Contratos ágiles

Marcos Bermejo

PID_00198058

UOC
Universitat Oberta de Catalunya

Los textos e imágenes publicados en esta obra están sujetos –excepto que se indique lo contrario– a una licencia de Reconocimiento-NoComercial-SinObraDerivada (BY-NC-ND) v.3.0 España de Creative Commons. Podéis copiarlos, distribuirlos y transmitirlos públicamente siempre que citéis el autor y la fuente (FUOC. Fundación para la Universitat Oberta de Catalunya), no hagáis de ellos un uso comercial y ni obra derivada. La licencia completa se puede consultar en http://creativecommons.org/licenses/by-nc-nd/3.0/es/legalcode.es

Índice

1. ¿Cuál es el propósito de un contrato? 5

2. *Win-win* 6
2.1. ¿Cómo evaluar los contratos? 7
2.2. ¿Qué información se debe incluir en un contrato? 7

3. Modelos de contrato 9
3.1. Contrato de *sprint* 9
3.2. Precio fijo/alcance fijo 10
3.3. *Time* & *materials* 11
3.4. *Time* & *materials* con alcance fijo y techo de gasto 11
3.5. Desarrollo por fases 12
3.6. Cláusulas de bonus y penalizaciones 13
3.7. Ganancia fija 14
3.7.1. *Money for nothing. Changes for free* 15

4. Conclusiones 17

Bibliografía 19

1. ¿Cuál es el propósito de un contrato?

Si bien es cierto que el *Agile Manifesto* valora la colaboración con el cliente por encima del contrato, en ocasiones es necesario establecer un conjunto de reglas que delimiten un determinado proyecto. La adecuación de estas reglas al proyecto ágil puede significar aumentar las posibilidades de éxito para ambas partes.

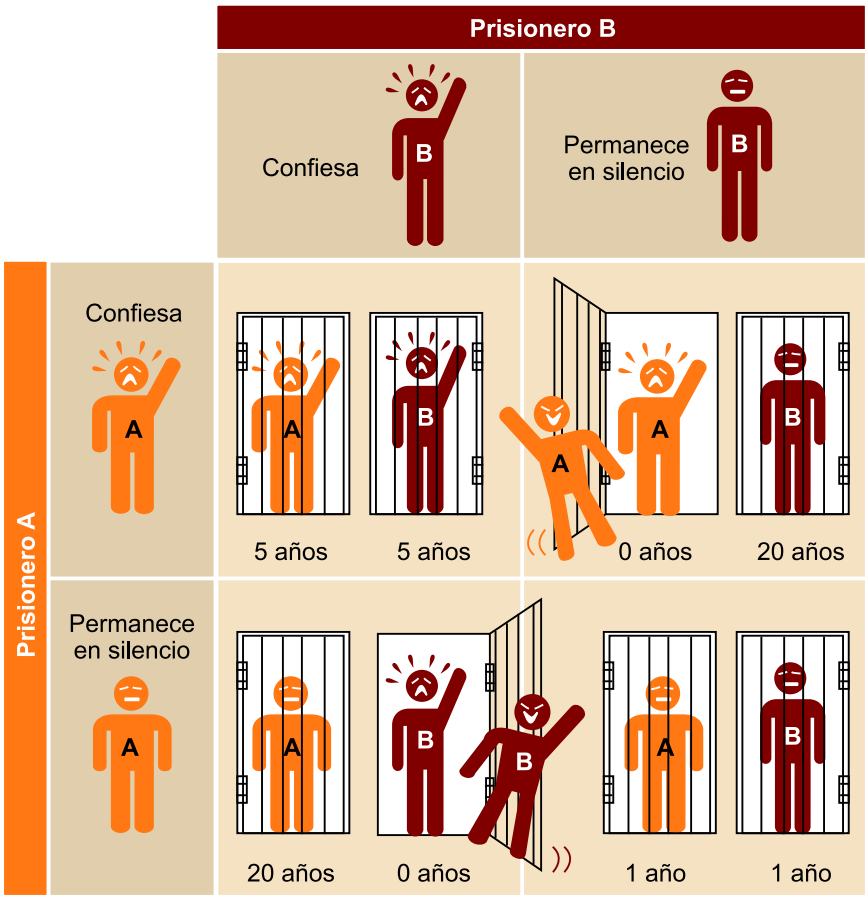
En teoría, estas reglas las acuerdan libremente ambas partes para crear condiciones óptimas que permitan completar con éxito el proyecto. En la práctica, los contratos se suelen ver como juegos competitivos, cuyo objetivo es dejar en desventaja a la otra parte. Muchas organizaciones grandes y gobiernos tienen condiciones comunes que se deben aceptar en conjunto como prerrequisito para hacer negocios con ellos.

