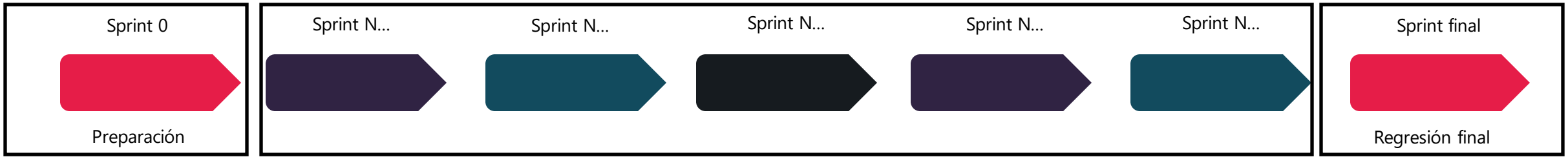
Automatizadas

Perspectiva Tecnológico

NEORIS

Modelo

El modelo de trabajo de testing en ágil, se puede visualizar en 3 grandes segmentos:
- **Sprint 0**
- **Sprints Intermedios**
- **Sprint Final**

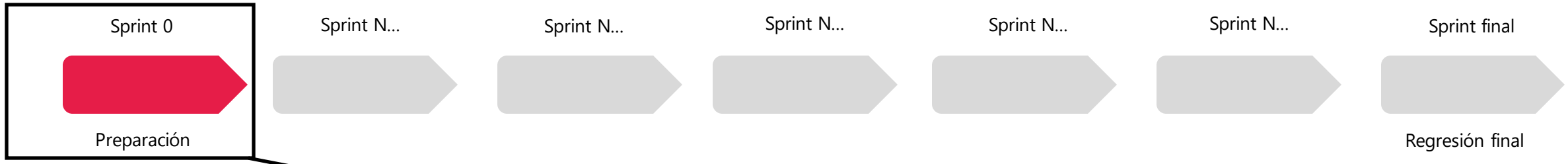


FUNDAMENTOS

- Control y seguimiento
- Perfiles
- Practicas metodológicas
- Evaluación de pruebas automatizadas
- Pair Testing
- Tipos de pruebas
- Ceremonias agiles
- Evaluación de pruebas rendimiento
- Niveles de pruebas
- Técnicas de diseño casos de prueba
- Herramientas

NEORIS

Modelo



•Líder de pruebas
•Arquitecto de pruebas

Actividades

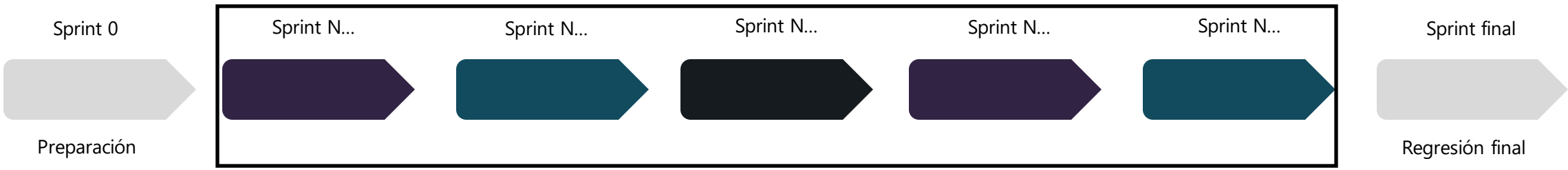
•Se valida alcance de proyecto: Historias, tipos y herramientas de pruebas
•Se participa en el refinamiento de historias con PO y Usuario Final
•Se detecta "testeabilidad" de requerimientos
•Realizar estimación de esfuerzo (conteo de casos de prueba funcionales)
•Recibir información técnica del proyecto (arquitectura, estrategia de desarrollo Vertical/horizontal, secuencia de liberación de funcionalidades)
•Acordar el nivel de encapsulamiento de desarrollo y entrega de funcionalidades
•Se realiza un análisis de necesidades de pruebas de rendimiento por componente a construir, y se planea la entrada del equipo de Pruebas de rendimiento en el sprint identificado
•Se acuerda el modelo de pruebas a seguir en el equipo de trabajo considerando tipos y niveles de pruebas.
•Participar en el sprint planning
•Integrar estimación y planeación de esfuerzo de testing en el proyecto utilizando el modelo "ágil"
•Construir matriz y casos de pruebas del siguiente sprint
•Se identifica requerimientos de prueba y fechas para contar con el licenciamiento e infraestructura necesaria.

Entregables

•Master test strategy
•Documento de análisis de riesgos de proyecto y producto
•Documento de huecos y dudas funcionales de historias recibidas
•Estimación de pruebas
•Plan de trabajo del Sprint siguiente
•Modelo de pruebas ajustado
•Matriz de pruebas del siguiente sprint
•Requerimientos de pruebas (hardware y software)

NEORIS

Modelo



•Líder de pruebas
•Tester Funcional /automatizador
•Tester Performance (si aplica)
•Arquitecto de pruebas (si aplica)

•Testing Funcional
  - •Se planea a detalle el sprint con : Diseño de Matriz, Pair Testing, Ejecución de Pruebas, Pre Demo con PO, Pre demo con Usuario, Regresión, Demo
  - •Se realizan actividades: Diseño de Matriz, Pair Testing, Ejecución de Pruebas, Pre Demo con PO, Pre demo con Usuario, Regresión, Demo
  - •Nota: El back log del alcance es : Historias construidas en el sprint, ajustes aprobados de demos de sprint anteriores, correcciones no entregadas en sprint anteriores y por corregir en este sprint

•Testing Automatizado (solo si aplica este tipo de pruebas como parte del esfuerzo acordado con el cliente)
  - •Se evalúa la madurez del Modelo integral ágil para confirmar si es factible iniciar a automatizar
  - •Si la evaluación resulta positiva, se realiza análisis de selección de casos de prueba de regresión a automatizar (de sprint anteriores)
  - •Se construye la automatización
  - •Se integra la ejecución de la automatización a la estrategia de testing Funcional del sprint

•Testing de Rendimiento (solo si aplica este tipo de pruebas como parte del esfuerzo acordado con el cliente)
  - •Se confirma que en el sprint siguiente se construirán componentes con necesidades de pruebas de rendimiento
  - •Si se confirma, se gestiona la integración del equipo de pruebas de rendimiento y la solicitud de hardware y software necesario
  - •En el sprint siguiente, se realiza el análisis y ejecución de pruebas de rendimiento por componente

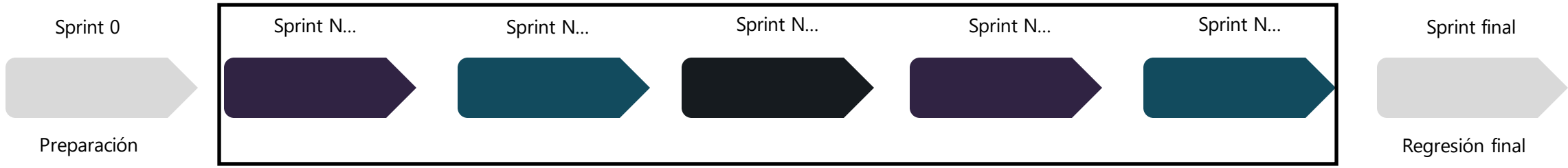
Entregables

•Test Plan del siguiente sprint
•Requerimiento del sprint
•Documento de huecos y dudas funcionales de historias recibidas del siguiente sprint
•Estimación de pruebas siguiente sprint
•Plan de trabajo del Sprint siguiente
•Matriz de pruebas s de pruebas (hardware y software) habilitados (cuando aplica)
•Resultados de pruebas, defectos, backlog de pruebas ajustado
•Casos de prueba automatizados y ejecutados (cuando aplica)
•Diseño de carga para pruebas de estrés (cuando aplica)
•Scripts automatizados y ejecutados de pruebas de estrés (cuando aplica)

