¿Qué es un requisito?

- Algo que el sistema debe hacer o una cualidad que el sistema debe poseer (Robertson-Robertson)
- Establece QUÉ DEBE HACER el sistema, pero no CÓMO HACERLO.

Requerimiento SMART

- Specific: concreto y bien definido: involucrados, qué, dónde y cuándo quiero lograrlo, y los obstáculos y limitaciones.
- Measurable: establecer criterios concretos para medir y evaluar el progreso. Indicadores cuantificables.
- Achievable: realista. Hay que tener en cuenta las posibilidades y limitaciones personales y económicas al definirlo.
- Relevant: está alineado con otras metas y tiene sentido en el conjunto del negocio. Agrega valor al negocio.
- Time bounded: se fija una fecha límite.

#Trazable: llevar registro histórico de dónde surge y cómo muta a lo largo del tiempo.

La definición de un requerimiento contiene:

- Descripción: Su contenido sigue el formato de una oración acompañado de un verbo modal. Evitar cualquier ambigüedad (palabras homónimas, definir contexto del requerimiento, evitar generalizaciones). Ejemplo: El producto **debe** de almacenar los caminos que han sido tratados. En caso de que se extienda en más de una oración, dividirlo en requerimientos más pequeños.
- Razón fundamental (opcional): añade contexto. Ejemplo: “Permite agendar los caminos sin tratamiento y destacar los posibles peligros”
- Criterio de ajuste: permite generar un método para medir los estándares de calidad. Suelen ser importantes al definir RNF.

Ciclo de vida de un proyecto: es útil para mantener un seguimiento, optimizar recursos, facilitar la comunicación, comprender el problema, permite la reutilización, facilitar el mantenimiento, garantizar nivel de calidad, cumplir plazos de tiempo, evaluación de resultados.

1. Obtención de requisitos: identificar qué motiva el inicio del proyecto y los recursos disponibles. Es fundamental entender el **contexto de negocio** y la interacción con los usuarios. Reconocer puntos críticos y definir el comportamiento.
2. Análisis de requisitos: se estudian los requisitos y se intentan solucionar las deficiencias que los requisitos puedan tener, además tiene como finalidad tener un control más completo en cuanto a tiempo de desarrollo y costo. Para brindar al usuario todo lo necesario se aplican las técnicas de relevamiento. Este análisis se plasma en el documento ERS (especificación de requisitos del sistema) cuya estructura puede venir definida por estándares. #Se habla de ingeniería de requisitos.

Técnica de especificación de requisitos:

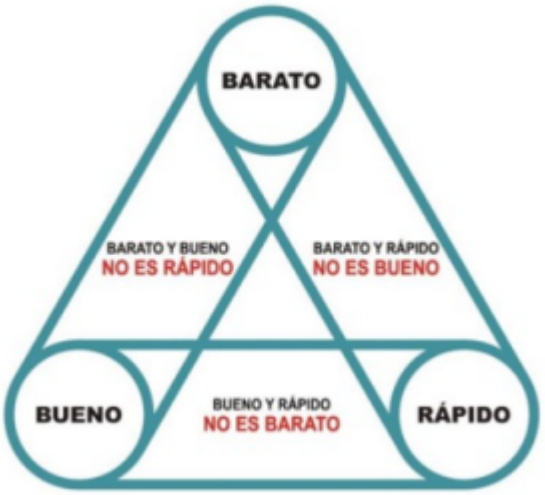
- casos de uso: es un diagrama que muestra el comportamiento del sistema mediante su interacción con los usuarios y/u otros sistemas. Suelen estar acompañados de un glosario de terminología y cada uno se centra en una única tarea. Centrados en el cliente, lenguaje natural. Aunque se asocia a la fase de test, es erróneo, se usa mayormente en las primeras fases de desarrollo. Establecen requisitos de comportamiento, pero no completamente requisitos funcionales y para nada no funcionales. Tipos de relaciones:
  - comunica: actor y caso de uso
  - usa o incluye: dos casos de uso
  - generalización: caso de uso es una variante de otro
  - extiende: caso de uso base declara puntos de extensión
  - También herencia entre casos de uso o actores.
- historias de usuario: describen el por qué y qué detrás del desarrollo, además del valor que provee al usuario final. Es una explicación general informal de un componente del software descrita desde la perspectiva del usuario final. Se usan metodologías ágiles como Scrum o Kanban. “As a [person], I [want to], [so that]”.

