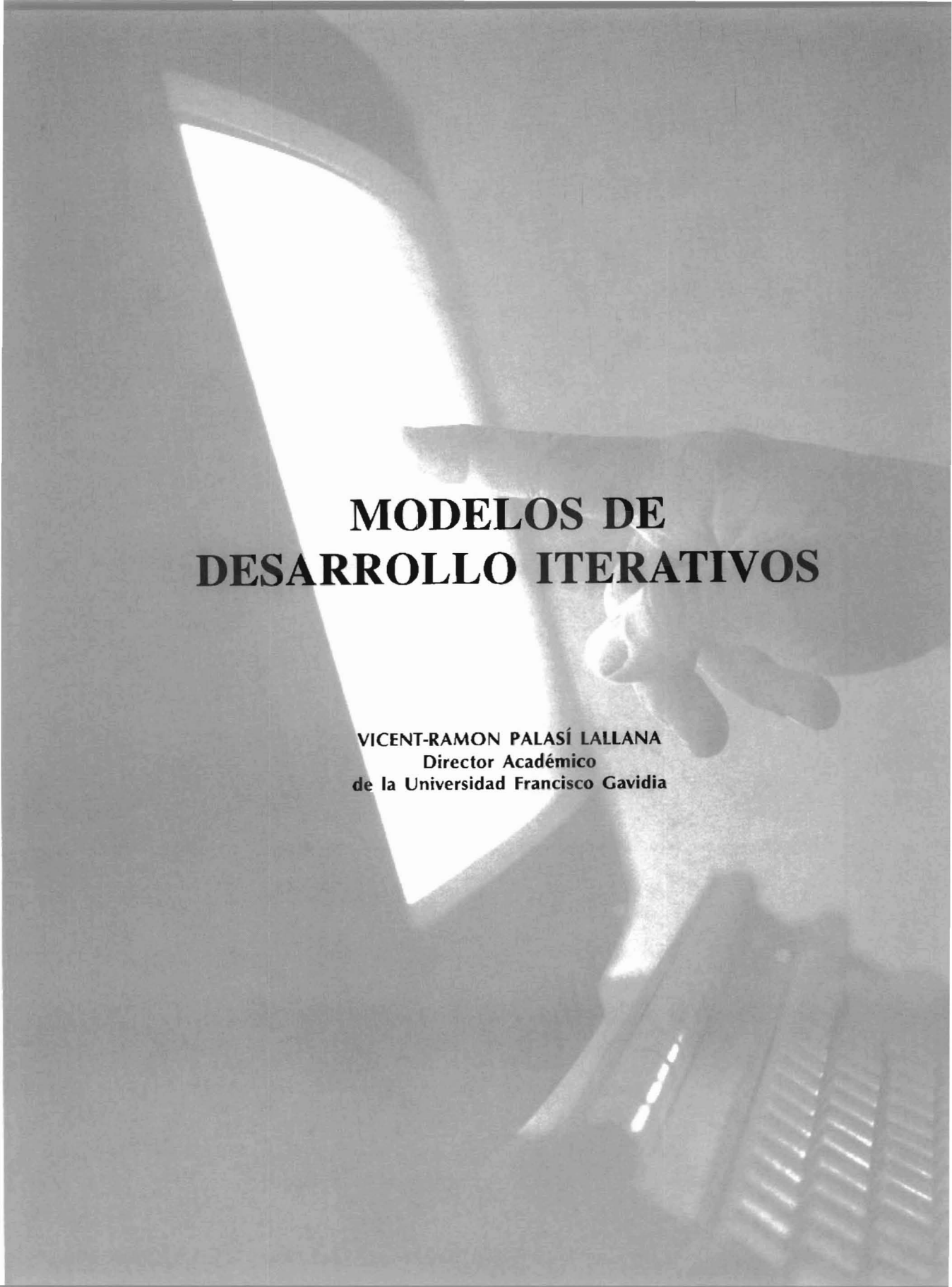


MODELOS DE DESARROLLO ITERATIVOS

VICENT-RAMON PALASÍ LALLANA
Director Académico
de la Universidad Francisco Gavidia