NEORIS

Modelo

Ejemplo de sprint de 15 días



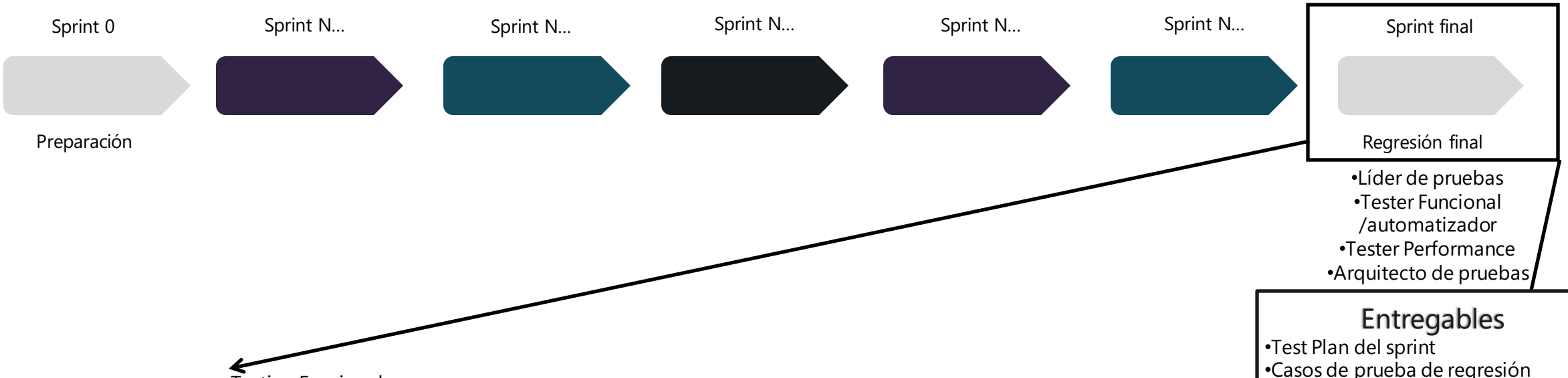
	Historia	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
Vista integral del esfuerzo	1	Construcción			Pair Testing		Pair Testing	Predemo Interna						Regresion		Demo
	2		Construcción				Pair Testing		Pair Testing	Predemo Interna						
	3		Construcción			Pair Testing	Predemo Interna									
	4					Construcción			Pair Testing		Pair Testing	Predemo Interna				
	5								Construcción				Pair Testing			
Actividades de Preparación Testing Funcional	1	Historia	Análisis													
	2			Historia	Análiis											
	3	Historia	Análiis													
	4				Historia	Anaáiis										
	5								Historia	Análiis						

- Pair Testing
- Automatización
- Rendimiento
- Perfiles

Construcción	Construir historia (s), con un sentido de negocio funcional de cara al cliente (Dev, UX/UI, Análisis, DBA, Arquitectura)
Pair Testing	Ejecución de pruebas en el ambiente del desarrollador, Desarrollador + Tester al mismo tiempo, se prueba basado en la matriz de pruebas, se debuquea y se corrige en el momento a menos que haya que planear una segunda sesión para poder corregir todos los errores. Tiene que tener un sentido funcional de cara al cliente, mencionado en "Construcción"
Pair Testing	Ejecución de pruebas en el ambiente del desarrollador, Desarrollador + Tester al mismo tiempo, se valida que ya no existen defectos
Predemo Interna	PO, verifica basado en la matriz de pruebas que se cumple con el requerimiento. Tiene que tener un sentido funcional de cara al cliente, mencionado en "Construcción"
Regresión	Se realiza regresión final para identificar impactos en los desarrollos posteriores al Pair testing
Demo	Se realiza Demo: Se prepara y se aclara: Alcance, Funcionalidades a Revisar, Escenarios por funcionalidad, defectos corregidos (si aplica), de sprints anteriores, Cambios realizados de sprints anteriores (si aplica). Testing documenta resultados, cambios, y defectos por corregir.
Historia	Testing analiza con detalle la historia junto con el analista y PO, usando técnica de grafos, detecta ambigüedades, inconsistencias, las resuelve con el analista y PO
Análisis	Se genera matriz de pruebas, se valida con el PO y Usuario , se aclara que este será el alcance de la DEMO

NEORIS

Modelo



•Testing Funcional
  - •Se realiza la represión final de la aplicación, asegurando la integración de todas las funcionalidades construidas anteriormente

•Testing Automatizado (solo si aplica este tipo de pruebas como parte del esfuerzo acordado con el cliente)
  - •Se integra la ejecución de la automatización de los sprints pasados a la estrategia de testing Funcional de la regresión final.

•Testing de Rendimiento (solo si aplica este tipo de pruebas como parte del esfuerzo acordado con el cliente)
  - •En el sprint siguiente, se realiza el análisis y ejecución de pruebas de rendimiento final

Entregables

•Test Plan del sprint
•Casos de prueba de regresión seleccionados y ejecutados
•Resultados de pruebas, defectos, backlog de pruebas ajustado
•Casos de prueba automatizados y ejecutados (cuando aplica)
•Diseño de carga para pruebas de estrés (cuando aplica)
•Scripts automatizados y ejecutados de pruebas de estrés (cuando aplica)

NEORIS

Actividad 4 - Modelo

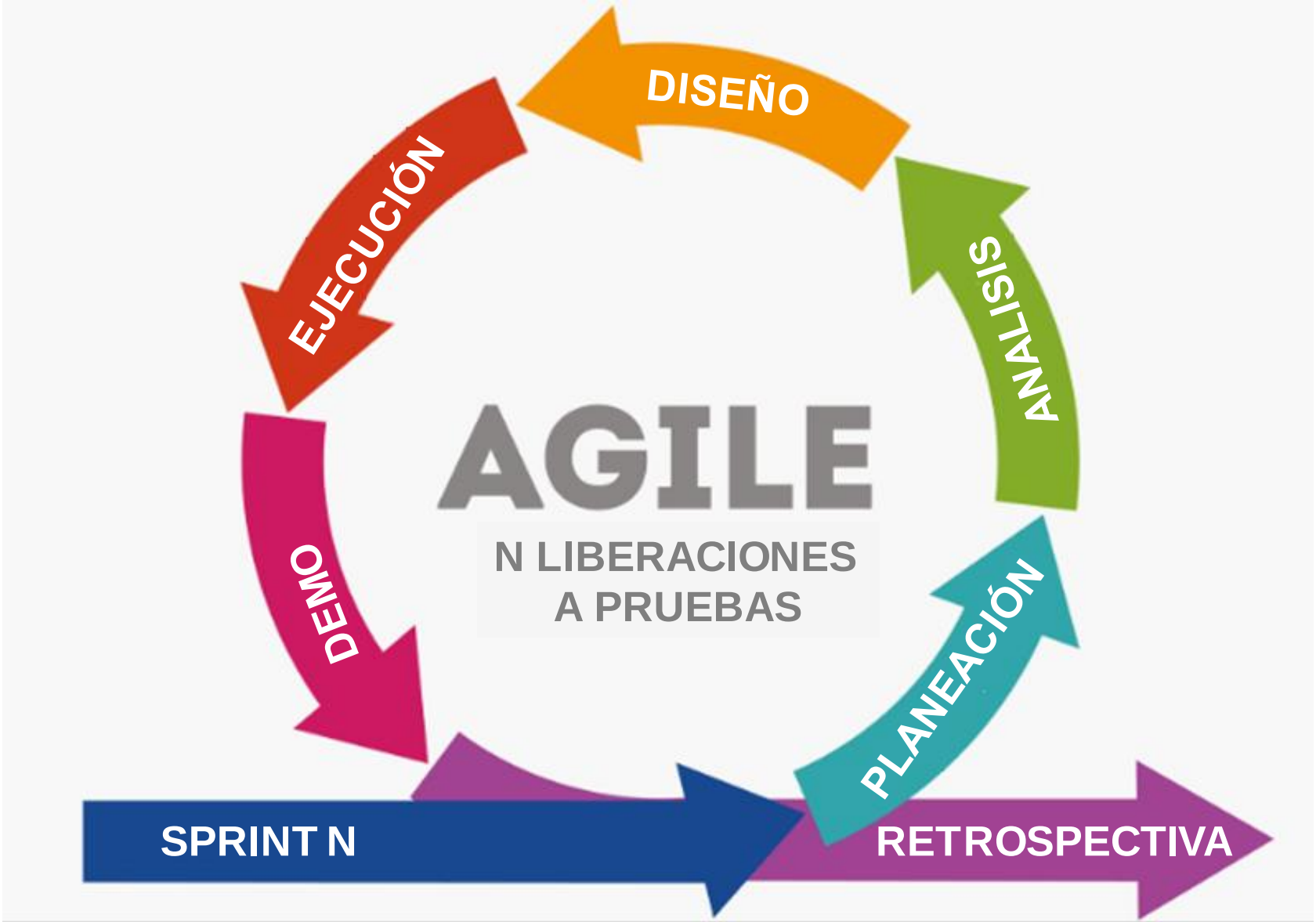
Equipos: 3 Equipos

Indicaciones:

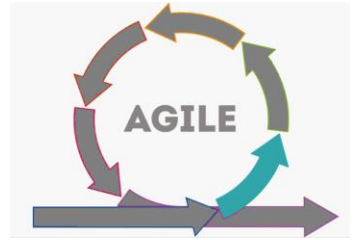
1. Se plantea un proyecto de Testing por el equipo.
2. Un equipo define las actividades del Sprint 0
3. El equipo 2 define las actividades del Sprint 1 a N
4. El equipo 3 define las actividades del Sprint final
5. Cada equipo explica lo que definieron y se revisa con el equipo que falto y que temas no aplican.

