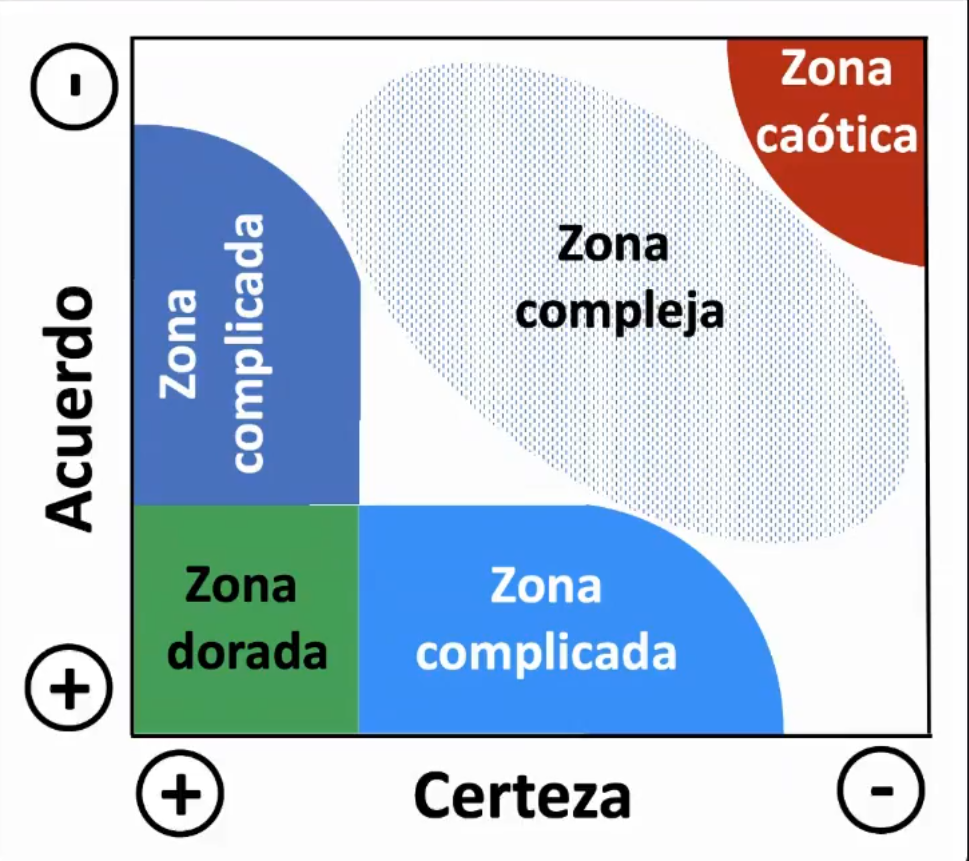
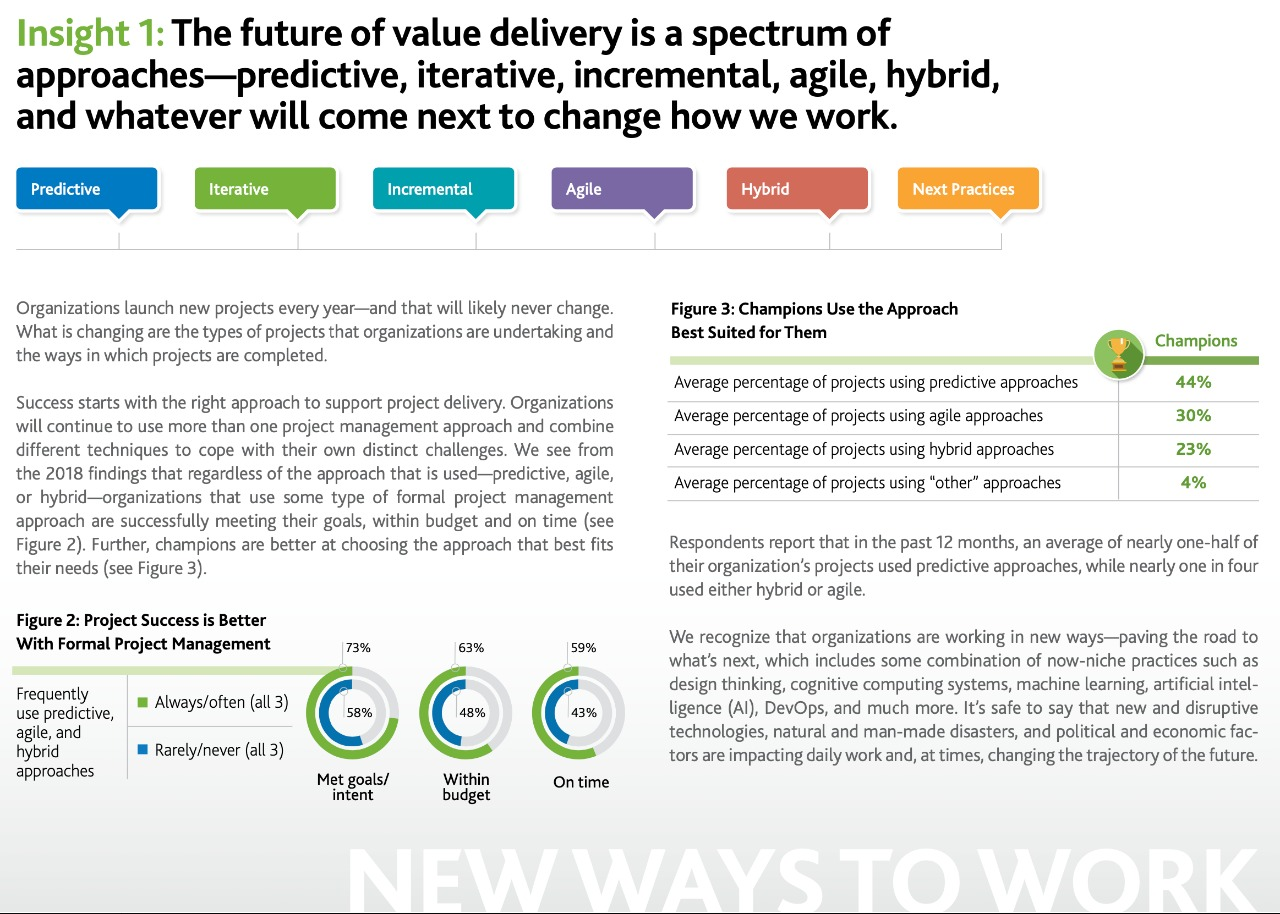
Un análisis de la complejidad llamado rack stacy en los años noventa