3. Arquitectura: El arquitecto añade valor con soluciones tecnológicas. Es una actividad de planeación a nivel de infraestructura. Elabora un plan de trabajo viendo la prioridad de tiempo y recursos disponibles. Permite visualizar la interacción entre entidades, para ello se utilizan diagramas de clase, MER. Las herramientas para el modelado de software se denominan CASE (computer aided software engineering).
4. Programación: implementar el código. Depende del lenguaje y el diseño previamente realizado.
5. Desarrollo de la aplicación: 5 fases
   a. desarrollo de la infraestructura
   b. adaptación del paquete
   c. desarrollo de unidades de diseño interactivas (diálogo usuario-sistema)
   d. desarrollo de unidades de diseño batch (diagramas de flujo para plasmar las especificaciones)
   e. desarrollo de unidades de diseño manuales (para proyectar todos los procedimientos administrativos entorno a la utilización de los componentes)
6. Pruebas de software: nunca perder contacto con los interesados
   a. pruebas unitarias (cada módulo)
   b. pruebas de integración

   a. compuesto por personal que desconoce para asegurar que la documentación es clara para cualquiera
   b. por programadores con experiencia porque pueden poner atención a detalles

7. Implementación: convertir a sistema ejecutable
8. Documentación: para eventuales correcciones y mantenimiento futuro
9. Mantenimiento: ⅔ del ciclo de vida de un proyecto, consiste principalmente en extender el sistema (80%) y eliminar errores.

Triángulo de hierro



PROCESO DE REQUERIMIENTOS

Concepción e investigación:
Surge con una idea. Se realizan estimaciones preliminares de alcance y costos ROM (hay un desvío entre -25% y +75%). ¿Cuándo se tiene un entendimiento suficiente y cuándo es suficiente?

-necesidad o motivo de la concepción claramente identificado
-lista preliminar de stakeholders
-lista preliminar de riesgos
-sponsor
-restricciones de tiempo y costo

Alcance:
Identificar el límite entre el área de trabajo y el resto del universo, entradas, servicios, stakeholders y su nivel de involucramiento, y especificar restricciones.

Determinación del producto:

-Área de trabajo

-BE(business event) casi siempre el origen del BE se origina fuera del sistema (humanos que interactúan con el sistema), cuando sistemas adyacentes hacen algo. Cada BUC son un conjunto de funcionalidades que dan respuesta a un BE.

-BUC(business use case) por cada business event existe una respuesta planificada que llamaremos caso de uso de negocio. Cada evento dispara un BUC.

-PUC(product use case) una vez identificado el BUC, elijo qué automatizar.

Construcción:
Una vez que están los requisitos bien definidos

-Funcionales
-No funcionales
-Restricciones
-Criterios de aceptación

Además de tener trazabilidad, es decir que cada requisito debe ser facilmente trackeable al BUC que lo originó.

Estrategias:

-Iterativa: El producto se contruye mediante pequeños incrementos. En cada iteración se agrega valor y tiene feedback, por lo que si hay algún error se corrige a menor costo. Cuando surge un imprevisto, tengo que volver al work investigation para ver cómo lo manejo.

-Cascada: Como no es iterativo, la comunicación con el cliente se corta en la primera etapa. Además, si surge un error en alguna de las etapas, se deberá repetir todo el ciclo o se arrastrará el error.

Rol del analista de negocios: antes era un rol muy pasivo, entrevistaba a los usuarios y escribía lo que le dijeran. Los requerimientos mal relevados son motivo de fallas en más de la mitad de los proyectos. Hoy en día el énfasis está en entender el verdadero problema a resolver, la necesidad. Enfocarse primero en la necesidad y no tanto en la solución. El analista termina entendiendo mejor que el cliente el problema a resolver.

Documentar requerimientos:
Hay que priorizar los BUC e incluir los más relevantes. La documentación sirve para relevar y verificar que se están haciendo las cosas como se pretende.

#A medida que un proyecto avanza, los costos de desarrollo van a incrementarse (mantenimiento evolutivo), el beneficio que agrega tener más funcionalidades va bajando respecto al costo.

BLASTOFF / KICKOFF

Esta fase incluye varias reuniones que tienen tareas previas y varios entregables. Es una documentación viva.