NEORIS

Proceso de Testing ÁGIL



Planificación del Sprint: Definición



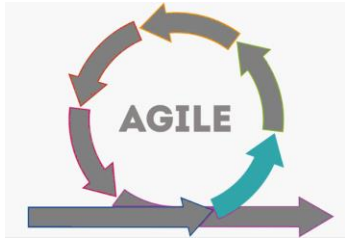
•En esta reunión se define la funcionalidad en el incremento planeado y cómo el Equipo de Desarrollo creará este incremento; la finalidad de este trabajo es definir el Objetivo del Sprint.

•La reunión de planificación de Sprint tradicionalmente consta de dos partes, cada una de la mitad de tiempo de duración de la Reunión de Planificación respondiendo a las siguientes dos preguntas:

1.¿Qué va a ser entregado en el incremento resultante del próximo Sprint?

2.¿Cómo se va a realizar el trabajo seleccionado?

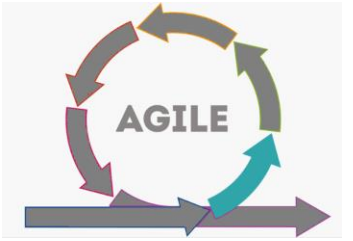
Planificación del Sprint: Qué?



•El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto, pone nombre a la meta de la iteración (de manera que ayude a tomar decisiones durante su ejecución) y propone los requisitos más prioritarios a desarrollar en ella.

•El equipo examina la lista, pregunta al cliente las dudas que le surgen, añade más condiciones de satisfacción y selecciona los objetivos/requisitos más prioritarios que prevé completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita

Planificación del Sprint: Cómo?



•El equipo planifica la iteración, elabora la estrategia que le permitirá conseguir el mejor resultado posible con el mínimo esfuerzo.

•Define las tareas necesarias para poder completar cada objetivo/requisito, creando la lista de tareas de la iteración (Sprint Backlog) basándose en la definición de hecho.

•Realiza una estimación conjunta del esfuerzo necesario para realizar cada tarea.

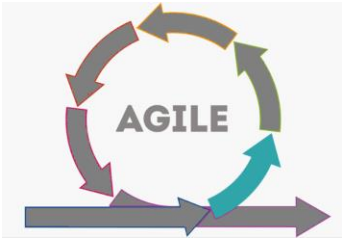
•Los miembros del equipo se auto-asignan las tareas que pueden realizar, se auto-organizan para trabajar incluso en parejas (o grupos mayores) con el fin de compartir conocimiento (creando un equipo más resiliente) o para resolver juntos objetivos especialmente complejos.

•Realizar la estrategia de pruebas cubriendo los criterios de aceptación y el alcance(listado de todas las tareas que se pretenden hacer durante el desarrollo de un proyecto).

•Realizar en conjunto del equipo la planeación de las actividades del Sprint y en específico las de Pruebas.

NEORIS

Estimación del Sprint



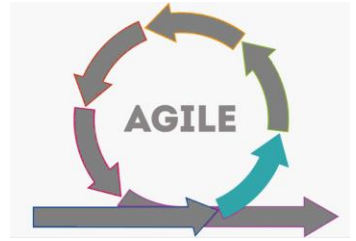
•Se realizan estimaciones sobre las historias del Backlog Del Producto.

•Se puede utilizar la Planificación De Póker como técnica para que los Miembros del Equipo de Scrum y el Scrum Master realicen la estimación.

•Si existen estimaciones muy dispares, el Scrum Master puede tomar los extremos y pedirle a las partes que justifiquen su estimación.

•En caso de variaciones menores, el Scrum Master puede decidir tomar el promedio de todas, o elegir el número seleccionado por la mayoría.

Estimación de Pruebas: Planning Poker



•Estimar el esfuerzo o complejidad de las tareas de un proyecto por consenso generalmente usando los valores 0,1/2,1,2,3,5,8,13,20,40,100 y una tarjeta de ?(inseguro).

•Es muy importante que dentro de la estimación se tenga en cuenta también todo el esfuerzo de la parte de calidad para poder calcular el esfuerzo correctamente.

0	1/2	1	2	3	5
8	13	20	40	100	?

NEORIS

Actividad 6 : Estimación con Poker

Equipos: Personal

Material:
Una baraja de Poker modificada
Historias de usuario a estimar

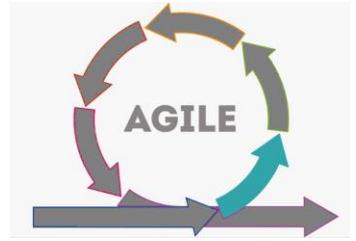
HU HU2

Indicaciones:
1. Leer la Historia a estimar
2. Discutir la Historia si se tienen dudas
3. Cada uno elige una carta en función al esfuerzo
4. Revisar y comentar los resultados

Para Poker Planing no presencial:
https://planningpokeronline.com/knb7zwUPF6Scv9XGkuiR