REALIDAD Y REFLEXIÓN
Reality and Reflection

Año 4, Nº 12, San Salvador, El Salvador, Centro América Septiembre-Diciembre 2004

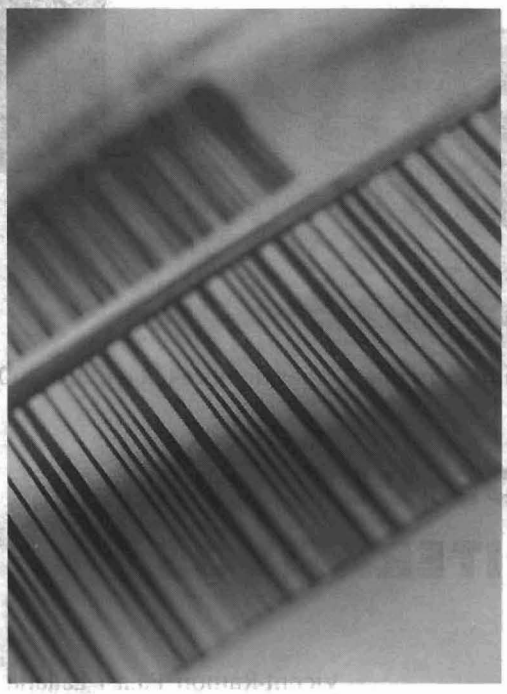
MODELOS DE DESARROLLO ITERATIVOS

Vicent-Ramon Palasí Lallana
Director Académico
de la Universidad Francisco Gavidia

The lack of development models (or the applications of useless models) for the matters of the projects systems are transformed in the lowest rates of success of the product. They should always use analysis, design and a model of an iterative development. The iterative models are examined and it is pretended to leave the desire to extend the knowledge to related aspects that are very important. SOFTWARE ENGINEERING, PROGRAMMING (ELECTRICAL COMPUTERS) DEVELOPMENT PROJECTS.

Es generalmente conocido que la tasa de éxito de los proyectos de desarrollo de software es notablemente más baja que la de cualquier otro proyecto de ingeniería. Los proyectos de software normalmente duran más de lo previsto, consumen más recursos y dinero de los presupuestado y frecuentemente producen sistemas defectuoso, con una arquitectura rígida o inestable y con numerosos errores, muchos de los cuales sólo se detectan en tiempo de explotación (para un estudio cuantitativo del tema, consultar(1).

Las causas principales de este problema son tres. Por una parte, los sistemas informáticos son mucho más complejos y abstractos que los sistemas físicos, por con-



tar con un mayor grado de libertad e interrelación que los primeros. Por otra parte, muchos proyectos informáticos no cuentan con una metodología de análisis, diseño y programación bien establecida, sino que se ejecuten de una forma empírica y desordenada. Finalmente, la gestión de proyectos informáticos muchas veces carece de un modelo de desarrollo o bien utiliza modelos obsoletos que han demostrado ser inadecuados para la tarea.

En este artículo, nos limitaremos a la tercera de estas causas. La discusión sobre modelos de desarrollo no es nueva y se ha dedicado mucha cantidad de estudio y de análisis a encontrar la mejor solución a este problema. Sin embargo, muchas de las empresas de nuestro entorno no dan la importancia que se merece a este asunto, por lo cual no es de extrañar la baja calidad y alto costo de los sistemas resultantes.

En este artículo, examinaremos tres tipos de proyectos de desarrollo de sistemas. El proyecto que no utiliza modelo de desarrollo, el proyecto que amplia un modelo en cascada y el que sigue un modelo iterativo, comparándolos y examinando sus diferentes ventajas e inconvenientes.

