PROGRAMA DE MEJORA

Metodología de desarrollo para la Actividad productiva de la UCI.

Universidad de las Ciencias Informáticas
Carretera a San Antonio Km 2 1⁄2 . Torrens. Boyeros. La Habana. Cuba

Control del documento

Título:Metodología de desarrollo para la Actividad productiva de la UCI.

Versión: 1.2.

	Nombre	Cargo
Elaborado por:	Tamara Rodríguez Sánchez	

Aprobado por:	Firma:
Cargo:	Fecha:

Reglas de Confidencialidad

Clasificación: <USO INTERNO>

Forma de distribución: <PDF Digital>

Control de cambios

Versión	Sección, Figura, Tabla	Tipo *	Fecha	Autor del cambio	Descripción del cambio
<1.0>	Todo el documento	A	12/11/2014	Tamara Rodríguez Sánchez	Creación del documento.
<1.1>	Sección	M	21/11/2014	Tamara Rodríguez Sánchez	Actualización de los roles de la metodología.
<1.2>	Sección	M	06/03/2015	Tamara Rodríguez Sánchez	Actualización de las disciplinas de la metodología.

* Sección del documento, Tabla, Figura.
** **A** Alta; **B** Baja; **M** Modificación

Índice de contenidos.

1 Introducción

1.1 Objetivo

El objetivo del presente documento es describir la Metodología de desarrollo a emplearse en los proyectos productivos de la Universidad de las Ciencias Informáticas (UCI). Dicha definición se basa en una variación de la metodología ¨Proceso Unificado Ágil¨ (AUP por sus siglas en inglés) en unión con el modelo CMMI-DEV v 1.3.

1.2 Alcance

Lo establecido por este documento es aplicable a todos los proyectos de desarrollo de software que se desarrollen en la Universidad.

1.3 Definiciones y Acrónimos

CMMI: Capability Maturity Model Integration.

PMC: Monitoreo y control de proyecto.

PP: Planeación de proyecto.

CM: Administración de la configuración.

1.4 Referencias

2 Antecedentes de la Actividad productiva de la UCI.

La UCI cuenta con 14 centros productivos en su gran mayoría pertenecientes a las facultades que conforman a la Universidad. Cada uno de estos centros se dedica al desarrollo de software y/o servicios asociados a un dominio de aplicación bien definido. Todos los desarrollos se encuentran organizados en proyectos, clasificados en: Gestión, Componentes y tecnología base, Portales y Sistemas operativos. Esta diversidad de centros y proyectos hace que la actividad productiva en la UCI sea cada vez más amplia, y trae consigo la heterogeneidad en el proceso de desarrollo de software.

Actualmente el desarrollo de software dentro de la actividad productiva de la UCI se caracteriza por el uso de diferentes metodologías de desarrollo entre robustas y ágiles, especificamente las nueve metodologías que se listan a continuación:

- XP
- OPEN UP
- RUP
- BPM
- DAC
- KIMBALL
- SXP
- SCRUM
- NOVA OPEN UP

A pesar de la variedad de metodologías usadas, se ha comprobado que muy pocos proyectos la aplican en su totalidad. Las diferencias entre estas metodologías no radica únicamente en los productos de trabajos que proponen o en sus roles, sino en su forma de planificar el proyecto y realizar las estimaciones del tiempo. Factor determinante en la culminación exitosa de todo desarrollo de software, por lo que uno de los principales problemas detectados es que sin importar la metodología que se usa se está planificando con un único cronograma tipo, además de forzar el método de estimación definido en la Universidad y que responde en su gran mayoría a la metodología RUP·

Para lograr erradicar los problemas detectados, se propone crear un cronograma tipo y un método de estimación para cada una de las metodologías que hoy se usan en los proyectos o converger a una única metodología que cubra las particularidades de cada uno. El esfuerzo en tiempo y en personas de la primera variante es mucho mayor que la posibilidad de converger a los proyectos hacia una sola metodología de desarrollo. Por lo tanto se decide escoger una metodología para ser adaptada a lo que ya la Universidad ha estado proponiendo como ciclo de vida de los proyectos, sin alejarse de lo que hasta el momento se ha trabajado e introducir la menor cantidad de cambios posibles. La propuesta que se trae a continuación responde a una

variación que se le realiza a el Proceso Unificado Ágil.