NEORIS

Planificación de Liberaciones: Tablero

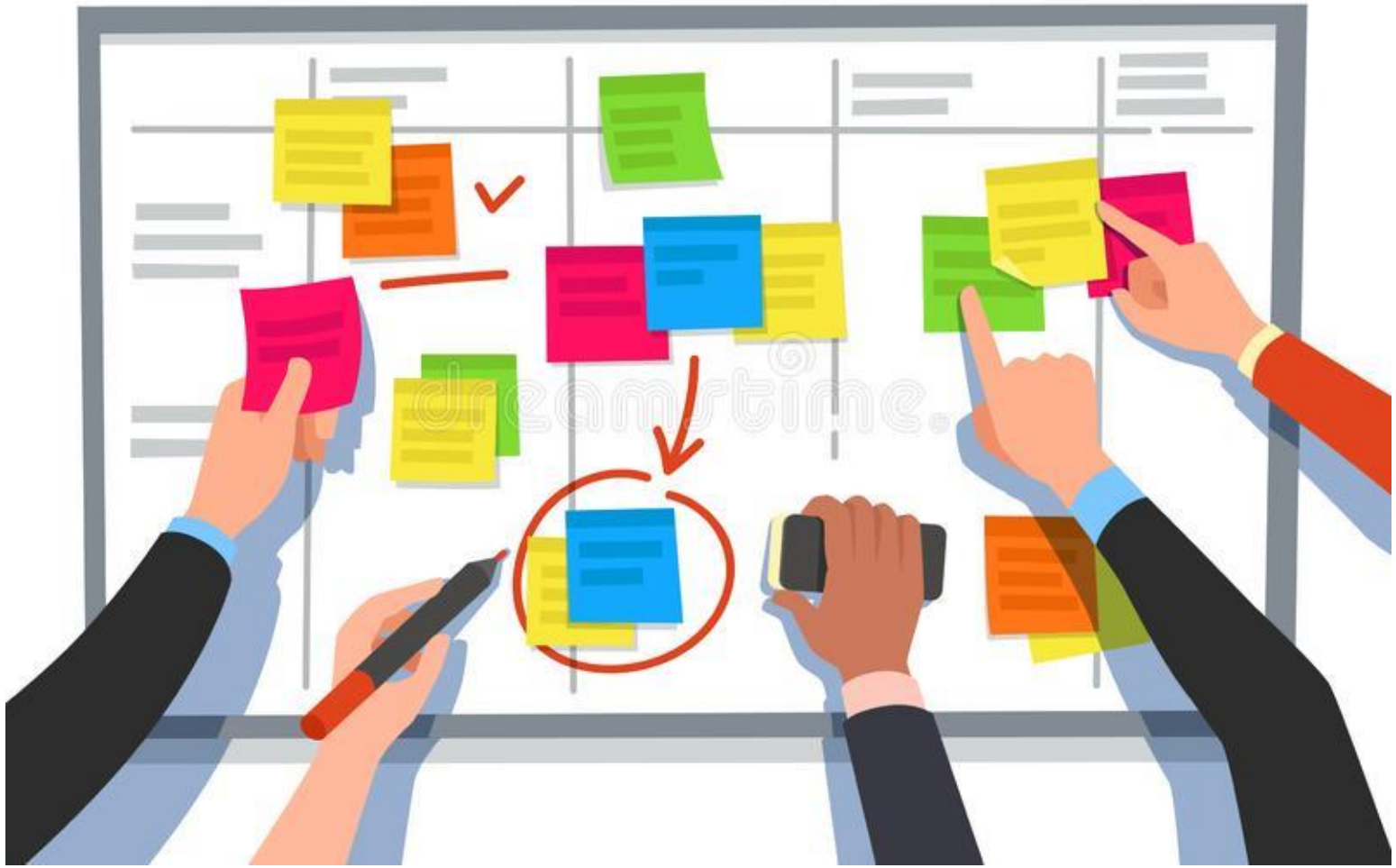


•En la columna de la izquierda se irán situando las historias del producto (Product Backlog Items) que el equipo se compromete a completar en la iteración, ordenados por prioridad para el cliente (Product Owner).

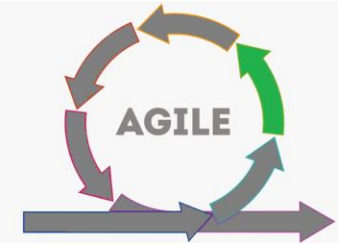
•A la derecha de cada objetivo se pondrán, en la columna "pendientes", las tareas necesarias para poder completarlo, indicando las horas estimadas para su resolución, que iremos actualizando en las reuniones diarias de sincronización (Scrum daily meetings).

•En su zona específica, dispondremos las tareas de mejora continua que se han derivado de la retrospectiva de la iteración anterior, que queremos resolver durante esta iteración y que, por tanto, consumirán tiempo de alguna persona.

Tablero



Análisis del Sprint



Cuando el sprint está en curso, debemos asegurar que:

Se realice el Análisis de los documentos de entrada y la ejecución de pruebas estáticas de las historias de usuario o algún otro artefacto donde se estén documentando las necesidades del usuario de la iteración, esto con la finalidad de:

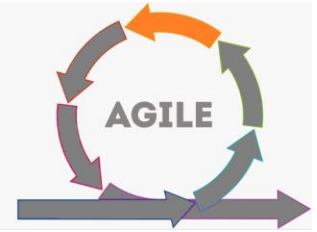
•Encontrar huecos funcionales

•Funcionalidades que no están completas en su definición

•Diferencia entre el documento de análisis y la pantalla.

NEORIS

Diseño del Sprint

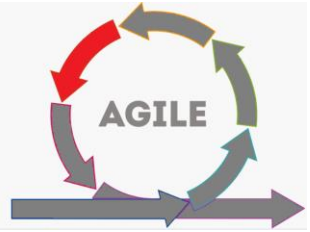


- Construir los casos de prueba manuales y scripts automatizados (ATDD, BDD) según el alcance y la definición de la estrategia de pruebas.

- Identificar datos de prueba requeridos para la etapa de ejecución.

- Identificar requerimientos de componentes o dispositivos en el ambiente de pruebas.

Ejecución del Sprint

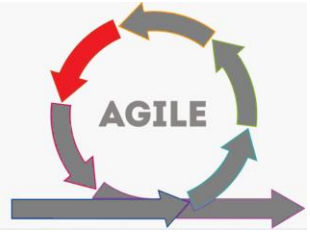


Cuando el sprint está en curso, debemos asegurar que:

- No se realizan cambios que afectan al objetivo del Sprint;
- No disminuyen los objetivos de calidad, y
- El Alcance podrá aclararse y re-negociarse entre el PO y el Equipo de Desarrollo a medida que se va aprendiendo.

- Cuando un Sprint es demasiado largo, la definición de lo que se está construyendo puede cambiar, puede aumentar la complejidad y puede aumentar el riesgo. Los Sprints permiten previsibilidad al garantizar la inspección y la adaptación de los avances hacia una meta de por lo menos cada mes de calendario.

Ejecución del Sprint: Tablero



•Las historias que se van moviendo en el tablero: Por hacer > En Progreso > Terminado; conforme se van construyendo

•Algunos flujos pueden tener más estados, incluso uno para QA

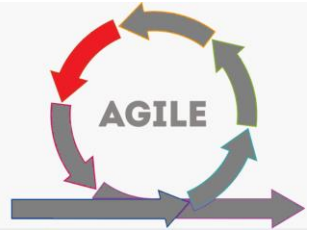
•Identificar las nuevas tareas que vayan apareciendo, sean:

- •Tareas asociadas a objetivos / requisitos / historias de usuario que no fueron identificadas en la reunión de planificación de la iteración.
- •Tareas inesperadas y no asociadas a objetivos pero que exigen nuestra resolución dentro de la propia iteración (en la zona "no planificado").

•Marcamos con rojo los objetivos y/o las tareas con más riesgos, aquellos que queremos tener controlados con más atención.

NEORIS

Ejecución del Sprint



•Ejecución de casos de prueba en modalidad “pruebas entre pares”.

•Ejecutar los casos de prueba construidos para el sprint, manuales y automatizados según sea el alcance y definición de la estrategia de pruebas.

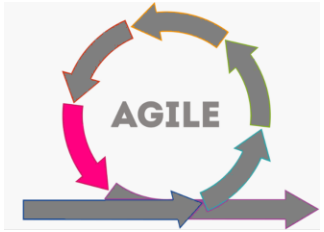
•Ejecutar ciclos de re-test y regresiones según sea la estrategia de pruebas.

•Reportar y dar seguimiento a defectos en caso de identificarse.

•Generar reportes de avance en la herramienta de gestión de proyecto.

NEORIS

Demo



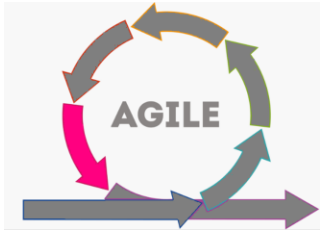
•Tiene lugar al finalizar el sprint, ya que al alcanzar el tiempo previsto para el mismo, es necesario revisar todo lo construido, ver si se ajusta a lo que necesitan los stakeholders y poder recibir feedback de los mismos.

•A esta reunión acuden el PO como representante de los Interesados, además del Equipo de desarrollo y el Scrum Master.

•Presentar al PO y a las Parte Interesada las funcionalidades (Historias de usuario) ya desarrolladas que fueron incluidas en el sprint.

•Acompañar al usuario final en la ejecución de los casos de prueba y dar seguimiento a los resultados de las pruebas y defectos en caso de encontrarse.