Estas condiciones casi nunca son justas, por lo que el resultado de un proyecto depende en gran medida de la buena relación con el cliente para así evitar discutir el contrato o recurrir a la ley.

2. *Win-win*

El equilibrio de Nash es, en teoría de juegos, un **concepto de solución** para juegos con dos o más jugadores en el que todos ellos han puesto en práctica, y saben que lo han hecho, una estrategia que maximiza sus ganancias dadas las estrategias de los demás.

Un buen ejemplo es el dilema del prisionero, que muestra que dos personas pueden no cooperar incluso si se juegan el interés de ambas.

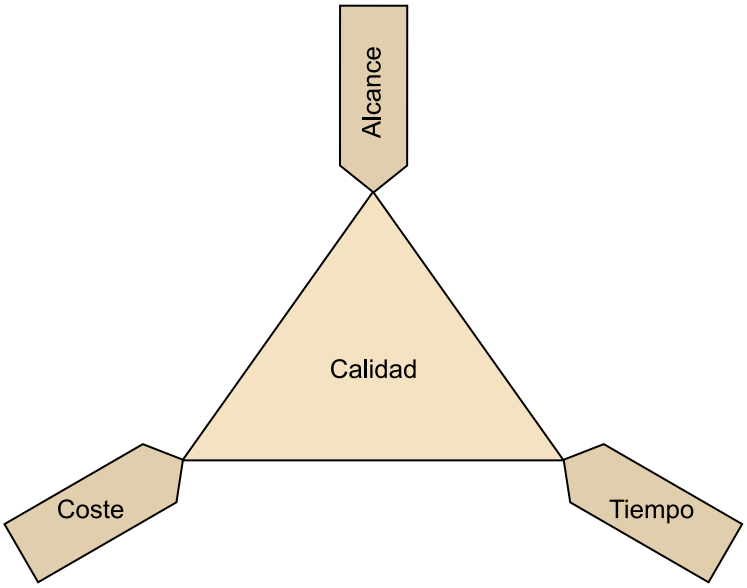


El dilema del prisioneroFuente: Adaptado de la *Encyclopaedia Britannica* (2006)

La estrategia más inteligente es la de maximizar el bien común, lo que llamamos un ***win-win***.

Un contrato distribuye el riesgo y refleja la confianza entre las partes. ¿Qué pasa cuando algo sale mal? ¿Quién paga cuando el proyecto es más difícil de lo que se esperaba? ¿Quién se beneficia si el proyecto acaba antes de lo previsto?

Usar reglas incorrectas puede ser perjudicial para el éxito del proyecto. Las reglas malas llevan a **precios** no realistas, **tiempos** imposibles o expectativas funcionales (**alcance**) que no se cumplen. Los juegos *win-lose* son perjudiciales para el éxito del proyecto. La calidad casi siempre sufre.



2.1. ¿Cómo evaluar los contratos?

Los contratos comerciales pueden tener muchas formas. En cualquier contrato se tiene que mirar:

- ¿Cómo está estructurado? ¿Cuáles son las reglas básicas para el alcance de las entregas y la factura por ingresos?
- ¿Cómo se reparten los riesgos y los beneficios entre cliente y proveedor?
- ¿Cómo se gestionan los cambios a los requerimientos?
- ¿Qué modelo de relación con el cliente fomenta? ¿Competitividad (*win-lose*), cooperación (*win-win*), indiferencia o dependencia?

2.2. ¿Qué información se debe incluir en un contrato?

Cuanta más confianza exista entre cliente y proveedor, menos se necesita escribir en el contrato.

Algunos puntos que tendrían que aparecer siempre en cualquier contrato son los siguientes:

1) Objetivos del proyecto y de la cooperación entre las organizaciones.

2) Un resumen de la estructura del proyecto: procesos de Scrum y funciones clave, entre otros.

3) Personal clave: quién es responsable a escala operacional y jerárquica.