Entregables:

-Propósito del proyecto: intenciones y beneficios que traerá el negocio. Explica el qué y por qué del proyecto.
-Alcance del trabajo: identifica las áreas de negocio afectadas.
-Stakeholders: identifica a los interesados.
-Restricciones: aquellas conocidas de antemano, nos limitan y focalizan el trabajo. Se pueden manejar como un requisito especial que guía el esfuerzo (tiempo y costo). Tener en cuenta Alcance-Tiempo-Costo
-Glosario: Terminología que se utilizará de forma continua en el proyecto, no puede tener dos significados.
-ROM(costo estimado): es lo que más le interesa al cliente, no es fácil de dimensionar, pero tenemos un volumen de información como para hacer una estimación.
-Hechos relevantes.
-Suposiciones.
-Riesgos: toda acta de constitución debe tener una lista preliminar de riesgos.
-GO/NO GO decision: luego de este análisis se tiene suficiente información como para determinar si el proyecto es viable. Si no, se cancela.

Acta de constitución:

-Requisitos de alto nivel
-Premisas(supuestos) y restricciones
-Descripción de alto nivel del proyecto y sus límites
-Riesgos de alto nivel
-Resumen del cronograma de hitos
-Resumen del presupuesto del proyecto
-Lista de Stakeholders (Patrocinador, quien pone la plata. Gerente, el responsable)
-Requisitos de aprobación del proyecto
-Criterios de cierre o cancelación del proyecto

#La parte más difícil de esta fase es buscar y definir correctamente la necesidad.

Requisitos SMART: Uno de los mayores problemas del desarrollo es el malentendido de requisitos. Deben ser SMART, comprendidos y validados por el cliente antes de que se empiece a construir. Deben incluir descripción (evitar ambigüedad, cuidado con homónimos y generalizaciones, y siempre definir el contexto), razón fundamental/rationale (opcional) y fit criteron (criterio de ajuste). Se escriben en un lenguaje que todos entiendan.

- Reutilización de requisitos: puede que lo que me pida el cliente ya lo haya resuelto en otro proyecto, lo que me da una base para lo pedido, no es copiar y pegar, ya que es otro proceso, cliente y contexto.

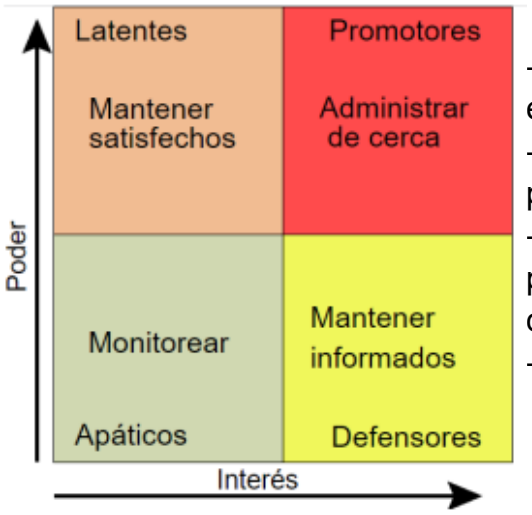
- Retrospectiva: una vez finalizada la fase de especificación de requisitos, se revisa la consistencia entre requisitos y entre todo el proyecto, ya que tengo más información y está más definido el alcance. Además se documentan las lecciones aprendidas: ¿qué hicimos bien? ¿qué hicimos mal? ¿qué haríamos distinto?

Los requerimientos van a seguir surgiendo en fases posteriores, los tipos de cambio son:

- corrección: corregir defectos
- adaptación: modificar por cambios externos
- mejora: ampliar los requisitos funcionales a petición del cliente
- prevención: cambio para facilitar el cambio

El ALCANCE define qué STAKEHOLDERS son afectados:

Matriz Mendelow



-P: son los más importantes. Tienen una gran influencia en el proyecto.
-L: tienen poder para afectar el proyecto, pero tienen poco interés.
-D: tienen un gran interés en el proyecto, pero tienen poco poder, no es necesario consultarlos en todas las decisiones.
-A: tienen poco poder y poco interés en el proyecto.

Luego del blastoff tenemos un área de trabajo definida. Si es muy grande, partirla en subáreas, una forma de hacerlo es por los BE (pueden tener una acción directa sobre los BUC o indirecta como por tiempo).