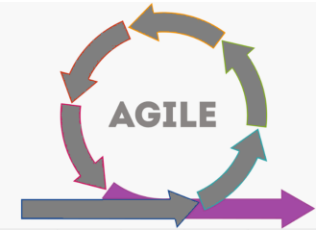
Demo



Puntos que se deben considerar para una DEMO:

- Listar todas las historias completadas en su agenda
- Elimine cualquier historia que no deba ser demostrada
- Organice las historias restantes de manera aproximada en escenarios o temas
- Decida si desea que los desarrolladores ayuden a dar partes de la demostración.
- Siempre establezca expectativas y de contexto a lo largo de la demo

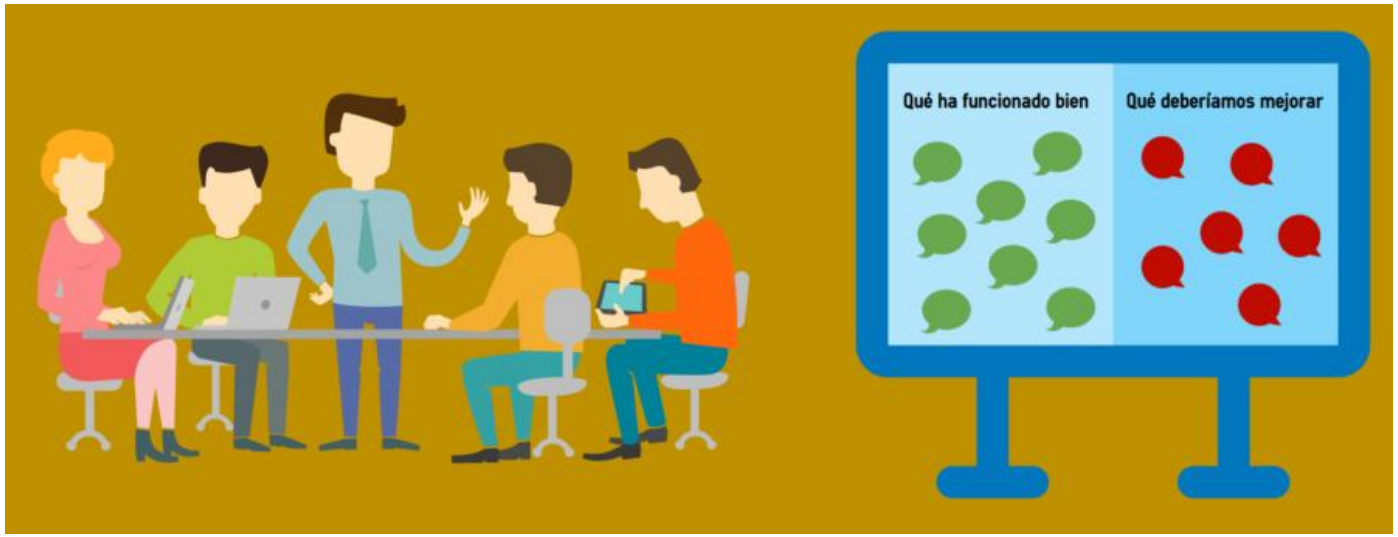
Retrospectiva de Sprint



•El equipo analiza cómo ha sido su manera de trabajar durante la iteración, por qué está consiguiendo o no los objetivos a que se comprometió al inicio de la iteración y por qué el incremento de producto que acaba de demostrar al cliente era lo que él esperaba o no.

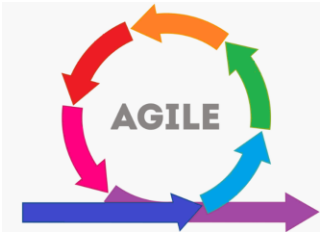
NEORIS

Retrospectiva de Sprint



- Qué cosas han funcionado bien.
- Cuales hay que mejorar.
- Qué cosas quiere probar hacer en la siguiente iteración.
- Qué ha aprendido.
- Cuales son los problemas que podrían impedirle progresar adecuadamente. El Facilitador se encargará de ir eliminando los obstáculos identificados que el propio equipo no pueda resolver por sí mismo.

Prácticas Metodológicas y Ceremonias Recomendas



Como parte del un Scrum Team, el Tester ágil se ve involucrado en todas las prácticas de un Sprint.

- Planeación del sprint
- Sesiones de Refinamiento
- Estimación del sprint
- Planificación de liberaciones
- Aplicación de técnicas formales de diseño de casos de prueba
- Pair Testing
- Ejecución de n ciclos de prueba
- Ejecución de sprint
- Scrum Daily
- Retrospectiva de sprint
- Siempre comunicación directa, asertiva y continua.

Scrum Daily Meeting

Es un evento de 15 minutos, cuyo objetivo es que el equipo de desarrollo sincronice actividades, y cree un plan para las próximas 24 horas.

El equipo de desarrollo utiliza el Scrum Diario para evaluar el progreso hacia la meta del Sprint y evaluar la tendencia del progreso en finalizar el trabajo en el Sprint Backlog.

Básicamente se responden 3 preguntas:

Reportes

Cuando ejecutamos proyectos de desarrollo de software, la fase de pruebas (Software Testing) suele ser crítica, y es un momento en el cual diversos interesados (stakeholders) requieren información al minuto sobre el estado de la calidad del software que se está desarrollando.

Para ello, se suele manejar un informe de avance de cómo van las pruebas, el cual según la criticidad del proyecto puede ser solicitado una o varias veces al día

.

La intención es comunicar a todos los involucrados de las áreas de pruebas, desarrollo, funcionales y área de negocio cual es la situación de las pruebas, que defectos críticos se están reportando y cuantos casos faltan por ejecutar.

Por ejemplo saber cuantos casos de prueba están con estatus exitoso, fallados, pendientes de probar, o bloqueados cual es la relación entre el porcentaje de avance planificado y el real, cual es la situación de los defectos cuantos están abiertos, han sido corregidos o no aplican, entre otros aspectos.

NEORIS

Sección	Información
Subject del correo	Estatus Testing: Cliente <Nombre del cliente>, Proyecto <Nombre del Proyecto> <Indicar consecutivo si es más de un reporte diario>
Estatus general	ESTADO GENERAL: EN CURSO <Mostrar el semáforo en base a la varianza y criterio, ver pestaña "Estatus General"> Fecha de Inicio del proyecto: <16 de Enero de 2020> <Las fechas de inicio y finalización son fechas reales y se obtienen de la estrategia de pruebas> Fecha de Finalización del proyecto : <23 de Enero de 2020> Fecha de Inicio del esfuerzo de testing: <16 de Enero de 2020> Fecha de Finalización del esfuerzo de testing: <23 de Enero de 2020> Reporte actualizado al día de hoy Martes 22 de Enero de 2020 a las 18:00 horas
Reporte ejecutivo	<Incluir tablas de la pestaña "Reporte ejecutivo">
Resumen	<Incluir comentarios : - Generales en bullets . - Que describan el estatus del proyecto en un lenguaje entendible y ejecutivo. - Que lleven congruencia y sea como plática resumida frente a frente. - Que incluyan que hicieron, si está o no terminado y/o que falta para terminar: - Que expliquen el punto por si mismo y no de pie a pregunten algo. - Que se respondan dudas.> < Ver ejemplos en la pestaña "Resumen">
Estadísticas de pruebas	<Si se encuentra en fase de Análisis: incluir información de la pestaña de "Análisis"> Si se encuentra en fase de ejecución: Incluir información de la pestaña "Ejecución">
Riesgos y problemas	<Mostar la descripción de riesgos abiertos y adjuntar el archivo de riesgos donde se les da seguimiento: FO-PRU-18 Plan de administración de riesgos y problemas>

Descripción
2 ingenieros de pruebas renunciaron en medio de la ejecución del ciclo 1, se tiene el riesgo de no terminar la ejecución en el tiempo planeado

<**IMPORTANTE:**
- Cuidar la congruencia en los números y estatus reportados,
- Conocerlos, entenderlos,
- Defenderlos y sustentar en caso de ser necesario,
- Respetar la periodicidad y hora de envío>

REPORTE

Estatus general

Avance	%
% Planeado	80%
% Real	60%
% Desviación	20%

<Validar el semáforo de acuerdo al criterio de la columna D >

Semáforo de estatus	Criterio	Comentarios
	Desviación < =10% No existen problemas que estén bloqueando la planeación. No existen riesgos detectados. Existen acciones de mitigación y se están llevando a cabo para corregir demoras existentes.	Consultar con el gerente que temas deberán ser reportados. Consultar en la estrategia de pruebas las alternativas de escalamiento
	Desviación > 10% y <= 20% Existe al menos 1 Riesgo de Bloqueo total a los 2 días posteriores.	
	Desviación > 20% Existe al menos 1 problema que esté bloqueando la planeación totalmente.	