4) Pagos y facturación, incluyendo las cláusulas de bonus y penalizaciones.

5) Finalización normal y finalización antes de la fecha fijada en el contrato.

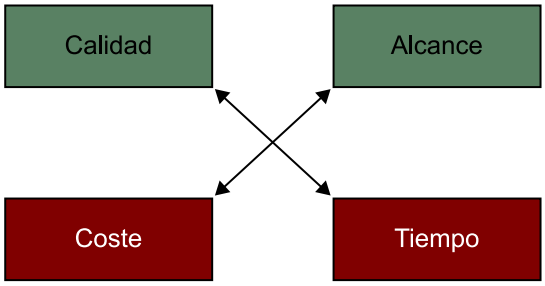
6) Detalles legales. En función de las leyes locales y costumbres legales, es posible que tenga que limitarse la responsabilidad civil, especificar la garantía o incluir otros textos que se necesitan legalmente.

¿Es necesario incluir el alcance del proyecto en un contrato? A pesar de que a menudo está presente, fijar el alcance en el contrato también lo hace inflexible. Si fuera posible, es mejor especificar cómo se va a gestionar el alcance (por ejemplo, con un *backlog* de producto, contratos por *sprint*) y dejar los detalles operacionales fuera para que los gestione el equipo del proyecto.

Los puntos 2, 3, 4 y 5 determinan las reglas de juego para el proyecto. Si son las correctas, tendremos una base sólida para un buen proyecto.

3. Modelos de contrato

3.1. Contrato de *sprint*



Para aquellos que trabajan con Scrum, la metáfora de contrato de *sprint* puede ser útil para comprender la relación entre el *product owner* y el equipo de trabajo.

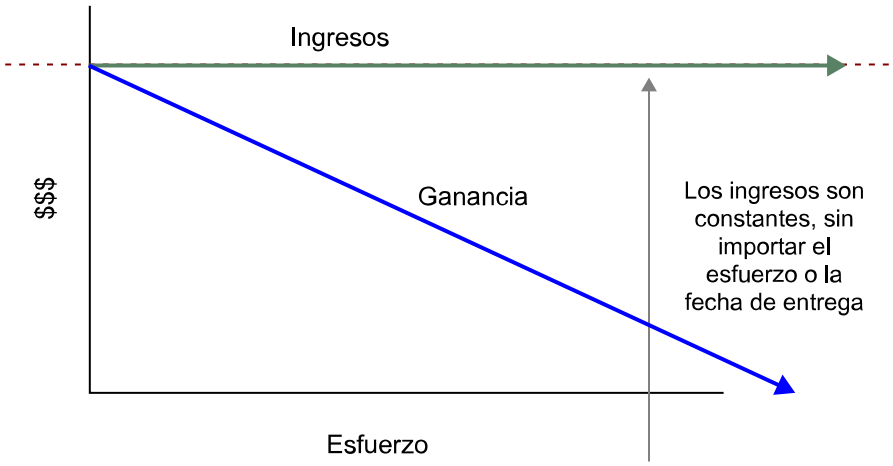
- **Estructura**: en realidad, no es un contrato comercial, sino simplemente un acuerdo entre el *product owner* y el equipo para un *sprint*.

- **Alcance**: el equipo acuerda esforzarse al máximo para entregar el conjunto de características acordadas (alcance) con un estándar de calidad definido al finalizar el *sprint*. El *product owner* acuerda no cambiar las instrucciones antes de finalizar el *sprint*.

- **Riesgo**: un proyecto de Scrum puede ser visto como una serie de miniproyectos con parámetros fijos:
  - tiempo (duración del *sprint*),
  - alcance (*backlog* del *sprint*),
  - calidad (*definition of done*), y
  - gastos (tamaño del equipo × duración del *sprint*).

  Tan solo se puede cambiar el alcance y esto se mide en cada *sprint*.

3.2. Precio fijo/alcance fijo



- **Estructura**: se acuerdan los entregables, se entregan. Se envía la factura. A los clientes les gustan los proyectos de precio fijado porque les da la sensación de seguridad.