Matriz compleja de Stacey

Zona dorada: el mejor de los mundos

**La certeza es cuando el problema lo conocemos y sabemos cómo actuar sobre eso, y los involucrados están de acuerdo**

En zona complicada (en acuerdo): entre los expertos no hay acuerdo de como resolver el problema

Zona complicada (en certeza): hay acuerdo, mismo nivel de acuerdo, hay acuerdo de lo que conocemos, pero lo conocemos es menos. Es decir, la falta de conocimiento

Zona Caótica: el peor de los mundos. Implica desconocimiento total del problema que nos enfrentamos y por otro lado, no estamos de acuerdo sobre lo que no conocemos. Ejemplo: al trtara de conocer lo que no conocemos, apaarce uan rivalidad que impide a saber mas del problema de forma colectiva. ~~No debemos estar acá y Si estamos aca lo mejor es renunciar a hacer lo que tenga sentido min 9:46~~.En este punto es imposible gestionar entonces e esa zona debemos salir.

Zona compleja: acá ninguna de las cosas son fáciles, es pero se salió del caos. Acá lo mejor es hacer estrategias y planes de cortos plazos o pasos cortos como quien dice, decisiones de corto plazo, revisando en cada paso si es mejor o que hay acomodar.

Zona Dorada: zona poco probable, pero puede ocurrir.

**La zona compleja es la más probable a cumplirse.**

A veces en certeza aparece cosas tecnológicas, SI MI CONOCIMIENTO EN TECNOLOGIA APLICA ES MAYOR O MENOR

Y en acuerdo aparece requerimiento, SE EXPRESA EN CUANDO AL GRADO DE REQUERIMIENTO TIENEN UN ENTENDIMIENTO COMUN CON LOS STAKEHOLDERS

En agiles, por ejemplo, el PO de scrum es que se sabe del negocio, este hace avanzar de forma más segura. El responsable es el que tiene conocimiento, y no solo el que se hace cargo.