NEORIS

REPORTE

Reporte ejecutivo

Tipo y Nivel de pruebas	Requerimientos	Analizados	Validados por funcional	Validados por Usuario	Pair testing	Pruebas QA	Pruebas de Aceptación de Usuario
Componentes	45	45	45	45	45	45	N/A
Integrales	23	23			N/A	23	N/A
UAT	50						34

Nombre requerimiento por nivel de prueba <ir pestaña de detalle requerimiento>

Pruebas de rendimiento

Flujos de negocio	Flujo detallados	Preparación de Datos	Construcción
23	23	23	23

Escenarios de prueba	Escenarios Validados
23	23

Ejecuciones	Preparadas	Ejecutadas	Reportadas
10	10	10	10

Semáforo	Descripción de semáforo
En proceso, dentro del plan	La actividad se está realizando dentro del plan definido
En proceso, fuera del plan	La actividad se está realizando, pero en fechas fuera de las planeadas
Realizado al 100%, fuera del plan	Requerimientos probado o validados, sin defectos, pero en fechas fuera de las planeadas
Realizado al 100%, dentro del plan	Requerimientos probado o validados, sin defectos, dentro del plan
Realizado parcialmente	Requerimientos Probados o validados parcialmente
No realizado	No se realizó la actividad

NEORIS

REPORTE

Detalle de reporte

<Actualizar las columnas de acuerdo al modelo de desarrollo Cascada o ágil>

Sprint	Flujo / Épica /Historia/Tarea testing	Dev	Unit Testing	Diseño de prueba	Diseño validado PO	Diseño validado Cliente	Pair Testing	Regresión	Pre Demo	Demo
1	Flujo / Épica /Historia/Tarea testing	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan
1	Flujo / Épica /Historia/Tarea testing	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan	En proceso, fuera del plan
1	Flujo / Épica /Historia/Tarea testing	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, fuera del plan
1	Flujo / Épica /Historia/Tarea testing	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan	Realizado al 100%, dentro del plan
1	Flujo / Épica /Historia/Tarea testing	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente
1	Flujo / Épica /Historia/Tarea testing	No realizado	No realizado	No realizado	No realizado	No realizado	No realizado	No realizado	No realizado	No realizado
2	Flujo / Épica /Historia/Tarea testing	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan
2	Flujo / Épica /Historia/Tarea testing	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan
2	Flujo / Épica /Historia/Tarea testing	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan

estar en verde la columna "Demo" que indica que se han cubierto los criterios de aceptación>

<Actualizar con Flujo/Épica Historia/Tarea de testing de cada Sprint>

Semáforo	Descripción de semáforo
En proceso, dentro del plan	La actividad se está realizando dentro del plan definido
En proceso, fuera del plan	La actividad se está realizando, pero en fechas fuera de las planeadas
Realizado al 100%, fuera del plan	Requerimientos probado o validados, sin defectos, pero en fechas fuera de las planeadas
Realizado al 100%, dentro del plan	Requerimientos probado o validados, sin defectos, dentro del plan
Realizado parcialmente	Requerimientos Probados o validados parcialmente
No realizado	No se realizó la actividad

REPORTE

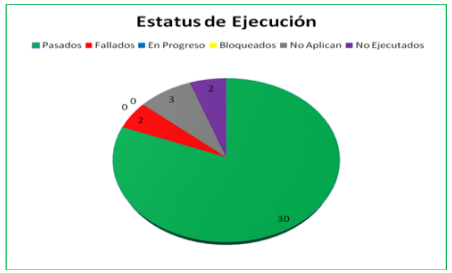
Detalle de reporte

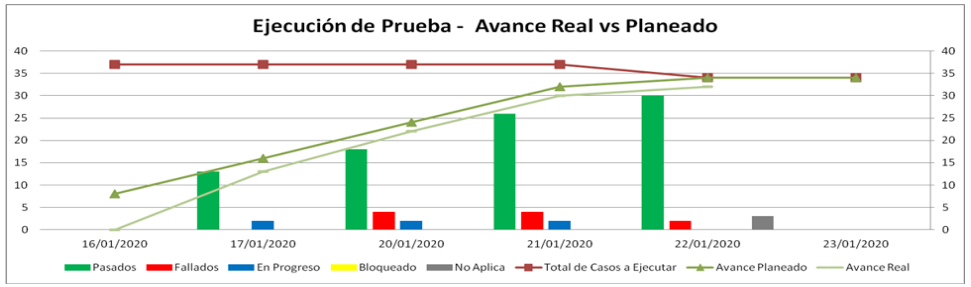
<Identificar por Requerimiento que es un flujo/Épica/Historia/Tarea de testing, los casos de prueba identificados durante el análisis, La traza la encuentran en la Matriz de Pruebas y en la Herramienta de gestión de pruebas>
<En el correo y presentación no mostrar la columna F y mostrar la información por requerimiento, en el ejemplo solo se mostrarían las líneas 8,12, 14>

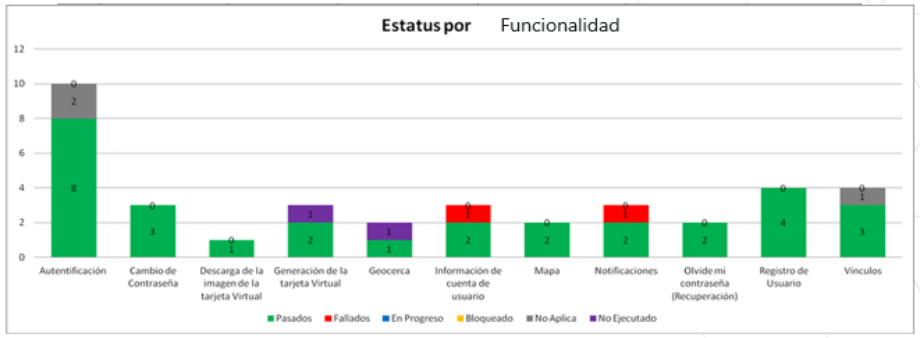
Producto	Sprint	Flujo / Épica /Historia/Tarea testing	Número de casos de prueba por requerimiento	Generación de Grafo	Detección de huecos funcionales	Solución de huecos funcionales	Diseño validado PO	Diseño validado Cliente
	1	Flujo / Épica /Historia/Tarea testing 1	0	En proceso, dentro del plan	En proceso, fuera del plan	Realizado al 100%, fuera del plan	Realizado al 100%, dentro del plan	Realizado parcialmente
	1	Flujo / Épica /Historia/Tarea testing 2	0	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente	Realizado parcialmente
	2	Flujo / Épica /Historia/Tarea testing 3	0	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	En proceso, dentro del plan	No realizado
		<Actualizar con Flujo/Épica Historia/Tarea de testing de cada Sprint>						

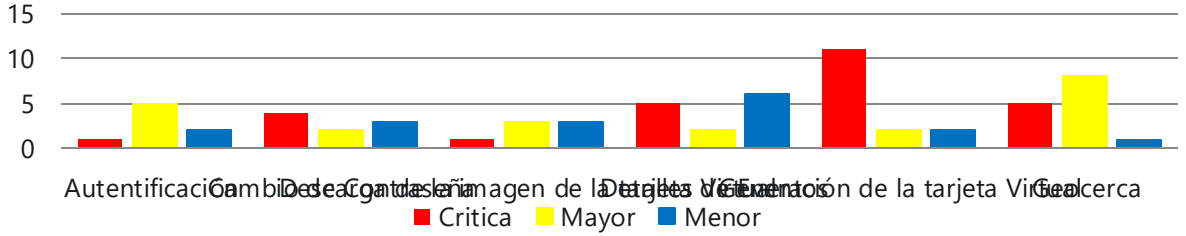
Semáforo	Descripción de semáforo
En proceso, dentro del plan	La actividad se está realizando dentro del plan definido
En proceso, fuera del plan	La actividad se está realizando, pero en fechas fuera de las planeadas
Realizado al 100%, fuera del plan	Requerimientos probado o validados, sin defectos, pero en fechas fuera d
Realizado al 100%, dentro del plan	Requerimientos probado o validados, sin defectos, dentro del plan
Realizado parcialmente	Requerimientos Probados o validados parcialmente
No realizado	No se realizó la actividad

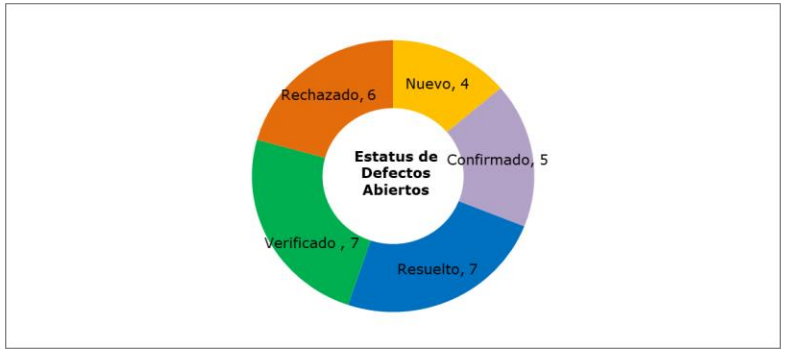
Reportes











NEORIS

Gestión de Defectos

La tolerancia a bugs debe ser muy baja en un ambiente de desarrollo ágil. Los bugs producidos por el desarrollo de una funcionalidad deben ser resueltos dentro del mismo Sprint o en el siguiente.