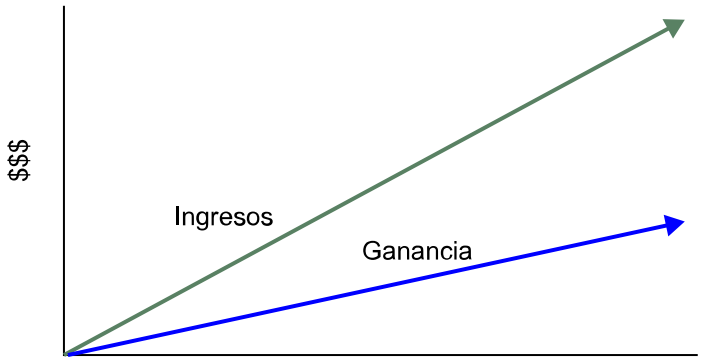
- **Cambios de alcance**: el proceso de solicitud de cambios está pensado para limitar el alcance de los cambios. Este proceso es costoso y los cambios en general no se pueden prever. Como el cliente casi por definición quiere más alcance, resulta difícil acabar el proyecto. El proveedor quiere que el cliente esté contento, por lo que el proveedor suele ceder. La palabra *etcétera* es muy peligrosa para definir especificaciones en un requerimiento de precio fijado.

- **Riesgo**: obviamente todo el riesgo es para el proveedor. Si la estimación es incorrecta, el proyecto pierde dinero. Otros riesgos menos obvios surgen del juego de petición de cambios, en el que el proveedor negocia ingresos adicionales para cambios en el alcance. Si el proveedor subestima muy mal el esfuerzo adicional, o da un precio irreal muy bajo, las pérdidas pueden llegar a afectar a la propia existencia del proveedor, lo que también es un problema para el cliente.

- **Relación**: competitiva. Los clientes generalmente quieren tener más y los proveedores hacer menos. El proveedor quiere que el cliente esté contento, por lo tanto el proveedor acostumbra a ceder.

Consejo: especificar los requerimientos funcionales con historias de usuario.

3.3. *Time* & *materials*



- **Estructura**: trabajar un mes, después enviar la factura al cliente.

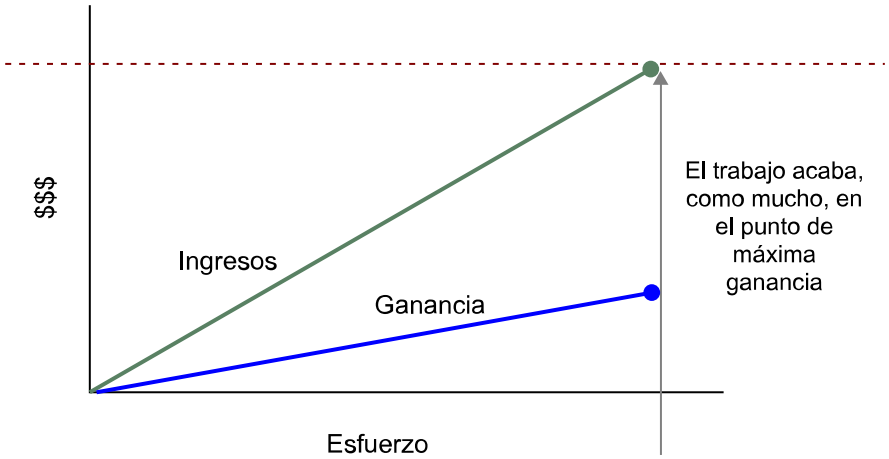
- **Alcance**: no se fija con prioridad. Tarde o temprano, el cliente no querrá pagar más y el proyecto llegará a su fin.

- **Riesgos**: los riesgos los asume 100% el cliente. Los proveedores tienen poca iniciativa para mantener costes bajos. Puede haber un esfuerzo importante para asegurarse de que solo se facturan esfuerzos y gastos legítimos.

- **Relación**: indiferente. El proveedor es feliz cuando hay más trabajo porque más trabajo significa más dinero.

Consejo: recomendado para proyectos donde el cliente puede manejar los riesgos mejor que el proveedor. Este contrato suele combinarse con un techo de gasto. En función de cómo se gestione el alcance, funcionará mejor o peor.