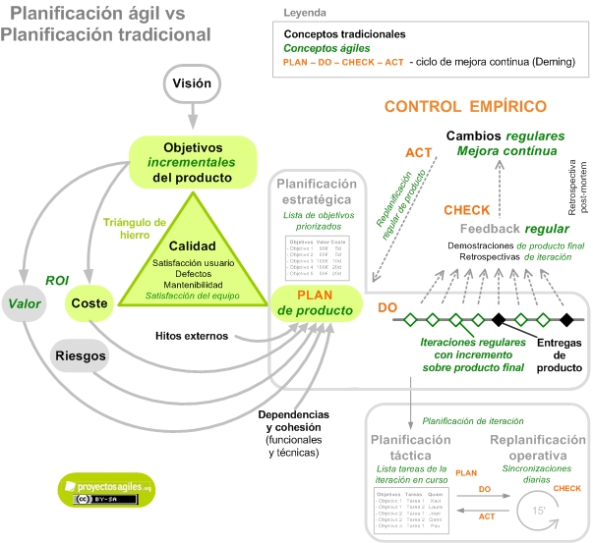
**Planificación tradicional**

Triangulo de hierro: 

El plan de proyectos contiene unas determinadas o serie de entregas, que permitirá la replanificación de las tareas

El control predictivo, es más fuerte (fuerte por que todos los proyectos tienen planes) por que el plan de proyecto completo y no de las próximas dos semanas, por ejemplo

**Planificación ágil**



A diferencia de la tradicional:

* Acá no se tiene los objetivos del proyecto completo
* Se tiene un plan de cada paso y esos planes se elaboran después un paso tras otro... plan de lo que se hace ahora
* Control empírico: ver si ocurre o no
* Hay iteraciones incrementales, que se pueden llamar sprints o en otras metodologías tiene otro nombre, siempre es orientado al producto final
* **Check**, relacionado con las introspectivas, los cambios son continuos
* **Act**: de acuerdo al resultado del chequeo se hará la próxima replanificación. se tiene un grado de preelaboracion

Entonces se tiene una lista estratégica de objetivos realizados, después una planificación táctica de cada iteración que se realice

En agiles siempre se hace lo que tiene más valor, también que un proyecto tradicional, aunque, lo que más vale, primero, aunque lo importante también puede estar enfocado en el final

Entonces este enfoque tiene impacto en el desempeño como como en los resultados finales de las contingencias

Aspectos de Calidad

La calidad importa a todas las metodologías

CMMI: un framework de mejora, no es muy relevante

El tema de calidad tiene que ver con el ciclo de Deming (PDCA)

**Conceptos de calidad**

***El ciclo de mejora continua de Deming (PDCA)***

En el ciclo de mejora continua de Deming tiene las siguientes actividades [1]:

* [**P**lan] **Planificar** cómo conseguir unos objetivos.
* [**D**o] **Ejecutar** esta planificación.
* [**C**heck] **Verificar** los resultados conseguidos.
* [**A**ct] Definir **acciones correctoras** a realizar en el siguiente ciclo para mejorar los resultados.

**Cumplimiento de la expectativa del cliente:** difícilmente se logren buenos resultado si este rol no esta bien desempeñado, se recomienda que en un Product Owner, si no lo tiene, se debe lograr tener a esta persona, aunque no se asegura que el proyecto no se pueda lograr, PO es una figura critica del proyecto, se puede crear el PO pero lleva su tiempo

Algo que no es propio ni exclusivo de las metodologías agiles es el aspecto de la **calidad externa**, el producto tiene que funcionarse como se espera, lo que se espera que haga,

**Satisfacción del equipo**: el orgullo tiene que mejorar la calidad, esto tiene consecuencias en las relaciones de trabajo

***Las dimensiones de la calidad***

Simplificando el modelo de calidad ISO 9126 [3], podríamos considerar que la calidad tiene las siguientes dimensiones:

1-     **Cumplimiento de expectativas del**[**Cliente (Product Owner)**](https://proyectosagiles.org/cliente-product-owner).

Yendo un poco más lejos y utilizando el concepto de Lean de mejorar el proceso de producción de valor (analizar desde la concepción de la idea hasta que alguien la utiliza o paga por ella, cuando se recupera la inversión realizada), más que cumplir con las expectativas del cliente, hay que **cumplir con las expectativas del consumidor o usuario final de producto**, que es quien va a comprarlo o verse beneficiado en su utilización. Por ello, para no crear un producto que nadie va a comprar o una aplicación que nadie va a utilizar, el cliente debería conocer quién es y qué necesita el consumidor o usuario final, hacer participar a personas representativas, investigar el modelo de negocio (cómo va a ganar dinero) y contar con las ideas y aportaciones del [equipo](https://proyectosagiles.org/equipo-team).