Los defectos se deben priorizar "reporte efectivo".

Es posible interrumpir un sprint para solucionar defectos.

Los defectos siempre están sobre una nueva funcionalidad, por lo que se deben mantener arriba en el Product backlog, para que sean solucionados lo antes posible.

Tener muy en cuenta los costos escondidos de los bugs.

Aplicar la refactorización para mejorar la calidad del código.

NEORIS

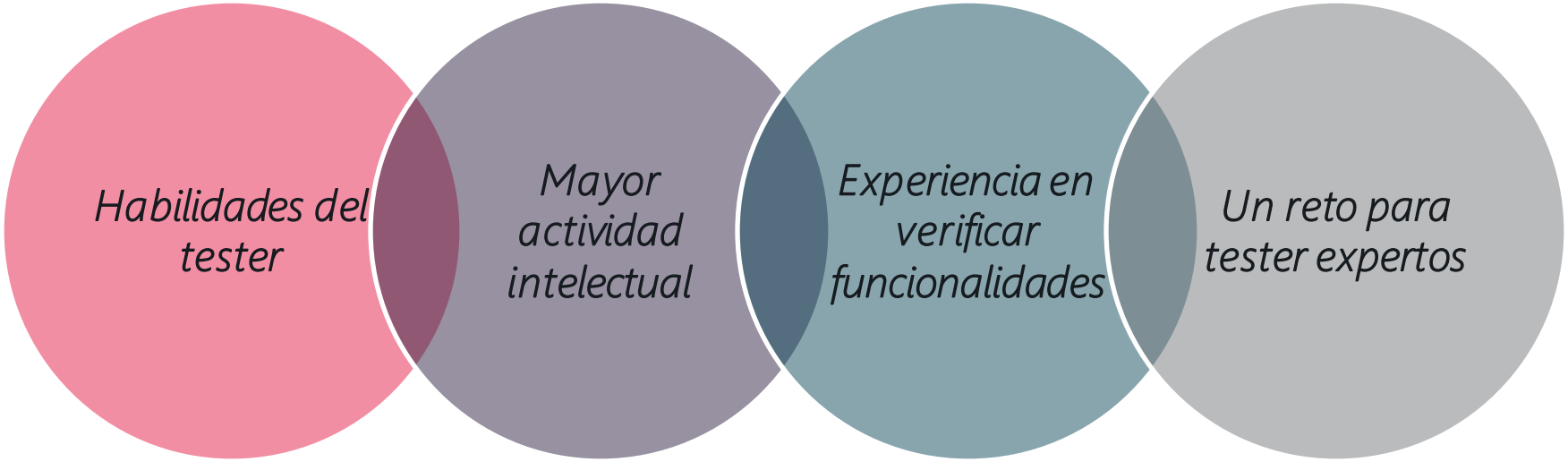
Testing Exploratorio

~~*Sin planificación alguna, sin límite de tiempo, sin documentación.*~~

✓El testing exploratorio es un enfoque en el que simultáneamente se aprende sobre la aplicación, se diseñan casos de prueba y se ejecutan esos casos de prueba.

✓Cuando una persona hace testing exploratorio , diseña y ejecuta pruebas con un objetivo concreto.

✓Con las conclusiones que obtienen se va aprendiendo sobre la aplicación y utiliza esa información para diseñar y ejecutar nuevas pruebas.



NEORIS

Tdd, Atdd Y Bdd.

TDD(Test Driven Development)
Desarrollo guiado por pruebas.

ATDD(Aceptance Test Driven Development
Desarrollo guiado por pruebas de aceptación

➢BDD(Behavior Driven Development
Desarrollo guiado por comportamiento

TDD¿Cómo se relacionan las TDD y las Pruebas Unitarias?

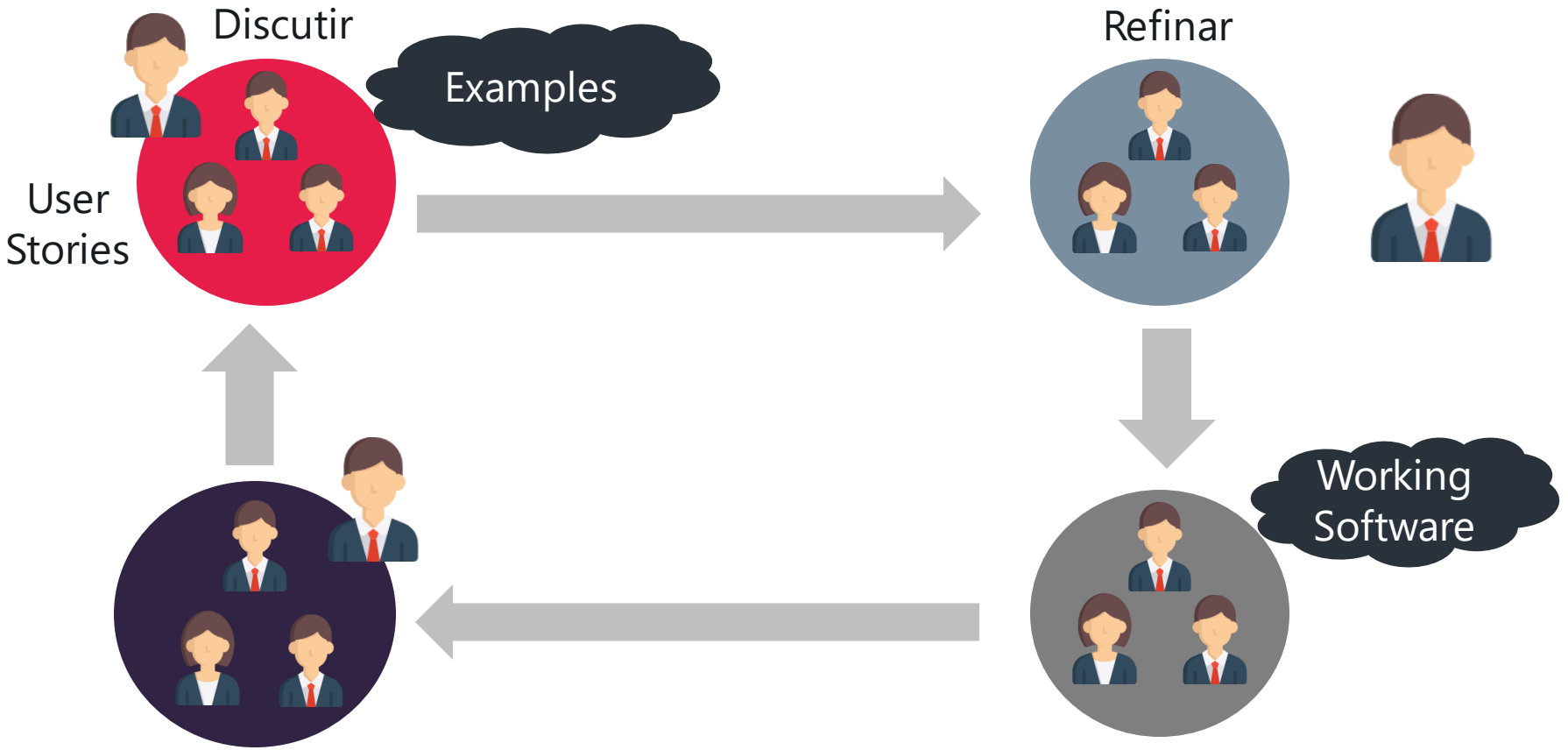
Las pruebas unitarias aseguran la aplicación se ha construido correctamente.

Muchas veces no sabemos exactamente qué clases hacer o qué métodos ponerle exactamente, perdemos el tiempo haciendo métodos y clases que pensamos que luego serán útiles.