3.4. *Time* & *materials* con alcance fijo y techo de gasto



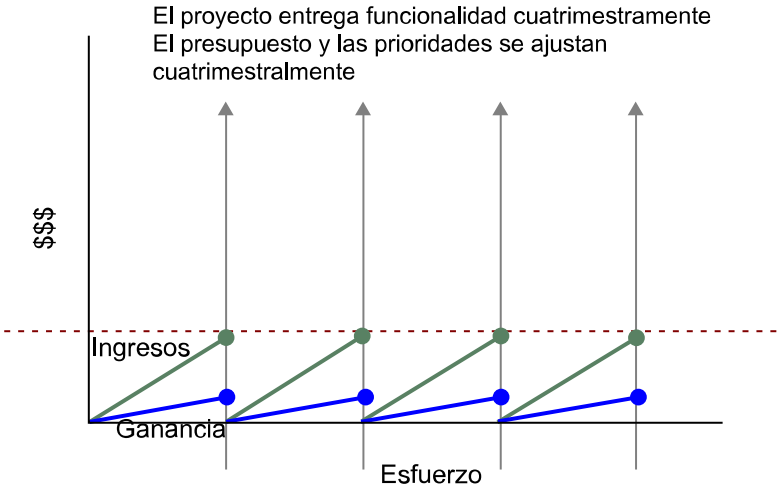
- **Estructura**: igual que el de precio fijo/alcance fijo, excepto que, si el proveedor acaba antes, el proyecto cuesta menos porque solo se factura el esfuerzo real.

- **Alcance**: igual que el de precio fijo/alcance fijo.

- **Riesgos**: desde el punto de vista del cliente, este contrato parece representar lo mejor de ambos mundos. Si se necesita menos esfuerzo del esperado, el proyecto tiene un coste menor. Y una vez se llega al techo de gasto, se comporta como un proyecto de coste fijo.

- **Relación**: dependiente. Desde un punto de vista del proveedor, el objetivo es lograr exactamente el techo de gasto. No hay ningún incentivo para que el cliente entregue por debajo del coste máximo presupuestado. El cliente probablemente hubiera tratado el proyecto como coste fijo si fuera interno, por lo que tampoco tiene ningún incentivo para acortar el alcance y ahorrar dinero.

3.5. Desarrollo por fases



- **Estructura**: financiar entregas cuatrimestrales y aprobar fondos adicionales después de cada entrega exitosa.

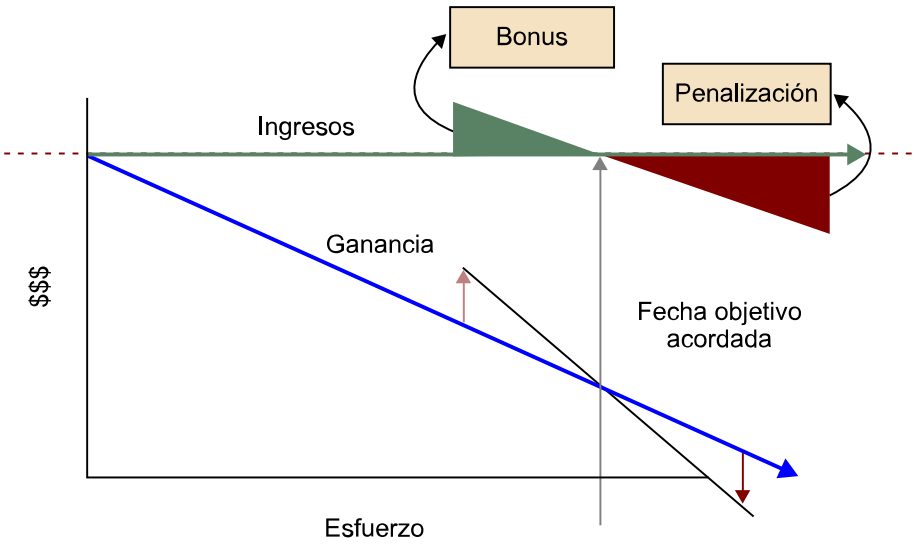
- **Cambios en el alcance**: no se definen explícitamente en este modelo. Las entregas están pautadas en el tiempo. Saber que habrá otra entrega el próximo cuatrimestre hace que resulte más fácil posponer una característica para lograr la entrega en el tiempo pautado.

- **Riesgo**: el riesgo del cliente se limita al coste de un único cuatrimestre.

- **Relación**: cooperativa. Tanto el cliente como el proveedor tienen un incentivo para que cada entrega sea exitosa, de forma que se aseguren fondos para las próximas entregas.