3 Marco conceptual

El Proceso Unificado Ágil de Scott Ambler o Agile Unified Process (AUP) en inglés es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo:

- Desarrollo Dirigido por Pruebas (test driven development - TDD en inglés)
- Modelado ágil
- Gestión de Cambios ágil
- Refactorización de Base de Datos para mejorar la productividad.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva.

3.1 Fases AUP

1. Inicio: El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
2. Elaboración: El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
3. Construcción: Durante la fase de construcción el sistema es desarrollado y probado al completo en el ambiente de desarrollo.
4. Transición: El sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción.

3.2 Disciplinas AUP

AUP define 7 disciplinas (4 ingenieriles y 3 de gestión de proyectos), las disciplinas son:

1. Modelo. El objetivo de esta disciplina es entender el negocio de la organización, el problema de dominio que se abordan en el proyecto, y determinar una solución viable para resolver el problema de dominio. Agrupa los flujos de trabajos de Modelado de negocio, Requisitos y Análisis y Diseño.

2. Implementación. El objetivo de esta disciplina es transformar su modelo (s) en código ejecutable y realizar un nivel básico de las pruebas, en particular, la unidad de pruebas.

3. Prueba. El objetivo de esta disciplina consiste en realizar una evaluación objetiva para garantizar la calidad. Esto incluye la búsqueda de defectos, validar que el sistema funcionatal como está establecido, y verificando que se cumplan los requisitos.

4. Despliegue. El objetivo de esta disciplina es la prestación y ejecución del sistema y que el mismo este a disposición de los usuarios finales.

5. Gestión de configuración. El objetivo de esta disciplina es la gestión de acceso a herramientas de su proyecto. Esto incluye no sólo el seguimiento de las versiones con el tiempo, sino también el control y gestión del cambio para ellos.
6. Gestión de proyectos. El objetivo de esta disciplina es dirigir las actividades que se lleva acabo en el proyecto. Esto incluye la gestión de riesgos, la dirección de personas (la asignación de tareas, el seguimiento de los progresos, etc), coordinación con el personal y los sistemas fuera del alcance del proyecto para asegurarse de que es entregado a tiempo y dentro del presupuesto.
7. Entorno. El objetivo de esta disciplina es apoyar el resto de los esfuerzos por garantizar que el proceso sea el adecuado, la orientación (normas y directrices), y herramientas (hardware, software, etc) estén disponibles para el equipo según sea necesario.

3.3 Roles definidos por AUP

1. Administrador de proyecto: Maneja a los miembros construye relaciones con los stakeholders, coordina interacciones con los stakeholders, planea, maneja y asigna los recursos.
2. Ingeniero de procesos: Desarrolla, adapta y apoya sus materiales del proceso del software.
3. Desarrollador: Escribe, testea y construye software.
4. Administrador de Base de Datos: Diseña, prueba, desarrolla, y apoya los esquemas de la BD.
5. Modelador ágil: Crea y desarrolla modelos, bosquejos o los archivos de la herramienta CASE, de una manera evolutiva y de colaboración.
6. Administrador de la configuración: Un encargado de la configuración es responsable de proporcionar la infraestructura total y el ambiente del CM al equipo de desarrollo.
7. Stakeholder
8. Administrador de pruebas: Responsables del éxito de la prueba, incluyendo el planeamiento, la gerencia, y la defensa para la prueba y las actividades de la calidad.
9. Probador: Encargado de ejecutar las pruebas.

4 Variación de AUP para la UCI.

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI.
Una metodología de desarrollo de software tiene entre sus objetivos aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas, para ello nos apoyaremos en el Modelo CMMI-DEV v1.3. El cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas se centran en el desarrollo de productos y servicios de calidad.

4.1 Descripción de las Fases

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre. Para una mayor comprensión se muestra la siguiente Tabla 1.

Fases AUP	**Fases Variación AUP-UCI**	**Objetivos de las fases (Variación AUP-UCI)**
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración Construcción Transición	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la transición se capacita a los usuarios finales sobre la utilización del software.
	Cierre	En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

4.2 Descripción de las disciplinas

AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno), se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas también, pero a un nivel más atómico que el definido en AUP. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas. Se mantiene la disciplina Implementación, en el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación. Las restantes 3 disciplinas de AUP asociadas a la parte de gestión para la variación UCI se cubren con las las áreas de procesos que define CMMI-DEV v1.3 para el nivel 2, serían CM (Gestión de la configuración), PP (Planeación de proyecto) y PMC (Monitoreo y control de proyecto). Para una mayor comprensión se muestra la siguiente Tabla 2.