TDD= primero pensamos en qué queremos hacer y después pasamos al cómo.

ATDD(Acceptance Test Driven Development)

Las pruebas de aceptación aseguran que se ha desarrollado la funcionalidad.



ATDD(Acceptance Test Driven Development)

Beneficios:

- Ejemplos reales un lenguaje común de entender el dominio
- Permite identificar correctamente las reglas del negocio
  - Los flujos de trabajo funcionan desde el primer momento
- Criterio visible para dar por finalizadas las historias del usuario
- No implementamos nada hasta tener definidos los test, se reduce tiempo perdido reprogramando.
- Validación automatizada.

BDD(Behavior Driven Development)

¿Qué es?

BDD busca un lenguaje común para unir parte técnica y de negocio, que sea desde ese lenguaje común desde donde arranque el Testing y, desde ahí, el desarrollo.

En BDD, las pruebas de aceptación se escriben usando historias de usuario.

"Como [rol] quiere[característica] para que [los beneficios]".
Dado que contexto inicial], cuando [se produce el evento], entonces [resultados].
Given [initial context], when [event occurs], the [ensure some outcomes].

NEORIS

Historias de Usuario y Bdd: Criterios de Aceptación de Escenarios

El ciclo básico de BDD sería el siguiente:

1. Escribir las historias de usuario.
2. Escribir los escenarios de prueba.
3. Automatizar las pruebas de aceptación.

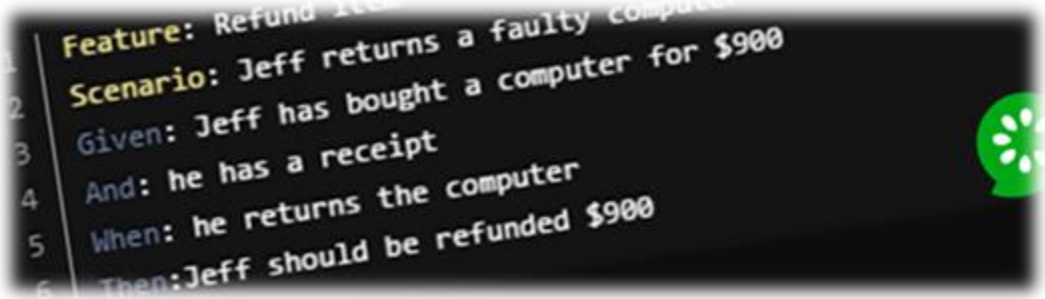
Bdd y Cucumber

Sirve de puente de comunicación entre el cliente/analista del negocio y el desarrollador.

Permite utilizar un lenguaje común entre los desarrolladores y los clientes.

Tiene un enfoque de fuera hacia adentro, esto es, ser parte de los criterios de aceptación hacia la implementación .

Se utiliza un lenguaje ubicuo Gherkin.



NEORIS

Diferencias Entre:

TDD
-Nivel unitario
-Técnica de Desarrollo que se enfoca en escribir el código
-El desarrollador escribe las pruebas

ATDD
-Nivel de aceptación
-Metodologías de trabajo para escribir pruebas para validar el comportamiento del código.
-Lo escriben los Analistas de pruebas.

BDD
-Nivel Integral
-Metodologías de trabajo para escribir pruebas para validar el comportamiento del código
-Lo escriben los Analistas de pruebas.

Mitos de Ágile Testing

1. Los tester no pueden obtener documentación detallada de los requerimientos para realizar pruebas.
2. En comparación a los proyectos tradicionales, en agile los tester no tienen tiempo suficiente para completar sus pruebas en sprint.
3. La calidad del producto es responsabilidad del tester.
4. La automatización es la única manera de incorporar el testing en equipos ágiles.
5. Pruebas en paralelo con el desarrollo no es posible.

Problemáticas

Alta de experiencia, un equipo nuevo, un cliente impaciente cualquier inconveniente extra puede hacer que los problemas más comunes del desarrollo con métodos ágiles se convierta en una bomba de relojería. Para evitar sus potenciales efectos perniciosos para el proyecto y sus resultados es necesario conocerlos.

Los más frecuentes son:

Insuficiente nivel de conocimientos: el conocimiento y la formación son imprescindibles para gestionar proyectos a través de los métodos ágiles. Asistir a alguna conferencia sobre Kanban, participar en un seminario de Scrum o matricularse en un curso ágil buscando obtener una certificación para iniciarse en el mundo ágil no basta para estar preparado. La práctica no es igual a la teoría y, si a pesar de todo se está en posición de responsabilizarse de una iniciativa de este tipo, merece la pena buscar mentores con experiencia en la implementación ágil.

Problemáticas

Problemas de localización: pese a que el trabajo con equipos remotos es cada vez más frecuente, en lo que concierne a este tipo de proyectos puede no ser la mejor opción. La cercanía entre los desarrolladores favorece la resolución de problemas y minimiza su aparición, además de permitir aumentar el rendimiento.

Síndrome del Burnout: trabajar con métodos ágiles puede aumentar el estrés de algunos de los desarrolladores. Esta circunstancia puede detectarse a tiempo de poner medidas, por lo que hace falta mantener los ojos bien abiertos antes de que los resultados se vean afectados y la desmotivación se extienda al resto del equipo.

Falta de participación del cliente: a pesar de su entusiasmo por trabajar en base a métodos ágiles, a la hora de la verdad, el cliente no lo pone tan fácil para mantener el ritmo de comunicación necesario. Dejarlo todo claro desde el principio acerca de la importancia de su involucración puede asegurar que el cliente será capaz de comunicarse con el equipo de desarrollo sobre una base constante

Problemáticas

Desarrolladores protagonistas: muchas veces puede suceder que algunos miembros del equipo se consideren superiores al resto, por su nivel de experiencia o conocimientos, y esto les anime a pensar que pueden tomar decisiones finales por sí mismos sin necesidad de consultar al resto.

Aumento del riesgo: puede suceder que el ritmo del trabajo y el entusiasmo por los logros alcanzados haga perder la aversión al riesgo que se necesita en cualquier proyecto. Si bien la innovación y la proactividad son atributos necesarios en toda iniciativa de IT, si no se emplean técnicas de gestión de riesgos dentro del desarrollo ágil los problemas no tardarán en aparecer.

Falta de controles de calidad: en la misma línea, la calidad no puede obviarse. De nada sirve avanzar a toda velocidad si el producto no reúne las condiciones necesarias. De hacerlo así, además de surgir problemas en algún punto del desarrollo, los clientes terminarán descontentos.

Problemáticas

Vuelta atrás: si los equipos que comienzan a trabajar con métodos ágiles se sienten inseguros, pueden tomar la decisión de volver atrás y, en lugar de pedir ayuda, restaurar los procesos que empleaban en el pasado. Es preciso controlar que no se llega a este punto.

En cualquier caso, la mejor forma de averiguar si se está preparado para llevar a cabo un proyecto apoyado por los métodos ágiles es probándolo. Probar la metodología ágil seleccionada en una parte de un proyecto o en una iniciativa menos importante puede ayudar a evaluar cómo funciona y determinar si es conveniente plantearse el exportar este tipo de prácticas a otros ámbitos o si es mejor continuar formándose y esperar hasta estar más preparado.

Actividad

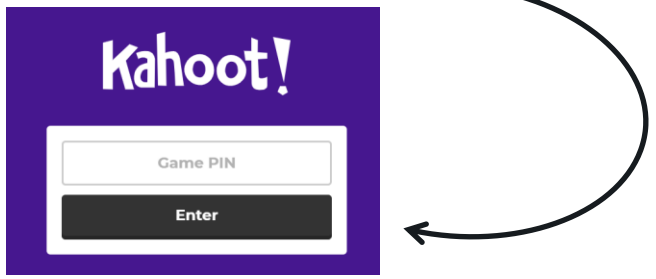
Instrucciones:

- Navega el Código QR que se muestra a continuación

QR

O captura en un navegador https://kahoot.it/

- Capturar el Game PIN: **4157106**



- Una vez entrando, no proporciones datos personales: Nombre completo, fecha de nacimiento, correo electrónico, teléfono, RFC, Nombre de la compañía, datos de tarjetas. Puedes utilizar un nickname

NEORIS

www.neoris.com

Gracias!!!

NEORIS