Consejo: los inversores de riesgo suelen trabajar de este modo. Es una buena mezcla de *time* & *materials* con alcance variable y techo de gastos. En el contrato comercial simplemente se especifica el objetivo de la entrega, los costes por hora y el techo de gasto. El cliente provee al *product owner*, todo el resto se determina en los contratos de *sprint*.

3.6. Cláusulas de bonus y penalizaciones



- **Estructura**: el proveedor recibe un bonus si el proyecto acaba antes y paga una penalización si acaba más tarde. La cantidad del bonus o la penalización está en función de la demora.

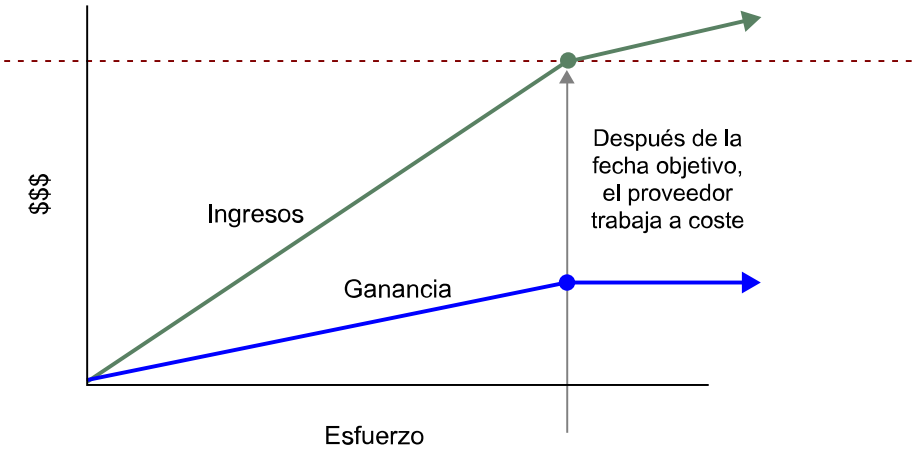
- **Cambios en el alcance**: el cambio es difícil de aceptar porque un cambio puede impactar potencialmente en la fecha de entrega, lo que se quiere evitar.

- **Riesgo**: ¿cuál es el incentivo del cliente para que se acabe antes? Se tienen que tener argumentos de ROI fuertes y transparentes. En caso contrario, el cliente tiene una salida facilista cuando las cosas llevan más tiempo.

- **Relación**: podría ser cooperativa, pero puede degenerarse a indiferente si el cliente no necesita realmente el software para la fecha pactada.

Consejo: este contrato se usa a menudo en proyectos de construcción (por ejemplo, carreteras y túneles), donde sí funciona bien. El cambio de alcance no es un problema y hay motivos económicos genuinos para que el cliente llegue a las fechas.

3.7. Ganancia fija



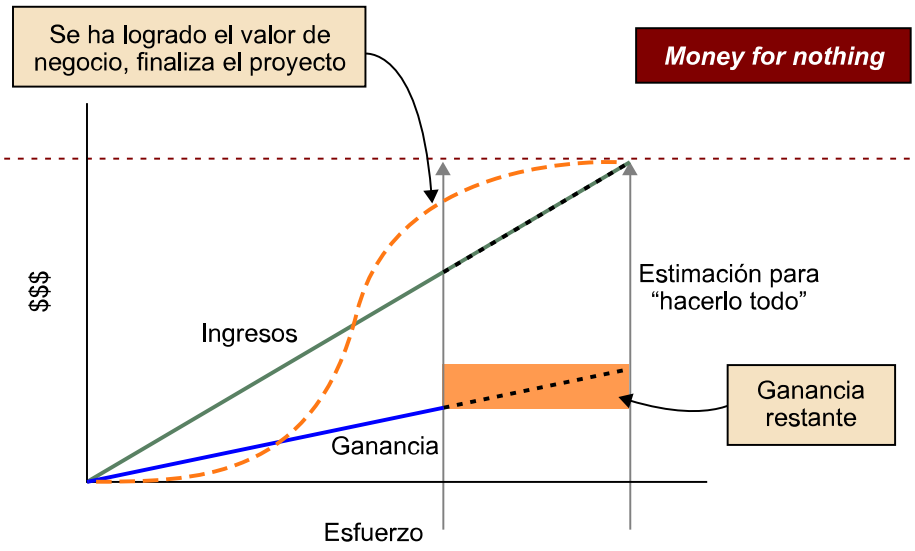
- **Estructura**: cualquier presupuesto de proyecto consiste en costes efectivos y ganancia. Las partes acuerdan la ganancia por adelantado (por ejemplo, 100.000 €). Sin importar cuándo se acaba el proyecto, el proveedor recibe ingresos por el gasto incurrido más la ganancia fija acordada.