Disciplinas AUP	**Disciplinas Variación AUP-UCI**	**Objetivos Disciplinas (Variación AUP-UCI)**
Modelo	Modelado de negocio	El Modelado del Negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito. Para modelar el negocio se proponen las siguientes variantes : **1- Casos de Uso del Negocio (CUN).** **2- Descripción de Proceso de Negocio (DPN).** **3- Modelo Conceptual (MC).** A partir de las variantes anteriores se condicionan cuatro escenarios para modelar el sistema en la disciplina **Requisitos**.

	Requisitos	El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CU), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el Modelado de negocio. ***Ver 4.2.1 Descripción de los escenarios para la disciplina Requisitos.***
	Análisis y diseño	En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son

		más formales y específicos que el de análisis.
Implementación	Implementación	En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.
Prueba	Pruebas interna	En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible componentes de prueba ejecutables para automatizar las pruebas.
	Pruebas de liberación	Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.
	Pruebas de Aceptación	Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

Gestión de configuración	Se cubren con las áreas de procesos PP, PMC y CM que propone CMMI-DEV v1.3. Las mismas son áreas de procesos de gestión y soporte respectivamente.	Consultar en *mejoras.prod.uci.cu* los libros de procesos de cada una de estas áreas.
Gestión de proyecto		
Entorno		

4.2.1 Escenarios para la disciplina Requisitos.

A partir de que el Modelado de negocio propone tres variantes a utilizar en los proyectos (CUN, DPN o MC) y existen tres formas de encapsular los requisitos (CUS, HU, DRP), surgen cuatro escenarios para modelar el sistema en los proyectos, manteniendo en dos de ellos el MC, quedando de la siguiente forma:

Escenario No 1:

Proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.

Escenario No 2:

Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.

Escenario No 3:

Proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP.

Escenario No 4:

Proyectos que no modelan negocio solo pueden modelar el sistema con HU.

4.2.2 Características por escenarios.

A partir de los escenarios antes expuestos, cada proyecto debe insertarse en uno de ellos en concordancia con las particularidades que caracterizan a cada uno de ellos:

Escenario No 1: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que puedan modelar una serie de interacciones entre los trabajadores del negocio/actores del sistema (usuario), similar a una llamada y respuesta respectivamente, donde la atención se centra en cómo el usuario va a utilizar el sistema. Es necesario que se tenga claro por el proyecto que los CUN muestran como los procesos son llevados a cabo por personas y los activos de la organización.

Escenario No 2: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

Escenario No 3: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan

Escenario No 4: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido donde el cliente estará siempre acompañando al equipo de desarrollo. Para elaborar HU el proyecto debe establecer conversaciones acerca de las necesidades de los clientes. Se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información.

Todas las disciplinas antes definidas (desde Modelado de negocio hasta Pruebas de Aceptación) se desarrollan en la Fase de Ejecución, de ahí que en la misma se realicen iteraciones y se obtengan resultados incrementales, ver Figura 1.