2-     Calidad externa, **funcionamiento correcto**. El producto debe comportarse de la manera esperada, tanto a nivel funcional como no funcional (rendimiento, usabilidad, etc.).

3-     Calidad interna. El producto debe ser **mantenible**, debe poder evolucionar a un ritmo sostenido.

En el encuentro se añadió, desde una perspectiva ágil, una cuarta dimensión:

4-     **Satisfacción del equipo**, que esté orgulloso de su trabajo, para mantener su motivación, su compromiso con el proyecto y con la empresa.

Atributos relacionados a los requerimientos

* Verificabilidad: el producto final sebe cumplirse de acuerdo al requerimiento (enfoque tradicional)
* Integración continua:

**Agilidad, calidad y mejora continua**

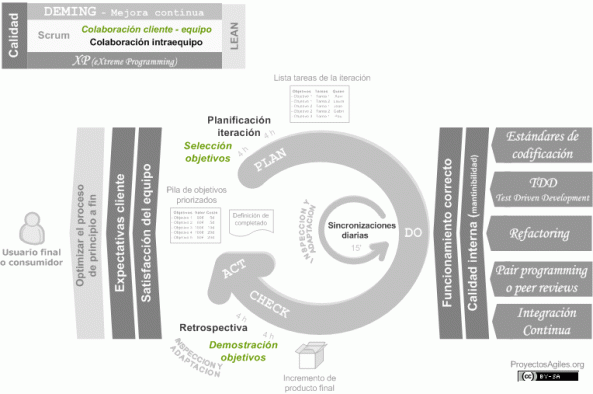
Scrum [2] es un proceso que en bloques temporales cortos y fijos (entre 2 y 4 semanas) entrega un incremento de producto final. En cada iteración de Scrum se ejecuta el ciclo **de mejora continua** de Deming en los siguientes ámbitos:

* Las **expectativas del cliente**, a las cuales se va acercando iteración a iteración mediante las siguientes actividades:
  + La [planificación al inicio de la iteración](https://proyectosagiles.org/planificacion-iteracion-sprint-planning), en que el equipo se reúne con el cliente, quien ha expresado su expectativas mediante la [lista priorizada de objetivos del proyecto (product backlog)](https://proyectosagiles.org/lista-requisitos-priorizada-product-backlog). El equipo coge tantos objetivos como se vea capaz de cumplir en la iteración, los refina con preguntas al cliente y hace una primera descomposición en las tareas necesarias para completar cada uno de ellos, creando la [definición de hecho](https://proyectosagiles.org/definicion-de-hecho-definition-of-done/) y la [lista de tareas de la iteración (sprint backlog)](https://proyectosagiles.org/lista-tareas-iteracion-sprint-backlog). [Es importante que el cliente haya preparado las iteraciones para que tengan una meta clara, que no sean un conjunto de objetivos poco cohesionados que produzcan pocas sinergias].
  + La iteración se desarrolla orientada a objetivos, minimizando el “número de objetivos en curso” (WIP, Work In Progress), para tener alguno completado cuando acabe la iteración y no muchos iniciados o a punto de finalizar.
  + La [demostración](https://proyectosagiles.org/demostracion-requisitos-sprint-review) de los resultados completados que el equipo realiza al cliente al final de la iteración. El feedback recibido del cliente permite ir acercándose a sus expectativas mediante [ajustes a realizar en siguientes iteraciones](https://proyectosagiles.org/replanificacion-proyecto), bien por que el equipo no consiguió entender correctamente sus necesidades, bien por que es el cliente quien ahora entiende mejor el producto, bien por que la velocidad de desarrollo no es la esperada o bien por cambios externos al proyecto que obligan a cambiar objetivos. Estos ajustes se incorporan al product backlog y el cliente lo reprioriza.

Notar que **dentro de la propia iteración hay otro ciclo de Deming**: la [reunión diaria de sincronización del equipo](https://proyectosagiles.org/reunion-diaria-de-sincronizacion-scrum-daily-meeting), que le permite inspeccionar lo que está sucediendo y realizar las adaptaciones necesarias para poder conseguir los objetivos de la iteración, añadiendo o quitando tareas del sprint backlog durante la iteración.