1. PROYECTOS SIN MODELO DE DESARROLLO

Comencemos con los proyectos que no utilizan ningún modelo de desarrollo. Aunque la necesidad de un modelo de desarrollo hace décadas que está firmemente establecida, es triste comprobar cómo, en nuestro entorno, la desidia y falta de profesionalidad así como un concepto corto de miras de la gestión empresarial hacen que la mayoría de proyectos de software no apliquen ningún modelo absoluto.

La filosofía subyacente a dichos proyectos suele ser que el análisis y diseño del sistema, así como cualquier planificación de sus desarrollo, son una pérdida de tiempo y que lo importante es comenzar a programar cuanto antes para entregar el producto lo más pronto posible. El hecho de que esta entrega a tiempo pocas veces se consiga, no impide que esta forma equivocada y poco profesional de desarrollo siga repitiéndose una y otra vez.

De la misma forma que se puede construir un edificio sin planos, también puede realizarse un sistema sin modelo de desarrollo. Pero en ambos casos el resultado dista de ser aceptable. Tanto el edificio como el sistema tardan más tiempo y cuestan más dinero en construirse que el caso en que hay una mínima planificación. El resultado en ambos casos es inestable y difícil de ampliar.

Aunque a nadie se le ocurriría construir un edificio sin planos, cantidad de empresas en nuestro entorno todavía creen que pueden desarrollar un programa (algo mucho más complejo que un edificio) sin análisis, ni diseño, ni modelo de desarrollo. Como consecuencia, se obtienen sistemas llenos de errores, difíciles de mantener, inestables y costosos, que rápidamente pasan a ser inmanejables, siendo desechados después de pocos años, con lo cual se debe construir un nuevo sistema, que invariablemente es desarrollado con la misma actitud poco profesional que el primero, presentando sus mismos defectos y repitiendo este círculo vicioso.

2. PROYECTOS CON MODELO DE DESARROLLO EN CASCADA

Si la empresa realmente aplica un modelo de desarrollo, es probable que utilice un modelo en cascada (waterfall model), ya que éste ha sido el modelo de desarrollo dominante en el área informática durante muchos años.

La figura 1 muestra un esquema simplificado de este modelo. Para los no versados en Ingeniería del Software, diremos que el análisis (más propiamente, "análisis de requerimientos") es una descripción de lo que debe hacer el sistema, el diseño son los "planos" del sistema y la programación es el proceso de construir el sistema acabado. Una vez realizada la programación, es necesario probar el programa para detectar los posibles errores y corregibles, etapa que se denomina con el nombre de "Pruebas".

Figura 1. El modelo de desarrollo en cascada

En un modelo en cascada, primero se realiza el análisis. Cuando se tiene un análisis acabado se comienza a desarrollar el diseño. Cuando el diseño está acabado, se lleva a cabo la programación. Cuando la programación está acabada se realizan las pruebas.

A primera vista, este orden parece lógico, pues el diseño se basa en el análisis, de la misma manera que la programación se basa en el diseño y las pruebas en la programación. Además, esto es análogo a lo que sucede en las otras ingenierías. Usando la analogía anterior, es evidente que no se empieza a construir un edificio hasta que los planos estén totalmente acabados.

Sin embargo, como hemos dicho, los programas informáticos son mucho más complejos y abstractos que un edificio o que cualquier producto resultante de otra ingeniería. Es por esto que este modelo no se adecua bien al desarrollo de sistemas. Sus principales defectos son los siguientes:

1. **Rigidez y poca adaptabilidad.**

   En un mundo perfecto, los clientes y los desarrollados tendrían claros los requerimientos (lo que debe hacer el sistema) desde un principio y estos requerimientos no cambiarían durante el proceso de desarrollo. Sin embargo, la realidad es que los requerimientos cambian constantemente, bien porque el cliente se da cuenta de necesidades que ignoraba, bien porque el mercado o la tecnología evolucionan o bien porque los desarrolladores se dan cuenta de requisitos técnicos que no habían previsto en un principio.