Es fundamental entender el área de negocio para entender qué tipo de producto construir. Hay que analizar el sistema en su totalidad, es decir al actor, la interacción con el sistema y el trabajo que rodea esta interacción.
Trabajo=sistema haciendo cosas.
Sistema=cualquier cosa que produzca bienes/servicios o información útil para su dueño.
Actores=personas o sistemas que interactúan con el producto automatizado (PUC).

Técnicas de relevamiento para descubrir los procesos del negocio y las personas involucradas en ellos.

#El punto de partida para descubrir los requisitos son los BUC.

TÉCNICAS DE RELEVAMIENTO

1. Apprenticing: el analista se pone en la posición del usuario, le muestra cómo trabaja y él definirá qué es lo que necesita. Puede ser que esté de incógnito o que sepan que está trabajando ahí. No es aplicable a todos los trabajos. Es apropiado cuando se espera **reimplementar partes significativas del trabajo** y los sistemas actuales.

2. Workshop (BUC): se juntan todos a hablar y registrar la información. La diferencia con el anterior es que antes el analista era espectador, ahora la persona también ayuda a decir lo que se necesita y cómo es el negocio. Es útil cuando se están realizando **cambios fundamentales en el trabajo** y deben incluir a los interesados especializados adecuados. Incluye el resultado deseado para el BUC, un escenario de caso normal y escenarios de excepción.

3. Entrevistas: no todas las personas sirven para ser entrevistadas, necesita hacer el trabajo y tener habilidades de comunicación, es por eso que es complejo, no toda persona está hecha para ser entrevistada y sacar de ella buenos **requerimientos**. No es ir y que el usuario cuente lo que quiere, hay que tener una guía preparada.

4. Reutilización: parte de 2 grandes supuestos.
    1-Es probable que alguien en su organización haya estudiado un trabajo similar y haya escrito **requisitos** para un producto similar.
    2-Se tiene una base de conocimiento lo suficientemente sólida como para poder navegar y encontrar esos requisitos.
Es posible establecer similitudes entre procesos y utilizar requisitos abstractos para ahorrar tiempo en proyectos posteriores.

5. Dirty Process Modeling: No son modelos formales, es para construir modelos de negocio y validar ideas rápidamente. Son **útiles cuando se reemplaza gran parte del sistema legacy**, construido con cosas físicas. Ejemplo: post–it

6. Prototipos low/high: especialmente útil **cuando se va a hacer algo que no existe** y que es difícil de imaginar. Es más fácil sacar información del prototipo (por ejemplo, toco un botón y espero que haga una cosa en vez de otra). Es desechable, no seguir construyendo sobre el prototipo. Útiles para **elicitar requisitos**.

7. Mind map: es una forma rápida de **representar gran cantidad de información** y organizarla. No necesariamente tiene que ser texto, también se pueden dibujar cosas.

8. Wikis, blogs, discussion forums: cuando tenemos **proyectos grandes**, es importante saber qué opina la comunidad para descubrir **requisitos**. Son efectivas porque a la gente le gusta dar su opinión.

9. Document archeology: Es una técnica que consiste en buscar **requisitos** subyacentes en informes y archivos existentes. Es útil cuando se tiene un **sistema existente o heredado que se planea modificar o renovar**.
Sesgo, existe la posibilidad de no pensar en cómo funciona el sistema y solo uso las cosas anteriores.

10. Family therapy: no buscan que las personas estén de acuerdo. En cambio, su objetivo es hacer posible que las personas escuchen y comprendan las posiciones de los demás individuos. Se utiliza en grupos diversos de personas, utiliza ideas de la terapia familiar y un **bucle de retroalimentación para evitar malentendidos**.

11. Persona analysis: construimos perfiles y empezamos a probarlos en como sería el proceso. Genera productos atractivos, relevantes, útiles y centrados en el usuario. Destaca la importancia de las personas, facilitando la comunicación entre los miembros del equipo. Se incluyen las habilidades (conocerlo para satisfacer sus necesidades), preocupaciones (necesidad), objetivos y competencias tecnológicas. El objetivo es el fin que estoy buscando lograr con el producto.