- **Alcance**: es fijo.

- **Riesgo**: es compartido. Si el proyecto acaba antes, el cliente paga menos y el proveedor obtiene igualmente su ganancia. Si el proyecto se excede de presupuesto, el cliente paga más y el proveedor no tiene ganancia adicional. Después de la fecha de entrega objetiva, el proveedor no puede facturar más ganancias, solo cubrir los costes.

- **Relación**: cooperativa. Tanto el cliente como el proveedor tienen un incentivo claro para acabar antes. El cliente quiere ahorrar dinero y el proveedor tener un margen de ganancia mayor.

3.7.1. *Money for nothing. Changes for free*

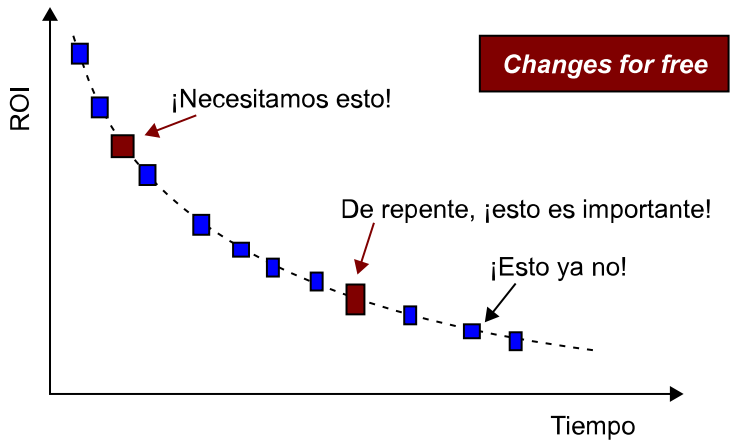


- **Estructura**: este contrato funciona con los proyectos de software ágiles porque casi no hay trabajo en progreso. Al final de cada *sprint*, la funcionalidad está acabada o no se empezó. El trabajo es del tipo *time* & *materials* con un objetivo de coste, a menudo con la intención de que el proyecto no use todo el presupuesto. Una vez se entrega cierta cantidad de funcionalidad, el cliente puede darse cuenta de que ya se ha entregado una cantidad suficiente de valor de negocio y que no se necesita más desarrollo, por lo que puede cancelar el proyecto. Hay un gasto de cancelación que es igual a la ganancia restante que faltaba.

- **Alcance**: puede cambiar. Las características planificadas sin implementar se pueden reemplazar por otras historias del mismo tamaño. Las características adicionales cuestan más.

- **Riesgo**: compartido. Ambas partes están interesadas en completar antes el proyecto. El cliente obtiene costes inferiores y el proveedor, un margen más amplio.

Consejo: si se excede el presupuesto, se pueden aplicar las reglas de los contratos de ganancia fija o techo de gasto. El enfoque de ganancia fija es más consistente con el objetivo de fomentar una relación cooperativa entre las partes.



* Siempre que no se supere la suma de horas restantes

4. Conclusiones

Un contrato prepara el terreno de un proyecto exitoso. Y el *Agile Manifesto* lo dice muy claro: trabajar con el cliente es más importante que el contrato. De esta forma, independientemente de lo que se haga, se tiene que mantener una relación positiva con el cliente.

El modelo *Money for nothing. Changes for free* convierte las ventajas de Scrum y el desarrollo ágil en una ventaja competitiva:

- Al priorizar y entregar valor de negocio de forma incremental, se reducen dramáticamente las probabilidades de un fracaso total.

- Es un modelo cooperativo con el que se incentiva a ambas partes a mantener los gastos bajos.

- La cláusula de cancelación temprana recompensa la mayor productividad que consigan los equipos de Scrum.

Bibliografía

Agile Contracts

Larman, Craig; Vodde, Bas (2010). *Practices for Scaling Lean and Agile Development: Large, Multisite, and Offshore Product Development with large-Scale Scrum*. Addison-Wesley. Disponible en: http://www.amazon.es/practices-scaling-lean-agile-development/dp/0321636406.