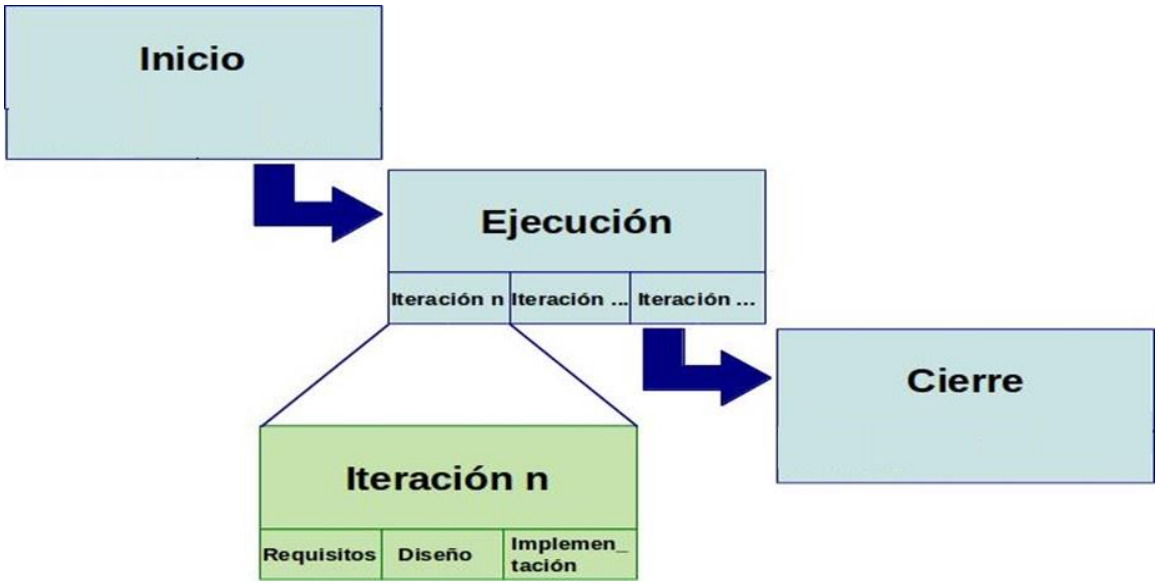

Figura 1. Fases e Iteraciones.

4.3 Descripción de los roles

AUP propone 9 roles (Administrador de proyecto, Ingeniero de procesos, Desarrollador, Administrador de BD, Modelador ágil, Administrador de la configuración, Stakeholder, Administrador de pruebas, Probador), se decide para el ciclo de vida de los proyectos de la UCI tener 11 roles, manteniendo algunos de los propuestos por AUP y unificando o agregando otros. Para una mayor comprensión se muestra la siguiente Tabla 3.

Roles AUP	**Roles Variación AUP-UCI**	**Responsabilidades Roles (Variación AUP-UCI)**
Administrador de proyecto	Jefe de proyecto	Las habilidades y competencias de cada uno de los roles definidos para la Variación de AUP-UCI se pueden consultar en *mejoras.prod.uci.cu*
	Planificador	
Ingeniero de procesos	Analista	
Modelador ágil	Arquitecto de información (Opcional)	
Desarrollador	Desarrollador	
Administrador de la configuración	Administrador de la configuración	
Stakeholder	Stakeholder (Cliente/Proveedor de requisitos)	
Administrador de pruebas	Administrador de calidad	

Probador	Probador	
Administrador de BD	Arquitecto de software (Sistema)	
	Administrador de BD	

4.4 Productos de trabajos

Entre las técnicas ágiles que utiliza AUP se encuentra el Modelado ágil, se hará uso de está técnica para los proyectos que necesiten por sus características encapsular sus requisitos funcionales en **Historias de usuarios** o en **Descripción de requisitos por procesos**. La otra forma de encapsular los requisitos se mantiene por **Casos de Uso (CU)**. De forma general se siguen utilizando los productos de trabajos definidos en el Expediente de proyecto, algunos son obligatorios independientemente del tipo de proyecto y otros son opcionales en base a las particularidades del mismo. Todos los productos de trabajos se encuentran en los documentos públicos ubicados en *excriba.prod.uci.cu* y en *mejoras.prod.uci.cu*.

5 Conclusiones

Con la adaptación de AUP que se propone para la actividad productiva de la UCI se logra estándarizar el proceso de desarrollo de software, dando cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3. Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos. Se redujo a 1 la cantidad de metodologías que se usaban y de más de 20 roles en total que se definían se redujeron a 11.

	Antes	Ahora
Metodologías	-XP -OPEN UP -RUP -BPM -DAC -KIMBALL -SXP -SCRUM -NOVA OPEN UP	Variación AUP-UCI
	-Jefe de proyecto -Planificador -Analista	-Jefe de proyecto -Planificador -Analista

Roles	-Arquitecto de información -Desarrollador -Administrador de la configuración -Administrador de calidad -Probador -Arquitecto de software -Administrador de BD -Programador -Diseñador de pruebas -Diseñador de hardware -Implantador de soluciones -Arquitecto de datos -Analista de componente -Programador de componente -Programador de arquitectura -Programador de la capa de visualización de información -Administrador de datawarehouse -Diseñador gráfico	-Arquitecto de información (Opcional) -Desarrollador -Administrador de la configuración -Stakeholder (Cliente/Proveedor de requisitos) -Administrador de calidad -Probador -Arquitecto de software (Sistema) -Administrador de BD