¿Cómo elegir la técnica? No existe una mejor, depende de varios factores. El primero es que el analista de negocio se sienta cómodo con la técnica.
Consideraciones adicionales:

- geografía (ubicación de los stakeholders, ej: remoto),
- legado (si hay que conservar mucho la implementación actual, entonces algunas técnicas más abstractas no serán adecuadas),
- abstracción (se puede abordar el negocio de manera abstracta o los stakeholders necesitan realidades físicas).

ESCENARIOS (BUSINESS USE CASE SCENARIO)
La creación de escenarios para modelar es una buena herramienta debido a su facilidad de entendimiento por parte de los interesados no técnicos.

El escenario se divide en una serie de pasos que describen el proceso actual. Se identifican las partes interesadas (y las activas), y se discuten posibles mejoras o soluciones en el proceso. Permite definir la funcionalidad de un BUC de manera comprensible. La idea es llegar a un consenso.

Consiste en despojar al problema de prejuicios tecnológicos. En lugar de centrarse en soluciones, se analiza el problema desde una perspectiva neutral para centrarse en lo que verdaderamente necesita el negocio. Es fundamental tener en cuenta el caso normal antes de explorar alternativas y excepciones.

-Alternativas: surgen cuando se desea que el usuario tenga opciones de posibles acciones.

-Excepciones: son desviaciones no deseadas pero inevitables del caso normal, por lo que debemos estar preparados. El objetivo es mostrar cómo el escenario maneja de manera segura la excepción.

#Diagramas: Es una alternativa. Muestra cierto grado de procesamiento en paralelo que el texto no permite. También muestra la fusión en el procesamiento (se llega al mismo punto).

Ejemplo:
Nombre BE: Cliente hace un pedido
Nombre y N° BUC: Tomar pedido N°1
Trigger: Cliente se acerca al mostrador
Precondiciones:
Interesados: Cliente, cafetería
Interesados activos: cafetería
Pasos del caso normal:

1- el cliente se acerca al mostrador
2- el cliente ve las opciones
3- el cliente hace el pedido
4- el trabajador registra el pedido en el sistema

Alternativas: 2A) el cliente decide no pedir nada.
Excepciones: 2E) no se cuenta con stock para realizar el pedido, se le ofrece al cliente cambiarlo.
Resultado: El cliente realiza su pedido.

ESCENARIOS NEGATIVOS
Muestra posibilidades negativas o dañinas, como alguien que abusa del trabajo o intenta defraudarlo. Puede ser útil utilizar escritura de ficción protagonista/antagonista. Examine todos los pasos del caso normal y pregunte si existe la posibilidad de que alguien se oponga o haga un mal uso de esa acción.

PRODUCT USE CASE SCENARIO
Se utilizan para comunicar la intención del producto automatizado a los interesados. Es importante que se le comunique a los stakeholders en una reunión, ya que el feedback es importante; para cuando acabe se debe tener una representación certera del producto a construir. Cuando las partes están satisfechas con el escenario del BUC se debe determinar cuánto del BUC se va a automatizar. Los BUC contienen todas las funcionalidades que corresponden al BE, mientras el PUC es solo aquella que será automatizada
Esto se decide en base a la combinación óptima de costo, riesgo y beneficio.

Hay que centrarse en el problema y no en la solución tecnológica.

VALOR
Lo que el cliente estaría dispuesto a pagar por él. 3 factores:

- recompensa: lo que se gana al tener una función específica
- penalización: lo que se pierde al no tenerla
- costo: lo que se debe pagar para obtenerla

El valor depende de la organización. Ejemplo: una organización comercial busca ganancias económicas, una científica exactitud en sus investigaciones.

Para medirlo asigno una puntuación (1-5) a cada factor. Si el valor total es mayor al costo de implementar la funcionalidad, entonces es valiosa y debe incluirse.

- Recompensa >= Costo y Penalización >= Costo 👍

LÍMITES
Hay que cuestionarlos y buscar eliminarlos. Una limitación es una restricción impuesta:

- una política empresarial que indica cómo realizar un proceso
- una directiva de cómo implementar la solución

Problema: las limitaciones se asumen reales e inmutables. Pero, la eliminación de una limitación puede conducir a una INNOVACIÓN.