Se puede observar que uno de los principales beneficios de Scrum es la [**flexibilidad**frente a cambios en el contexto](https://proyectosagiles.org/beneficios-de-scrum#flexibilidad-adaptacion), el hecho de poder modificar y repriorizar los objetivos del proyecto iteración a iteración, así como las de tareas para poder conseguir un objetivo (se pueden llegar a cambiar todas las tareas si el planteamiento inicial ya no es aplicable).

* Los**procesos de trabajo del equipo**, mediante:
  + La [replanificación de las tareas a ejecutar](https://proyectosagiles.org/replanificacion-proyecto).
  + La [retrospectiva](https://proyectosagiles.org/retrospectiva-sprint-retrospective) de cómo fue su ejecución, qué cree que hay que mantener, mejorar o probar hacer de manera diferente, con lo que se generar objetivos y tareas de ajuste de los procesos a ejecutar en las siguientes iteraciones.
* La **satisfacción del equipo** se consigue mediante:
  + El hecho de poder entregar resultados finales de forma regular, de forma que el cliente los pueda utilizar o entregar a los usuarios/consumidores finales.
  + El mejor entendimiento de las expectativas del cliente, que permitirá dar mejores resultados en las siguientes iteraciones.
  + La mejora continua del proceso de trabajo entre el equipo y con terceros, cosa que le permite ser más productivo, reducir re-trabajo y tareas de poco valor.



**Agilidad y calidad interna**

Es necesario construir el producto con **calidad interna** para poder tener una **velocidad de desarrollo sostenible** y crecer de manera iterativa e incremental. Ya desde la segunda iteración se debe construir sobre la parte del producto final ya existente sin poner parches que al final se transformen en “deuda técnica” (*Technical Debt*). Si la deuda técnica fuese creciendo, al final el coste de ir incrementando la funcionalidad se haría enorme. Para ello, eXtreme Programming utiliza las siguientes prácticas de ingeniería:

* **Estándares de codificación**, incluyendo código bien comentado, para facilitar la propiedad colectiva.
* **Pair programming** y/o **peer review**, para conseguir un código más **simple, fácil de mantener** **y revisado**por otra persona, con lo que **la probabilidad de errores se reduce** (arquitectónicos, de diseño o de funcionamiento).
* **Arquitecturas flexibles**tanto en interfaz de usuario como en patrones de diseño como en información, de manera que en el futuro se puedan introducir cambios minimizando el impacto sobre el resto del sistema / los objetivos ya completados. Esto no debe implicar preparar la arquitectura o el código para requisitos futuribles (que puede que no lleguen nunca), si no simplemente cubrir cada objetivo conforme se va desarrollando intentando no hacer demasiado en contra de los futuribles.
* **TDD** (Test Driven Development) para desde el inicio **conseguir un código se puede probar**así como su **simplicidad**. Para ello se escribe primero las pruebas (funcionales, de integración o unitarias) y después se escribe el mínimo código necesario para pasarlas.
* **Refactorización**, para **simplificar** **y mejorar el código**una vez la prueba ha pasado, con la garantía de que sus pruebas asociadas actuarán como regresión protegiendo de la introducción de errores.
* Ejecutar **pruebas automatizadas en un sistema de integración continua**para poder realizar **cambios controlados**. De esta manera, si algún componente que ya existía deja de funcionar, las pruebas asociadas inmediatamente detectan el fallo tan pronto se integra. Dado que estos errores colaterales no tardan en ser detectados, su corrección es más rápida ya que el desarrollador todavía tiene en mente el código que los produjo.

Es importante contar con [**métricas**](https://proyectosagiles.org/metricas-agiles-cuadro-mandos-balanceado-scrum) para ir evaluando de manera objetiva si cómo se están produciendo las mejoras [8].

ROI: para decidir cual proyecto encarar primero, esto permite si debo invertir o no, analiza el rendimiento desde el punto de vista financiero. Compara cuanto nos va a beneficiar de acuerdo a lo que vamos a gastar en el proyecto

Ejemplo proyecto=100, beneficio=80, el 20 seria a favor de lo que puse

* No depende del tiempo
* A veces erróneamente se la hace de forma anualizada, o mensual, por lo que no debe compararse así el ROI, pero si el ROI de un proyecto a 3 meses, no debe de usarse el mismo para un año o para extender el tiempo que sea
* Periodo de retorno si, pero no tiene que ver el tiempo, solo en la RECUPERACION a través del beneficio
* La ventaja que logran los clientes que trabajan con nosotros,
* TIR !=VAN , importantes a la hora de realizar una inversión

Min h:11 hora kahhot