   Un estudio realizado(2) revela que los requerimientos no anticipados en el comienzo del proyecto pueden suponer un 25% del total para proyectos de desarrollo medios y hasta un 50% para proyectos grandes (similares resultados se presentan en el artículo(3)).

   El modelo en cascada no permite acomodar estos cambios, ya que los requerimientos quedan fijados desde el comienzo, no pudiendo ser modificados con posterioridad.

2. **Baja mitigación de riesgos.**

   Con el modelo en cascada no es hasta el final del proyecto cuando se pueden hacer pruebas y determinar la viabilidad o eficiencia de nuestra arquitectura. Así, los elementos más riesgosos (como la viabilidad de la arquitectura del sistema) se determinan al término del proceso de desarrollo, cuando es más difícil y costoso modificarlos, además de que se ha perdido valioso tiempo y recursos diseñando una arquitectura que al final se revela como no viable.

3. **Falta de retroalimentación.**

   Es bien conocido que, la mayoría de las veces, el cliente comienza con una idea aproximada y vaga de lo que quiere y, sólo cuando ve el programa funcionando, comienza a comprender en detalle lo que realmente necesita. Sin embargo, en el modelo en cascada, sólo se tiene un ejecutable del sistema hasta el final del proyecto. En este punto, los cambios son caros o poco posibles ya que la estructura del sistema está firmemente establecida.

Como consecuencia, el modelo de desarrollo en cascada es demasiado rígido para un proceso tan dinámico como es el desarrollo de software. Es por eso que adoptarlo es contraproducente para la correcta ejecución del proyecto de Softwarte. De hecho, en un estudio de dos años sobre proyectos de desarrollo exitosos que se publicó en (4), se determinó que el primer factor de éxito para un proyecto de desarrollo es adoptar un modelo de desarrollo diferente del modelo en cascada.

3. PROYECTOS CON MODELO DE DESARROLLO ITERATIVO

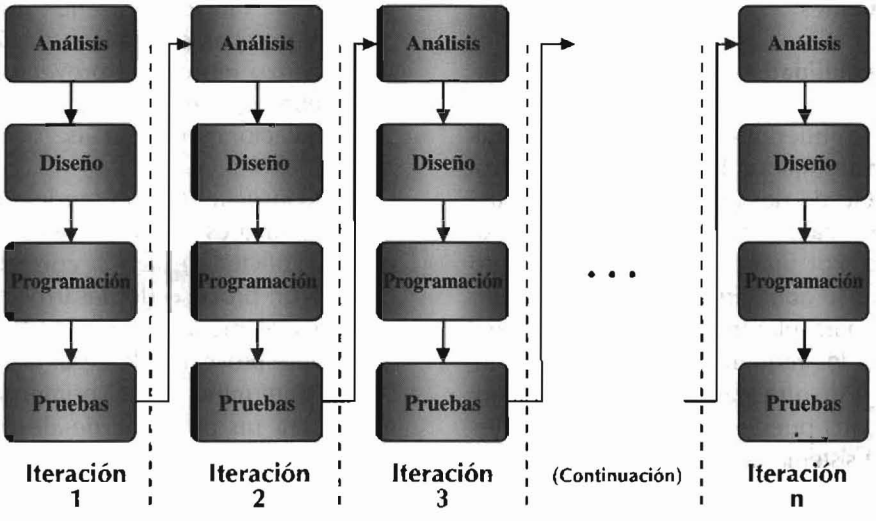
Como se puede deducir de lo dicho hasta ahora, un proyecto de desarrollo requiere un cambio constante. El modelo de desarrollo en cascada intenta evitar el cambio, fijando de forma temprana los requerimientos del sistema. En cambio, los modelos de desarrollo iterativos intentan adaptarse a este cambio, de ahí su idoneidad para el desarrollo de programas.

Hay varios modelos de desarrollo iteractivos. Entre ellos podemos destacar el "Unified Process" y su variante el "Rational Unified Process", que son estándares actuales a nivel internacional. Un modelo iterativo nuevo que ha surgido con fuerza últimamente es el "Extreme Programming (XP)". Cabe mencionar también el modelo llamado "Feature Driven Development".