DESIGN THINKING
Proceso no lineal e iterativo para entender a los USUARIOS, cuestionar supuestos, redefinir problemas y crear soluciones innovadoras a través de prototipos y pruebas.
Se enfoca en el usuario y su verdadera necesidad:
Feasible(posible)-Desirable(necesidad)-Viable(vale la pena)
Se divide en 5 fases (no necesariamente secuenciales):

- empatizar: entender el problema a través de la investigación de usuarios
- definir: analizar las observaciones y definir problemas centrales
- idear: generar ideas creativas que busquen la innovación. De estas se eligen de acuerdo al valor al usuario, facilidad de implementación e impacto en el negocio.
- prototipar: producir versiones reducidas del producto o características específicas para investigar las ideas generadas
- probar: poner a prueba los prototipos de manera rigurosa. Usar los resultados para nuevas iteraciones y refinamientos. Recibir FEEDBACK de usuarios.

BRAINSTORMING
Técnica para generación de ideas, es un proceso de diverengencia (lluvia de idea) y luego convergencia (elección de ideas).

HOW/FUTURE

Una vez que entendimos el negocio y el problema a resolver, pasamos del problema a la solución.
¿Qué parte del problema se beneficia de la automatización/solución?
La solución debe ser adaptable a los usuarios, satisfacer las necesidades operativas, contribuir a los objetivos de la organización y tener un precio justo → si es muy barato genera desconfianza y no le da una retribución acorde al desarrollador. El costo debe ser proporcional al beneficio. A partir del costo, riesgo y beneficio se define el límite del producto.

ITERATIVE DEVELOPMENT

+ Son releases de software para medir la aceptación del diseño y descubrir las necesidades reales de la solución
- Puede llevar mucho tiempo

ESSENTIAL BUSINESS

Las necesidades funcionales y no funcionales pueden ser clasificadas como esenciales o no. Las esenciales son las necesidades reales del negocio.

EXTENSIÓN DEL PRODUCTO

Se trata de encontrar la mejor respuesta: costo bajo en tiempo y esfuerzo, rápido y agradable. El BUC son respuestas a solicitudes del exterior y el producto es lo que elegimos automatizar del BUC. Cuanto menos se implemente, más chico será el valor, al aumentar lo implementado tiene que ser proporcional el valor.

CONSIDERAR AL USUARIO

El producto tiene que ser atractivo para el público objetivo. Debe considerar a los usuarios y lo que es más adecuado para ellos (costumbres, cultura, etc).
Observación:

- Fly on the wall: analizar el contexto sin preguntar
- Mystery shopper: entras a probar productos/servicios, pero nadie sabe que no sos un comprador

USER EXPERIENCE (UX)

Importa más cómo el usuario se siente con el producto que la funcionalidad. Así, es más probablie que los usuarios acepten el producto. El nuevo producto no debe modificar los requisitos esenciales. En este contexto, involucra pensar diferente, innovar:

- conexión: mejor conexión con el cliente a través de los productos o servicios de la empresa. Involucra una buena atención al cliente y recordar datos del cliente.
- conveniencia: valor: que la gente esté dispuesta a pagar más por productos y servicios que ofrecen comodidad. El producto debe apuntar a la facilidad de uso en aquella necesidad del usuario.

- información: los clientes necesitan mucha información y la quieren rápido. Debe informar sin abrumar ni confundir. Pensar en qué le importa al cliente, no códigos internos. Hay que definir los límites de tu producto en base a la información.
- "feeling": los productos se pueden aceptar o rechazar en función a cómo se siente el usuario utilizándolo.

ORIGEN DEL BE

Casi siempre se origina fuera del sistema, cuando sistemas adyacentes hacen algo.
Sistemas adyacentes (se tiene en cuenta si reciben y/o suministran datos):

- Activos: humanos que interactúan con el sistema
- Autónomos: sistemas que no tienen que estar automatizados, que no interactúan directamente con el trabajo. Se comunican de forma unidireccional, no esperan una respuesta.
- Cooperativos: sistemas automatizados que colaboran con el trabajo durante el curso de un BUC, se puede pensar como **parte del sistema**. Son cajas negras. Se comunican de forma bidireccional.

REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES

- Funcional: se centra en el comportamiento. Necesarios para definir y especificar el funcionamiento del software.
- No funcional: demuestran que tan bien el producto hace las cosas que tiene que hacer. El foco no está en el funcionamiento, sino en la operativa. Deben ser medibles, verificables y limitantes a la hora de la construcción. Muy asociado a los sentimientos del usuario.