Explicar alguno de estos procesos requeriría mucho más espacio del que tenemos aquí. Sin embargo, nos centraremos en las características generales y más importantes. Para mayor ampliación, el lector puede buscar en Internet o bien contactar al autor de este artículo.

Los modelos iterativos se basan en dividir el proyecto de desarrollo en varias etapas, llamadas iteraciones. Las ateraciones son cortas (unas cuantas semanas, excepto en proyectos enormes) y en educación es fija (no puede alargarse si hay retrasos, éstos se incluyen en otra iteración).

Figura 2. Un modelo de desarrollo iterativo



La idea central es que, en cada una de esa iteracciones, se construye una parte pequeña del sistema (esto se llama a veces "desarrollo incremental"). Para esa parte del sistema, se realiza todo el proceso; análisis, diseño, programación y pruebas. Se acaba la iteración con un ejecutable que incluye todas las partes del sistema construidas hasta el momento. Los aspectos del sistema con más riesgo (por ejemplo, la arquitectura) se construyen en las primeras iteraciones. El esquema de un modelo iterativo se muestra gráficamente en la figura 2.

Las ventajas de este tipo de modelo son las siguientes:

1. **Flexibilidad.**

   Los requerimientos no quedan totalmente fijados hasta el final del proyecto de desarrollo. Por ello, se pueden realizar cambios de forma flexible. Por una parte, el conocimiento que se adquiere en una iteración sirve para plantear de forma más realista los requerimientos de la siguiente. Por otra parte, este conocimiento nos puede hacer reformar partes del sistema construidas en iteraciones anteriores. En una palabra, todos los documentos del sistema (requerimientos, diseño y código) no son rígidos sino que pueden cambiarse durante todo el proceso de desarrollo. (Típicamente suelen ser modificados en mayor medida en las primeras iteraciones y en menor medida en las últimas).

2. **Mitigación de riesgos.**

   Como las pruebas se hacen desde el principio del proyecto, puede determinarse la viabilidad o eficiencia de las decisiones de diseño. Además, los elementos con más riesgo se tratan en las primeras iteraciones, con lo cual se puede implementar una mitigación de riesgos más temprana y exitosa.

3. **Retroalimentación.**

   Como hay ejecutables desde el mismo comienzo del proyecto, el cliente puede examinarlos y proponer los cambios que necesita para su negocio. También los desarrolladores tienen una rápida retroalimentación de lo que funciona y lo que no, ya que las pruebas se realizan desde el comienzo mismo del proyecto y no se debe esperar al final para hacer modificaciones necesarias.

Como consecuencia, un modelo de desarrollo iterativo es condición necesaria (aunque no suficiente) para la correcta ejecución de un proceso de desarrollo de software. En el mencionado estudio de dos años sobre proyectos de desarrollo exitosos (4), se determinó que el primer factor de éxito para un proyecto de desarrollo es adoptar un modelo iterativo, en vez de un modelo en cascada.

En este artículo sólo hemos examinado la superficie de los modelos iterativos. Un estudio más profundo revelaría una serie de aspectos de suma importancia que no han podido incluirse aquí. Sin embargo, a pesar de la limitación del espacio, se confía en haber proporcionado al lector una idea general de qué tipo de modelo de desarrollo es el más conveniente para su empresa y fomentado la inquietud de seguir informándose con más amplitud sobre ello.

Referencias bibliográficas

(1) *Jones, C. Patterns of Software Failure And Success. International Thomson Publishing, Ene. 1996.*

(2) *Jones, C. Applied Software Measurement, NY: McGraw-Hill*

(3) *Boehm, B. and Papaccio, P. 1988 Understanding and Controlling Software Costs. IEEE Transactions and Software Engineering, Oct 1998.*

(4) *MacCormak, A. Product-Development Practices That Work. MIT Sloan Management Review. Volume 42, Number 2.*