#limitaciones externas: afectan de forma indirecta al producto. Ejemplo: compatibilidad del SO, regulaciones y leyes.
#Hay requerimientos condicionales que sólo se disparan si la excepción es real. Mientras hay otros requerimientos condicionales que son caminos alternativos al tratarse de variaciones esperadas.
#Hay requerimientos tecnológicos (pseudorrequisitos) necesarios para que el producto funcione correctamente, ligadas a la implementación técnica. Referidos al entorno donde será instalado el sistema.

Caracteristicas:

- no ambiguo: tener una única interpretación
- conciso: lenguaje comprensible referenciando los aspectos importantes
- consistente: ningún requisito debe entrar en conflicto con otro
- completo: no deben remitir a fuentes externas que lo expliquen
- alcanzable: realista con el dinero, tiempo y recursos disponibles
- verificable: verificable si fue satisfecho.

Metodología: determina los pasos a seguir y cómo realizarlos para finalizar una tarea, con el fin de mejorar la productividad en el desarrollo y la calidad del producto software.

Tipos de requerimientos no funcionales:

- **Look and feel**
- **usabilidad**: qué tan fácil de usar es la aplicación. 'el producto debe ser fácil de usar para ingenieros mecánicos', 'el producto debe evitar preguntar al usuario si tiene que duplicar la data ingresada'
- **rendimiento y disponibilidad**: performance de la aplicación, asegurar cierta velocidad o poder de procesamiento. También está la disponibilidad, por ejemplo: "en los primeros 3 meses de funcionamiento, el producto estará disponible el 98% del tiempo entre 8 y 20 hrs"
- **operacional y ambiental**: para operar correctamente en su ambiente, "el celular debe ser usables en condiciones variables de luz", "el celular debe conservar la vida útil de su batería", "el celular debe sobrevivir una caída desde la altura del hombro"
- **mantenimiento y soporte**:"el producto debe estar listo para desplegar en Android y IOS", "los cambios en el código deben poder implementarse en producción en un plazo no superior a 1 día hábil"
- **seguridad**: incluye accesos autorizados, privacidad, integridad (no se corrompe) y auditoría (se pueden verificar las operaciones y movimientos)
- **cultura y política**: "el producto debe usar la configuración teclado para Uruguay"
- **legal**: "el producto debe cumplir con las normas ISO / marco NIST"

QUALITY GATEWAY

Dado un requisito, la actividad de prueba debe demostrar que el producto cumple con ese requerimiento. Para ello se necesita un punto de referencia. Un requisito no puede ser simplemente una descripción de texto, necesita una razón y un criterio adecuado. Ejemplo: tener encuesta 6 meses después del lanzamiento no resulta conveniente.

Pruebas subjetivas: Hay que tratar que la información que se pueda relevar sea lo más temprano posible y cuando hay un tester probando las cosas, está parcializado y es diferente a los usuarios finales. Por eso se hacen paneles de prueba que simulan esta realidad, para que si falla algo, se pueda tomar acciones rapido. También está el juicio de expertos, por ejemplo: "El producto debe ser certificado según los estándares de marca corporativa por el jefe de comunicaciones".

Estándares: cuando existen, es bueno citarlos, estableciendo un punto de referencia conocido y sin desacuerdos.

- Internos: definidos por algún área de la organización

- Externos: basados en alguna norma ISO, IEEE, Corba, etc.

#Para seguridad se usan varios estándares: JWT, SSO, HTTPS, x509, OWASP TOP 10, etc

En el caso de los requisitos funcionales, se verificará su criterio de aceptación con casos de prueba.

QG
Intenta asegurarse de que cada requisito sea lo más perfecto posible antes de incluirlo en la especificación. Tienen en cuenta que respeten el alcance del proyecto y la relevancia del mismo. Los requisitos los valido por estándares internos o externos y por los números (performance y magnitudes).

Gold plating: características o requisitos innecesarios que contribuyen más al costo de un producto que a su funcionalidad o utilidad. ¿importa si el requisito no se incluye?

Requirements Creep: cuando se agregan nuevos requisitos a la especificación después de que se consideran completos. Este fenómeno puede tener un impacto serio en el presupuesto del proyecto (30% del costo total). Es importante asegurarse de que cada requisito sea relevante tanto para el propósito del producto como dentro del alcance del trabajo.